COURSE TRANSFORMATION: CONTENT, STRUCTURE AND EFFECTIVENESS ANALYSIS

by

Linda P. DuHadway

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

School of Computing

The University of Utah

May 2016

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Linda P. DuHadway**

has been approved by the following supervisory committee members:

Thomas C. Henderson,	Chair(s)	**3 Mar 2016** Date Approved
H. James de St. Germain,	Member	**3 Mar 2016** Date Approved
Robert R. Kessler,	Member	**3 Mar 2016** Date Approved
Vivek Srikumar,	Member	**3 Mar 2016** Date Approved
Robert Z. Zheng,	Member	**3 Mar 2016** Date Approved

by **Ross T. Whitaker**, Chair/Dean of

the Department/College/School of **Computing**

and by **David B. Kieda**, Dean of The Graduate School.

ABSTRACT

The organization of learning materials is often limited by the systems available for delivery of such material. Currently, the learning management system (LMS) is widely used to distribute course materials. These systems deliver the material in a text-based, linear way. As online education continues to expand and educators seek to increase their effectiveness by adding more effective active learning strategies, these delivery methods become a limitation. This work demonstrates the possibility of presenting course materials in a graphical way that expresses important relations and provides support for manipulating the order of those materials. The **ENABLE** system gathers data from an existing course, uses text analysis techniques, graph theory, graph transformation, and a user interface to create and present graphical course maps. These course maps are able to express information not currently available in the LMS. Student agents have been developed to traverse these course maps to identify the variety of possible paths through the material. The temporal relations imposed by the current course delivery methods have been replaced by prerequisite relations that express ordering that provides educational value. Reducing the connections to these more meaningful relations allows more possibilities for change. Technical methods are used to explore and calibrate linear and nonlinear models of learning. These methods are used to track mastery of learning material and identify relative difficulty values. Several probability models are developed and used to demonstrate that data from existing, temporally based courses can be used to make predictions about student success in courses using the same material but organized without the temporal limitations. Combined, these demonstrate the possibility of tools and techniques that can support the implementation of a graphical course map that allows varied paths and provides an enriched, more informative interface between the educator, the student, and the learning material. This fundamental change in how course materials are presented and interfaced with has the potential to make educational opportunities available to a broader spectrum of people with diverse abilities and circumstances. The graphical course map can be pivotal in attaining this transition.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to acknowledge the support and assistance of the many people who made the completion of this work possible. To my husband, Carl, who has always encouraged me to continue my educational pursuits and sacrificed his time and efforts in my behalf. To my children who have supported the effort, cheered me on, and found ways to materially assist in the work. To my major professor, Dr. Thomas C. Henderson, who was willing to work with my uncommon interests and continually returned my focus to innovative computer science. To Tamara who adds expressiveness to the words. To my father, Wendell Pope, for creating a legacy of education. To my mother, Dorothy Pope, for her smile and for financing this endeavor. To the School of Computing at the University of Utah for providing me this opportunity.

CHAPTER 1

INTRODUCTION AND OVERVIEW

1.1 Introduction

In an effort to meet the changing landscape of education, many departments and universities are offering more online courses – a move that is likely to impact every department in some way [4]. This will require more instructors to create online courses. Other innovations in instructional strategies are also widely impacting educators [5], including peer instruction, flipped classrooms, problem-based learning, just-in-time teaching, and a variety of active learning strategies. Implementing any of these strategies requires changes to existing courses.

As the demand for online and hybrid courses increases [6], teachers who are experienced and talented at designing and presenting synchronous, face-to-face courses are being asked to adapt their material and presentation to an asynchronous, online format. To many this is an unfamiliar approach and the process of making the transition is unclear. Since there is already an investment in existing material and methods, it is not desirable to abandon them entirely. At the same time, this new approach provides opportunities for new methods and material to be of benefit.

Sometimes an educator is so familiar with the current course organization that it becomes a stumbling block for visualizing alternative options. When anticipating change it is valuable to see how existing learning materials can be organized and used in new ways. ENABLE is a system of tools and algorithms developed to explore the possibilities of organizing and presenting learning materials in new ways. ENABLE is not an acronym but rather a name that reflects the purpose to enable the implementation of quality educational strategies. One of the purposes of ENABLE is to provide assistance in making informed change. Educators seeking to transition to an online course or implement innovative changes in the classroom could benefit from a deeper analysis of the contents and structure of the course material.
The overall objectives of this project are as follows.

1. to create a set of methods to analyze the content and structure of existing learning materials that have been used in a synchronous, linearly structured course and provide

insight into the nature and relations of the course material and provide alternative ways to organize them,

2. to provide a Bayesian framework to assist in the discovery of causal relations between course learning items and student performance, and

3. to develop some simple student behavior models to probe the methods' efficacy and accuracy.

This set of methods is called ENABLE, which is not an acronym, but rather a name that reflects the purpose to enable the implementation of quality educational strategies. With ENABLE, educators are able to see the relations of the existing learning items using a visual, nonlinear presentation, to predict the impact of the organization of learning items, and to discover poorly organized or presented material.

As an example, consider the variation in the graphs shown in Figure 1.1. Information about the learning items for three sample courses was gathered from Canvas (a standard learning management system), and graphs were produced representing the current organization of each course. Figure 1.1 shows three graphs from one of those courses, a CS0 course, Foundations of Computer Science, taught at Utah State University. Figure 1.1(upper) shows all the learning items for this course laid out in order across the days of the semester. Figure 1.1 (middle) shows the initial course graph constructed directly from the learning material, and visually exhibiting the relations of interest: *precedes*, *occurs in*, and *includes*. The orange nodes (small, no fill color) represent the learning items. The orange edges between the learning item nodes are the *precedes* relations. The green nodes (larger with solid fill color) represent the topics. The green edges go between the topic nodes and the learning item nodes and represent the *occurs in* relations. The unit relations are expressed visually by locating nodes *included* in a unit near the same vertical location. Figure 1.1 (lower) shows the class material after text analysis and graph transformation by the ENABLE system. Note that this is one possible reorganization of the course. In this transformation the learning items are organized by topic. Only the topically related *precedes* relations are included, which allows a clearer separation of learning items by topic. The *occurs in* relations are all represented. In this particular transformation, *includes* relations are not expressed.

Due to the size of these graphs it is difficult to see the details clearly in this small of an image, but the changes are significant enough that you get a sense of the differences the transformations make. Figure 1.2 shows a small section of the graph and allows the

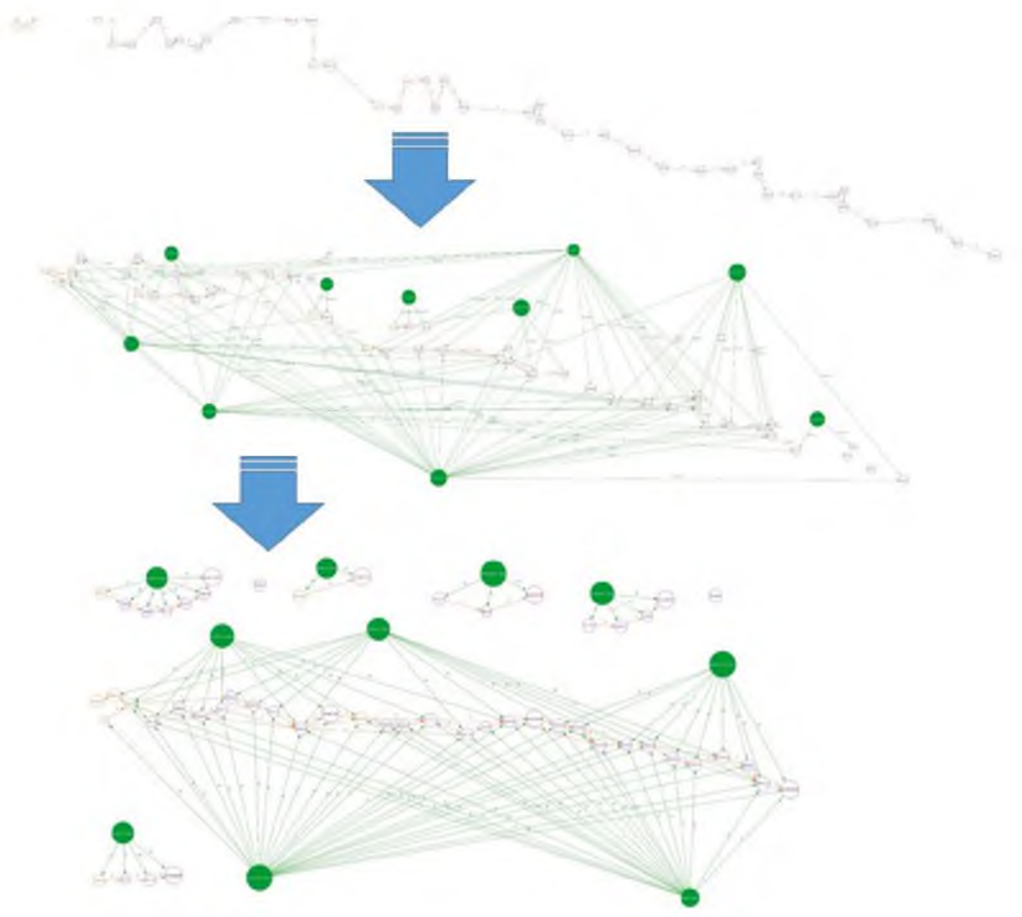


Figure 1.1. CS0, Foundations of Computer Science Original Course Organization (upper). CS0, Initial Course Graph Relations (middle). CS0, Transformed Course Organization (lower).

details to be more visible. This image shows a single topic node (solid circle) and three learning items (circles with no fill color). The learning item to the far right is an exam and the relations coming into this exam consist of four *occurs in* relations and three *precedes* relations. This is an ideal situation to apply the split exams transform that is discussed in chapter 4, The Course Map.

Current learning management systems display materials in a textual, linear format primarily based on chronology. This presentation of material provides a limited view of the course. An asynchronous course is not limited to a linear, chronological organization and can be enhanced by different arrangements of the learning material. ENABLE provides various organizational options of the course materials. This supports the identification of new ways to organize the materials and restructure the delivery to exploit the flexibility of the online setting, possibly including real-time, context-dependent reorganization.

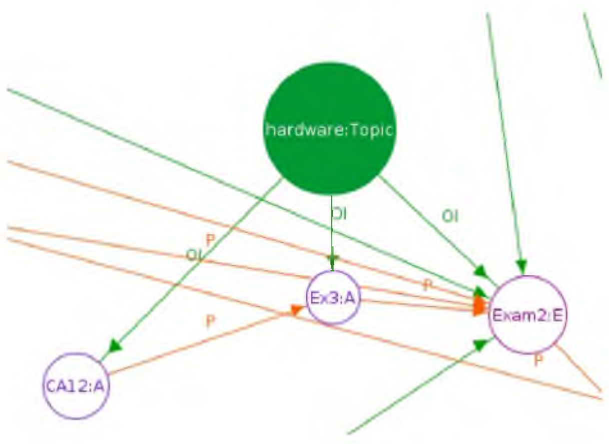


Figure 1.2. Close Up of Small Section of Graph From Figure 1.1 (middle).

Identifying relationships between learning opportunities has been done at the curricular level. There is a body of research that looks at using curriculum maps to represent the relations between courses and program learning objectives. The use of curriculum maps has been adopted in a variety of educational settings. The process of creating these maps is time- and resource-consuming, forestalling their implementation by many [7]. These maps are largely textual and often presented in tables. Some universities have presented the curriculum maps in a more visual way by using a diagram or textual web [8, 9]. Some list the learning resources and learning objectives without identifying how they are connected, while others identify the level of mastery expected of a course for each program objective.

Curriculum maps are designed to provide information about the curriculum and the connections between learning opportunities to teachers, students, curriculum designers, administrators, and managers [7,10], but the most often reported benefits are related to the development of the maps. The first step in developing a curriculum map is for individual teachers to develop maps of their course as it is being taught, and then working in groups to combine and revise the individual maps of courses in the curriculum [11], ultimately mapping the courses to program outcomes [12]. The process of looking carefully at a course by the teacher and then sharing and discussing the insights with others involved in the process, produces improved communication and greater attention to teaching [10, 12]. It is also reported that it improves collegiality between faculty [11].

This work was built on some of the principles used in curriculum mapping, such as teacher course analysis and identifying connections. However, the use of concepts from curriculum mapping has been extended to use with an individual course. Currently the connections identified during the mapping process are between courses and program outcomes. A broader analysis of the course will be undertaken using additional connections, such as temporal and topical relations between individual learning items. The focus of this analysis is to enable the transformation of an existing face-to-face synchronous course to an online asynchronous course. The contribution of this work is to create algorithms embedded in an interactive tool that will

- support the instructor during the analysis phase by providing information about the existing course through automated analysis and receiving and incorporating additional input from the instructor,
- generate a graphical representation of the learning items and the relations between them, and
- allow transformation of the learning item graph to facilitate consideration of new organizations.

Figure 1.3 shows an overview of the ENABLE system.

These features of the system provide previously unavailable assistance during the process of transforming a course for new delivery methods. This support goes beyond what is currently available using curriculum maps, both by automated assistance during the analysis phase and by the graphical display of the learning items and their relations.

Providing the ability for the instructor to see and interact with a visual, graphical representation of the course requires the system to enforce limitations that protect the integrity of the representation. For example, a prerequisite will not be added if it creates a cycle. Such a cycle would represent an impossible prerequisite grouping (see Figure 1.4). Similar problems can occur with the addition of *precedes* relations. Other correctness violations can occur when nodes or edges are removed. ENABLE maintains the integrity of the course map.

Analyzing the exposed alternatives using student agents and a Bayesian inference network have been investigated to see if such tools can be applied to identify preferences between the resultant organizations. The results of this examination are reported.

1.2 Course Content Analysis

The first step of the content and presentation analysis is to identify the course materials and organization that are currently being used (for our first results in this domain, see [13]). We conduct this study using Canvas, where much of this information can be acquired

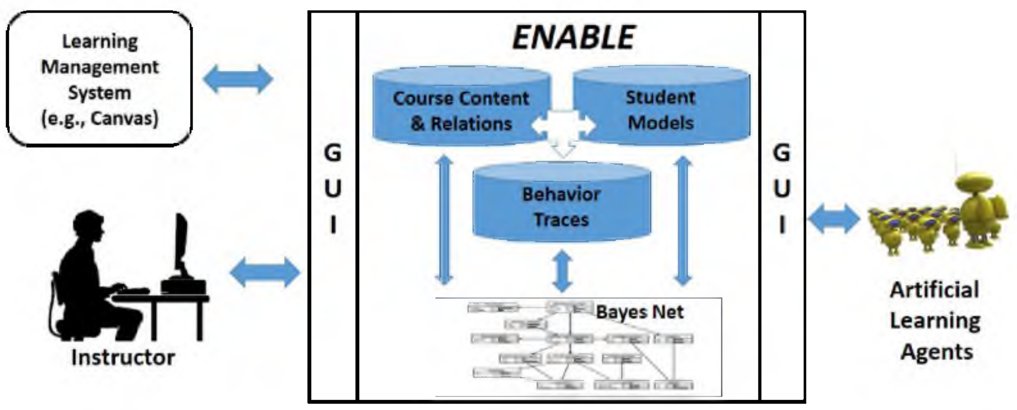

Learning Management System (e.g., Canvas)
ENABLE
Course Content & Relations
Student Models
Behavior Traces
Bayes Net
G U I
G U I
Instructor
Artificial Learning Agents

Figure 1.3. ENABLE System Architecture.

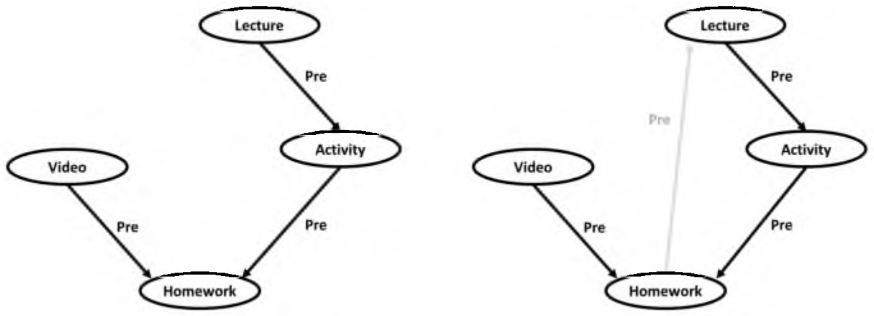
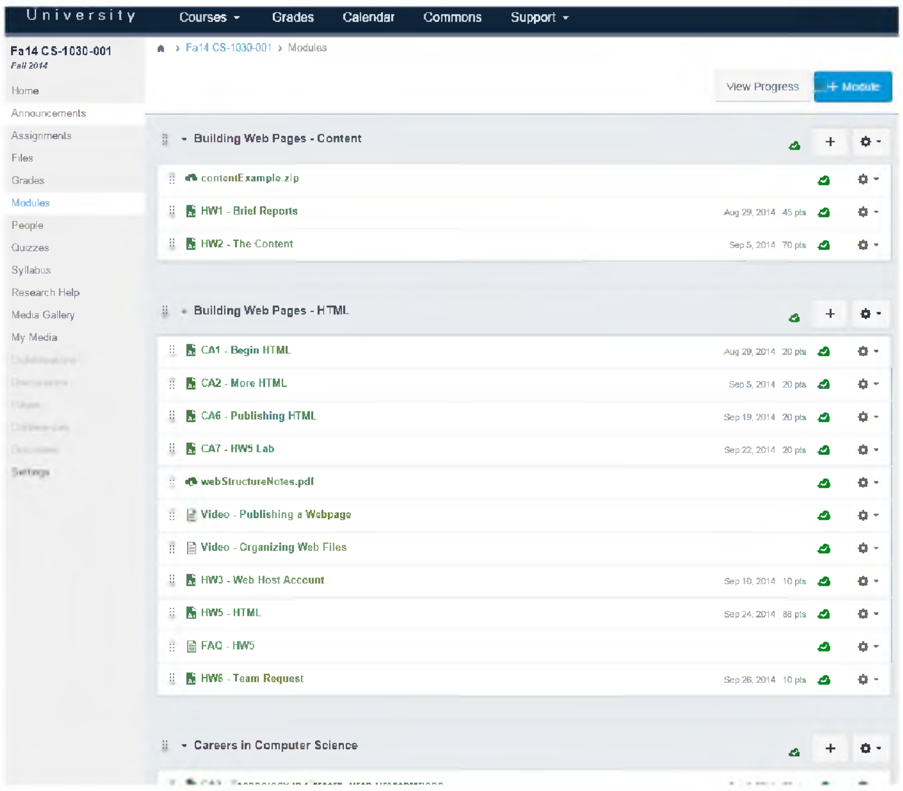


Figure 1.4. Addition of Prerequisite Relation Not Allowed.

directly through the Canvas application program interface (API). The learning materials for a course come in many varieties, such as assignments, quizzes, videos, lecture slides, reading material, etc. We refer to these various materials as learning items (see Figure 1.5).

Some distinction between types of learning items can be made. For example, some learning items require a submission of some type from the student, have a due date, and result in points that apply to the student's grade for the course. Examples of these are homework assignments, quizzes, online discussions, and class activities. These are referred to as assignments. Others have no due date and no points are associated with them. Examples of these learning items are videos, frequently asked questions, readings, and lecture slides. These are categorized as learning resources. One more distinction is made



Building Web Pages - Content

contentExample.zip		
HW1 - Brief Reports	Aug 29, 2014	45 pts
HW2 - The Content	Sep 5, 2014	70 pts

Building Web Pages - HTML

CA1 - Begin HTML	Aug 29, 2014	20 pts
CA2 - More HTML	Sep 5, 2014	20 pts
CA6 - Publishing HTML	Sep 19, 2014	20 pts
CA7 - HW5 Lab	Sep 22, 2014	20 pts
webStructureNotes.pdf		
Video - Publishing a Webpage		
Video - Organizing Web Files		
HW3 - Web Host Account	Sep 10, 2014	10 pts
HW5 - HTML	Sep 24, 2014	88 pts
FAQ - HW5		
HW6 - Team Request	Sep 26, 2014	10 pts

Careers in Computer Science

Figure 1.5. Sample of Learning Items Available in Canvas.

for exams. Exams have many of the characteristics of the assignment, such as a due date and points, but the exam is a special case in that it can be used to assess learning items that are not otherwise related. Identifying the exam as a distinct category increases the options available when considering new organizations.

From Canvas the system gathers textual information about the learning items, such as a problem statement for an assignment, questions for a quiz, or text from slides. Additional details about learning items are also gathered, such as due date, points possible, which module the item is in, and files associated with the learning item.

Some materials associated with learning items may not be available in the learning management system. For example, an exam that is distributed during class will have an item in Canvas associated with the exam. This item has a title, due date, and points possible, but the actual text of the exam is not included. The text of an exam can be meaningful during the analysis phase. To accommodate such circumstances, the system allows additional materials to be uploaded so they can be included in the analysis.

ENABLE also gathers information from Canvas to establish the current organization of the course materials. In Canvas as well as other learning management systems, the organization of material is presented in a linear fashion based on dates. This arrangement identifies temporal relations in the existing organization.

Some courses use the Modules tool available in Canvas. This tool allows an instructor to group learning items into groups or units. Many different groupings are used. Some instructors group the material based on a textbook, such as a unit for each chapter. Others use it to organize temporally, such as one unit for each week in the course. Another approach is to organize by specific topic coverage. Current grouping in these modules reflects groupings that are in some way meaningful to the instructor and represent the current organization of the course.

The next step is to analyze the material to establish relations between learning items. One of the primary purposes of ENABLE is to identify and present a variety of relations between learning items. A review of the relations between learning items in a sample course revealed distinct differences in their types. The most readily available relation is the *precedes* relation. This temporal relation expresses the relation in time between learning items. A learning item (Item A) *precedes* another learning item (Item B) when the due date of Item A is before the due date of Item B. These relations are transitive such that if Item A *precedes* Item B, and Item B *precedes* Item C, then Item A *precedes* Item C. Many of these relations are trivial and can be removed without loss of meaning.

A stronger relation is the *prerequisite* relation, such as a HW (homework) that is a *prerequisite* to an exam or a video that is *prerequisite* to a class activity. This type of relation indicates that one learning item comes before another and successful completion of the activity in some way increases the likelihood of success on the following item.

Another relation is the topical relation *occurs in*. This relation expresses that a topic *occurs in* a learning item. Connecting topics to specific learning items shows how learning items are topically related to other learning items.

The final relation is the unit relation *includes*. This relation expresses that a specific unit *includes* a learning item. A unit is a group of learning items. These learning items are grouped into units in the existing course using the Modules tool in Canvas.

This phase of the project involves both the analysis of the course data gathered from the LMS and input from the instructor. One of the primary benefits of the curriculum-mapping process is the insights the teacher acquires about their own course as they analyze the course as it is currently delivered. The ENABLE system is designed to involve the instructor during the analysis phase to facilitate this increase of awareness from examination. The interactive interface of ENABLE is designed to gather and incorporate information from the instructor and to display the analysis as it develops.

The final step of this phase is to display the current course organization in a graphical way. To do this the system uses a graph structure that includes nodes representing individual learning items and edges that express the relations between learning items. This view includes the relations as they exist in the current organization, including the temporal relations as they exist in the synchronous face-to-face course. This graphical representation of the course is one of the contributions of this project.

Course materials are often presented in a textual, linear way, with one item coming directly after another item with little distinction of importance or relations between items. Initial evaluation of Canvas found the presentation of material is largely based on *precedes* relations. Of the relations being analyzed by ENABLE, the *precedes* relation is the most trivial. By adding more meaningful relations like *occurs in* and *prerequisite* relations to the presentation of learning items, the amount of information delivered is increased. To deliver this richer set of information, a visual representation of the course has been developed.

This step requires finding and using graph visualization techniques that present the learning items and course structure in a way that expresses this greater collection of information. ENABLE uses nodes to represent learning items and edges to represent relations. Visual properties such as color, size, and spatial location are used to deliver information about the learning items and relations between them.

To better understand the size of these graphs, three actual courses were reviewed, Foundations of CS, CS1 Lab, and a Web Development course, all from Utah State University. There were 52, 90, and 87 learning items in these courses (see Figure 1.6 (a)). Graphs with this many nodes are considered small graphs. However, it is important to create a visualization that expresses the learning items, how they are related, and how their organization can be rearranged.

There will be times when a simplified version of the structure may be useful. One way to simplify the presentation is to group learning materials into units. In Canvas an educator can group learning items together to create a unit of learning. Items are grouped together in different ways. Some instructors group the material based on a textbook, such as a unit for each chapter. Others use it to organize temporally, such as one unit for each week in the course. Another approach is to organize by specific topic coverage. The way items are organized into units in the current course reflects groupings that are in some way meaningful to the instructor.

Using the unit grouping with the three courses mentioned previously, there were 12, 11, and 14 units, respectively (see Figure 1.6 (b)). Using this grouping method, the visualization

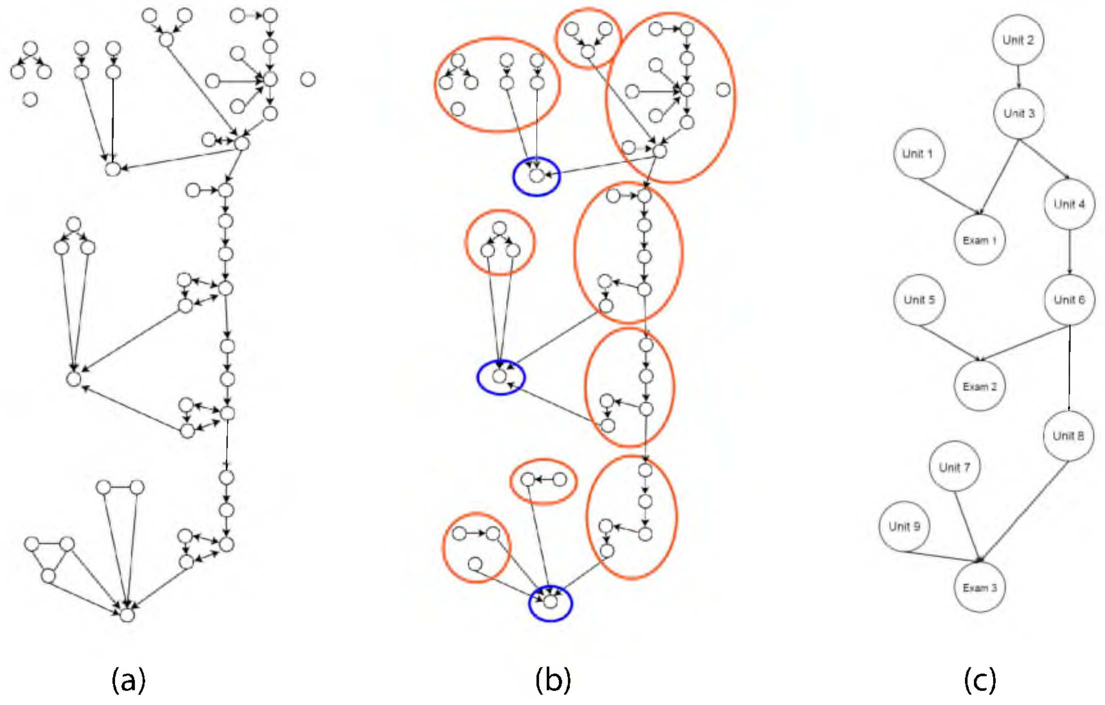


Figure 1.6. Sample Course with 52 Learning Items. (a) Detailed View (b) Detailed View with Units (c) Simplified View

can be significantly simpler and still expressive (see Figure 1.6 (c)). In some instances the information available in the LMS defines what learning items belong in a particular unit. In other cases other techniques, such as community detection in graphs may be needed to identify these units [14].

Illustrations are a powerful explanatory tool [15]. For example, when creating the graphical images of the sample course, the units with a single node stood out (see Figure 1.6). What is the point of a unit with only one learning item? This initiated a closer look. These single-item units were the exams for the course. This made sense, but seeing it visually fostered new insights. Seeing that multiple units fed into the exam unit without being connected to each other caused us to look more closely. We discovered that these units were not topically related to each other, and the idea that exams could be subdivided was born. This led to the Topic-based Exam Splitting Rule described in Section 3. Although this information was available in the linear, textual presentation of the course, these new ideas were generated by seeing the visual, graphical representation of the same learning items. Graphically representing the course provides a fresh view.

Just as we gained new insights while transforming a sample course, an instructor who is familiar with the current representation will see new things when it is presented in an innovative, visual way. Participating in the process of transformation generates insights into how learning items are related and how course materials can be delivered. ENABLE provides an interactive interface, making the instructor an integral part of the transformation process. Much of the benefit of the process derives from the insights the instructor gains as they participate in this process.

1.3 Graph Transformations

Once the initial course graph is available, it becomes possible to begin a conversion process from a linear (chronological) style class organization to a more nondeterministic, multipath organization of the learning items more suitable to online delivery.

Because ENABLE identifies alternative course structures, it is necessary to transform the graph while still keeping the relations intact. Graph grammars and graph transformation systems provide a means for doing this. There is much research and many successful applications based on the research in this area [16]. One of the application areas of graph transformation systems is model transformations. This area of model transformation has become important to the field of software engineering [17]. The models used in software engineering have enough similarities to the graphical representation of learning materials to motivate the consideration of model transformation as the graph transformation technique to be used for ENABLE. Those similarities include typed nodes, node attributes, and edges that represent different types of relations.

To further investigate the possibility of using model transformation tools, a comparison of some currently available tools, namely AGG, AToM3, VIATRA, and VMTS, was conducted [18–22]. These model transformation tools are based on approaches that draw from the theoretical work on graph transformations [23]. These four approaches to the model transformation problem are in many ways similar and are built upon the solid practices developed through research of graph grammars and graph transformation systems. ENABLE incorporates algorithms that enforce mathematically sound graph transformation techniques.

We now consider some desirable transforms and their meanings. We begin with the consideration of how to eliminate unnecessary *precedes* relations. We define a restraint as an unnecessary constraint between two items. Thus, restraints are removed in order to open up more possibilities for the relations between learning items. When removing restraints it is important to maintain the integrity of the course representation.

1.3.1 T1: Topic-based Precedes Elimination Rule

A major restraint is the *precedes* relation. It restricts any change in the order of learning items. However, many of the *precedes* relations are not necessary and can be removed without changing the necessary relations. The first step is to remove unnecessary *precedes* relations. The fact that one learning item comes before another provides only limited information. Once ENABLE has identified the temporal and topical relations, the system can identify *precedes* relations that have no topical connections. The *precedes* relations between these topically unrelated learning items can be removed. This reduces the number of precedes relations to those that have common topics.

1.3.2 T2: Topic-based Exam Splitting Rule

One result of building course organization based on temporal relations is illustrated by exams. Commonly, an exam is written to assess the material that has been covered over a specific period of time, such as since the last exam or since the beginning of the semester. This time-based connection is not required for assessment. Therefore it is possible to divide the material assessed in an exam by topic. Separating the temporal grouping inherent in exams provides additional possibilities for change.

The split exams rule separates a single exam into multiple exams when there are learning items that *precede* the exam but are not topically related to each other. It is applied after the remove *precedes* rule has been applied. Enforcing this rule application order prevents any exams being split when preceding learning items are topically related. This, then, is another example of a restraint: when exams tie learning items together that are not related in any way other than temporally.

1.3.3 T3: Replacing Precedes Relations with Prerequisite Relations

Even after the *precedes* relations have been restricted to those that have common topic relations, they still express limited information. A more informative relation is the *prerequisite* relation that expresses a recommendation that one learning item be completed before another learning item. The *precedes* relation has one learning item directly following another learning item. This limits the way learning items can be connected and does not allow flexibility in ordering. Eliminating *precedes* relations in favor of *prerequisite* relations provides a more accurate representation of the course material. This transformation step requires more input from the instructor. Once the previous transforms have been applied, the resulting graph is presented to the instructor and ways to gather information about *prerequisite* relations are provided.

The application of these three transformations removes the temporal restraints imposed by a linear, time-based presentation of learning material. This removal of temporal restraints facilitates more flexible organizations that become particularly valuable in asynchronous settings such as online courses, technical training, or competency-based learning.

1.3.4 T4: Instructor Directed Transformation

Once the temporal restraints of a synchronous course have been removed, the opportunities for restructuring the course are greater. The ENABLE system can now display a graph based on the topical and prerequisite relations without the time-based restraints.

There are many ways to arrange learning items based on these remaining relations. The ENABLE system displays the learning items and relations as a graph in a variety of ways, such as topic-based or unit-based. These views allow the instructor to see new options for restructuring the course. There are, however, additional ways to organize the learning items. ENABLE provides an interactive display of the learning items and relations that allows the instructor to move learning items around while keeping the relations connected. This freedom to change and see the different ways the items can be laid out helps the instructor envision possibilities.

Although time-based relations have been removed from this graphical representation of a course, the reality is that the student actually moves through the learning items one after another in time. What is gained is a variety of ways that learning items can be traversed. The actual movement through the learning materials is displayed. The system can also provide a variety of organizations that maintain the relations.

