ABSTRACT

BARNES, TIFFANY MICHELLE. The Q-matrix Method of Fault-Tolerant Teaching in Knowledge Assessment and Data Mining. (Under the direction of Donald L. Bitzer).

A major challenge for today's schools is to create individualized instruction in inexpensive, expandable ways. To achieve this goal, educational systems must adapt to each student, mining student data to determine what a student knows and does not know. Using this knowledge, teaching systems can guide students in the learning process.

This dissertation investigates the q-matrix method of fault tolerant teaching (FTT). FTT systems are adaptive teaching systems that tolerate student, teacher, and system errors in diagnosing student misconceptions. The q-matrix method is easily applied to any tutorial system data, independent of topic, and can be used to model student behavior and guide student knowledge remediation. We found that, in addition, the model is easily interpretable and can be used to understand large sets of student data, pinpointing problem areas for student learning.

In this work, we applied the q-matrix method in three computer tutorials, presenting the first experiment to use the method on a large group of students. Overall, students felt that these tutorials were beneficial. For each tutorial, the q-matrix method was used to create a student knowledge model. When compared with other data mining/knowledge discovery methods, including factor analysis and cluster analysis, the q-matrix method was superior in that it was able to fit the observed data well, while still offering the interpretability needed to devise remediation methods. Our results indicate that the q-matrix model may predict student misconceptions at least as well as students were able to predict themselves, and students generally felt that the tutorial knew which concepts each student least understood.

For our logic proofs tutorial, we devised q-matrices as data mining tools, used to extract the axioms needed to solve a proof. In this analysis, we were able to isolate sets of students using similar strategies, and to interpret the strategies of these groups using the q-matrix model.

We also compared extracted q-matrix models to expert models, and found that the extracted and expert q-matrices were not a good match, but that extracted q-matrix models were quite useful in understanding student data. This shows that expert models do not necessarily predict student behavior and more accurate student knowledge models, such as q-matrices, are needed to understand student knowledge. An extracted q-matrix can reveal student behavior that might not be predicted by an expert's understanding.

This work resulted in the construction of a fully automated, fault tolerant, intelligent tutoring system, which can diagnose and correct student misconceptions. This system also provides an interpretable model for each topic that relates each tutorial question to its underlying concepts. The experimental analysis provides valuable insight into the factors that influence the extraction and interpretability of these models, as well as their value in automatically assessing student knowledge. In addition, the q-matrix method is used as a general data mining tool in one tutorial where a traditional application of the q-matrix method would not be appropriate. This application and its favorable comparison with other data mining tools mark the q-matrix method as a viable data clustering and interpretation tool for data mining and knowledge discovery.

THE Q-MATRIX METHOD OF FAULT-TOLERANT TEACHING IN KNOWLEDGE ASSESSMENT AND DATA MINING

by
TIFFANY MICHELLE BARNES

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

COMPUTER SCIENCE

Raleigh

2003

APPROVED BY:

_____________________________ _____________________________

_____________________________ _____________________________

____
Chair of Advisory Committee

Dedicated to Okan Pala,
to my parents Pamela Rathbun and Danny Barnes,
to my brother Andrew Barnes,
and to my sister and her family
Crystal, Doug, Heather, Steven, and Jayce Jones.

Biography

Tiffany Barnes, a native North Carolinian, received the Bachelor of Science degrees in Computer Science and Mathematics (1995), Masters in Computer Science and Mathematics (2000), and Doctor of Philosophy in Computer Science (2003) from NC State University. Her work on the q-matrix method was presented at the ACM Southeastern Conference in Raleigh, NC (2002) and the E-Learn Conference in Montreal, Canada (2002). She has published other papers in the *Electronic Journal of Combinatorics* (1995) and *Discrete Applied Mathematics* (1997), *Meridian* magazine at NCSU (2000) and presented papers at numerous conferences.

Dr. Barnes is interested in research in effective teaching methods for both on-campus and distance education, and theoretical computer science including algorithms, computations theory, artificial intelligence, discrete mathematics, combinatorics, and coding theory. She also specializes in web and information design and evaluation.

At NCSU, Dr. Barnes was a CSC Graduate Student Association President, Hewlett Initiative Graduate Fellow, Preparing the Professoriate award winner, Outstanding Teaching Assistant award winner, Women in Mathematics Mentor, Girls on Track Summer Program Technical Director, and Ballroom Dance Club President and Advisor. She learned to scuba dive, mountain bike, and to speak Spanish and Hungarian. She spent a year studying mathematics in Budapest, where she traveled and met amazing mathematicians including Paul Erdos. Dr. Barnes plans to continue her career in research and teaching as a professor of Computer Science.

Acknowledgements

This dissertation would not have been possible without the help, support, and guidance, of many wonderful people, especially: Okan Pala; my brother Andrew Barnes; my family; my committee members Donald Bitzer, Mladen Vouk, R. Michael Young, Rob St. Amant, and Brad Mehlenbacher; my teaching assistants Jeff Ligon, Evan Ernst, Lynn Albers, and Matt Kendall; my friends Denise Lancaster and Zeynep Savas; and my dance team Dancing with Wolves.

This work was partially supported by NSF grants #9813902 and #0204222.

Contents

List of Tables vii

List of Figures ix

1 Introduction 1

1.1 Data mining, teaching, and assessment 3
1.2 Adaptive tutorial systems and student assessment 6
1.3 Research challenge 8
1.4 Research objectives & questions 9
1.5 Research hypotheses 11
1.6 Dissertation Outline 12

2 Theoretical background and related work in FTT 15

2.1 Procedural models of student knowledge 15
2.1.1 Buggy, repair theory, and Sierra theory 17
2.1.2 OLAE and POLA 19
2.1.3 Rule space theory and Q-matrices 20
2.2 Prediction of student data and changing Q-matrices 22
2.3 Q-matrix extraction methods and fault tolerance 25
2.3.1 Extraction methods 26
2.3.2 Factor analysis 26
2.3.3 The q-matrix method 29
2.4 Question generation and q-matrix extraction with actual students 31
2.5 Data mining and q-matrix methods 33
2.6 Summary of related works 34

3 Methodology 36

3.1 Adaptive tutorial design 38
3.2 Testing our research hypotheses 40

4 Student Model Extraction 42

4.1 Q-matrix method 42
4.2 Factor Analysis 43
4.3 Cluster Analysis 48
4.3.1 Comparison of q-matrix to cluster analysis error 49
4.3.2 Graphical comparison of q-matrix concept states and clusters 53
4.3.3 Q-matrix concept states as one step of a cluster analysis 58
4.4 Discriminant analysis 60
4.5 Summary 61

5 Knowledge Assessment 63

5.1 Comparison of Binary Relations tutorial scores with student grades 63
5.2 Comparison of model with expert analysis 69

5.2.1 Expert v. extracted q-matrices for the binary relations tutorial section 1 69
5.2.2 Expert v. extracted q-matrices for the binary relations tutorial section 2 70
5.2.3 Expert v. extracted q-matrices for the binary relations tutorial section 3 75
5.2.4 Summary of comparison of expert and extracted q-matrices 78
5.3 Comparison of predicted direction vs. student for remediation 78
5.4 Summary of Knowledge Assessment 80

6 Survey Results 82

6.1 Tutorial Survey Results, Questions 1-10 82
6.2 Binary Relations Tutorial Free Response Questions 88
6.3 Counting Tutorial Survey Free Response Questions 91
6.4 Proofs Tutorial Survey Free Response Questions 93
6.5 Summary of Tutorial Results 96

7 Proofs Analysis & Data Mining 98

7.1 Method 100
7.2 Proof models & discussion 103
7.2.1 Proof 1 Discussion 104
7.2.2 Proof 2 Discussion 111
7.2.3 Proof 3 Discussion 115
7.2.4 Proof 4 Discussion 120
7.2.5 Proof 5 Discussion 125
7.2.6 Proof 6 Discussion 129
7.2.7 Proof 7 Discussion 132
7.2.8 Proof 8 Discussion 136
7.2.9 Proof 9 Discussion 140
7.2.10 Proof 10 Discussion 144
7.2.11 Q-matrix method as a data mining tool 148

8 Summary 152

8.1 Chapter Overviews 152
8.2 Conclusions 156
8.3 Future Work 161

References 164

Appendix A: Glossary of Terms 171

Appendix B: Author's past research contributions 172

Appendix C: Q-matrix Pseudo-code 173

Appendix D: Axiom List 174

List of Tables

Table 2-1 Binary q-matrix example 21
Table 2-2: Example binary q-matrix 30
Table 2-3: Expert versus extracted q-matrix........ 32
Table 4-1 Number of clusters generated by 2, 3, & 4 concept q-matrices 50
Table 4-2 Convergence of q-matrix "clusters" using cluster criterion........ 59
Table 5-1 Binary Relations Tutorial Section 1 Comparison with Student Exam Grades . 64
Table 5-2 Binary Relations Tutorial Section 2 Comparison with Student Exam Grades . 65
Table 5-3 Binary Relations Tutorial Section 3 Comparison with Student Exam Grades . 65
Table 5-4 Correlation of Binary Relations Tutorial Section 1 Scores with Final Exam Scores........ 68
Table 5-5 Correlation of Binary Relations Tutorial Section 2 Scores with Final Exam Scores........ 68
Table 5-6 Binary Relations Tutorial Section 3 Comparison with Student Exam Grades . 68
Table 5-7 Topics in Binary Relations Tutorial, Section 1 69
Table 5-8 Binary Relations Section 1 Extracted q-matrix........ 70
Table 5-9 Binary Relations Section 1 Expert q-matrix 70
Table 5-10 Topics in Binary Relations Tutorial Section 2 71
Table 5-11 Binary Relations Section 2 Sellers' Expert v. Extracted Q-matrices 71
Table 5-12 Binary Relations Section 2 Fall 2002 Extracted q-matrix, 4 concepts, Err/stud: 0.74 73
Table 5-13 Binary Relations Section 2 Fall 2002 Extracted q-matrix, 3 concepts, Err/stud: 1.11 74
Table 5-14 Binary Relations Section 2 Fall 2002 Extracted q-matrix, 2 concepts, Err/stud: 1.27 74
Table 5-15 Topics in Binary Relations Tutorial Section 3 75
Table 5-16 Binary Relations Section 3 Expert q-matrix 76
Table 5-17 Binary Relations Section 3 Fall 2002 Extracted q-matrix, 4 concepts, Err/stud: 0.72 77
Table 5-18 Student v. q-matrix Selection of Least-Understood Question 79
Table 5-19 Final Exam Performance for Self-Directed Students 80
Table 6-1 Tutorial Survey Questions 83
Table 6-2 Tutorial Survey Results for Fall 2002 Data, by Percentage of the 283 Respondents 84
Table 7-1 Proofs Tutorial Section 2 Proofs 104
Table 7-2 Proof 1 q-matrix 105
Table 7-3 Proof 1 Example of Direct Proof........ 107
Table 7-4 Proof 1 Ideal and Student Response Vectors 110
Table 7-5 Proof 1 Example of Proof by Contradiction 110
Table 7-6 Proof 2 q-matrix 111
Table 7-7 Proof 2 Example of Direct Proof........ 112
Table 7-8 Proof 2 Example of Proof by Contradiction 113
Table 7-9 Proof 2 Ideal and Student Response Vectors 114

Table 7-10 Proof 3 1-concept q-matrix 115
Table 7-11 Proof 3 Ideal and Student Response Vectors for One Concept 116
Table 7-12 Proof 3 Example of Direct Proof.. 116
Table 7-13 Proof 3 Example of Proof by Contradiction 117
Table 7-14 Proof 3 2-concept q-matrix 117
Table 7-15 Proof 3 Ideal and Student Response Vectors for Two Concepts 119
Table 7-16 Proof 4 q-matrix 121
Table 7-17 Proof 4 Ideal and Student Response Vectors 123
Table 7-18 Proof 4 Example of Proof by Contradiction 124
Table 7-19 Proof 4 Example of Direct Proof.. 124
Table 7-20 Proof 5 q-matrix 125
Table 7-21 Proof 5 Ideal and Student Response Vectors 126
Table 7-22 Proof 5 Example of Proof by Contradiction 128
Table 7-23 Proof 5 Example of Direct Proof.. 128
Table 7-24 Proof 6 q-matrix 130
Table 7-25 Proof 6 Ideal and Student Response Vectors 130
Table 7-26 Proof 6 Example of Direct Proof.. 131
Table 7-27 Proof 6 Example of Proof by Contradiction 131
Table 7-28 Proof 7 q-matrix 132
Table 7-29 Proof 7 Ideal and Student Response Vectors 134
Table 7-30 Proof 7 Example of Direct Proof.. 134
Table 7-31 Proof 7 Example of Direct Proof (State 110)... 135
Table 7-32 Proof 7 Example of Proof by Contradiction 135
Table 7-33 Proof 8 q-matrix 137
Table 7-34 Proof 8 Ideal and Student Response Vectors 138
Table 7-35 Proof 8 Example of Direct Proof.. 139
Table 7-36 Proof 8 Example of Proof by Contradiction 139
Table 7-37 Proof 9 q-matrix 141
Table 7-38 Proof 9 Ideal and Student Response Vectors 142
Table 7-39 Proof 9 Example of Direct Proof.. 143
Table 7-40 Proof 9 Example of Proof by Contradiction 144
Table 7-41 Proof 10 q-matrix 144
Table 7-42 Proof 10 Ideal and Student Response Vectors 146
Table 7-43 Proof 10 Example of Direct Proof.. 147
Table 7-44 Proof 10 Example of Proof by Contradiction 147

List of Figures

Figure 3-1 Adaptive tutorial system flow 39
Figure 4-1 Example q-matrix 43
Figure 4-2 Factor analysis and q-matrix errors per student 46
Figure 4-3 Ratio of q-matrix to factor error and relative # of distinct observations 46
Figure 4-4 Ratio of cluster over q-matrix error for 2,3, & 4 concept q-matrices 51
Figure 4-5 Average ratio of cluster/q-matrix error v. relative number of distinct observations 52
Figure 4-6 Binary sec. 1 clusters 54
Figure 4-7 Binary sec. 2 clusters 54
Figure 4-8 Binary sec. 3 clusters 55
Figure 4-9 Count clusters 55
Figure 4-10 Proof 1 clusters 56
Figure 4-11 Proof 2 clusters 56
Figure 4-12 Proof 3 clusters 56
Figure 4-13 Proof 4 clusters 56
Figure 4-14 Proof 5 clusters 56
Figure 4-15 Proof 6 clusters 56
Figure 4-16 Proof 7 clusters 57
Figure 4-17 Proof 8 clusters 57
Figure 4-18 Proof 9 clusters 57
Figure 4-19 Proof 10 clusters 57
Figure 4-20 Ratio of different/total cluster assignments 57
Figure 6-1 Question 1 confidence responses 85

1 Introduction

As the world becomes more interconnected, information can be shared between people as never before. While this opens up exciting possibilities for communication and research, it also results in an overwhelming amount of information that each person must process every day in order to function. With each new person joining the ‘global village,’ the need to develop automated methods to filter, sort, search, and share information in meaningful ways grows exponentially. The popularity of effective web search engines, such as Yahoo [Yah03], Google [Goo03], and Alta-Vista [Alt03] demonstrates only a small part of the need for ‘data mining’ in society today.

Our schools are facing an even greater need for the benefits that data mining promises to deliver. Both teachers and students are faced with teaching and learning an ever-growing number of ideas – all in the same school day as before. Ideally, as more issues are addressed, schools would add more teachers and classes to cover new topics. Instead, schools face teacher shortages, inadequate teacher training, and growing classroom sizes.

At the same time, the typical demographic of universities is changing – from young on-campus students to diverse professionals of all ages taking classes in the evenings, online, or even at a distance. Colleges and universities are now challenged to deliver quality service to students of widely ranging abilities and backgrounds across all distances. To rise to this challenge, information technology assisted education must pair individuals with expert instruction that is interactive, adaptive, and accessible. The promise of information technology assisted education is great, but research into methods for the delivery of education with information technology is still a rich, open field.

We believe that one new area of research, termed data mining and knowledge discovery, can be applied to educational problems to achieve our goal of individualized instruction, at hopefully much lower costs than more traditional adaptive teaching systems that use knowledge-based models to understand and direct student learning. Many excellent intelligent tutoring systems exist today [Dav03, Kav02, Bur88, Cla86, Fra96, Ger00, Hum95, Kav00, Kav02, Mis96, Mor84, Sel98, Ste91, Tye96, VN01] However, the majority of these intelligent tutoring systems require the construction of complex models that are applicable only to a specific tutorial in a specific field, necessitating a large number of experts to create and then test these models on students. Data mining and knowledge discovery, on the other hand, might be applied to the problem of understanding student knowledge, and using this understanding to direct knowledge remediation. In this dissertation we show that the q-matrix method of fault tolerant teaching is one such application of a data mining approach to understanding and directing student learning.

In this Introduction, we continue to discuss the motivating factors behind this research. In Section 1.1, we discuss data mining as a specific method for helping assess student knowledge automatically, and improve teaching in a way that tolerates both student and teacher errors to the benefit of both. Section 1.2 expands this discussion to show how important assessment is to adaptive tutorials, which are important aspects of effective education systems. The following Section 1.3 gives an overview of the challenges researchers face in developing online educational systems. Sections 1.4 and 1.5 discuss the main research objectives, questions, and hypotheses addressed in this dissertation. The final section of this chapter, Section 1.6, outlines the format of this dissertation.

1.1 Data mining, teaching, and assessment

Data mining is the process of finding hidden characteristics or common concepts in a large amount of data. For example, a researcher may wish to find all the documents related to computer-based education in universities. However, many documents may contain each of the words: computer, based, education, and university, but still not be related to using computers to teach at the university level. In many other cases, relevant documents may use another name for computer-based education, such as computer-assisted instruction, computer-based training, web-based education, online instruction, etc. For the most part, researchers must create their own associations to discover related items, and filter through many irrelevant items while searching.

Data mining is often used in what is called knowledge discovery, to discover common attributes in data as a person might understand them. Based on this idea, we might be able to apply data mining techniques to understand student knowledge: in other words, to determine what a student does and does not know during the course of instruction. In the classroom, teachers perform this type of 'knowledge discovery and data mining' every day, using information from classroom behavior, assignments, and tests to assess each student's level of knowledge and expertise. From this assessment, teachers decide how best to adapt their teaching to fill in the gaps in student knowledge. It is a challenge for teachers to adapt their classroom style to accommodate each student – but it is much easier for a personal tutor to do so for each student he or she teaches.

Not only can a personal tutor assess student knowledge more rapidly and effectively than a classroom teacher, but a tutor can also develop ways to best instruct that particular student, based on their past experiences together. For both teachers and private tutors, a great deal of real intelligence and intuition are usually required to achieve the greatest teaching results. Students are not always consistent, committing errors on one question and not on the next, understanding one topic quickly and another only after repeated tries, interpreting examples in unique ways, attacking problems in completely unexpected ways, guessing correct answers without knowing how to achieve them, and making mistakes on items they clearly understand. Teachers can also be a source of added complexity and confusion, by miswording problems, asking questions that require more knowledge than has been taught, not asking enough questions for each separate topic, etc. These issues confound the task of determining what a student knows – to the point of seeming impossible!

Human get things wrong for the right reasons, and get things right for the wrong reasons. The complex task of assessment requires admittance of such errors while still robustly measuring student knowledge. In real classrooms, teachers combine tests, observance of classroom behavior, and assignments to understand of student knowledge. An entire field of research is devoted to the issues of student knowledge assessment [see AA96, Air91, Ash98, Die91, Cha87, Ebe91, Fla85, Lin87, Wis98].

Two commonly used tools for student modeling include overlay models and Bayesian networks. An overlay model assumes that student knowledge is a subset of expert knowledge. In the fields of science and mathematics, procedural knowledge models are often constructed to represent knowledge, and student behavior is understood by comparing their actions to the model. As these systems were implemented, researchers discovered that student beliefs do not always coincide with expert knowledge. In Chapter 2, we discuss how several prominent procedural knowledge models were adapted to explain "buggy" behavior of students [Bro78, Bro80 and others]. Another type of student knowledge model is a Bayesian network that probabilistically reasons about a student's knowledge state based on their behavior [Mar95,VM98]. These and other systems including neural networks have also been used to model student behavior for use in intelligent tutoring systems [Hum95]. In general, most of these intelligent tutoring systems included models that were specific to a particular domain or application and are not re-usable [Tye96]. Each of these models requires significant investment of time and effort to build, making it very desirable to build knowledge models and diagnostic tools that can be applied independently of the knowledge domain.

Though assessment is a difficult task, many teachers are able to assess what their students understand, and use that knowledge to guide the teaching process. This requires a human with real intelligence - our goal is to use a computer to perform these key elements of data mining with artificial intelligence. We hope to build fault tolerant teaching systems that capture that ability: to assess student knowledge in a way that tolerates performance errors, and effectively guides the teaching process. We also wish to make this ability available to any and all students, in any topic, across any distance.

To that end, this dissertation explores the promising q-matrix method of fault tolerant teaching (FTT), which will provide distance education tools with the ability to adapt tutorials for any subject to all types of students, and whose application can also be used to measure and tailor the effectiveness of varied teaching strategies. We propose that the q-matrix method can also be used as a data mining tool, not just to mine for student knowledge, but also to find relationships in many other types of data [see discussion of data mining and the q-matrix method in Section 2.5].

1.2 Adaptive tutorial systems and student assessment

The goal of any tutorial system is to guide students from understanding little or nothing of a topic to a higher level of understanding and ability. Instructors typically determine a minimal level of prerequisite knowledge, and design instruction assuming that students understand the prerequisites and have no other preparation for the material. In reality, many students may have met prerequisites without truly understanding them, or may understand much more than the basics when starting a course.

Adaptive tutorial systems, on the other hand, strive to meet the same instructional goals, while tailoring the learning experience to the actual knowledge a student can demonstrate. This approach can address many of the problems of traditional instruction, offering students the chance to build prerequisite skills before moving on to more complex material, or the chance to move ahead more quickly as they become ready. In order to provide this individualized instruction, adaptive teaching systems must include knowledge assessment that can guide the teaching process.

There are many types of adaptive tutoring systems [Dav03, Kav02, Bur88, Cla86, Fra96, Ger00, Hum95, Kav00, Kav02, Mis96, Mor84, Sel98, Ste91, Tye96, VN01]. As discussed before, most of these systems are based on complex user models, including Bayesian networks, neural networks, procedural models, and other domain-specific models. These systems have generally been shown to be effective in assisting student learning, allowing systems to adapt and give hints to students based on their behavior.

However, most of these systems also require extensive work on the part of area experts to develop models capable of directing student learning.

The Alberta Research Council (1995) reports "in order to allow instruction to be individually designed, it is first necessary to capture the student's understanding of the subject." [Sta96] Knowledge assessment is an essential component of any adaptive teaching system. Without knowledge assessment, there is no way of measuring the results of teaching, or tailoring further education.

In their article, "What does research say about assessment?" Dietel, et al., write:

> The results of a good test or assessment, in short, represent something beyond how students perform on a certain task or a particular set of items; they represent how a student performs on the objective which those items were intended to assess. [Die91]

A good assessment, in other words, does not just report those questions a student missed, but offers a stronger reflection of the skills and understanding underlying a student's performance. The results of a good assessment, then, offer reliability and robustness across time and circumstance [Die91, AA96].

When we are assured of a good assessment, we can then use that assessment to make decisions about the teaching process. Not only will a good assessment provide an adaptive tutorial system with a method to measure student understanding and readiness to move on, it will also provide education researchers with a method for comparing teaching strategies across students. Since an assessment provides a measure of success in teaching, we can compare the results of assessments before and after the application of several teaching strategies across a large group of students to determine which strategies are the most effective.

1.3 Research challenge

> Computer technology may be an efficient and versatile tool for teaching, but all computer-based instruction must be governed by the same theories and research applicable to all teaching tasks. [Ove94]

One of the most effective teaching methods is private tutoring [Blo84b]. However, the expense of private tutoring is prohibitive for most people and schools. On the other hand, with lowering computer and Internet access costs, most US schools and many students have access to computer-based educational systems like never before.

Ideally, computer based educational systems can approximate the benefits of private tutoring at a much lower cost. To deliver this performance, researchers must develop a systematic approach to designing, implementing, and evaluating online educational systems.

As discussed in the previous section, one of the main challenges of creating tutoring systems online is assessing just what a student does and does not know [Sta96]. In other words, a computer tutor must be able to diagnose and correct student misconceptions, and distinguish these from careless errors or guesses. Indeed, much of the research in diagnosing misconceptions acknowledges the importance if distinguishing 'slips' from true misconceptions [Bro78, Bro80, Tat83, Bir93, Mar95, Van98 and others]. Once student knowledge is assessed, a tutor, human or computer, must then determine the path to best lead a student to reach her educational goals.

This dissertation explores the q-matrix method, which examines the inputs of many students to automatically extract relationships between questions and underlying concepts, and then uses those relationships in diagnosing and correcting student misconceptions. This system builds fault tolerance and robustness, optimizes for student performance, and can be used to optimize teaching strategies for effectiveness. This combination of assessment and the ability to optimize teaching strategies will help make it possible to deliver individualized, high quality education at a distance – near or far.

1.4 Research objectives & questions

One primary goal of this research is to further develop scientific, fault tolerant methods to improve education. Most intelligent and adaptive tutoring systems do have methods for determining student knowledge states, but many of these require the construction of domain-specific knowledge models. One of the most important improvements FTT methods, such as the q-matrix method, offer is to provide a reliable way to determine student knowledge states that is domain-independent. Once student knowledge can be assessed automatically, computer-based educational systems can be tailored to each student's learning needs in an expandable way, without human intervention [as in Kav00, Kav02, and others]. In addition, once we can assess student knowledge, we have a framework for the scientific investigation into the effectiveness of different teaching strategies.

In particular, this research solidifies the foundations of fault tolerant teaching methods through further experimentation and analysis. To study FTT methods further, we developed a new adaptive tutorial, and augmented 2 existing tutorials with FTT methods. These experiments compare the effectiveness of the q-matrix models across varied difficulty levels and topics, and also between remediation paths chosen by the student versus those prescribed by our methods.

A combinatorial experiment design allows the comparison of FTT methods across three different topics, topic difficulty, and student ability. This research analyzes the validity of FTT methods for diagnosis and interpretability. It also compares automated diagnosis and topic decomposition with those determined by field experts, and compare self-guided student work to that guided using FTT method.

The evolution of q-matrix theory lead to several questions that will determine its validity and usefulness in the field of fault tolerant teaching and its applications to other fields. Some of these questions are:

- Can we effectively guide remediation?
- Can we interpret q-matrix scores?
- What factors affect q-matrix extraction and accuracy?
- How well does the q-matrix method compare with other data mining methods?
- How do extracted q-matrices compare with expert-constructed q-matrices?

The answers to these questions will provide guidance in future uses of q-matrix theory and fault-tolerant teaching systems. This research found that q-matrices can be useful in guiding remediation, their scores can be easily interpreted, and that the q-matrix method compares well with both factor analysis and cluster analysis for data mining for student knowledge.

The question, "Can we effectively guide remediation?" is a complex question to answer. This really refers to the teaching strategy that underlies the remediation technique. In this research, the remediation technique is to direct students to review one question related to the concept they least understood. For one half of the students, the tutorial automatically took students to the section we predicted they should review. For the other half, the students were asked which section they felt they most needed to review. This provided us a comparison of the students' self-diagnosis as compared with the q-matrix knowledge assessment.

The meaning of q-matrix scores is extremely important in applying the q-matrix method to other fields of research. If, in fact, we have determined a method which will provide researchers with an understandable and usable model of a large set of data, this will be an advance over other data mining methods, which often partition data into groups without providing interpretations of these groupings. Our findings indicate that q-matrix scores are easily interpreted, and can be used to predict and analyze data sets.

In this research, we compare the q-matrix method with other data mining techniques of factor and cluster analysis to determine guidelines for the application of q-matrix methods in both fault tolerant teaching systems and data mining. As in Patrick Brewer's research, we found that the q-matrix method provides a fault tolerant extraction technique in the presence of noise and smaller data sets, making the q-matrix method a good tool for preliminary data mining applications.

Since the diagnostic accuracy of the q-matrix method has not before been tested, this aspect of the research is important in determining future applications of the method. Our research suggests that the q-matrix method can predict the topics that students most need to study to improve their knowledge.

1.5 Research hypotheses

Several hypotheses about the answers to our research questions were made based on preliminary findings at the beginning of this study:

- Q-matrix scores will converge to zero and one values.
- Q-matrix scores will demonstrate latent relationships among questions.

- Factors that might affect the extraction of Q-matrices are: the number of questions, the redundancy of questions and the existence of relations among the questions, etc.
- Q-matrix diagnosis of concept states will correspond well with human knowledge assessment. In other words, student misconceptions could be diagnosed at least as well as students themselves could determine which topics they misunderstood.
- The q-matrix method provides a good guide for remediation.
- The q-matrix method can perform the data mining task of data clustering and assist in the resulting cluster interpretations.

The purpose of this dissertation was to investigate these hypotheses in a more in-depth, multi-faceted study to reveal the strengths and weaknesses of the q-matrix method as applied for knowledge assessment and remediation, and also as a data mining tool. In the next section, we briefly outline the remaining chapters of this dissertation.

1.6 Dissertation Outline

In this chapter, we have provided the motivation and challenge to create a fault-tolerant teaching system that is adaptive, inexpensive, and expandable. As we have mentioned before, numerous teaching systems exist with excellent performance in teaching students, but few systems with general application to all domains exist. Thus, the development of the q-matrix method as a tool to make any educational system adaptive is an important step in improving educational systems. We have also shown some of the commonalities between data mining, teaching, and assessment. After these motivating questions, we listed our primary research objectives, questions, and hypotheses. In the course of this dissertation, we discuss the foundations of fault tolerant teaching, and how we have built and assessed our own fault-tolerant teaching system. We have also shown how the q-matrix method, the central aspect of this system, can be applied as a data mining tool.

In Chapter 2, we trace the history of the q-matrix method from its inception to its current state as a statistical method for student knowledge modeling. The q-matrix method began as the rule-space method to diagnose student errors in basic math. Later, as it was applied to quizzes on several thousand students, researchers began to wonder if expert models would ever be useful in diagnosing student misconceptions, and came up with ways to build models using actual student data. We also discuss related procedural models of student knowledge and how they have contributed to the field of knowledge assessment.

The methodology and experimental design for this dissertation are given in Chapter 3. In this chapter, we discuss our experimental design, the population and study environment, and the data collection and analysis procedures used in this dissertation.

Chapter 4, entitled "Student Model Extraction," compares the q-matrix method with two other major data-modeling methods, including factor analysis and cluster analysis, to determine its differences and the advantages of applying each method to student data to build a student knowledge model that can be effectively used for knowledge remediation.

One of the central aspects of this dissertation, Knowledge Assessment is the subject of Chapter 5. In this chapter, we examine the q-matrix method in three important ways. First, we use the q-matrix model to predict student test scores. Second, we compare the extracted q-matrices with those generated by area experts for this and other research. Finally, we compare the q-matrix predictions of the concepts that students least understood with student choices in their own learning process. With these three comparisons, we can determine the usefulness of the q-matrix method as a model for knowledge assessment, how closely it correlates to models that instructors might build for their own topics, and also its effectiveness in choosing a learning path for students.

In the course of this research, students use three tutorials to learn three topics in a Discrete Mathematics course. The sixth chapter discusses the findings of our tutorial surveys, where students were able to comment on the tutorials, and their effectiveness when compared to other learning methods.

Chapter 7, Proofs Analysis, demonstrates the use of the q-matrix method as a true data-mining tool. In this chapter, we see the q-matrix method applied in a novel way, and use our knowledge of the subject area to determine if this method builds models we can use to understand and interpret observed data.

In Chapter 8 we summarize our findings. First, we give a brief overview of each of these chapters. We then summarize the conclusions drawn from each of these chapters. Finally, we make suggestions for promising areas of future work in the field of q-matrix methods in teaching and data mining. At the end of this dissertation, we include our References and Appendices.

2 Theoretical background and related work in FTT

Many experts have dedicated their lives to better education. In this chapter, we will discuss the work of others in the field of fault tolerant teaching. In the Introduction, we have mentioned several adaptive and intelligent tutoring systems that are also effective learning tools. In this section, we focus on the systems that led to the development of the q-matrix method as a domain-independent knowledge model that can be used for directing student learning. We first examine rule-based systems used to predict and diagnose student errors in mathematics, including work by Tatsuoka, Brown, VanLehn, and others. In Section 2.2 we then discuss Robert Hubal's comparisons of Tatsuoka's predictions and randomly generated rules with actual student performance. Section 2.3 discusses how Patrick Brewer tested the theory that rules can be derived strictly from student responses without knowledge of the subject area. In Section 2.4, we discuss the work of Susan Jones and Jennifer Sellers, who wrote interactive lessons using question generators and answer judging, and collected data for small empirical tests of q-matrix theory. In addition, we show how Jennifer Sellers compared Q-matrices derived from student data with those determined by subject instructors, testing the interpretability of the extracted "rules" or concepts. In the final section of this chapter, we discuss some very basic concepts in data mining, and how the q-matrix method can be thought of as a data-mining tool.

2.1 Procedural models of student knowledge

As discussed above, student knowledge assessment is imperative in the design of individualized instruction. One way to assess student knowledge is to create a model of the subject being taught and a corresponding model of the knowledge a student has attained so far. These two models can be used together to evaluate and guide a student's learning until his knowledge model closely corresponds to the subject area model.

Researchers investigating learning processes have typically built models such as these in areas such as mathematics and the sciences, where the structure of knowledge is assumed to be highly procedural [e.g. Bir93, Bro78, Bro80, Bur82, Ste82, Tat83, Mar95]. J.S. Brown and his associates have built a procedural model for solving arithmetic problems, and have designed similar procedural models of how students solve arithmetic problems as their understanding is growing [Bir93, Bro78, Bur82, Van90]. Van Lehn extended these ideas to build Bayesian networks to model student procedures while learning basic physics [VM98, VN01]. These and other systems offered researchers important insights into human learning and cognition.

Like Brown, Van Lehn, and others, Kikumi Tatsuoka and her associates began investigating the procedures students use in solving algebra problems [Tat83, Bir93, Mor84, Edd84]. However, since Tatsuoka's goal was to diagnose and correct misconceptions, rather than to understand and model human cognition, her research took a practical turn that led to the development of q-matrix theory and eventually to the idea of fault tolerant teaching.

Procedural models of knowledge have been very important in building understanding of human cognition and learning, as we will discuss below, but have several drawbacks in adaptive tutorial systems. These models require extensive time and effort to construct – and are good only for one specific subject area. After they are constructed, research has shown that even extensive libraries of errors cannot predict all the procedural errors that students commit, and not all student errors can be explained by erroneous procedures (for example, when a student only makes mistakes when a zero is present). In cases where erroneous procedures can account for student errors, it is not certain that these procedures are those being used by students. Student errors are also not consistent over time, so these diagnoses may not consist of an accurate assessment of student knowledge.

Because of these limitations, procedural models of student knowledge may not be the best choice in teaching and assessment in adaptive tutorial systems. However, these models have formed the basis of the theory of fault tolerant teaching methods. In this section, we will discuss these procedural models, and the beginnings of their evolution into q-matrix theory and fault tolerant teaching methods.

2.1.1 Buggy, repair theory, and Sierra theory

In the late seventies, J.S. Brown et al. found that students are "remarkably competent" rule followers, but often follow erroneous rules [Bro80]. Based on this idea, Brown and his colleagues developed Buggy, a diagnostic model that would simulate students with 'buggy' arithmetic skills – using procedures that are slight variants of correct rules [Bro78, Bro80, Bur82]. Brown et al. constructed "an extensive catalogue of precisely defined systematic errors" made by students learning new arithmetic skills. Based on this catalogue, Brown built a diagnostic model that solves problems using procedural networks containing variants of correct rules. Using the model, educators could learn about 'buggy' student behavior and practice diagnosing student errors.

Buggy was an important step in understanding student misconceptions. However, because of its procedural structure, its application is restricted to well-defined procedural domains. In addition, Buggy depends on a fixed library of bugs, which requires considerable effort to construct, and restricts the power of the model to already known errors. Another deficiency in the model is that students do not always consistently apply the same rules to similar problems, as Buggy does.

To better model student behavior, Brown and VanLehn incorporated learning theories into the model, evolving into repair theory and eventually Sierra theory [Bro80, Bur82, Van90]. These systems are also simulations of students learning new skills. In repair theory, it is postulated that students learn when they reach in impasse, or a place in a problem where more than one learned rule can apply. At an impasse, the learner model implements a 'repair' – a heuristic used to continue solving the current problem.

Sierra (VanLehn's learner model) produces rule sets, which are predictions of procedures that students could learn during a lesson. These rule sets are passed to a solver, which implements the rules determined to solve problems. These generated answers are diagnosed as buggy or not by an automated diagnosis tool called Debuggy [Bur82]. Sierra is evaluated by comparing the overlap between observed and predicted bugs. In one study of 1175 test items, 75 bugs were observed [Van90]. Sierra predicted 28 of these, failed to predict 47, and also predicted 21 unobserved bugs. Most of the unpredicted bugs are 'pattern errors,' such as n*0=n, which could be easily incorporated into the Sierra model. However, VanLehn resists including such errors, saying, "empirical quality is not the only measure of theoretical validity. It must be balanced against explanatory adequacy…" [Van90]

The motivation behind developing these systems was to understand basic knowledge structures and the development of human cognition. If these systems display behavior similar to human learning and problem solving, then they will be judged as plausible explanations of the processes that students may go through while learning. However, when designing a model to diagnose student learning, we must measure its performance against its effectiveness in identifying areas in which students need remediation. Students often display behavior that is unexplained by these systems. However, they have provided a strong basis of theory with which to develop further research into learning.

2.1.2 OLAE and POLA

VanLehn has continued his work in the field of student modeling, using Bayesian nets to model student learning [Ger00, Mar95, VM98, VN01]. The OLAE (Online Assessment of Expertise) system observes students doing introductory physics problems and computes probabilities that a student knows and is using each of the rules in a given knowledge domain. Van Lehn's POLA (probabilistic online assessment) system is similar, but turns the knowledge tracing of OLAE into a system of probabilistic reasoning, generating predictions about the solution path the student is taking. This results in an assessment of student mastery of knowledge.

POLA and OLAE are similar to the Buggy system in that they still require an expert to construct a procedural model of the domain. In POLA and OLAE, the procedural networks are augmented with probabilities that a student has used each step in the model. These probabilities provide more specific understanding of the rules that students may be using, and can help direct remediation. However, Bayesian networks still require extensive time and effort to construct, are not extendable to other subject areas, and may not accurately predict or assess the actual procedures that students are using. In addition, these systems have no ability to explain, predict, or remedy non-procedural errors that students may commit.

2.1.3 Rule space theory and Q-matrices

The original inspiration for the q-matrix method came from Kikumi Tatsuoka and her associates, who, over the course of more than a decade, explored student misconceptions as they learned basic math concepts, such as adding fractions [Bir93, Edd84, Mor84, Tat83, Tat87]. The main goals of this research were diagnosis of students' misconceptions that could be used to guide remediation, assess group level performance as a measure of teaching effectiveness, and discover difficult areas for students [Bir93].

Tatsuoka, et al. began their exploration by hypothesizing erroneous rules that students may apply to solve problems. However, it takes countless hours of work by expert teachers and researchers to diagnose all the possible misconceptions that students may have – and in many cases, instructors may have little time or data to solve this problem. To address these issues, Tatsuoka developed the rule-space method to diagnose the rules used by students in fraction addition. In the rule-space method, all question responses resulting from erroneous and correct addition rules derived by experts are mapped into a plane, and student responses are compared to these points. The point in rule space closest to the student response corresponds to the rule the student is assumed to be using. This method improves on other procedural models, by creating a space where all student responses can be mapped and compared to expert predictions.

This idea of determining a student's knowledge state from her responses to test questions resulted in the creation of a q-matrix, originally a binary matrix showing the relationship between test items and latent or underlying attributes, or concepts [see Bir93]. In Tatsuoka, et al.'s work, a q-matrix is the result of expert analysis of questions into their underlying attributes. To build the q-matrix, experts constructed a relationship between test questions and concepts (referred to as attributes) and students taking the test were assigned knowledge states based on their test answers and the constructed q-matrix [see Ham85 for a discussion of item-response theory].

An example of a binary q-matrix is given in Table 1. In Tatsuoka's work, a q-matrix, also called an attribute-by-item incidence matrix, contains a one if a question is related to the concept, and a zero if not. Brewer later extended these to values ranging from zero to one, representing a probability that a student will answer a question incorrectly if he does not understand the concept [Bre96]. In Table 1, question 1 depends on concept 2 but not concept 1. Therefore, if a student understands concept 1 and concept 2, he will ideally answer all questions correctly. If a student understands concept 2 and not concept 1, then he will answer questions 1 and 2 correctly and 3, 4, and 5 incorrectly.

Table 2-1 Binary q-matrix example

	Questions				
	1	**2**	**3**	**4**	**5**
Concept 1	0	0	1	1	1
Concept 2	1	1	1	1	0

Using the rule space method, Tatsuoka et al. were able to identify students in need of remediation, un-predicted student errors, and topics of difficulty, and to construct "partial mastery charts" which would guide in remediation strategies for instructors.

Partial mastery charts are paths through the rule space from a current student knowledge state to the desired state. Tatsuoka and her associates assumed that the easiest transitions for students to make in correcting their misconceptions were to move from one knowledge state to the next best (higher mastery level) state, which is closest in its associated ability level. This assumption corresponds to Vygotsky's theory that learners best learn concepts that are only slightly more difficult or complex that those they already understand [Vyg86].

Tatsuoka's rule space research opened a new space of possibilities, showing it possible to diagnose student knowledge states in an automatic way, based solely on student item-response patterns and the relationship between questions and their concepts. The rule space method, however, is very time consuming and topic-specific, and requires expert analysis of questions. Its other weakness is that, it is well known that experts and novices differ greatly in problem-solving strategies. The rule space method has no way to measure or validate that the rules and relationships derived by experts are in fact those used by students. In fact, in [Bir93] two separate Q-matrices were used on the same questions and each q-matrix resulted in different difficulty measures for the same question. Although we know that different students may perceive the same question in different ways, we wish to find a diagnostic method that can be used across all problems, and that will differentiate among those students with different mastery levels.

2.2 Prediction of student data and changing Q-matrices

Tatsuoka's q-matrix methods appear promising, offering researchers an automated way to diagnose student misconceptions, partially bridging the gap between procedural models of knowledge and student behavior. Q-matrices were originally designed to encompass the procedures that experts believed to be important in solving problems, but also included problem "attributes" that don't necessarily correspond to procedures; for example, having a negative number in a problem would be one attribute of that problem. The strength of this idea is that it has created a model that encompasses both rules and attributes in a way that has transcended rules, assuming that students with similar responses have similar understandings and misunderstandings of the topic.

Though Q-matrices seemed to allow the diagnosis of errors that may not be procedural, research was still needed to compare actual student responses with q-matrix predictions.

In his 1992 Master's Thesis, Robert Hubal investigated how well student data in Tatsuoka's experiments supported the Q-matrices she created [Hub92]. In his experiment, Hubal used Tatsuoka's database of thousands of student responses to a 20-question math quiz. Hubal then used Tatsuoka's q-matrix of these 20 questions and 8 concepts to analyze student responses to determine how well the data supported the hypothesized q-matrix.

Since it was unclear whether the student responses were saved in order from left to right or right to left, the student responses were compared to the q-matrix in both directions. The assumption was that it would be apparent from the q-matrix comparison which ordering of the question responses was valid. Instead, the correlation of student responses with those predicted by the given q-matrix was the same in either direction, suggesting that the q-matrix made by hand did not well predict student behavior.

In further explorations, Hubal randomly generated rules that students may use to solve problems, and selected those which resulted in solutions that were similar to those found by students. Hubal found that students did seem to be using a small set of possible rules, but that none of these rules was predicted by Tatsuoka's q-matrix.

In using Q-matrices to understand student behavior, we can choose to either: 1) select a q-matrix and try to create or find questions whose relationships to underlying questions are supported by student data, or 2) use questions we already have and try to construct a q-matrix based on actual student data. In his research, Hubal experimented with the first option, by generating questions, testing them on students, and deleting those questions that did not exhibit the relationships predicted by Tatsuoka's q-matrix.

For several reasons, limiting questions to those that exhibit behavior predicted by a hand-constructed q-matrix may not be the best choice. First, the questions have been designed by an instructor to test student performance, and a q-matrix is a much more abstract measure of the relationships of questions to concepts. We might assume that the questions designed to test students are a more accurate reflection of the teaching objectives than an abstract construct which relates questions to underlying concepts. Second, the selection of questions requires many students to answer questions that will later be ignored, and data from these questions is then lost. It is quite possible that questions that do not exhibit the relationships predicted by a q-matrix can reveal unknown aspects of student behavior and understanding.

The alternative to this strategy is to design a method to extract a q-matrix, which explains student behavior, and reveals the underlying relationships between questions. Using such a strategy, student responses to existing questions would be analyzed to create a q-matrix that can explain the observed data. Experts can examine the resulting q-matrix to ensure that the extracted relationships seem to be valid, and then use that q-matrix to guide the generation of new problems.

2.3 Q-matrix extraction methods and fault tolerance

Inspired by Hubal and Tatsuoka's work, Patrick Brewer [Bre96] investigated methods of extracting the q-matrix directly from student data. Since it is impossible to truly know what a student knows, Brewer created "ideal" students on the computer, to design an experiment to study the feasibility of extracting the q-matrix when we know what students do and do not know.

Starting with a pre-determined q-matrix, Brewer generated all possible concept states to create "ideal" students. He then generated their predicted response vectors based on the q-matrix. Random bit flips in their 0/1 answers were added at rates from no flips up to fifty percent changes. For each noise rate, Brewer extracted the q-matrix using two techniques: the "q-matrix method:" an iterative hill-climbing technique, and common factor analysis. Comparing the two techniques, Brewer found that the q-matrix method was able to recover the original q-matrix for as few as 25 students with error rates as high as fifteen to twenty percent, whereas factor analysis required hundreds or thousands of students for equal recovery.

Brewer's research demonstrated that q-matrix extraction could succeed in spite of noise, introducing the possibility of fault tolerance in assessing and diagnosing student knowledge. In the following sections, we will discuss the two q-matrix extraction methods implemented in Brewer's research.

2.3.1 Extraction methods

The goal of q-matrix construction is to extract underlying, or latent, variables, which account for students' differential performance on questions. This extraction may employ any method that will successfully account for student response patterns. Some methods that might apply would include: traditional iterative hill-climbing methods, neural networks, cluster analysis, and factor analysis. We do not propose to explore the differences between these methods. However, Brewer's research, along with preliminary work on this thesis, have shown that traditional iterative hill-climbing techniques augmented with several random starting positions to be as effective, if not more effective, than factor analysis. In the next two sections, we will give a brief overview of common factor analysis and the q-matrix method devised by Patrick Brewer.

2.3.2 Factor analysis

In the social sciences, researchers often wish to measure a phenomenon, e.g. general intelligence, which cannot be directly observed or measured, often referred to as a latent variable. For example, to measure 'intelligence,' researchers would construct a survey with questions requiring verbal, spatial, mathematical, and other skills. Factor analysis would then be used to extract the underlying characteristics that might account for a respondent's survey answers. It is assumed in this case that general intelligence would influence all responses. In general, thousands of observances are necessary to extract reliable factors – the latent variables.

There are extensive works including details of factor analysis, including [Bre96, Kli94, Tat71, Thu47]. Here we will provide a brief overview of factor analysis. Factor analysis is a statistical process that attempts to extract latent (i.e. unknown and unobservable) variables from observed data. It starts using a correlation matrix, and then iteratively extracts eigenvalues and eigenvectors from the correlation matrix until a certain percentage of the variance in that matrix is explained by the extracted factors. Once the factors are extracted, they are rotated to better understand their meaning.

For our application, the variables to be analyzed are binary student responses to questions. Using this data, we compute the correlation matrix that represents the relationship between each pair of questions. These correlations are intended to give a measure of the predictability of the value of one variable with another. Elements on the diagonal of the correlation matrix are said to be communalities, a measure of how often observances of the same variable will coincide. In the case of test questions, students may not necessarily answer the same question in the same way a second time. Therefore, though the computed correlation matrix will have all ones on the diagonal, it is standard practice to modify these values when each variable is expected to have its own unique variance, as in the case of test questions. Communalities can be estimated in several ways, but it is often most effective to set the communality of each variable to the maximum correlation it had with any other variable.

Once the correlation matrix and communalities are prepared, factor analysis is applied iteratively to find 'factors.' The procedure finds the first eigenvalue of the matrix, whose magnitude is the variance accounted for by the first factor, and its corresponding eigenvector, which is the factor. Next, the residual matrix is calculated by subtracting the variance due to the first factor from the correlation matrix. (I.e. the residual matrix is calculated by subtracting the magnitude of the eigenvalue from each diagonal element of the correlation matrix). The values on the diagonal of the residual matrix represent the remaining variance to be accounted for by extracting factors. We apply the procedure iteratively to the residual matrix until the remaining variance or eigenvalues reach a pre-determined stopping criterion.

We could theoretically extract as many factors as there are variables, but since our goal is to reduce the dimensionality of the data, we wish to extract many fewer factors than there are variables. A common stopping criterion for factor analysis is to stop when the change in consecutive eigenvalues is leveling off.

When the factors are all found, they can be used to recover values near those in the original correlation matrix. It is difficult, however, to interpret the factors. Since the factors are extracted in magnitude order, the first factor tends to influence all the variables much more than the other factors. In addition, an infinite number of rotations of the factors are equally mathematically valid explanations of the data.

To address the problem of interpreting factors, several methods of rotation have been developed that attempt to create factors that have equal influences on all the variables, but have the most simple structure possible – meaning that the values of each factor are as close as possible to zero or one [Thu47]. Varimax rotation is most commonly used, since it obtains an easily interpreted orthogonal simple structure.

Mathematically, the extracted factors are orthogonal linear combinations of the observed variables. This has raised some controversy in the field since the goal of factor analysis is to reduce the dimensionality of the observed data. Some have argued that, instead, factor analysis adds dimensions to the data, since in the final analysis we have a set of factors, which are linear combinations of the original variables, and a representation of each of the original variables in terms of the factors.

In addition to this problem, Brewer found that in terms of analyzing generated student data, factor analysis was much less tolerant to noise (random flips in student responses) than the q-matrix method with small numbers of students (less than 100). This may be explained by the fact that factor analysis first reduces data to a correlation matrix, which represents a loss of information. The q-matrix method is optimized to minimize the prediction error for each individual student, which may explain some of the difference in prediction between the two methods.

2.3.3 The q-matrix method

The q-matrix method is a simple hill-climbing algorithm that creates a matrix representing relationships between concepts and questions directly. The algorithm varies c, the number of concepts, and the values in the q-matrix, minimizing the total error for all students for a given set of n questions. To avoid of local minima, each hill-climbing search is seeded with different random Q-matrices and the best of these is kept.

First, c, the number of concepts, is set to one, and a random q-matrix of concepts versus questions is generated with values ranging from zero to one. Also, 2^c concept states are generated. A binary string of length c represents each state where a zero in position k represents that a student in this state does not understand concept k, and a one represents that he does. For each concept state q, its corresponding ideal response vector is generated based on the q-matrix.

For example, for the state 01, and the following 3 by 5 q-matrix (Table 2), the ideal response vector would be 11000. To find this response vector, we examine the column for each question to determine a student answer. For question 1, the q-matrix tells us that concept 1 is not needed for the concept, while concept 2 is. Since the student does not know concept 1 and does know concept 2, the student should answer question 1 correctly. Similarly, the student will answer question 2 correctly. However, questions 3, 4, and 5 all require understanding concept 1 to answer each correctly. Therefore, we predict that this student will answer questions 3, 4, and 5 incorrectly, giving us the final predicted "ideal" answer vector of 11000.

Table 2-2: Example binary q-matrix

	Questions				
	1	2	3	4	5
Concept 1	0	0	1	1	1
Concept 2	1	1	1	1	0

For q-matrix values other than 0 or 1, the q-matrix value (x, y) is interpreted as the probability that a student will answer question y incorrectly given that he does not understand concept x [Bre96]. Using this value, the most probable ideal response vector is calculated for each concept state.

The next step is to compute the total error for the q-matrix computed over all students. For efficiency, we create an array of size 2^n of all possible response vectors, and the *i*th element of the array contains the number of students with response vector *i*. Each observed response vector is compared with each ideal response vector, and is assigned to the one that is closest to it in Hamming distance. This distance is the error for that response vector. To compute the total error for the q-matrix, the individual errors for each observed response vector are multiplied by the total number of students with that response, and summed over all observed response vectors.

After the error has been computed for a q-matrix each value in the q-matrix is changed by a small amount, and if the overall q-matrix error is improved, the change is saved. This process is repeated for all the values in the q-matrix several times, until the error in the q-matrix is not changing significantly.

After a q-matrix is computed in this fashion, the algorithm is run again with a new random starting point several times, and the q-matrix with minimum error is saved, to avoid falling into a local minimum. It is not guaranteed to be the absolute minimum, but provides and acceptable q-matrix for a given number of c concepts.

To determine the best number of concepts to use in the q-matrix, this algorithm is repeated for increasing values of c. The final q-matrix is selected when adding an additional concept does not decrease the overall q-matrix error significantly, and the number of concepts is significantly smaller than the number of questions.

Brewer found this hill-climbing method to be effective in extracting Q-matrices for small groups of ideal students with random noise added to their responses. This method was more effective for small groups of students than factor analysis. This difference in performance is most probably a result of pre-processing the student response data into a correlation matrix before implementing factor analysis. When forming a correlation matrix, we lose individual student data in favor of calculating average relationships between questions. The q-matrix method is optimized to assign each student the most appropriate knowledge state, using all available response data for each student.

2.4 Question generation and q-matrix extraction with actual students

Once Brewer verified the q-matrix extraction is feasible even in the presence of noise, it was important to test this idea with real questions and students. Soon after Brewer completed his research, Susan Jones [Jon96] and Jennifer Sellers [Sel98] both wrote interactive lessons on NovaNET, developing techniques for question generation and automatic answer judging. They each also collected preliminary student data and created Q-matrices from these data. In both experiments, and in recent work, students have felt positive about the online lessons. In both experiments, the extracted Q-matrices appeared reasonable, even with small numbers of students (7 and 17). These results suggest that it is feasible to extract Q-matrices from larger groups of student responses.

Jennifer Sellers also compared the extracted Q-matrices to those constructed by instructors. For two of her Q-matrices, expert and extracted Q-matrices corresponded in all but one matrix value. In Table 3, element (2,6) shows the only difference between the expert and extracted Q-matrices for one set of questions. In this case, instructors indicated that concept 2 indicated the application of more than one definition, and had a zero in position (2,6) since it corresponds to applying the definition of a transitive relation. However, the extracted q-matrix, with a one in that position, reflects that students find applying this definition much more complex than applying any combination of two other properties of binary relations. Upon a second look, experts agreed that this extracted q-matrix was appropriate.

Table 2-3: Expert versus extracted q-matrix

	Question							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Concept 1	1	1	1	1	1	1	1	1
Concept 2	0	0	0	0	1	0/1	1	1

For more complex problems, expert and extracted Q-matrices differed more greatly. In fact, instructors were less sure of their partitioning of questions into concepts. This raises the questions: is this characteristic of complex problems, or the small sample size? (There were only 17 students so Sellers chose to extract a 3-concept q-matrix). This suggests that computer extraction of concepts from real student data may be preferable to expert analysis.

Some of Sellers recommendations for future research are: 1) use few concepts for many questions and many students, 2) compute Q-matrices for small chunks of questions, 3) design a thorough experiment to test the correlation of q-matrix information and teacher assessment and question analysis, 4) further investigate the meaning of q-matrix values, and 5) to test the validity of using the q-matrix to guide teaching for remediation.

2.5 Data mining and q-matrix methods

Q-matrix methods are used to extract underlying relationships among observed variables, which is also one of the main goals of data mining. J. H. Friedman defines data mining as "the computer automated exploratory data analysis of (usually) large complex data sets" [Friedman 1998]. More specifically, "Data mining is a set of methods used in the knowledge discovery process to distinguish previously unknown relationships and patterns within data" [Ferruza, quoted in Friedman 1998].

Data mining researchers today often use principal components analysis, one form of factor analysis, as a preliminary step in processing large data sets. Since factor analysis and the q-matrix method described herein both offer methods of understanding larger data sets in terms of a much smaller set of "concepts," it is quite feasible that q-matrix methods can be used as data mining techniques. The advantage of applying q-matrix methods to data mining is the potential improvement in interpretability that q-matrix methods may offer. As Sellers found in her research, the results obtained through q-matrix analysis seem to describe relationships among variables in interpretable ways. Factor analysis and principal components analysis, on the other hand, do not readily offer interpretable results.

There are several important steps in a typical data mining process. The first step is data selection, where variables for analysis are selected. The second step is data clustering, usually achieved using cluster analysis, factor analysis, or other pattern-recognition algorithms. After this step, researchers analyze the resulting data clusters to interpret the meaning of each group of data. These results can then be used for other applications. We believe the q-matrix method will offer a new tool that can be used for data clustering and interpretation.

Data mining is an extremely important and ever-growing area of research [Cle93, Ede98, Eld98, Fay98, Fay96, Kol98, Mit97, Rip96]. As they are developed, new methods of pattern finding and recognition will be extremely important in gaining advances in the field of data mining.

2.6 Summary of related works

In this chapter, we have highlighted several works that led to the development of the q-matrix method. The theory began in parallel with research in procedural models of knowledge, but was developed as more of a diagnostic tool. Later researchers found that, although the q-matrix model was a good way to compare student data to a concept model, expert-constructed q-matrices did not correspond to student data any better than random q-matrices did. As a result, the q-matrix method of deriving q-matrices from student data evolved. The fault-tolerance of this statistical method was measured and compared with that of factor analysis on simulated students. Finally, two experiments extracted q-matrices from two small sets of actual student data, demonstrating that it was possible to determine useful q-matrices from real data. This dissertation is the first application of the q-matrix method on a large set of student data.

In our literature review, we discussed data mining and how its major goals are aligned with those of the q-matrix method. This research represents the first application of the q-matrix method as a general data mining tool – not to extract student knowledge, but to extract and understand basic relationships among a set of observed data.

3 Methodology

To compare performance of FTT methods, this research compared the Q-matrix methods in three NovaNET lessons in discrete mathematics: two existing lessons in binary relations and propositional calculus, and an additional new lesson in counting (combinations and permutations), which is a difficult area for students [Gar89]. These three lessons provide insight into the performance of Q-matrix methods in several levels of thinking, from knowledge and its application, to synthesis and evaluation [see Blo84a].

The binary relations and counting lessons provide students with instruction in basic definitions and asks questions relating to the application of those definitions, with most of its questions residing in the first three levels of Bloom's taxonomy of thinking. The propositional calculus lesson provides students with proofs to solve, allowing students to apply what they know at the highest level of Bloom's taxonomy: synthesis.

For the proofs tutorial, traditional q-matrix extraction was not appropriate. The tutorial is broken into two separate sections. The first section is designed to acclimatize students to the entry of logic symbols into the system, along with an elementary review of some basic logic axioms. These questions were not designed with underlying concepts tying them all together, so a q-matrix extraction for these questions is not appropriate. The second section of the tutorial is where students have ten different proofs to solve. In order to use the q-matrix method, we need a list of questions that all students answer, so we can build a model of relationships among these questions. However, section 2 of the proofs tutorial does not have simple questions whose answers we could compare across students. Instead, this section is more like a proof verifier, where students can each take their own path to a solution and the program verifies that their path is a valid one.

Because the format of its questions did not lend itself to q-matrix analysis, we used the q-matrix method as a data mining tool for the Proofs tutorial. Instead of comparing student responses on questions, we looked at the components of their solutions, and for each proof, we created a vector for the proof showing which of the possible components they used. The idea was to find underlying concepts relating the components used to solve a problem. This kind of application of the q-matrix method can have a tremendous impact in the field of data mining. A traditional data mining approach typically looks at a set of data and attempts to find patterns in that data, by some form of cluster or pattern analysis. After that step, the data miner looks for explanations for the discovered patterns. As a data mining tool, we hypothesized that the q-matrix method would succeed in simultaneously creating clusters of data and explaining the purpose of each cluster. In the Proofs tutorial, these clusters would represent the axioms that were necessarily used in conjunction to arrive at a successfully completed proof.

In the fall of 2002, students in CSC 226 Discrete Mathematics were required to use the three programs as supplements to their course. We planned to guide half of these students through remediation using FTT methods described herein, and to allow half to review and repeat parts of the lessons as they saw fit.

For each tutorial, we compared the attitudes and performance of both guided and self-guided students. The paths the self-guided students take through the lesson was compared with the path the diagnostic engine prescribed for them using FTT methods. We also compared performance for each of these groups to determine if there is a significant difference between the groups. This research will not be insignificant if these groups have similar performance. The most important aspect of this research is whether we are able to ascertain correctly what a student knows, in a way that can help us direct future study.

3.1 Adaptive tutorial design

The following figure demonstrates the design of the adaptive tutorial system used for the Binary Relations tutorial in this experiment. In the diagram, a student is using a NovaNET lesson, containing a Question Engine and a Diagnostic Engine. The Question Engine corresponds to the tutorial along with its question generators and answer judgers. The Diagnostic Engine contains the concept and student models, and a teaching strategy to apply with each student. When a student is using the lesson, they will be given examples and asked questions. The binary scores of the answers to these questions are fed to the Diagnostic Engine, which determines the student knowledge state and uses the prescribed teaching strategy to direct the student to the next portion of the lesson, by selecting new material within the Question Engine.

From the student perspective, this system can be a static system or an adaptive one. A static system will have a teaching strategy that corresponds to allowing the student to progress through the lesson as it is written, in the order prescribed by the lesson author. If the teaching strategy calls for redirecting students to review material, the system becomes adaptive to meet student needs, changing the path for learning.

There are several changes going on at all times within the system, even with a static teaching strategy. As a student learns, the Diagnostic Engine will track how his knowledge state changes. In addition, the Q-matrix contained in the Diagnostic Engine can change with the results of each student, or allow updates only after a batch of students have completed the lesson. In theory, even the teaching strategy can change along with student performance. This will be an important application of the Q-matrix method in future research.

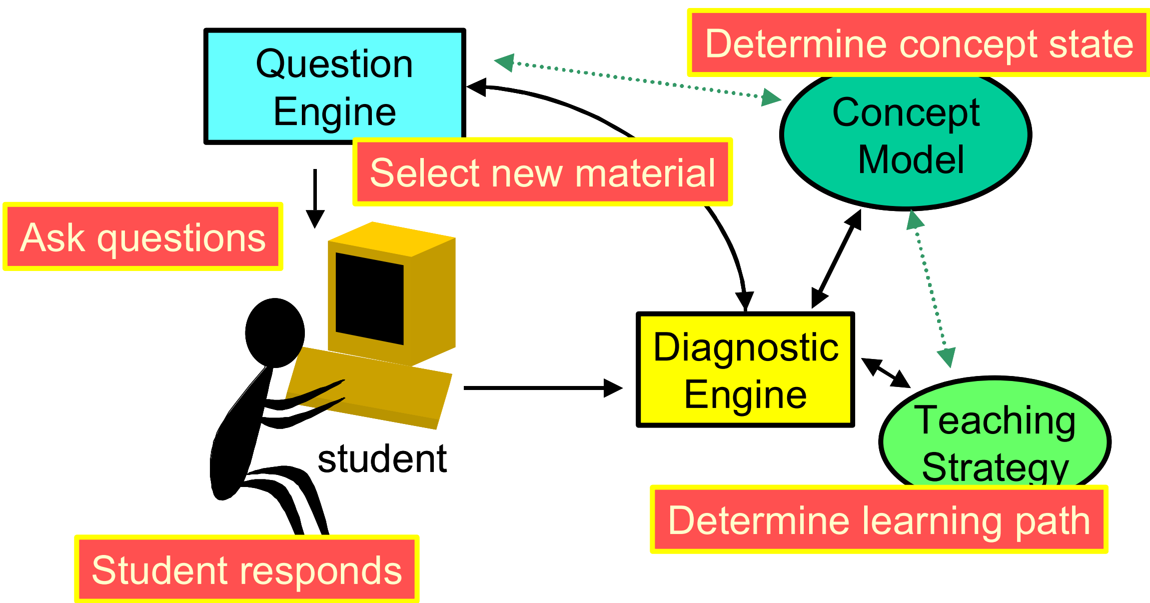


Figure 3-1 Adaptive tutorial system flow

The three lessons in discrete mathematics: binary relations, proofs, and counting, were administered to over 100 students in the Fall 2002 Discrete Mathematics course, CSC 226, at NC State University. The tutorials were required for credit in the class, but only a completion grade was assigned to each student. Students were randomly assigned to guided or self-guided groups.

The tutorials were administered both on NovaNET terminals in our office and students were also allowed to download a NovaNET client called Portal to complete the tutorials on their home computers. Since all of our data collection was electronic, the location where students took the tutorials should not affect our results. However, we note here that students taking the tutorial in our office almost always had quick access to help if they had trouble with the tutorials, and would probably cause their experience with the tutorials to be more positive, affecting their survey data.

3.2 Testing our research hypotheses

The hypotheses of this research, as we listed in the Introduction, are:

1. Q-matrix scores will converge to zero and one values.
2. Q-matrix scores will demonstrate latent relationships among questions.
3. Factors that might affect the extraction of Q-matrices are: the number of questions, the redundancy of questions and the existence of relations among the questions, etc.
4. Q-matrix diagnosis of concept states will correspond well with human knowledge assessment. In other words, student misconceptions could be diagnosed at least as well as students themselves could determine which topics they misunderstood.
5. The q-matrix method provides a good guide for remediation.
6. The q-matrix method can perform the data mining task of data clustering and assist in the resulting cluster interpretations.

Hypothesis 1 was tested throughout the research through inspection of each extracted q-matrix. Hypotheses 2 and 3 are tested in Chapter 4 (Student Model Extraction). First we analyze our collected tutorial data using factor analysis, cluster analysis, and the q-matrix method, comparing these results to determine the validity of the relationships found by the q-matrix method. Second, we compare these methods to determine the factors that affect the performance of the q-matrix method when compared to other methods.

In Chapter 5 (Knowledge Assessment), we examine our q-matrices as knowledge assessment tools, testing our fourth and fifth hypotheses with the Binary Relations tutorial. In this chapter, we compare student and q-matrix based choices for knowledge remediation. We also compare the extracted q-matrices with those created by area experts, and determine if the extracted q-matrices make sense in the topic areas.

In the next chapter, Survey Data, we examine the results of our tutorial surveys to determine areas for improvement in the surveys. A question that is particularly important for this research asks students if they felt the Binary Relations tutorial (the only one augmented with automated question review) knew which questions they least understood, offering support for our hypotheses 4 and 5.

Proofs Analysis, Chapter 7, offers an in-depth study of the q-matrix method as a data mining tool. Of particular importance in this chapter is our interpretation of each q-matrix model, offering us an in-depth understanding of the data, helping us investigate hypotheses 2 and 6.

4 Student Model Extraction

One major goal of this research was to compare the q-matrix model of student knowledge with other potential data-based models of student knowledge. In this section, we compare several methods to extract student knowledge with the q-matrix method.

4.1 Q-matrix method

The q-matrix method of model extraction is a hill-climbing method that varies q-matrix values to minimize the error in student answer predictions.

In the q-matrix method, student responses are also grouped into clusters by concept states. Each cluster in the q-matrix method is represented by its concept state, a vector of bits where the ith bit is 0 if the students do not understand concept i, and a 1 if they do. Each concept state also has associated with it an ideal response vector (IDR). We use the concept state with the q-matrix to determine the IDR. For each question q in the q-matrix we examine the concepts needed to answer that question. If the concept state contains all those needed for q, we set bit q in the IDR to 1, and otherwise to 0. When the q-matrix contains only binary values (not probabilities between 0 and 1), this can be calculated for a concept state c and the q-matrix Q by the following procedure, composing $\neg c$ with Q:

$$IDR = \neg ((\neg c)Q)$$

For example, given concept state c = 0110 and the q-matrix Q given in Figure 4-1 below, $\neg c = 1001$, $(\neg c)Q = 101001$. Therefore, $IDR = \neg((\neg c)Q) = 010110$. This can be explained by viewing $(\neg c)Q$ as all the questions that require knowledge in the concepts that are unknown for a student in concept state c. Thus, the IDR for c is exactly the remaining questions, since none of these require concepts that are unknown for a student in concept state c.

Figure 4-1 Example q-matrix

	q1	q2	q3	q4	q5	q6
c1	1	0	0	0	0	1
c2	1	1	0	1	0	0
c3	1	1	1	0	0	0
c4	1	0	1	0	0	0

When the q-matrix consists of continuous probabilities, we compute the IDR as explained above, but the negation symbol is interpreted as the probability of the opposite outcome, so in each case where a not appears, we interchange any following values x with 1-x.

4.2 Factor Analysis

Factor analysis is a statistical process that attempts to extract latent (i.e. unknown and unobservable) variables from observed data. The final results of a factor analysis include a factor matrix and factor loadings, respectively, the eigenvectors and eigenvalues of the correlation matrix of the observed data. Factor loadings are interpreted to tell us the relative significance of each of the factors. The factor matrix is typically interpreted to demonstrate which observed variables correspond to the latent, or underlying, factors that caused the observations. If the magnitude of a factor matrix score is 0.3 or more, it is said that the corresponding variable is related to the factor. If it is less than 0.3, the corresponding variable is assumed to be unrelated to the factor. Before interpretation, this factor matrix has been "rotated" to yield factors that are distinct from one another and also account for most of the observed variance. [Thu47]

In this analysis, we used SAS to perform a factor analysis on student answer vectors for the binary relations and counting tutorials, and on each proof solved in the proofs tutorial. (Please see the Proofs chapter for a discussion of how student proofs were mapped into answer vectors).