1.4 Artificial Student Agents

A major goal of the ENABLE framework is to allow the building of models during system usage that

- provide student outcome projections (e.g., exam scores, grades),
- provide advice to students on how to improve their performance (e.g., where to spend more time, how to order the material, etc.), and
- provide instructors with insight into performance causality relations (e.g., linking poor results with the required material).

In order to achieve this, it is necessary to study student traversals of the course graph and the subsequent outcomes. This must be done in a controlled manner in order to ensure that the correct conclusions are drawn. To this end, we propose to develop *artificial student agents* to go through the course material, take exams, etc., and produce example outcomes.

Artificial agents are defined as software agents with a variety of factors informing their makeup (see [1] for an overview of agent decision-making architectures). The factors making up a student agent include

- Native Intelligence
- Work Ethic
- Distractability
- Background Preparation.

Each of these factors has a range of values with a probability distribution over those values. These are taken into account when deciding upon an action within the course graph traversal context.

Artificial student agents are created to represent students of varied abilities and learning styles. A review was done of previous students to identify rates of successful completion of the learning items. This information is used to inform the design of the student agents in this study (see Figure 1.7). ENABLE allows the application of these agents to the course graphs produced as described previously.

1.5 Behavior Modeling and Bayesian Analysis

The next area of study is the modeling of how students traverse the course material as organized in the various transformations and to identify course content and organization modifications which will most likely improve performance. We treat the graph nodes as a set of random variables where the value of a node represents the probability that the student understands the material at that node. Given this set of random variables, a Bayesian network provides an efficient mechanism to express knowledge about the full joint probability distribution. Using Bayes' theorem, conditional probabilities may be determined if priors and conditional relations between variables are known. *A priori* probabilities and conditional relations have been drawn from the data available in the existing course. In this way, conditional probabilities may be expressed, say $p(B \mid A)$; i.e., the probability that a student knows B given that they know A. Information of this sort can be used to report on the relative success of the student agents traversing the learning item graphs.

When a Bayesian network is drawn, the links represent interdependencies that are expressed by conditional probability tables of a child that are based on its parents [24].

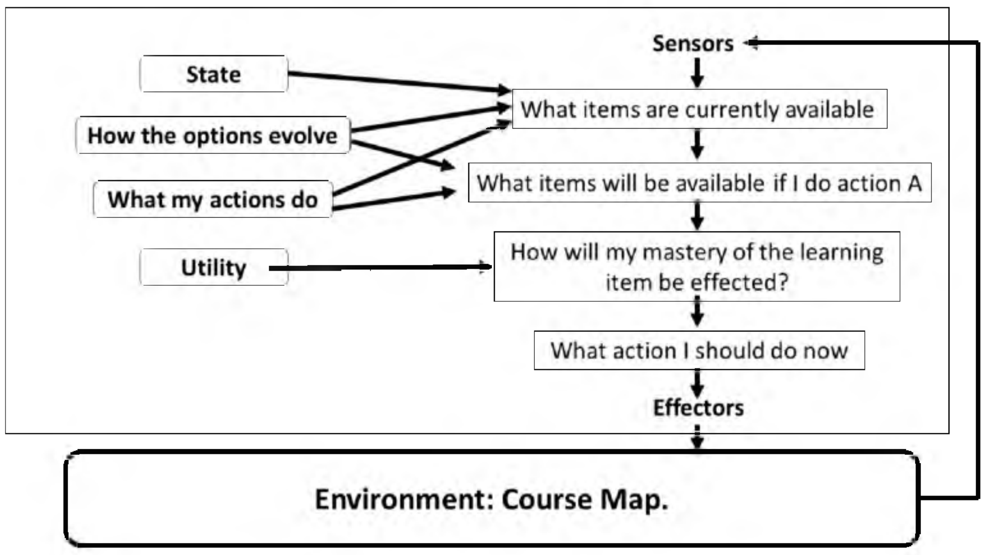

Sensors
State
What items are currently available
How the options evolve
What items will be available if I do action A
What my actions do
Utility
How will my mastery of the learning item be effected?
What action I should do now
Effectors
Environment: Course Map.

Figure 1.7. Student Agent Architecture (modified from [1]).

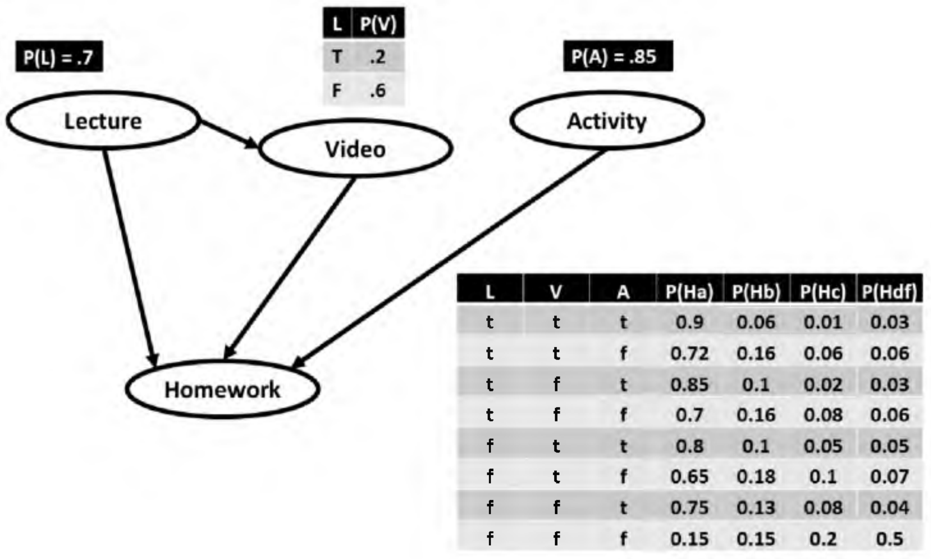
The size of these tables varies, depending on how many parents and whether the values are Boolean, discrete, or continuous. For an example see Figure 1.8. This shows a portion of a possible network for the sample course. The lecture, video, and activity learning items are Boolean values representing attended, watched, and participated, respectively. The homework learning item has four discrete values showing grade results of A, B, C, or D/F.

Bayesian network analysis has been used for a variety of educational purposes, such as individualization in intelligent tutoring systems [25], student modeling [26,27], and detecting learning styles [28]. In [29] a Bayesian network is used to analyze student grades in course prerequisite structure and make recommendations for interventions to improve student retention. They have not been used to support course organization, and thus ENABLE uses Bayesian networks in a novel way.

A Bayesian inference network has been created using existing data related to the learning items. Student agents traverse the learning network and the Bayesian network provides feedback data.

1.6 Tracking Mastery of Learning Items

The final area of study is the tracking of the mastery of learning items. As students work through individual learning items, their mastery of the learning item is impacted by the efforts applied to the current learning item as well as the efforts they have made on previous learning items. To track the mastery of learning items a Kalman Filter, also known as a



L	P(V)
T	.2
F	.6

L	V	A	P(Ha)	P(Hb)	P(Hc)	P(Hdf)
t	t	t	0.9	0.06	0.01	0.03
t	t	f	0.72	0.16	0.06	0.06
t	f	t	0.85	0.1	0.02	0.03
t	f	f	0.7	0.16	0.08	0.06
f	t	t	0.8	0.1	0.05	0.05
f	t	f	0.65	0.18	0.1	0.07
f	f	t	0.75	0.13	0.08	0.04
f	f	f	0.15	0.15	0.2	0.5

Figure 1.8. Hypothetical Section of ENABLE Bayesian Inference Network.

dynamic Bayesian network, is used. The technical details of this approach are described in Chapter 7.

1.7 Evaluation Criteria

The evaluation criteria of the research project are divided into four sections, with each section identifying the (1) capabilities, (2) correctness requirements, and (3) performance measures of a specific component of the project.

1.7.1 Course Content Analysis

1. Metadata about the learning materials are gathered from Canvas. The data gathered are analyzed and used to create a graph representation of the existing course, representing learning items as nodes and relations as edges. The system provides an interface that allows the instructor or other expert to add data that are not available from the learning management system (LMS).

2. Using the data gathered from the LMS and the user, the analysis will correctly identify the type of each learning item and the existing relations between those learning items.

3. The course content and analysis phase has been run on three existing courses that have been taught in previous semesters. The resulting graphical representation of the course has been directly compared to the linear, textual representation of the course in the LMS. Correctness is quantified on a scale of 1-5, which measures whether the

course map correctly represents the following details of the course as outlined in the LMS (since many of the details expressed in the course map are not apparent in the textual information available in the LMS, this comparison is limited to the information that can be determined using the standard interface available in the LMS):

- Learning items are of the correct type.
- Temporal ordering of learning items is the same.
- Topical relations in the course map correctly represent the topical relations identified in the LMS.

The results of these analyses must average to 4 or better in order to be considered correct.

1.7.2 Graph Transformations

1. After a course map has been developed in the form of a graph with learning items as the nodes and relations as the edges, graph transformations can be made. These transformations consist of system-generated and user-directed transformations.
2. Transformations are restricted to valid organizations that conform to the rules of directed acyclic graphs and the semantic rules imposed by the relations.
3. The system produces the following transforms.
   - T1: Topic-based Precedes Elimination Rule.
   - T2: Topic-based Exam Splitting Rule.
   - T3: Replacing Precedes Relations with Prerequisite Relations.
   - T4: Instructor-directed Transformation.

1.7.3 Artificial Student Agents

1. Five student agents are included each with its own variation of properties. These student agents are able to perform multiple and varied traversals.
2. The paths traversed by the student agents are correct based on constraints specified by the relations between learning items.
3. Each student agent is able to traverse the course map, following a variety of paths through the learning material, and report the relative value of the traversal. Each

path traversed represents a valid path based on the existing relations of the course map.

1.7.4 Behavior Modeling and Bayesian Analysis

1. A Bayesian inference network is created that provides probability estimates that represent a model of the interdependencies of learning materials for a sample course. These estimates are based on a combination of data from the actual course and expert knowledge sources.

2. The Bayesian network will use the relations in the course map to connect parents of each node for the CPT tables.

3. Performance is measured by using sampling from the Bayesian network to produce a statistically meaningful number of sample sets. Each sample set includes a score for each learning item. Using these individual score sets, a final score is computed. These final scores are compared to the final scores of the existing course using a variety of measures, including L1 errors, L2 errors, log-likelihood, and final grade distribution.

1.8 Summary of Contributions

The contributions of this work will consist of the following items.

1.8.1 The Implementation of a Graphical, Course Map (or Graph)

To date, learning items for a course are presented in a linear, textual way. Although familiar, this linear approach of presenting course materials has limited capacity to represent the real organization and relations between learning items. Its primary focus is on the *precedes* relation keeping teachers and students informed of which learning item comes before another learning item. But there are much richer relations that can be expressed in a visual, graphical representation of a course, such as topical and prerequisite relations. This research provides a method for representing the course learning items in a nonlinear, visual way. It also provides the functionality to manipulate that representation.

1.8.2 The Capability to See and Change How Learning Items Are Organized in Terms of the Course Graph

This facilitates the consideration by an instructor of new ways to organize and deliver the material. Using visual representations to increase the amount of information that can be delivered and enhance the meaning of that information is not new. This technique is used in many applications across many different fields. It is not yet available for the representation of course organization. Course mapping has the potential to change how teachers and students see a course, giving teachers new insights and students a better understanding of how to negotiate the course materials.

1.8.3 An Automated Way to Gather Data about Course Materials and Analyze That Data to Provide a Realistic Representation of the Course Organization and Then Allow Changes to be Made in That Organization

When making changes, there is value in looking at the course from many different perspectives. An instructor who has developed and taught a course in a certain way for some time may have difficulty seeing alternative organizations. This method of extracting current organization from the LMS and supplying the resources to transform that organization provides an instructor with a different view of the existing course and the means to see the course from different perspectives. This facilitates change.

1.8.4 The Design and Use of Student Agents to Traverse Course Graphs

Since the work is being done on existing courses, information about the students who actually took the course can be used to inform the characteristics and actions of these artificial student agents. Using these agents with the model provided by the course map and the Bayesian network provides a means for comparing traversals of the learning materials. This comparison can be used to inform the instructor of which organizations will more likely promote student success.

1.8.5 The Novel Application of Bayesian Inference Networks to Compute the Interdependencies Between Learning Items

Bayesian inference networks have been used in a variety of ways in education. They are extensively used in automated tutoring systems. This project delivers a new way to use this established approach to probabilistic reasoning.

1.8.6 The Capability of Combining Complementary Techniques to Increase the Effectiveness of Course Transformation

Both innovative and established methods have been combined to analyze current course organization and inform course transformation. Automated analysis and instructor engagement are combined to augment broader solutions. Teacher experience, existing structure, and new views are assembled to generate new insights. This connecting of novel and well-known provides a solid foundation for the introduction of nonlinear, graphical course mapping.

CHAPTER 2

REVIEW OF LITERATURE

The ENABLE system incorporates concepts and methods from many areas of research. There is currently not a system that functions as ENABLE does, so the background research comes from a variety of arenas, including research that defines the problem, existing research from which ideas were gleamed, and research that focuses on some of the technical aspects of the project.

2.1 Forces of Change

Some educators call for significant educational changes based on generational changes among students. A term coined for these differences is digital natives, and the assertion is that teachers are digital immigrants and need to adapt their teaching to fit a different kind of learner [30]. This cry has resounded among instructors until it has become a familiar term and an accepted premise. Some question the validity of the argument and challenge its wide acceptance as a reason for changing educational practices. They demonstrate that the need for such educational changes is not founded on solid research or empirical data [31]. Although the quality of research may be lacking, the demand for change is considerable [30]. Even those who challenge the validity of the digital native claims still call for the need to consider change based on the increased presence of technology [31,32]. Two significant areas of change are online education and active learning strategies.

This section addresses some of the research surrounding changes currently being implemented in educational circles. For the work in this dissertation it is the existence and value of change in education that motivate the research into practices and methods that support educators in the process of making changes. ENABLE provides tools to see existing courses as they currently are, exposing details that are not currently apparent in the presentation of those courses in an LMS. It also provides an interface for the educator to consider a variety of possible changes.

2.1.1 Online Education

The number of higher education students taking online courses continues to rise (see Figure 2.1). The number of students enrolled in at least one online course has risen steadily since Allen (2014) began keeping track in Fall 2002. The numbers have grown from 1.6 million students in the Fall of 2002 to 7.1 million for Fall 2012. This represents an annual growth rate of 16.1%. This is a significantly higher growth rate than the overall growth rate for higher education (2.5%) during the same time period. The proportion of students taking at least one online course is at an all-time high of 33.5% (see Figure 2.2 [2]).

In an effort to meet the changing landscape of education many departments and universities are offering more online courses – a move that is likely to impact every department in some way [4]. This will require that more instructors create online courses [13].

The increase in online education impacts both faculty who teach these classes and those who are using web-based elements in their traditional face-to-face classes. Eventually it is expected that Internet use in courses will involve all faculty as blended approaches become the norm on college campuses [33].

A major area of concern is the quality of education in these online courses. [34] reviewed meta-studies of distance education (some going back as far as 1928) to compare the quality of education in online courses to those in a face-to-face classes. When these studies were looked at as a whole the results showed no significant difference between the two. However, there was considerable variation between individual studies. Individual factors considered included publication year, whether the instructor is the author, type of interaction, level of instruction, and instructor involvement. In addition, there were differences within these individual characteristics. Overall it was concluded that there was a great variance in distance education - about the same variation that there is in face-to-face education.

Some studies have very specific recommendations about how to make an online course successful. These often include both student and faculty requirements. In [35] several variables are considered, including student self-discipline and motivation, faculty member training, simple course design, and instructor involvement. One consistent finding is that the quality of the course is related to the amount of instructor involvement [35, 36].

The transition from face-to-face to online teaching is being supported in a variety of ways. Innovative web-based e-learning authoring tools have been created to enhance learning object creation [37]. Additionally, there is a tool to help those already teaching distance education to have increased online presence [38].

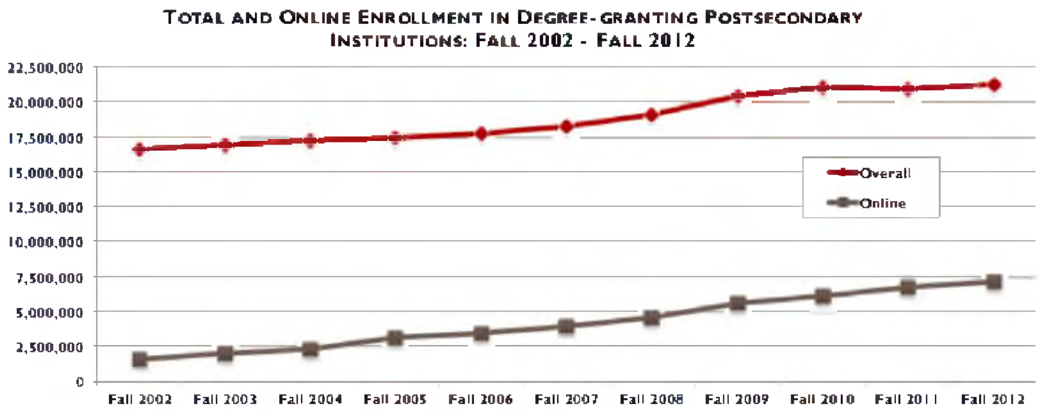


Figure 2.1. Graph Showing Steady Increase in Online Enrollment (from [2]).

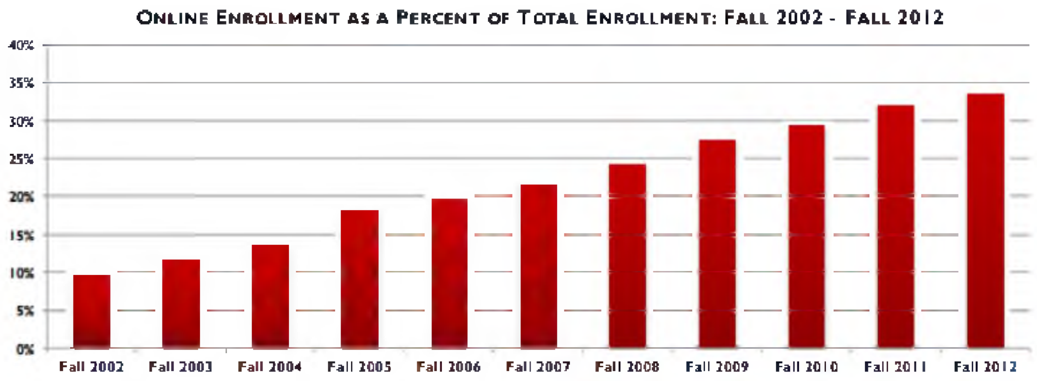


Figure 2.2. Graph Showing Steady Increase in Online Enrollment as a Percentage of Total Enrollment (from [2]).

2.1.2 Active Learning

Active learning is a term that encompasses several alternative approaches to the traditional lecture. Active learning has been a topic of discussion in education for a long time. It is listed as one of the seven principles for good practice in undergraduate education by Chickering & Gamson, which was originally published in 1987 and still widely quoted today [39]. In the early 1900s, teachers of medical students reduced the amount of didactic learning and brought instruction into the laboratory to better prepare them for medicine [40]. Long before either of these pivotal movements, Lauo-tse, a fifth-century B.C. Chinese philosopher, is quoted as saying "If you tell me, I will listen. If you show me, I will see. But if you let me experience, I will learn" [41]. As educators have long considered active learning practices, it makes sense for an educator to consider the value it could add to their classroom.

The current literature typically consists of individual studies examining how a specific implementation of active learning is tested in a particular setting. These very specific observations illuminate how active learning is affecting the learning process. Articles about this practice are scattered across the literature of many different fields. It seems there is even active learning for machines [42].

One of the problems is defining exactly what active learning is [43–45]. Definitions vary dramatically in the literature, ranging from interaction between fellow students and between student and teacher [46] to the responsibility of learning being placed on the student [47]. Some definitions are so broad, one author argues that active learning can include homework problems, online discussion boards, or written assignments all done outside of the classroom [47].

Other definitions include the specification that activities occur in the classroom [48]. Additional definitions hold a high standard for what is considered active learning. For example, the instruction must allow the students to "create authentic, hands-on learning experiences in order to learn new information" [41]. Some argue that learning by its very definition is active [44]. Considering the breadth of papers available on this topic requires liberally allowing a wide range of definitions. To better understand a specific study, it is important to consider how it is defining active learning.

The use of active learning techniques vary almost as much as its definition. Approaches range from polling systems used to inject one or two questions for audience response into a traditional lecture to problem solving activities that span the entire class period. The list of examples is long and includes audience response systems [46], in-class discussion, case study discussion, short written exercises, role-playing, games, hands-on activities, debate, academic service learning, experimental learning, discovery learning [47], peer instruction [49], thinkpairshare discussion, case study problem solving, round-robin discussion [50], problem solving, collaboration and discussion, animations, technology-enhanced activities [51], process-oriented guided inquiry learning [52], concept maps, collaborative writing, brainstorming, collaborative learning, one minute paper/free write, scenarios/case studies, problem-based learning, team-based learning, case-based instruction, panel discussions, teaching to learn/peer teaching, role playing, drama, and simulations [53].

In contrast, the list of approaches that fall into the passive learning category is much shorter. They include lectures, reading textbooks, traditional homework [47], PowerPoint presentations [54], and instructor-centric learning [41].

With so many techniques considered active learning and so few as passive learning, it is surprising that teachers are not including more active learning events. Meanwhile, it seems that passive learning approaches are still predominant [48]. The student rating data from the IDEA Center (an organization that promotes teacher assessment tools) shows traditional lecture is still the primary approach to teaching [55]. Despite the wide acceptance of the effectiveness of active learning techniques, many educators are reluctant to adopt them. Concerns include the perception that time to cover material will decrease, difficulty in developing new material, and a general reluctance to change established teaching methods [49]. This study also demonstrated that an instructor can implement material developed by another author and realize similar and even better results than the developer. With this in mind, availability of teaching materials based on the principles of active learning could reduce some of the concerns of adoption and increase the likelihood that more educators will incorporate these types of activities. Some groups are making such materials readily available. For example, the organization that promotes process-oriented guided inquiry learning (POGIL) has a webpage with links to several publications that between them contain hundreds of activities [56].

Many studies report improved outcomes with the addition of active learning practices. Specific improvements and how to measure them vary widely. For example, [50] compared first-year students taught using active learning techniques to expected outcomes of more advanced students in the program. Other studies used the same group of students and presented some material using active learning methods and other material using traditional lecture. Then the students were tested on the material to identify how effectively the student had learned the material based on the teaching method used [54, 57, 58]. Many studies included surveys to gather student responses about their view of learning experiences [41, 46, 49, 50, 54, 59, 60]. These studies show that student responses to active learning is strongly positive. Many of these studies included interventions that controlled how the teaching was done, while others surveyed existing instruction without adaptation [43]. Many used pre- and post- testing [43, 58, 61], another used free student recall at the end of the semester [57].

In an effort to identify more accurately areas of improvement using active learning techniques, some have employed Bloom's taxonomy to add specificity [62]. Two studies used pre- and post-tests to analyze knowledge gain. Both used the same group of students for the entire study. One compared the difference between topics taught using active learning formats and those taught using direct instruction [58]. The results of this study showed slight knowledge gains in the lower level questions when active learning was added and significant knowledge gains in the higher level questions. The other compared lab sections where the lab activities were done individually to sections that worked in cooperative groups to complete the same activities [61]. This study saw similar small gains in the lower level questions and high knowledge gains in the higher level questions. Although these studies were comparing different things – addition of active learning practices and the impact of cooperative groups – both demonstrate that the impact of the intervention was greater on higher level thinking. This provides insight into the quality of the value added by the choices educators make in how material is presented.

Despite such positive support for active learning, there are some who challenge its effectiveness. One author compares traditional lecture, active learning, and assimilation learning theory [47]. He provides an example of each method used to teach a module on decision support systems in an introductory information systems course. He then compares them for consistency with epistemological, psychological, and pedagogical research. The author concludes that active learning techniques are better than the traditional lecture, but that higher principles should be applied that offer increased value over active learning [63]. Another study gathered data from introductory biology courses about teaching natural selection [43]. This study used pre- and post-tests to measure knowledge gain. It then analyzed the results of these tests and found no association between the frequency of active learning exercises reported and how much students learned about natural selection. These authors conclude that many of the improvements reported by incorporating active learning techniques may largely depend on the educator being knowledgeable in education research, and that the same results may not be realized by instructors who have less understanding of the learning process and may be implementing these techniques ineffectively.

The research about active learning is broad and diverse. The overall consensus is that much can be gained by incorporating active learning in the college classroom. With the wide range of approaches to active learning, it seems likely that some form would benefit any course. Adding active learning experiences to an existing course is one of the key ways to make changes to and innovate the classroom.

2.2 Existing Systems and Strategies

The existing systems and strategies looked at in this section are related to connecting learning items and the presentation of learning material in a graphical way. These focus on two of the hurdles to be overcome in the current method of presenting learning materials.

The first is the limitation of presenting materials in a linear, sequential order. Although this approach is easy to incorporate, it does not express many of the ways learning items are connected. Curriculum mapping attempts to identify connections between courses beyond the sequential prerequisite listing. There are lessons in this research that provided both ideas to replicate, such as engaging the educator in the process, and things to avoid, like making the process so involved and time consuming that it is often not finished.

The second hurdle is the incorporation of a graphical presentation of learning materials. Displaying things graphically is widely used in many other domains but is still very limited in the presentation of learning materials. Text-based presentations are the standard in the current LMS. In section 2.2.2 we see the use of graphical approaches in the learning environment and studies that analyze the impact of this different way to interact with educational materials.

2.2.1 Curriculum Mapping

A curriculum map is like a roadmap through a curriculum, connecting the different components of the curriculum. For example, it connects learning outcomes to individual courses. Curriculum mapping provides users with an overview of the curriculum that they might not otherwise have [10]. Users are students, faculty members, teachers, curriculum planners, evaluators, and administrators [7].

Curriculum mapping is about representing the different components of the curriculum and identifying how they are connected. These maps serve two key functions. First, they make the curriculum more transparent to users of the curriculum. Secondly, the maps demonstrate links between learning outcomes and learning opportunities [10]. These maps are typically stored in relational databases that contain information about the curriculum, the people using them, and the links between various elements of the curriculum [7].

The process of developing curriculum maps requires the cooperation of faculty members, staff members, and administration. There are significant time demands in preparing and building such maps. The process begins by analyzing what is currently being taught throughout the semester as a course is being delivered. The purpose of this stage is to record what is actually being taught as it relates to the program outcomes. Several courses can be analyzed during the same semester. After the courses are analyzed, a group of faculty members gather to aggregate their maps. This may take a significant amount of time and is often conducted through regular meetings over the course of another semester. This part of the process provided the greatest benefit of creating curriculum maps: an increase in collegiality and collaboration among the faculty members involved in the process [11].

The product of this process is a document that identifies the anticipated learning outcomes and the connection between the outcomes and the individual courses in the curriculum. Most of these documents are charts or tables. They list the courses on one dimension and learning outcomes on the other dimension.

The development of curriculum maps comes with a great deal of challenges. Perhaps the greatest challenge is the high demand for time in human resources to complete the maps. Another problem is getting faculty participation. Once a faculty member has been convinced of the value of curriculum maps, they may become strong advocates of them, but there can be some initial difficulty convincing them of their importance [7].

The amount of time and resources required to create and maintain curriculum maps remains a roadblock for many schools [7]. It is not an easy task [10]. The process takes an extended period of time from beginning to end and requires the commitment of time and effort of several faculty members. This high cost of implementation leaves many departments with curriculum maps in process but not complete [7].

The curriculum map provides several benefits. They provide information for administrators and accrediting bodies by providing an overview of the curriculum and how each course is related to specific learning outcomes for students and faculty. Curriculum maps provide an opportunity to improve the alignment between content taught in the classroom and the expected outcomes of the program.

A surprising outcome is the benefit realized from individual analysis and the collaboration among the faculty during the development of the curriculum map. Faculty reported benefits from looking closely at their own courses. The process of faculty members collaborating to discuss the issues with an eye for improving the program increased the feeling of collegiality and cooperation among the faculty. It was this process that provided the most benefit from curriculum map development. [11, 12, 64].

2.2.2 Seeing Learning Graphically

Much of what is presented in education is text-based, but there are several areas of research proposing the power of visualization. These tools build on the idea that knowledge can be represented in a graphical way to demonstrate the interconnectedness of concepts and that these connections are not necessarily linear.

Concept maps are visual, semantic, node-link representations of knowledge that allow students to formulate and present information from a personal perspective [65]. Concept maps provide a graphical way for both the learner and the educator to describe concepts and the connections between them. This form of spatial-semantic display provides an alternative to traditional linear presentations of information.

Concept maps can be used as a basis for effective studying and learning strategies. For example, they are effective in cooperative interactions, as study aids, as a substitute for traditional text, and for the updating of knowledge [65]. They provide a method for learners to visually review the knowledge they have and capture the connections between what they have learned. Concept maps teach concepts, provide an overview of what is currently being learned, and how it relates to things that have already been learned or will be learned in the future [66].

Hay [67] demonstrates that concept maps can be effective tools in higher education. Specifically, using concept maps can support teachers and learners in engaging in the process of discovery. [68] identifies three benefits of mapping. First, students who can represent and manipulate complex relations in a diagram are more likely to remember and understand those relationships. They are also better able to analyze the individual parts. Secondly, for most people, graphical maps are easier to follow than written or spoken instructions. Thirdly, the work involved in creating the map itself requires active engagement on the part of the learner, which leads to greater learning.

Making and using concept maps has become easier with the availability of digital concept-mapping software. Several tools are now available that make concept mapping as an individual or with a group easier to create and manipulate. These software tools produce high-quality visual representations of knowledge [69].

[70] did a meta-analysis of studies that have been conducted about concept maps in the learning domains of science, psychology, statistics, and nursing. Analysis of these studies showed concept mapping was better at producing knowledge retention and transfer than other forms of knowledge delivery, such as reading texts, attending lectures, or participating in class discussions. Concept maps make learning visible [67].

Another way to get a visual perspective on education is the graphical syllabus. [71] used a graphical syllabus in an introductory finance course and found several advantages. First, it communicated knowledge to students who preferred the nonverbal learning style. Secondly, it acquainted students with visual learning and study tools. Thirdly, it was easier to retain than verbally presented material. The graphic syllabus was better at conveying the big picture and the underlying structure of the course.

Nilson [72] presents many ways that a course can be outlined using visual methods. Though we often think of syllabi as being text-based, contract-like documents, this is an unnecessary limitation. Much information can be presented in a single diagram.

2.3 Technical Components of Work

This section is a review of several technical research areas related to the work of this dissertation. These techniques provide the background and serve as a springboard for many of the technical components of the ENABLE system. Drawing on these practices, ENABLE uses new approaches and combines techniques from a variety of research areas.

2.3.1 Text Analysis Methods

One of the impediments to concise analysis when doing text analysis is the existence of polysemy: terms that are the same but have different meanings. There are many such words. For example: fly (travel), fly (insect), fly (term in the game of baseball); down (a lower position), down (furry feathers); proceeds (continues), proceeds (profits gained). Some are pronounced differently, such as minute (60 seconds), minute (very small); refuse (reject), refuse (garbage); tear (rip), tear (drop of water from eye); wind (moving air), wind (to turn). However, when analyzing just the text, the pronunciation is not considered.

When analyzing a large corpus of unrestricted, domain-independent text for similarity, this type of overlap can skew the interpretation. Consider the polysemy of chicken (bird) and chicken (attitude). It is likely that an article discussing coercion preceding group violence would be identified as related to an article discussing the protection of small farm animals.

This kind of mistaken grouping becomes more problematic when professional fields use common words as technical terms to express a specific meaning that is different than the common usage. In computer architecture a bus is a connection that transfers data. In common usage a bus is a vehicle that carries many passengers. In an unrestricted corpus an article about the recent improvements in a computer's motherboard bus system may be highly correlated with a document about the improvements in Maryland's mass transit bus system.

Instead of comparing single words, it would be more robust to use co-occurrence strategies that search for documents that have multiple words in common. Consider the example of the word tree. In common usage this refers to a category of plants. In the field of genealogy it is a technical term that describes an organizational chart of family relations. In computer science it refers to a specific type of data structure. Even looking at multiple word overlaps, one would likely find several common words in documents that were entirely unrelated. For example: tree, parent, child, ancestor, branch, root, and leaf. In this particular case it may be impossible to distinguish separate topics, even using word order or language patterns, as these would be common between these disparate fields.

Once the corpus is limited to a specific domain, many of these word overlap problems no longer exist. In a corpus of text related to computer science you would no longer have documents related to genealogy or public mass transit. The problem is not entirely limited by restricting the corpus. Consider the word if. This word is a conjunction used in both common and technical text. It is also a selection control structure. This word would exist in computer science documents with both these meanings. Many text analysis algorithms would eliminate it as too common to provide meaning, but in an introductory computer science course it is one of the required topics.

One way to identify that documents are topically related is to count the occurrences of individual words or groups of words (n-grams) within a document. With these counts a word frequency matrix can be created that has words on one axis and documents on the other. The value in each element of the matrix is the count of a word or word group in a single document. This will produce a vector of term counts for each document. These vectors can then be compared for similarity. There are several means to identify similarity and create clusters with these term vectors, including Pearson correlation, K-means clustering, and spectral graph clustering. This process of counting word occurrences has no way to distinguish word meaning and the word 'bus' would be counted the same whether it was a vehicle or a computer component.