The SAS "proc factor" was used to perform a common factor analysis, as opposed to a principle components analysis. The difference between these two methods is in the communalities assumed for each variable in the analysis. For principle components analysis, the communalities are assumed to be one for each variable – in other words, variables are expected to vary perfectly with themselves. When we make this assumption, if the same experiment were to be run again with the same students in the same conditions, we would expect their answers to be identical to their first response. However, in this experiment we assume that student responses are subject to several factors that would influence their answers, such as random mistakes, and slightly different readings of the question. For this reason, common factor analysis is more appropriate, where the communalities at each stage in the analysis are estimated to be the maximum of the variance between one variable and all other variables in the experiment. In SAS, we set "PRIORS=max" in proc factor to obtain this analysis.

The stopping criterion for extraction of factors was the percent of variance criterion. This criterion stops extracting factors when 100% of the variance in the correlation matrix is accounted for by a combination of the extracted factors and the communality estimates. With this criterion, most of the experiments yielded a four-factor solution. For a fair comparison of the methods, the q-matrix analysis for each experiment was performed with the number of concepts determined by the number of factors in the factor solution. Then, for the factor method, the factor matrix was converted to zero and one values, where a one represents that the variable relates to the factor (meaning that the factor score magnitude was at least 0.3), and a zero represents that it does not (the factor score magnitude was less than 0.3). For the q-matrix method, the final q-matrix was converted to binary values as well using the same criteria. Note that these criteria were much more than was needed, since the q-matrix values were already near 0 or 1.

For both factor and q-matrix methods, the error per student is calculated in the same way. Since the factor matrix is used as a q-matrix, we'll describe the error computation based on a q-matrix of zeroes and ones. First, we create an array whose indices are concept states, from 0 up to 2^q-1, where q is the number of questions in the tutorial. We then tally the number of student responses in each state. Then, for each concept state, 0 up to 2^c-1, where c is the number of concepts in the q-matrix, we compute its ideal response vector (IDR). For each question q in the q-matrix we examine the concepts needed to answer that question. If the concept state contains all those needed for q, we set bit q in the IDR to 1, and otherwise to 0. This can also be calculated for a concept state c and the q-matrix Q by the following procedure:

$$IDR = \neg ((\neg c)Q)$$

Then, for each response with at least one student, we compare the response with all IDRs and choose the one closest in Hamming distance. This distance is called the "error" for the student response. We sum all errors over all students to determine the overall error for the q-matrix.

In all experiments, the q-matrix method had a lower error per student measure than factor analysis. However, the difference between the factor analysis error and q-matrix error varied widely. In Figure 4-2 the error per student is shown for both methods.

As we will see in the Figure 4-3, the largest differences in performance for the two methods occur when the number of distinct observations is small.

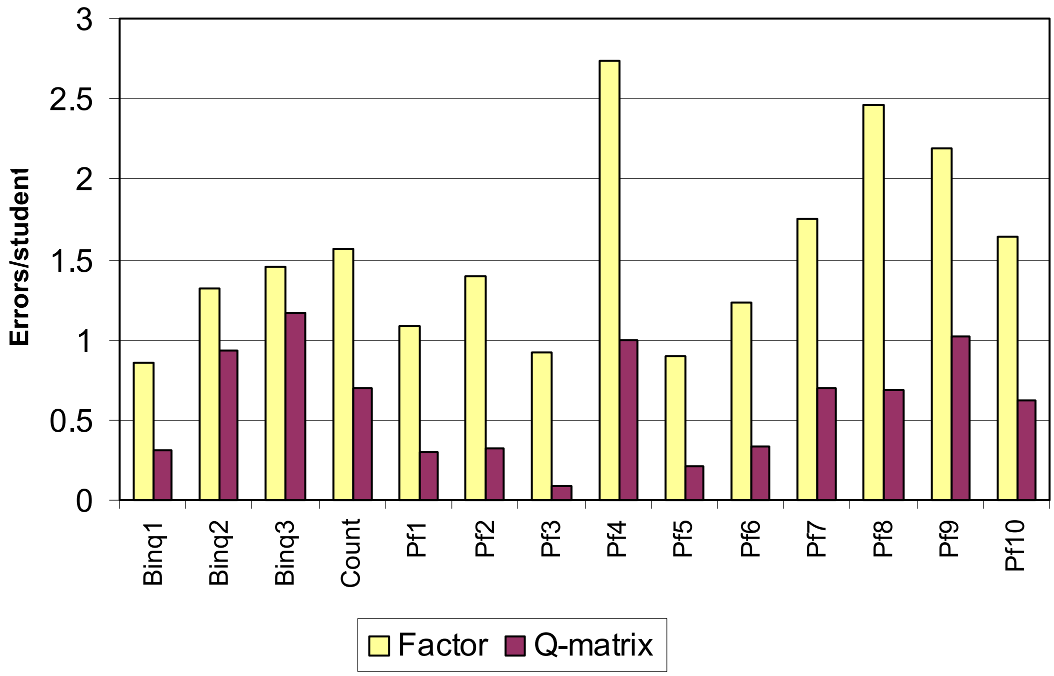


Figure 4-2 Factor analysis and q-matrix errors per student

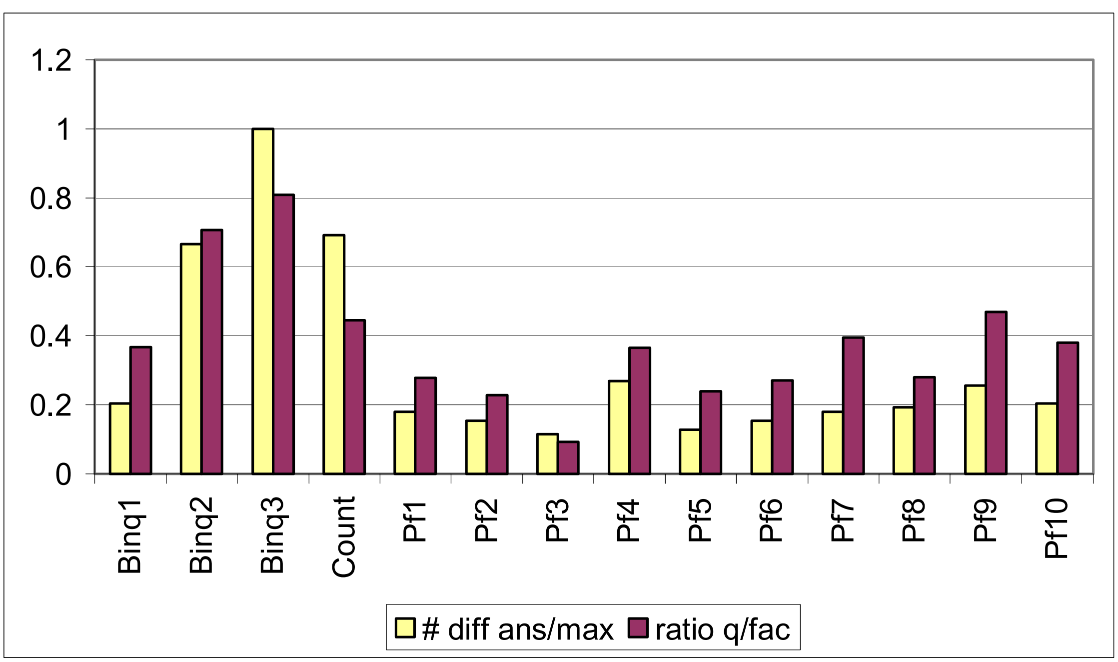


Figure 4-3 Ratio of q-matrix to factor error and relative # of distinct observations

Figure 4-3 shows that the difference in performance between the factor analysis and q-matrix methods can be primarily understood by examining the relative number of distinct observations made in each experiment. For each experiment, the first column plotted, # diff ans/max, is the actual number of distinct observations, divided by the maximum number of distinct observations made in an experiment. The third section of the binary relations tutorial, denoted by binq3, displayed the largest number (78) of distinct answer vectors, as shown by a value of 1 in the graph. The second column represents a ratio of the q-matrix error per student over the factor analysis error per student. Here, we can see that the higher the number of distinct observations, the higher the ratio of q-matrix error to factor error. In effect, this means that the advantage of using the q-matrix method is best seen when there are fewer distinct observations. As the number of distinct observations increases, the performance of the factor method more closely approximates the performance of the q-matrix method.

This observation corresponds to the findings in Brewer's previous research, which found that the factor analysis method performed poorly in comparison with the q-matrix method when fewer observations were available.

From this graph we can also note that the errors per student for the q-matrix method in each experiment are all less than 1.25, while the errors for the factor analysis method range from less than one to almost 3 errors per student. In each experiment, there were less than 10 questions in each quiz, making an error rate of 2 per student more than 20% error in answer prediction. Our goal is less than 1 error per student per quiz. This is much more consistently attained using the q-matrix method.

The difference in performance between factor analysis and the q-matrix method can be explained simply through an examination of the two methods of model extraction. In factor analysis we pre-process our data to create a correlation matrix, and the goal of our analysis is to create a model from which we can reproduce the observed correlation matrix. This pre-processing doesn't allow the model to explain the differences in individual student responses, as the q-matrix can do, since the q-matrix method builds its model by minimizing the sum of all individual student errors.

As Patrick Brewer predicted, the factor analysis method would require much more data to produce a reliable and accurate model to get an equivalent fit to student data [Bre96]. He estimated that the amount of data needed would be 100-1000 students, which is infeasible for most classes.

4.3 Cluster Analysis

Cluster analysis is a common technique of data mining that forms clusters of data based on a distance criterion. This type of data mining can provide researchers with a way to classify data without a priori knowledge of the classification types. In a similar way, the q-matrix method also forms clusters of data by concept states.

We performed several analyses to compare cluster and q-matrix analysis methods. The first method we used was to compare the q-matrix and cluster analyses in a straightforward way, as we did with the factor analysis results. As we might expect, cluster analyses generally created clusters that fit the student data better (since clusters are not constrained as q-matrix clusters are to have relationships between the clusters). However, in addition to clusters, the q-matrix method offers us one more step in the data mining process towards cluster interpretation. Second, we created graphs to demonstrate the overlap in the clusters created by these 2 methods, and found that the two methods did tend to keep groups of responses together, even if their clusters did not always correspond well. Finally, we considered each extracted q-matrix clustering as a step in an iterative k-means cluster analysis, to see if they met the convergence criteria for a k-means cluster analysis.

4.3.1 Comparison of q-matrix to cluster analysis error

In this section, we describe a straightforward comparison of the q-matrix and cluster analysis models using the total error for the model as we did with factor analysis. First, we ran the q-matrix analysis for each experiment, and determined the number of concept states that student responses were divided into. These states form a partition of the student responses, much as a cluster analysis would. Therefore, we then ran a SAS cluster analysis for each data set using the number of clusters actually used in the q-matrix analysis. Finally, we compared the error for the q-matrix with cluster analysis.

To compare the performance of cluster analysis with the q-matrix method, we ran a cluster analysis for each experiment using the SAS procedure fastclus, with parameters least = 1 and maxclusters set to the number of clusters set to match the number of clusters found in the q-matrix analysis. The "least = 1" parameter sets the distance metric to the Hamming distance metric, so that the distance between two observations is measured by the number of differing bits in the two strings.

The proc fastclus algorithm is an iterative algorithm. First, a random seed is chosen from the data set for each cluster (where the number of clusters is set by the maxclusters parameter). Then, each observation is compared with the seeds, and assigned to the cluster whose seed is nearest (using the Hamming metric). At the end of an iteration, each cluster seed is reset to be the median of the cluster. The observation clusters are recalculated, and cluster seeds reset, iteratively, until the maximum relative change in the cluster seeds is less than 0.0001. The relative change in a cluster seed is the difference between the new and old seed values, divided by the mean absolute deviation from the cluster seeds in the current iteration. [See SAS help system]

The number of clusters generated for each experiment is given in Table 4-1. These were the number of clusters created when the q-matrix method was run using 2, 3, and 4 concepts for the same experiment. When generating each cluster analysis, we set the number of clusters to that used in the q-matrix method for a fair comparison.

Note that, though the q-matrix method can potentially model students falling into 2^n concept states (or clusters) when the number of concepts is n, it does not always do so. When the q-matrices were generated, student response vectors of all zeroes or all ones were disregarded. In a final analysis, students with all-one responses would be placed in the state of knowing all concepts, which students will all-zero responses would be placed in the state of knowing no concepts. For most experiments, when 2 concepts were used, all 4 possible concept states were used. When 3 concepts were used, students could fall into 8 difference concept states, but most analyses used 6 or 7 of these states. When 4 concepts were used, the number of clusters used ranged from 7 to 11 clusters in each experiment, though 16 different states were available with 4 concepts.

Table 4-1 Number of clusters generated by 2, 3, & 4 concept q-matrices

	binq1	binq2	binq3	count	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
2-con clus:	4	4	4	4	3	4	4	4	4	4	4	4	4	4
3-con clus:	7	7	8	8	6	6	5	6	6	6	7	6	7	6
4-con clus:	11	14	16	16	10	9	7	10	8	7	10	10	11	11

Figure 4-4 is a chart of the ratio of cluster over q-matrix error for each run of the experiment. In all cases, except for Proof 2 with 2 concepts, and Proof 3 with 3 concepts, the cluster analysis method resulted in a much smaller error for students than the q-matrix method. In the exceptional cases, there were fewer distinct answers in the data set, so a skew towards a q-matrix predicted IDR could cause this difference in performance.

In our comparison of factor analysis with the q-matrix method we found that the number of distinct answer vectors determined the performance difference between those 2 methods. We therefore plotted the difference in performance for cluster and q-matrix analysis versus the number of distinct answer vectors in Figure 4-5. However, the pattern of differences between the two methods does not seem to vary with the number of distinct answers in the data set.

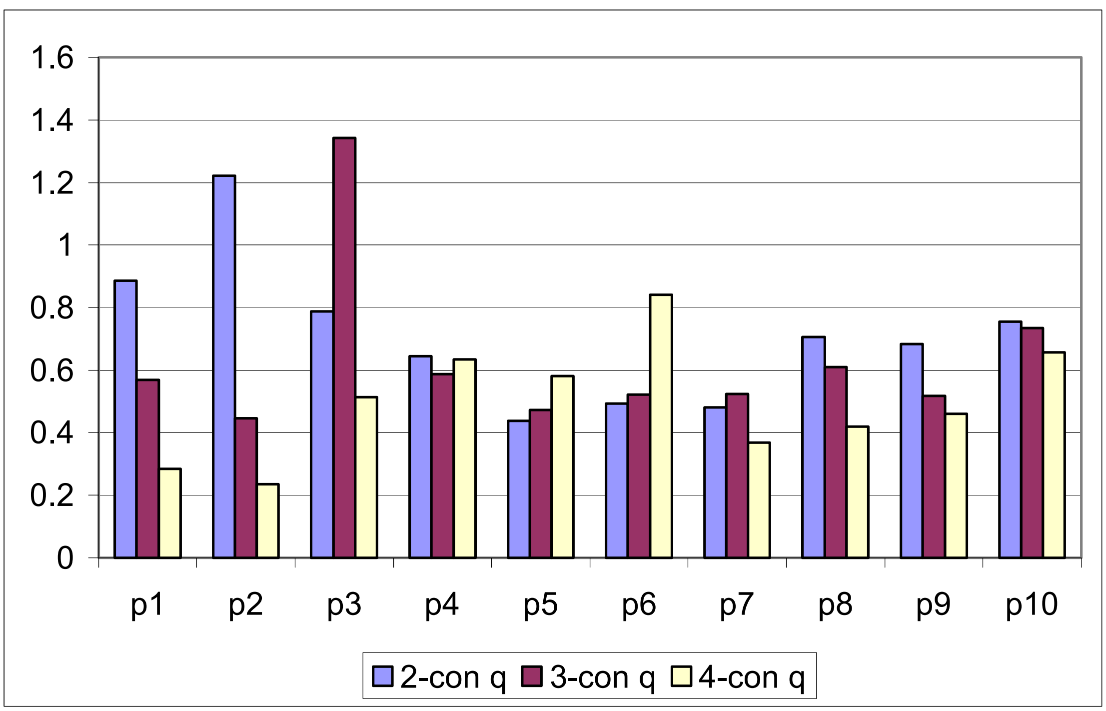


Figure 4-4 Ratio of cluster over q-matrix error for 2,3, & 4 concept q-matrices

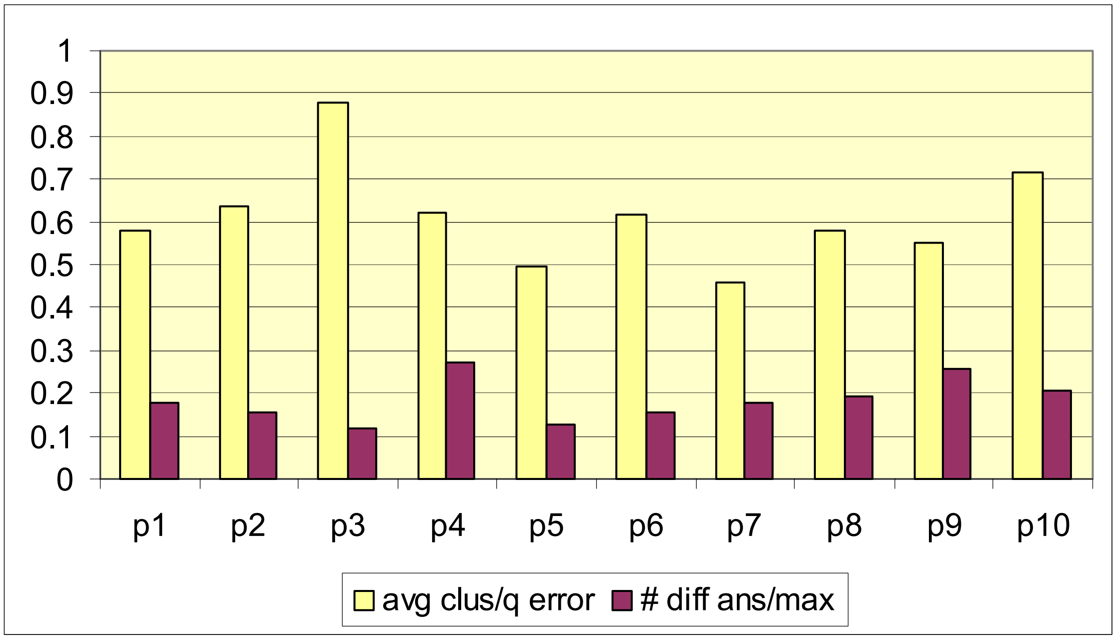


Figure 4-5 Average ratio of cluster/q-matrix error v. relative number of distinct observations

Since cluster analysis does, in general, perform better as measured by the total error summed over all students, we may conclude that cluster analysis would be a better tool for student knowledge assessment. However, this conclusion assumes that the error here relates well to truly understanding what it is that a student has learned. In the q-matrix method, we have deduced a student knowledge state, whereas in cluster analysis we have only a "cluster seed" that we compare each student response to. When guiding students through further remediation, what good would this cluster seed do us? It tells us only which questions students in this particular cluster missed in common. Using this information, we could guide students to repeat lessons on commonly missed questions. However, there are some drawbacks to this approach:

- What do we do when the cluster seed has values other than 0 or 1? In other words, when the "representative" cluster seed has a value of 0.5, do we or don't we direct a student to review this question?

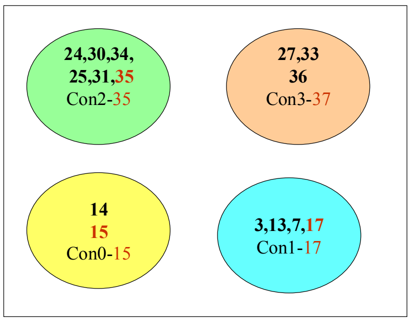
- We may not be able to understand the meaning of a cluster seed – do the students truly understand those questions all marked with 1 responses?
- We do not generate a concept model that can make it easier to diagnose student misconceptions.
- When a new student population is analyzed, cluster seeds can shift greatly since they are created as something like averages of each cluster. When this shift occurs, how do we interpret it? Students with similar response vectors can be treated quite differently after a new analysis.

In the q-matrix method, we can hold a q-matrix constant when analyzing a new student population. Both the cluster analysis and the q-matrix method may have convergence problems when analyzing new data but trying to preserve an old model. However, with the q-matrix method we have additional states available in a model that may not already be used, allowing a fixed q-matrix model more flexibility than a fixed cluster analysis model. In addition, the q-matrix model has offered us a way to interpret the relationships among questions in a tutorial, where cluster analysis does not.

4.3.2 Graphical comparison of q-matrix concept states and clusters

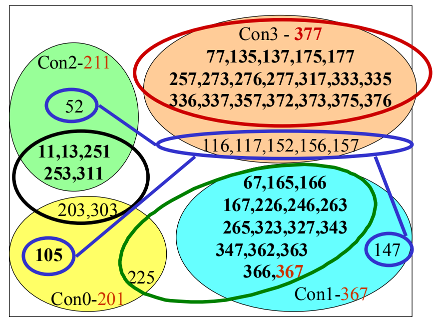
From the concept states and cluster analyses we also created a graphical representation for the clusters to determine the overlap in these two procedures. These graphs often demonstrate that the two procedures keep the same groups of responses together.

In each of these diagrams, we demonstrate the overlap in "clustering" between the q-matrix analysis and a SAS cluster analysis. To produce these diagrams, a q-matrix analysis for 2 concepts was run on each experiment. Then, each response was assigned to a concept state based on the q-matrix. These assignments are demonstrated in the diagrams by 4 colored circles. For any 2-concept q-matrix model, there are up to 4 concept states: 0,1,2, and 3. In each diagram, the concept states are labeled by Con0, Con1, Con2, and Con3, and after each of these we list the ideal response vector associated with that concept state. (These responses are in octal reverse; e.g. 10111 = 35 in octal reverse).



Binary Relations Section 1 Q-matrix Clusters none diff

Figure 4-6 Binary sec. 1 clusters



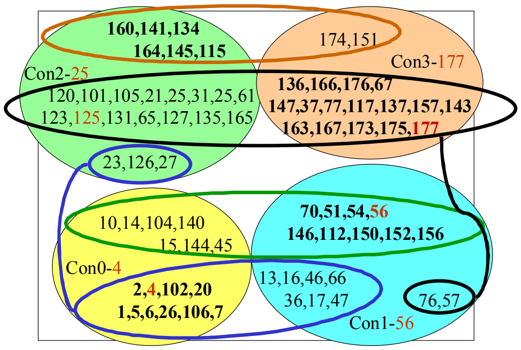
Binary Relations Section 2 Q-matrix Clusters 10/51 diff

Figure 4-7 Binary sec. 2 clusters

For Proof 1, only 3 concept states are used. For each experiment, we ran a SAS fastclus cluster analysis to place responses in clusters (3 clusters for Proof 1; 4 for the rest). Circles overlaid on the 4 q-matrix concept states show these clusters. For the Binary Relations Section 1 tutorial, the q-matrix and fastclus cluster assignments correspond exactly, as we can see in Figure 4-6. Section 2 contained many more responses, and its cluster diagram, given in Figure 4-7, is therefore more complex. However, this diagram shows that although the clusters for each method do not correspond exactly, large groups of responses were in the same groups in each method.

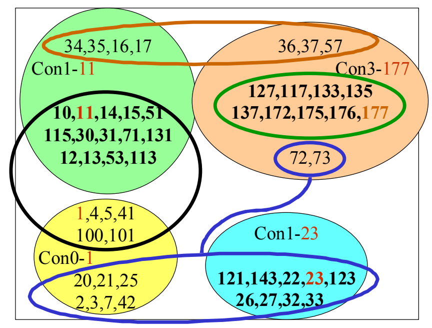
To obtain a mathematical measure of the overlap of these groups, we mapped each concept state to a cluster based on the highest overlap, and these matches are shown in bold in the diagram. Those responses that are not considered to be in "matching" concept states/clusters are in normal typeface. For example, in the Section 2 diagram in

Figure 4-7, in concept state Con1, responses 67-367 are bolded and match with one cluster. The other response in this cluster is 225, which is assigned to concept state Con0, so it is not shown in bold. Likewise, response 147 is in concept state Con1 but is not in this cluster, so it is also not in bold typeface.



Binary Relations Section 3 Q-matrix Clusters 36/77 diff

Figure 4-8 Binary sec. 3 clusters



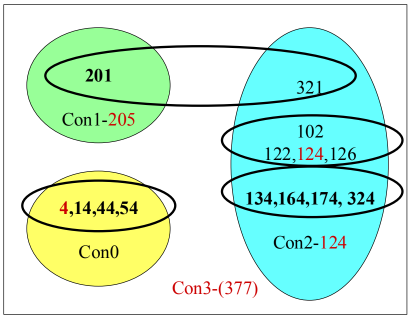
Count 1 Q-matrix Clusters 22/54 diff

Figure 4-9 Count clusters

In most diagrams, there does seem to be a significant overlap between at least 3 clusters and three concept states. Even in partitions of the largest response sets, large groups of responses are kept together in states. For example, in the Count clusters shown in Figure 4-9, three clusters corresponding to concept states 1, 2, and 3 strongly overlap.

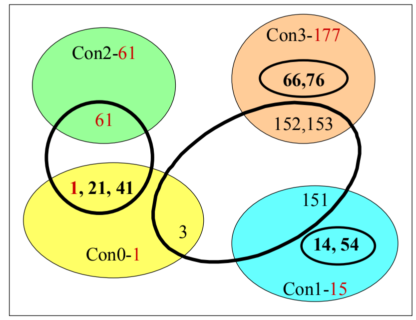
In Figure 4-20 we have graphed the ratio of different cluster assignments to the total number of responses for each experiment. This graph demonstrates that less than half of the responses were placed in different clusters for each experiment, and in some cases a very small ratio of the total were placed in different clusters. These are: Binary Relations Sections 1 and 2, and Proof 3. For these problems, the ideal response vectors were probably not so different from the cluster medians found in the cluster analysis.

We did not expect these graphs to have total overlap, simply to show that groups of similar responses are kept together in both q-matrix and cluster analysis. In this, we wished to show that the q-matrix method does keep similar responses together. We can



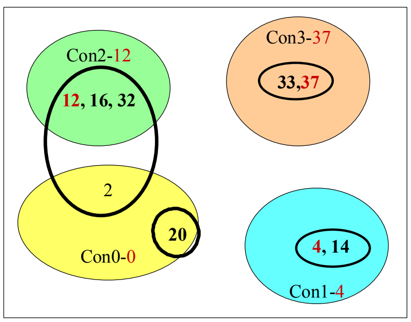
Proof 1 Q-matrix Clusters 5/14 diff

Figure 4-10 Proof 1 clusters



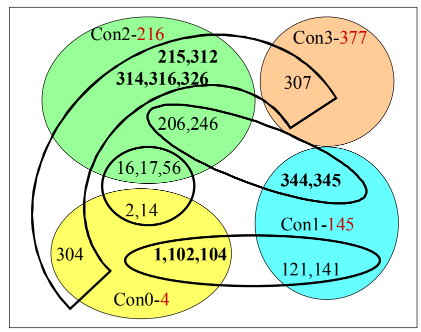
Proof 2 Q-matrix Clusters 5/11 classified diff

Figure 4-11 Proof 2 clusters



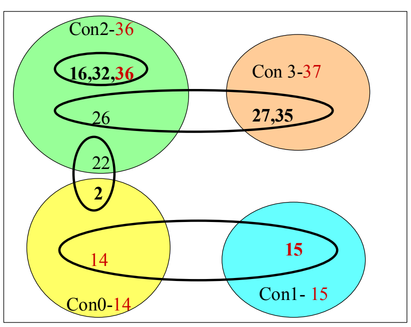
Proof 3 Q-matrix Clusters 1/9 diff

Figure 4-12 Proof 3 clusters



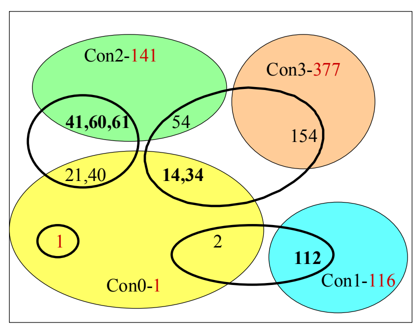
Proof 4 Q-matrix Clusters 9/20 in diff clus

Figure 4-13 Proof 4 clusters



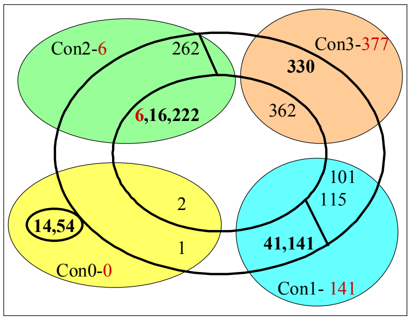
Proof 5 Q-matrix Clusters 3/10 clus diff

Figure 4-14 Proof 5 clusters



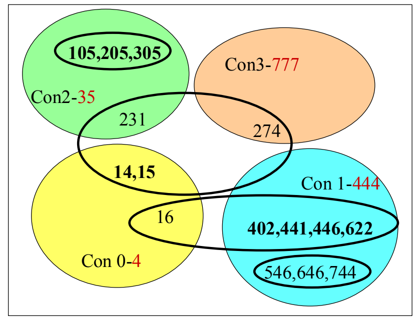
Proof 6 Q-matrix Clusters 6/12 clus diff

Figure 4-15 Proof 6 clusters



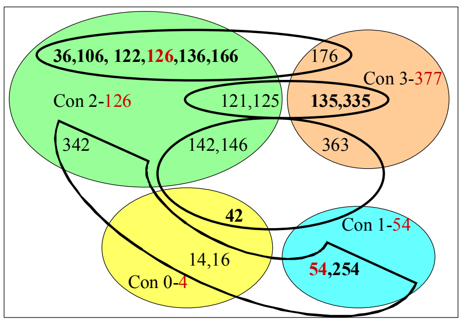
Proof 7 Q-matrix Clusters 6/14 clus diff

Figure 4-16 Proof 7 clusters



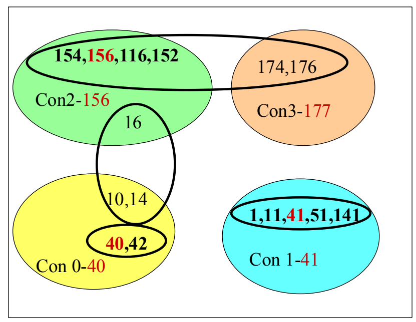
Proof 8 Q-matrix Clusters 6/15 clus diff

Figure 4-17 Proof 8 clusters



Proof 9 Q-matrix Clusters 9/20 diff

Figure 4-18 Proof 9 clusters



Proof 10 Q-matrix Clusters 5/16 clus diff

Figure 4-19 Proof 10 clusters

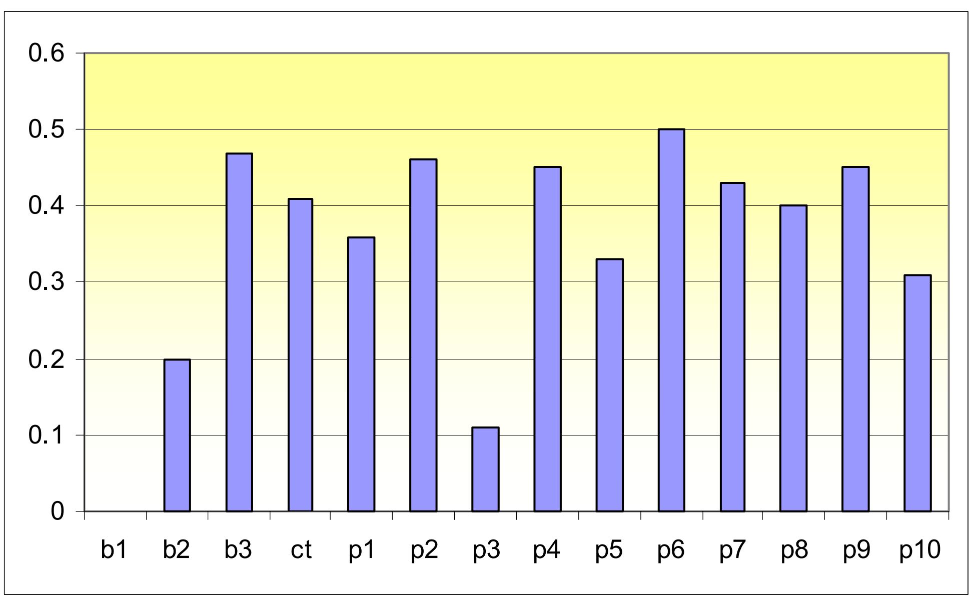


Figure 4-20 Ratio of different/total cluster assignments

explain the differences in these two methods through the nature of the clusters we extract. In the q-matrix method, the clusters are called concept states, and these states can be modeled with a matrix where the actual number of concepts is approximately the log of the number of clusters. Sometimes, two responses that are close together in distance might be assigned to two different concept states. In this case, we believe that these responses were separated into two separate states because the responses in which they differ are important in distinguishing concept states.

We could use a cluster analysis with k resulting clusters to determine k concepts, with "ideal response vectors" of the cluster medians. Then, when a student was assigned to a cluster, these medians could be used to decide which questions a student needed to review. However, this has several drawbacks. First, the number of clusters is much larger than the number of concepts in the q-matrix method. Second, we would have to choose a method to determine the number of clusters to extract. Third, it might require an expert analysis to determine the next step to take given a cluster and its median. The q-matrix method allows us to determine our next step automatically.

In the case of large data sets, if the q-matrix method is too costly to run, a fast cluster analysis could be used to first group the data set into clusters, and the medians of these clusters could be analyzed to determine possible ideal response vectors for the data.

4.3.3 Q-matrix concept states as one step of a cluster analysis

Finally, we treat the q-matrix analysis as one step in a cluster analysis to determine how close to cluster "convergence" we might be. This comparison may be more fair than our other comparisons, since several different cluster groupings could conceivably have a similar error pattern for the students. Table 4-2 demonstrates the evaluation of the q-matrix method as a step in a cluster analysis for each of the Proofs tutorial sections. In the first column we list the experiment. In the second column, we list "Yes" if the q-matrix extracted using the q-matrix method meets the convergence criteria for a cluster analysis, and "No" if it does not. Four of the 10 q-matrices tried met the convergence criteria for a cluster analysis. Of the remaining six, 4 of these converged after one more iteration of a cluster analysis, as we can see in the third column. The 2 remaining q-matrices for Proof 5 and Proof 6 converged after 2 more iterations of a cluster analysis. In the last column, we list the error over all students for each q-matrix before and after cluster convergence.

For this analysis, a k-means cluster analysis was performed using SAS. In a k-medians cluster analysis, k seeds are iteratively chosen for the k clusters in an analysis. The first set of seeds is chosen at random from the data set. Then, each data point is assigned to the cluster whose seed is closest. Then, for each cluster, the median of the new cluster is then taken as the new seed and the process is repeated. We reach convergence when the medians do not change significantly [see SAS help for a description of the stopping criteria].

Table 4-2 Convergence of q-matrix "clusters" using cluster criterion

Experiment	Converged?	Converged on next?
Proof 1	No	Yes
Proof 2	Yes	-
Proof 3	Yes	-
Proof 4	No	Yes
Proof 5	No	No, +2 iterations
Proof 6	No	No, +2 iterations
Proof 7	No	Yes
Proof 8	Yes	-
Proof 9	No	Yes
Proof 10	Yes	-

These results indicate that the clusters formed by a q-matrix analysis are similar to those formed by a cluster analysis. The reason that not all of the cluster formations created by the q-matrix analysis meet the convergence criterion is probably that, for the q-matrix clusters, there are relationships imposed on the clusters created so that the q-matrix model can explain the clusters using approximately the log base 2 of the number of clusters for its number of concepts. The advantage we gain by these relationships, along with the fact that the clusters created by q-matrices are “close” to the results of a converged cluster analysis, outweighs the difference in error for the two models. In addition, the extra generality of the q-matrix model, which may offer more concept states than are actually used, probably allows an advantage for the q-matrix model over the cluster analysis model when applying these models statically to new data.

4.4 Discriminant analysis

Discriminant analysis is one form of data mining that assumes each observation in a data set has been classified into categories, and the analysis is used to determine which of the observed variables has the greatest effect on the classifications. For example, if we were to track high school students using all types of data including socioeconomic status, family income, high school grades, et cetera, and then recorded whether or not a student went to college, we could analyze these data to find any factors that might have the greatest influence on whether a student went to college. We could use this analysis to identify the factors that we might want to target in a program designed to improve the numbers of high school graduates who went on to attend college.

In the case of analyzing student responses to questions, we do not have a priori knowledge of the classification of different students. We could classify students based on percentage scores an dividing the class into A, B, C, D and F students, and run a discriminant analysis to determine which minimal set of questions would best predict the grade a student would get. However, with the data available to us in a computer tutorial, we do not have pre-classifications of students. Therefore, we chose not to compare the q-matrix method with discriminant analysis.

4.5 Summary

As a data mining method, we can see that the q-matrix method has several advantages over other types of data mining, including factor analysis, cluster analysis, and discriminant analysis. The q-matrix method builds an interpretable model for data, without a priori knowledge of concepts or clusters needed for a group of students, in a fully automated process. It needs much less data for acceptable performance than methods such as factor analysis that are based on a covariance analysis. This is important when using the q-matrix method for data mining student knowledge states, where classes of students number in the tens and hundreds instead of the thousands.

Cluster analysis can be used to group students into groups with lower response prediction errors, but the restriction of this method is that it may not apply well to a new group of students, and does not allow for a clear interpretation of knowledge model. In addition, the number of clusters for analysis must be pre-determined, or a method of convergence must be chosen before analysis. These drawbacks will make it necessary to run a cluster analysis for each class and each tutorial while changing parameters to suit each problem. The q-matrix method can be set to choose the number of concepts automatically, based either on a maximum error per student or by comparing the relative change in error as the number of concepts increases.

In finding the appropriate number of concepts or clusters, the q-matrix and cluster analysis methods are somewhat similar. However, cluster analysis does not offer the benefit of cluster interpretation as a q-matrix does. A q-matrix is a more abstract model of the concepts used in the tutorial. Although the creation of the q-matrix model is data-dependent, the concept model created always leaves 2^c concept states for students. A given q-matrix analysis with c concepts may not use all of these states, but they are still available for use when applying a q-matrix to a new set of student data. Cluster analysis will also be even more sensitive than the q-matrix method to student sets that are strongly skewed toward understanding some concepts and not understanding others.

5 Knowledge Assessment

Originally, q-matrices were invented as simple mathematical tools to compare student performance with pre-conceived concepts that experts determined were related to solving the given problems. In later research, it was postulated that a q-matrix could be created from student responses, to get a better match to actual student performance. In his research, Patrick Brewer created simulated students to test this idea for feasibility [Bre96]. He found that, in fact, with a given original q-matrix and student states, the original q-matrix could be recovered from simulated student data with added noise. Jennifer Sellers and Susan Jones both applied the q-matrix method to small groups of students [Sel98 & Jon96].

In this research, we apply the q-matrix method to a larger group of students and analyze the results in several ways. This chapter examines the effects of the Binary Relations tutorial on student grades, compares q-matrices with expert models, and compares student and automated knowledge remediation paths. First, we investigate whether the Binary Relations tutorial scores predict final exam grades in Section 5.1. In Section 5.2 we compare expert-created q-matrices with extracted q-matrices, and analyze the extracted q-matrices for interpretability. Third, we determine if q-matrices could be used for the purpose of student remediation, choosing which question(s) students might need to review, and these results are presented in Section 5.3.

5.1 Comparison of Binary Relations tutorial scores with student grades

During the course of the Fall 2002 semester, students were required to take all three tutorials and a final exam. We hypothesized that taking the tutorials would have a positive effect on students' final exam scores in those topics. To determine this effect, we compared students' Binary Relations tutorial scores with their scores on the final exam. A chi-squared analysis for each of the questions in common between the tutorial and the final exam resulted in chi-squared values of less than 1, so we were unable to reject the null hypothesis that the scores that occurred on each test question and tutorial were unrelated. In this analysis, a table was made for each question, crossing B and –B with E and –E, where B represents answering the question correctly on the tutorial, and E represents answering the question correctly on the final exam.

A conditional probability analysis was done for all students on the three sections of the binary relations tutorial to test whether or not their tutorial scores would predict their performance on the final exam. The results of that analysis are given in Table 5-1,

Table 5-2, and Table 5-3. In each table, questions that were asked in both the tutorial and the final exam for the fall 2002 CSC 226 course are listed according to their number in the tutorial. In these tables, E denotes that a student answered a question correctly on the exam, while –E is its logical opposite. B denotes that a student answered the question correctly during the binary relations tutorial, and –B denotes that they missed the question on the tutorial. P(E|B) denotes the probability that a particular question was answered correctly on the final exam (E), given that the corresponding question was answered correctly in the binary relations tutorial (B).

Table 5-1 Binary Relations Tutorial Section 1 Comparison with Student Exam Grades

	q1.2	**q1.4**	**q1.5**
P(E)	0.95	0.929	0.721
P(E|B)	0.95	0.923	0.71
P(E|-B)	0.923	1	0.767
Abs. Difference	0.027	0.08	0.06
SE (P(E|B)-P(E|-B))	0.271	0.289	0.15
P(-E|B)	0.05	0.077	0.29
P(-E|-B)	0.077	0	0.233
Abs. Difference	0.027	0.08	0.06
SE (P(-E|B)-P(-E|-B))	0.018	0	0.053

Table 5-2 Binary Relations Tutorial Section 2 Comparison with Student Exam Grades

	q2.1	q2.2	q2.3	q2.4	q2.5	q2.6	Q2.7	q2.8
P(E)	0.81	0.83	0.73	0.80	0.76	0.56	0.98	0.95
P(E|B) (a)	0.83	0.83	0.72	0.82	0.75	0.6	0.98	0.95
P(E|-B) (b)	0.63	0.75	0.8	0.75	0.8	0.37	1	0.94
Abs. Diff. |a-b| (1)	0.2	0.08	0.1	0.07	0	0.23	0	0.01
Std. Error for (1)	0.19	0.24	0.21	0.16	0.16	0.1	0.27	0.18
P(-E|B) (c)	0.17	0.17	0.28	0.18	0.25	0.4	0.02	0.05
P(-E|-B) (d)	0.38	0.35	0.3	0.35	0.2	0.63	0	0.06
Abs. Diff. |c-d| (2)	0.2	0.08	0.1	0.07	0	0.23	0	0.01
Std. Error for (2)	0.07	0.06	0.06	0.04	0.05	0.11	0	0.01

Table 5-3 Binary Relations Tutorial Section 3 Comparison with Student Exam Grades

	q3.1	q3.2	Q3.3	q3.6	q3.7
P(E)	0.674	0.851	0.865	0.809	0.716
P(E|B) (a)	0.674	0.854	0.842	0.819	0.708
P(E|-B) (b)	0.674	0.846	0.963	0.787	0.733
Abs. Diff. |a-b| (1)	0	0.008	0.12	0.032	0.03
Std. Error for (1)	0.121	0.148	0.193	0.143	0.13
P(-E|B) (c)	0.326	0.146	0.158	0.181	0.292
P(-E|-B) (d)	0.326	0.154	0.037	0.213	0.267
Abs. Diff. |c-d| (2)	0	0.008	0.12	0.032	0.03
Std. Error for (2)	0.059	0.026	0.016	0.035	0.05

Our hypothesis was that a student who answered a question correctly on the tutorial would be more likely to answer its corresponding question correctly on the final exam, in other words, that P(E|B) would be high and P(-E|B) would be low. In addition, these probabilities would need to be significantly different, respectively, from P(E|-B) and P(-E|-B). To test this hypothesis, we calculated each conditional probability in the tables below. We also computed the standard error (SE) for the differences P(E|B)-P(E|-B) and P(E|-B)-P(-E|-B). In order for these differences to be significant, these differences should be greater than twice their corresponding standard errors.

Standard error for the difference in two conditional probabilities is as follows:

$$SE(|\, p(a\,|\,b) - (p(a\,|\,-b)\,|) = \sqrt{p(a\,|\,b) * p(a\,|\,-b) * (1/w + 1/z)}$$

where w= (the number of occurrences of b) and z= (the number of occurrences of –b).

Each absolute difference |P(E|B)-P(E|-B)| and |P(-E|B) - P(-E|-B)| was computed and compared with the standard error for that difference. In all cases except questions 2.1, 2.6, and 3.3, these differences were not found to be significant.

For question 2.1, |P(-E|B) - P(-E|-B)| was 0.2, while the standard error of the difference was 0.07. Since the difference was more than twice the standard error, we can conclude that a significant dependence exists between -E and -B for this question. Students were much more likely to miss the final exam question in this case if they had also missed the tutorial question for it. This question asks whether a particular binary relation is reflexive. The relationship between -E and -B implies here that if a student did not understand this question in the tutorial, they were more likely to miss this question on the final exam as well.

Question 2.6 asks whether a given binary relation is transitive. This is notoriously the most difficult property for students to determine for a given binary relation. Our analysis shows that dependencies exist between both E with B and -E with -B. The absolute differences |P(E|B)-P(E|-B)| and P(-E|B) - P(-E|-B)| are both 0.23, while their standard errors are 0.1 and 0.11, respectively. Since these differences are each more than twice their respective standard errors, we can conclude that each of these differences is significant. In other words, students who answered question 2.6 correctly on the tutorial were more likely to answer its corresponding exam question correctly when compared to students who missed this question on the tutorial. Likewise, students who missed question 2.6 in the tutorial were more likely to miss this question on the exam as well.

Question 3.3 is about minimal elements in Hasse diagrams. In this case, P(-E|B) is larger than P(-E|-B), indicating that more students who answered this question correctly on the binary relations tutorial answered this question incorrectly on the final exam. This is the opposite relationship from what we might expect. However, there are several factors that can explain this occurrence. The Hasse diagrams given in the tutorial were drawn by the tutorial program, while on the final exam, the students were required to draw their own Hasse diagrams. The tutorial almost always lines up all the minimal elements in a row on the bottom of the diagram, which assists in determining which elements are minimal. When drawing their own diagrams, students are not as careful, and may miss seeing minimal elements when they draw them in places other than the very bottom of their pictures. Another explanation may be that, students who missed this question on the tutorial reviewed this question again until they understood the concept better, so that it was unlikely that they would miss this question on the exam.

For each question, P(E), the probability that any student answered the question correctly on the final exam, is also given in the tables for each section of the binary relations tutorial. The finding that there are no significant dependencies between the tutorial answers and the final exam answers can also be seen by examining the differences between P(E), P(E|B), and P(E|-B). For all questions but 2.1 and 2.6, these probabilities are all similar. For questions 2.1 and 2.6, these probabilities are not within 0.1 of one another, as they are for the other questions, further supporting that exam performance and tutorial performance are related for these two questions.

To better determine the strength of the relationship between tutorial and exam scores, a correlation analysis was run for each tutorial section. As expected from the probability analysis, the correlations are low (below 0.1) for all questions except 2.1, 2.6, and 3.3. Question 2.6, with a correlation of 0.28 between the tutorial and final exam scores, has the strongest relationship, while the other two are weaker.

Table 5-4 Correlation of Binary Relations Tutorial Section 1 Scores with Final Exam Scores

	q1.2	q1.4	q1.5
Correlation	0.03	-0.08	-0.05

Table 5-5 Correlation of Binary Relations Tutorial Section 2 Scores with Final Exam Scores

	q2.1	q2.2	q2.3	q2.4	q2.5	q2.6	q2.7	q2.8
Correlation	0.16	0.06	-0.05	0.07	-0.04	0.28	-0.05	0.01

Table 5-6 Binary Relations Tutorial Section 3 Comparison with Student Exam Grades

	q3.1	q3.2	q3.3	q3.6	q3.7
Correlation	0	0.01	-0.14	0.04	-0.03

This analysis shows that there is no correlation between tutorial and exam scores for the binary relations tutorial, except for questions 2.1, 2.6, and 3.3. Questions 2.1 and 2.6 demonstrated a positive correlation, suggesting that correct answers on these questions in the tutorial predict correct answers on these final exam questions. Question 3.3 showed a negative correlation, and our probability analysis suggests that those students who missed this question on the tutorial were more likely to get this question correct on the final exam.

The lack of correlation between the tutorial and final exam scores can be understood by examining the circumstances of this class and final exam. In general, the students performed well on the final exam, leaving little variance for us to account for in a correlation analysis. There was a time lapse of several weeks between students taking the tutorial and taking the final exam. In the interim, several confounding factors can influence a student's exam score, including the amount each student studied, and the resources a particular student used to study the given topic.

5.2 Comparison of model with expert analysis

Jennifer Sellers [Sel96] designed the binary relations tutorial and conducted an experiment to determine if the q-matrix extracted from student responses corresponded well with expert-created q-matrices. With her small sample of students, she found that Sections 1 and 2 corresponded well with expert analysis but that section 3 did not. We further tested the hypotheses that the q-matrices formed through analyzing student responses will be interpretable and perhaps also comparable to an expert analysis of the subject. In this section we compare the three q-matrices extracted for the Binary Relations tutorial in Fall 2002 with the expert q-matrices generated in Sellers' experiment. (See Chapter 7 for the interpretation of Proofs q-matrices).

5.2.1 Expert v. extracted q-matrices for the binary relations tutorial section 1

Section 1 of the binary relations tutorial covers Cartesian products and binary relations in general. The topics in this section are listed in Table 5-7. In this section we will discuss how experts related these topics (and their cquestions) by concepts, and compare this to the q-matrix extracted from the fall 2002 discrete math class.

Table 5-7 Topics in Binary Relations Tutorial, Section 1

Section 1	**Cartesian Products & Binary Relations**
Question	**Topic**
q1.1	Cartesian product (one point)
q1.2	Cartesian product (all points)
q1.3	Relations as subsets of Cartesian products
q1.4	Matrix representations of relations
q1.5	Composite of two relations

Table 5-8 lists the Section 1 matrix extracted by the q-matrix method both in Sellers' work and in our fall 2002 experiment. Table 5-9 lists the Section 1 matrix created by 3 instructors in Sellers' experiment. Concept 1 in Table 5-8, the extracted q-matrix, corresponds to concept 3 in Table 5-9, the expert q-matrix. The extracted q-matrix tells us several things: most students answered questions 1-4 correctly, because their predicted answers to those questions would be correct answers. Students found question 1.5, that on composite relations, the most difficult. This corresponds with the expert analysis of this tutorial. It is no surprise that the q-matrix method did not find the expert concept 1, since this concept consists of a relationship to all the tutorial questions. In other words, this concept was the over-arching concept that relates all questions in this tutorial, so most students would understand the main concept by the time they took the quiz, and their answers would not reflect this relationship among all the questions. Concept 2 in the expert matrix relates those questions dealing with relations, which are subsets of Cartesian products. In this tutorial, there was not enough difference in student responses to extract this concept.

Table 5-8 Binary Relations Section 1 Extracted q-matrix

	q1.1	q1.2	q1.3	q1.4	q1.5
con 1	0	0	0	0	1

Table 5-9 Binary Relations Section 1 Expert q-matrix

	q1.1	q1.2	q1.3	q1.4	q1.5	Description
con 1	1	1	1	1	1	Cartesian Products
con 2	0	0	1	1	1	Relations
con 3	0	0	0	0	1	Composites

Concept 1 represents question 5, which is composition of relations. This is a difficult concept for most students to understand. This q-matrix agrees with that determined for Section 1 in Seller's thesis, both by q-matrix analysis and by expert analysis.

5.2.2 Expert v. extracted q-matrices for the binary relations tutorial section 2

The topics taught in Section 2 of the Binary Relations tutorial are given in Table 5-10. This section gives definitions and examples of the properties of binary relations.

In this section we discuss how experts related these topics by concepts, and compare this to the q-matrix extracted from the fall 2002 discrete math class.

Table 5-10 Topics in Binary Relations Tutorial Section 2

Section 2	**Properties of Binary Relations**
Question	**Topic**
q2.1	Reflexive
q2.2	Symmetric
q2.3	Irreflexive
q2.4	Antisymmetric
q2.5	Asymmetric
q2.6	Transitive
q2.7	Equivalence relation
q2.8	Partially-ordered set (poset)

Jennifer Sellers compared her extracted Q-matrices to those constructed by instructors. For two of her Q-matrices, expert and extracted Q-matrices corresponded in all but one matrix value. In Table 3, element (2,6) shows the only difference between the expert and extracted (auto) Q-matrices for one set of questions. In this case, instructors indicated that concept 2 indicated the application of more than one definition, and had a zero in position (2,6) since it corresponds to applying the definition of a transitive relation. However, the extracted q-matrix, with a one in that position, reflects that students find applying this definition much more complex than applying any combination of two other properties of binary relations. Upon a second look, experts agreed that this extracted q-matrix was appropriate.

Table 5-11 Binary Relations Section 2 Sellers' Expert v. Extracted Q-matrices

	q2.1	**q2.2**	**q2.3**	**q2.4**	**q2.5**	**q2.6**	**q2.7**	**q2.8**	**Expert Description**
Con 1	1	1	1	1	1	1	1	1	Special properties of relations
Con 2	0	0	0	0	1	exp 0/ auto 1	1	1	Multiple properties

In the analysis of fall 2002 data, we extracted a q-matrix with 4 concepts. In Seller's work, with only 17 students, it was more appropriate to extract a q-matrix with fewer concepts. With 2 concepts, we have 4 concept states available and expect about 4 students to be assigned to each concept state. With so few students, even a 1-concept q-matrix might have been more appropriate for her data. With 142 students, the fall 2002 data can support a more complex model. With 4 concepts, there are 16 concept states available, so we'd expect an average concept state to contain about 8-9 students each. If we had used 3 concepts, we'd expect about 18 students in each of 8 states, perhaps a bit high for our purposes.

The 4-concept q-matrix for fall 2002 is given below in Table 5-12. This q-matrix is quite different from those given by both experts and Seller's previous analysis. In this matrix, questions on antisymmetric (q2.4), equivalence relations (q2.7), and partially ordered sets (q2.8) were all partitioned into separate concepts. These questions were the biggest separators among students, since 74 out of 142 students missed 1 question. Out of the remaining students, 39 missed only two questions. This skew towards missing only one question caused the q-matrix analysis to work harder to separate these large groups of students into groups with the lowest error. This resulted in the simple separation by the questions causing the most single errors.

In Table 5-12, concept 1 relates questions on irreflexive (q2.3), asymmetric (q2.5), and transitive (q2.6) properties of binary relations. Out of the properties given in this section, asymmetric and irreflexive are the two properties whose definitions contain negations. These are notorious for causing student difficulty. However, these two properties are relatively easy to check in a matrix. Transitivity, on the other hand, has a more complex definition, and is also difficult to check for in a matrix.

It makes sense for the irreflexive and asymmetric properties to be paired, since an asymmetric relation must be both antisymmetric and irreflexive. Antisymmetric (q2.4) was not paired with these since it was grouped on its own to explain students who missed this as the only question they answered incorrectly. The transitive property (q2.5) was probably grouped with irreflexive and asymmetric since, if a student missed more than one question, it was probably in this group of three questions.

Concept 2 is only for the antisymmetric property, which causes the most difficulty for students in remembering and applying its definition. Concept 3 represents only equivalence relations, and concept 4 represents only partially ordered sets. Answering these questions on a matrix is much more complex than the other properties since each of these requires the student to check for three properties. Equivalence relations must be reflexive, symmetric, and transitive. Partially ordered sets must be reflexive, antisymmetric, and transitive. The difficulty in checking each of these usually lies in checking for transitivity. In addition, it is easy to make a mistake in checking for so many properties in one question.

Table 5-12 Binary Relations Section 2 Fall 2002 Extracted q-matrix, 4 concepts, Err/stud: 0.74

	q2.1	q2.2	q2.3	q2.4	q2.5	q2.6	q2.7	q2.8
con 1	0	0	1	0	1	1	0	0
con 2	0	0	0	1	0	0	0	0
con 3	0	0	0	0	0	0	1	0
con 4	0	0	0	0	0	0	0	1

We have also listed the q-matrices for Section 2 fall 2002 data with 3 concepts in Table 5-13, and for 2 concepts in Table 5-14. As we stated before, the 3-concept model for this problem would place about 18 students in each concept state, and we'd expect about 36 students per concept state for the 2-concept model. As we see below, the errors for each of these are more than 1 per student. The two-concept q-matrix does not correspond to the 2-concept expert q-matrix given in Table 5-11.

As we can see by comparing Table 5-14 with Table 5-12, concepts 1 and 4 in the 4-concept q-matrix correspond exactly with the concepts in the 2-concept q-matrix. Although each run of the q-matrix method starts with random values, it is often the case that q-matrices with increasing numbers of concepts have similar or even identical concepts. Similarly, concept 3 in the 3-concept q-matrix appears again in the 4-concept model as concept 2.

Table 5-13 Binary Relations Section 2 Fall 2002 Extracted q-matrix, 3 concepts, Err/stud: 1.11

	q2.1	q2.2	q2.3	q2.4	q2.5	q2.6	q2.7	q2.8
con 1	0	0	0	0	1	1	0	1
con 2	0	0	1	0	0	1	1	0
con 3	0	0	0	1	0	0	0	0

Table 5-14 Binary Relations Section 2 Fall 2002 Extracted q-matrix, 2 concepts, Err/stud: 1.27

	q2.1	q2.2	q2.3	q2.4	q2.5	Q2.6	q2.7	q2.8
con 1	0	0	0	0	0	0	0	1
con 2	0	0	1	0	1	1	0	0

The recurring nature of concepts in increasing q-matrix models indicates that the q-matrix models created are fairly robust. In addition, although the q-matrix method is a heuristic hill-climbing method, running the q-matrix analysis again has almost always returned the same q-matrix, even though the q-matrices are started with random values each time. This is another indication that the q-matrix method is a robust algorithm.

5.2.3 Expert v. extracted q-matrices for the binary relations tutorial section 3

Section 3 of the binary relations tutorial covers Hasse diagrams and properties of these diagrams. The topics in this section are listed in Table 5-15. In this section we will discuss how experts related these topics (and their corresponding questions) by concepts, and compare this to the q-matrix extracted from the fall 2002 discrete math class.

Table 5-15 Topics in Binary Relations Tutorial Section 3

Section 3	**Posets and Hasse Diagrams**
Question	**Topic**
q3.1	Hasse Diagrams
q3.2	Maximal elements
q3.3	Minimal elements
q3.4	Upper bounds
q3.5	Lower bounds
q3.6	Least upper bound
q3.7	Greatest lower bound

The expert-generated q-matrix for the binary relations tutorial, section 3, from Sellers' work is given in Table 5-16. Instructors broke this section of 7 questions down into 3 concepts. The first concept was named "Hasse diagrams" and contained all questions, since this is the general topic of this section. The second concept grouped together questions that examined subsets of partially ordered sets for upper and lower bounds. The third concept grouped together questions relating to maximal and minimal elements. This description is as given by Sellers in her thesis. However, upon examining these tables we question whether concept 3 ought to combine q3.2 and q3.3 and concept 2 the last 4 questions, or if the general explanation of this expert q-matrix was in error. Another interpretation of these concepts, assuming that the q-matrix below is the true expert q-matrix, as given in Sellers' work, is that the last 2 questions combined the ideas of maximal and minimal elements with upper and lower bounds.

Using the q-matrix method, we would be very unlikely to extract concept 1 from this tutorial. If these questions were grouped with a significant group of tutorial questions, not all of which were based on Hasse diagrams, we might expect the q-matrix method to extract this concept because of the relative relationships among these questions in comparison to other quiz topics. When comparing this matrix to that extracted for fall 2002, we notice that q3.6 is singled out both in the expert and the extracted q-matrices. This agreement between instructors and the tutorial indicates that, as perceived by experts and by students, questions relating to lower bounds are likely to give students trouble. We also see q3.2 and q3.4 grouped together in concepts in both models. This means that more students are missing questions on maximal elements and upper bounds together, suggesting that students may have a hard time interpreting the meaning of upward movement in a Hasse diagram.

Table 5-16 Binary Relations Section 3 Expert q-matrix

	q3.1	q3.2	q3.3	q3.4	q3.5	q3.6	q3.7	Expert description
con 1	1	1	1	1	1	1	1	Hasse diagrams
con 2	0	1	1	1	1	0	0	Groups of elements
con 3	0	0	0	0	0	1	1	Max & min

194 students completed the Binary Relations Section 3 quiz, and we extracted a 3 concept q-matrix for this data. We have 8 concept states, so we expect an average of 25 students per state. This is quite a good number for our error to be less than one per student on this section, indicating that there was quite a bit of overlap in student responses. In fact, there were only 78 distinct answers on this tutorial. The q-matrix extracted from fall 2002 data for the binary relations tutorial, section 3 is given in Table 5-17. When we compare this q-matrix with the expert q-matrix in Table 5-16, we find

that concept 3 is similar in both of these – in both, q3.6 and q3.7 are related, but in the fall 2002 q-matrix, these are also related to q3.4.

In the fall 2002 q-matrix, concept 3 combines questions q3.4, q3.6, and q3.7. These questions are on upper bounds, least upper bounds, and greatest lower bounds. Concepts 2 and 3 both refer to q3.4 and q3.6, implying that these two questions, by being related to 2 concepts each, are more complex than other questions. This concept grouping would indicate to instructors that upper bounds and least upper bounds were not as well understood as the other questions. This might encourage instructors to add more practice working with these ideas, or improving these two sections of the tutorial.

In Table 5-17, concepts 1 and 4 both select one question each: concept 1 represents only q3.1 on Hasse diagrams. and concept 4 contains only q3.5 on lower bounds. This suggests that some significant groups of students had difficulty with these two questions, or that these questions were missed alone in the tutorial (i.e. no other questions were missed when a student missed one of these).

Table 5-17 Binary Relations Section 3 Fall 2002 Extracted q-matrix, 4 concepts, Err/stud: 0.72

	q3.1	q3.2	q3.3	q3.4	q3.5	q3.6	q3.7
con 1	1	0	0	0	0	0	0
con 2	0	1	0	1	0	1	0
con 3	0	0	0	1	0	1	1
con 4	0	0	0	0	1	0	0

Concept 2 combines questions q3.2, q3.4, and q3.6 into one concept. These questions are on maximal elements, upper bounds, and least upper bounds. This combination is significant, since least upper bounds depend on understanding upper bounds. We might also suppose that, due to the arrangement of Hasse diagrams, with edges leading upward, that students have more difficulty interpreting behavior in the poset moving up the diagram, as in finding a maximum or an upper bound.

5.2.4 Summary of comparison of expert and extracted q-matrices

In this section, we have compared the extracted q-matrices with expert q-matrices created for the Binary Relations tutorial. We have found that there is sometimes overlap in the expert and extracted q-matrices, but often these q-matrices do not correspond. Since the q-matrix method of extraction was designed since expert q-matrices did not correspond well with student data, we would not expect expert q-matrices to correspond particularly well with those extracted from student data. In some cases, we did find a correspondence, and this usually occurred on the most difficult or complex questions.

In addition to comparing the expert and extracted q-matrices, we also examined the extracted q-matrices to understand the relationships in the questions in this tutorial. In both Sections 2 and 3, the extracted q-matrices separated out several questions into one concept each. In this tutorial, we would expect something like this since the questions here are very much the application of definitions. Many students taking this section missed only 1 or 2 questions at a time, making a q-matrix with several 1-question concepts a good model. As we hypothesized, understanding the relationships shown in the q-matrices was not difficult, and our interpretations can be used to both understand student data and determine which questions were most difficult.

5.3 Comparison of predicted direction vs. student for remediation

In this experiment, the q-matrix method was effective in automatically choosing questions that students least understood. When compared with the choices that students made on their own, the q-matrix method chose the same question as students did for more than half the students in all three sections of the binary relations tutorial, as demonstrated in Table 5-18. Our intention was to have half of the students choose their own progress, and half to be self-guided, but concurrently-running students were often assigned the same value for this choice, so fewer students to guided their own remediation process.

Table 5-18 Student v. q-matrix Selection of Least-Understood Question

	Section 1	Section 2	Section 3
Total Students	255	251	246
Auto-guided	106	204	199
Self-guided	149	47	47
# self-guided who chose other than predicted least understood concept	11	14	20
Percentage	7.38%	29.78%	42.55%

In this experiment, random students were chosen to select the question they least understood upon completion of each section of the binary relations tutorial. Most of these students chose a question related to the q-matrix predicted "least understood concept". In Table 5-18, we see that on Section 1, only 7% of self-guided students chose to review a different question than selected by the q-matrix. Since our q-matrix was a one-concept q-matrix relating only to question 5, the most difficult on the quiz, this is not a surprising result. On section 2, 30% of self-guided students chose a different question to review than the q-matrix would have. This means that 70% of the time the q-matrix method predicted the same concept to review as students did. On section 3, however, 43% of students chose differently that the q-matrix method would have. Since this number is high, we cannot conclude that the q-matrix choice corresponds to student choices often. This result is not discouraging, though, since on a more complex topic such as this one, students may need to review more than one concept. In order to better measure the choices made by the q-matrix method, for students who chose to review questions that the system would not have chosen for them, we examine student performance on these two questions on the final exam.

Table 5-19 lists the performance on the final exam of those students who chose differently than predicted. On the final exam, these students performed equally or worse on the q-matrix predicted questions than they did on the questions they selected as those they least understood. This indicates that sometimes, a student does not realize when he should review a particular topic.

Table 5-19 Final Exam Performance for Self-Directed Students

	Section 1	Section 2	Section 3
# students with diff choices	11	14	20
# of these with finals	7	10	13
# who missed the self-directed question on the final exam	0	2	0
# who missed the q-matrix predicted question	2	2	1
# who performed the same on both	5	6	12

Based on this small sample, the q-matrix method did at least as well as the students did in choosing their "least understood question". Since we had such a small sample size, we cannot conclude more from these data. Further experiments will be needed to determine if the "least-understood question" method is truly a good educational choice. We refer to the survey data given in Chapter 6 to get the students' opinions on whether this tutorial knew what they did not understand.

5.4 Summary of Knowledge Assessment

In this chapter, we examined the results of our experiment using the Binary Relations tutorial in fall 2002. In the first section of this chapter, we compared student scores on the tutorial with their scores on the final exam. Since the final exam scores on the Binary Relations problems were, on average, quite high, we were unable to conclude much from this analysis. In addition, these questions all reside in the lower levels of

Bloom's taxonomy, relating to definitions and their applications. Since these are simple processes, we would expect students to be able to apply these definitions with ease on a take-home exam, where it is possible to look up definitions and apply them.

In the second section of this chapter, we compared extracted and expert q-matrices for each section of the Binary Relations tutorial. As we predicted, expert and extracted q-matrices did not often coincide. In this section, we also examined the extracted q-matrices for interpretability, and were able to understand student responses based on these interpretations.

In the last section of this chapter, we compared the questions that self-guided students chose to review with the questions that the q-matrix method would have chosen for them. We found that the q-matrix method often chose the same questions for review as the self-guided students chose for themselves. We also found that students who chose differently than the q-matrix method could have benefited from reviewing a q-matrix selected concept. However, due to the small sample size of self-guided students, choosing to review different questions than would have been suggested, who also took the final exam, these findings are preliminary and not conclusive.

In the next Chapter, we review the results of our tutorial surveys. In the tutorial, students were asked if the tutorial seemed to know which concepts they understood and redirected them to study those concepts again. We use these subjective results to measure some of the effectiveness of our remediation method.

6 Survey Results

In this chapter, we present the results of our surveys on the three tutorials studied in this dissertation. As discussed in the Methodology chapter, we administered the tutorials to a portion of the students in the Fall 2002 CSC 226 Discrete Mathematics course. The surveys were offered via a web-based homework administration system called WebAssign, which was developed at NC State University. Completing the tutorials as well as the surveys on the tutorials were required for student grades. The tutorial survey was developed and used by Jennifer Sellers in her master's thesis, and was added into WebAssign in the course of this experiment. WebAssign provided automated summaries of the survey data, whose results we present here. In the first section of this chapter, we list the results of the Likert-scale portions of the three surveys, and in the next three sections we discuss the free-response question results. In the final section of this chapter we summarize these survey results.

6.1 Tutorial Survey Results, Questions 1-10

The questions used on the survey are listed below. The first ten of these questions were answered on a Likert scale with 5 replies, including: Strongly Agree, Agree, Neutral, Disagree, and Strongly Disagree. The remaining four questions were free-response questions.

Table 6-1 Tutorial Survey Questions

1. I feel that I know the material with more confidence now that I have been through the tutorial.
2. Overall, I feel that taking this tutorial was a beneficial exercise.
3. I feel that this tutorial was more effective than a traditional lecture.
4. I feel that taking this tutorial was more effective than working homework problems alone.
5. I feel that taking this tutorial was more effective than working problems with others.
6. The audio lectures and slides (if available) were a valuable addition to the tutorial.
7. The examples were very important in understanding the material.
8. There were enough examples presented.
9. I felt like the program knew which concepts I did not understand, and directed me back to lessons on concepts I understood the least.
10. I would recommend this tutorial to other students in my class.
11. Was the feedback offered helpful and clear? Was there enough?
12. What aspects of the tutorial did you like least?
13. Are there any other features or improvements you would like to see included in the tutorial?
14. What aspects of the tutorial did you like best?