This approach will produce a large number of word counts that are not significant. Words like the, and, a, etc. will have large frequency counts and have significant weight during the clustering phase. There are ways to avoid this problem. One is to use a stop-word list – a list of commonly occurring words – to eliminate these frequently occurring words. Another approach is to use a computation called tf-idf [73] to weight the counts, giving less weight to terms that occur in more of the documents and greater weight to terms that only occur in a few documents.

Another problem with using co-occurrence counts is that different versions of a word such as code, codes, coding, and coded are counted as different words. There are stemming algorithms that can be applied to the text to reduce all the words in the text to the stem word. The words code, codes, coding, and coded would be replaced with the stem word code and they would all be counted as the same word. A more difficult problem is synonymy, when different words have the same meaning. Word frequency occurrence counts will count synonyms separately and not consider them the same topic even though these words are accurately describing the same topic.

A process that overcomes many of the problems associated with simple co-occurrence counting is probabilistic latent semantic analysis (PLSA) [74]. PLSA is a statistical technique for the analysis of co-occurrence data. It can use word frequency counts, tf-idf values, or other computed values. These values are stored in a matrix with words in one dimension and documents or portions of documents on the other dimension. As with the basic co-occurrence approach described above, this produces a sparse matrix. From this sparse matrix PSLT derives a low-dimensional representation of these observed words in terms of their affinity to hidden variables. It then uses a generative latent class model to perform a probabilistic mixture decomposition [75].

This process produces results that reflect the semantic meaning of words based on where they occur in the passage [76]. It is able to overcome the polysemy problem by rejecting associations between documents that contain the same words but have different meanings. PLSA associates a latent context variable with each occurrence of a word, which explicitly handles words with multiple meanings. How a word is used in a document varies based on the topic mixture. The probability of using a specific word depends on the topic [77].

This process of associating a latent variable with each occurrence of a word also handles synonymy. The use of the words in the document will reflect that they have the same meaning. PLSA also computes meaningful associations between pairs of documents, even when they do not have any common words [75, 78].

PSLA can work on a small set of documents but shows marked improvement as the training set moves out of the very small size [79]. This provides motivation to include a training corpus when possible. Many forms of text classification fail when applied to this kind of short and sparse text [80].

2.3.2 Graph Transformation

When making changes to the nodes and edges in a graph, it is important to transform the graph while still keeping the relations intact. Graph grammars and graph transformation systems provide a means for doing this. There is much research and many successful applications based on the research in this area [16]. One of the application areas of graph transformation systems is model transformations. This area of model transformation has become important to the field of software engineering [17]. The models used in software engineering have enough similarities to the graphical representation of learning materials. Those similarities include typed nodes, node attributes, and edges that represent different types of relations.

A comparison of some currently available tools is below. Specifically, this review is limited to model transformation tools that are based on approaches that draw from the theoretical work on graph transformations [23]. The four tools compared are AGG, AToM3, VIATRA, and VMTS. [20] used and compared these four tools to demonstrate how they each solved the problem of transforming class diagrams to relational database models. In [20] the strengths and weaknesses of these tools are compared. Below, the tools are compared as they relate to the work of developing the ENABLE system. The following information was gleaned from [20, 23] and the current website for each individual tool.

2.3.2.1 AGG

AGG is a development environment for attributed graph transformation systems, and can be downloaded [81]. It is based on an algebraic approach to graph transformations. The implementation of this approach closely follows the formal, theoretical foundation of algebraic graph transformation and so provides validation support [19] and sound behavior concerning graph transformation [20]. This clear and consistent adherence to the formal rules of algebraic graph transformation is this tool's strength. It uses attributed graphs. It has some additional capabilities, such as allowing deletions from the source model and helper structure. AGG has a nondeterministic rule and match selection but provides control of this with rule layers.

Another strength of AGG is that rule applications can be applied in either interactive mode or interpreter mode. After choosing a rule in the interactive mode, a match can be given element-wise by the user or automatically computed. In the interpreter mode, rules are applied as long as possible.

AGG aims to provide rapid prototyping of applications with complex, graph-structured data. Transformations can be applied at different levels of abstraction. Attributes can vary significantly from none to complex processes. AGG supports typed graph transformations, type inheritance, and allows the multiplicity of relations.

The disadvantage of AGG is that the interface and output are very simplistic. It appears to be in maintenance mode, with only minor revisions taking place. Another disadvantage is the user has to specify similar rules individually for both attributes and associations.

2.3.2.2 AToM^3

AToM^3 (A Tool for Multi-formalism and Meta-Modeling) is a tool for multiparadigm modeling, and can be downloaded [82]. It was developed as a tool to design domain specific visual languages. The user can define the syntax of the meta-model by using abstract or concrete syntax. The meta-model generates a customized modeling environment for the described language. It can generate environments for multiview visual languages like UML.

Source elements cannot be deleted. The tool allows incremental transformations. Rule scheduling includes both interactive rule selection and priorities.

AToM^3 uses an algebraic approach to graph transformations that allows both the double-pushout (DPO) approach and the single-pushout (SPO) approach. This is a strength since it allows the user to choose whether to use the more conservative DPO approach or not. This tool also uses triple graph grammars, which means it has three separate graphs: the source, target, and correspondence graphs.

Another advantage is that it keeps source and target models separate, keeping the intermediate model hidden for internal use only. This is a cleaner approach because interim states are in the correspondence graph, so there is nothing left to clean up in the target graph.

The significant disadvantage of AToM^3 is that it is no longer being developed or maintained. Another disadvantage is you have to specify similar rules for both attributes and associations.

2.3.2.3 VIATRA

VIATRA (VIsual Automated model TRAnsformations) is a framework whose main objective is to provide general purpose support for the entire lifecycle of engineering model transformations, and can be freely downloaded. This tool has been added as a subproject to the open source IDE, Eclipse, and currently has an active development team. It is in the Eclipse incubation phase – a phase new projects are in for one to two years after being added to Eclipse. This incubation period allows the Eclipse community to help a project learn the ropes of creating open source developer, adopter, and user communities, and is not a reflection of the state of the project code base. With the precondition VIATRA supports recursive graph patterns and attribute conditions. Actions and control are accomplished using abstract state machine (ASM) rules. The use of ASM rules is one of this tool's strengths. It provides complex model transformations with many powerful control structures, including a sequencing operator, rule calls to other rules, if-then-else structures, nondeterministic selected and executed rules, and iterative execution. This gives users the ability to build state machines to schedule transformation rules. VIATRA combines the rule- and pattern-based standard of graph transformation and the general, high-level standard of ASM into a single framework.

Another strength of VIATRA is the features it employs to improve performance. The application of elementary algebraic graph transformation rules is frequently driven by complex control structures, which reduces nondeterminism and improves run-time performance.

Additionally, input and output parameters may be passed to a rule or a pattern to improve performance. The forall ASM control structure provides a more efficient solution, which handles all the matches of the rule in parallel.

VIATRA uses triple graph grammars, which means it has three separate graphs: the source, target, and correspondence graphs. It also uses the VPM metamodeling approach [22], which supports arbitrary meta-levels making models from very different domains easily integrated into the model space. Model queries are captured intuitively with recursive graph patterns. The disadvantage of VIATRA is it does not have a graphical interface to create and modify patterns.

2.3.2.4 VMTS

The Visual Modeling and Transformation System (VMTS) is a graph-based, domain-specific (meta)modeling and model processing framework that can be downloaded. One of the major strengths of this tool is that it is still actively under development. It was created in 2003 and its current team is actively engaged in ongoing development. The current version is VMTS4, and the team is working toward a fully configured workbench [18]. In its current state it is very useful and well designed. The meta-model is defined in a visual editor and the interface and output are nicely done.

A noteworthy feature is that it can actually generate code in either JavaScript or C. Code generation is one of the major motivations of model transformation research [83,84].

VMTS allows entity and data types, typed references to other elements, both abstract and built-in attributes. It also allows custom types and derived attributes. It uses the object constraint language (OCL) for adding constraints. It follows the DPO rules, and the DPO gluing conditions are enforced by the tool.

The downside of this tool is that it is specifically designed to work with the Unified Modeling (UML) models. The VMTS Control Flow Language (VCFL) is a stereotyped UML activity diagram. It uses OCL to specify pre- and post-conditions, constraints, and to validate correctness. Although it allows flexibility and definitions within the UML model, it appears to be less able to work with other types of models.

In [20] there is a table comparing the above four tools with Query/View/Transformation (QVT), a standard set of languages for model transformation defined by the Object Management Group. When comparing the model transformation approaches, solutions, and tools with QVT, the four tools are far more alike than they are different. None of them match QTV completely and all add tools and solutions that the QVT standard does not include. These four approaches to the model transformation problem are in many ways similar and are built upon the solid practices developed through research of graph grammars and graph transformation systems.

2.3.3 Automated Agents

An automated agent is a computer system that exists in some environment and is capable of flexible autonomous action within that environment in order to meet its design objectives. These agents are used in many applications covering a wide range of systems that vary from small email delivery systems to large, complex, mission critical systems, such as air traffic control [85].

Agents are defined by the way they perceive their environment and how they act on that environment. Perceiving the environment is done through sensors. Sensors range from cameras and infrared range finders to keystrokes and data. Acting on the environment is accomplished with actuators that might be mechanical devices operated with motors or output displayed to a screen [1]. Both the way an agent perceives its world and how it acts on its environment are very different based on the purpose of the agents. This promotes the use of agents in a broad range of activities. The research presented here focuses on their use in educational applications.

Automated agents are used in a variety of educational applications. Intelligent tutoring systems have been in development and use for decades. An intelligent tutoring system is a system in which a computer simulates a tutor [86]. [87] describes the major intelligent tutoring systems that were implemented prior to 1981. These covered a range of subject areas, including arithmetic, informal gaming, electronics, and medicine. Later, many intelligent tutoring systems incorporated the use of automated agents [88–92].

A form of automated learning companion is the pedagogical agent. [93] defines a pedagogical agent as an animated character that facilitates learning in computer-based learning environments. Pedagogical agents are built on the research of intelligent tutoring systems, adding both a different interface as well as the objective to not only be a tutor but also a companion in learning [86]. This addition of lifelike characteristics may improve engagement and increase the enjoyment of learning. Pedagogical agents use nonverbal communication techniques to aid learning and provide feedback to the student [94]. They use voice and gestures to focus student attention on important information. The addition of an interactive, lifelike presence in computer-based learning may add elements of social interaction that have been shown as beneficial to the learning process [95]. This feature may have significant impact in educational settings where learners are remote and lack physical contact with fellow learners and instructors [96].

Adding a pedagogical agent to a computer-based learning system has mixed results. In [97], the authors reviewed five studies. Only two of the studies used control groups for comparison. Of those two, only one found improvements in transfer, motivation, and interest, but no improvement in retention [98]; while the other found no difference between the two groups [99]. [92] found that student enjoyment was increased and their perception of the level of difficulty was decreased when there was a pedagogical agent. However, the presence of the agent did not improve the short-term learning outcomes.

A study that included comparing pedagogical agents with differing levels of appeal found that while the addition of a pedagogical agent had little impact on learning or motivation, the use of an unappealing agent can actually hinder learning. The author recommends careful consideration of the design of the character itself [100]. Another study compared a pedagogical agent that gave adaptive advice based on student behavior as they navigated the learning environment, the same pedagogical agent that gave general advice at regular time intervals, and no pedagogical agent at all. This study found that students performed better when a pedagogical agent was present and that those who were given regular interval, general advice outperformed those who were given the more specific, adaptive advice [101].

There have been several more recent studies on the use of pedagogical agents. These studies focus more on comparing different pedagogical agents rather than whether there is a benefit to using pedagogical agents at all [102].

Often, educational agents are designed to replicate the way interactions are currently done in educational settings. In [103] the authors question the validity of this, based on the increasingly accepted tenet that much of what is currently done in education is less than effective. They attempt to use agents in new and different ways to promote improved learning opportunities. Building on the premise that teaching is an effective way to learn, they have developed a teachable agent that provides students the opportunity to teach and, in the process, learn themselves.

Virtual worlds are finding a place in education, and intelligent agents are finding a place in those virtual worlds. Open Wonderland is an open source toolkit for creating 3D virtual worlds [104]. This toolkit has been used to create virtual worlds for teaching foreign language [105, 106], creating simulators for a computer science lab [107], learning about agile software development [108], and conducting a natural science experiment [96]. Using intelligent agents in this new educational format has been explored by [96, 109].

Intelligent agents have found a variety of applications in the field of education. The use of these agents has been the basis of many research projects. However, there are many remaining research questions about the use of intelligent agents and how they improve education [95, 102].

2.3.4 Student Modeling and Tracing

User modeling is used by a wide range of systems. Specifically, student models have been created and used in educational systems. Most of the work involving student agents has been done in the field of intelligent tutoring systems. These learner models developed in the research laboratory are now used in advanced commercial learning environments. These intelligent learning systems have successfully integrated learner models and have achieved widespread usage [24]. The study of student modeling and its potential usage continues to be an active area of research.

2.3.4.1 Variety of Models

Centralized modeling can be represented by user modeling servers or user modeling shells [110]. A central user modeling system gathers and stores a variety of information about the students from multiple components of the system. This information is stored by the central user modeling server. This approach allows a lot of flexibility in how information is produced by the different components of the system [111].

Decentralized modeling is widely used in agent-based architectures. Rather than information being gathered and stored on one server, it is stored among the individual components of the system [111].

Constraint-Based Modeling (CBM) represents the domain knowledge as a set of constraints. These constraints are used to analyze student solutions and provide feedback on errors [112]. This is a form of problem solving using solution analysis tutors. Another type of model that falls in this category are Cognitive Tutors (CT). Both CBM and CT are distributed commercially and have hundreds of thousands of users [24].

Another approach to tutoring systems is curriculum sequencing. Curriculum sequencing consists of defining learning paths through a set of learning objectives and didactic content. These systems have to build a path through a large number of items when little evidence is available. This ratio of the amount of evidence to the breadth of the assessment can be critical for systems that cover a large set of skills. This requires a model that can build links between learning items. This relies on a transfer model – a model that can transfer evidence between items and skills that are not directly linked [24].

2.3.4.2 Representing the Student

A student model identifies particular traits or features of the student that it will represent. These features vary and may include the user's knowledge, interests, goals, background, and individual traits [113]. Attitudinal factors can also be considered, such as student motivation or level of boredom or frustration [24]. In addition, some models represent a context of the user's work [113].

The scalar model is the simplest of models and estimates the level of the user's domain knowledge by a single value. This value can be represented by a numeric range such as 0 to 5 or use a qualitative scale such as good, average, poor, etc. Simple as they are, these models can be effective in supporting simple adaptation techniques. The scalar model averages the users' knowledge over the entire domain. When more specific detail is needed, structural models consider that domain knowledge can be divided into independent fragments or concepts [113].

Another approach is to use a stereotype model. In this model the user is represented by a collection of attributes. These models can provide an important role in modeling because they allow the system to make plausible inferences based on a limited amount of observations. These inferences are then overridden and the individual student model is updated when observations contradict them. The inferred values are abandoned in favor of learned facts [113,114].

These student models can be constructed based on the use of rational analysis, user self-evaluation, objective testing, and other manual analysis techniques. These manual techniques for constructing models are often time consuming and require expert input. In [115] the authors demonstrate the effectiveness of constructing student models using machine learning techniques. Other data mining and machine learning techniques are presented in [116] and can be effectively used in systems that have large amounts of data.

These student models are adaptive models in that they update the representation of the student as more evidence becomes available. For example, when a correct or incorrect answer occurs, the state of that student's knowledge can be updated based on this new evidence. This update process is made more complex by the existence of slips and guesses. A slip is when someone gets an incorrect answer even when they have the knowledge, while a guess is when a correct answer occurs even when the knowledge does not. [117] presents an approach where machine learning is used to estimate the probability that a correct answer is a guess rather than evidence of knowledge, and the probability that an incorrect answer is a slip rather than evidence of lack of knowledge.

2.3.5 Bayesian Analysis

Bayesian inference is used broadly in a wide range of applications, including science, engineering, medicine, and education. The area of application that is most closely aligned with the current research work is education. Even within education it is used in a variety of ways. [113] states that Bayesian networks are the most important approach to knowledge modeling.

A Bayesian network (BN) can be used to represent uncertain relationships between items in a domain. A BN is a directed acyclic graph (DAG). The nodes in this graph represent random variables and edges between the nodes are a probabilistic correlation between those variables. The full joint probability distribution can be represented by a set of conditional probability tables (CPT). Although probabilistic inference can be computationally intractable in some cases, it can be done efficiently in many practical situations [1,28].

The CPT can be set using a combination of expert-derived estimates and data-driven approaches [118]. However, requiring experts to assess content mastery is often impractical. [24] outlines some of the key simplifying approaches that are used to determine conditional probabilities. Other projects used data-centric methodologies in which the structure of the BN was developed from actual student data [112].

In [29] a Bayesian network (BN) was used to identify probabilistic relations between courses in a prerequisite course structure. This project used the predictive capability of the BN to consider ways to improve student retention and assist student success. They were able to identify courses that were critical in the overall program and would be responsive to tutoring support.

Educational testing can be enhanced by the use of BNs. In [118] the authors use the BN to increase the amount of details that were extracted from student testing. They were able to identify gaps of knowledge, student abilities, and weak points at the beginning of the course. Other research projects used a BN to identify student learning styles [26,28].

A large area of BN use is in the intelligent tutoring field. The modeling tasks in an interactive intelligent tutor involve a high level of uncertainty. These systems infer student knowledge of specific concepts from the interactions they record and provide knowledge assessment, plan recognition, and prediction of student actions during problem solving tasks [27]. In [25] researchers used a BN to add individualization to the standard knowledge tracing capabilities of previous tutor systems.

A BN was used to develop a probabilistic model for an SQL-Tutor used to make decisions about which problem to give the student next. The BN was used to make predictions on specific constraints relevant to that problem. The system was a constraint-based system [112].

There are a variety of ways to analyze the effectiveness of the BN. Actual data can be separated into a test set and a learning set. The BN can be trained on the learning set and used to predict the values of the test set [29]. K-fold cross-validation, Bayesian Information Criterion, and validating assessments with external measures are ways to control for over-fitting [24]. [117] tests for model degeneracy, empirical degeneracy, and limited degeneracy on a newly developed guess and slip model.

CHAPTER 3

COURSE CONTENT ANALYSIS

A beginning point for initiating change is to discover where one currently is. The first stage of this work is to discover what data are currently available from the existing course and identify if information can be extracted from that data to inform the process of change. An enhanced representation of the course will benefit from both syntactic and semantic analysis. Is that information available? Is it directly available or will it require manipulation to attain? Can the data available through the learning management system (LMS) be enhanced by expert knowledge? These questions provide the direction for the course content analysis stage of this work.

ENABLE is able to extract syntactic information about the content and structure of the course from the data available in Canvas. Some information is directly available, such as due dates and unit associations. Analysis of the data provides additional information, such as topical connections and learning item type. Much of the semantic information required expert knowledge. When acquiring information from an expert, ENABLE provides an interface that leverages the information extracted from the data in the LMS to aid the expert. This process engages the instructor in the analysis phase and provides them a different view of the existing course.

3.1 Gather Data from Learning Management System

This study is about course transformation. This implies that there is an already existing course. The first step is to acquire the details available in that existing course. The courses we will analyze use Canvas, a commercially available learning management system (LMS). Data about the course and the learning items in the course are available through this LMS.

3.1.1 Using the API

Canvas has a well-documented application program interface (API). This API allows programmatic access to much of the data that resides in Canvas. The Canvas API uses the REST architecture and can be accessed using HTTPS.

The canvas LMS API documentation is available at `https://canvas.instructure.com/doc/api/index.html`. This documentation specifies how to request data from canvas. Access to the data requires authentication, which is done with OAuth2, a protocol for authentication and authorization. This requires that an access token be included in the header or sent in the query string.

An access token can be acquired through standard login access to canvas. These tokens are available to those who have administrative or teacher accounts on canvas. To get an access token, go to settings in Canvas and select the New Access Token button. This provides an interface to generate a token. You can put an expiration date on the token or leave it blank, which will leave the token without an expiration date. Once you generate the token, it is no longer available from the system. It is up to you to keep track of it.

Parameters for POST and PUT requests are sent using standard HTML form encoding. ENABLE uses the PHP client URL library (cURL) to make these requests. This library provides many functions that allow connection and communication with many different types of servers using a variety of protocols.

3.1.2 Details about Learning Items

3.1.2.1 Getting Data from Canvas

The API is used to acquire details about individual learning items. A course may contain many different materials that are used for instruction. For this work, these various materials are referred to as learning items. Each learning item has its own characteristics, such as title, due date, delivery method, and whether it is graded or not. ENABLE uses four general categories to classify learning items: exam, assignment, resource, and quiz.

In Canvas the learning items can be found as assignments, pages, and files. These are terms used in the Canvas system. In Canvas an assignment has points associated with it and a specific due date. Pages are a way for instructors to build content and can contain text information, links, videos, and other media. Files can be of many types and are uploaded from an external source and stored in Canvas.

ENABLE accesses the following details about assignments:

- id
- name
- description
- quiz_id, this is available for assignments that have an associated quiz

- due_at
- assignment_group_id
- points_possible
- published

For pages it acquires the following:

- title
- body
- published

For files it acquires the following:

- display_name

3.1.2.2 Building Learning Items in ENABLE

To build learning items in ENABLE we begin with the data that come from Canvas. The id, name, quiz_id, due_at, assignment_group_id, quiz_id, and points_possible are added to the learning item without change. The published status is used as a filter. Only assignments that are published are built into learning items.

The type of the learning item is determined. This type will be used in the user interface to provide information to the instructor. There are three types: assignment, exam, and resource. If the data do not have points_possible, the type is set to resource. This happens in the case of a page or a file. For the remaining items it needs to be determined whether they are assignments or exams. To make this determination the name is parsed to identify if it contains the word exam or test. If either of these words are contained in the title, type is set to exam. For all other items the type is set to assignment.

The description that comes from Canvas contains the body of the material. The description will be used when determining relations. For those purposes, it is helpful for the description to be as complete as possible. To the description ENABLE adds the text of the title. In the case of a quiz, the text of the quiz questions are added to the description. String parsing is used on the description to determine if there are any references to files. If there is a reference to a file, a request is sent to the canvas API to get the name of that file. The file itself is not directly available from Canvas. The actual file can be added to the files folder. The system looks in this files folder for a file with the specified name. If the file is a plain text file, that text is added directly to the description. If the file is a .pdf, a conversion process is executed and the text extracted. This extracted text is added to the description. This process provides a more complete description of the learning item.

The name as it comes from Canvas is useful in some situations, but for the course map a shorter name is needed for labeling purposes. Since the purpose of the labeling is for the understanding of the instructor, it is important that the labels are meaningful to them. To accomplish this, an interface is provided that allows the instructor to edit the labels. The interface provides the full name of the learning item and the first four characters. These four characters may or may not be meaningful. The instructor is able to edit these labels to their liking. The left side of Figure 3.1 shows the labels that were extracted directly from the names of the learning items in Canvas. The right side of Figure 3.1 shows the labels after editing by the instructor.

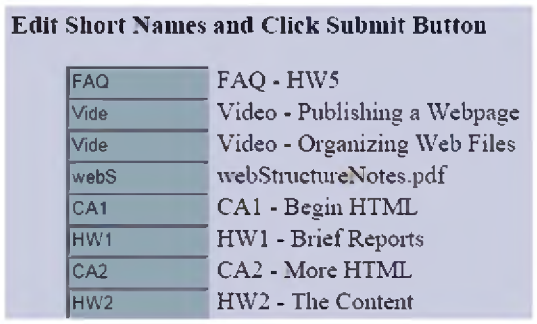
3.1.2.3 Getting Information from the Instructor

Additional information about the learning items is gathered from the instructor. ENABLE provides an interface that allows the user to upload files (see Figure 3.2). The textual data from these files is added to the description of the learning item. This allows an instructor to add whatever textual data they consider important to the learning item. In some cases a learning item may have very little textual information available in Canvas. For example, an exam that is given on paper to the students has very little textual information available in Canvas. In this situation Canvas would have the title and the due date. This user interface provides a way for the body of the exam to be included in the description of the learning item. Files can be uploaded for learning items, like exams, that have no textual description, as well as for items that already have textual data. The text that is uploaded is simply added on to whatever description was extracted from Canvas.

ENABLE also provides an interface that allows the instructor to add learning items that did not exist in Canvas (see Figure 3.3). This allows the instructor to broaden the scope of what is included as a learning item. For example, a lecture may not appear at all in Canvas and yet the instructor may be interested to include the lecture material in the course analysis. The interface allows the user to enter a name, a date, points possible, and topics included. Additional description can be added through the file interface just described.

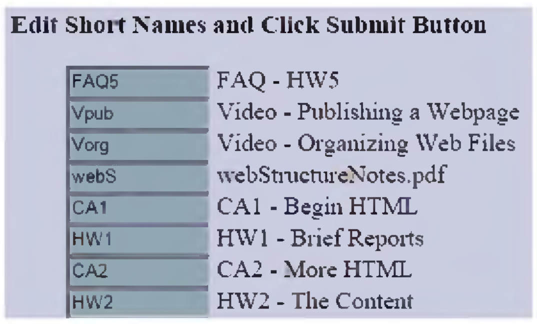
3.1.3 Details about Course Organization

Another thing that needs to be established is the current organization of the course. The due date is used to establish the temporal ordering of the graded learning items. There is



Edit Short Names and Click Submit Button

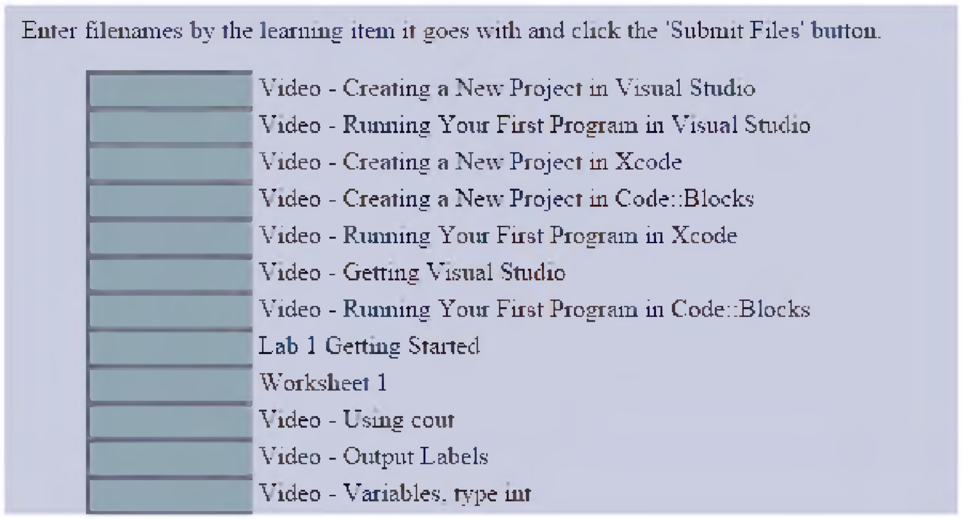
FAQ	FAQ - HW5
Vide	Video - Publishing a Webpage
Vide	Video - Organizing Web Files
webS	webStructureNotes.pdf
CA1	CA1 - Begin HTML
HW1	HW1 - Brief Reports
CA2	CA2 - More HTML
HW2	HW2 - The Content



Edit Short Names and Click Submit Button

FAQ5	FAQ - HW5
Vpub	Video - Publishing a Webpage
Vorg	Video - Organizing Web Files
webS	webStructureNotes.pdf
CA1	CA1 - Begin HTML
HW1	HW1 - Brief Reports
CA2	CA2 - More HTML
HW2	HW2 - The Content

Figure 3.1. The Left Side has Labels as Extracted From Titles. The Right Side Shows Labels After Editing.



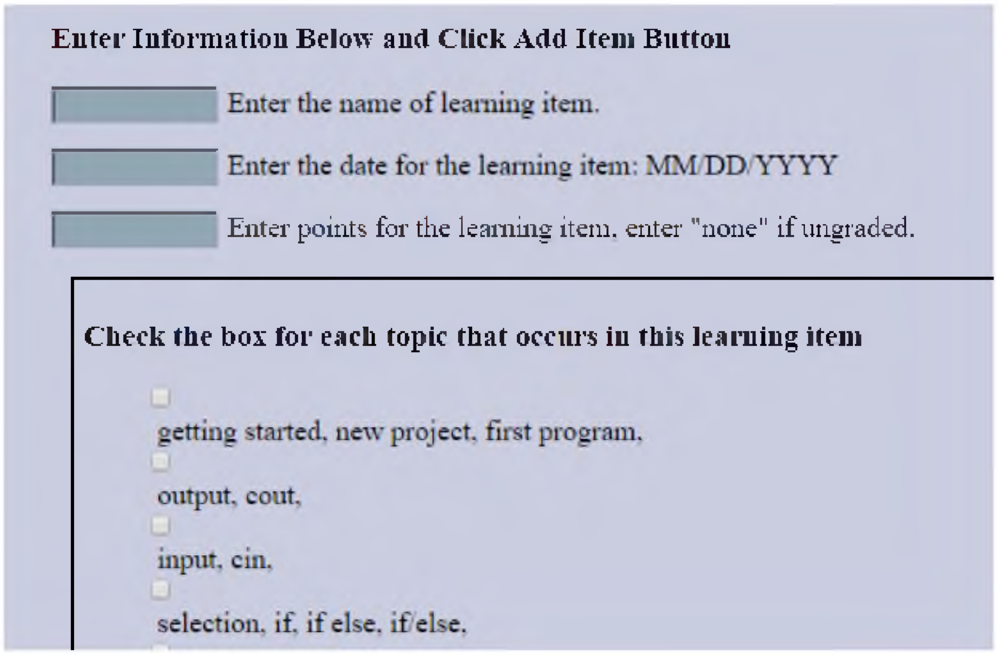
Enter filenames by the learning item it goes with and click the 'Submit Files' button.

	Video - Creating a New Project in Visual Studio
	Video - Running Your First Program in Visual Studio
	Video - Creating a New Project in Xcode
	Video - Creating a New Project in Code::Blocks
	Video - Running Your First Program in Xcode
	Video - Getting Visual Studio
	Video - Running Your First Program in Code::Blocks
	Lab 1 Getting Started
	Worksheet 1
	Video - Using cout
	Video - Output Labels
	Video - Variables, type int

Figure 3.2. Interface to Add Additional Text For Learning Items.

no date information for the ungraded items. Their order is assigned based on which module they are contained in. They are added in order before any assignments, exams, or quizzes in the module they are in and after items in preceding modules.

Modules are a way to organize learning items in Canvas. Canvas allows the instructor to create a module and populate it with any of the types of learning items. This grouping of learning items is another form of organization that can be gathered from Canvas. The learning items contained in a single module are identified to be in the same unit. The unit is simply a set of learning items that have been grouped together for some purpose. Canvas instructors use the module grouping in a variety of ways. Some group them based on a textbook. For example, putting all the learning items related to a single chapter or section in a module. Others use it as a form of temporal ordering and group items together


Enter Information Below and Click Add Item Button

Enter the name of learning item.

Enter the date for the learning item: MM/DD/YYYY

Enter points for the learning item, enter "none" if ungraded.

Check the box for each topic that occurs in this learning item

- ☐ getting started, new project, first program,
- ☐ output, cout,
- ☐ input, cin,
- ☐ selection, if, if else, if/else,

Figure 3.3. Interface to Add Additional Learning Item.

based on some date relation, such as all the items to be covered in a week. Another way to organize these groups is by topic, having a module group items covering a single topic together.

The information available from Canvas is a module ID for each module. Each learning item contains the module ID for the any module it is a member of. It also contains a value that identifies its order in the module.

3.2 Discover Relations

ENABLE not only wants to establish information about individual learning items, it also wants to discover the relations between those items — e.g., whether one item must be covered in order to prepare the student for another item, whether items cover a similar topic, etc. Some information about relations can be extracted from the data in Canvas. Other relational information can be gathered from the instructor.

3.2.1 Temporal Relations

Temporal relations have to do with how learning items are related over time. The due date data extracted from Canvas are used to establish this temporal ordering. Using this date information, ENABLE organizes the learning items using the precedes relation. The *precedes* relation expresses that one learning item comes before another in time. For example, Activity 5 *precedes* Homework 3.

Using this temporal ordering, the first learning item is preceded by nothing. The second learning item is preceded by the first. The third learning item is preceded by the second and the first learning item. The fourth learning item is preceded by the third, second, and first learning item. This pattern continues until you get to the final learning item, which is preceded by all other learning items. The *precedes* relation connects one dated learning item to all other dated learning items. An exception would occur when two learning items had exactly the same due date and time.

When each learning item is included as a node in a directed graph, these precedes relations become the edges between the nodes. This produces both in-edges and out-edges for each node. A learning item at location k in the sequence of this graph has $k-1$ in-edges and $N-k$ out-edges, where N is the number of learning items. This produces far too many edges in the directed graph to be informative (see Figure 3.4, upper).