These questions were created to measure the effectiveness of the NovaNET tutorials in comparison with class lectures and homework, and also to determine the effectiveness of features such as the audio lectures supplied in the Binary Relations Tutorial and the remediation guidance provided to a portion of the students after they took each quiz.

The results of the first 10 questions on the surveys are given in Table 6-2. For each question, the five responses are listed in a row, and beneath these titles a percentage of the responding surveyed students answering with this response is given. There were 283 students responding to each of the three surveys administered. There were not 283 students who completed each of the tutorials; many students started but did not complete the tutorials, and a small number probably filled out the survey hoping to get credit for completing the tutorials.

Note in Table 6-2 that 14-17 percent of students did not respond to each question; this accounts for 29-48 students, bringing the total to about 240 students, much closer to those who actually did take the tutorials. The students were informed that their responses would not affect their grade; they would receive credit for taking the tutorial, so students in general did not have reason to make their responses more or less positive than they would be inclined to answer.

Table 6-2 Tutorial Survey Results for Fall 2002 Data, by Percentage of the 283 Respondents

1	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	No response
Binary	20	48	13	4	1	14
Count	17	49	12	4	1	17
Proofs	18	34	18	8	4	17
2	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	22	43	16	4	1	14
Count	23	43	11	4	3	16
Proofs	17	33	14	10	9	17
3	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	19	27	25	12	3	14
Count	21	33	20	7	2	17
Proofs	11	23	22	16	10	18
4	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	19	37	18	11	1	14
Count	13	22	25	18	5	16
Proofs	14	26	19	15	8	17
5	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	9	14	30	25	6	15
Count	13	22	25	18	5	16
Proofs	9	12	25	24	12	18
6	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	8	20	43	10	5	15
Count	5	17	50	6	4	18
Proofs	6	13	51	9	4	18
7	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	30	42	11	2	1	14
Count	28	41	9	4	2	17
Proofs	14	33	17	11	7	18
8	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	18	40	11	16	1	14
Count	17	48	10	7	1	17
Proofs	14	32	20	11	6	17
9	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	10	36	29	10	3	14
Count	14	41	17	8	3	17
Proofs	7	25	23	17	11	17
10	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**	**No response**
Binary	19	41	16	8	2	14
Count	21	39	14	5	4	17
Proofs	16	26	16	15	11	17

Question 1 asks whether student confidence is improved because of the tutorials. A plot of their responses is given in Figure 6-1. As the bar chart shows, the majority of students in all three students agreed that they were more confident in the topics after taking the tutorials. Although responses for the Proofs Tutorial were less positive, more than half of the students feel more confident in proofs after taking the tutorial. Question 2's responses were similar to those in question 1, indicating that students generally felt that taking the tutorials was a beneficial exercise.

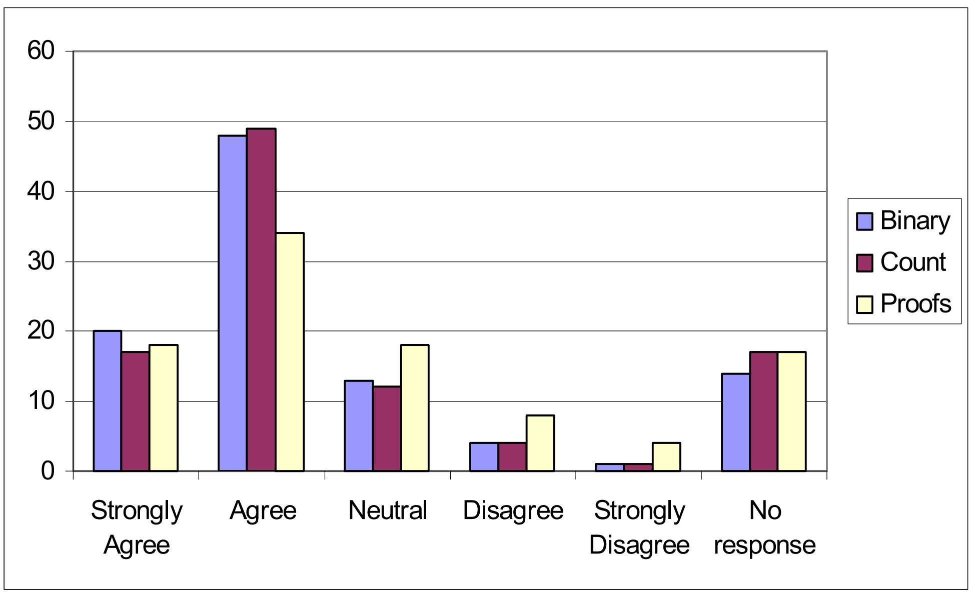


Figure 6-1 Question 1 confidence responses

Question 3 asked the students if each tutorial was more effective than a traditional lecture. As we see in Table 6-2 by adding "Strongly Agree" and "Agree" responses, 48% of students agreed that the Binary Relations tutorial was more effective, while 54% thought the Counting tutorial was. For the Proofs tutorial, 34% percent felt the tutorial was more effective, while 26% felt it was less. For all three tutorials, about 20% of the responses were neutral. In general more students agreed that the tutorial was at least as effective as a lecture.

Question 4 compares the effectiveness of the tutorials with homework. For the three tutorials, 35 to 56 percent of students agreed that the tutorials were more effective than homework. The Binary Relations tutorial was much more often judged to be more effective than homework, at 56 percent, while 35 and 40 percent of students felt the Counting and Proofs tutorials were more effective. For both of these tutorials, about 23 percent felt that homework as more effective, and about 60 percent felt that the tutorials were at least as effective as homework.

Audio lectures were only provided with the Binary Relations Tutorial, so this question only truly applies to that tutorial. In Table 6-2 we see that 28 percent of students felt that the audio lectures were a valuable addition to the tutorial, while 58 percent were neutral or had not response and 15 percent responded that they were not valuable. From these responses, we'd deduce that most students did not use the optional audio lectures, but those that did choose to use them felt that they were valuable.

Our analysis of free responses overwhelmingly showed that students appreciated having many examples available in the tutorials, and their responses to questions 7 and 8 support this finding as well. For the Binary Relations and Counting Tutorials, 58 to 65 percent of students agreed that the examples were important in understanding the material and that there were enough examples. 17 percent wanted more examples in the Binary Relations tutorial, and as we'll see later in the free response analysis, students wanted more variety in examples in both of these tutorials. The Proofs tutorial responses indicated that students weren't as satisfied with the quality and number of examples, 46-47% agreeing that they were important and were enough, but 17-18% disagreeing with those statements. In the free responses, many students asked for more examples of proof solutions and help when they were stuck on proofs.

Question 9 asks students to agree or disagree with the statement, "I felt like the program knew which concepts I did not understand, and directed me back to lessons on concepts I understood the least." Forty-six percent of students agreed that the Binary Relations tutorial seemed to know what they did not understand and directed them to study the concepts they least understood, and 55 percent of students agreed that the Counting tutorial did so. Only 11-13% of students disagreed, feeling that these tutorials did not know which concepts they least understood. This indicates that most students felt that the Binary Relations and Counting tutorials were able to determine what they understood and redirect their progress to learn more.

On question 9, 32 percent agreed and 28 percent disagreed that the Proofs tutorial knew which concepts they least understood. This is appropriate since this was more of a free-form answer program. Section 2 of the Proofs tutorial had no assessment and redirection of student responses.

Our last question on the Likert scale, question 10, asks students if they would recommend using this tutorial to other students. For both the Binary Relations and Counting tutorials, 60% of students responding would recommend this tutorial to other students, while 10% or less would not. Forty-two percent of students would recommend the Proofs tutorial, while 26% would not, showing students have more ambivalence over the Proofs tutorial. From the free responses, we can determine that this ambivalence has to do with the nature of the program, since it takes a lot of effort to enter answers and there is little help when you get stuck. In general, these survey results indicate that students found these tutorials were beneficial in learning and were at least as effective as lectures and homeworks.

The remaining questions 11-14 were free response questions. In the following sections, we will discuss the free response answers to these questions for each of the three tutorials. In the interest of brevity, we have compiled these results by classifying each response into categories and show examples of the most common types of responses.

6.2 Binary Relations Tutorial Free Response Questions

In this section we describe the results of the free response portion of the Binary Relations tutorial Survey. In general, students liked the feedback and examples offered on this tutorial, and felt that it clarified some topics for them. Students often wanted to know why their answers were correct or incorrect, as they commented in this survey. One student put the reason for this into words:

> The first section was probably the most helpful and most well done because the practice questions were a little more difficult. Making them too difficult however would just discourage people from doing this format, so it is a delicate balance.

In this tutorial, most of the questions in sections 2 and 3 were yes/no questions, where students were more able to guess the responses, whereas in section 1, students were required to enter free response answers to the questions. We believe this is the cause of this student's observation, and also explains why students needed more feedback on why certain answers were correct or not. Students also wanted a larger variety of examples in this tutorial, saying that the examples often did not cover the variety of types of problems they could expect. One student wrote, "If the matrix or Hasse diagrams are picked randomly change that and instead specifically design diagrams to test for understanding of the particular subject. Also, matrix math could be included in the tutorial." His point is taken; students need to see both positive and negative examples of each type of problem so they can determine the overall differences between them. Students often learn by examples, and the depth and breadth of your example set can determine the depth and breadth of student knowledge.

Question 11 asks, "Was the feedback offered helpful and clear? Was there enough?" Most students answered this question yes or no, with 75% of the 243 students replying to this question answering yes, that the feedback offered was helpful and clear, and that there was enough. Nineteen percent stated that there was not enough; as we discussed above, they would like to know why their answers were right or wrong.

Question 12 asked each student "What aspects of the tutorial did you like least?" About 20% of the responses were "none," and the second most popular response was that students wished to have more variety in the examples presented. Some students picked out particular topics they needed more help on even after the tutorial, including (ordered in decreasing frequency): Hasse diagrams, upper and lower bounds, transitivity, and composition. In general, students felt that Section 3 of the tutorial, on Hasse diagrams, was the most difficult, and wished for more graphics and animations, and less confusing explanations of these topics. This is understandable, since this area required the use of pictures along with matrices and relations, and was the most complex of the three sections. Students also needed more help in understanding matrix composition and transitivity, two of the most challenging subjects in Binary Relations. About 11 percent of the students felt that the interface and navigation could be improved. The NovaNET system is not difficult to navigate, but since students are used to browsers with forward and back buttons, they found the use of control keys for navigation archaic, and many of them did not read the supplemental material provided which listed the navigation commands and also how to get buttons for these commands to appear.

Survey question 13 asked, “Are there any other features or improvements you would like to see included in the tutorial?” Forty percent of students responding said no, making no suggestions for improvements. Twenty-four percent of respondents wanted more variety in the examples, with more in-depth explanations of answers, and some of these wished for more challenging questions. Twelve percent wanted improved navigation with an indicator of their progress through the tutorial, while five percent felt that NovaNET was a slow old style program, and they would prefer a web-based or other format for the tutorial. Still other students requested more graphics and animations to better explain some of the concepts, and a few students suggested more variation in the answer types to make sure students understood the material and weren’t simply guessing.

The final survey question asked the students which aspects of the tutorial they liked best. Thirty-two percent of responses cited the number of examples and the ability to have more when you need them as the best feature of the Binary Relations tutorial. Ten percent answered “none,” while 22 percent felt that they learned a lot through the tutorial and the practice it offered. Nine percent liked the feedback offered, while five percent thought the explanations were the strong point of this tutorial. Other responses listed that they liked being able to do the tutorial at home at their own pace, and also liked the variety this tutorial and its audio lectures offered in teaching the course material. Six percent felt that the best aspect of the tutorial was its ability to take students back to review material they did not understand. Some of these pointed out that it would be more useful to review these sections a different way, or that they wanted to review more than just one topic after taking a section test.

In general, these survey results indicated that students found the Binary Relations tutorial to be informative and very useful as a supplement to the class. Several students wished for more topics to be taught using lessons such as these. One final student comment in the Binary Relations tutorial is given below:

> The multiple examples and detailed explanations … helped me understand the topics. I liked being able to see multiple examples of concepts I did not understand as well. I understood Binary Relations better after doing this tutorial. I learned more from the tutorial than from WebAssign homework.

6.3 Counting Tutorial Survey Free Response Questions

In this section we describe the results of the free response portion of the Counting tutorial Survey. Question 11, asking if the feedback was helpful and clear and whether there was enough, received a yes answer from 82% of the responses, and only 9% of the responses were no. The remaining responses specified that formatting instructions for questions was needed, and that the feedback was somewhat helpful and clear.

Question 12 asked each student "What aspects of the tutorial did you like least?" Twenty-four percent of those responding answered none, while another 24% answered that they had trouble determining the right format for answers, and that some formats should have been accepted. Twelve percent wanted more variety in the questions, to demonstrate the many different types of counting questions they see in class. Eight percent felt that the tutorial was too long. Five percent thought the tutorial was good as it was, while 3% disliked the whole tutorial. Another 5% found parts of the tutorial vague or confusing. About 5% disliked aspects of the interface, suggesting a web interface, better navigation, and indicators of their progress in the tutorial. Of the remaining students, some felt that the tutorial was slow, or that the program did not know what material they needed to review.

Survey question 13 asked, "Are there any other features or improvements you would like to see included in the tutorial?" Forty-nine percent of the responses were "none." Seventeen percent wanted more variety in the questions and to see more examples. Twelve percent wanted more instructions on the format of answers, or for the program to accept more answer formats. Five percent wanted a different interface that would be easier to use; a few of these suggested a web or graphical user interface. 5% liked the program as it was, and 3% referred to their answers to question 11.

The final survey question 14 asked students to list their favorite aspects of the counting tutorial. Twenty-four percent of the responses cited the examples in the tutorial as the best aspect. Thirteen percent chose no features as the best. Eight percent thought the tutorial was good overall, while 7% really appreciated how much they learned from this tutorial. Another seven percent liked the feedback on this tutorial, which differed from the others in that, while students were working practice examples, successively more detailed hints were given as students missed the questions. Yet another seven percent said they liked the tutorial's ability to determine what they misunderstood and given them more practice on these topics. Eight percent appreciated the detailed, step-by-step explanations while some others liked how this one used pictures and diagrams to explain concepts. Five percent liked this tutorial since it was easy, but another five thought that having a test at the end of the tutorial made them learn more. The remaining students cited aspects like learning at their own pace, the concise and clear nature of the tutorial, and that this tutorial was a good supplement to what they learned in class.

6.4 Proofs Tutorial Survey Free Response Questions

In this section we describe the results of the free response portion of the Proofs tutorial Survey. Question 11, asking if the feedback was helpful and clear and whether there was enough, received a yes answer from 65% of students responding, and no from 24%. Five percent found the feedback confusing or vague, while four percent found the feedback somewhat helpful.

Question 12 asked each student “What aspects of the tutorial did you like least?” A large 24% portion of the students found this tutorial to be quite long, and felt that it should have required them to do about half of the 10 proofs offered in section 2. Another 21% felt that the interface was hard to use, making it harder to enter proofs in the tutorial than it is to do them by hand. Eleven percent felt that there was not enough help or feedback in the program, particularly when you got stuck on a problem, and another six percent describe the program as being “hard.” Five percent of the students listed the process of entering variables for substitutions in verifying the truth of a proof line as the more difficult and least useful aspect of the tutorial (although some later said that this made them understand how they actually used axioms in a proof). Five percent asked for more examples relevant to solving and entering proofs in the tutorial. Still other students

Survey question 13 asked, “Are there any other features or improvements you would like to see included in the tutorial?” 24% of the responses listed “none,” while 12% thought this tutorial was not worth doing on NovaNET – they felt it was too cumbersome and preferred doing proofs on paper or perhaps with a newly written program. 8% wanted more examples, to learn the use of the tutorial and some strategies in doing proofs. 12% wanted a better interface for this program, or to somehow make the program easier to use.

Eight percent of students wanted improved navigation within and without writing proofs. Within proofs, students want to be able to go back and edit completed lines of a proof and edit incorrect lines (currently the program erases incorrect lines). Some students also wished to be able to go back to the main menu or stop and resume the program later. These functions do exist in the program and are explained to students in their instructions for installing and running NovaNET, but these comments make it apparent that they are not reading and using the instructions as they might.

Eight percent of students wanted better feedback, which would be more specific about what the student did wrong on a proof line. 7% wanted help while writing proofs, and another 7% wished to work fewer proofs. Two percent of students wanted the program itself to determine the correctness of their statements without having to input variable substitutions (which is admittedly the most tedious part of the proof entry process). Some other suggestions made by the remaining students included: breaking the proofs into groups and listing them in order by difficulty, having better instructions on how to input proof lines, entering axioms by name instead of number, having the axiom list available during proofs (it is but students did not always realize how to get it), allowing more shortcuts in proofs, and having the program be less picky in general.

In general, students found the proofs tutorial to be hard to use, but appreciated that when the tutorial accepted their answers, they could be pretty sure they had a correct proof. In addition, after working so many of the problems, students became much better at writing proofs. However, students often got stuck and were unable to continue in some cases, as demonstrated by this student's suggestion:

> I would like there to be less proofs to prove by yourself. I believe the tutorial would be better if there were maybe 5 proofs that it helped you do and then maybe 5 that you have to do on your own. I was not able to finish all the proofs because I got stuck on certain parts. If I the tutorial gave clues when you get stuck I believe it would be more beneficial.

The final survey question 14 asked students to list their favorite aspects of the counting tutorial. Twenty-eight percent of students listed "none." However, 33% of students appreciated the practice and depth of learning they achieved using this tutorial. Eight percent liked how the program checked for proof correctness; that feedback was reassuring to them. Ten percent of the students liked the first section of the tutorial, which provided a brief outline of proof methods and provided a few small examples of how to use the axioms. Five percent of the students appreciated the format of the program (using the correct symbols for logic, etc) and that it was online, so they could practice proofs any time and from home.

Three percent responded that the best aspect of the program was that it was intelligent enough to understand their proofs and whether or not they were correct. For example, one student wrote: "I did enjoy how the [program] lets you know that you are skipping steps, or just lets you know that you might have a line, rule, or number missing or messed up on one of your lines." Other students commented that the program was fun and neat, lenient in accepting skipped steps or other shortcuts, allowed you to work on proofs step by step, it was interactive, and it built some students confidence in being able to work proofs.

Overall, students found the proofs tutorial hard to use, but felt that once they got used to the program, appreciated its ability to check their work and allow them to solve proofs in any order. Students wanted an added help feature in the program, which would assist them in the course of writing proofs, and thought that a few examples of completed proofs, shown step-by-step, would help them understand the process of entering proofs.

6.5 Summary of Tutorial Results

As we have seen in both the Likert-scale questions 1-10 and the free-response questions 11 through 14, students generally had positive experiences using the three NovaNET tutorials. With each of these tutorials, students felt that using the tutorials was at least as effective as traditional lectures and homework. Many students liked the individual pacing of the tutorials, and several requested that videos be added to each of the three tutorials.

For the Counting and Proof tutorials, students most disliked the form that answers were required to be written in. In the case of Counting, the tutorial expected answers written out with factorials, not computed on a calculator, and wasn't always flexible about the order they wrote their answers in, or whether or not they placed parentheses around parts of their answers. In the case of the Proofs tutorial, students grew weary of entering variable substitutions that would allow the program to verify the correctness of each line in their proof. In each case, these input methods can be addressed and improved. However, we also feel that these issues are important in mathematics. In counting, a student needs to know how to write answers in factorial form. In solving proofs, a student should be able to explain just how they used a rule to combine two lines. In future work, we can require that students perform these tasks a certain number of times until we feel they have learned enough, and then allow students to skip more of these steps as they understand more.

Some students did not like the process of installing NovaNET and running it on their home computers. Although they were provided with quite detailed instructions, many students still had problems with installation and connecting to NovaNET. Some students who waited to run the tutorials right before the deadline could not always log in, since our service is limited to 20 logins at once. (Note that this is not inherent in NovaNET, just in our service). Several students felt that putting the tutorial material on the web would have been beneficial, since they would have forward and back buttons and freer navigation. We chose to use NovaNET because these functions are available and display formatting and response parsing are much easier. We would like to eventually move these kinds of tutorials to the Internet to allow more students to access them.

Question 9 asks students to agree or disagree with the statement, "I felt like the program knew which concepts I did not understand, and directed me back to lessons on concepts I understood the least." This question is important in providing us with a subjective measurement of the success of our remediation process, particularly in the Binary Relations tutorial, the only tutorial we had enough data to have a previously-extracted q-matrix to use to determine student misconceptions. For this tutorial, 46% of students surveyed agreed with the statement, and 13% disagreed. The remaining students were neutral or gave no response. These data indicate that a much larger portion of students felt the tutorial was able to direct their review. Subjectively, we consider this a success for the q-matrix method.

7 Proofs Analysis & Data Mining

One finding of this research is the application of the q-matrix method as a general data mining tool. There are several steps in any data mining process. Two of the major steps in any data mining process are 1) data clustering and 2) interpretation of the data clusters. The q-matrix method combines these two steps into one, providing data clusters, and a first pass at understanding the meaning of the clusters created.

Use of the q-matrix method was clear in the Binary Relations and Counting Tutorials, and their structure permitted the extraction of concepts that related the questions in each. However, with the Proofs Tutorial, the first section did not lend itself to q-matrix extraction. There are 10 questions, design to train students to enter logic symbols on the computer. These questions were not necessarily expected to have underlying concept relationships.

The problems in Section 2 of the Proofs Tutorial were not well-suited to a traditional application of the q-matrix method, where we analyze student responses as answers to questions common to all students. Section 2 can be thought of as a proof-verifier, rather than a quiz with right and wrong answers. Instead of answering a single question, students type in consecutive lines of a proof, which consist of 4 parts: the statement, reference lines, the axiom used, and the substitutions which allow the axiom to be applied in this particular case. After the student enters these 4 parts to a line, the program verifies that:

1. The statement given is true, based on the previous lines of the proof.
2. The axiom and appropriate substitutions listed correspond to the given reference lines and statement.

The program prompts the student to continue if the proof line is valid, or warns the student that there was a problem with the line if it is not. In each case, the student is returned to entering his proof.

For the q-matrix method to work, students must all answer the same question, whose answers can be graded as correct or incorrect. For the proofs in Section 2, we could have compared whether the students completed proofs or did not complete them. However, eventually most students complete all of the proofs, and this comparison would yield little information. In this case, it would be much more useful to compare more detailed information from each student response.

There are several reasons that a straightforward q-matrix analysis could not be used for the proofs in Section 2. These are:

1. The program does not accept "wrong" proof lines. Therefore, there is no data to make a model that discriminates based on right verses wrong answers.
2. A proof can be incomplete for many reasons, including: lack of time, or the inability to continue.
3. Statements in one student's proof may or may not appear in another's.
4. Students solve proofs with different methods and in different orders.
5. There are too many valid proofs to anticipate every one.
6. If we did anticipate all possible valid proofs, considering each different answer as a "question" would require considerably more student data. Also, many proofs would really be the same approach executed in different orders.

We note here that although the program does not display invalid proof statements, these data were collected and saved along with each student proof. This type of data can later be analyzed to determine the types of errors students make in proof writing. Here we did not use this data, in favor of modeling the steps needed in correct proofs.

7.1 Method

Since the data for the proofs program did not lend itself easily to interpretation using the q-matrix method, we looked for data that would overlap in each student's responses. It is common practice in data mining and other data modeling applications to select only some of the observed variables for an analysis. This corresponds to ignoring outliers in a curve-fitting exercise. Since, for proofs, we looked for general relationships among axioms, and classes of problem solutions, axioms used by only one student solving a proof were not considered for analysis. For each problem, we found the rules that more than 2 students used, that were rules that resulted in a change of the knowledge base in the proof (for example, use of the commutative axiom was not considered to change the knowledge base in the proof). Then for each student we constructed a bit string for each proof that indicated which axioms the students used in each proof. If an axiom was used, the bit string contained a 1 in that place, and otherwise a 0. For example, if the rules used in a proof were Hypothetical Syllogism, Modus Ponens, and Contrapositive, and a student used only Contrapositive, then his answer vector is 001.

Once we determined the major rules used for a proof, we ran a q-matrix analysis for these answer vectors. Our hypothesis was that the concepts found by the q-matrix correspond to groups of rules that students must use together to complete parts of a proof.

To solve each proof, students chose rules from an Axiom List, which is included in the Appendix. There are 36 rules on this list, several of which are very similar. The first 10 of these rules are logical implications, meaning that these axioms result in more than a simple rearrangement of the variables; usually a variable is eliminated or added using one of these rules. For example, Modus Ponens combines a → b with a to find b is true. In a way, we have eliminated the variable a. These types of rules are most powerful in logic proofs. The remaining rules are logical equivalences, and only a few of these are extracted for analysis in this section.

For each proof, there are many possible solutions. However, our conjecture was that many student proofs, though appearing different, might be using the same underlying approach. Fundamentally, these approaches should overlap in the axioms that students used in the process of finding a solution. Therefore, our goal was to find groups of rules that, when used together, would devise all or part of the most crucial steps in solving a particular proof. To do so, we first narrowed the list of axioms we would consider as those most important in solving proofs. As in the general data mining process, this step was necessary to restrict our analysis to those items we felt were most important.

When compiling our list of rules, we first grouped rules that were only slightly different ways of writing the same approach. For example, we grouped DeMorgan's (27+29) for AND and for OR together as one rule. We also grouped Constructive Dilemma for AND and OR together (7+8). Modus Tollens and Modus Ponens (3+4) are nearly equivalent; we can prove Modus Tollens by using Modus Ponens and Contrapositive (29).

The rules considered for extraction for each proof are:

- Hypothetical Syllogism (1)
- Disjunctive Syllogism (2)
- Modus Ponens and Modus Tollens (3+4)

- Constructive Dilemma (7+8)
- Idempotent $(p \wedge p) \Leftrightarrow p$ (19) – used only in proof 9
- Tautology & Contradiction (25+26) – used in proofs 4,6,7, and 8
- DeMorgan's (27+28)
- Contrapositive (29)
- Implication (30+31)
- Negation of Conclusion (36)

The most important rules in writing proofs are those that result in a "change in the knowledge base" of a proof. In other words, the structure of the knowledge in the proof undergoes a change when these rules are applied. Most of these rules result in the removal of variables, such as Hypothetical Syllogism (1), Disjunctive Syllogism (2), Modus Ponens and Modus Tollens (3+4). Some of these rules result in the addition of or combination of variables into one line, including Constructive Dilemma (7+8) and Addition (9). We also included Negation of Conclusion (36), which differentiates direct proofs from proofs by contradiction. In our analysis, we also considered rules that would be crucial steps in being able to apply those rules listed above. The rules include Implication (30+31), Contrapositive (29).

In proof 4, we also included Tautology and Contradiction (25+26). In proof 6, we also used the Idempotent rule (19). These rules were used often by students, and seemed important in the solutions that students found for these two problems.

Rules that do not result in a "change in the knowledge base" of the proof include Simplification, Conjunction, and the remaining logical equivalences, including rules such as Double Negation, Commutative and Associative rules. An addition reason for eliminating these rules from consideration was that the proofs program often lets students skip these simple of steps, so we'd have no way of knowing whether students skipped these steps or not.

It is important to note two important properties of q-matrices, before we enter our proofs analysis discussion. In a q-matrix, items that relate to all concepts (i.e. with 1's for every concept) are the least frequently answered items. Therefore, for all concept states except the all-one concept state, their ideal response vectors will predict that these items are not used. In other words, these items will have zeroes in most ideal response vectors. On the other hand, items related to no concepts (i.e. with 0's for all concepts) are the most frequently used items. In this case, all ideal response vectors will predict a 1 for these items. These values have the least change over all students. In a q-matrix, we can then interpret the rows as ideal response vectors if we first reverse the bits for these two types of questions.

7.2 Proof models & discussion

In Section 2 of the Proofs Tutorial, students select proofs to solve from a given list, which is shown in Table 7-1. As students work each proof, their work is saved. We later process their proofs to extract the rules used by each student in a particular proof, creating "answer vectors" for each student – where the ith bit of the answer vector contains a one if the student used the ith rule, and a zero if not.

Table 7-1 Proofs Tutorial Section 2 Proofs

Given:	Prove:
1 a → b, c → d, ¬(a → d)	b ^ ¬c
2 a → b, ¬c → d, ¬b ∨ ¬d	a → c
3 (b ^ a) → c	a → (b → c)
4 a ∨ (b → c), b ∨ c, c → a	a
5 a → b, c → (a ^ d), (b ^ d) → (a ^ e), ¬e	a → ¬c
6 a → b, c → ¬b	¬a ∨ ¬c
7 a ∨ b, b → d, a → c	d ∨ c
8 (d ∨ b) → a, c ∨ d, ¬c ∨ b	a
9 ¬a ∨ d, (a ^ ¬c) → b, ¬b	a → (c ^ d)
10 a → (b ∨ c), b → d	a → (c ∨ d)

In the following sections, we discuss the concept extraction for each of these individual proofs. For each proof, we first list the proof to be solved. We then list the rules students used to solve this proof, and the q-matrix extracted from the student data. We interpret the q-matrix model of the proof, and compare this interpretation with actual student proofs and analyze this model to determine its validity for the problem given.

7.2.1 Proof 1 Discussion

The problem to be solved for Proof 1 is given below. In this proof, students used the rules: Hypothetical Syllogism (1), Disjunctive Syllogism (2), Modus Ponens and Modus Tollens (3+4), Constructive Dilemma (7+8), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36), for a total of eight rules used in this proof. Therefore, if a student used rules Modus Ponens, Contrapositive, and Implication, his answer vector would be 00100110, since these are the third, sixth, and seventh rules in our list.

Proof 1: Given: a → **b, c (d, ((a (d)** **Prove: b ((c**

 REF _Ref54352132 \h Table 7-2 lists the q-matrix extracted from student data in Fall 2002. By analyzing this q-matrix, we can draw several conclusions about student proofs without seeing them. First, we notice that columns 2, 7+8, and 29 (Disjunctive Syllogism, Constructive Dilemma, and Contrapositive, respectively), are related to all concepts. This means that most students do not use these rules in their proofs. We also note that columns 27+28 and 30+31 (DeMorgan's and Implication) are not related to the extracted concepts, so we can conclude that most students do use these rules in their proofs.

Table □ STYLEREF 1 \s □7□-□ SEQ Table * ARABIC \s 1 □2□ Proof 1 q-matrix

	1	2	3+4	7+8	27+28	29	30+31	36
Con 1	1	1	0	1	0	1	0	1
Con 2	0	1	1	1	0	1	0	0

Now we examine the extracted concepts. Aside from those rules in common between both states, concept 1 includes Hypothetical Syllogism (1) and Negation of Conclusion (36). Forty responses were assigned to this state 01, whose ideal response vector represents a proof that uses Hypothetical Syllogism, Negation of Conclusion, Implication, and DeMorgan's. Most students were assigned to state 10, using rules from concept 2 but not concept 1, indicating that most solutions in this state required the combination of DeMorgan's and Implication with Modus Ponens and Modus Tollens.

By examining concept 2, we conclude that most direct proof solutions for Proof 1 require the use of Modus Ponens and/or Modus Tollens, DeMorgan's, and Implication. This combination makes sense, since we would pull apart the statement ((a (d) by using Implication and DeMorgan's to conclude that a and (d are true, and then use Modus Ponens/Tollens to combine these with the other givens to get b and (c separately.

An example student proof using these rules is given in Table 7-3. The resulting answer vector for this proof is 00101010 (124 in octal reverse). As shown in

Table 7-4, 40 students used these same rules for their proofs, making this answer vector the most common. However, instead of having to read each proof separately in order to determine the most-used strategy, an instructor can view the extracted q-matrix for her students to determine which strategies they are applying.

Concept 2 of our q-matrix indicates that indirect proofs require the use of Negation of Conclusion along with Modus Ponens/Tollens, Implication, and Hypothetical Syllogism, and this set of rules corresponds to 23 of the 40 responses in state 10 (as we can see in

Table 7-4). Eight responses in this state used all these rules except for Modus Ponens/Tollens, indicating that this proof can be solved using Negation of Conclusion, Hypothetical Syllogism and Implication. An example of this type of student solution is given in Table 7-5.

Table 7-3 Proof 1 Example of Direct Proof

Given: $a \rightarrow b$, $c \rightarrow d$, $\neg(a \rightarrow d)$ Prove: $b \wedge \neg c$

1.	$a \rightarrow b$	Given	
2.	$c \rightarrow d$	Given	
3.	$\neg (a \rightarrow d)$	Given	
4.	$\neg (\neg a \vee d)$	3	rule 30 implication
4.	$a \wedge \neg d$	4	rule 27 demorgans
5.	a	4	rule 6 simplification
6.	$\neg d$	4	rule 6 simplification
7.	b	1,5	rule 3 modus ponens
8.	$\neg c$	2,6	rule 4 modus tollens
9	$b \wedge \neg c$	7,8	rule 10 conjunction

Table 7-4 provides a summary of Proof 1 responses, and their corresponding concept states. In this table, we first list the concept states and their corresponding ideal response vectors (IDRs). We then list actual student responses in each of these states and the number of students responding with this vector. Finally, we list the error associated with assigning each actual response to a concept state (by counting the number of different bits in the IDR and actual response) and the total error for all the students with a given actual response vector. The last row of the table sums the student and total error columns. The "octal reverse" columns list the response vectors in reverse-order octal.

Table 7-4 shows that concept state 10 is the most important state for understanding Proof 1, since most students fall into this state (132/189). Concept 01 seems to correspond to proofs where Negation of Conclusion (36) is used, since all students in that state use that rule. There are students with octal responses 304 and 324 that were assigned to state 10 who also used Negation of Conclusion. This seems to show that there are 2 distinct ways to use Negation of Conclusion in this proof, one using Hypothetical Syllogism, as in state 01, and the other using Modus Ponens/Tollens, as in state 10. This rich understanding of solutions to Proof 1 resulted solely in examining the q-matrix, and verifying our conclusions with actual student data.

For this q-matrix, we extracted 2 concepts. This matches with our heuristic for choosing the number of concepts to use, $\log_8(189) \approx 2$, so our number of concepts should have been near 2, and the error for 1 concept was more than 1 per student.

We conclude that using the q-matrix method to analyze student responses for Proof 1 is an effective tool in understanding the groups of rules important in solving Proof 1. In the next section we discuss our data mining analysis of Proof 2.

Table 7-4 Proof 1 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
00	00001010	120	00000010	100	1	1	1
			01000010	102	1	2	2
			01001010	122	1	1	1
			01001110	162	2	2	4
			00000011	300	1	2	2
01	10001011	321	10000011	301	3	1	3
			10001011	321	23	0	0
			01001011	322	2	2	4
			10101011	325	8	1	8
			01101011	326	1	3	3
			00001111	360	1	2	2
			10001111	361	2	1	2
10	00101010	124	00100010	104	11	1	11
			01100010	106	1	2	2
			00110010	114	1	2	2
			00101010	124	40	0	0
			10101010	125	2	1	2
			01101010	126	24	1	24
			11101010	127	1	2	2
			00111010	134	13	1	13
			00100110	144	1	2	2
			10100110	145	1	3	3
			00110110	154	1	2	2
			00101110	164	13	1	13
			00111110	174	18	2	36
			00100011	304	2	2	4
			00101011	324	3	1	3
11	11111111	377	10110110	155	1	3	3
			01111110	176	8	2	16
			11101011	327	1	2	2
			01100111	346	1	3	3
					189		175

Table 7-5 Proof 1 Example of Proof by Contradiction

Given: $a \to b$, $c \to d$, $\neg(a \to d)$ Prove: $b \wedge \neg c$

1.	$a \to b$	Given	
2.	$c \to d$	Given	
3.	$\neg (a \to d)$	Given	
4.	$\neg (b \wedge \neg c)$		Neg of Conclusion (36)
5.	$b \to c$	4	rule 31 implication
6.	$a \to c$	1,5	rule 1 hyp. syllogism
7.	$a \to d$	2,6	rule 1 hyp. syllogism
8.	$(a \to d) \wedge \neg (a \to d)$	3	rule 10 conjunction
9.	contradiction		

7.2.2 Proof 2 Discussion

The problem to be solved for Proof 2 is given below. In this proof, students used the rules: Hypothetical Syllogism (1), Modus Ponens and Modus Tollens (3+4), Constructive Dilemma (7+8), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36), for a total of seven rules used in this proof. Therefore, if a student used rules Modus Ponens, Contrapositive, and Implication, his answer vector would be 0100110, since these are the second, fifth, and sixth rules.

Proof 2: Given: $a \rightarrow b$, $\neg c \rightarrow d$, $\neg b \vee \neg d$ Prove: $a \rightarrow c$

Analysis of the q-matrix for this proof will show us the varying methods that students used to approach solutions to Proof 2. The q-matrix for Proof 2, listed in Table 7-6, consists of two concepts. Rule 7+8 (Constructive Dilemma) is related to all concepts, indicating that most students do not use this rule in their proofs. Most student proofs used rule 30+31 (Implication), as shown by its all-zero column in the q-matrix. This common usage of the implication rule makes sense; it is easiest to combine the given statement $\neg b \vee \neg d$ with other statements (using rules like Hypothetical Syllogism and Modus Ponens or Modus Tollens) by first converting it to an implication ($b \rightarrow \neg d$).

Table 7-6 Proof 2 q-matrix

	1	3+4	7+8	27+28	29	30+31	36
Con 1	0	1	1	1	0	0	1
Con 2	1	0	1	0	1	0	0

Concept 1 in the q-matrix contains rules Modus Ponens/Tollens (3+4), DeMorgan's (27+28), and Negation of Conclusion (36), while concept 2 contains Hypothetical Syllogism (1) and Contrapositive (29). From this division of rules into concepts, we might assume that proofs by contradiction correspond to concept 1, using rules 3+4, 27+28, and 30+31. We also deduce that concept 2 corresponds to direct solutions, using rules 1, 29, and 30+31.

If we assume that concept 2 corresponds to direct proofs, our q-matrix indicates that they will use the rules Hypothetical Syllogism, Implication, and Contrapositive. An example proof using these rules is given in Table 7-7. This combination makes sense, since we can convert all our givens into implication using the Implication rule, reverse

Table 7-7 Proof 2 Example of Direct Proof

Given: $a \rightarrow b$, $c \rightarrow d$, $\neg b \vee \neg d$ Prove: $a \rightarrow c$

	STATEMENTS	REASONS	
1	$a \rightarrow b$	Given	
2	$\neg c \rightarrow d$	Given	
3	$\neg b \vee \neg d$	Given	
4	$b \rightarrow \neg d$	3	rule 30 implication
5	$a \rightarrow \neg d$	1, 4	rule 1 hyp. syllogism
6	$\neg d \rightarrow c$	2	rule 29 contrapositive
7	$a \rightarrow c$	5, 6	rule 1 hyp. syllogism

the order of one using Contrapositive, and combine them using Hypothetical Syllogism to get our result. Table 7-9 shows that 1000110 is indeed the most popular answer vector in concept state 10, with 83 students completing the proof using these rules. Eighteen students in this state did not use Contrapositive in their proofs, probably using commutative and the implication rule to change the positions of variables.

Concept 1 in the q-matrix corresponds to solving Proof 2 using proof by contradiction. As we can see in Table 7-9, 41 of the 46 students assigned to concept state 01 completed proofs by contradiction for Proof 2 (using rule 36 corresponds to proof by contradiction). Table 7-8 gives an example of the most popular type of solution in this state. This proof corresponds to answer vector 0101011, used by 23 students in this state, with solutions using the rules Modus Ponens/Tollens (3+4), DeMorgan's (28), Implication (30+31), and Negation of Conclusion (36). The remaining students in this state sometimes used Hypothetical Syllogism in addition to the above rules, or in place of using Modus Ponens/Tollens.

Table 7-8 Proof 2 Example of Proof by Contradiction

Given: a → b, ¬c → d, ¬b ∨ ¬d   Prove: a → c

	STATEMENTS	REASONS	
1	a →b	Given	
2	¬c →d	Given	
3	¬b ∨ ¬d	Given	
4	¬(a → c)	Negation of Conclusion	
5	a ∧ ¬c	4	rule 31 implication
6	a	6	rule 6 simplification
7	¬c	6	rule 6 simplification
8	d	2, 7	rule 3 modus ponens
9	b	1, 6	rule 3 modus ponens
10	(b ∧ d)	3, 9	rule 10 conjunction
11	¬ (b ∧ d)	3	rule 28 DeMorgan's
12	¬ (b ∧ d) ∧ (b ∧ d)	10,11	rule 10 conjunction

Table 7-9 provides a summary of Proof 2 student responses and the concept states they were assigned to during the q-matrix analysis. As in Table 7-9, we list first the concept states and their corresponding IDRs, then actual responses, and finally the error associated with assigning responses to concept states. We can understand the q-matrix model created by examining the frequency of each student response; large numbers of students with one response tend to cause concepts to match IDRs with those most frequent responses. Since we run the q-matrix extraction routine until the average error per student is less than 1, we know that most student responses are within 1 or 2 of one of the ideal response vectors extracted. Therefore, we can have confidence in our conclusions drawn from a q-matrix analysis based on the concept states and ideal response vectors associated with those states.

Table 7-9 Proof 2 Ideal and Student Response Vectors

State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
00	0000010	040	1000000	001	3	2	6
			0100000	002	1	2	2
			0000010	040	1	0	0
			0100010	042	1	1	1
			0000110	060	1	1	1
01	0101011	152	0111010	056	5	2	10
			0100011	142	6	1	6
			1100011	143	3	2	6
			0001011	150	2	1	2
			1001011	151	1	2	2
			0101011	152	23	0	0
			1101011	153	4	1	4
			0011011	154	2	2	4
			0111011	156	1	1	1
10	1000110	061	1000010	041	18	1	18
			1001010	051	2	2	4
			1000110	061	83	0	0
			1100110	063	3	1	3
			1010110	065	1	1	1
			1001110	071	6	1	6
			1000111	161	5	1	5
11	1111111	177	0110110	066	3	3	9
			0111110	076	2	2	4
			1010011	145	1	3	3
					178		98

For this q-matrix, we extracted 2 concepts. This matches our heuristic for choosing the number of concepts to use; $\log_7(178) \approx 2$ so our number of concepts should have been near 2, and the error for 1 concept was more than 1 per student.

From the q-matrix analysis of student responses, we were able to determine the groups of rules that students used to solve Proof 2. We verified that the solutions predicted by the q-matrix were viable solutions to Proof 2 through actual proofs demonstrating the combination of rules found in the ideal response vectors for each state. From these results, we can conclude that using the q-matrix to determine groups of rules that are crucial in proof solving was effective. In the next section we continue our discussion with an analysis of Proof 3.

7.2.3 Proof 3 Discussion

The problem to be solved for Proof 3 is given below. In this proof, students used the rules: Modus Ponens and Modus Tollens (3+4), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36), for a total of five rules used in this proof. Therefore, if a student used rules Modus Ponens, Contrapositive, and Implication, his answer vector would be 10110, since these are the first, third, and fourth rules in our list.

Proof 3: Given: $(b \wedge a) \rightarrow c$ Prove: $a \rightarrow (b \rightarrow c)$

In this section, we will demonstrate the differences in our extracted model when we extract two different numbers of concepts for the same problem. The one-concept extracted q-matrix for Proof 3 is given in Table 7-10. The error associated with this q-matrix on this problem is 46, or about 46/42 students, or 1.1 per student. This is above the threshold of less than one student per concept, but we will show how even this first level of concept extraction can be effective in understanding student proofs.

For a one-concept q-matrix, all the relationships we observe will relate to "all" or "none" of the concepts, so it will differ slightly from an analysis with multiple concepts. First we notice that columns 3+4, 29, and 36 (Modus Ponens/Tollens, Contrapositive, and Negation of Conclusion, respectively) are related to all concepts. This means that most students do not use these rules in their proofs. We also note that columns 27+28 and 30+31 (DeMorgan's and Implication) are not related to any extracted concepts, so most students do use these rules.

Table 7-10 Proof 3 1-concept q-matrix

	3+4	27+28	29	30+31	36
Con 1	1	0	1	0	1

Since this q-matrix contains only one concept, we have two concept states for students. As we can see in Table 7-11, State 0 has IDR 01010, and 27 of the 42 response vectors are assigned to this state, meaning that most student responses included rules DeMorgan's and Implication.

Table 7-11 Proof 3 Ideal and Student Response Vectors for One Concept

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
0	01010	12	00010	10	5	1	5
			01010	12	4	0	0
			00110	14	5	2	10
			01110	16	2	1	2
			00011	30	2	2	4
			01011	32	9	1	9
1	11111	37	10011	31	1	2	2
			11011	33	13	1	13
			01111	36	1	1	1
					42		46

There were 16 direct proof solutions for Proof 3, and 26 proofs by contradiction. An example direct proof demonstrating the answer vector 01010, using rules DeMorgan's and Implication, is given in Table 7-12. Note that this proof also includes both associative and commutative rules, but these rules were not considered for analysis, since they only affect the order of the statement they modify. A large group of students in state 0 also used Negation of Conclusion (36), solving this proof by contradiction, and using DeMorgan's and Implication.

Table 7-12 Proof 3 Example of Direct Proof

Given: (b ∧ a) → c   Prove: a → (b → c)

	STATEMENTS	REASONS	
1	(b ∧ a) → c	Given	
2	¬(b ∧ a) ∨ c	1	rule 31 implication
3	¬(a ∧ b) ∨ c	2	rule 13 commutative
4	(¬a ∨ ¬b) ∨ c	3	rule 28 DeMorgan's
5	¬a ∨ (¬b ∨ c)	4	rule 14 associative
6	a → (¬b ∨ c)	5	rule 30 implication
7	a → (b → c)	6	rule 30 implication

Table 7-13 lists a solution of Proof 3 using proof by contradiction. Since this proof uses Modus Ponens, DeMorgan's, Implication, and Negation of Conclusion, its answer vector is 11011, which corresponds to 13 student responses in Table 7-11. These students are in state 1, whose ideal response vector (IDR) is all ones, so the error associated with assigning these students to state 1 is just 1 per student. State 1 corresponds to understanding concept 1, which is the only concept extracted here for Proof 3, and also corresponds to using all or most of the extracted rules.

Table 7-13 Proof 3 Example of Proof by Contradiction

Given: $(b \wedge a) \rightarrow c$ Prove: $a \rightarrow (b \rightarrow c)$

	STATEMENTS	REASONS	
1	$(b \wedge a) \rightarrow c$	Given	
2	$\neg (a \rightarrow (b \rightarrow c))$	Negation of Conclusion	
3	$\neg (\neg a \vee \neg(b \rightarrow c)$	2	rule 30 implication
4	$a \wedge (b \wedge \neg c)$	3	rule 27 DeMorgan's
5	$a \wedge b$	4	rule 6 simplification
6	c	1,5	rule 3 modus ponens
7	$\neg c$	4	rule 6 simplification
8	$c \wedge \neg c$	6,7	rule 10 conjunction
9	Contradiction		

We now contrast the one-concept q-matrix extraction with the 2-concept extraction to determine its added effectiveness in understanding solutions to Proof 3. The 2-concept q-matrix for Proof 3 is given in Table 7-14. The error associated with this q-matrix is 19/42 = 0.48 per student. Since this error is less than 1, we choose this as the q-matrix appropriate for Proof 3, since most students will have responses that differ from their predicted responses by 1 or less on average. We now examine the differences between the Proof 3 solution model created for 1 and 2 concepts.

Comparing Table 7-14 with Table 7-10, we see that with one concept, our q-matrix row 1 was 10101, indicating frequent use of rules 27+28 and 30+31, and relatively infrequent use of rules 3+4, 29, and 36. Our 2-concept q-matrix shows that rule 30+31 is truly used by most students, and our 1-concept solution did not use 27+28 to separate students between the 2 concept states it had available. Using 2 concepts, we now have 4 concept states available, and between these concepts, rule 29 differentiates the 2 concepts the most strongly.

Table 7-14 Proof 3 2-concept q-matrix

	3+4	27+28	29	30+31	36
Con 1	1	1	0	0	1
Con 2	0	0	1	0	0

In Table 7-15 we see that the 4 ideal response vectors for our 4 states are 00010, 11011, 00110, and 11111. State 00 demonstrates the "default rules" – those that all students use. State 11 demonstrates the combination of all rules, so students who use almost all the rules are placed in this state. State 01 groups students primarily using rules in the ideal response vector (IDR) for concept 1: 11011. This means that students in this group used most of the rules except for Contrapositive. State 10 is comprised of students whose IDR is 00110 – using rules Contrapositive and Implication. Comparing the concept states between the two extractions, we observe that State 0 for the 1-concept solution was split mainly into two groups to form concept states 00 and 10 in the 2-concept solution. The largest change was in the formation of concept state 01 in the 2-concept solution; its combination of rules from states 0 and 1 in the 1-concept solution resulted in better IDR predictions by the 2-concept q-matrix.

In using the q-matrix method, as in many other data mining methods, we must set criteria for the number of clusters for our data; in other words we must choose how many concepts to extract in a given situation. In this case, we have demonstrated the differences between selecting 1 or 2 concepts for Proof 3. In both cases, we were able to understand the sets of rules students used to solve Proof 3. If a quick analysis is needed where we simply want to determine the most and least occurring variables, a 1-concept solution suffices. However, since we have 42 observations, and only 5 variables, we can justify a 2-concept solution that separates responses into 4 concept states. In this case, we have an expected value of 10 students per state, an acceptable number.

Table 7-15 Proof 3 Ideal and Student Response Vectors for Two Concepts

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
00	00010	10	00010	10	5	0	0
			01010	12	4	1	4
			00011	30	2	1	2
01	11011	33	10011	31	1	1	1
			01011	32	9	1	9
			11011	33	13	0	0
10	00110	14	00110	14	5	0	0
			01110	16	2	1	2
11	11111	37	01111	36	1	1	1
					42		19

Our range of possible numbers of concepts is 1-5, the number of variables. The number of concept states, or clusters, for these is 2, 4, 8, 16, and 32. With 42 responses, the respective expected number of responses per cluster is 21, 10, 5, 2, and 1 (rounding down). The use of 4 or 5 concepts does not give a sufficient level of abstraction; these would be almost like examining the full data set. In some cases, where there is little variance in responses, one concept will suffice to explain the data, while in most cases two or three concepts are better used. A rule of thumb for the number of concepts, aside from our error criterion of an average of less than one per response, would be the log (base = number of variables) of the number of responses collected This heuristic can reduce the q-matrix method run time by providing a target number of concepts.

From this analysis, we conclude that proof 3 could be solved using only 2 rules, DeMorgan’s and Implication. This tells us that this proof is a transformation of the givens into the final solution – because Modus Ponens/Tollens was not necessary in a direct proof, and the remaining rules cause changing of symbols, not the elimination of any variables. We continue our analysis in the next section with Proof 4.

7.2.4 Proof 4 Discussion

In Proof 4, students used the rules: Hypothetical Syllogism (1), Disjunctive Syllogism (2), Modus Ponens and Modus Tollens (3+4), Tautology & Contradiction (25+26), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36), for a total of eight rules. Therefore, if a student used rules Hypothetical Syllogism, Modus Ponens, Contrapositive, and Implication, his answer vector would be 10100110, since these are the first, third, sixth, and seventh rules in our list. The problem to be solved for Proof 4 is given below.

Proof 4: Given: $a \vee (b \rightarrow c)$, $b \vee c$, $c \rightarrow a$ Prove: a

Table 7-16 lists the Proof 4 q-matrix extracted from student data in Fall 2002. Our heuristic for the number of concepts is $\log_8 (143) \approx 2$, but 3 concepts were needed to reduce the error sufficiently. In this q-matrix, rule 27+28 (DeMorgan's) is not related to the extracted concepts, so most students do use these rules in their proofs. Rule 1 (Hypothetical Syllogism) is related only to concept 1, meaning that this rule will discriminate between responses assigned to concept 1 and those not assigned to concept 1. Rule 30+31 relates only to concept 2, so this rule will be common among responses assigned to concept 2. Rules 3+4 and 36 are related only to concept 3, so these will similarly be common among concept 3 responses, so we see that most proofs by contradiction will involve use of concept 3. No rules are related only to concept 4; this indicates that this concept is used to separate responses among the other concepts.

Table 7-16 Proof 4 q-matrix

	1	2	3+4	25+26	27+28	29	30+31	36
Con 1	1	0	0	1	1	1	0	0
Con 2	0	0	0	1	1	1	1	0
Con 3	0	1	1	0	1	0	0	1
Con 4	0	1	0	0	1	1	0	0

Table 7-17 lists student responses to Proof 4 by concept state. Since we have 4 concepts for Proof 4, we have up to 16 concept states for students, but only 11 are used. The two most common student responses are 01100001 (o206) and 00100011 (o304), each with 10 student responses. Response 01100011 (o306) runs a close third with 9 student responses. These responses correspond to the ideal response vectors for three concept states: 0110, 1100, and 1110. This indicates that concept 3, combined with concept 2 or concept 4, or both, covers a large portion of the responses (66 of the 143 total are in these three states). This indicates that Modus Ponens/Tollens (3+4) and Negation of Conclusion (36) are commonly used in popular responses.

Table 7-18 lists a proof with response 0110001 (o206), using Disjunctive Syllogism (2), Modus Ponens (3+4), and Negation of Conclusion (36). Proofs with response 0110011 (o306) are similar, but also use Implication. Proofs with response 00100011 (o304) replace all uses of Disjunctive Syllogism with Implication and Modus Ponens/Tollens.

Direct proof responses are generally those not using concept 3. This corresponds to concept states 0001 (4 responses), 0010 (7 responses), 0011 (11 responses), and 1011 (15 responses), whose ideal response vectors (in octal reverse) are o1, o100, o111, and 0151, respectively. An example of a direct proof solution to Proof 4 is given in Table 7-19. In this solution, note that the “Given” statement on line 3 us used twice in the proof (on lines 5 and 9). This is an uncommon occurrence in simple proofs; most of the time students use a given statement only once. This proof was also difficult in that it required many conversions of the form of each statement to combine lines together – this can be seen in the many uses of the implication rule in this solution.

In examining these proofs, we found that a few students were able to arrive at proof solutions that are not technically valid (e.g. o1 = 10000000, using only Hypothetical Syllogism), so later versions of the Proofs Tutorial will need to be corrected. These invalid proofs arose when students entered proof statements that were true, but were not direct results of the rules they cited. While the proof tutorial checks for commonly skipped steps, such as commutative, double negation, and distributive, and allows these, we've discovered that it sometimes allows a student to skip several steps that are necessary in the proof.

It is excellent news that the q-matrix analysis reveals places that need improvement in the tutorial. Since there are few of these invalid responses, they do not affect q-matrix error significantly, so we have not removed these data points from the set. We also note here that although some responses may appear to be invalid, our extraction method did not always extract all the rules that students may have used in a proof; at least 2 students had to use a rule before it would be extracted. Therefore, some response vectors do not reflect all the rules that students used in solving proofs. We feel that this drawback to the method is affordable, since we are concerned with general proof solution methods, and not specific or rare responses.

Table 7-17 Proof 4 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
0001	10000000	001	10000000	001	4	0	0
0010	00000010	100	01000010	102	3	1	3
			00010010	110	3	1	3
			00000011	300	1	1	1
0011	10010010	111	10000010	101	5	1	5
			10100010	105	1	2	2
			10010010	111	3	0	0
			10000011	301	1	2	2
			10010011	311	1	1	1
0100	00100001	204	00100000	004	1	1	1
			00101001	224	1	1	1
0110	00100011	304	00100010	104	2	1	2
			00110010	114	1	2	2
			00100011	304	10	0	0
			00101011	324	6	1	6
			00100111	344	5	1	5
			00101111	364	1	2	2
0111	10110011	315	10110010	115	1	1	1
			10100011	305	2	1	2
			00110011	314	6	1	6
			10110011	315	5	0	0
			11110011	317	2	1	2
			10100111	345	1	2	2
			10110111	355	3	1	3
1011	10010110	151	10000110	141	3	1	3
			10100110	145	1	2	2
			00010110	150	5	1	5
			10010110	151	4	0	0
			00010111	350	1	2	2
1100	01100001	206	01100001	206	10	0	0
			01110001	216	1	1	1
			01101001	226	1	1	1
			01111001	236	1	2	2
			01100101	246	2	1	2
1101	11100001	207	11000001	203	1	1	1
			11100001	207	1	0	0
			11110001	217	2	1	2
			11100011	307	5	1	5
1110	01100011	306	01000011	302	2	1	2
			01100011	306	9	0	0
			01010011	312	2	2	4
			01110011	316	7	1	7
			01101011	326	5	1	5
			01100111	346	1	1	1
1111	11111111	377	11100110	147	1	2	2
			11101011	327	1	2	2
			11100111	347	2	2	4
			01110111	356	1	2	2
			11110111	357	3	1	3
			00111111	374	1	2	2
			01111111	376	1	1	1
					143		113

Table 7-18 Proof 4 Example of Proof by Contradiction

Given: a ∨ (b → c), b ∨ c, c → a   Prove: a

	STATEMENTS	REASONS	
1	a ∨ (b → c)	Given	
2	b ∨ c	Given	
3	c → a	Given	
4	¬a	Negation of Conclusion	
5	b → c	1, 4	rule 2 disj. syllogism
6	¬c	3, 4	rule 4 modus tollens
7	b	2, 6	rule 2 disj. syllogism
8	c	5, 7	rule 3 modus ponens
9	¬c ∧ c	6, 8	rule 10 conjunction

Table 7-19 Proof 4 Example of Direct Proof

Given: a ∨ (b → c), b ∨ c, c → a   Prove: a

	STATEMENTS	REASONS	
1	a ∨ (b → c)	Given	
2	b ∨ c	Given	
3	c → a	Given	
4	¬b → c	2	rule 30 implication
5	¬b → a	3, 4	rule 1 hyp. syllogism
6	¬a → b	5	rule 29 contrapositive
7	a ∨ ¬b ∨ c	1	rule 30 implication
8	¬ (a ∨ ¬b) → c	7	rule 30 implication
9	¬ (a ∨ ¬b) → a	3, 8	rule 1 hyp. syllogism
10	a ∨ ¬b ∨ a	9	rule 30 implication
11	a ∨ ¬b	10	rule 19 idempotent
12	a ∨ b	6	rule 30 implication
13	a ∨ (b ∧ ¬b)	11,12	rule 16 distributive
14	a ∨ 0	13	rule 26 contradiction
15	a	14	rule 20 identity

Proof 4 offered a challenge for students, as we can see in the large number if different response vectors for this proof. We believe this also caused students to attempt entering statements into the program without understanding how to use the rules to get those results. The good news is that the q-matrix analysis still revealed the main methods of proof for this problem, and indicates that most students preferred proof by contradiction for this proof. These findings can help us build a help program, which will be able to suggest popular solution paths to students who get stuck on a proof. We continue our exploration of the q-matrix method as applied to Proof 5 solutions in the next section.

7.2.5 Proof 5 Discussion

The problem to be solved for Proof 5 is given below. Although its given statements appear to be complex, student proofs used only about five different rules to solve this proof, including: Disjunctive Syllogism (2), Modus Ponens and Modus Tollens (3+4), DeMorgan's (27+28), Implication (30+31), and Negation of Conclusion (36). Therefore, if a student used rules Modus Ponens and Implication, her answer vector would be 01010, since these are the second and fourth rules in our list.

Proof 5: Given: $a \rightarrow b$, $c \rightarrow (a \wedge d)$, $(b \wedge d) \rightarrow (a \wedge e)$, e Prove: $a \rightarrow c$

Table 7-20 lists the Proof 5 q-matrix extracted from Fall 2002 student data. Since this is a one-concept q-matrix, all relationships demonstrated in the q-matrix are "all" or "none" relationships. Here, rule 2 (Disjunctive Syllogism), relates to all concepts, while the remaining rules do not relate to any concepts. In this case, we would predict that most students used a combination of rules 3+4, 27+28, 30+31, and 36 in their proofs.

Table 7-20 Proof 5 q-matrix

	2	3+4	27+28	30+31	36
Con 1	1	0	0	0	0

Table 7-21 lists the responses used in Proof 5 solutions, grouped by concept state. We have listed the ideal response vectors associated with each concept state, and compared these with actual student responses to obtain the error associated with assigning responses to their particular concept states. We can see in Table 7-21 that the most common actual student response is 01111, corresponding to using Modus Ponens/Tollens, DeMorgan's, Implication, and Negation of Conclusion. This response corresponds to our prediction, using only the q-matrix, that most student proofs used these four rules. An example proof by contradiction with this answer vector is given in Table 7-22. Students also solved this problem with similar proofs, but using perhaps Implication (31) instead of Implication (30) and DeMorgan's as in Table 7-22, resulting in a response vector of 01011. However, combinations of rules that do not use either Disjunctive Syllogism (2) or Modus Ponens/Tollens (3+4) cannot be valid solutions to this proof, since at least one of these rules is needed to combine multiple lines together. We cannot combine multiple given lines using rules DeMorgan's, Implication, or Negation of Conclusion. This helped us to detect incomplete response vectors, including 00110 (1 response), 00001 (2 responses), and 00011 (1 response). These responses could have been invalid, as we discussed for Proof 4 in the previous section, or, more likely, we simply did not extract all the rules that students might have used in their solutions. The incompleteness of these response vectors does not adversely affect our analysis, though, since it represents only 4 out of 124 responses and the total error on this subset is 10.

Table 7-21 Proof 5 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
0	01111	36	01010	12	1	2	2
			00110	14	1	2	2
			01110	16	3	1	3
			00001	20	2	3	6
			01101	26	1	1	1
			00011	30	1	2	2
			01011	32	16	1	16
			00111	34	2	1	2
			01111	36	92	0	0
1	11111	37	10110	15	4	2	8
			10111	35	1	1	1
					124		43

Direct proof solutions to Proof 5 correspond to those not using Negation of Conclusion. In Table 7-21 we see that there are a total of 9 responses not using rule 36, indicating that few students used direct solutions for Proof 5. These 9 students had response vectors 01010, 00110, 01110, and 10110, with rule 30+31 in common. Although a specific concept was not extracted for this, we can still conclude that Implication was commonly used in direct solutions to Proof 5. A sample direct proof is given in Table 7-23. Note that this proof uses the Addition rule (9) and Hypothetical Syllogism (1). However, there were not sufficient responses using these rules to extract them for processing.

Most students found Proof 5 to be among the most difficult to solve; most were unable to solve it by direct proof and attempted it by contradiction. From our extracted q-matrix, we determined that proofs by contradiction applied Modus Ponens/Tollens, DeMorgan's, Implication, and Negation of Conclusion. This makes sense, since Negation of Conclusion must be converted to an OR before using DeMorgan's, and then the simplified results (a and c) demand the use of Modus Ponens with the givens. This technique is how the students were taught to solve proofs by contradiction.

Table 7-22 Proof 5 Example of Proof by Contradiction

Given: a → b, c → (a ∧ d),(b ∧ d) → (a ∧ e), ¬e   Prove: a → ¬c

	STATEMENTS	REASONS	
1	a → b	Given	
2	c → (a ∧ d)	Given	
3	(b ∧ d) → (a ∧ e)	Given	
4	¬e	Given	
5	¬(a → ¬c)	Negation of Conclusion (36)	
6	¬(¬a v ¬c)	5	rule 30 implication
7	a ∧ c	6	rule 27 DeMorgan's
8	a	7	rule 6 simplification
9	c	7	rule 6 simplification
10	b	8, 1	rule 3 modus ponens
11	a ∧ d	9, 2	rule 3 modus ponens
12	d	11	rule 6 simplification
13	b ∧ d	10, 12	rule 10 conjunction
14	a ∧ e	13, 3	rule 3 modus ponens
15	e	14	rule 6 simplification
16	e ∧ ¬e	15, 4	rule 10 conjunction

Table 7-23 Proof 5 Example of Direct Proof

Given: a → b, c → (a ∧ d), (b ∧ d) → (a ∧ e), ¬e   Prove: a → ¬c

	STATEMENTS	REASONS	
1	a → b	Given	
2	c → (a ∧ d)	Given	
3	(b ∧ d) → (a ∧ e)	Given	
4	¬e	Given	
5	(a ∧ e) → e		rule 6 simplification
6	¬ (a ∧ e)	4,5	rule 4 modus tollens
7	¬ (b ∧ d)	3,6	rule 4 modus tollens
8	b → ¬d	7	rule 31 implication
9	a → ¬d	1,8	rule 1 hyp. syllogism
10	¬a ∨ ¬d	9	rule 30 implication
11	¬ (a ∧ d)	10	rule 28 DeMorgan's
12	¬c	2, 11	rule 4 modus tollens
13	¬a ∨ ¬c	12	rule 9 addition
14	a → ¬c	13	rule 30 implication

A difficult problem to solve is how to help a student when he or she is stuck on a proof. The q-matrix method can help us solve this problem by showing us which rules are most applicable to a proof solution. Then, when a student becomes stuck on the problem, a help program could suggest some of the most popular rules used, determined from the q-matrix analysis. For this, we could choose the rules in the ideal response vector of the concept state containing the largest number of student responses. This choice is justified, since it most likely corresponds to the simplest, or easiest to understand, solution. We continue our discussion with Proof 6 in the next section.

7.2.6 Proof 6 Discussion

The problem to be solved for Proof 6 is given below. Although it has few givens, students used a variety of rules to solve this problem, demonstrating the richness of possible proof solutions, even in the simplest of cases. Students used seven rules to solve Proof 6: Hypothetical Syllogism (1), Modus Ponens/Tollens (3+4), Tautology/ Contradiction (25+26), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36). Therefore, if a student used rules Modus Ponens, Tautology, and Implication, his answer vector would be 0110010, since these are the second, third, and fifth rules in our list.

Proof 6: Given: $a \rightarrow b, c \rightarrow \neg b$ Prove: $\neg a \vee \neg c$

Table 7-24 lists the Proof 6 q-matrix extracted from Fall 2002 student data. In this two-concept q-matrix, rule 25+26 relates to all concepts, while the remaining rules relate to one concept each. Accordingly, we'd expect students using concept 1 to include rules 1, 29, and 30+31, while students using concept 2 would use rules 3+4, 27+28, and 36. Therefore, most students solving Proof 6 by direct proof should be in concept state 00 or 01 while proofs by contradiction would fall into concept states 10 or 11. We would expect most students in state 11 to use all or most of the extracted rules.

Table 7-24 Proof 6 q-matrix

	1	3+4	25+26	27+28	29	30+31	36
Con 1	1	0	1	0	1	1	0
Con 2	0	1	1	1	0	0	1

Table 7-25 lists the actual responses used to solve Proof 6, grouped by concept state. Here we see that almost half (79 of 160) of the responses were 1000110 (o61), corresponding to the ideal response vector for concept state 01. Concept state 10's ideal response vector, 0101001 (o112), corresponds to the second most popular response, used by 29 students.

Table 7-25 Proof 6 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
00	0000000	000	1000000	001	2	1	2
			0011000	014	3	2	6
			0011100	034	1	3	3
			0000010	040	1	1	1
			0010010	044	1	2	2
			0011010	054	3	3	9
			0000001	100	1	1	1
			0010011	144	1	3	3
01	1000110	061	1000100	021	1	1	1
			1000010	041	17	1	17
			1010010	045	1	2	2
			0000110	060	1	1	1
			1000110	061	79	0	0
			1100110	063	2	1	2
			1001110	071	1	1	1
10	0101001	112	0100001	102	1	1	1
			0110001	106	1	2	2
			0001001	110	2	1	2
			0101001	112	29	0	0
			0011001	114	1	2	2
			0111001	116	3	1	3
			0001011	150	3	2	6
			0101011	152	1	1	1
11	1111111	177	0111010	056	1	3	3
			0011011	154	2	3	6
			1001111	171	1	2	2
					160		79

Table 7-26 lists a sample solution of Proof 6 with response: 1000110 (o61). This popular direct solution uses rules Hypothetical Syllogism, Contrapositive, and

Implication to solve Proof 6. The second most used solution in concept state 01 is 1000010, without using Contrapositive. These proofs used Implication and Commutative rules to achieve the same result.

Table 7-26 Proof 6 Example of Direct Proof

Given: a → b, c → ¬b   Prove: ¬a ∨ ¬c

	STATEMENTS	REASONS	
1	a → b	Given	
2	c → ¬b	Given	
3	b → ¬c	2	rule 29 contrapositive
4	a → ¬c	3	rule 1 hyp. syllogism
5	¬a ∨ ¬c	4	rule 30 implication

Table 7-27 lists an example of proof by contradiction with response vector 0101001 (o112), the ideal response vector and most common response in concept state 10. This proof uses the most straightforward approach for proof by contradiction, immediately combining the results of the Negation of Conclusion with the givens using DeMorgan's and Modus Ponens/Tollens.

Table 7-27 Proof 6 Example of Proof by Contradiction

Given: a → b, c → ¬b   Prove: ¬a ∨ ¬c

	STATEMENTS	REASONS	
1	a → b	Given	
2	c → ¬b	Given	
3	¬(¬a ∨ ¬c)	Negation of Conclusion	
4	a ∧ c	3	rule 27 DeMorgan's
5	a	4	rule 6 simplification
6	c	4	rule 6 simplification
7	b	5,1	rule 3 modus ponens
8	¬c	2,7	rule 4 modus tollens
9	c ∧ ¬c	6,8	rule 10 conjunction

Our q-matrix analysis of Proof 6 drew out the 2 most popular solutions, letting us know which rules could be combined for its solution. The q-matrix analysis also assigned other solutions in groups with these, allowing us to see alternative paths with the same general solutions. We continue our analysis with Proof 7 in the following section.