To manage the excessive number of edges this produces in the graph, the transitive nature of the precedes relation is exploited. The transitive rule for the precedes relation can be expressed this way. If Item A *precedes* Item B and Item B *precedes* item C then Item A *precedes* Item C. Using this rule, the edges expressing the precedes relation can be reduced to the edges connecting adjacent nodes (see Figure 3.4, lower). Each of these edges expresses the immediately precedes relation, which can be stated Item A *immediately precedes* Item B. This means that Item A comes directly before Item B such that when all items are temporally ordered, there is no item that comes after Item A and before Item B.

This graph is easier to read as every item except the first item has exactly one in-edge, and every item except the last item in the graph has exactly one out-edge. This produces a graph with $N-1$ edges, where N is the number of learning items. Because of the transitive nature of the precedes relation, the immediately precedes relation captures all the precedes relations in the previous graph. This property allows us to continue to refer to these more specific immediately precedes relations as precedes relations without loss of accuracy. It also continues to be correct to state that Item A *precedes* Item G whenever Item A comes before Item G temporally, even if there is not a precedes relation edge directly connecting Item A to Item G.

The meaning of the precedes relation is limited to a temporal relation only. Learning items connected in this way are not necessarily related by topic or grouped in the same unit. The precedes relation does not infer any other relation between adjacent, temporally ordered nodes. There is no assumption of a prerequisite relation. A prerequisite relation

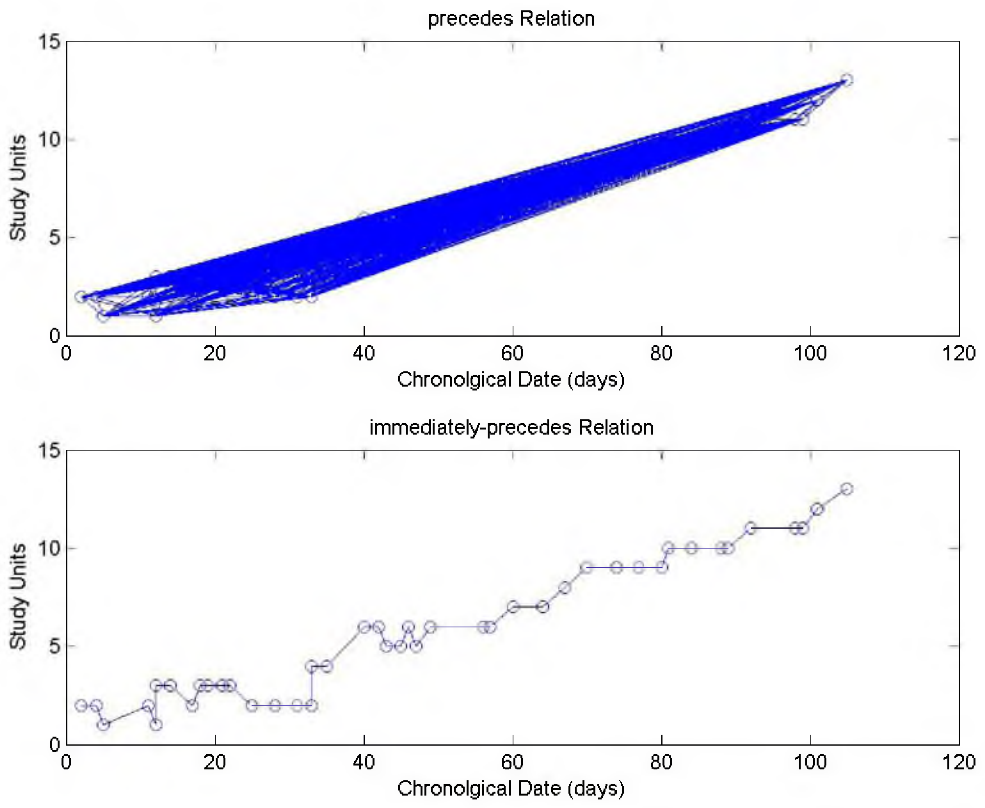


Figure 3.4. Detailed Graph of Precedes Relations (upper). Graph of Immediately Precedes Relation (lower).

expresses that there is benefit in completing one learning item prior to working on another learning item.

The temporal relations between learning items are the most predominant relations presented to both students and educators in an LMS. An educator who has designed and implemented a course is aware of other relations between the learning items, such as how they are grouped together to create a unit of learning, how they are related by a single topic or a group of topics, and prerequisite recommendations. Although the educator may consider these other relations more significant, the learning management systems currently available use the temporal relation as the dominant organizational aspect when presenting learning materials. In Canvas, learning items are presented in this temporally based order in the syllabus, the assignments page, the grades page, and the calendar.

When combined with topical and unit relations, the temporal relations add some information. For example, if there are two assignments that cover the same topic and one precedes the other, it is likely that there is a noncommutative relation between the two learning items, and it is important that the first is completed before the other.

Not all items in Canvas have due dates. In the sample CS0 course, 8 out of 49 learning items do not have due dates associated with them. These undated learning items include lecture notes, videos, and frequently asked questions. It is likely these items are informational materials that are most beneficial when preceding other items in the same unit or other items of the same topic. As topics extend across a larger time period than units, associating the undated learning items with other items in the unit is preferred. These learning items are dated two days before the first dated item in their unit.

3.2.2 Topical Relations

To identify the topical relations, the text of each learning item is gathered. Canvas stores a title or name and a text description of each learning item. This information can be acquired through the canvas API. The title and text description become the basis of the text. This text is analyzed to see if there is a link to a file. Canvas has a specific way of referencing files that have been uploaded, making it possible to use text parsing and regular expressions to identify these references. Once a filename is identified, the file extension is considered to see if it is one of the file types that can be used. Currently ENABLE is able to add .txt and .pdf files to the text description. Pdf files first need to be converted to text before being added. This conversion process is handled in ENABLE, and the plaintext added to the description. One type of learning item is a quiz. These quizzes contain individual questions in addition to the text description. For these types of learning items, the text of the questions and answers are gathered through the canvas API and added to the text description.

ENABLE uses a .txt file to store a series of topic lists. These lists contain topic words, word groups, and variations. Each line in the file represents a single topic. Individual topic variations are separated by a comma. The original list of topics for the sample CS0 course includes

1. <content>
2. <html, structure>
3. <attribute, attributes>
4. <tag, tags>
5. <element, elements>

6. <publishing, host, publish, published>
7. <careers, career, careers in cs, cs careers>
8. <darpa>
9. <history, cs history>
10. <css, style>
11. <hardware, system>
12. <javascript, script>
13. <functions, function>
14. <textboxes, textbox>
15. <using the web, use the web>.

The use of a list of topic variations allows different versions of the same word such as **publishing**, **publish**, and **published** to be counted as a single topic. Stemming algorithms [119] can be used to accomplish this same grouping, but these algorithms do not allow the inclusion of additional words that are not from the same base word. For example, the word **host** is included in the list with **publishing**. The list is not restricted to words that share the base word. Any combination of words can be included in a single list. This allows the instructor a great deal of flexibility in associating a variety of words or word phrases with a single topic.

Using these lists of topic words, a term frequency vector is created for each learning item document. Term frequency (*tf*) is a count of how many times a term occurs in the learning item document [120]. The document in this case is the description of the learning item. This description includes any text available in Canvas or uploaded by the instructor. The frequency count of terms found in a topic list is combined to produce a single *tf* count for each topic.

When computing *tf* for all the terms in a corpus of documents, this process produces high-dimensional, sparse vectors [80]. Techniques such as the application of singular value decomposition (SVD) to a topic similarity matrix (i.e., spectral graph analysis) may allow the reduction of dimension to make computationally intensive text analysis more efficient [121]. In the CS0 example here, the limited number of specific terms found in the topic lists produced *tf* vectors for which no dimensional reduction was possible.

In ENABLE, the corpus of documents consists of the description for each learning item. Singular value decomposition (SVD), spectral graph analysis, and K-means were all applied to the frequency matrix that was produced by combining the individual frequency vectors of each learning item. In the sample courses, no dimensional reduction was possible. This is a very small corpus and the sparseness of the resulting matrix was not excessive (see Table 3.1, which shows the frequency matrix after topic reduction). This small size is exaggerated by the fact that only the topic words are included in this analysis. Often all words found in the corpus are included in the frequency counts, resulting in a much larger and more sparse frequency matrix.

This raw count of how many times a term occurs in a document can be more informative if it is weighted. The weighting approach used by ENABLE is *tf-idf*. *tf-idf* starts with the *tf* and then multiplies it by the inverse of the document frequency. The document frequency (*df*) of a term is the count of how many of the documents in the corpus contain that specific term. If the *df* is high, the term is very common, so the fact that it shows up in a document is not as significant as a term that is less common. When the *df* is low, the occurrence of the term in a document is more significant. By multiplying *tf* by the inverse document frequency (*idf*), the consequent value results from a weighting based on the relative frequency of the term in the corpus. The ENABLE system computes *tf-idf* using log weighting of the *tf* count and log inverse frequency weighting on the document counts.

$$tfidf_{t,d} = (1 + log(tf_{t,d}))log(\frac{N}{df_t}) \tag{3.1}$$

A Pearson correlation was done between the *tf-idf* values of the topics. For each correlation that was greater than 0.8, the topics were considered for combining. In the sample CS0 course there was a correlation between the *HTML*, *attribute*, *element*, and *tag* topic lists. Combining these was obvious once the correlation pointed them out. These are all parts of the HTML language. The other topics that were highly correlated were *JavaScript*, *functions*, and *textboxes*. Although *functions* is a topic that exists outside of JavaScript, in the context of this course, functions are only discussed or used in JavaScript. This correlation made the instructor aware that their broader view of the computer science curriculum was reflected in this separation of topics and would best be adapted to fit the content of this specific course. This provided the instructor a fresh perspective informed by feedback from ENABLE. This illustrates one of the many benefits of gaining another perspective when considering changes to current courses.

This process of identifying correlations between topics provided new insights into possible changes to the topic lists. These insights were not recognized when the original topic lists

Table 3.1: Topic Frequency Matrix. There is a column for each of the ten topics and a row for each learning item. Each value is a count of how many times that topic occurs in the learning item.

	Topics									
	1	2	3	4	5	6	7	8	9	10
FAQ	0	5	0	0	0	0	0	0	0	0
Video	0	0	0	0	0	0	0	0	0	0
Video	0	0	3	0	0	0	0	0	0	0
Note	7	57	7	0	0	0	2	0	0	0
CA1	6	77	0	0	0	0	3	0	0	0
HW1	2	1	0	0	0	0	0	0	0	0
CA2	3	56	0	0	0	0	1	0	0	0
HW2	4	7	0	0	0	0	0	0	0	0
Note	0	0	0	1	0	0	0	0	0	0
CA3	0	0	0	5	0	0	0	0	0	0
HW3	0	0	5	0	0	0	0	0	0	0
CA4	0	0	0	3	0	0	0	0	0	0
Ex1	0	0	0	3	0	0	0	0	0	0
CA5	0	0	0	6	0	0	0	0	0	0
HW4	0	0	0	2	0	0	0	0	0	0
CA6	0	29	9	0	0	0	0	0	0	0
CA7	0	4	2	0	0	0	0	0	0	0
HW5	1	11	1	0	0	0	2	0	0	0
HW6	0	0	0	0	0	0	0	0	0	0
Notes	0	0	0	0	1	0	0	0	0	0
Exam1	3	15	0	0	1	0	0	0	0	0
Notes	1	40	0	0	0	0	33	0	0	0
CA8	0	28	0	0	0	0	47	0	0	0
Note	0	0	0	0	0	1	0	12	0	0
Ex2jv	0	0	0	0	0	2	0	0	0	0
CA9	0	21	0	0	0	0	62	0	0	0
HW7	0	0	0	0	0	1	0	0	0	0
CA10	10	69	0	0	0	0	77	0	0	0
CA11	0	4	2	0	0	0	3	0	0	0
Ex2v	0	0	0	0	0	0	0	0	0	0
HW8	3	12	2	0	0	0	20	0	1	0
CA12	0	2	2	0	0	0	1	5	0	0
Ex3jv	0	0	0	0	0	0	0	1	0	0
Exam2	0	14	0	0	0	0	19	0	0	0
CA13	1	69	0	0	0	0	2	0	99	0
CA14	1	112	0	0	0	0	12	0	58	0
CA15	0	2	2	0	0	0	1	0	0	0
Ex3v	0	14	2	0	0	0	7	0	3	0

Continued on next page

Table 3.1 – *continued*

	Topics									
	1	2	3	4	5	6	7	8	9	10
HW9	1	17	2	0	0	0	11	0	5	0
CA16	0	215	0	0	0	0	16	0	145	0
CA17	0	145	0	0	0	0	4	0	109	0
CA18	0	0	2	0	0	0	0	0	1	0
Ex4v	0	0	0	0	0	0	0	0	1	0
HW10	0	6	2	0	0	0	4	0	7	0
CA19	0	0	1	0	0	0	0	0	0	0
CA20	0	0	0	0	0	0	0	0	0	0
Ex4jv	0	1	0	0	0	0	0	0	0	0
Teval	0	0	0	0	0	0	0	0	0	0
Exam3	0	28	0	0	0	0	0	0	24	0

were made. This process led to the reduction of topics from the original fifteen to the following ten:

1. <content>
2. <html, structure, attribute, attributes, element, elements, tag, tags>
3. <publishing, host, publish, published>
4. <careers, career, careers in cs,cs careers>
5. <darpa>
6. <history, cs history>
7. <css, style>
8. <hardware, system>
9. <javascript, script, functions, function, textboxes, textbox>
10. <using the web, use the web>.

Each item has its own frequency vector. Whenever there is a count greater than zero in the frequency vector, that item has an *occurs-in* relation with that topic. When combined, these vectors produce a frequency matrix that includes a row for each learning item with an entry for each topic (see Table 3.1). These vectors are also used when necessary to identify if two items are topically related.

3.2.3 Unit Relations

Units are a way of grouping learning items. This grouping may be based on a temporal factor, such as a group of items that all happen in a given week. It may be based on topics if all the items in a unit cover the same topic. It may be based on an external resource, as when all items included in the unit are about a chapter in a textbook. These units are identified by the module tool in Canvas. This tool allows instructors to group items. The variation in how they are grouped is not questioned. The assumption is that the instructor has a specific reason for grouping items into a single module. These modules in canvas provide information about the existing organization of the course. Each learning item has the module ID for any module it is included in. This is one of the details gathered when ENABLE is gathering data through the canvas API. This module ID becomes one of the properties of the item.

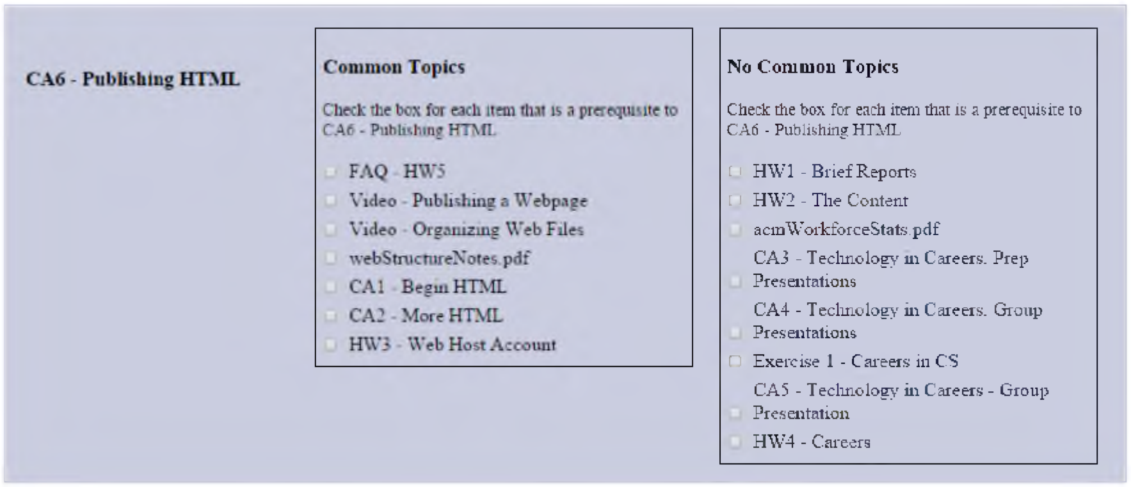
The relation between a unit and a learning item is the *includes* relation. A unit includes a learning item. This relation is easy to extract from the data and store. No manipulation is required.

3.2.4 Prerequisite Relations

The *prerequisite* relation expresses that there is value in doing one learning item before another. It may be that the following learning material is difficult to understand if the previous learning item has not been completed. It is stated that one learning item is a prerequisite to another learning item. This is a directed relation that is not reversible. It can be but is not necessarily transitive. The transitive rule would say that if item A is a prerequisite to item B and item B is a prerequisite to item C, then item A is a prerequisite to item C. Because each of these items could have different topics, it is possible to have the topical relation between item A and item B be based on a different topic than the one that is common between item B and item C. In this case it is possible that the prerequisite relation between item A and item B is related to a different topic than the prerequisite relation between item B and item C. In ENABLE a *prerequisite* relation is considered transitive only in the case when all the topics that are common to item A and item C are also included in item B.

The *prerequisite* relation is not identifiable from the data available in Canvas. There are two relations that can be used to inform the *prerequisite* relation but not determine it. Those relations are the *precedes* relation and the *occurs-in* relation. The *precedes* relation must exist for a *prerequisite* relation to exist. By definition the item that is a prerequisite to another item must precede it. But there are many cases when the *precedes* relation exists and there is no *prerequisite* relation. The *occurs-in* relation informs about the topics that occur in the learning item. When two items have common topic associations it is more likely there is a *prerequisite* relation between them. Although more likely, it is not always the case. Common topics are just an indicator of the greater possibility of the *prerequisite* relation.

Determining the *prerequisite* relations requires expert knowledge. ENABLE provides an interface that allows the instructor or another expert to identify the *prerequisite* relations (see Figure 3.5). This interface provides information from the *precedes* relation and the topical *occurs-in* relation. On the interface each learning item is listed. Next to the item are listed all the items that precede it. These preceding items are displayed in two categories: those with common topical relations and those with no common topical relations. The expert simply checks the box by each learning item that is a prerequisite to the current


CA6 - Publishing HTML

Common Topics

Check the box for each item that is a prerequisite to CA6 - Publishing HTML

- ☐ FAQ - HW5
- ☐ Video - Publishing a Webpage
- ☐ Video - Organizing Web Files
- ☐ webStructureNotes.pdf
- ☐ CA1 - Begin HTML
- ☐ CA2 - More HTML
- ☐ HW3 - Web Host Account

No Common Topics

Check the box for each item that is a prerequisite to CA6 - Publishing HTML

- ☐ HW1 - Brief Reports
- ☐ HW2 - The Content
- ☐ acmWorkforceStats.pdf
- ☐ CA3 - Technology in Careers. Prep Presentations
- ☐ CA4 - Technology in Careers. Group Presentations
- ☐ Exercise 1 - Careers in CS
- ☐ CA5 - Technology in Careers - Group Presentation
- ☐ HW4 - Careers

Figure 3.5. Interface to Add Prerequisites.

item. This information is stored as a property of the learning item object. It is a list of all the items that are prerequisite to it. This *prerequisite* relation is used when creating the course map and when building the probability models.

3.3 Summary

There is much data available in Canvas that can be accessed through the API. Other information is determined through text analysis and by gathering expert knowledge. This gathering and analysis of data is used to identify the relations and structure of an existing course. This phase of the system establishes the basis of the future phases by providing them with the data and information needed for creating and transforming course maps, designing and running student agents, and generating probability models.

The evaluation criteria of the content analysis phase include the representation of the analysis in a graph. This is accomplished in Chapter 4. A discussion of how the evaluation criteria are met is found in the summary of that chapter.

CHAPTER 4

THE COURSE MAP

ENABLE has gathered, extracted, analyzed, and produced a representation of the existing course. This representation consists of learning items and the relations between them. This information can be represented as a directed acyclic graph (DAG). In the learning management support system (LMS), learning items are presented in a linear, textual way. Using a graph representation increases the opportunity to express the nonlinear relations between the items. Presenting it in this visually rich method provides a way to express more information.

A *course map* is a graph, $\mathcal{M} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of learning item, topic and unit nodes, and $\mathcal{E}$ is the set of *precedes*, *topically precedes*, *prerequisite*, *occurs in* and *includes* edges (relations). Then the *class map* is $\mathcal{C} = (\mathcal{O}, \mathcal{S})$, where $\mathcal{O} \subset \mathcal{N}$ is a set of nodes that includes the learning item nodes and $\mathcal{S} \subset \mathcal{E}$ is a set of edges that includes all of the *precedes*, *topically precedes*, *prerequisite*, *occurs in* or *includes* edges.

The course mapping capability was developed and used on the three sample courses previously mentioned. To provide continuity, the figures included in this chapter are of only a single sample course. The course used in the figures is a CS0 course, The Foundations of Computer Science.

4.1 Choosing the Structure

The purpose of the ENABLE system is to support educators when making changes to existing course structure. To support effective change it is important to identify the current course structure. Current organization of learning items is often limited by the tools used to deliver them. Today many educators use some form of LMS to deliver course materials to students, communicate electronically, receive work from students, and publish scores. These management systems present learning items in a textual, linear way. This linear representation is not representative of how learning items are actually related. The first phase of ENABLE development extracted the details of learning items and identified relations between them. Multiple relations were used to more closely represent how the learning items are actually related.

To clearly represent current course organization it is necessary to select a data structure that is well suited to the complexity of the varied relations between learning items in a course. An array or a list could be used to illustrate the representation of the course structure as it is delivered in the learning management system. This representation has one learning item directly following another learning item. This limits the connections between learning items and does not allow flexibility in ordering. It is easy to identify cases when this representation is too limited to express how the learning items are actually related. For example, there may be several learning items that are designed to prepare a student to complete a particular homework assignment such as a lecture, a class activity, a video, and a reading assignment. Represented as a list it would look something like that shown in Figure 4.1(a). Representing it this way indicates *prerequisite* relations between the other learning items, when, in fact, these *prerequisite* relations do not exist. The lecture, video, activity, and reading can be done independently of each other, and it is not necessary to complete them in any particular order. A nonlinear structure would better represent this relation, as shown in Figure 4.1(b).

Another consideration is the tree data structure. As shown in Figure 4.1(b), the four prerequisites are parents of the homework node. This breaks a basic rule of trees – a node has exactly one parent, except the root node, which has none. We could invert the relation and list the prerequisites as child nodes of the homework node, as shown in Figure 4.2. This solves the problem of multiple parents but introduces another problem, nodes without parents, as there may be items that are not a prerequisite to anything. For example, an exam would have several learning item prerequisites. Although another exam may follow this exam, the two may not be related, in which case, the first exam is not a prerequisite for the second exam. This leaves the first exam without parents effectively creating a nonroot node without parents, as shown in Figure 4.3.

Another data structure to consider is a directed graph. Directed graphs allow multiple incoming or outgoing edges. A standard component of moving through a course is the common practice of moving through each learning item only one time. Although this is not the only approach, it is the approach widely used in practice and is used in this project. To enforce this, single coverage of the material cycles is not allowed. These characteristics can all be represented with a directed acyclic graph (DAG). A DAG allows us to represent a learning item with multiple prerequisites without a restriction on the direction of those

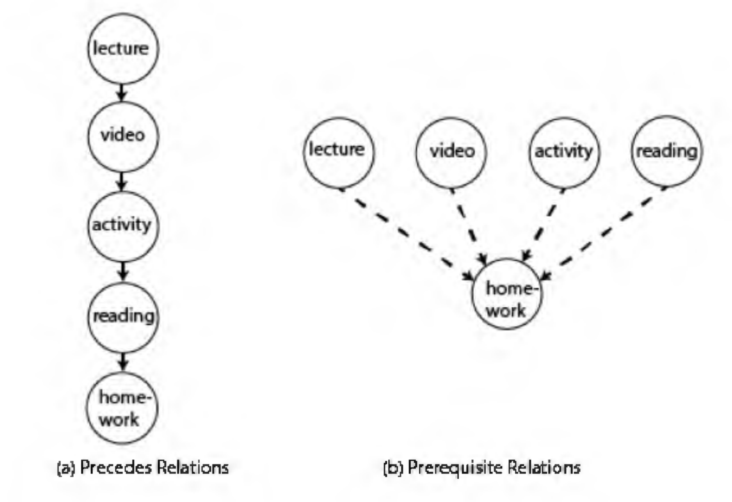


Figure 4.1. *Precedes* vs. Prerequisite Structure.

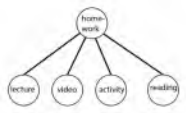


Figure 4.2. A Tree Representation of Learning Item Relations.

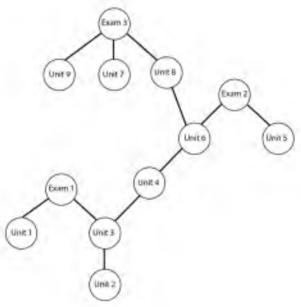


Figure 4.3. Inverted Representation of Learning Items Extended to Demonstrate Resulting Structure Does Not Produce a Tree.

relations, allowing the representation shown in Figure 4.1(b), and it also has no concept of root or parent nodes, so there are no limitations based on hierarchy. With this structure we can also represent the exam that has multiple prerequisites and which is not a prerequisite to any other learning item, as shown in Figure 4.4. Without the narrow options of a list or the hierarchal rules of a tree, the DAG is an ideal data structure to represent the existing course organization and provide flexibility to represent many alternative course structures.

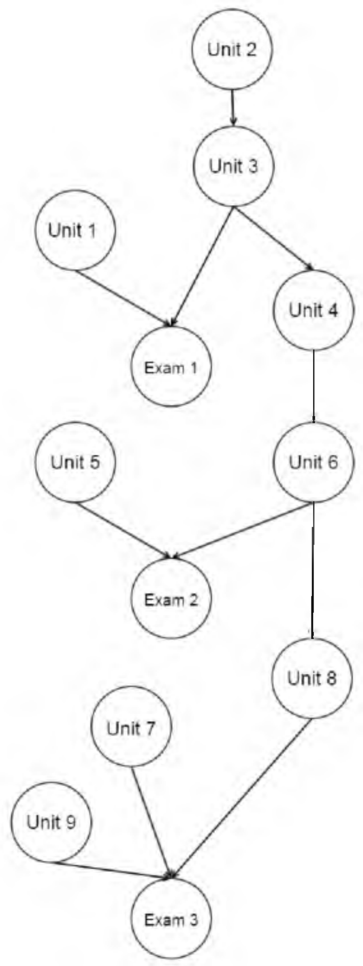


Figure 4.4. DAG Representation of Learning Items.

4.2 Building the Map

4.2.1 Adding *Precedes*

We start with the relations that are predominantly expressed in the existing course through the LMS, the temporal relations. The *precedes* relation indicates that one learning item *precedes* another in time. Applying the transitive nature of this relation, a graph can be produced that expresses all these relations. In this graph the nodes represent the learning items and the edges represent the *precedes* relation. These are directed edges that go from a learning item that occurred earlier in time to a learning item that occurred later in time. This graph has N nodes and $N-1$ edges. Every node except the first and the last one have one in edge and one out edge.

This simple graph is linear in nature. It can be laid out in either a horizontal or vertical direction with each node equidistant from the next. Displaying the graph in this manner is very reflective of how the learning items are presented in the LMS. This expresses the very minimal amount of information about how the learning items are connected, showing only which learning item comes before another learning item. The graph can be enhanced by adding relative temporal information. The learning items are related in time but the time between each learning item varies. This can be expressed visually in the graph by separating the nodes by a distance that relates to the difference in time between each learning item. This quantifies the temporal relation. Using the date information of the learning item, the X value can be computed to reflect this relative distance.

$$x = (d_i - d_s)d_f \tag{4.1}$$

where x is the x-coordinate on a two-dimensional Cartesian plane, d_i is the item date, d_s is the course start date, and d_f is a factor determined to allocate the nodes across the map in a way that separates them within the horizontal space available.

This adds a visual component that not only expresses the order of the learning items but also how temporally distant they are one from another. This information exists in the linear list available in the LMS as the item dates are listed. But displaying these relative dates in a visual way makes it easy to identify how the items are related temporally.

Additional information can be added to the graph by using color to represent the different types of learning items. There are three distinct learning item types: assignments, exams, resources. Each of these types is given a distinct color. The background of each item node is made up of this color. Each node is labeled with the short label identified during the analysis phase.

4.2.2 Adding Units

The *includes* relation expresses that a unit includes a learning item. Units are a way to group learning items. The separation into units can be based on a variety of factors. For example, it may be based on temporal details such as learning items to be covered in a given week. The separation into units may be topically based, as in the case when the unit includes learning items that cover a specific topic. It may also be based on some external element, for example, when a unit contains the learning items related to a chapter of the textbook.

This *includes* relation can be represented as a bipartite graph with edges connecting the set of units to the set of learning items. This adds N additional edges in the case where each learning item is included in exactly one unit. This bipartite graph is directed from the set of units to the set of learning items. This maintains the status of the graph as a

DAG. This approach was considered but never implemented in the ENABLE system. Each time the number of edges is increased, the complexity of the graph is increased and the visual representation becomes less clear. It was determined that the unit relations were less significant than the topical relations and therefore edgeless approaches to representing the unit relations are implemented.

An edgeless way to express the *includes* relations in the graph is to use the unit number when computing the y value of the learning item node. This approach does not add any additional nodes or edges for the *includes* relation. This can be done with the following formula.

$$y = n_u d_y \tag{4.2}$$

where y is the y-coordinate on a two-dimensional Cartesian plane, n_u is the unit number of the learning item represented by the node, and d_y is a factor determined to allocate the nodes across the map in a way that separates them within the vertical space available.

Learning items within a single unit often come close to each other temporally. This produces a series of learning items with the same y value. To increase the visual diversity, a random number is added to the formula. This produces variation in the vertical location of each node and creates more movement in the graph. The variation that is added by this random value allows nodes in the same unit to have slightly different y values. This variation is sufficiently smaller than the multiplicative factor, d_y, that it keeps the y value of all the nodes in a specific unit within an identifiable range. Formula 4.3 includes noise.

$$y = n_u d_y + \in \tag{4.3}$$

where $\in \sim N(0, 12)$.

The edges of the *precedes* relations are included in this graph as well as the horizontal spacing used to express the relative temporal relations (see Figure 4.5).

ENABLE provides one additional way to represent the *includes* relation. In this approach both the x and y coordinates are used to locate item nodes included in a specific unit in a circular pattern. The pattern for each unit takes up a specific section of the available space. This gives distinct unit patterns, making it easy to identify the learning items within a given unit. Using this approach removes the relative temporal spacing added previously. Since both the y and the x values are used to express the *includes* relation, the location of the node on the graph is limited to this one relation. The edges expressing the *precedes* relation are still included in the graph (see Figure 4.6).

Identifying a more effective way to represent the unit nodes in a meaningful visual way is a matter of research in the human computer field and is left for future work.

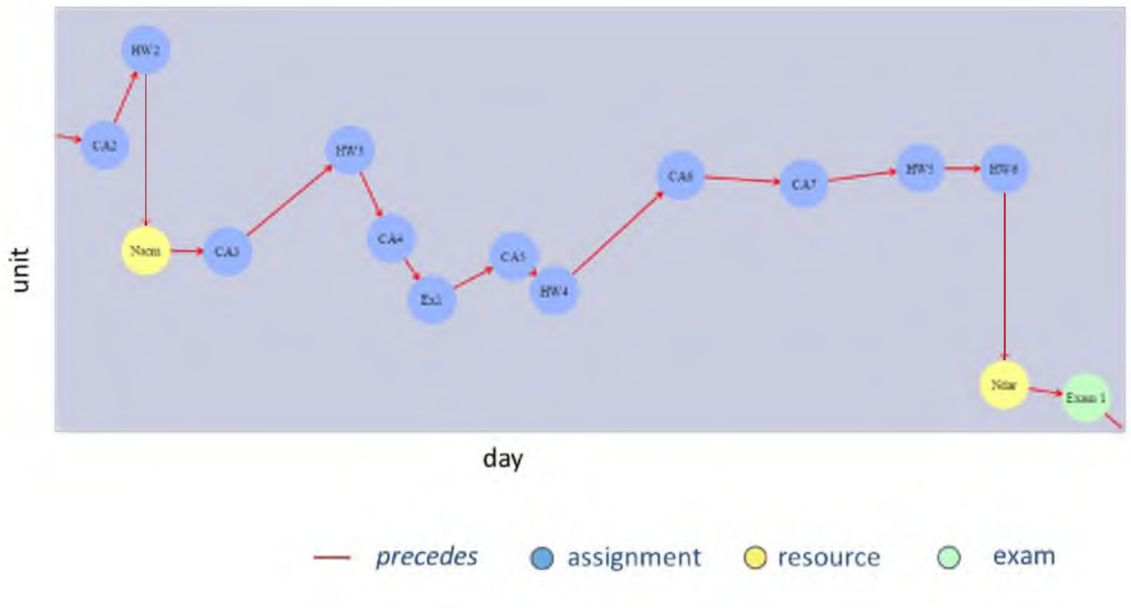


Figure 4.5. Section of Course Map Showing *Precedes* Relations.