7.2.7 Proof 7 Discussion

The problem to be solved for Proof 7 is given below. In this proof, students used the rules: Hypothetical Syllogism (1), Modus Ponens/Tollens (3+4), Tautology/ Contradiction (25+26), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36), for a total of seven rules. Therefore, if a student used rules Modus Ponens, Tautology, and Implication, his answer vector would be 0110010, since these are the second, third, and fifth rules in our list.

Proof 7: Given: $a \vee b$, $b \rightarrow d$, $a \rightarrow c$ Prove: $d \vee c$

Table 7-28 lists the Proof 7 q-matrix extracted from Fall 2002 student data. As in proof 6, rule 25+26 (Tautology & Contradiction) is infrequently used, and relates to all concepts. Concepts 1 and 2 could be combined to create proofs by contradiction, since we see rule 26 split between these two concepts. Concept 1 is distinguished by the use of Modus Ponens/Tollens (shown by a single 1 relating concept 1 to rule 3+4 in the q-matrix) and Constructive Dilemma, while we'd expect responses employing concept 2 to use Hypothetical Syllogism, Contrapositive, and Implication instead. Responses using concepts 1 and 2 use DeMorgan's and Negation of Conclusion. Direct proof solutions seem to correspond to concept 3, but since this concept does not contain an exclusive relationship to any of the extracted rules, we expect this concept to be combined with others to find direct solutions.

Table 7-28 Proof 7 q-matrix

	1	3+4	7+8	25+26	27+28	29	30+31	36
Con 1	0	1	1	1	1	0	0	1
Con 2	1	0	0	1	1	1	1	1
Con 3	1	0	1	1	0	1	1	0

Table 7-29 lists the actual response vectors for proof 7, arranged by the concept state assigned to each by the q-matrix analysis. We use this data to verify our conjectures made by examining the q-matrix. First, we note that very few responses correspond to using any one concept. Concept state 011 corresponds to using both concepts 1 and 2, suggesting that in these solutions rules 27+28 and 36 are both used. This can be seen in this state's ideal response vector, 01001001 (o222), which includes Modus Ponens/Tollens (3+4), DeMorgan's (27+28), and Negation of Conclusion (36). Thirty-one students were assigned to this state, and 13 of these had responses equal to the IDR.

As we predicted, direct proof solutions use concept 3, but are divided between also using concept 1 or concept 2. Concept state 101, using concepts 1 and 3, contains a group of 45 students solving proof 7 using rules Modus Ponens/Tollens and Constructive Dilemma. In contrast, concept state 110 combines concepts 2 and 3, where the ideal response vector of 10000110 shows that students in this state used Hypothetical Syllogism, Contrapositive, and Implication. As we can see in Table 7-29, most students in this state did use these rules, while a significant subset used Hypothetical Syllogism and Implication but not Contrapositive.

Table 7-30 demonstrates a sample proof assigned to concept state 101, whose response vector of 01100000 indicates that students used Modus Ponens/Tollens and Constructive Dilemma to solve this proof. This straightforward proof was evident to students familiar with Constructive Dilemma, since the given statement a ∨ b and the result d ∨ c clearly matched with the hypotheses and conclusions of the given statements 2 and 3. Students not comfortable with Constructive Dilemma fell into concept state 110.

Table 7-29 Proof 7 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
000	00000000	000	10000000	001	3	1	3
			00010100	050	1	2	2
			00010001	210	1	2	2
			00000011	300	1	2	2
001	01000000	002	01000000	002	1	0	0
011	01001001	222	01001001	222	13	0	0
			01011001	232	3	1	3
			01001101	262	2	1	2
			00001011	320	1	2	2
			01001011	322	6	1	6
			11001011	323	1	2	2
			00011011	330	2	3	6
			01011011	332	1	2	2
			01001111	362	2	2	4
101	01100000	006	01100000	006	45	0	0
			01100010	106	3	1	3
110	10000110	141	10000010	101	10	1	10
			10100010	105	2	2	4
			10110010	115	5	3	15
			10000110	141	47	0	0
			10000111	341	1	1	1
111	11111111	377	11101010	127	1	2	2
			10111010	135	2	3	6
			11001111	363	3	2	6
			01011111	372	2	2	4
				tot:	159		87

Table 7-30 Proof 7 Example of Direct Proof

Given: a ∨ b, b → d, a → c Prove: d ∨ c

	STATEMENTS	REASONS
1	a ∨ b	Given
2	b → d	Given
3	a → c	Given
4	(b ∨ a) → (d ∨ c)	2, 3 rule 7 const dilemma
5	(a ∨ b) → (d ∨ c)	4 rule 12 commutative
6	d ∨ c	1, 5 rule 3 modus ponens

Students in concept state 110 used rules Hypothetical Syllogism, Contrapositive, and Implication to solve Proof 7. An example proof using these rules is given in Table 7-31. Using first commutative on line 1 to get b ∨ a and then applying the Implication rule would result in the equivalent proof without using Contrapositive, resulting in a response vector of 10000010, which is used by 10 students to solve this proof.

Table 7-31 Proof 7 Example of Direct Proof (State 110)

Given: a ∨ b, b → d, a → c Prove: d ∨ c

	STATEMENTS	REASONS	
1	a ∨ b	Given	
2	b → d	Given	
3	a → c	Given	
4	¬a → b	1	rule 30 implication
5	¬a → d	2,4	rule 1 hyp. syllogism
6	¬d → a	5	rule 29 contrapositive
7	¬d → c	3,6	rule 1 hyp. syllogism
8	d ∨ c	7	rule 30 implication

Students solving Proof 7 by contradiction were assigned to states 011 and 111. State 011's ideal response vector is 01001001 (o222), which includes Modus Ponens/Tollens (3+4), DeMorgan's (27+28), and Negation of Conclusion (36). Thirty-one students were assigned to this state, and 13 of these had responses equal to the IDR. A sample proof using these rules plus Implication is given below.

Table 7-32 Proof 7 Example of Proof by Contradiction

Given: a v b, b → d, a → c Prove: d v c

	STATEMENTS	REASONS	
1	a v b	Given	
2	b → d	Given	
3	a → c	Given	
4	¬(d v c)	Negation of Conclusion	
5	¬d ∧ ¬c	4	rule 27 DeMorgan's
6	¬d	5	rule 6 simplification
7	¬c	5	rule 6 simplification
8	¬b	6,2	rule 4 modus tollens
9	¬a	7,3	rule 4 modus tollens
10	¬a → b	1	rule 30 implication
11	b	9,10	rule 30 modus ponens
12	b ∧ ¬b ok	8,11	rule 10 conjunction

The q-matrix analysis of Proof 7 yielded rich insights into the types of proofs used by students, and was supported by the actual data. Further exploration of the q-matrix method used to data mine proofs continues in the next section discussing Proof 8.

7.2.8 Proof 8 Discussion

The problem to be solved for Proof 8 is given below. The complex given statements and the many ways students can apply rules to solve this proof resulted in the use of more rules than any other proof in our analysis. Students used nine rules in solving Proof 8, including Hypothetical Syllogism (1), Disjunctive Syllogism (2), Modus Ponens/Tollens (3+4), Constructive Dilemma (7+8), Tautology/Contradiction (25+26), Contrapositive (29), Implication (30+31), and Negation of Conclusion (26).

Proof 8: Given: $(d \vee b) \rightarrow a$, $c \vee d$, $\neg c \vee b$ Prove: a

The q-matrix extracted from our Fall 2002 data is given in Table 7-33. In this q-matrix we can see that the Implication rule relates to no concepts, suggesting that this rule was used in most responses. Only 2 questions related exclusively to a single concept, namely rule 1 relates to concept 1 and rule 3+4 relates to concept 3. Because of this, we'd expect that students employing concept 1 to use rule 1 and students using concept 3 to use rule 3+4. Since Negation of Conclusion relates to 2 concepts, we expect that students who solved this proof by contradiction employed both concepts 2 and 3. In addition, since DeMorgan's also relates only to these concepts, we expect that proofs by contradiction almost always used DeMorgan's. Direct proofs would probably be any combination of the four concepts, except those that combine both concepts 2 and 3.

Table 7-33 Proof 8 q-matrix

	1	2	3+4	7+8	25+26	27+28	29	30+31	36
Con 1	1	0	0	1	1	0	1	0	0
Con 2	0	1	0	1	1	1	0	0	1
Con 3	0	1	1	0	0	1	1	0	1
Con 4	0	1	0	1	1	0	1	0	0

Table 7-34 lists the student responses to Proof 8, grouped by concept state. Since we have 4 concepts to explain this model, there are 16 possible states that responses could have been assigned to. As in Proof 4, which also used 4 concepts, we do not use all 16 in this case. There were 5 unused states: 0010, 0011, 0100, 1001, and 1010. This suggests that concept 2 was most often used in conjunction with 2 or more other concepts to explain student responses. This makes sense since using this concept alone would result in a predicted answer vector of 000000010, the same as the all-zero state, because each rule this concept relates to is also related to at least one other concept. We also see that concept 2 combined with neither concepts 1 nor 4 were used by students. Again, this result makes sense since these combinations still predict the same IDR as the all-zero state. The remaining unused state is 1001, the combination of concepts 1 and 4. This combination brings no new values to the ideal response vector than those of concept 1 or 4 alone, so its use is not necessary.

From Table 7-34 we can see that there is great diversity in the responses to Proof 8. There are a total of 37 distinct response vectors, and no one vector has more than 28 students with that response. The concept states with the most assigned responses are: 1101, 0110, 1110, and 0101. These states all contain concept 3, which indicates that Modus Ponens/Tollens is a commonly used rule in this proof. Also we can note that 2 states combine concepts 2 and 3 with and without concept 4, and the other 2 combine concepts 1 and 3 with and without concept 4. This suggests that the most common responses are combinations of concepts 2 and 3, and 1 and 3, and we add in concept 4 when Disjunctive Syllogism or Contrapositive is used.

Table 7-34 Proof 8 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
0000	000000010	200	000000001	400	1	2	2
0001	100000010	201	100000000	001	4	1	4
			100000010	201	2	0	0
0101	101000010	205	101000010	205	17	0	0
			101000011	605	1	1	1
0110	001001011	644	001001001	444	1	1	1
			001011001	464	1	2	2
			001100011	614	1	2	2
			001001011	644	14	0	0
			001011011	664	12	1	12
			001001111	744	3	1	3
0111	001001011	644	101010011	625	1	3	3
			111001011	647	1	2	2
			101101011	655	1	2	2
1000	000000010	200	000010011	620	1	2	2
			000000111	700	1	2	2
1011	100110010	231	100110010	231	7	0	0
			101110010	235	1	1	1
			001111010	274	1	3	3
1100	001000010	204	001100010	214	1	1	1
			001110010	234	8	2	16
1101	101000110	305	101000110	305	28	0	0
			100100110	311	1	2	2
			101010110	325	1	1	1
			001110110	334	2	3	6
			101000111	705	5	1	5
			101010111	725	2	2	4
1110	011001011	646	011001001	446	13	1	13
			011011001	466	1	2	2
			011001101	546	3	2	6
			010010011	622	2	3	6
			011010011	626	1	2	2
			010001011	642	1	1	1
			011001011	646	5	0	0
			011011011	666	2	1	2
1111	111111111	777	011111001	476	1	3	3
			011111011	676	2	2	4
				tot:	150		116

The most common concept state is 1101, with an ideal response vector of 101000110 (o305), corresponding to the use of rules Hypothetical Syllogism (1), Modus Ponens/Tollens (3+4), Contrapositive (29), and Implication (30+31). An example proof using these rules is given in Table 7-35. A similar response of 101000010 (o205) replaces Contrapositive with an application of commutative before applying the Implication rule to the given in line 2, and this is the ideal response vector for state 0101.

Table 7-35 Proof 8 Example of Direct Proof

Given: (d v b) → a, c v d, ¬c v b   Prove: a

	STATEMENTS	REASONS	
1	(d v b) → a	Given	
2	c v d	Given	
3	¬c v b	Given	
4	¬c → d	2	rule 30 implication
5	¬d → c	4	rule 29 contrapositive
6	c → b	3	rule 30 implication
7	¬d → b	5, 6	rule 1 hyp. syllogism
8	d v b	7	rule 30 implication
9	a	1,8	rule 3 modus ponens

Table 7-36 Proof 8 Example of Proof by Contradiction

Given: (d v b) → a, c v d, ¬c v b   Prove: a

	STATEMENTS	REASONS	
1	(d v b) → a	Given	
2	c v d	Given	
3	¬c v b	Given	
4	¬a	Negation of Conclusion	
5	¬(d v b)	1, 4	rule 4 modus tollens
6	¬d ∧ ¬b	5	rule 27 DeMorgan's
7	¬d	6	rule 6 simplification
8	¬c → d	2	rule 30 implication
9	c	2, 8	rule 4 modus tollens
10	¬b	6	rule 6 simplification
11	¬c	3, 10	rule 2 disj. syllogism
12	c ∧ ¬c	9, 11	rule 10 conjunction

Another common concept state for Proof 8 is 1110, with a response vector of 011001011 (o646), where students used the rules: Modus Ponens/Tollens, DeMorgan's, Implication, and Negation of Conclusion. An example proof using these rules to solve Proof 8 is given in Table 7-36. Response 001001011 (o644), the IDR for concept state 0110, is a similar response that doesn't use Disjunctive Syllogism (2). In the example proof, we could apply Implication to line 3 before combining it with line 10 to get ¬c. Similarly, this proof could be easily modified not to use the implication rule.

Proof 8 yields a rich diversity of student answers, but from the q-matrix analysis we were able to understand how proof rules interact and can be substituted for one another, and to determine how the combination of concepts would affect the different concept states. We continue our analysis with Proof 9 in the following section.

7.2.9 Proof 9 Discussion

The problem to be solved for Proof 9 is given below. In this proof, students used a total of eight different rules, including: Disjunctive Syllogism (2), Modus Ponens/Tollens (3+4), Constructive Dilemma (7+8), Addition (19), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36). A student using the rules Constructive Dilemma, Addition, and Contrapositive would have a response of 00110100 since these are the third, fourth, and sixth of the 9 rules used to solve this proof.

Proof 9: Given: $\neg a \vee d$, $(a \wedge \neg c) \rightarrow b$, $\neg b$ Prove: $a \rightarrow (c \wedge d)$

The q-matrix extracted for Proof 9 solutions in Fall 2002 is given in Table 7-37. In this q-matrix, rules 3+4, 27+28, and 30+31 are not related to the extracted concepts, so the majority of students used these rules in their proofs. Rules 19 and 29, on the other hand, were related to all concepts, indicating that they were seldom used in student proofs. Concept 1 is distinguished by rules 2 and 36, meaning that responses in concept state 01 will be proofs by contradiction that use Disjunctive Syllogism (2). Concept 2 is independent from concept 1 only in rule 7+8, so responses in concept state 10 will probably be direct proofs using Constructive Dilemma. Since we have so many rules in common among all responses, it is not surprising to find a large number of responses assigned to concept state 00, as we see in Table 7-38.

Table 7-37 Proof 9 q-matrix

	2	3+4	7+8	19	27+28	29	30+31	36
Con 1	1	0	0	1	0	1	0	1
Con 2	0	0	1	1	0	1	0	0

Table 7-38 lists the response vectors collected in Fall 2002, ordered by their assigned concept states. As we predicted, concept state 01 contains students who solved Proof 9 by contradiction, and most students in this state also used Disjunctive Syllogism in their proofs. Concept state 10 consists of direct proofs, all using Constructive Dilemma along with those rules in common among most proofs. Concept state 00 consists of students using primarily the common rules: Modus Ponens/Tollens, DeMorgan's, and Implication. Concept state 11 contains only 3 students, who used most of the extracted rules.

Concept state 00 contains the rules that most students use, regardless of their concept states. An example of a complete proof using these "default rules" with answer vector 01001010 (o122) is given in Table 7-39. This proof uses rules Modus Tollens, DeMorgan's, and Implication. Other responses, such as o121, replace the use of Modus Tollens with Disjunctive Syllogism, or replace DeMorgan's with Contrapositive and Implication (o142). Some students simply skipped DeMorgan's, which is allowed in the tutorial, to get answer vector o102.

Many students combined the "default rules" with Constructive Dilemma to solve Proof 9. These students were in state 10, with an ideal response vector of 01101010 (o126). This type of proof would continue from line 6 in Table 7-39, but would use Implication to convert $\neg a \vee c$ and $\neg a \vee d$ into implications $a \rightarrow c$ and $a \rightarrow d$, and combine these using Constructive Dilemma to get the final result. As in state 00, students could also replace DeMorgan's with Contrapositive and Implication (o146) or also use the Idempotent rule (19) after applying a different version of Constructive Dilemma (o136).

Table 7-38 Proof 9 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
00	01001010	122	01000010	102	2	1	2
			10001010	121	2	2	4
			01001010	122	19	0	0
			01000110	142	1	2	2
			00000011	300	1	3	3
			00001011	320	1	2	2
01	11001011	323	10001011	321	4	1	4
			01001011	322	10	1	10
			11001011	323	13	0	0
			11101011	327	1	1	1
			01001111	362	1	2	2
			11001111	363	2	1	2
10	01101010	126	01100010	106	8	1	8
			01110010	116	2	2	4
			00101010	124	1	1	1
			10101010	125	2	2	2
			01101010	126	46	0	0
			01111010	136	10	1	10
			01100110	146	2	2	4
			01101110	166	10	1	10
11	11111111	377	10111010	135	1	3	3
			01111110	176	1	2	2
			10111011	335	1	2	2
				tot:	141		78

Proofs by contradiction generally added two rules to the "default rules," Disjunctive Syllogism (2), and Negation of Conclusion (36). This is seen in the ideal response vector for state 01, 11001011 (o323). An example proof using these rules is given in Table 7-36. The use of Disjunctive Syllogism is the simplest way to combine the results of Negation of Conclusion with the given statement on line 1; given a and ¬a ∨ d, the simplest way to combine these is rule 2.

Table 7-39 Proof 9 Example of Direct Proof

Given: $\neg a \vee d, (a \wedge \neg c) \rightarrow b, \neg b$ Prove: $a \rightarrow (c \wedge d)$

	STATEMENTS	REASONS	
1	$\neg a \vee d$	Given	
2	$(a \wedge \neg c) \rightarrow b$	Given	
3	$\neg b$	Given	
4	$\neg(a \wedge \neg c)$	2, 3	rule 4 modus tollens
5	$\neg a \vee c$	4	rule 28 DeMorgan's
6	$\neg a \vee (c \wedge d)$	1, 5	rule 16 distributive
7	$a \rightarrow (c \wedge d)$	6	rule 30 implication

Proof 9 is an example of a problem whose solutions truly called for a set of rules that is always used in its solution. First, the given statements were in different formats (both ors and implications), meaning that eventually the implication rule would be needed. Also, the number of negation signs in the proof made DeMorgan's a necessary step in most proofs. Finally, either Modus Ponens/Tollens or Disjunctive Syllogism was needed to combine the givens to eliminate unneeded variables in the conclusion. The q-matrix method was able to quickly show us this set of default rules by the ideal response vector for concept state 00. We were also able to quickly tell the main differences between proof by contradiction and direct proofs by simple examination of the q-matrix. We continue our discussion of the use of the q-matrix method in data mining proof solutions in the following section.

Table 7-40 Proof 9 Example of Proof by Contradiction

Given: ¬a v d, (a ^ ¬c) → b, ¬b   Prove: a → (c ∧d)

	STATEMENTS	REASONS	
1	¬a v d	Given	
2	(a ∧ ¬c) → b	Given	
3	¬b	Given	
4	¬(a → (c ∧ d))	Negation of Conclusion	
5	a ∧ ¬(c ∧ d)	4	rule 31 implication
6	a	5	rule 6 simplification
7	¬(c ∧ d)	5	rule 6 simplification
8	d	1, 6	rule 2 disj. syllogism
9	¬c v ¬d	7	rule 28 DeMorgan's
10	¬c	8, 9	rule 2 disj. syllogism
11	a ∧ ¬c	6, 10	rule 10 conjunction
12	b	11, 2	rule 3 modus ponens
13	b ∧ ¬b	3, 12	rule 10 conjunction

7.2.10 Proof 10 Discussion

The problem to be solved for Proof 10 is given below. Although this proof appears simple, yet again students used a variety of approaches in its solution. Students used a total of seven rules in Proof 10 solutions, including: Hypothetical Syllogism (1), Disjunctive Syllogism (2), Modus Ponens/Tollens (3+4), DeMorgan's (27+28), Contrapositive (29), Implication (30+31), and Negation of Conclusion (36). A response containing rules 2, 3+4, 29, and 36 would then have response vector 0110101.

Proof 10: Given: a → (b ∨ c), b → d   Prove: a → (c ∨ d)

The q-matrix extracted for Proof 10 in Fall 2002 is given in Table 7-41. Here, rule 30+31 relates to no concepts, being used frequently, while rules 2 and 29 relate to all concepts, being used sparingly in student proofs.

Table 7-41 Proof 10 q-matrix

	1	2	3+4	27+28	29	30+31	36
Con 1	0	1	1	1	1	0	1
Con 2	1	1	0	0	1	0	0

The remaining proof rules are split between the two concepts, suggesting that concept 1 users preferred rules 3+4, 27+28, and 36. This means that proofs by contradiction generally used rules Modus Ponens/Tollens, DeMorgan's, Implication, and Negation of Conclusion. Concept 2 users preferred rule 1, so their direct proofs would include rules Hypothetical Syllogism and Implication. Responses in concept state 11 would use a combination of these rules. Students who used few rules other than Implication are placed in state 00.

Student responses are shown according to their concept states in Table 7-42. As predicted, most direct proof solutions are in concept state 10, using Hypothetical Syllogism and Implication, for an ideal response vector of 1000010 (o041). Most proofs by contradiction were assigned to concept state 01, and were distinguished by their use of the four rules discussed above. In other proofs, we've often seen most large sets of student responses separated into separate concept states, where here, our two most popular responses, o154 and o156, are both placed in concept state 01. Since these responses differ in the use of only one rule (Disjunctive Syllogism), and we had only 4 concept states, it makes sense that these two large responses were assigned to the same concept state. Were we to run a q-matrix analysis with more concepts, we'd expect an additional concept state to differentiate these two groups to minimize error. This does happen eventually, but not until the method is run using 4 concepts.

Table 7-42 Proof 10 Ideal and Student Response Vectors

Concept State	IDR	Octal Reverse	Actual Responses	Octal Reverse	# Students	Error	Err * # Stud
00	000010	040	0000010	040	7	0	0
			0001010	050	1	1	1
			0010110	064	2	2	4
			0000011	140	1	1	1
01	0011011	154	0010011	144	2	1	2
			0001011	150	3	1	3
			0101011	152	3	2	6
			0011011	154	32	0	0
			1011011	155	3	1	3
			0111011	156	27	1	27
			0011111	174	6	1	6
10	1000010	041	1000000	001	7	1	7
			1000010	041	21	0	0
			1001010	051	9	1	9
			1000110	061	8	1	8
11	1111111	177	1101011	153	1	2	2
			1011111	175	3	1	3
			0111111	176	2	1	2
				tot:	138		84

As with each proof, we give two sample proofs to verify that the extracted ideal response vectors are valid proof solutions. A direct proof solution using response 1000010 (o041) is given in Table 7-43. Because of the format of the given statements, repeated uses of the implication rule were necessary to manipulate the result to the desired form. Some solutions in the same concept state used only Hypothetical Syllogism for this proof (o001), implying that these students probably used rules that were not extracted to manipulate the forms of each statement. For example, a special case of Constructive Dilemma can be applied to $b \rightarrow d$ to get $(b \vee c) \rightarrow (d \vee c)$ and we can combine this statement with line 1 and Hypothetical Syllogism to arrive at our result.

Proofs by contradiction were generally placed in concept state 01, whose ideal response vector is 00110110 (o154). These proofs applied rules Modus Ponens/Tollens, DeMorgan's, and Implication in addition to Negation of Conclusion. The proof given in Table 7-44 demonstrates the other primary answer vector of 01110110 (o156) in concept state 01, which also uses Disjunctive Syllogism. (This proof could be modified to replace Disjunctive Syllogism with Implication and Modus Ponens/Tollens, this demonstrating the most common answer vector of 00110110.) This proof shows that these rules must be used in concert to arrive at a contradiction.

Table 7-43 Proof 10 Example of Direct Proof

Given: a → (b v c),b → d Prove: a → (c v d)

	STATEMENTS	REASONS	
1	a → (b v c)	Given	
2	b → d	Given	
3	¬a v b v c	1	rule 30 implication
4	¬a v c v b	1	rule 12 commutative
5	¬(¬a v c) → b	4	rule 30 implication
6	¬(¬a v c) → d	5, 2	rule 1 hyp. syllogism
7	¬a v c v d	6	rule 30 implication
8	a → (c v d)	7	rule 30 implication

Table 7-44 Proof 10 Example of Proof by Contradiction

Given: a → (b v c),b → d Prove: a → (c v d)

	STATEMENTS	REASONS	
1	a → (b v c)	Given	
2	b → d	Given	
3	¬(a → (c v d))	Negation of Conclusion	
4	¬(¬a v (c v d))	3	rule 30 implication
5	a ∧ ¬(c v d)	4	rule 27 DeMorgan’s
6	a ∧ ¬c ∧ ¬d	5	rule 27 DeMorgan’s
7	a	6	rule 6 simplification
8	¬c ∧ ¬d	6	rule 6 simplification
9	¬c	8	rule 6 simplification
10	¬d	8	rule 6 simplification
11	b v c	1, 7	rule 3 modus ponens
12	¬b v d	2	rule 30 implication
13	b	9, 11	rule 2 disj. syllogism
14	¬b	10, 12	rule 2 disj. syllogism
15	b ∧ ¬b	13, 14	rule 10 conjunction

The q-matrix method was successful in determining sets of rules used to solve Proof 10, separating direct and indirect proofs into two main concept states, and finding those rules that were common among most student solutions to Proof 10. We also found that response vectors falling into each concept state were similar enough to the ideal response vectors to feel confident that these different responses were variations on the main themes. In the following section, we discuss the strengths and weaknesses of the q-matrix method as a data mining tool for proof solutions.

7.2.11 Q-matrix method as a data mining tool

The q-matrix method evolved as a process for data mining student responses to tutorials to build concept models of the topic area. A novel and much less obvious application of the q-matrix method is its application as a general data mining tool in creating concepts which are not either "known" or "unknown," but are simply relationships inherent in an observed data set. In this chapter, we have discussed our application of the q-matrix method for data mining student proofs. The results of these analyses have shown that the q-matrix method can, in fact, be used to data mine proofs to find rule sets that are important in solving proofs. This application of the q-matrix method as a general data mining tool is one of the major contributions of this dissertation.

As we have discussed in the Introduction, data mining contains several steps, including data selection, data clustering, and cluster interpretation. Our hypothesis was that the q-matrix method would be and effective tool in combining the data clustering and cluster interpretation steps. For each of the proofs in the Proofs Tutorial program on NovaNET, we applied the q-matrix method to find a q-matrix for each, and used these q-matrices to make predictions about the data we extracted them from. The ease of interpretation of a q-matrix of zeroes and ones allowed us to determine the common responses in each cluster formed in the q-matrix analysis, and their interactions in the q-matrix allowed us to predict which concept states would or would not be used in the cluster assignments.

In these proofs, we were able to easily see the main approaches to proof solutions by simple examination of our q-matrices. When few concepts were used, we were able to quickly get an idea of the main rules used and not used in student proofs. When more concepts were used, we were also able to determine more subtle differences among solutions, such as how the use of one rule would place a response in a different cluster.

Our observations for each proof were supported by a table demonstrating the student responses and their assignments to concept states, which correspond to clusters in a data mining approach. These tables were used to verify that the predicted responses for each concept state corresponded well to the data in each of these states, and to expose the different responses that were used in each solution. The number of clusters, and therefore concepts, for each proof was determined individually for each proof by our error criteria of an average of less than one error per response, assuring that data in each concept state, or cluster, were similar enough to be placed together. As we saw in most proofs, the error for a particular response was never more than 3, and was usually 0 or 1.

Along with a table of student responses, we've included sample student solutions for each proof that demonstrate the use of ideal response vectors in the most populated concept states for each proof. An important outcome of this study was that the ideal response vectors are valid proof solutions, since we were looking for general sets of rules that could be used in conjunction to create proof solutions.

In our discussion of Proof 3, we demonstrated some of the differences that occur in our q-matrix model when we use varying numbers of concepts for extraction. We found that, although one number of concepts was best when we wanted low error per student, models with fewer concepts could also be quite useful in understanding the general use of rules in student proofs. In addition, the models constructed in successively larger q-matrix analyses usually are similar to previous models, but offer better predictions for large subsets of the data. As in most data mining methods, the larger the number of clusters we allow, the more precise our cluster assignments will be. It's comforting to know, though, that even though the q-matrix method is a heuristic method, the extracted concepts in successive models are quite similar, and the cluster assignments do not make drastic jumps. This shows that the method is robust and can offer good and reliable results even when the optimal number of concepts is not known.

Our heuristic of log (base = number of variables) of the number of responses turned out to be a pretty good measure of the number of concepts extracted for each proof. The validity of this heuristic in other applications of the q-matrix method will have to be checked, but as in other clustering and latent variable analysis methods, our choice of abstraction level will determine the size of our model.

These analyses were particularly useful to us in understanding the general use of rules in student proofs. Teachers who teach logic proofs are familiar with basic concepts and probably have a good sense of when several rules can be replaced with other equivalent sets of rules, but the q-matrix analysis of these 10 proofs has demonstrated their actual application in proof solutions. Some proofs, such as proof 3, were manipulations of the order and format of the given statements, while most others required a combination of the given statements to eliminate several variables to arrive at the proof conclusion.

Even in the course of a more complex proof where variable elimination is necessary, "formatting" rules such as DeMorgan's, Implication, and Contrapositive were very important in transforming statements into forms useable by the “elimination” rules Modus Ponens/Tollens, Hypothetical Syllogism, and Disjunctive Syllogism. These rules are differentiated on our Axiom List (see Appendix D) as logical implications, instead of logical equivalences.

We know that these rules are important in solving complex proofs, and can often be interchanged, but the exciting finding is that the q-matrix models extracted for these proofs often demonstrated when students could substitute one method for another, as we see when Disjunctive Syllogism is used for some students while others use Modus Ponens/Tollens to solve the same proof. These rules, along with Negation of Conclusion, were often those that differentiated concepts from one another. This indicates that our intuitive knowledge that rules can be interchanged is often found by the q-matrix method.

In conclusion, we believe that the q-matrix method offered a way to examine and predict student proof solutions without having to examine student data. The zero-and-one q-matrix model and its easy conversion into predicted responses allowed us to understand concepts and concept states without actually reading student proofs.

8 Summary

In the first section of this chapter, we briefly discuss the material presented in this dissertation. We then discuss the conclusions and contributions of this work, and finally, end by listing some promising areas for future work in the area of q-matrix analysis and data mining.

8.1 Chapter Overviews

Chapter 1 motivated our research, discussing the growing need for automated, individualized instruction that is inexpensive, expandable, and accessible. We also listed the qualities that such educational tool would need, finding a commonality in assessment and teaching needs with data mining. We then introduced the idea of Fault Tolerant Teaching and how the q-matrix method could be used to augment a computer-based tutorial with a student knowledge model. This model could be used both for student knowledge assessment and for evaluating tutorial questions. We also suggested that this method might be used as a data mining tool, performing the tasks of clustering data and also extracting a data model that explained the clusters in an easy-to-interpret way.

The q-matrix method is the product of evolution of several research threads over the years. In Chapter 2, we discuss the evolution of the ideas and research that led to the development of the q-matrix method, from procedural models of knowledge, to the modern statistical method for building our concept models. Tatsuoka's first use of the rule-space model to diagnose student misconceptions in basic math represented a revolutionary idea in the field of education: that we can use a concept of distance from expected types of performance to determine what strategies a student uses to solve a problem. This idea eventually led to the q-matrices of concepts versus questions that we use today. A detailed explanation of the q-matrix method and the foundational research in its use as an educational tool are given in this chapter. Also in this chapter, we discuss some of the basic qualities of data mining, and its importance in today's informational world. One contribution of this research is the novel application of the q-matrix method as a data mining tool.

Chapter 3 discusses the overall design of this experiment, including a model for Fault Tolerant Teaching and the application of the q-matrix method in this model. Also in this section we outline our data collection methods, the population we analyzed, and the results we expected to find, based on our research hypotheses.

After deciding to use a q-matrix model for concepts versus questions, there are still many ways we can arrive at such a model given a set of student data. In Chapter 4, we discuss three methods to derive a q-matrix model of questions verses concepts, including the hill-climbing heuristic algorithm devised by Patrick Brewer, factor analysis (as used by social scientists and others in extracting latent variables from observed ones), and cluster analysis, a tool often used in data mining. Our results showed that the q-matrix method had several advantages over both factor analysis and cluster analysis, yet also showed commonalities with these methods that suggest that the q-matrix method is suitable for applications in other areas.

As discussed in depth in the Introduction, knowledge assessment is a very important aspect of every adaptive tutorial system. Ours is no exception, so Chapter 5 is devoted to Knowledge Assessment. In this chapter, we used three approaches to determine the validity of the q-matrix method used as a knowledge assessment tool. First, we compared student grades on their final exams with their scores on the tutorials, to determine if there were a relationship between these scores. We found no significant relationships between most scores on the tutorial and the final exam, but determined that this was probably due to the simple nature of most of the tutorial questions, and the fact that the final exam was an open-book take-home exam. In this case students could look up definitions and would be likely to get simpler questions correct.

The second section of Chapter 5 compares the q-matrices extracted from the Binary Relations tutorial from those predicted by area experts (instructors of the Discrete Mathematics course). In general we found that expert q-matrices did not correspond well to those extracted from student data. This was not unexpected, since in the origin of q-matrix theory researchers found that expert knowledge models did not correspond with student data with any higher probability than randomly generated models [Hub92]. In fact, as the size of a q-matrix grows, predicting the concept relationships among the observed variables or questions becomes a much more complex process, and even experts may have trouble constructing useful relationships in this case. In this sense, the q-matrix method is truly a data mining tool, finding relationships latent in data that even area experts might not be able to see.

In the last section of Chapter 5 we compare the questions chosen for remediation by the q-matrix method with those chosen by individual students, to determine if student perception of what they misunderstood corresponded with our q-matrix predictions. Since these two were found to correspond well, we believe that our remediation method will be at least as good as a student choosing which concepts to review. In some cases, the q-matrix method indicated that a student should review a topic that the student did not review, and later scores on their exams showed that these students would have done well to review the topics suggested by the q-matrix method.

Along with each tutorial, the students were asked to complete a survey of their experience with the tutorial, to compare its effectiveness with that of homework and lectures, and also to determine areas in the tutorials that could use improvement. In Chapter 6, we discuss the results of these surveys. In general, we found that students found the tutorials to be effective in learning and reinforcing the topics they learned in class, particularly by providing extra practice and examples that expanded their understanding of the tutorial topics. Students appreciated another form of learning and several expressed wishes for tutorials to be developed for all topic areas in their discrete math class. Some students did have trouble with the delivery system, such as inputting answers and learning to navigate, but these types of problems are often inherent in using a new tool for teaching and for the solutions to mathematics problems. In general, students felt that the Binary Relations Tutorial was able to determine what they did not understand and direct them for more study.