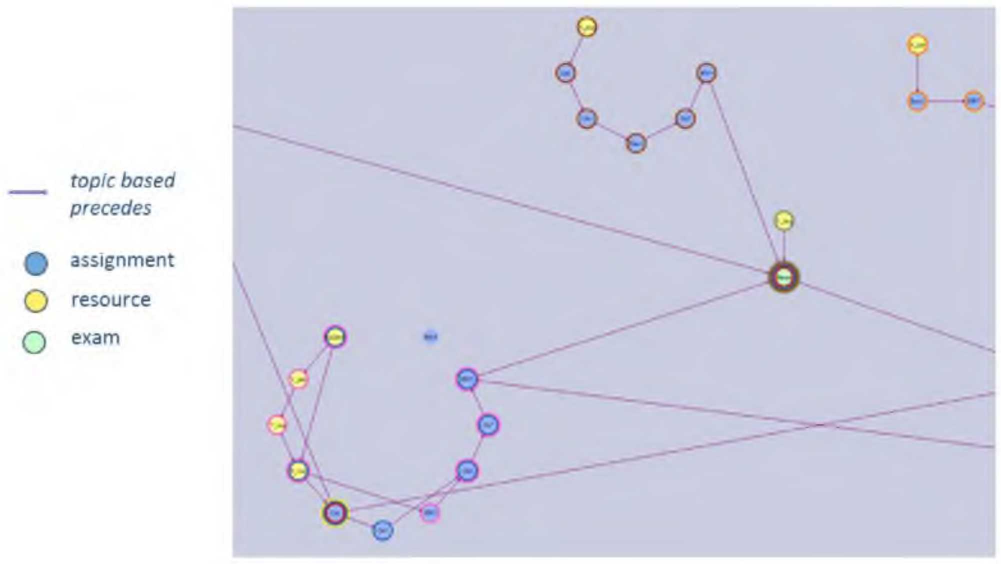


Figure 4.6. Section of Course Map Showing Grouped Includes Relations.

4.2.3 Adding Topics

Using text analysis and expert knowledge, topical relations of the existing course are established. The topical relation is the *occurs-in* relation. It expresses that a topic occurs in a learning item. These relations can also be included in the graph. ENABLE incorporates the *occurs-in* relations in two distinct ways. The first approach represents the topics as nodes in the graph. These nodes have a unique color to distinguish them from the learning item nodes. Then a bipartite graph is created with directed edges from the set of topic nodes to the set of learning item nodes. Because of the directed nature of these additional relations, the graph remains a DAG.

As it is possible for each learning item to have an *occurs-in* relation with multiple topics, this adds more than N edges, where N is the number of learning items in the course. To reduce the number of edge crossings introduced by the addition of this large number of edges, the topic nodes are located above and below the learning item nodes. This allows the edges representing the *precedes* relations to be clearly visible. It also maintains the relative temporal spacing between the learning items (see Figure 4.7).

It is possible to represent the topical relations without adding edges. This is done by adding colored rings to the learning item nodes. Each topic is assigned a specific color. A legend is added to the course map that shows which color goes with which topic. Whenever there is an *occurs-in* relation between a topic and a learning item, a ring of the specific color associated with the topic is added to the learning item. Multiple rings can be added to a single learning item node. Adding multiple rings increases the size of the learning item node. This visually expresses the quantity of topics occurring in a single learning item. These colored rings can be included with any of the previous versions of the course map (see Figure 4.8).

Occasionally there are learning items that have no *occurs-in* relations. In this sample CS course there were two. One was a sign up to determine team assignments and the other was the teacher evaluation assignment. When the *occurs-in* relations are added to the graph, these learning items without topics become very apparent. In either version of the course map that includes topics, it is easy to identify learning item nodes that are not associated with topics. When the *occurs-in* relation is expressed by edges, these learning items have no in-edges coming from topic nodes. In the colored ring version, these learning item nodes have no colored ring, making them visually distinct.

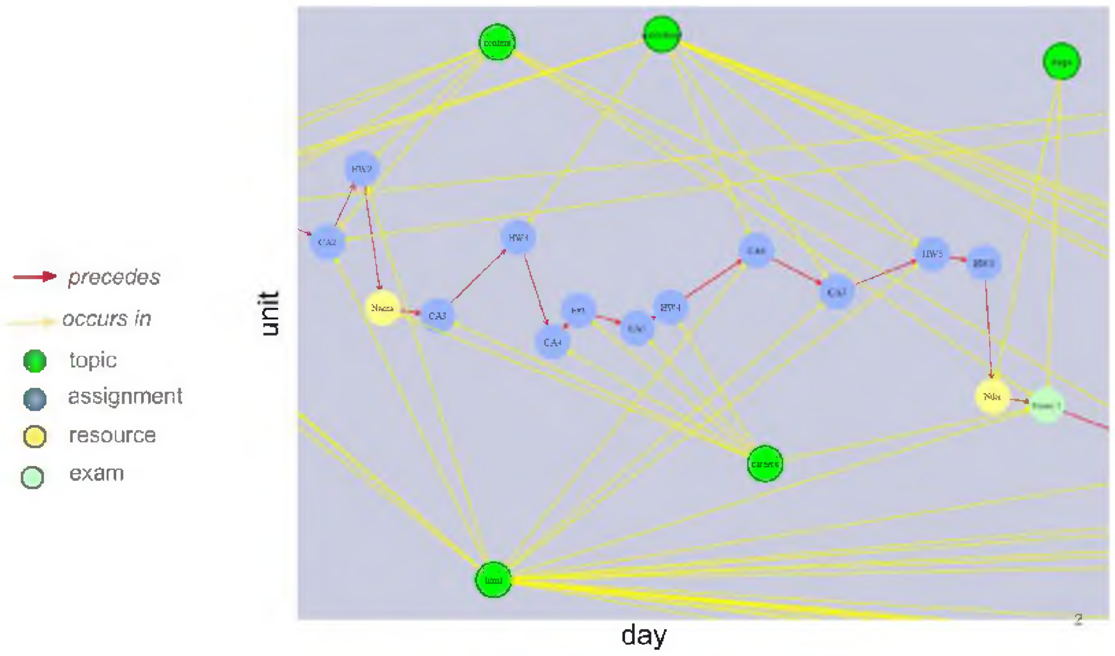


Figure 4.7. Section of Course Map Showing Topical Relations in Bipartite Graph.

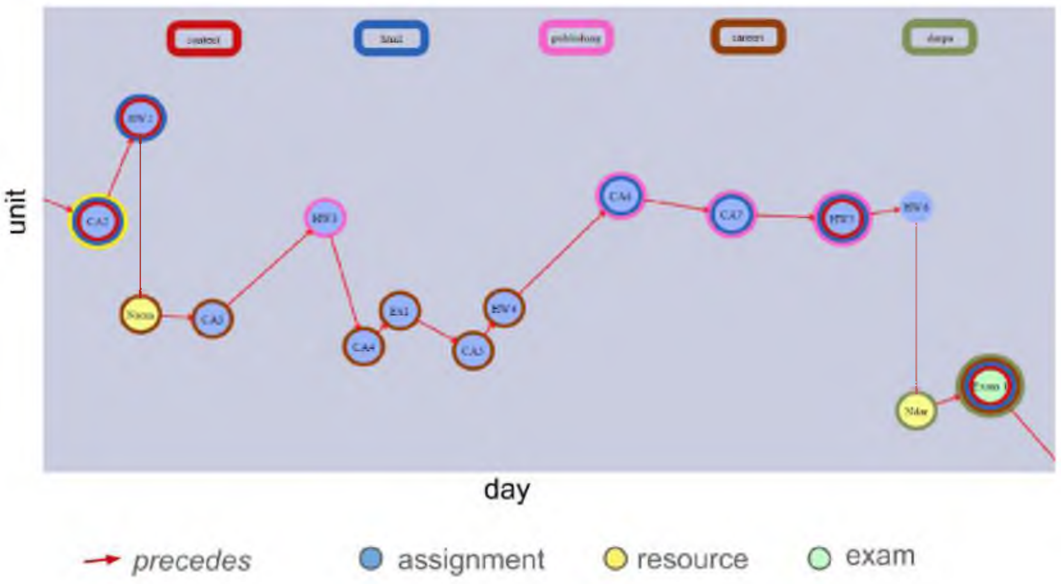


Figure 4.8. Section of Course Map Showing Occurs In Relations Drawn as Colored Rings.

4.2.4 Comparing Graph Representation to Presentation in LMS

Once the course graph is created, the presentation of the learning items in the graph can be compared to the presentation of the learning items in the LMS. A comparison was done to identify if this graphical representation was a correct representation of the existing course. The following three things were compared.

- Learning items are of the correct type.
- Temporal ordering of learning items is the same.
- Topical relations in the course map correctly represent the topical relations identified in the LMS.

ENABLE assigns each learning item one of three types: assignment, exam, and resource. In each of the three sample courses, all of the learning items were correctly categorized.

Temporal ordering is limited to learning items that have a due date. For these dated items the order was the same in both ENABLE and the LMS. There are differences in the ordering of the ungraded learning items within units in Canvas and the order in which they are in the graph. Since these items do not have due dates, there is no temporal information to compare.

Comparing topical relations presented some difficulty since the LMS does not represent topic relations. To do this comparison, the title of each learning item as it is presented in the LMS was scanned for words from the topic list. A count was taken of how many topics were represented in these titles. For the three sample courses the counts were 31, 57, and 70. All of these topical associations were present in the graphical representation in ENABLE. In addition, ENABLE included many topical relations that were not apparent in the LMS. For these same sample courses, the graph had 83, 136, and 116 topical relations presented, respectively. The graphical representation was able to display many more topical relations than the LMS.

There are two learning items that are in the LMS that are not represented in the graph, one in the first sample course and one in the second. These learning items were .zip files. Currently ENABLE does not have the capacity to analyze .zip files, so they are not included in the graph.

Each of the comparison features was rated from 1 to 5, with 5 being the highest quality. All three items compared for all three sample courses were rated at a 5. These were correctly represented in the graph.

4.3 Transforming the Graph

Now an initial course graph has been developed that represents the existing course. This basic graph can be used as a basis from which to transform the course map in a variety of ways. The phase provides insight into alternative organizations. This can be done using graph transformations. Graph grammars and graph transformation systems provide a means for doing this. One of the research application areas in graph transformation is model transformations. This area of model transformation is applied in the field of software engineering. The models used in software engineering have many similarities to the graphical representation of learning materials in ENABLE. The similarities include typed nodes, node attributes, and edges that represent different types of relations.

For graph transformation, initially ENABLE used AGG, a development environment for attributed graph transformation [20]. It is based on an algebraic approach to graph transformations [23]. The implementation of this approach closely follows the formal theoretical foundation of algebraic graph transformation and so provides validation support and sound behavior concerning graph transformation. However, this tool did not provide support for the extended analysis needed for some of the transformations used. For example, it could perform a localized version of the topic-based *precedes* elimination transformation, but was unable to consider the topical relations that needed to be considered from earlier *precedes* relations. Alternatively, the graph transformations were encoded directly in ENABLE.

We now consider some desirable transforms and their meanings. We define a restraint as an unnecessary constraint between two items. Thus, restraints are removed in order to open up more possibilities for the relations between learning items. When removing restraints it is important to maintain both the semantics of the course representation and the integrity of the DAG structure.

4.3.1 T1: Topic-based *Precedes* Elimination Rule

The *precedes* relation enforces the current organization by requiring that the learning items occur in the order they came in the original course. This is an unnecessary limitation, a restraint. To increase the opportunity for new organizations, this restraint needs to be removed when possible. Many of the learning items that are connected by the *precedes* relations are not related topically. They do not have any topics in common. This is an indication that the order of these nodes could be manipulated without reducing the effectiveness of the ordering.

When learning items are connected with the *precedes* relation and they have common topics, there is the possibility that the *precedes* relation is required for the topic to be covered effectively. A portion of the topic covered in an earlier learning item may be necessary for understanding the coverage of that topic in a later item. This topical connection indicates that the *precedes* relation may have significance and should not be removed. On the other hand, when two learning items are related by *precedes* relations and have no topics in common, the *precedes* relation can be removed without the same disruptive consequences for the learner.

The topic-based *precedes* elimination rule uses the distinction between topically related *precedes* relations and *precedes* relations with no common topics to remove some of the restraints. If A *precedes* B and B *precedes* C and there are no common topics that *occurs-in* both A and B, the *precedes* relation from A to B can be removed. When removing this relation it is important to keep the relation that A *precedes* C and B *precedes* C. Note, however, that the net number of *precedes* relations is reduced by 1, as there was an implied $P(A, C)$ before the application of *T1* (see Figure 4.9). The notation $P(x, y)$ means x *precedes* y.

More formally, this can be stated as follows:

$$\text{if } P(A, B) \wedge P(B, C) \wedge \nexists T \ni (OI(T, A) \wedge OI(T, B)) \tag{4.4}$$

$$\text{then remove } P(A, B) \text{ and add } P(A, C) \tag{4.5}$$

where P means *precedes*, OI means *occurs-in*, A, B, and C are learning items, and T is a topic. We call this the *Topic-based precedes Breaking* rule (*T1*). The notation $OI(x, y)$ means x *occurs-in* y; Figure 4.10 shows a graphical representation of this transform.

When applying this transformation, it is not sufficient to look locally at only three nodes. It is possible that a preceding node has topical relations with future nodes. Removing a local *precedes* relation between two nodes that have no topics in common may inadvertently remove a topically related *precedes* connection that was expressed in a transitive way.

When looking at the nodes C, D, and E in Figure 4.11(a), it is not sufficient to only look at the topic connections between these three nodes. Figure 4.11(b) shows three negative application conditions that must be considered to identify which *precedes* relations can be removed and which cannot. Before removing the relation between C and D, the topical relations between A and D, B and D, and C and D must all be considered. Figure 4.11(c) shows the transformation of the graph when there is no topical relation between D and any of A, B, or C. Figure 4.12 shows two of the transformations that allow the *precedes* relation between C and D to be removed when there are topical relations between D and previous nodes. Each of these transformations allows the removal of the *precedes* relation between C and D while maintaining the topically related *precedes* of the previous nodes.

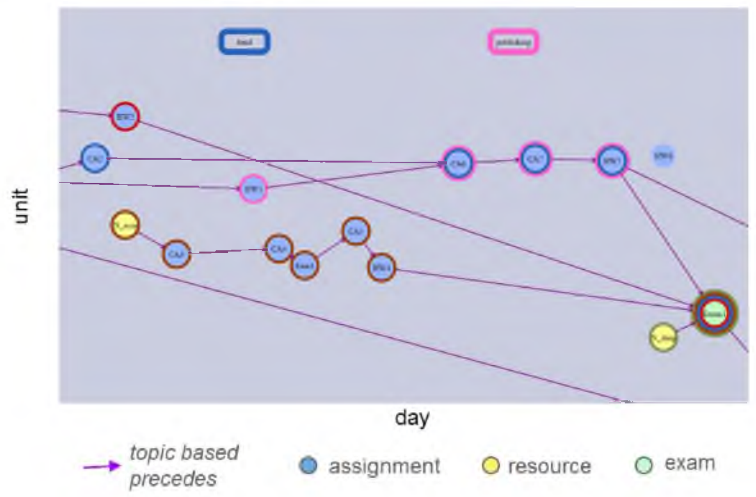


Figure 4.9. T1: Topic-based *Precedes* Elimination Rule.

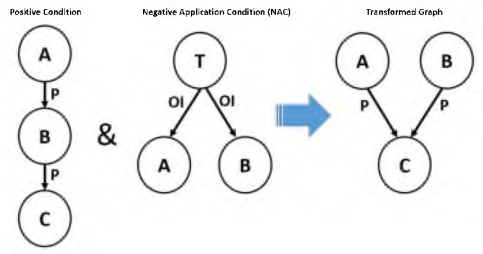


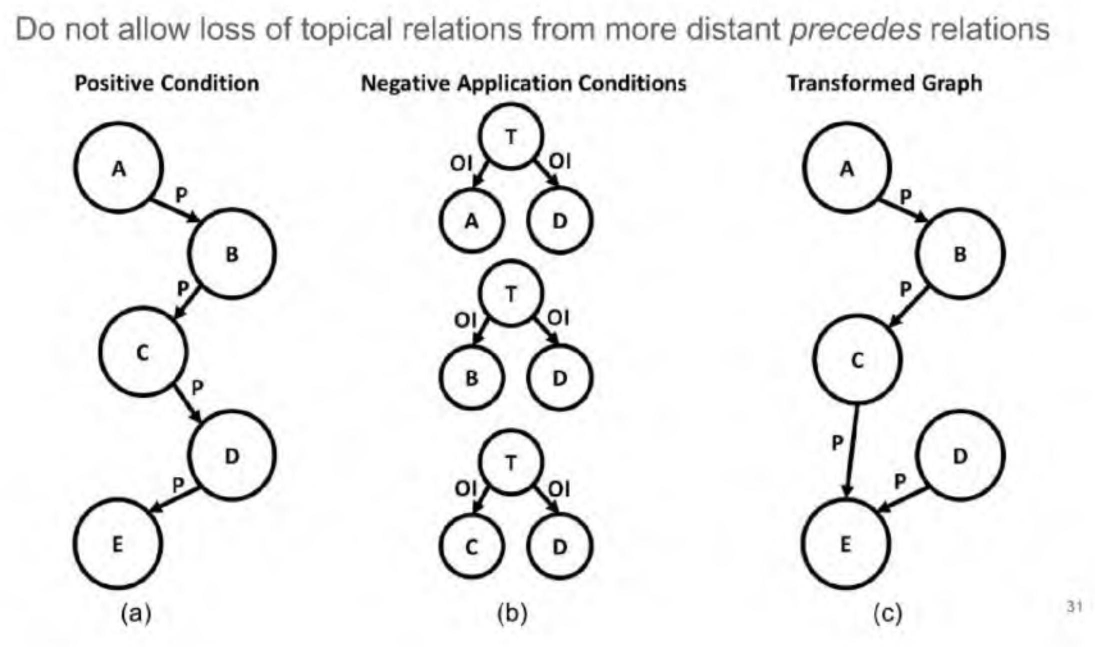
Figure 4.10. Graph Transformation to Eliminate Unnecessary *Precedes*.

This demonstrates in detail how a simple *precedes* relation can be eliminated when there are no topics in common between adjacent learning items. Although this demonstration only includes the case of five nodes, it is important that the process is extended to include any number of learning items.

The application of this rule will reduce the number of *precedes* relations in the complete *precedes* graph. It is likely to increase the total number of edges in the graph that uses transitive reduction to simplify the complete *precedes* graph. Before the application of this rule, each node in the simplified graph had a single incoming edge and a single outgoing edge. Now a learning item that occurs in multiple topics may have an outgoing edge for each of those topics. This same node may have multiple incoming edges as well.

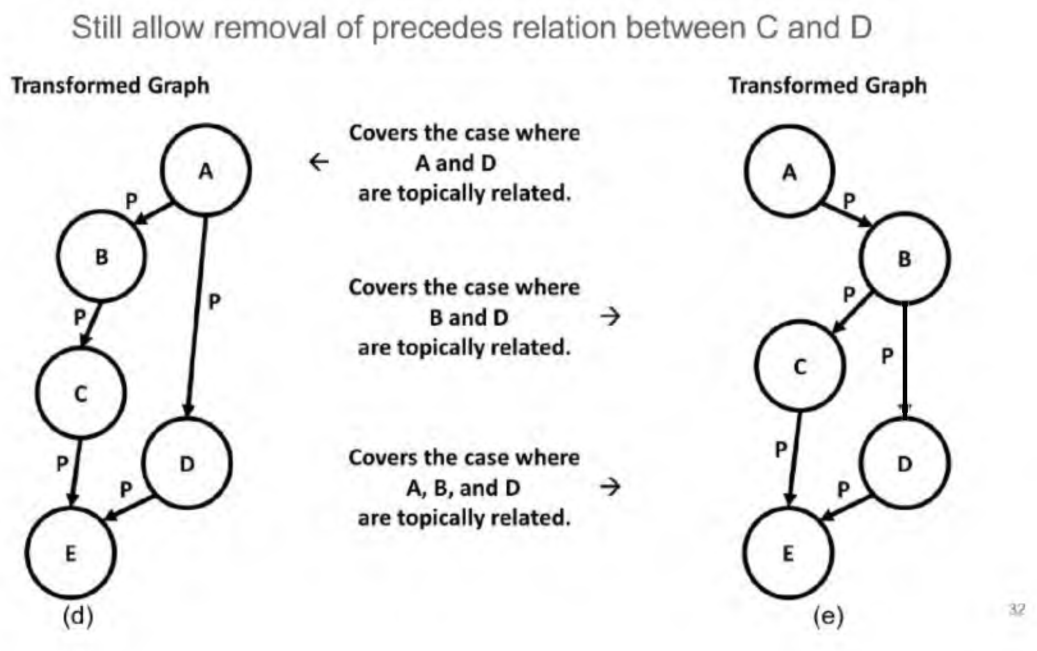
Figure 4.9 shows a section of the course map. This version of the course map incorporates the topic-based precede relations. It also expresses the *occurs-in* relation using colored rings, and the *includes* relation.

After the topic-based *precedes* elimination rule was first applied to the sample CS0



Do not allow loss of topical relations from more distant *precedes* relations
Positive Condition
Negative Application Conditions
Transformed Graph
A
B
C
D
E
P
T
OI
(a)
(b)
(c)
31

Figure 4.11. When Transforming the *Precedes* Relations Alternatives Exist.



Still allow removal of precedes relation between C and D
Transformed Graph
Transformed Graph
Covers the case where A and D are topically related.
Covers the case where B and D are topically related.
Covers the case where A, B, and D are topically related.
A
B
C
D
E
P
(d)
(e)
32

Figure 4.12. When Transforming the *Precedes* Relations Care Must be Taken to Consider Maintain the Transitive Relations From Previous Nodes.

course, the instructor reviewed the resulting graph. Several unexpected findings were encountered.

4.3.1.1 Learning Items with No Topical Relations

There were five learning items that had no topic relations. Upon examination, two of the items were truly not related to any topic, an assignment in which students were to submit which team they wanted to be on, and the teacher evaluation. Both items were left unchanged, with no topical relations.

The other three clearly were related to a specific topic, but none of the terms in the topic list were found in the description. This could be remedied by using ENABLE's file upload tool that provides a way to add additional text to the description of a learning item, or by using the user interface to edit the topic relations. This also caused the instructor to consider the value of more frequently using the topic terms explicitly in the textual presentation of the learning items.

4.3.1.2 Learning Items with Meaningless Topical Relations

There were six topical relations that connected learning items to topics mistakenly. In five of these learning items, the topic words occurred but were being used in a more general way. For example, one of the topics is **content**. This is specifically related to selecting content when creating a web site. However, the word **content** was used in its more general way in three of the learning items. In the other case the instructions included a restriction to not use JavaScript, which was a future topic. These relations were removed through the graphical interface.

4.3.2 T2 Topic-based Exam Splitting Rule

Exams are another limitation to course restructuring. Exams can cover multiple topics. This connects many learning items with varied topics to a single exam. Sometimes these connections are arbitrary, often occurring because of the temporal relation between the learning items. Commonly, an exam is written to assess the material that has been covered over a specific period of time, such as since the last exam or since the beginning of the semester, even though many of these learning items may not be related topically. When exams tie learning items together that are not related in any topical way, restraints are produced. These restraints are imposed by the temporal ordering of the existing course.

This time-based connection is not required for assessment. Therefore it is possible to divide the material assessed in an exam by topic. Separating the temporal grouping inherent in exams provides additional possibilities for change. This allows learning items to only be connected to the portion of the exam that covers the topics in that specific learning item. This disconnects learning items that are not related topically, allowing for more flexibility in organizing the learning items in different orders.

The topic-based exam splitting rule allows for the removal of these restraints. This rule cannot be applied until after the topic-based *precedes* elimination rule (T1) has been applied. Before the application of T1, there is only one incoming edge to an exam. There is nothing to split. After the application of the T1 rule, there may be multiple incoming edges to an exam. It is at this point that the learning items connected to the exam by these incoming edges can be considered as associated with only a portion of the exam.

If A *precedes* Exam 1 and B *precedes* Exam 1 and there are no common topics that occur in both A and B, then Exam 1 can be split into two exams, Exam 1A and Exam 1B, such that A *precedes* Exam 1A and B *precedes* Exam 1B, and A and B are independent of Exam 1B and Exam 1A, respectively (see Figure 4.13). This transform must be applied after T1.

After the application of this exam splitting rule, the course map shows the separateness of the items coming into each split portion of the exam. This increases the amount of movement available in the course map (see Figure 4.14).

The first time the topic-based exam splitting rule was applied, there were fewer exam splits than expected. Upon closer review it was discovered that the exam asked questions about a topic without using a topic word explicitly. This seemed pedagogically sound. For example, one question about computer science history was “Why was the invention of the integrated circuit important?” Although this question does not use the term **history**, it is clearly assessing the student’s familiarity with the computer science history covered in the course. These missing meaningful relations can be included by adding text that includes the missing topic words to the description using ENABLE’s file upload tool or by editing the topical relations using the user interface. This adds the text to the ENABLE system without altering the exam itself.

There was one case where the review of the exam exposed the possibility of adding a word to the topic list. The word **occupation** was used in the exam that covered careers in computer science. This word was also used in other learning items about the topic. It was determined that adding this word to the topic list would add clarity. Adding the term to the topic list resolved this missing relation.

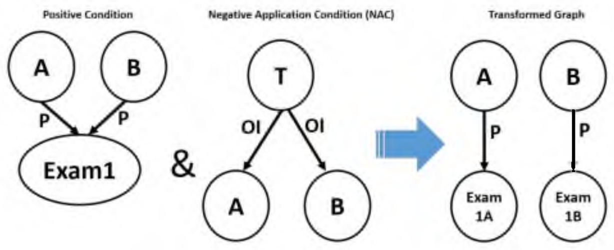


Figure 4.13. T2 Topic-based Exam Splitting Rule.

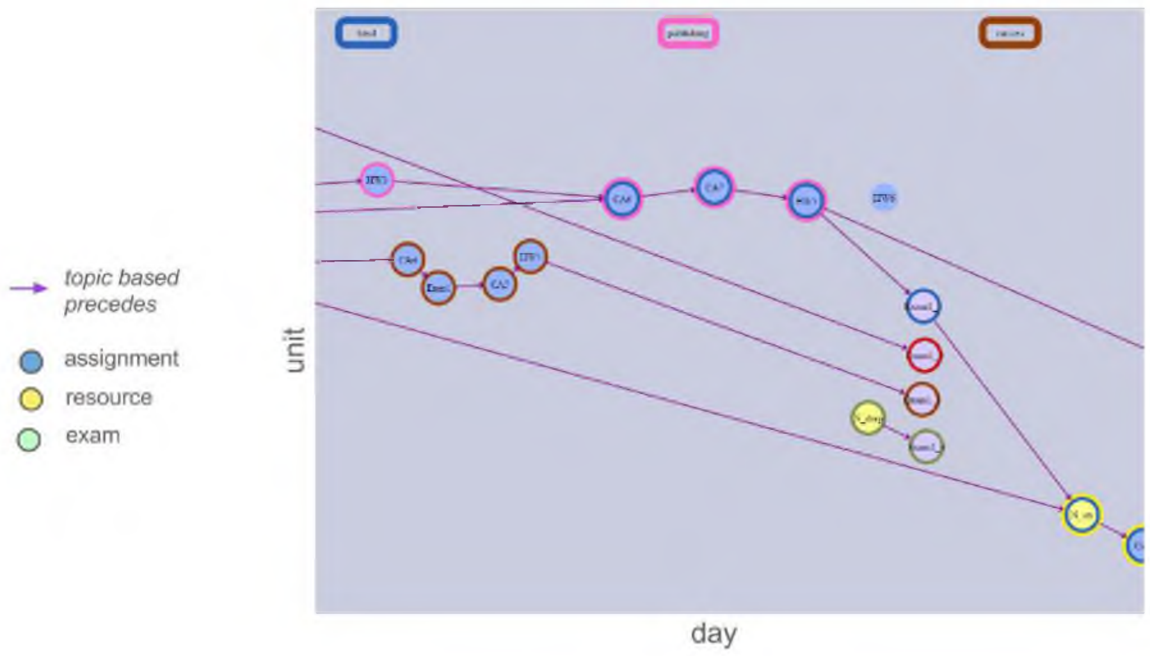


Figure 4.14. Section of Course Map Showing Exam Split Transformation.

4.3.3 T3: Replacing *precedes* Relations with *prerequisite* Relations

The *precedes* relation is the predominant relation expressed in the LMS. It is a weak relation tying learning items together only by the order in which they came in the original course.

In some cases this order is critical. And it is important to do one learning item before the other, as the second item is dependent on knowledge acquired from the previous item. In other cases the items are unrelated and the temporal relation is a restraint. A way to capture the difference between a critical *precedes* relation and an unnecessary *precedes* relation is by introducing the *prerequisite* relation. The *prerequisite* relation expresses a *precedes* relation that has specific benefit. This *prerequisite* relation identifies that there is educational value in doing one learning item before another.

The *precedes* relation has one learning item directly following another learning item. This limits the connections between learning items and does not allow flexibility in ordering. It is easy to identify cases when this representation is too limited to express how the learning items are actually related. As described earlier, there may be several learning items that are designed to prepare a student to complete a particular homework assignment, such as a lecture, a class activity, a video, and a reading assignment. Using *precedes* relations, a graphical representation would look similar to that shown in Figure 4.1. Representing it this way indicates a specific ordering between the learning items when, in fact, this ordering is not required. The lecture, video, activity, and reading can be done independently of each other, and it is not necessary to complete them in any particular order. A prerequisite order would better represent the meaningful relations. Eliminating *precedes* relations in favor of *prerequisite* relations provides a more accurate representation of the interconnectedness of the course material.

The topically related *precedes* relation removes *precedes* relations whenever there is no topical relation in common. This is a good first attempt at establishing prerequisites. *Prerequisite* relations are not explicit in the data gathered from canvas, and require expert knowledge to be accurately identified. Using the topically related *precedes* relations, an interface for the instructor or other expert is created so that this expert knowledge can be gathered. This interface uses the topically related *precedes* relations to present the learning items in a way to reduce the cognitive load for the expert. For each learning item the interface displays all learning items that precede it. These preceding items are separated into two categories, those with topics in common and those with no common topics. Each preceding learning item has an associated checkbox. The expert simply checks the box by those items that are prerequisites. They can choose items from either category.

Once the prerequisites have been identified, the course map is redrawn using these *prerequisite* relations to connect the learning items. Topically based *occurs-in* relations and unit-based *includes* relations are both included in this updated course map. No *precedes* relations are included. They have been replaced by the *prerequisite* relations (see Figure 4.15).

4.3.4 T4: Instructor Directed Transformation

Now that the temporal restraints of a synchronous course have been removed, the opportunities for restructuring the course are greater. The ENABLE system can now display a graph based on the *occurs-in*, *includes*, and *prerequisite* relations without the

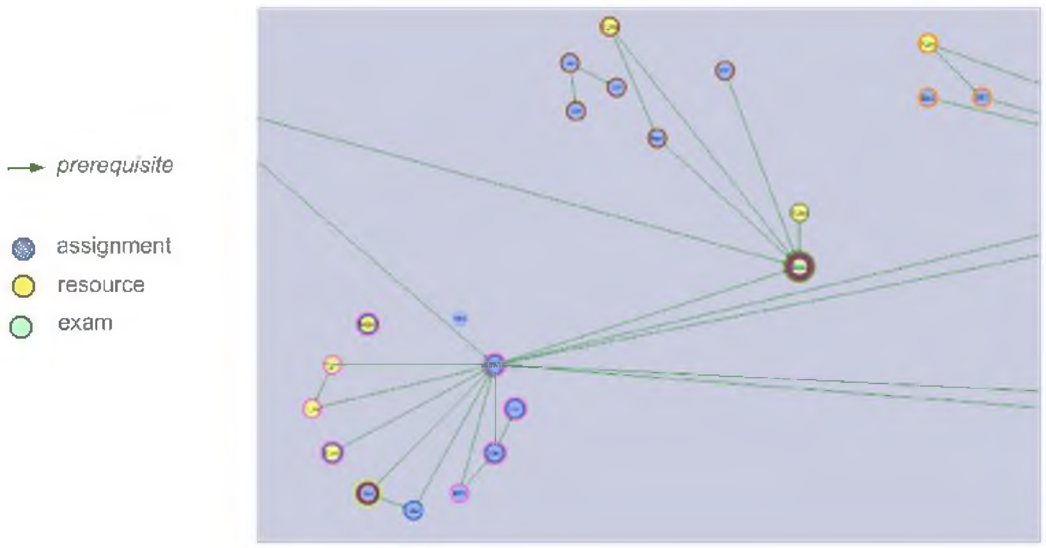


Figure 4.15. Section of Course Map Showing Prerequisite Relations.

time-based restraints. The final transformation is the instructor-directed transformation. The instructor can transform the course in two distinct ways: editing the course information and manipulating the course map.

ENABLE provides an interface that allows the instructor to edit many of the details about the course. They can add additional learning items. Additional content can be added to existing learning items. The short names used for labeling can be edited. Topical and *prerequisite* relations can be added or removed. Some details are not available in Canvas. This ability to add and edit details helps get the existing course represented more correctly. It also allows the instructor to make changes to the representation of the existing course. Once the changes have been entered, the system incorporates the new data to create updated course maps.

The other transformation that an instructor can do is manipulate the actual course map. The course map display is designed in such a way that the nodes can be moved about. As a node is moved, any connecting edges move with it. Keeping these connections intact during moving maintains the integrity of the graph structure. This manual manipulation of the course map provides a way to see the course with many different layouts. The learning items can be organized by topic, by exam, by learning item type, by prerequisite chains, etc. This provides the instructor the opportunity for exploration and discovery.

When thinking about course change, it is difficult to stay aware of all the details as you make changes. It is easy to move learning items around and lose track of some of the ordering details. For example, it might look like a good idea to move activity seven earlier and forget that there is a *prerequisite* relationship between activity five and activity seven. If this error is not caught, students will have difficulty completing activity seven in this new order. This creates an unnecessary crisis during the delivery of the course. Being able to manipulate the course graph in this way that maintains the integrity of the structure, helps an educator be able to see the course in a variety of ways that are potentially successful options. In the same scenario, when the instructor drags activity seven to somewhere before activity five, they will visually see that *prerequisite* relation and schedule accordingly. This is a powerful tool.

4.4 Summary

The application of these four transformations removes the temporal restraints imposed by a linear, time-based presentation of learning material. This removal of temporal restraints facilitates more flexible organizations that become particularly valuable in asynchronous settings such as online courses, technical training, or competency-based learning. The ability to manipulate this less restrictive structure becomes a valuable tool for exploration.

Figure 4.16 shows the result of the application of the transforms *T1* and *T2*. The revised graph affords much greater leeway in the organization, presentation, and order of selection of material for the instructor and the student.

Once the *T1* and *T2* transforms have been applied, many of the original organizational limitations have been removed. This opens the way for alternative arrangements of the learning items.

The graph in Figure 4.17 shows the learning items organized by topic. This arrangement separates the learning items in several distinct topic groups. The large group in the middle reflects the interrelated nature of several topics. This provides a visualization of how topics are related and how they might be rearranged. There is no visualization of *includes* relations.

The graph in Figure 4.18 is clustered by units. The similarity between the graph arranged by topic and this one indicates that the units in the original course organization grouped learning items into units by topic. Order of the units is not restricted by either temporal or topical relations. There are *precedes* relations between individual learning items in the five units in the center of the graph. In the first and second grouping there

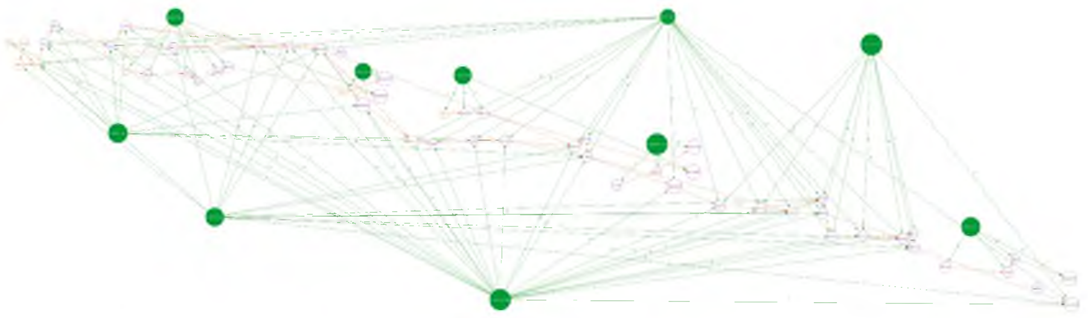


Figure 4.16. Course Graph After the Application of Transforms *T1* and *T2*.

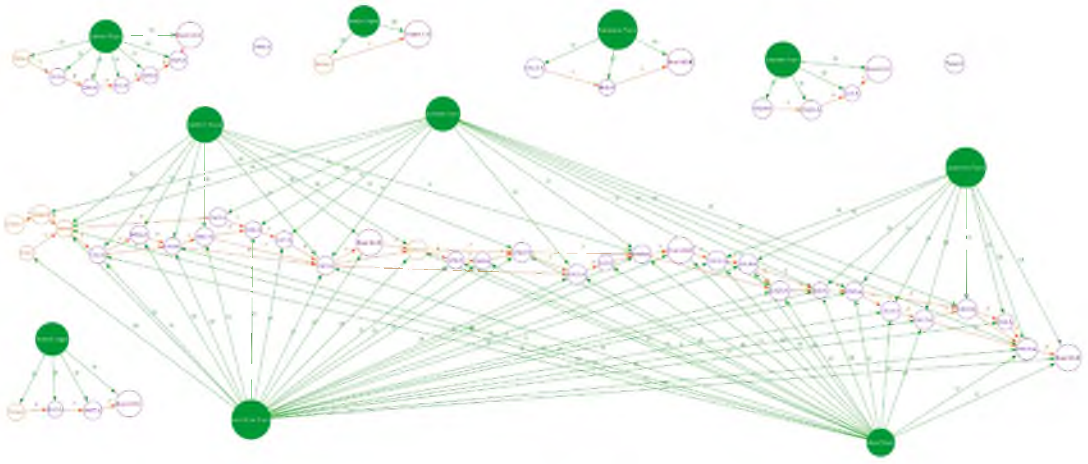


Figure 4.17. Course Graph Arranging Learning Items by Topic.

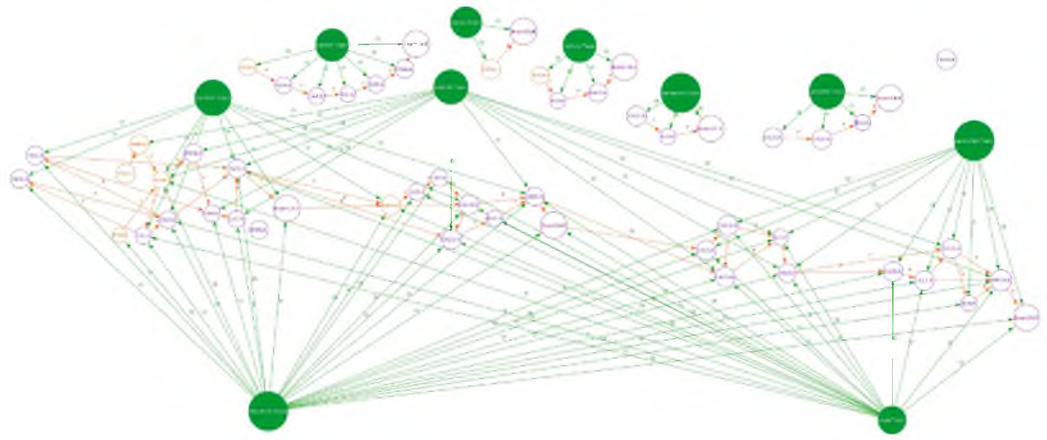


Figure 4.18. Course Graph Arranging Learning Items with Clustering by Units.

are *precedes* edges going from one item in unit one to a learning item in unit two, and also *precedes* relations from learning items in unit two to learning items in unit one.

There is information in these graphs that can be visually retrieved. Consider the following:

- How many topics occur in a specific assignment? This question can be answered by looking at how many *topic* edges come into an assignment. In the sample course, only one topic *occurs-in* each of HW3, HW4, and HW7 while four topics *occur-in* each of HW5, HW8, HW9, and HW10.
- What units can be rearranged without interfering with *precedes* relations? The answer to this question can be found by looking at the *precedes* edges between unit clusters. For those units with no *precedes* edges between them the order can be changed without disrupting the temporal order restrictions expressed by these edges.

As the educator manipulates and considers, possibilities become more concrete and manageable. Having access to this kind of visual information has the potential to provide meaningful insights very quickly. This facilitates change.

4.4.1 Evaluation

At this point the evaluation criteria for Section 1.7.1, Course Content Analysis, and Section 1.7.2, Graph Transformations, has been met. Below is an analysis of the evaluations.

4.4.1.1 Evaluation Course Content Analysis

ENABLE is able to extract metadata about the learning materials from the LMS using the Canvas API. This data is analyzed to identify information about individual learning items as well as the structure of the existing course. This information is incomplete, so additional information is gathered from the user through a graphical user interface. If date information for a learning item does not exist, ENABLE assigns a date related to the unit the learning item is included in. For missing topical information, the user interface presents the learning items to the user and provides a means for the user to add additional content or directly add or remove topical relations. The data gathered from the LMS and the information entered by the user are analyzed to identify relations between learning items. This extracted information is used to create a graph representation of the existing course that represents learning items and topics as nodes and relations as edges within the graph. This graph shows the type of each learning item and the *precedes*, *occurs in*, and *includes* relations.

The course content and analysis phase has been run on three existing courses, cs1030, a general education CS0 course, cs2610, a sophomore-level web applications course, and cs1405 - a CS1 lab course. All three courses were taught at Utah State University. The courses are referred to as course 1, course 2, and course 3, respectively, in the following reports.

ENABLE associates each learning item with a specific type. There are three types of learning items. The learning items of the resource type have no points associated with them and no due date. The learning items of the assignment type have a due date and points possible. The learning items of the exam type are a special case of an assignment. This type has a due date and points possible and is associated with an exam in the course. In all three courses ENABLE was able to identify the correct type of all learning items gathered from the LMS.

There were five learning items that were not gathered from the LMS, one from course 1 and four from course 2. These learning items consisted of two .zip files and three links to external sources. Currently ENABLE does not process .zip files or links to external sources, so these items were not included in the work. The capability to handle these types of learning items could be added in the future.

Temporal ordering is determined by the due dates. These data are available from the LMS for both assignment and exam learning items. Resource learning items have no date associated with them in the LMS, so they are not considered in this comparison. The temporal ordering of the 28, 47, and 38 dated learning items from course 1, course 2, and course 3, respectively, was accurately identified. The temporal ordering of the graph was accurate in all cases.

Topics are not directly represented in the LMS, so to do this comparison the title of the module the learning item was in and the title of the learning item itself were considered. If a topic word or word phrase showed up in either of these, the topic was counted. Each learning item then had a count of how many topics occurred in it in the LMS. A count was also done of the number of topic relations for each learning item in ENABLE. These topic counts were compared.

For course 1 there were 24 learning items that had the same topic count in both the LMS and ENABLE. There were 22 learning items that had a higher topic count in ENABLE. There were five learning items that had a larger topic count in the LMS. For all five of these learning items the additional topic was an error in the LMS that resulted from a topic word in the module title that was not actually in a learning item. In all cases the number of correct topic relations was either the same or better in ENABLE (see Figure 4.19).

For course 2 there were 45 learning items that had the same topic count in both the LMS and ENABLE. There were 22 learning items that had a higher topic count in ENABLE. There were five learning items that had a larger topic count in the LMS. For four of these learning items the additional topic was an error in the LMS that resulted from a topic word in the module title that was not actually in a learning item. In the remaining one, it was an error in ENABLE where the topic relation was left out. In all cases the number of correct topic relations was either the same or better in ENABLE. See Figure 4.20.

For course 3 there were 59 learning items that had the same topic count in both the LMS and ENABLE. There were 28 learning items that had a higher topic count in ENABLE. There were no learning items that had a larger topic count in the LMS. See Figure 4.21.

In all three performance areas for the course content analysis section of the work, ENABLE's performance was high. That performance is rated in Table 4.1.

4.4.1.2 Evaluation Graph Transformations

Using the course map developed for each course, graph transformations can be made. Some of these transformations are system generated, some are user directed, and some use a combination of system-generated and user-directed information to perform the transformations. All transformations are restricted to valid organizations that conform to the rules of directed acyclic graphs and the semantic rules imposed by the relations.

ENABLE is able to produce the following transforms.

- T1: Topic-based *Precedes* Elimination Rule.
- T2: Topic-based Exam Splitting Rule.
- T3: Replacing *Precedes* Relations with *Prerequisite* Relations.
- T4: Instructor-directed Transformation.

4.4.2 Additional Courses Produce Course Maps

In addition to the three sample CS courses analyzed and reported in the previous sections, ENABLE was used with seven other courses. These courses included the subject areas of Instructional Technology, Chemistry, English, Languages, and History. Each course had a different teacher. As with the three CS courses, ENABLE was able to gather learning item data using the Canvas API and generate course maps. The instructors of the two Instructional Technology courses were available to provide expert knowledge about those

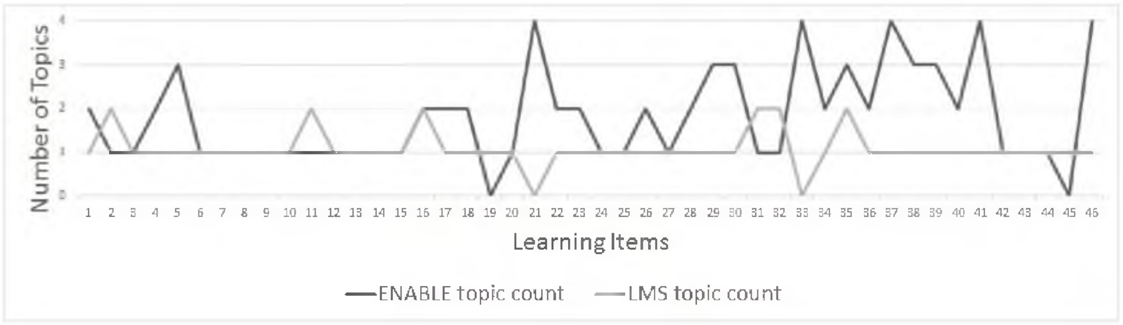


Figure 4.19. Comparison of Topic Counts in LMS to Topic Counts in ENABLE for Sample Course 1.

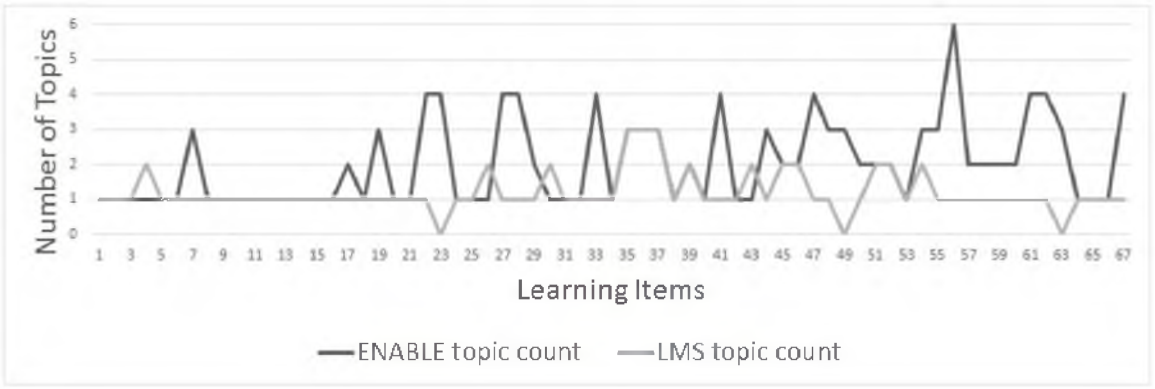


Figure 4.20. Comparison of Topic Counts in LMS to Topic Counts in ENABLE for Sample Course 2.

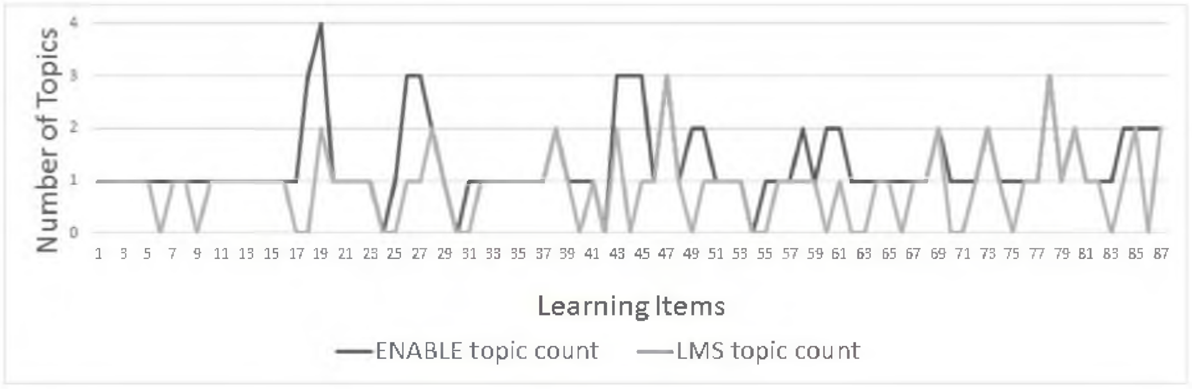


Figure 4.21. Comparison of Topic Counts in LMS to Topic Counts in ENABLE for Sample Course 3.

Table 4.1. Performance of ENABLE on Performance Measures in Course Content Analysis Section.

	Course 1	Course 2	Course 3
Learning Items of Correct Type	5	5	5
Temporal Ordering of Learning Items is the Same	5	5	5
Topical Relations Correctly Represented in Course Map	5	5	5

two courses. This allowed the course maps for these courses to include details about topically based *occurs in* and *prerequisite* relations. For all the courses the temporally based *precedes* relations were extracted from the data available from Canvas.

Each of the additional courses used the Modules tool in Canvas. This allowed the inclusion of the unit-based *includes* relations. However, some of the learning items were not included in the modules tool. This left these learning items with no *includes* relations. After discussion with the available instructors, it was noted that an interface that facilitated the editing of *includes* relations, similar to the interface that provides editing of the *occurs in* and *prerequisite* relations, would be beneficial.

The graph transformations T1, T3, and T4 were produced in the cases where an instructor was available. None of these courses contained exams, so the T2 transform was ineffectual.

This demonstrates the ENABLE system can be used in a variety of subject areas and with courses developed by different instructors. The system was able to produce all the course maps on all the additional courses. Each of the course maps was able to be manipulated. When expert knowledge was available, additional information was represented in the course maps.

CHAPTER 5

ARTIFICIAL STUDENT AGENTS

One of the differences between face-to-face courses and online courses is the possibility of individual students moving through the learning items in different orders. In a classroom setting it is not likely that students could be working on different learning items. However, in an online course each student could be on a different learning item at any given time. This introduces an entirely different component to a course. Is it possible to allow students to choose for themselves what order to complete the learning items? Can such flexibility be supported by the ENABLE system? Is it important to establish some limitations to the ordering? Can those limitations be enforced?

To explore these questions, artificial student agents were created. These agents are able to traverse the course map in a variety of node sequences. The agents are limited only by prerequisite relations. A learning item cannot be attempted until the agent has visited all the prerequisite learning items. When the agent visits a node, it can choose to work on the learning item or not. A trace of the order the learning items are visited is recorded as each agent moves through the learning items, also referred to as traversing the graph. These agents can be used to traverse the graph over and over, producing a new trace for each traversal. The traces from these traversals demonstrate a large variety in the order in which the learning items can be attempted. This demonstrates the possibility of allowing students to choose the order in which they move through the learning items. This increases the variety of changes that are possible.

Each automated agent has a set of characteristics and implements decision-making. Each agent can perform multiple and varied traversals through the course map. These traversals are quantified by two values, the final overall score and a rating for topic cohesion. The final overall score is a percentage computed from the individual scores and weights of the learning items (see Equation 5.11). The topic cohesion is a value that indicates how topically related the order of the traversal is. A high number for topic cohesion means the traversal ordered the learning items closely by topic. A lower number indicates there was more topic deviation between the learning items during the traversal. Topic deviation is when there are no common topics between adjacent learning items.

5.1 Student Agent Decision Making

In an effort to more closely simulate real students, each agent has four characteristics. These characteristics impact how often an agent decides to not complete a learning item, how well they do on a learning item, and how they choose which learning item to do next:

- Intelligence
- Work Ethic
- Background
- Distractibility

They are abbreviated in the formulas as I, W, B, and D. Each of these characteristics can have a value of 1, 2, or 3 (integers).

The value 1 represents the least favorable and 3 the most favorable value for each characteristic:

- Intelligence: 3-high 2-medium 1-low
- Work Ethic: 3-high 2-medium 1-low
- Background: 3-high 2-medium 1-low
- Distractibility: 3-low 2-medium 1-high

These numbers can be used directly in the formulas described below.

An agent first decides which learning item to consider next. The ones that are allowed are any learning item which has no remaining prerequisites. But there is a bias about working on learning items in the same unit. This bias is based on the agent values for background and distractibility. The assumptions are that students with greater background are more comfortable moving between multiple units, and the higher the agent's distractibility the more likely they are to move from unit to unit. The number of units to consider is determined using a table look-up. The key to the table is computed using the following formula:

$$k = B + (4 - D) \tag{5.1}$$

where B is the agent's rating for background and D their rating of distractibility. Table 5.1 shows the maximum number of units to consider at any given time based on the key.

Table 5.1. Number of Units to Consider When Choosing Next Learning Item.

Key	Number of Units to Consider
6	All remaining
5	3
4	2
3	1
2	1

A *course map* is a graph, $\mathcal{M} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of learning item, topic and unit nodes, and $\mathcal{E}$ is the set of *precedes*, *topically precedes*, *prerequisite*, *occurs in*, and *includes* edges (relations). Then the *course map* is $\mathcal{C} = (\mathcal{L}, \mathcal{R})$, where $\mathcal{L} \subset \mathcal{N}$ is the set of learning item nodes and $\mathcal{R} \subset \mathcal{E}$ is the set of *prerequisite* edges. A *path* of length k is any legal sequence $P = \{n_1, n_2, ..., n_{k+1}\}$, where $n_i \in \mathcal{L}$ and $\neg\exists i,\ j \ni n_j$ *prerequisite* n_i and $i < j$. Let P^S be the set of nodes in the path P. Then a *traversal* of $\mathcal{C}$ is a sequence of paths $T = (P_1, P_2, ..., P_q) \ni \forall i P_i$ is a path and $\forall ij P_i^S \cap P_j^S = \emptyset$.

Each learning item, n_i, has a list of prerequisites $Q_i = n_1, n_2, ..., n_l$ where $n_j \in \mathcal{L}$ and $\exists i,\ j n_j$ *prerequisite* n_i. Each learning item also has a remaining prerequisite list, S_i, where $S_i \in Q_i$ is the set of prerequisite items that have not yet been visited. T^s is the set of all remaining prerequisite lists. A learning item n_i cannot be visited until its remaining prerequisite list $T_i = \emptyset$.

The selection of which learning item to work on next is implemented by selecting the first N units that have remaining items to consider, where N is the maximum number of units the agent will consider. If there are fewer than N units remaining, all the remaining units are considered. Within each of these units any learning item n_i where $T_i = \emptyset$ is added to the consideration list.

The agent randomly selects one item from the consideration list. This item is then visited. Once an agent visits a learning item, it will determine whether to work on the item or not. When an agent decides not to work on a learning item, they receive a zero score for that item. When an agent decides to work on a learning item, a nonzero score is generated for that item. When a student agent visits a learning item node, that node is removed from all remaining prerequisite lists.

In the real data the zeros are significant. Although they have low probability, they have significant impact on the final score. Consider Table 5.2. The first value in each pair shows the final scores from the score data from the existing course. The second value in each

Table 5.2: Compare Final Scores Computed With and Without Zero Scores. Each Pair Consists of Final Score Computed with Zeros/ Final Score Computed without Zeros.

Final Score Pairs

7.89	87.44	82.14	85.81	92.25	94.18
11.71	97.43	82.49	93	92.32	94.69
18.44	77.09	82.95	84.77	92.63	94.9
20.63	51.44	83	90.83	92.81	94.83
27.06	84.63	83.13	86	93.02	96.34
30.12	63.93	83.32	83.4	93.16	92.61
30.84	79.41	83.93	93.05	93.17	93.05
41.68	69.92	84.21	83.86	93.23	94.65
43.3	63.5	84.25	88.98	93.58	93.38
53.86	90.36	84.28	91.44	93.6	93.55
59.06	74.24	84.86	89.35	93.68	95.52
61.09	75.73	84.87	84.16	93.88	93.88
61.83	81.3	84.9	84.71	93.98	93.44
64.19	76.76	85.01	86.58	94.22	93.77
66	68.52	85.21	88.99	94.25	94.1
67.23	78.62	85.34	85.24	94.36	96.74
68.39	76.1	85.34	88.93	94.5	96.89
68.64	76.21	85.64	87.63	94.55	95.53
68.77	77.83	86.11	85.91	94.6	94.6
68.79	73.35	86.26	86.48	94.7	93.87
69.77	76.52	86.35	96.92	94.71	94.66
69.95	74.4	86.67	91.05	94.73	96.9
70.23	87.5	87.33	89.16	94.81	94.36
70.24	73.64	87.5	87.29	94.91	94.8
70.3	75.09	87.76	92.12	94.94	94.33
71.01	71.95	87.91	87.61	95.04	94.7
71.16	81.4	88.07	87.9	95.25	95.05
71.21	84.97	88.08	87.67	95.36	95.11
72.34	80.76	88.28	90.4	95.69	95.69
72.75	76.79	88.51	90.78	95.89	95.29
73.04	78.18	88.77	95.07	95.94	96.33
73.22	74.56	88.88	92.15	96.04	96
73.38	76.5	89.04	88.93	96.13	96.12
74.54	84.94	89.23	90.65	96.24	96.24
74.68	80.35	89.28	88.4	96.29	95.7
74.86	74.38	89.69	89.24	96.45	96.36
74.96	82.4	90.17	90.17	96.46	96.46
75.46	84.37	90.23	90.18	96.55	98.92
77.42	85.9	90.43	90.78	96.7	96.65
77.47	91.49	90.78	92.9	96.87	96.82
77.59	81.14	90.8	90.61	96.93	96.7

Continued on next page

Table 5.2 – *continued*
Final Score Pairs

78.61	80.17	90.8	90.62	97.15	97.12
79.72	79.36	91.36	93.39	97.37	97.86
80.15	79.42	91.38	90.74	97.95	97.91
81.12	85.67	91.42	90.72	98.67	98.67
81.34	84.14	91.73	96.65	98.77	98.77
81.92	88.64	91.79	94.14	98.86	98.84
81.99	84.9	91.95	93.97		

pair shows the final score when zeros are not included in the computation. This number expresses how well the students did when they choose to do a learning item.

For the students who are at the top 20% of the class, the difference is not significant. Here we see where very few learning items are skipped. The difference when the average total score is computed with zeros and without zeros is .0002862. This number could be overlooked without making much difference on how a student agent works. However, for the bottom 20% of the student population the difference between the total average score computed with zeros and without zeros is .230675. This number is important.

To see the significance of this issue, let us consider a few specific students. The first student we consider received 27.06% as their final grade. Simply looking at this final score would indicate that the student did very poorly, and if we created an agent to represent them, we might choose to give them low scores on all the learning items they attempted. But looking closer we see that this student received an overall average of 84.63% on the learning items that they completed. This student participated actively for the first third of the class, including exam one. The scores they received on the work they did indicate this student could have successfully completed the course.

The next student looked at had an overall final score of 59.06%. This student did 18 of the 38 graded learning items. For the learning items they completed they scored an average of 90.36%. This student participated through to the end of the semester and could obviously do the material. Somehow this student missed the first exam.

The last student we will look at received a final score of 77.47%. This is certainly a passing grade. This student did 34 of the 38 graded learning items. He took all three exams, missing just two homework assignments and two quizzes. The score he received on the learning items he completed was 91.49%, a solid A-. In many different scenarios, it is the learning items that are not attempted which have the most impact on the final grade.

The zero score is an anomaly, not covered by the normal Gaussian curve. But to reflect more closely the actual data, it is an important detail that should not be overlooked. Therefore zeros are handled in a distinct way. Zeros are handled differently for graded learning items and ungraded learning items. We begin with an explanation of the graded items

5.1.1 Determining Whether to Do a Graded Learning Item

The number of zeros to include in any agent traversal of the learning items is computed from the actual data. Canvas allows the instructor to categorize learning items into different groups. Analysis of the data shows that there is a significant difference in the number of zeros, depending on the assignment group. Table 5.3 shows the percentage of zeros for each assignment group in one of the sample courses. The first row lists the percentage of zeros for all students in the data set. The following rows only include a cross section of the students. The students are separated by percentile rankings.

The decision of whether to do a graded learning item or not is made based on the characteristics of work ethic and distractibility. The assumptions are that those who rate high on work ethic are more likely to complete a learning item while those with high distractibility are more likely to skip them.

The percentage of how often a student agent will choose not to work on a graded learning item is computed from the existing score data. The percentage of zeros is determined for each assignment group and three groups of students. Assignment groups are identified during the course analysis phase and a group number is associated with each learning item. The groups of students used in this analysis are determined using percentiles and the final course score. The high student group are those with a final score between 90-94 percentiles, inclusive. The middle group are those students with a final score between 45-54 percentiles, inclusive. The low student group are the students with a final score between 5-9 percentiles, inclusive.

For each group of students and each assignment group the percentage of zeros is computed with the following formula:

$$P = Nz/N \tag{5.2}$$

where P is the percentage of zeros, Nz is the number of zero scores, and N is the number of scores.

Each percentage is evenly split between work ethic and distractibility.

$$W_i = .5 * P_i \quad \text{for } i = 1 \text{ to } 3 \tag{5.3}$$

$$D_i = .5 * P_i \quad \text{for } i = 1 \text{ to } 3 \tag{5.4}$$

where W_i is the work ethic component of the percentage rate of zeros for student agents, with work ethic = i, and D_i is the distractibility component of the percentage rate of zeros for student agents, with distractibility = i. P_i is the computed percentage for the high (3), middle (2), or low (1) student group. The work ethic component and the distractibility component are added together to get the students' overall rate of zeros, Z_i, for each assignment group.

$$Z_i = W_i + D_i \tag{5.5}$$

Table 5.3. Percentages of Zeros for Assignment Types and Student Percentile Categories.

	Overall	Assignments	Quizzes	Activities	Exams
All	11.8%	9.7%	18.7%	12.7%	4.0%
90-94	1.1%	0.0%	3.5%	1.4%	0.0%
45-54	5.6%	4.9%	10.0%	6.0%	0.0%
5-9	33.8%	32.5%	50.0%	34.3%	14.3%

A random number is generated in the range of 0-1. If the number falls below the zero frequency rate for that agent and learning item type, the agent does not work on the learning item. If the number is equal to or greater than the frequency rate, the agent works on the learning item.

If the agent decides not to work on the current graded learning item, they receive a score of zero and the item is removed from the consideration list and all remaining prerequisite lists.

If the agent decides to work on a graded learning item, the next step is to compute the score for that learning item. The computation of the score is different based on whether the learning item is an exam or not.

5.1.2 Determining Whether to Do an Ungraded Learning Item

To determine whether a student agent works on an ungraded learning item, work, distractibility, and background are considered. As with graded learning items, the assumptions are that those with a high work ethic rating are more likely to complete a learning item, while those with high distractibility are less likely to complete an item. In addition, background is considered for ungraded items. The assumption is that those with more background are less likely to engage with the supplemental material. The following formula computes a percentage of how often an agent chooses to not work on an ungraded learning item.

$$P = \alpha_0(2 - W\alpha_1 + B\alpha_2 - D\alpha_3) \tag{5.6}$$

where W is the agent's value of work ethic, B is the agent's value of background, D is the agent's value of distractibility, and α_i are coefficients determined from analysis of previous classes.

The result of this computation becomes the zero threshold for this particular agent on ungraded learning items. A random number is generated in the range of 0-1. If the number falls below the computed percentage, the agent does not work on the learning item. If the number is equal to or greater than the computed percentage, the agent works on the learning item.

The value for an ungraded item is either true or false, depending on whether the agent decided to work on that item or not. If the agent decides not to work on the current ungraded item, the value is set to false. When the agent decides to work on an item, the value is set to true. In either case, the item is removed from the consideration list and all remaining prerequisite lists.

5.1.3 Scoring Non-Exam, Graded Learning Items

The grade computation depends on the value of all four agent characteristics. The greatest weight is on work ethic and intelligence. The max of these two values is used. The assumption is that either hard work or intelligence will produce a good result on these learning items. A smaller weight is put on the sum of background and distractibility. The assumption is that increased background and better focus will help with satisfactory completion of the item.

Each learning item has a different maximum points possible, and the distribution of scores from the existing course varies significantly from item to item. There were assignments that resulted in a nice Gaussian score distribution and others that had a more uniform distribution. To accommodate such variation in the possible scores, the first formula for determining the score computes a percentile.

$$P = max(0, min(0.99, max(I, W)\alpha_1 + sum(B, D)\alpha_2 - \alpha_3 + \in)) \qquad (5.7)$$

where I is the agent's value of intelligence, W is the agent's value of work ethic, B is the agent's value of background, D is the agent's value of distractibility, α_i are coefficients determined from analysis of previous classes, and $\in$ is the amount of noise computed with $\in \sim N(0, \sigma^2)$. This noise adds random variation to the scores produced by the agent. The maximum amount of noise can be set at run time. The default value when not specified is $\sigma = .15$.

The score associated with this percentile is computed using the following process. First, remove all zero scores. Zero scores have already been accounted for when the agent determined whether to complete a learning item or not. To include them here as well would give greater weight to zeros than is justified by the data. The remaining scores are then sorted in ascending order.

Using this sorted list of scores, the rank is determined using the following formula:

$$R = P(N + 1) \qquad (5.8)$$

where P is the desired percentile and N is the total number of scores.