We see the application of the q-matrix method as a data mining tool in Chapter 7, where we use the method to build models of axioms used to solve proofs. In this chapter, we found the q-matrix method to be an effective tool in both clustering and understanding groups of student responses. Using the extracted q-matrices, we were able to make predictions about the student responses that were supported by student data. Using the predicted response vectors, we could also almost always create proofs that used only the predicted responses in the solution. This data analysis yielded a much better understanding of the influence of student data on the creation of a q-matrix. We found that large clusters of similar student responses skewed the q-matrix model towards these responses, but that a balance was still maintained by keeping the number of concepts extracted low, so that we never created a model that could not accommodate and explain new student responses at a later time.

In the proofs analysis of Chapter 7, and in our other q-matrixes in this experiment, we found each extracted q-matrix to converge on binary values, even though the q-matrices were allowed to vary as probabilities between 0 and 1. This finding leads us to believe that good q-matrix models will usually converge to binary values, making the q-matrix method very valuable in understanding a data set. If q-matrix values were to converge to probabilities between 0 and 1, we would have a much harder time predicting binary student responses and understanding the relationships among concepts and variables in the analysis.

8.2 Conclusions

To guide our conclusions, let us reconsider the questions we raised in Chapter 1:

- Can we effectively guide remediation?
- Can we interpret q-matrix scores?
- What factors affect q-matrix extraction and accuracy?
- How well does the q-matrix method compare with other data mining methods?
- How do extracted q-matrices compare with expert-constructed q-matrices?

In this research, we sought to determine if the q-matrix method would be able to effectively guide remediation of student knowledge. For those students who guided their own study, we found that the q-matrix method often chose the same topics for review as they chose for themselves. We also found that students who did not choose to study the topics that would have been suggested by the system could have benefited from reviewing those topics, since their exam scores on those questions was low. However, due to the final sample size of this group of students, these findings are preliminary and not conclusive. Subjective results showed that a large portion of students felt that the Binary Relations tutorial was able to determine what they least understood. Although our results in this area are not conclusive, we have also provided a framework and design for future educational experiments that can compare methods of choosing topics for student review.

In answer to our second question, we have found that binary q-matrix scores can be simply and quickly interpreted by a person to determine predicted answer vectors and the variables that were most important in the q-matrix creation. We found that the behavior of a variable is demonstrated not only by the single concept-variable relationships, but is also very much determined by the combination of these concept relationships. For example, our proof analyses all began by considering those variables that were related to either all or none of the extracted concepts, since the predicted values for these variables would be determined by the state of the whole column, and not just one row. So, although the q-matrix model creates and forces us to consider multi-concept relationships, with practice we were able to more clearly understand the relationships they described.

Our fourth question asks what factors that affect the accuracy of extracted q-matrices. Here, we define accuracy as the total error over all responses when these responses are compared to predicted responses. First, we found that not only the total number of observations but also the number of distinct observations used in the extraction determined the performance of the q-matrix method, especially when compared with the factor analysis extraction method. In the case of fewer distinct observations, the q-matrix was able to model the data with much better accuracy than the factor method. As the number of total observations and distinct observations increased, the lead the q-matrix method has over the factor method closes.

In determining factors that affect the q-matrix method, we also found that, when there were truly no inherent relationships in the data, such as in Section 1 of the Proofs tutorial, the q-matrix method as we applied it never converged on a solution. We believe this is because of the scattered nature of the student responses in this section. As with other data modeling and analysis tools, the effectiveness of a model is based on whether the assumptions needed for the model correspond well with the observed data. Since there are really no concepts relating the questions in this tutorial, we should not expect to be able to find a q-matrix that predicts student responses well.

Our fifth question asks how the q-matrix method compares to other data mining methods. In this research, we found that the models created by the q-matrix analysis fit student data well, created meaningful clusters, and was useful in the interpretation step of a data mining process. The q-matrix method modeled student data better than factor analysis, and not as well as cluster analysis. However, its added benefit in interpretation, we felt, makes the q-matrix method a better choice when attempting to both cluster and understand a data set.

Our final question was an investigation that extends previous research in q-matrix theory: whether extracted q-matrices compare well with expert q-matrices. We found that these two models do not necessarily correspond well, as we predicted based on previous research findings.

The hypotheses we offered in our introduction are given below:

- Q-matrix scores will converge to zero and one values.
- Q-matrix scores will demonstrate latent relationships among questions.
    - Factors that might affect the extraction of q-matrices are: the number of questions, the redundancy of questions and the existence of relations among the questions, etc.
- Q-matrix diagnosis of concept states will correspond well with human knowledge assessment. In other words, student misconceptions could be diagnosed at least as well as students themselves could determine which topics they misunderstood.
- The q-matrix method provides a good guide for remediation.
- The q-matrix method can perform the data mining task of data clustering and assist in the resulting cluster interpretations.

Each of these hypotheses was confirmed in this research, though further research is needed to determine the effectiveness of the remediation method we chose. However, this research did demonstrate that we could choose and implement a teaching strategy, and measure its effectiveness. Future research will be needed to find the best remediation strategy for teaching.

This experiment was designed to test the q-matrix method in several ways. First, the q-matrix method was used as a tool to create a student knowledge model. The nature of the q-matrix method causes the extracted q-matrices to fit the observed data as well as a researcher wishes. In this research, we wished to fit the data so that student responses were predicted with an error of about one per student. When compared with other methods of knowledge model extraction, including factor analysis and cluster analysis, the q-matrix method was superior in that it was able to fit the observed data well, while still offering the interpretability we need to devise remediation methods for students.

The second aspect of this research was to determine the effectiveness of the q-matrix method as a tool in knowledge assessment. Our final goal was not to determine a grade for a student, but a prediction of the next step needed in a student's learning process. We were able to achieve this goal, leading most students to review topics they misunderstood in each tutorial.

As done in previous work, we also compared extracted q-matrix models to expert models, and found that the extracted and expert q-matrices were not a good match, but that extracted q-matrix models were quite useful in understanding student data. This shows, as in previous research, that expert models do not necessarily predict student behavior and more accurate student knowledge models, such as q-matrices, are needed to understand student knowledge. In addition, using an expert-created q-matrix to analyze tutorials created by the same expert will not necessarily find the flaws in that tutorial, while an extracted q-matrix can reveal student behavior that might not be predicted by an expert's understanding of the tutorial topic.

In our Proofs Analysis chapter, we devised q-matrices as data mining tools, used to extract the axioms needed to solve a proof. Although we discussed this method as a data mining method, it also revealed different groups of students, giving us an idea of which rules certain groups of students understood better. For example, when a direct proof required the use of a rarely-used rule, students most often solved the proof using an indirect proof. In other cases, concepts were differentiated by single rules, and a student's preference for one rule over another would place him in a different concept state. These analyses can help us understand in which rules students might need more instruction, and which are well understood.

8.3 Future Work

This research lays the groundwork for future work in fault tolerant teaching and computer-based education. Any online lesson with the capability to assess student answers can be augmented with the q-matrix method of FTT. As FTT methods are applied to more lessons and more students, they can be further validated and improved.

Fault tolerant teaching methods can be applied to other topics, not limited to mathematics, since they rely only on the correctness or incorrectness of student answers. They may be applied to multiple-choice tests and much more open-ended questions. These results are limited only by the automation of judging responses, and not by the type of student response. Future work could extend the model to include partial credit for student responses.

Future experiments can compare teaching strategies by applying different teaching strategies to different groups of students and comparing the time students take to master a set of concepts. Future research also might test the effectiveness of different strategies on individual students, comparing the change in student knowledge before and after each strategy is applied.

In a similar extension to this work, q-matrix methods might be used to extract other student characteristics, such as student learning style. Such an experiment could determine students' learning styles using a test such as the Felder-Silverman Index of

Learning Styles independently, and administer a lesson with two styles, such as visual and verbal examples, and attempt to extract students' learning styles using the q-matrix extraction method. This type of experiment would suppose that a student's learning style has an effect on question responses similar to the effect of "concepts" supposed in the current research.

Automated student knowledge assessment can be applied in research to understand the changes in student knowledge as they learn. In this case, researchers would map student knowledge states after each question or group of questions to track changes. This could be used in several different ways. First, it could be used to determine the most effective areas of a tutorial, and identify those that are not causing changes in student knowledge. It could also be used to further understand the process of human learning, assessing whether students are learning in great leaps or with small, gradual changes. Alternatively, this could also be used to redirect a student's learning. Tracking changes in student knowledge can alert an automated system to change its strategy.

Another major future application of this work is in data mining and latent variable analysis. Since q-matrix extraction techniques in the research yielded interpretable results, this can be applied in other areas, such as data mining, knowledge discovery, and latent variable analysis. To date, these areas use many different artificial intelligence techniques to detect underlying relationships in data. As an example, a search engine on the web could record the occurrence of all keywords in a document, and use q-matrix analysis to find concepts that relate keywords, and related documents to these concepts.

Then, when a user entered a keyword search, the search engine can look for concepts and not just word occurrences.

In conclusion, this research has provided an in-depth analysis of the factors that affect the use and interpretation of q-matrix models in online tutorials and also as a data mining tool. From our findings, we conclude that the q-matrix method of fault tolerant teaching is robust, extracting data relationships similar to those found using other data mining techniques, while also providing a tool to interpret these extracted relationships. These findings, the application of the q-matrix method in diagnosing student misconceptions, and its application as a data mining tool as we performed with the Proofs tutorial analysis make a significant contribution to the fields of computer based education, fault tolerant teaching, and data mining.

References

[AA96] AAHE Assessment Forum. (1996). *Nine principles of good practice for assessing student learning*. Online. [http://www.aahe.org/principl.htm]

[Air91] Airaisian, P.W. (1991). *Classroom assessment*. New York: McGraw-Hill.

[Alp70] Alpert, D. & Bitzer, D.L. (1970). Advances in computer based education, *Science*, 167, 1582-1590. (Sellers)

[Alt03] Alta-Vista Online Search Engine. (2003). Online: [http://www.altavista.com].

[Ash89] Ashcraft, M.H. (1989). *Human memory and cognition*. Glenview, IL: Scott Foresman.

[Bir93] Birenbaum, M., Kelly, A., & Tatsuoka, K. (1993). Diagnosing knowledge state in algebra using the rule-space model. *Journal for Research in Mathematics Education*, 24(5), 442-459.

[Bit73] Bitzer, M.D. & Bitzer, D.L. (1973). Teaching nursing by computer: An evaluation study. *Computers in Biology and Medicine*, 3: 187-204.

[Blo84a] Bloom, B. (1984). Taxonomy of educational objectives. *Cognitive Domain*. New York: Longman.

[Blo84b] Bloom, B. (1984). The 2 sigma problem: the search for methods of group instruction as effective as one-on-one tutoring. *Educational Researcher*, 13, 4-16.

[Bra89] Bransford, J., Vye, N., Adams, L., & Perfetto, G. (1989). Learning skills and the acquisition of knowledge. In A. Lesgold and R. Glaser (Eds.), *Foundations for a Psychology of Education*. Hillsdale, NJ: Lawrence Erlbaum Associates.

[Bre95] Brewer, P., Gibson, E., Dholakia, A., Vouk, M., & Bitzer, D. (1995). *Advanced Computer Based Education on the World Wide Web*. NCSU Computer Science Technical Report, August 1995.

[Bre96] Brewer, Patrick. (1996). Methods for concept mapping in computer based education. Computer Science Masters Thesis, North Carolina State University.

[Bro78] Brown, J.S. & Burton, R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, 2, 155-191.

[Bro80] Brown, J.S. & VanLehn, K. (1980). Repair Theory: A generative theory of bugs in procedural skills. *Cognitive Science*, 4,379-426.

[Bur82] Burton, R. (1982). Diagnosing bugs in a simple procedural skill. In D.H. Sleeman & J.S. Brown (Eds.) *Intelligent tutoring systems*. London: Academic Press.

[Bur88] Burns, H, & Capps, C. (1988). Foundations of intelligent tutoring systems: An introduction. In M.C. Polson & J. Jeffrey Richardson (Eds.) *Foundations of intelligent tutoring systems* (pp. 1-19). Hillsdale, NJ: Lawrence Erlbaum Associates.

[Cha87] Charles, R., Lester, F., & O'Daffer, P. (1987). *How to evaluate progress in problem solving*. Reston, VA: National Council of Teachers of Mathematics.

[Chi89] Chi, M., Bassok, M., Lewis, M., Reimann, P., & Glaser, R. (1989). Self explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, 13, 145-182.

[Chi87] Chickering, A. & Gamson, Z. (1987). Seven principles for good practice in undergraduate education. *AAHE Bulletin*, 39.7, 3-7.

[Cla86] Clancey, W. (1986). Intelligent tutoring systems: a tutorial survey. *STAN-CS-87-1174*. Palo Alto, CA: Stanford University.

[Cle93] Cleveland, W.S. (1993). *Visualizing Data*. Summit, New Jersey: Hobart.

[Dav03] Davidovic, A., Warren, J., & Trichina, E. (2003). Learning bebefits of structural example-based adaptive tutoring systems. *IEEE Transactions on Education*, vol. 46, no. 2, May 2003.

[Dav90] Davis, R., Maher, C., & Noddings, N. (Eds.) (1990). *Constructivist views on the teaching and learning of mathematics*. Reston, VA: National Council of Teachers of Mathematics.

[Dav97] Davis, R. & Maher, C. (1997). How students think: The role of representations. In L. English (Ed.), *Mathematical reasoning: Analogies, metaphors, and images* (pp. 93-115). Hillsdale, NJ: Lawrence Erlbaum.

[Die91] Dietel, R.J., Herman, J.L., and Knuth, R.A. (1991). *What does research say about assessment?* NCREL, Oak Brook. Online. [http://www.ncrel.org/sdrs/areas/stw_esys/4assess.htm]

[Ebe91] Ebel, R.L. & Frisbie, D.A.(1991). *Essentials of educational measurement*. Englewood Cliffs, NJ: Prentice-Hall, Inc.

[Edd84] Eddins J.M. (1984). *Dimensionality, scoring, and related problems in adaptive testing*. Computer-Based Education Research Laboratory, University of Illinois at Urbana-Champaign, January 1984.

[Ede98] Edelstein, H. (1998). Quoted in Wilson, R. Beware of fools' gold in the data mine. *Canberra Times*, Tuesday, November 10, 1998, p. 5.

[Eld98] Elder, J.F., & Abbott, D.W. (1998). A comparison of leading data mining tools. *Fourth International Conference on Knowledge Discovery and Data Mining*. Online. [http://www.dataminglab.com].

[Edu98] *Educational Leadership*, Issue on "How the Brain Learns," Vol. 56 (3), November 1998.

[Erl02] Erlich, Z., Gelbard, R. & Spiegler, I. (2002). Data Mining by Means of Binary Representation: A Model for Similarity and Clustering. *Information Systems Frontiers*, 4(2): 187-197, July 2002.

[Fay98] Fayyard, U. (1998). Editorial. *Data Mining and Knowledge Discovery* 2:5-7.

[Fay96] Fayyard, U.M., Piatetsky-Shapiro, G. & Smyth, P. (1996). From data mining to knowledge discovery: An overview. In Fayyard, U.M., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. *Advances in knowledge discovery and data mining*, pp. 1-34. Cambridge, Massachusetts: AAAI Press/MIT Press.

[Fla85] Flavell, J.H. (1985). *Cognitive development*. Englewood Cliffs, NJ: Prentice-Hall.

[Fra96] Frasson, C., Gauthier, F., & Lesgold, A. (1996). Intelligent Tutoring Systems. Third International Conference, ITS. Montreal, Canada: Springer.

[Fri97] Friedman, J.H. (1997). Data mining and statistics: What's the connection? *Proc. of the 29th Symposium on the Interface: Computing Science and Statistics*, May 1997, Houston, Texas.

[Fri98] Friedman, J.H. (1998). *Statistics 315B: Statistical Aspects of Data Mining* (Winter 1998). Online. [http://www.stanford.edu/~jhf/Stat315B.html]

[Gar89] Garfield, J., & Ahlgren, A. (1989). Difficulties in learning basic concepts in probability and statistics: Implications for research. *Journal for Research in Mathematics Education*, 19(1), 44-63.

[Ger00] Gertner, A. & VanLehn, K. (2000). Andes: A coached problem solving environment for physics. In Gauthier, G., Frasson, C., & VanLehn, K. (Eds.), *Intelligent Tutoring Systems: 5th International Conference, ITS 2000* (pp.131-142). Berlin: Springer.

[Goo03] Google Online Search Engine. (2003). Online: [http://www.google.com].

[Gro94] Gros, B., & Spector, J.M. (1994). Evaluating automated instructional design systems: A complex problem. *Educational Technology*, 34(5), 37-46.

[Hal01] Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On Clustering Validation Techniques. *Journal of Intelligent Information Systems*, 17:2/3, 107-145.

[Ham85] Hambleton, R.K. & Swaminathan, H. (1985). *Item response theory: Principles and applications*. Kluwer-Nijhoff Publishing. Boston.

[Hua98] Huang, Zhexue. (1998). Extensions to the k-means Algorithm for Clustering Large Data Sets with Categorical Values. *Data Mining and Knowledge Discover,* vol. 2, 283-304.

[Hub92] Hubal, Robert. (1992). Retro-adaptive testing and evaluation system. Computer Science Masters Thesis, North Carolina State University.

[Hum95] Hume, Gregory. (1995). Using student modeling to determine when and how to hint in an Intelligent Tutoring System. Computer Science PhD Dissertation, Illinois Institute of Technology.

[Ill91] The Illinois Universities *Test of College Preparatory Mathematics Skills*, Part I, Proceedings of International Educational Statistics and Measurement Symposium, April 18-20, 1991, Taiwan.

[Jon96] Jones, Susan. (1996). Computer assisted learning of mathematical recursion. Computer Science Masters Thesis, North Carolina State University.

[Kav00] Kavcic, A. (2000). The role of user models in adaptive hypermedia systems. *IEEE Mediterranean Electrotechnical Conference 2000*, Vol I, 119-122.

[Kav02] Kavcic, A., Privosnik, M., Marolt, M. & Divjak, S. (2002). Educational hypermedia: an evaluation study. *IEEE Mediterranean Electrotechnical Conference (MEleCon 2002)*, Cairo, Egypt, May 7-9, 2002.

[Kha02] Khalifa, M. & Lam, R. (2002). Web-based learning: effects on learning process and outcome. *IEEE Transactions on Education*, 45(4), 350-356.

[Kli94] Kline, Paul. (1994). *An easy guide to factor analysis*. London: Rutledge.

[Kol98] Kolsky, J. (1998). *Statistics and data mining in the analysis of large data sets*. Online. [http://www.infosense.com/news/article/article1.html]

[Lin87] Lindheim, E., Morris, L.L., & Fitz-Gibbon, C.T. (1987). *How to measure performance and use tests*. Newbury Park, CA: Sage Publications.

[Mar95] Martin, J. & VanLehn, K. (1995). Student assessment using Bayesian nets. *International Journal of Human-Computer Studies*, Vol. 42, pp. 575-591. Academic Press.

[Mei01] Meila, M. & Heckerman, D. (2001). An experimental comparison of model-based clustering methods. *Machine Learning*, 42, 9-29.

[Mis96] Mislevy, R.J., & Githomer, D.H. (1996). The role of probability-based inference in an Intelligent Tutoring System. *User Modeling and User-Adapted Interaction*, 5, 253-282.

[Mit97] Mitchell, T. (1997). *Machine learning*. New York: McGraw-Hill.

[Mor84] Morimoto, Y., & Tatsuoka, K. (1984). An interactive diagnostic system for fraction addition. Computer-Based Education Research Laboratory, University of Illinois at Urbana-Champaign.

[NCES98] National Center for Education Statistics. (1998). *Pursuing excellence: A study of US twelfth-Grade mathematics and science achievement in international content*. US Department of Education. Office of Educational Research and Improvement. [http://nces.ed.gov/timss/twelfth/98049.pdf]

[NLE98] National Library of Education. (1998). Riley urges students to take tougher courses: Challenges schools and states to raise academic, testing, and teaching standards in math and science. [http://nces.ed.goc/pressrelease/timss298.html]

[Ove94] Overbaugh, R.C. (1994). Research-based guidelines for computer-based instruction development. *Journal of Research in Computing in Education*, 27(1), 29-47.

[Pia52] Piaget, J. (1952). *The origins of intelligence in children*. New York: Norton & Co.

[Rei89] Reiser, B. (1989). Pedagogical strategies for human and computer tutoring. *Proceedings of the American Educational Research Association*. San Francisco, CA.

[Rip96] Ripley, B.D. (1996). *Pattern recognition and neural networks*. Cambridge, U.K.: Cambridge University Press.

[Sab01] Sabelli, Nora. (2001). What can data mining and retrieval contribute to the study of education? Invited talk at the NSF Workshop on Information and Data Management, May 2001. Online. [http://itlab.uta.edu/idm01/presentation/nora.ppt]

[Sel98] Sellers, Jennifer J. (1998). An empirical evaluation of a fault-tolerant approach to computer-assisted teaching of binary relations. Computer Science Masters Thesis, North Carolina State University.

[Sta96] Stauffer, Karen. (1996). *Applications of student modeling*. Athabasca University. Online. [http://ccism.pc.athabascau.ca/html/students/stupage/Project.abstract.htm]

[Ste91] Steinberg, Ester R. (1991). *Computer-assisted instruction – Synthesis of theory, practice, and technology*. Lawrence Erlbaum Associates, Hollsdale, NJ.

[Ste82] Stevens, A., Collins, A., & Goldin, S. (1982). Misconceptions in students' understanding. In D. Sleeman & J.S. Brown (Eds.), *Intelligent tutoring systems* (pp. 13-24). London: Academic Press.

[Taa97] Taagepera, M., Potter, F., & Miller, G.E. (1997). Mapping students' thinking patterns by the use of the knowledge-space theory. *International Journal of Science Education*, Vol. 19 (Mar 97), 283-302.

[Tat83] Tatsuoka, Kikumi. (1983). Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20(4), 345-354.

[Tat87] Tatsuoka, K. & Tatsuoka, M. (1987). Bug distribution and statistical pattern classification. *Psychometrika*, Vol. 52, no. 2. 193-206.

[Tat71] Tatsuoka, Maurice. (1971). *Multivariate analysis: Techniques for educational and psychological research*. New York: Jon Wiley & Sons, Inc.

[Thu47] Thurstone, L. (1947). *Multiple factor analysis*. The University of Chicago Press. Chicago.

[Tye96] Tyerman, S., Woods, P., & Warren, J. (1996). Loop Tutor and Hypertutor: Experience with Adaptive Tutoring Systems. *Proceedings of the 1996 Australian New Zealand Conference on Intelligent Information Systems*, Adelaide, Australia, 18-20 November 1996.

[Van82] VanLehn, K. (1982). Bugs are not enough: Empirical studies of bugs, impasses, and repairs in procedural skills. *The Journal of Mathematical Behavior, 3*, 3-71.

[Van90] VanLehn, K. (1990). *Mind bugs: The origins of procedural misconceptions*. Cambridge, MA: MIT Press.

[Van98] VanLehn, K. (1998). Analogy events: How examples are used during problem solving. *Cognitive Science*, 22(3), 347-388.

[VJ88] VanLehn, K. & Jones, R. (1988). Student modeling. In M.C. Polson & J. Jeffrey Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 1-19). Hillsdale, NJ: Lawrence Erlbaum Associates.

[VM98] VanLehn, K., & Martin, J. (1998). Evaluation of an assessment system based on Bayesian student modeling. *International Journal of Artificial Intelligence and Education*, 8(2), 179-221.

[VN01] VanLehn, K. & Niu, Z. (2001). Bayesian student modeling, user interfaces and feedback: A sensitivity analysis. *International Journal of Artificial Intelligence and Education*, 12, printed version to appear.

[Vyg86] Vygotsky, L. (1986). *Thought and language*. Cambridge, MA: MIT.

[Wen87] Wenger, Etienne. (1987). *Artificial intelligence and tutoring systems: Computational and cognitive approaches to the communication of knowledge*. Morgan Kaufman. Los Altos, CA.

[WLS] Web Lecture System (WLS) was developed by NC State University Regional Training Center for Parallel Processing. [http://renoir.csc.ncsu.edu/WLS]

[Wis98] Wise, S.L., Plake, B.S. & Mitchell, J.V. (Eds.) (1998). *Applied measurement in education*. Hillsdale, NJ: Erlbaum.

[Yah03] Yahoo Online Search Engine. (2003). Online: [http://www.yahoo.com].

Appendix A: Glossary of Terms

Fault Tolerant Teaching (FTT) – FTT will be used to refer to systems that tolerate student, teacher, and system errors in diagnosing student misconceptions.

Data mining – Data mining is the practice of searching for a group of items in a large database which are related in a non-obvious way.

Q-matrix – A Q-matrix is a matrix relating observed variables to latent variables, or in this research, questions to concepts needed to understand and answer the questions.

Concept – In this research, a concept is not necessarily what a human might label as a concept. It is, rather, a construct which explains the differences in responses to questions, given a set of student data. It is an underlying variable which will help differentiate performance.

Concept State – A concept state is, given a list of concepts and a particular student, a guess at the student knowledge or predicted performance on each of the concepts.

Response Vector (ideal/actual) – A response vector is a vector containing student responses to questions, in this research after being judged correct or incorrect. An ideal response vector would be the result of a student acting exactly as predicted given his concept state and a Q-matrix, whereas an actual response vector would be his real responses to questions.

Question generation & answer judging – Question generation refers to the creation of new problems from a framework by a computer. Answer judging refers to the ability of a computer system to determine if a given student answer is correct or incorrect.

NovaNET – NovaNET is a computer-based educational network which evolved from the first computer-based educational network, PLATO [REF]. NovaNET is a system-quality network with response times in fractions of seconds. Its programming language, TUTOR, is particularly effective for writing educational lessons, including modes for answer judging and integrated help. Lessons on NovaNET integrate text and graphics, and can also be linked to Internet resources, including audio-visual presentations, like those using WLS.

Web Lecture System (WLS) – WLS is an NCSU-developed system for creating and transmitting on-line presentations on the Internet. One NovaNET lesson has been augmented with WLS lectures to add audio and visual slides as an alternative teaching strategy.

Appendix B: Author's past research contributions

Publications

Barnes, T., & D. Bitzer. Evaluation of the Q-matrix Method of Fault Tolerant Knowledge Assessment**.** *Proceedings of the E-Learn 2002 World Conference on E-Learning in Corporate, Government, Heatlhcare, & Higher Education*, Montreal, Canada, October 15-19, 2002.

Barnes, T., & D. Bitzer. Fault Tolerant Teaching and Automated Knowledge Assessment**.** *Proceedings of the 40th Annual ACM Southest Conference (ACMSE'02)*, Raleigh, NC, April 27, 2002.

Cavey, L., & T. Barnes, "Teachers on Track with Technology - Problem-based mathematics teacher preparation," 12th International Conference of the Society for Information Technology & Teacher Edcuation (SITE 2001), Orlando, FL, March 8, 2001. [Proceedings to be posted at http://www.aace.org/conf/site/]

Berenson, S. B., Droujkova, M., Cavey, L., Smith, N., & Barnes, T. Girls on Track with Information Technology. *Meridian* 3 (1), Winter 2000. Online. [http://www.ncsu.edu/meridian/2000wint/math/index.html].

Alspaugh,T., A. Antón, T. Barnes, & B. Mott. An Integrated Scenario Management Strategy. IEEE Fourth International Symposium on Requirements Engineering (RE`99), University of Limerick, Ireland, 7-11 June 1999.

Tiffany Barnes and Carla D. Savage. A recurrence for counting graphical partitions. *Electronic Journal of Combinatorics*. Volume 2. 1995.

Tiffany Barnes and Carla D. Savage. Efficient generation of graphical partitions. *Discrete Applied Mathematics* 78 (1997). p 17-26.

Presentations at Refereed Professional Meetings

Barnes, T., Cavey, L., Smith, N. (September 2000). Girls on Track: Using Technology as a Tool for Community Investigations. Paper presented at the Annual Technology Exposition, North Carolina State University, Raleigh, NC.

Cavey, L., Barnes, T., Smith, N., & Droujkova, M. (1999). Mathematical Explorations of Urban Problems. Presentation at the Joint Meeting of School Science and Mathematics and the North Carolina Council of Teachers of Mathematics, Greensboro, NC.

Knight, V., Cavey, L., Barnes, T., & Smith, N. (January 2000). Girls on Track: Middle Grade Girls Modeling Community Problems - An Experiment in Progress. Presented at the Joint Mathematics Meetings of AMS-MAA-MER, Washington D.C.

Appendix C: Q-matrix Pseudo-code

```
Main Q-matrix Algorithm {
    Set Delta = 0.1
      MaxIterate = 5
         NumberofConcepts=0
         KeepGoing=1

    Do
        NumberofConcepts += 1
        Create Q-matrix[NumberofConcepts][NumberofQuestions]
        RandomInitialize(Q-matrix)           /* random numbers between 0 and 1 */
        MinError(NumberofConcetps) = ErrorCalc(Q-matrix)
        For Iterate = 1 to MaxIterate        /* repeat to avoid local min */
                For ques = 1 to NumberofQuestions
                        For con = 1 to NumberofConcepts
                                For i= 1 to 2                /* +/- delta */
                                        If i=2 then Delta = -2*Delta
                                Q-matrix(con,ques) += Delta
                                CurrError = ErrorCalc(Q-matrix)
                                If CurrError < MinError(NumberofConcepts) then
                                    Refine(Q-matrix) (run loop w/ delta<<Delta)
                                                BestQ-Matrix(NumberofConcepts) = Q-matrix
                                                MinError(NumberofConcepts) = CurrError
                                        endIf
                                endFor(+/- Delta)
                        endFor(con)
                endFor(ques)
        endFor(Iterate)
        If MinError(NumberofConcepts) is significantly better than
        MinError(NumberofConcepts-1) then
                BestQ-matrix = BestQ-matrix(NumberofConcepts)
                MinError= MinError(NumberofConcepts)
                KeepGoing=1
        Else KeepGoing=0                /* Previous Q-matrix is kept */
    While (KeepGoing = 1)
}
    Procedure ErrorCalc(Q-matrix)       /* Returns error for current Q-matrix */
    {
        NumConStates = 2**NumberofConcepts
        NumberofResponses = 2**NumberofQuestions
        TotalError = 0

        For ConState = 0 to NumConStates-1
                Calculate IDR(ConState)
        endFor(ConState)

        For Response = 0 to NumberofResponses
                If ResponseArray[Response] > 0 then /* Calcuate error */
                        MinRespError = 100000
                        For ConState = 0 to NumConStates – 1
                                CurrRespError = BitDifference(Response, IDR(ConState))
                                If CurrRespError < MinRespError then
                                        MinRespError = CurrRespError
                                endIf
                        endFor(ConState)
                        TotalError += MinRespError * ResponseArray[Response]
                endIf
        endFor(Response)
      Return TotalError
    }

    Procedure CreateResponseArray
    {
        Zero(ResponseArray)   /* Set to all zeroes */

        For Student= 1 to NumberofStudents
                ResponseArray[StudentResponse(Student)] += 1
        endFor(Student)
    }
```

Appendix D: Axiom List

Logical Implications

1.	$[(p \to q) \wedge (q \to r)] \Rightarrow (p \to r)$	Hypothetical Syllogism
2.	$[(p \vee q) \wedge \neg p] \Rightarrow q$	Disjunctive Syllogism
3.	$[p \wedge (p \to q)] \Rightarrow q$	Modus Ponens
4.	$[(p \to q) \wedge \neg q] \Rightarrow \neg p$	Modus Tollens
5.	$(p \to 0) \Rightarrow \neg p$	Absurdity
6.	$(p \wedge q) \Rightarrow p$	Simplification
7.	$[(p \to q) \wedge (r \to s)] \Rightarrow [(p \vee r) \to (q \vee s)]$	Constructive Dilemma ($\vee$)
8.	$[(p \to q) \wedge (r \to s)] \Rightarrow [(p \wedge r) \to (q \wedge s)]$	Constructive Dilemma ($\wedge$)
9.	$p \Rightarrow (p \vee q)$	Addition
10.	$[p \wedge q] \Rightarrow (p \wedge q)$	Conjunction

Logical Equivalences

11.	$\neg \neg p \Leftrightarrow p$	Double Negation
12.	$(p \vee q) \Leftrightarrow (q \vee p)$	Commutative Laws (12-13)
13.	$(p \wedge q) \Leftrightarrow (q \wedge p)$	
14.	$[(p \vee q) \vee r] \Leftrightarrow [p \vee (q \vee r)]$	Associative Laws (14-15)
15.	$[(p \wedge q) \wedge r] \Leftrightarrow [p \wedge (q \wedge r)]$	
16.	$[p \vee (q \wedge r)] \Leftrightarrow [(p \vee q) \wedge (p \vee r)]$	Distributive Laws (16-17)
17.	$[p \wedge (q \vee r)] \Leftrightarrow [(p \wedge q) \vee (p \wedge r)]$	
18.	$(p \vee p) \Leftrightarrow p$	Idempotent Laws (18-19)
19.	$(p \wedge p) \Leftrightarrow p$	
20.	$(p \vee 0) \Leftrightarrow p$	
21.	$(p \vee 1) \Leftrightarrow 1$	
22.	$(p \wedge 0) \Leftrightarrow 0$	Identity Laws (20-24)
23.	$(p \wedge 1) \Leftrightarrow p$	
24.	$(p \to p) \Leftrightarrow 1$	
25.	$(p \vee \neg p) \Leftrightarrow 1$	Tautology (25)
26.	$(p \wedge \neg p) \Leftrightarrow 0$	Contradiction (26)
27.	$\neg(p \vee q) \Leftrightarrow (\neg p \wedge \neg q)$	
28.	$\neg(p \wedge q) \Leftrightarrow (\neg p \vee \neg q)$	DeMorgans (27-28)
29.	$(p \to q) \Leftrightarrow (\neg q \to \neg p)$	Contrapositive (29)
30.	$(p \to q) \Leftrightarrow (\neg p \vee q)$	
31.	$(p \to q) \Leftrightarrow \neg (p \wedge \neg q)$	Implication (30-33)
32.	$[(p \to r) \wedge (q \to r)] \Rightarrow [(p \vee q) \to r]$	
33.	$[(p \to q) \wedge (p \to r)] \Rightarrow [p \to (q \wedge r)]$	

Additional Logical Implications

34.	$(p \to q) \Rightarrow [(p \vee r) \to (q \vee r)]$	Transitivity of $\to$
35.	$(p \to q) \Rightarrow [(p \wedge r) \to (q \wedge r)]$	Transitivity of $\to$