If R is an integer, the Pth percentile is the score with rank R. When R is not an integer, the Pth percentile is computed by interpolation. Define R_I as the integer portion of R. Define R_F as the fractional portion of R. Find the scores with Rank R_I and with Rank $R_I + 1$. Interpolate by multiplying the difference between the scores by R_F and add the result to the lower score. Round this result to the nearest integer.

5.1.4 Scoring Exams

This grade computation depends on the value of all four agent characteristics. The greatest weight is on intelligence. The assumption is that intelligence will be the most likely indicator of success on an exam. The next greatest weight is on the value of work ethic. The assumption is that hard work on previous learning items will result in a better exam score. A smaller weight is put on background and distractibility. The assumption is that increased background and better focus will help with satisfactory completion of the exam.

The following formula is used to compute the percentile for this exam:

$$P = I\alpha_1 + W\alpha_2 + sum(B, D)\alpha_3 - \alpha_4 + \in \tag{5.9}$$

where I is the agent's value of intelligence, W is the agent's value of work ethic, B is the agent's value of background, D is the agent's value of distractibility, α_i are coefficients determined from analysis of previous classes, and $\sigma where \in \sim N(0, 12)$ is the amount of noise.

The resulting score is recorded by the agent. The item is then removed from the consideration list and all remaining prerequisite lists.

5.1.5 Computing the Final Score

After the agent has visited each node in the graph, a final score is computed. This final score it computed from the individual scores of each learning item and the weight of that item. Canvas allows instructors to set weighted percentages on each assignment group. These weights specify how much of the final grade is based on the specific assignment group. The assignment groups and the weights are shown for each of the sample courses in Table 5.4, Table 5.5, and Table 5.6.

ENABLE uses these weights to compute an individual weight for each learning item, using the following formula:

$$W_i = W_j^G(\frac{P_i}{P_j^G}) \tag{5.10}$$

Table 5.4. Weights of Individual Assignment Groups in Sample Course One.

Assignment Group	Weight
Homework	30%
Exercises	10%
Class Activities	20%
Exams	40%

Table 5.5. Weights of Individual Assignment Groups in Sample Course Two.

Assignment Group	Weight
Assignments	40%
Class Activities	20%
Exams	40%

Table 5.6. Weights of Individual Assignment Groups in Sample Course Three.

Assignment Group	Weight
Worksheets	50%
Labs	50%

where W_j^G is the weight of the assignment group, P_i is the points possible for this learning item, and P_j^G is the points possible for the entire assginment group.

The final score can then be computed using the following formula:

$$F = \sum_{i=1}^{N}(W_i S_i) \tag{5.11}$$

where W_i is the weight of the learning item, S_i is the score for the learning item, and N is the number of learning items.

5.2 Agents at Work

At this point the agents have been created and their decision-making has been encoded. The system can now use the agents to generate data. Each agent has four characteristics: Intelligence, Work Ethic, Background, and Distractibility. Each of those characteristics can have three values: high, medium, and low. There are a total of 81 unique character combinations. Information about seven agents with the characteristics shown in Table 5.7 is presented here.

Table 5.7. Characteristics of Student Agents.

	Agent_1	Agent_2	Agent_3	Agent_4	Agent_5	Agent_6	Agent_7
Background	3	2	1	3	3	3	1
Intelligence	3	2	1	1	2	3	3
Work ethic	3	2	1	3	2	1	3
Distractibility	3	2	1	2	3	1	1

Using the computation for zero frequency, each agent is assigned a threshold for zeros in each category, as shown in Table 5.8.

Now each agent traverses the course a certain number of times. For each run, a trace is produced that identifies the order the agent traversed the learning items. A score is kept of each learning item. At the completion of the run a final score is calculated. This final score is used to compare one run to another.

As a student agent traverses the course map, it keeps track of each learning item as it encounters it. This trace can be displayed. The printed trace includes a list of the labels for each learning item. For the learning items that the agent did not complete, the label is in parentheses. Following are three traces of one of the student agents with characteristics of background: 2, intelligence: 2, work ethic: 2, and distractibility: 2

[Nweb HW6 HW1 CA1 CA2 HW3 Vpub faq5 (Vorg) CA6 CA7 HW2 HW5 Ndarp CA3 Exer2 CA5 Nhist CA4 HW7 Nacm (CA8) Exer1 Nsty HW4 CA9 CA10 HW8 CA11 Exer3 Exam1 CA15 CA12 CA13 Exam2 CA18 CA19 CA14 Exer4 HW9 (CA20) CA16 Idea (CA17) HW10 Exam3]

[HW1 CA1 HW6 (Vpub) (faq5) Nweb Vorg CA2 HW2 HW4 HW3 CA6 HW5 CA7 Nacm CA3 CA5 Ndarp Exer2 CA4 Exer1 Exam1 CA8 Nhist Nsty CA9 CA11 HW7 Exer3 CA12 CA10 HW8 CA13 CA14 CA15 HW9 CA16 Exam2 CA19 Exer4 CA20 CA18 CA17 Idea HW10 Exam3]

[(Vpub) HW1 Vorg (CA1) HW3 (faq5) HW6 HW2 Nacm HW4 CA3 CA2 CA6 (Nweb) CA4 HW5 CA5 CA7 Exer1 (Ndarp) Exam1 Nsty Exer2 Nhist CA8 CA9 CA11 HW7 CA10 CA12 HW8 Exer3 CA13 CA14 HW9 Exam2 (CA18) CA15 CA20 Exer4 CA16 CA17 CA19 Idea HW10 Exam3]

These traversals demonstrate the variety of ways the course map can be traversed by a single agent. At the end of a series of traversals, the mean, max, min, and standard deviation of the final scores are computed. We can compare these values to agents with different characteristics. When comparing these we see that running the agent 1000 times

Table 5.8. Percentage of Zeros for Each Student Agent.

	Agent_1	Agent_2	Agent_3	Agent_4	Agent_5	Agent_6	Agent_7
Homework	.0000	.0485	.3247	.0242	.0242	.3247	.1623
Quizzes	.0357	.1000	.5000	.0679.	.0679	.5000	.2679
Activities	.0143	.0867	.3429	.0505	.0505	.3429	.1786
Exams	.0000	.0000	.1429	.0000	.0000	.1429	.0714
Resources	.1860	.3240	.4620	.2760	.2640	.5220	.3060

produces steady mean values (see Table 5.9 and Figure 5.1). The variation is still apparent in the max and min scores. This number of traversals allows extreme highs and lows. The topical cohesion also stabilizes with many traversals. It varies only by a single point between large traversals for any given agent.

5.2.1 Evaluation of Artificial Student Agents

In this section the evaluation criteria for Section 1.7.3 are considered. Seven student agents were created and many varied traversals were run. The traversals followed the constraints of the prerequisite relation and did not allow a learning item to be considered until any prerequisite learning items had been visited. The agent reported a final score which is a value between 0-1 and is representative of the a final weighted percentage that a grade would be based on in the example course. It also reported a topic cohesion value. A high topic cohesion value indicates the adjacent learning items in the traversal were often topically related. A lower topic cohesion value identifies that there were fewer adjacent learning items that were topically related.

Table 5.9. The Cumulative Results of Each Student Agent for Traversals of 1000 Paths.

B	I	W	D	topic cohe-sion	95% confidence interval	max	min	σ
3	3	3	3	22	[97.9-98.06]	100.00	91.24	1.37
2	2	2	2	22	[87.22-87.64]	95.24	74.54	3.43
1	1	1	1	22	[43.35-44.31]	74.94	17.58	7.80
3	1	3	2	19	[89.06-89.44]	96.83	74.10	3.12
3	2	2	3	22	[91.81-92.15]	98.39	80.48	2.72
3	3	1	1	19	[68.75-69.93]	88.78	36.64	9.51
1	3	3	1	22	[81.7-82.56]	95.94	57.22	6.96

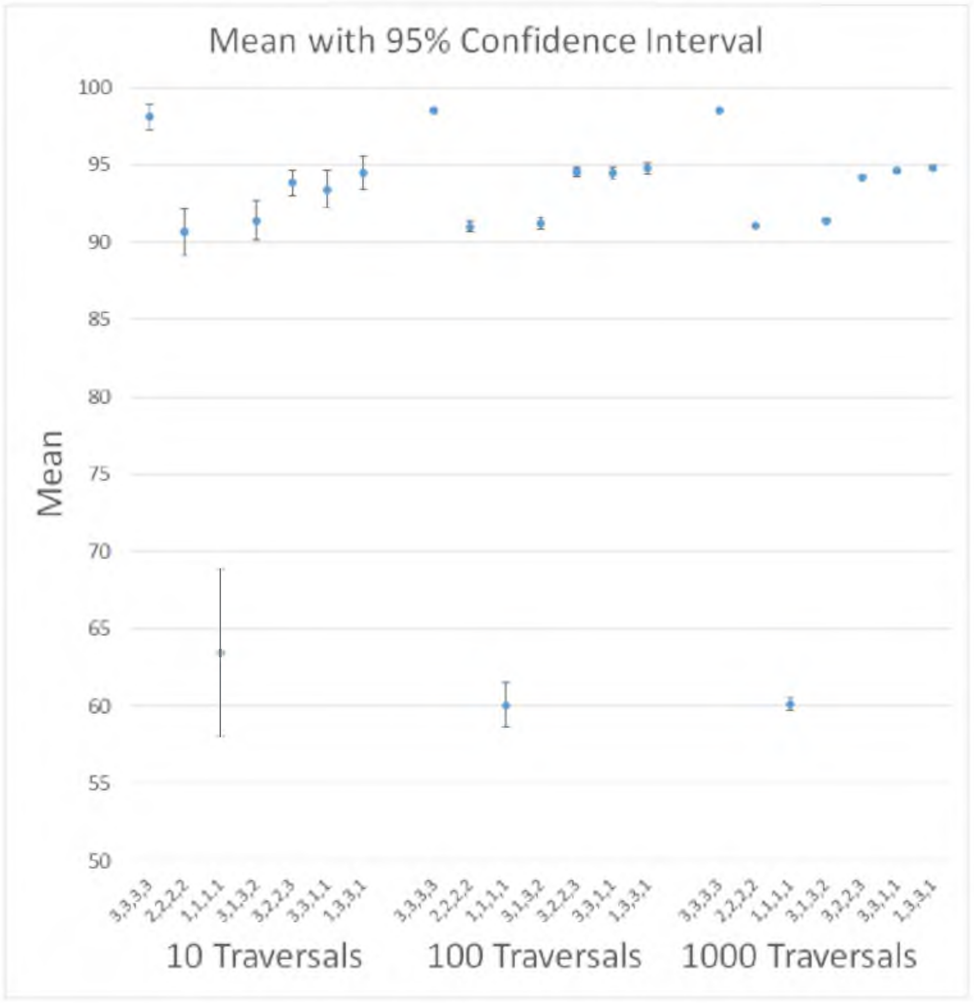


Figure 5.1. Chart of Mean and Confidence Rating for Seven Agents with 10, 100, and 1000 Traversals.

CHAPTER 6

BEHAVIOR MODELING

A probability model gives us a way to make predictions. Predictions can be used to inform students and educators about possible outcomes. With the data available in the existing course, can probability models be generated? If so, how accurate are they?

To answer these questions, several probability models to predict grades on learning items were created. These models are trained using the existing score data. What was discovered is that, indeed, probability models can be generated from existing data. Many of those models are able to predict individual scores with over 70% accuracy. These models can be sampled to produce data that have a distribution similar to the original data.

This predictive capability can be used to inform students about the likely results of choices they make. One of the clear messages from this work is that zeros have a significant impact on the overall success of a student. If a student was informed of the probable outcome of skipping an assignment, they may reconsider. If a situation required a large set of data that is representative of this specific course, these predictive models could produce those data.

6.1 Bayesian Network

The course map described in Chapter 3 represents the learning items and relations between those items using a graph. Chapter 4 discusses how these graphs are then transformed to provide better representations of the actual course and alternative organizations. In all these cases the graph that is produced is a directed acyclic graph (DAG). A Bayesian network is a DAG that represents a full joint distribution over a set of random variables, where each random variable represents the probability of a grade.

6.1.1 Building a Bayesian Network

ENABLE produces a representation of a course that includes learning items and a variety of relations between those items. Using this information a Bayesian network can be created. This network has nodes that represent the learning items and edges corresponding to the relations. Anonymized score data were provided by the Center for Innovative Design and

Instruction, the department at Utah State University that supports Canvas on that campus. These data are used to create the Bayesian network.

To build the conditional probability tables (CPTs), several possibilities must be considered. First, how to allocate individual values for each item must be decided. The scores themselves are discrete values. The number of possible values varies, depending on the points possible. Creating a column in the table for each possible value is problematic because of the size of the table and how sparse it would be. The effect is compounded by the fact that these values become inputs into other CPTs. For example, an assignment in one of the courses is worth 88 points. Creating a column in the table for each possible score from 0-88 requires 89 columns. This particular assignment has five prerequisites with possible points of 20, 20, 70, 10, and 20. Using each discrete value for all of these parents will produce 7,232,841 possible input combinations, creating a table with 643,722,849 cells. The actual data contains one score for each of the students for a total of 143 values for this learning item. This produces a table that consumes too much memory and is far too sparse. This problem can be resolved by separating the values into fewer possible choices.

To address this problem, the scores are separated into a specified number of buckets. Two different processes for defining the bucket allocation are considered. The first separation is determined using percentiles. In this approach variance is allowed for how many buckets to include. The percentile range is determined by dividing 100 by the number of buckets. Once the percentile range is determined, the actual percentile scores are computed. There are times when the example scores fall evenly into the nonzero buckets. This happens when the percentile breaks happen between different scores. However, it is often the case that there are many scores with the computed percentile score. When this occurs, all the scores equal to the computed percentile score are allocated to the same bucket.

The second approach for allocating scores is to define buckets by grades A, B, C, and D/F. This results in four buckets when zeros are included in the D/F bucket and five buckets when a separate zero bucket is included.

When analyzing scores for learning items, the score of zero has special significance. It is a score that shows up on most learning items but it regularly falls outside the normal curve. The zero score most often reflects that the student did not participate in the learning item. This reflects something very different than a low score. A low score indicates that a student participated but did poorly. The existence of the zero score is an anomaly in data that is often considered a Gaussian distribution. To increase the accuracy of the predictive model, this score will be considered in a distinctive way.

The other anomaly occurs at the maximum points possible. Many learning items have a higher than expected value at this point. This arises from the fact that higher scores are not possible. This upper limit will disrupt the normal extension of a Gaussian distribution and congregate an increased number of scores at this maximum attainable score.

To create the CPTs, the parent nodes must be clearly specified. ENABLE provides a variety of representations of the course map. This allows us to produce several variations of the Bayesian Network. The relations that will provide parent node connections in the CPTs will come from the course maps. Both the immediately precedes and the prerequisite relations will be represented.

The immediately precedes relation produces exactly one parent for each node except the first node, which will have no parent. This will produce a consistent size throughout the CPTs. The first learning item will have a $1xB$ table where B is the number of buckets. For all the learning items after the first, each table will be BxB in size. This limited dimension allows us to increase the number of buckets without running out of resources.

The prerequisite relations produce tables with varying dimensions. The width of each table will still be determined by the number of buckets specified and will be B wide. It is the number of inputs that will vary. As the number of parents increase, the size of the table increases exponentially. The table will have B^P rows, where B is the number of buckets and P is the number of parents. This table will have $B^{(P+1)}$ entries.

There are cases where a learning item has several prerequisites. For example, in three of the sample courses the maximum number of prerequisites was 8, 16, and 4. In the case where the item has this maximum number of prerequisites and there are four buckets, the table will have 262,144, 17,179,869,184, and 1,024 entries, respectively. These same courses have 143, 57, and 216 actual scores for this item. This leaves most of the entries empty. These empty entries are regularized, so they are considered to have some small probability, but the ratio between actual data points and entries is so small as to reduce the accuracy of the predictions.

One variation that can be used to reduce the number of parents and accordingly the number of entries is to prune the prerequisites. In some cases a prerequisite list contains a prerequisite that has as its prerequisite an item that is in the original prerequisite list. This could be considered redundant. It is possible to consider the prerequisite relations as transitive. This would mean that if A *prerequisite* B and B *prerequisite* C, then A *prerequisite* C. If this transitive rule holds, then it is sufficient to list A *prerequisite* B and B *prerequisite* C without including A *prerequisite* C. This transitive property will hold when all the topics that are common in A and C are also included in B. This pruned prerequisite list is significantly smaller. In the three sample courses the maximum pruned prerequisite list is 5, 4, and 3. This reduces the number of entries when 4 buckets are used to 4,196, 1,024, and 256. This significantly increases the ratio of data points to entries.

6.1.2 Variations of the Bayesian Network

There are several factors that can be manipulated to produce different versions of the Bayesian network. Following is a list of the variations that were considered.

- Number of Buckets
  - This is a value that can be entered as a parameter. This number directly influences the number of entries in each table.
- Parent relations - How the parents are determined can be specified in the following ways:
  - no parents
  - immediately precedes
  - prerequisites
  - pruned prerequisites
- Buckets determined by
  - percentiles
  - grades
- Zero bucket
  - zero bucket used: one of the buckets is reserved exclusively for zero scores
  - zero bucket not used: zero scores are entered in the grade or percentile bucket they fit

The number of buckets changes the model in two significant ways. First, increasing the number of buckets increases the number of input combinations that need to be considered. This increases the number of entries in each table. For example, consider a node that has three parents. If there are two possible values for each parent and the current item, there will be 8 possible input combinations and 16 entries in the table. If the number of buckets is changed to seven, there are 343 inputs into the table and 2041 entries (see Figure 6.1). As the CPT tables grow in size, the cost of processing, memory usage, and sparseness increase. By restricting the number of parents to four, we are able to increase the number of buckets to 10 and still be able to produce results. Any larger than that and it increased the demand for memory beyond capacity.

The second issue with increasing the bucket size is reducing the size of the target. Because each bucket contains fewer possible values, projections are less likely to hit the correct one. This increases the error rate of the model. For most of the analysis, the bucket size was set at three to five buckets.

Even more than the number of buckets, the number of parents impacts the number of entries in any given CPT table. Keeping the bucket size at four and changing the number of parents, we see an exponential increase in the entries in the CPT (see Figure 6.2). This significantly increases the cost of processing, memory usage, and sparseness of the data as the number of parents increases.

There are several ways to impact the number of parents in the Bayesian network. As listed, four options were considered. The first option was to have no parents. This is to consider the learning item in isolation without exposing any influence from other learning items. This keeps the size of each CPT table to the number of buckets. This option has the advantage of small CPTs but the disadvantage of not exploiting the relations between learning items. It also represents a common way to analyze a learning item. Using no parents restricts the data to the scores for that individual learning item. Contained in those scores are the mean, the min, the max, and the standard deviation. These are the statistics that educators often look at when analyzing a learning item.

The immediately precedes relation connects a learning item with the learning item that came just before it in time. Using this relation to identify parents keeps the number of parents for each node at one, making the size of the tables B x B, where B is the number of buckets. This option both gives us information about a related learning item and limits the size of the CPT.

The next variation based on parents considers the prerequisite relation. ENABLE extracts information about prerequisite relations using expert knowledge. These prerequisite relations become part of the course map and are used here in the modeling. The number of prerequisites varies from course to course and from item to item. Incorporating prerequisites into the model results in one parent per prerequisite. This can cause the problems associated with a higher number of parents as described above.

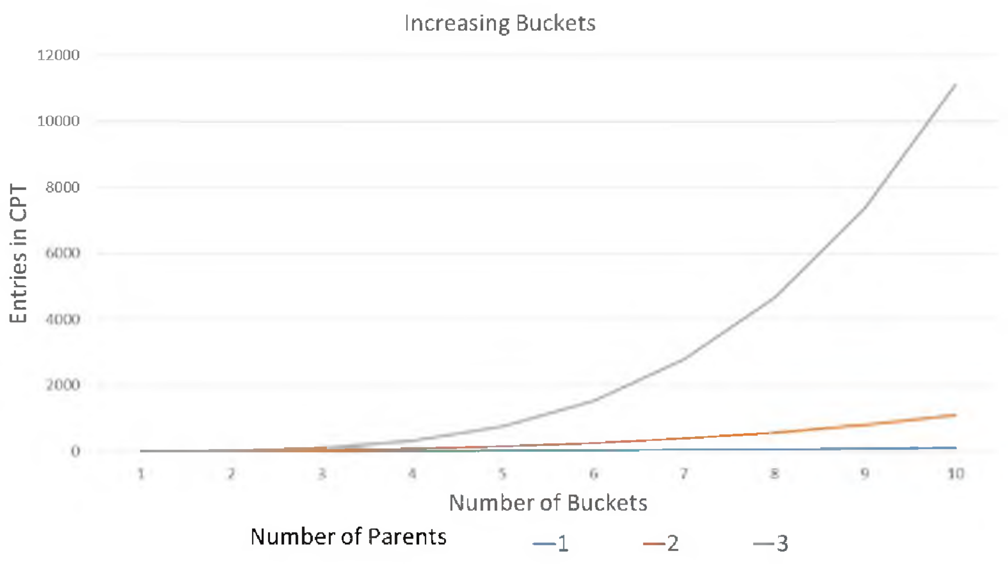


Figure 6.1. Chart Showing the Increase in CPT Table Elements as Number of Buckets Increases.

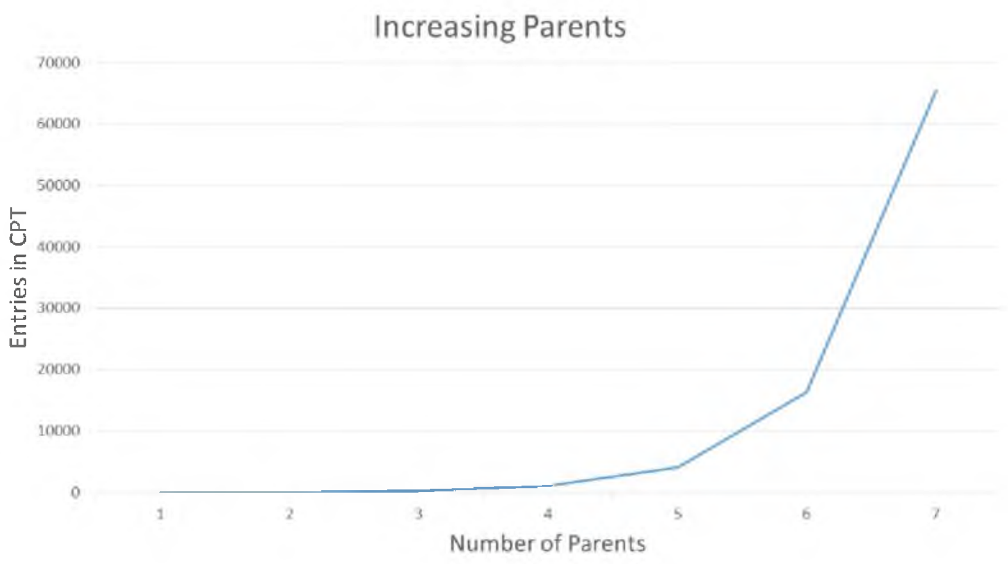


Figure 6.2. Chart Showing the Increase in CPT Table Elements as Number of Parents Increases.

Pruned prerequisite lists take advantage of the transitive nature of this relation and reduces the number of parents. This reduces the problem of the CPT sizes by reducing the number of prerequisites and therefore the number of parents for any given learning item.

There are two distinct ways to consider separation of data into buckets. They can be separated into grade buckets that have one bucket for each grade, A, B, C, and D/F. The other way is to separate scores by percentiles. To have the same number of buckets as grades, the buckets would be divided into 0-24, 25-49, 50-74, and 75-99 percentiles. These produce variations in the spread of scores between buckets. The percentile buckets tend to have more even numbers of items in each bucket. This is the case except when there are a large number of identical scores. There are some learning items that, if you do them, you almost always get full points; these learning items have two groups of scores, the full mark score and the zero score. These items do not spread evenly across buckets no matter which bucket classification strategy is used.

Finally, the issue of how to categorize learning items that were not worked on is addressed. When a learning item is not worked on, the student receives a score of zero. Zeros can have their own distinct bucket or can be included in the classification they would fit naturally. In the grade buckets they would fall in the D/F category. In the percentile buckets they would fall in the 0-24 percentile bucket. Separating the zeros into their own distinct bucket requires one more bucket, which increases the size of each CPT. As a score of zero identifies a distinct situation, namely, that a student did not attempt a learning item, it is more representative of the structure of the data to represent it in the model as a distinct category.

6.2 Linear Probability Model

Another approach to creating a probability model is to use linear functions. These models are also based on the actual course data. The linear model is able to scale because the amount of memory and computational resources needed increases at a linear rate. This produces a model that does not need to restrict the number of parents that each learning item has.

The linear probability model is created using the linear function and a variance based on the error model. The error model considers the difference between the actual values and the result computed by the linear function. These differences are used as the variance. The linear function, combined with this variance, is used for both prediction and sampling.

6.2.1 Building a Linear Model

The linear model in ENABLE uses linear regression. Linear regression works as follows: Given a random sample

$$(Y_i, X_{i1}, X_{i2}, ..., X_{ip}) \tag{6.1}$$

where $i = 1, \ldots, n$ and p is the number of features, the relation between the observations Y_i and the independent variables X_{ij} is formulated as

$$Y_i = W_0 + W_1 X_{i1} + \cdots + W_p X_{ip} + \varepsilon_i \qquad i = 1, \ldots, n \tag{6.2}$$

In the above, the $W_j's$ are the regression coefficients and $\varepsilon_i = N(0, \sigma)$ is the standard error.

The predicted values corresponding to the above model are linear functions of W_j. One function is produced for each learning item and may consider the scores of the preceding learning items. The formula includes an initial value, W_0, and a term for each feature, $W_j(X_{ij})$. Although there may be dependencies between the learning items, this is not considered when doing the linear model.

6.2.2 Feature Selection

The linear model can be varied by changing the features that are considered in the functions. Several options are considered. Except for the first item, any given learning item is preceded by other learning items. Each of those preceding learning items can be considered as a feature. The following list identifies ways the preceding learning items are considered.

- No other learning items are considered in the feature list.
- The immediately preceding item makes up the entire feature list.
- Differing numbers of immediately preceding learning items are included in the feature list.
- All the learning items listed as prerequisites to the current item are included as features.
- All the pruned prerequisites are considered as features.

Increasing the number of items included as features is not restricted by this model. This provides a great deal of flexibility in selecting what features to include in the model. The first linear model is one that does not include any other learning items in the feature list. This model uses the information that is available exclusively from the scores of the learning item currently being considered. This provides a good baseline. This information is commonly used to analyze the results of a learning activity and provides statistics like mean, max, min, and standard deviation.

The three relations considered in the remaining models are the precedes relation, the prerequisite relation, and the pruned prerequisite relation. The first one that uses the precedes relation is the immediately precedes model. This model uses the one learning item that comes immediately before the current learning item in time. We also considered two, three, five, nine, and 13 immediately preceding items. This is possible because the increased number of features can be incorporated in the functions without adding significant computation or memory expenditure. The final case that uses the precedes relation is the all precedes model. This model uses all the learning items that come before the current item.

There are two linear models that are based on prerequisite relations. The basic prerequisite model uses all the prerequisites for the current learning item in the feature list. The other model uses pruned prerequisites. This model uses the transitive property of prerequisites where some of the prerequisites can be removed from the list because they are already included as a prerequisite to an existing prerequisite. This reduced list of prerequisites, referred to as pruned prerequisites, is used to establish the feature list for this model.

The linear model does not consider zeros in any distinct way. To identify if there is an effective way to handle the special case of the zero score, a variety of methods were tried. In all the approaches discussed previously, zero is considered as nondistinct and handled like any other value.

One approach was to remove the data for any student with a zero score in the current item. This allowed the comparison of scores for students who had completed the learning item. Another approach removed the data for any student with a zero score in the current item and the data for students with zeros in any preceding learning items. This approach removed so many data points that the model ran into overfitting problems. Overfitting is when there are too few data points to support the number of features. To solve this problem, instead of eliminating the data of students with zeros, the zero was replaced using the function that had been created for that learning item. This produced a set of data that excluded zeros. None of these approaches yielded informative results and they are not included in the final analysis.

6.2.3 Mixed Distribution

The simple linear model does not represent zero scores very well and sometimes gives near zero probabilities for possible scores that simply have not been observed. For this reason the grade probability model is built using a mixture of three distributions:

1. Gaussian model that predicts the score, assuming the student completed the item. This is the same model as described in the previous section.
2. A model that predicts an incomplete (i.e., zero) grade. This distribution assigns a likelihood of 1 to a score of 0 and gives all other scores a likelihood of 0.
3. A uniform model that predicts the same probability for each score. The likelihood of each score is simply $1/number\_of\_possible\_scores$.

The likelihood of a given score is simply the weighted sum of the three component distributions.

$$l(s) = w_g * l_g(s) + w_z * l_z(s) + w_u * l_u(s) \tag{6.3}$$

w_z and w_g are determined by a logistic regression model that estimates the probability a given student will complete an assignment. w_u is determined by hand and is set to 0.02 for all reported experiments. This value can be adjusted to give greater or less weight to the uniform distribution. The logistic regression model uses the same features as the linear model described in the previous section.

Logistic regression uses the logistic function, which can take an input with any value from negative infinity to infinity and produce a value between zero and one that can be interpreted as a probability. The logistic function is defined as follows:

$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}} \tag{6.4}$$

where t is a function of a linear combination of explanatory variables and expressed as

$$t = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n \tag{6.5}$$

Now the logistic function can be written as

$$F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + ... + \beta_n x_n)}} \tag{6.6}$$

$F(x)$ is the probability of the dependent variable, given a linear combination of explanatory variables, x. β_j are the regression coefficients.

This approach of combining distributions is applied to sensor measurements in probabilistic robotics [3]. Its use here is a novel application of that process. This combined model is included in the analysis. See Figures 6.3 through 6.6 for an example of the component distributions and the resulting mixed distribution.

6.3 Comparing Models

After the Bayesian networks, linear models, and mixed models are created, the next step is to compare them. A variety of comparisons have been measured. These comparisons were done on a sample CS0 course. This course is a foundations of computer science course for non-CS majors. Each model generated 1,430 sample sets of data. Every sample set included a score for each learning item. This sampling was done by taking each learning item in the order it was presented in the original course. A score was generated for the learning item by sampling from the score probability distribution created by the individual model for that item.

The individual scores produced by this sampling were used to compute a final score. For a listing of the individual scores generated by each model and the computed final score, see Appendix D.

Some of the models being compared are built using a combination of the models. For example, the PrecedesThree/PrunedPrereqs uses a combination of the precedes three model which includes the three learning items immediately preceding the current item and the pruned prerequisite model.

6.3.1 Comparing Final Scores

Using the computed final scores, histograms were made to show the frequency of the scores. This first histogram, Figure 6.7, is the 143 scores that came from the actual course. There are fewer data points in this chart than the others. This histogram has 143 scores compared to 1430 in the other three final score histograms, so the scale is different. Each of the marks below the 67 score represent a single final score.

The next histogram is of a CPT model. This particular CPT model uses both the immediately precedes and the pruned prerequisite relations. It combines both the very current information by including the immediately preceding item as well as a more extended view of the preceding scores by including the pruned prerequisite information.

This histogram reflects some of the features of the original data, Figure 6.8. It has a tighter curve on the right side than the left. It does a nice job of showing the extended tail

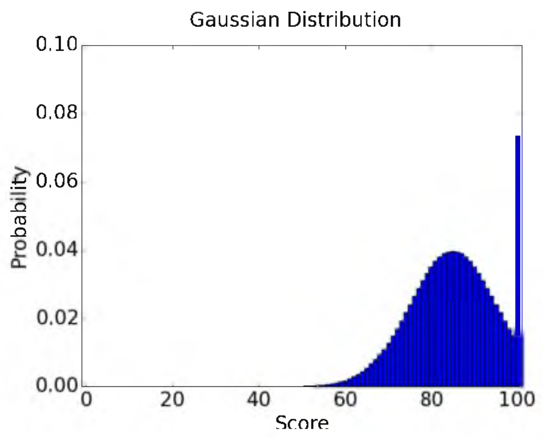


Figure 6.3. Histogram of Gaussian Probability Distribution.

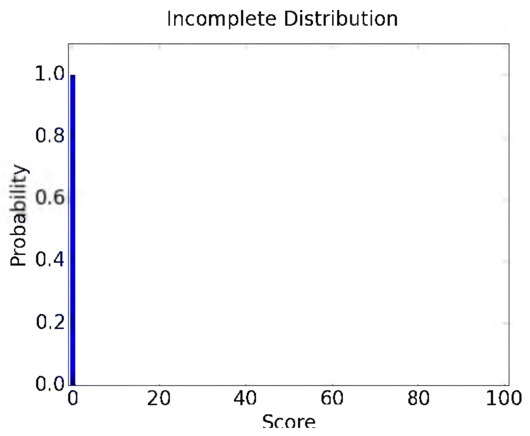


Figure 6.4. Histogram of Zero Probability Distribution.

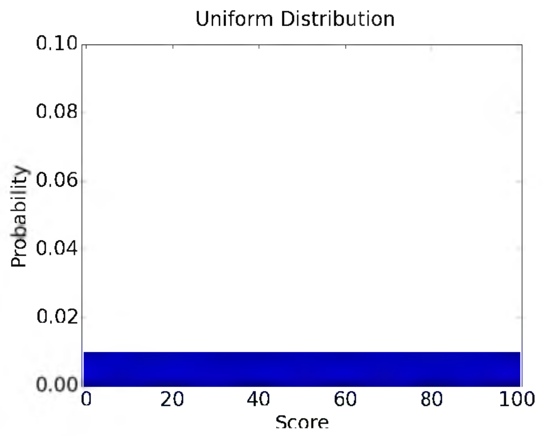


Figure 6.5. Histogram of Uniform Probability Distribution.

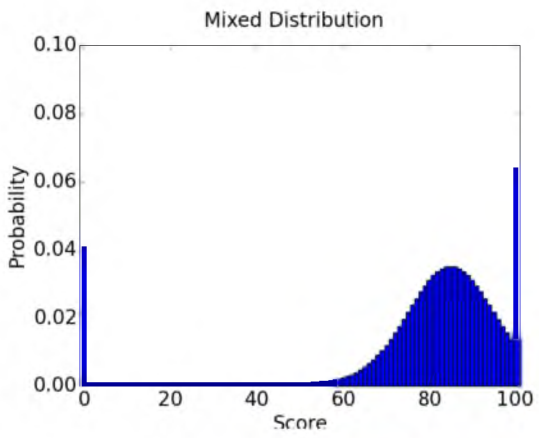


Figure 6.6. Histogram of Mixed Probability Distribution.

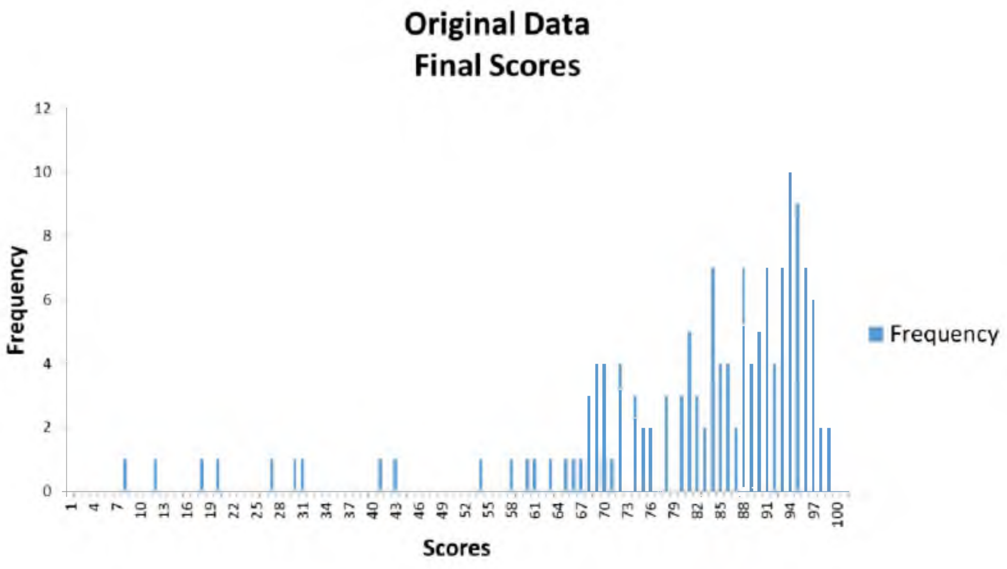


Figure 6.7. Histogram of Original Final Score Data.

on the left. It fails, however, to get the range of upper scores correct. It has only one score above 85. In the original data 56.6% of the scores were above 85.

The linear model histogram uses all the preceding learning items in the feature list, Figure 6.9. This model does a better job at producing the higher scores. It has a slightly steeper curve on the right side than the left, which also occurs in the original data. It does not produce any scores below 32. In the current data 5% of the scores are below 32.

The mixed linear model produces a histogram that most closely resembles the original data, Figure 6.10. It has the steep curve on the right side and a long tail on the left. The peak of this curve, however, is lower than the peak of the original data. In the original data,

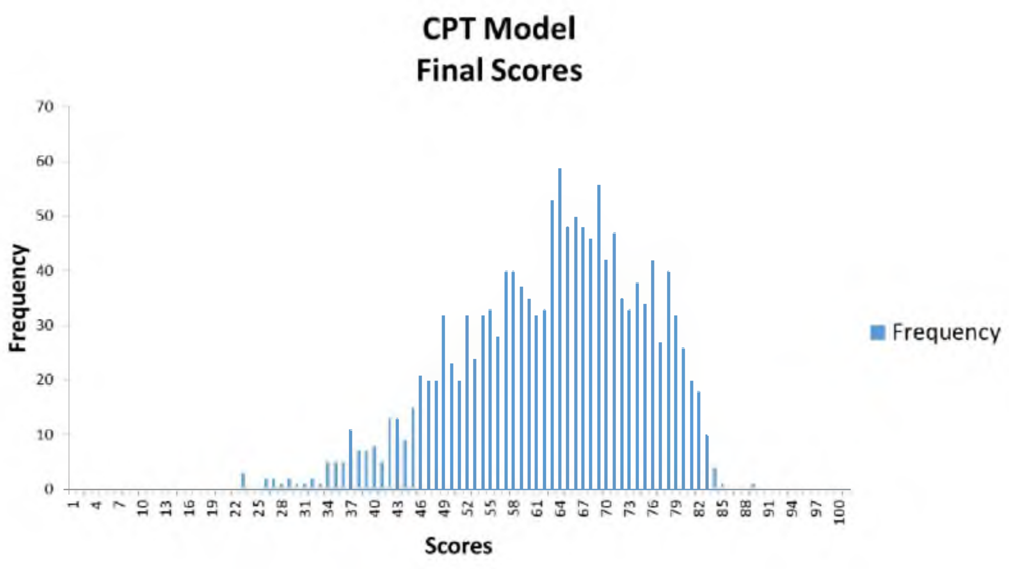


Figure 6.8. Histogram of CPT Final Score Data.

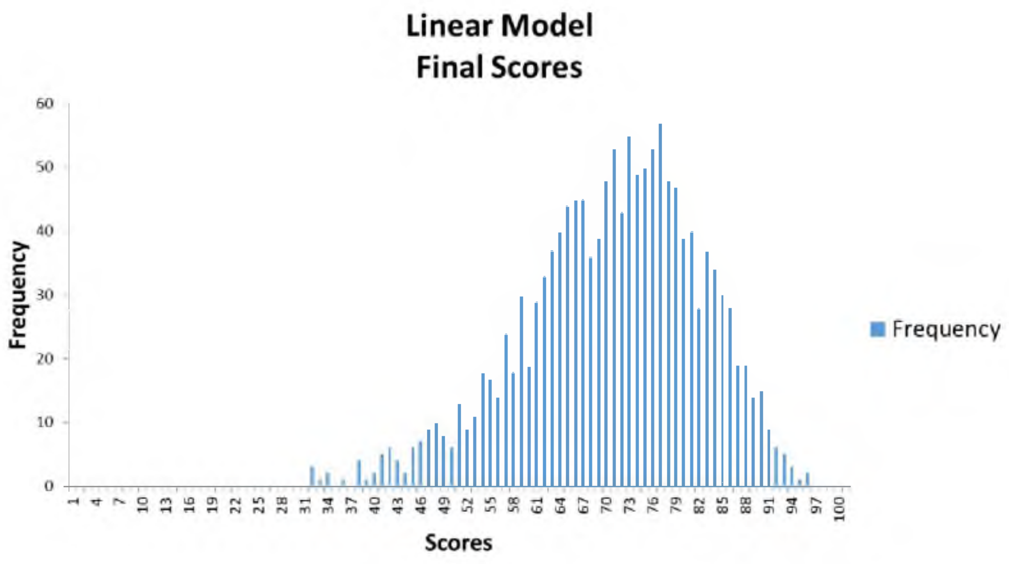


Figure 6.9. Histogram of Linear Final Score Data.

94 is the most frequent score. In this mixed linear sample the most frequently occurring score is 87. This is significantly closer than the other linear model or the CPT model. The linear model has a most frequently occurring score of 77. The CPT model has the most frequently occurring score at 64.

Table 6.1 shows the mean, max, min, and standard deviation of the original final score data and the final score data of each of these three models. Using the histogram and these

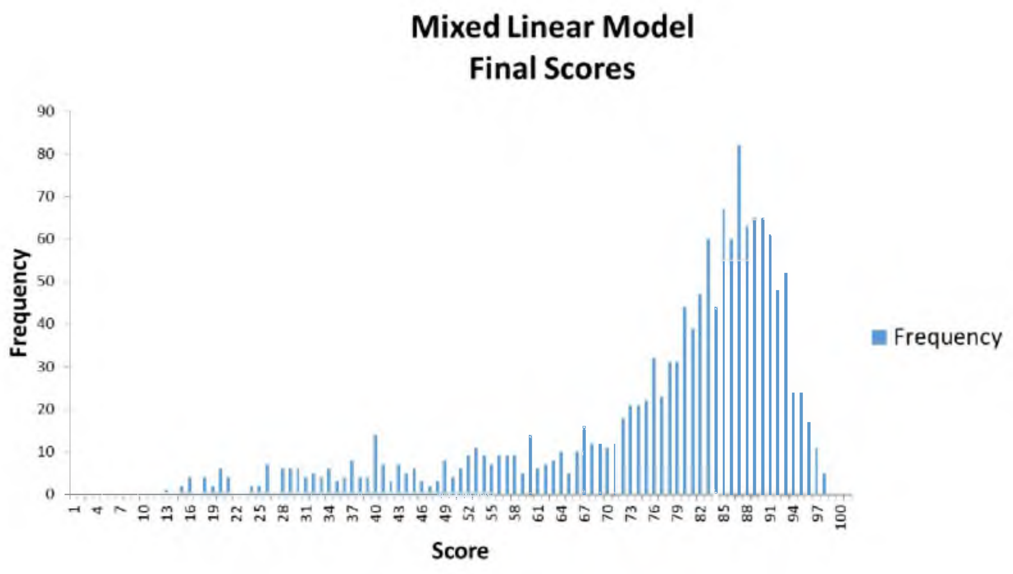


Figure 6.10. Histogram of Mixed Linear Final Score Data.

Table 6.1. Compare Final Scores of Original, CPT, Linear, and Mixed Model Data.

	mean	**max**	**min**	**stdDev**
Original	80.80	98.77	7.59	17.84
CPT	62.36	88.42	22.21	11.78
Linear	70.26	95.62	31.04	11.42
Mixed linear	76.17	97.92	12.82	17.97

score statistics, the mixed linear model produces the closest representation of the original data.

6.3.2 Common Representation

Each model is converted to a discrete histogram. This allows us to sample, predict and compute likelihoods in comparable ways across all models. For example, see Figures 6.11 through 6.13 of computed histograms for a CPT, linear, and mixed model.

6.3.3 Comparing Score Prediction

A valuable piece of functionality that is produced by these models is the ability to predict scores. This ability can be used to make regular recommendations to students and projections for educators. The process for generating these scores is to use leave out one cross validation. This process uses the original actual data and separates them into a

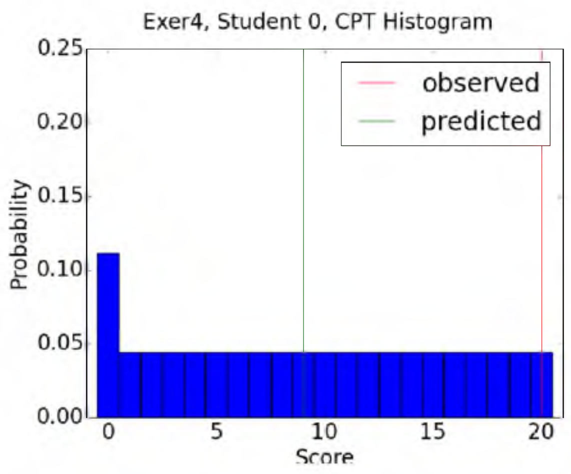


Figure 6.11. Histogram of Probabilities for One Student Score as Sampled by the CPT Model.

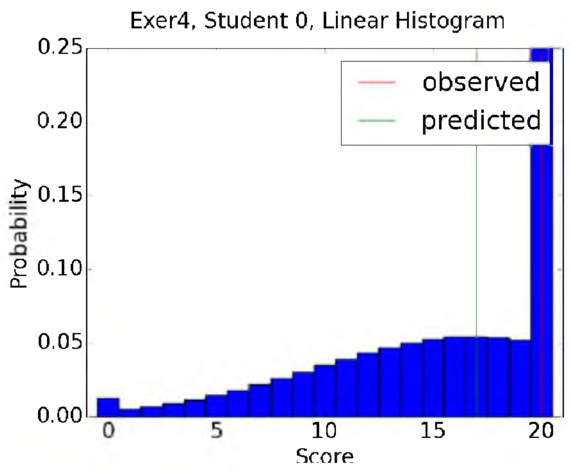


Figure 6.12. Histogram of Probabilities for One Student Score as Sampled by the Linear Model.

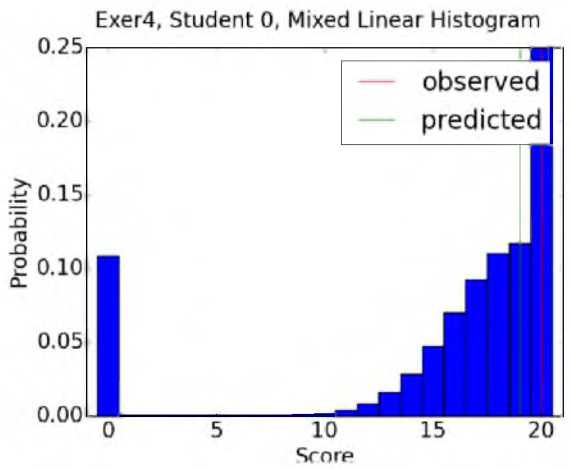


Figure 6.13. Histogram of Probabilities for One Student Score as Sampled by the Mixed Linear Model.

training set and a test set. The training set consists of the scores for all the learning items except the one that is currently being predicted. The test set are the scores that will be generated by the models for the learning item that is being predicted. In this case it is the single score left out of the training set. This is called "leave one out" cross validation. The scores in the training set are used to train the model. Then the trained model generates a set of scores for the single learning item being predicted.

These generated scores are compared to the original scores to identify how accurately the model predicted the score. The accuracy comparison is reported as a percentage. This is the percentage of times the score generated by the model is correct. Correctness is identified by identifying the grade category of the generated score. If the generated score belongs in the same grade category as the actual score, it is correct.

The scores are generated using the sampling process that minimizes the L1 error. Each score in the model's probability distribution is considered and the L1 error for it is computed. These computed L1 errors are compared and the score with the smallest L1 error is selected.

Table 6.2 shows a varying degree of accuracy in making these score predictions. The mixed linear model is clearly the winner, with grade buckets CPTs shortly behind. The CPTs with percentile buckets performs very poorly.

6.3.4 Comparing L1 Errors

L1 errors are used to compare these models. The L1 error is the mean of the absolute value of the errors and is computed using the following formula:

$$E = \frac{1}{N}\sum_{i=1}^{N} |y_i - f(\bar{x}_i)| \tag{6.7}$$

where y_i is the actual value and $f(\bar{x}_i)$ is the result of the function. In our case $\bar{x}_i$ is the vector of actual values associated with the features in the linear equation for the linear models. It is the vector of the actual values of the parents that are inputs to the CPT table for the CPT models. N is the number of scores being generated for this learning item.

Table 6.3 shows the L1 error for each of the models. This table is sorted in order by the value of the L1 error.

6.3.5 Comparing L2 Errors

L2 errors are used to compare these models. The L2 error is the square root of the mean of the square of the errors and is computed using the following formula:

$$E = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - f\bar{x}_i)^2} \tag{6.8}$$

Table 6.2. Compare Grade Accuracy Between Models.

Model Type	Dependencies	Grade Accuracy
Mixed linear	Precedes Three	77%
Mixed linear	Precedes Three / Pruned Prereqs	77%
Mixed linear	Precedes Five	76%
Mixed linear	Precedes Two	76%
Mixed linear	Precedes Nine	76%
Mixed linear	Precedes One	75%
Mixed linear	Precedes One / Pruned Prereqs	75%
Mixed linear	Precedes Thirteen	75%
CPT Grade Buckets	Precedes One	75%
CPT Grade Buckets	Precedes One / Pruned Prereqs	75%
CPT Grade Buckets	Precedes Two	75%
Mixed linear	Precedes	73%
CPT Grade Buckets	Empty	73%
CPT Grade Buckets	Pruned Prereqs	73%
CPT Grade Buckets	Prereqs	73%
CPT Grade Buckets	Precedes Three	73%
Mixed linear	Pruned Prereqs	72%
Mixed linear	Prereqs	72%
Mixed linear	Empty	72%
Linear	Precedes Three	72%
Linear	Precedes Three / Pruned Prereqs	72%
Linear	Precedes Five	72%
Linear	Precedes Two	72%
Linear	Precedes One	72%
Linear	Precedes One / Pruned Prereqs	72%
Linear	Precedes Nine	72%
Linear	Precedes Thirteen	72%
Linear	Precedes	70%
Linear	Empty	67%
Linear	Prereqs	67%
Linear	Pruned Prereqs	67%
CPT Percentile Buckets	Precedes Three	32%
CPT Percentile Buckets	Precedes One	32%
CPT Percentile Buckets	Precedes One / Pruned Prereqs	32%
CPT Percentile Buckets	Precedes Two	31%
CPT Percentile Buckets	Empty	31%
CPT Percentile Buckets	Pruned Prereqs	31%
CPT Percentile Buckets	Prereqs	31%

Table 6.3. Compare L1 Errors Between Models.

Model Type	Dependencies	L1 Error
Mixed linear	PrecedesFive	4.8
Mixed linear	PrecedesThree	4.9
Mixed linear	PrecedesThreePrunedPrereqs	4.9
Mixed linear	PrecedesNine	5
Mixed linear	PrecedesTwo	5
Mixed linear	PrecedesThirteen	5.1
CPT Grade Buckets	PrecedesTwo	5.4
CPT Grade Buckets	PrecedesThree	5.5
Linear	PrecedesThirteen	5.6
Mixed linear	Precedes	5.6
Linear	PrecedesNine	5.6
Linear	PrecedesFive	5.6
Linear	Precedes	5.8
Mixed linear	PrecedesOne	5.8
Mixed linear	PrecedesOnePrunedPrereqs	5.8
Linear	PrecedesThree	5.9
Linear	PrecedesThreePrunedPrereqs	5.9
CPT Grade Buckets	PrecedesOne	6.1
CPT Grade Buckets	PrecedesOnePrunedPrereqs	6.1
Linear	PrecedesTwo	6.2
Mixed linear	PrunedPrereqs	6.2
Mixed linear	Prereqs	6.2
Mixed linear	Empty	6.2
CPT Grade Buckets	Empty	6.3
CPT Grade Buckets	PrunedPrereqs	6.3
CPT Grade Buckets	Prereqs	6.3
Linear	PrecedesOne	6.7
Linear	PrecedesOnePrunedPrereqs	6.7
Linear	Empty	7.6
Linear	Prereqs	7.6
Linear	PrunedPrereqs	7.6
CPT Percentile Buckets	PrecedesTwo	9.5
CPT Percentile Buckets	PrecedesThree	9.5
CPT Percentile Buckets	PrecedesOne	10
CPT Percentile Buckets	PrecedesOnePrunedPrereqs	10
CPT Percentile Buckets	Empty	10.2
CPT Percentile Buckets	PrunedPrereqs	10.2
CPT Percentile Buckets	Prereqs	10.2

where y_i is the actual value and $f\bar{x_i}$ is the result of the function. In our case $\bar{x_i}$ is the vector of actual values associated with the features in the linear equation for the linear models. It is the vector of the actual values of the parents that are inputs to the CPT table for the CPT models. N is the number of scores being generated for this learning item.

Table 6.4 shows each of the models and their L2 error. This table is sorted in order by the value of the L2 error.

6.3.6 Comparing the Log-Likelihood

The log-likelihood of each model is compared in the table below Table 6.5. The likelihood of a set of values, θ, given the outcome x is the probability of that observed outcome given the parameter values. That can be written as

$$L(\theta|x) = P(x|\theta) \tag{6.9}$$

The likelihood function is defined differently for discrete probability distributions and continuous probability distributions. The CPT models are discrete probability distributions. Linear models are continuous probability distributions; however they are converted to a discrete representation for comparison purposes and use the discrete likelihood function that can be written in the following ways:

$$P_\theta(X = x) \text{ or } P(X = x|\theta) \text{ or } P(X = x; \theta) \tag{6.10}$$

where x is the outcome of the probability model and θ is the set of observed values, in our case the actual values of the parents that are inputs to the CPT table for the CPT models or associated with the features in the linear equation for the linear models.

The natural logarithm of this likelihood function (see Equation 6.10) is used as the reported measure. This is used because it is more convenient to work with. The logarithm is a monotonically increasing function and therefore achieves its maximum value at the same points as the function itself. Therefore, the log-likelihood can be used in place of the likelihood when estimating maximum likelihood.

6.3.7 Comparing Probability Histograms

The system can generate histograms of the probability distribution for a specific student on a specific learning item. Figure 6.14 contains the probability distributions for two different students on a class activity. The two students are distinguished by their final grade. There is one each for grades A and B. There are three different models for each student: CPT, linear, and mixed linear.

Table 6.4. Compare L2 Errors Between Models.

Model Type	Dependencies	L2 Error
Linear	PrecedesFive	0.1411
Linear	PrecedesThirteen	0.1419
Linear	PrecedesNine	0.1423
Linear	Precedes	0.1466
Mixed linear	PrecedesFive	0.1468
Linear	PrecedesThree	0.1478
Linear	PrecedesThreePrunedPrereqs	0.1478
Mixed linear	PrecedesThree	0.1492
Mixed linear	PrecedesThreePrunedPrereqs	0.1492
Mixed linear	PrecedesNine	0.1523
Linear	PrecedesTwo	0.1536
Mixed linear	PrecedesTwo	0.1549
Mixed linear	PrecedesThirteen	0.1569
CPT Grade Buckets	PrecedesThree	0.1572
CPT Grade Buckets	PrecedesTwo	0.1583
Linear	PrecedesOne	0.1638
Linear	PrecedesOnePrunedPrereqs	0.1638
Mixed linear	Precedes	0.1675
Mixed linear	PrecedesOne	0.1711
Mixed linear	PrecedesOnePrunedPrereqs	0.1711
CPT Grade Buckets	PrecedesOne	0.1767
CPT Grade Buckets	PrecedesOnePrunedPrereqs	0.1767
Linear	Empty	0.1798
Linear	Prereqs	0.1798
Linear	PrunedPrereqs	0.1798
CPT Percentile Buckets	PrecedesThree	0.1814
CPT Percentile Buckets	PrecedesTwo	0.1816
Mixed linear	PrunedPrereqs	0.1836
Mixed linear	Prereqs	0.1836
Mixed linear	Empty	0.1836
CPT Grade Buckets	Empty	0.1878
CPT Grade Buckets	PrunedPrereqs	0.1878
CPT Grade Buckets	Prereqs	0.1878
CPT Percentile Buckets	PrecedesOne	0.1947
CPT Percentile Buckets	PrecedesOnePrunedPrereqs	0.1947
CPT Percentile Buckets	Empty	0.2058
CPT Percentile Buckets	PrunedPrereqs	0.2058
CPT Percentile Buckets	Prereqs	0.2058

Table 6.5. Compare Log-likelihood Between Models.

Model Type	**Dependencies**	**Log Likelihood**
Mixed linear	PrecedesThree	-8663
Mixed linear	PrecedesThreePrunedPrereqs	-8663
Mixed linear	PrecedesFive	-8685
Mixed linear	PrecedesTwo	-8711
Mixed linear	PrecedesOne	-8813
Mixed linear	PrecedesOnePrunedPrereqs	-8813
Mixed linear	PrecedesNine	-8894
Mixed linear	PrecedesThirteen	-9083
Mixed linear	PrunedPrereqs	-9093
Mixed linear	Prereqs	-9093
Mixed linear	Empty	-9093
Mixed linear	Precedes	-9960
CPT Grade Buckets	PrecedesOne	-11109
CPT Grade Buckets	PrecedesOnePrunedPrereqs	-11109
CPT Grade Buckets	PrecedesTwo	-11191
CPT Grade Buckets	Empty	-11261
CPT Grade Buckets	PrunedPrereqs	-11261
CPT Grade Buckets	Prereqs	-11261
CPT Grade Buckets	PrecedesThree	-11408
Linear	PrecedesThree	-11888
Linear	PrecedesThreePrunedPrereqs	-11888
Linear	PrecedesFive	-11910
Linear	PrecedesTwo	-11931
Linear	PrecedesOne	-12006
Linear	PrecedesOnePrunedPrereqs	-12006
Linear	PrecedesNine	-12100
Linear	Empty	-12233
Linear	Prereqs	-12233
Linear	PrunedPrereqs	-12233
Linear	PrecedesThirteen	-12258
Linear	Precedes	-13061
CPT Percentile Buckets	PrecedesTwo	-16093
CPT Percentile Buckets	PrecedesThree	-16108
CPT Percentile Buckets	PrecedesOne	-16147
CPT Percentile Buckets	PrecedesOnePrunedPrereqs	-16147
CPT Percentile Buckets	Empty	-16405
CPT Percentile Buckets	PrunedPrereqs	-16405
CPT Percentile Buckets	Prereqs	-16405

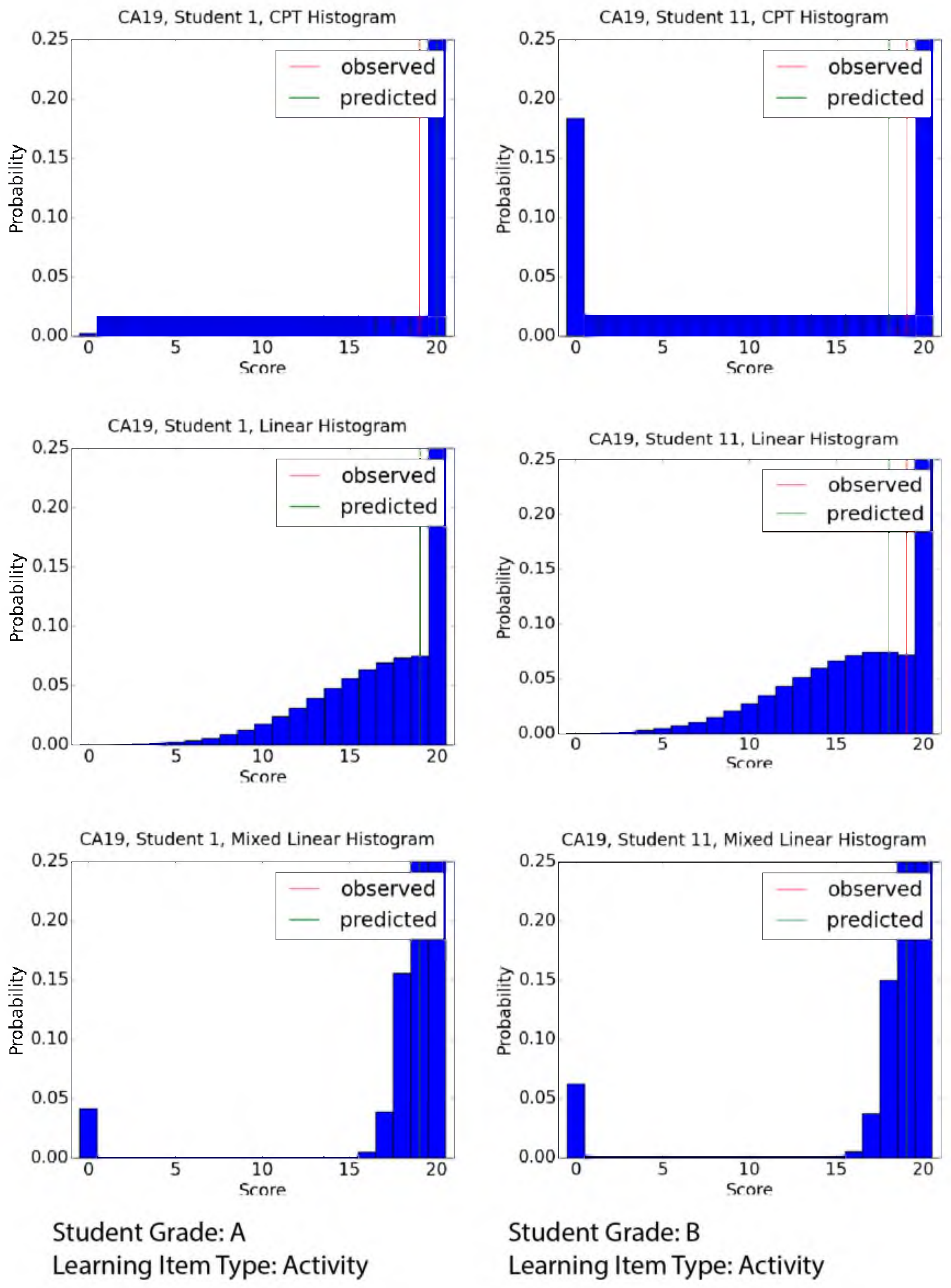


Figure 6.14. Probability Distributions of Sample Students on a Class Activity.

The CPT model has five grade buckets, including a zero bucket. It uses the immediately precedes/pruned prerequisites combination model. The linear and mixed linear models use all the preceding items in the feature list. Comparing these histograms allows us to see how the distributions vary based on the individual student and which model is building the distribution.

6.3.8 Summary

To do comparisons, 38 variations of the Bayesian network and linear models were created. These variations were then compared in a variety of ways. Although there were some differences in the ordering of the models, depending on which comparison was being used, several of the same models were frequently ranked at the top.

When comparing the histograms of the final score data to the histogram of the original data, the mixed linear model was clearly more representative of the original data then either of the other versions. This mixed linear model produced a mean that is within 4.63% of the original data and a standard deviation only .13% different than the original data.

When comparing models for accuracy in predicting scores, 28 of the 38 models got 70% or higher. Again, several of the mixed linear models were at the very top, with a grade accuracy of 77%. Right behind the mixed linear models are the CPT models that used grade buckets. Three of these versions had a grade accuracy of 75%. There was only an 11% variance in the top 21 models, with all the variations of the mixed linear, CPT using grade buckets, and linear models having a grade accuracy from 67% to 77%. There was a distinct separation between these models and the CPT models using percentile buckets. The CPT models using percentile buckets had a score accuracy of 31% to 32%, significantly lower than the other models.

The comparisons of L1 errors, L2 errors, and highest likelihood values produced different orderings of the models. The L1 errors ranked half of the mixed linear models at the top, followed by a couple of the CPTs with grade buckets. The L2 errors put the linear models at the top. This is the only comparison that ranked the linear models at the top. The likelihood comparison is the only one that separated the models exactly into their basic types. The comparison of the likelihood scores put all the variations of the mixed linear model at the top, followed by all the variations of the CPT with grade buckets models. These two were followed by all the variations of the linear models and then all the variations of the CPT with percentile buckets models.

For the L1 errors, L2 errors, highest likelihood, and score accuracy values, an analysis was done of the different ratings for individual assignment groups. The sample CS0 course has four categories of assignment groups: exams, class activities, exercises, and homework. These are based on the assignment groups in Canvas. All three of the rankings found the exams ranked the lowest, followed by homework. The class activities and exercises ranked first or second, depending on what was being compared. These rankings were a direct correlation of the points possible and standard deviations of these categories. The exams were 100 points possible each and had a standard deviation of 22.7. Homework varied in the points possible, with an average of 55 points possible and a standard deviation of 10.4. Exercises and class activities were each worth 20 points, with standard deviations of 7.7 and 6.8, respectively. L1 errors, L2 errors, log likelihood, and score accuracy all measure how close to the actual score the predicted score is. It is certainly more difficult to get close to the correct score when there is a significantly larger range of values possible. The comparisons of these rankings based on assignment group categories can be found in Appendix G.

By doing several variations of each general model, a comparison can be done between the different variations. The general models are CPT with grade buckets, CPT with percentile buckets, linear, and mixed linear. For both the CPT models there were seven variations. The linear and mixed linear each had 12 variations. For the CPTs with grade buckets, identifying which variations are preferred is unclear as the precedes two and precedes three ranked first and second in this general model for the L1 and L2 errors but ranked last for log likelihood and grade accuracy. While the precedes one and precedes one/pruned prerequisite variations were ranked lowest on L1 and L2 errors and highest for log likelihood and grade accuracy. For the linear model there was no consistent pattern between the variations. There was only one variation that ranked in the top three of each comparison, the precedes five. It ranked third in all the comparisons except L2 errors, in which it was ranked first. For the mixed linear general model, three variations were at the top of all the rankings, precedes five, precedes three, and precedes three/pruned prerequisites. Although which of these were ranked the highest varied, these were the top three of the mixed linear models in every ranking. For the CPT with percentile buckets model the precedes three was ranked either first or second in all the comparisons. Precedes two was top ranked in L1 errors and log likelihood, it was second in L2 errors, but dropped to fourth in grade prediction accuracy. The precedes one variation was consistently ranked third, except in the grade accuracy it was ranked second.

Comparing the probability histograms is a way to visually see how each of the models is able to create very different probability distributions based on the student and the learning item. Since these variations exist in the student population and show up in the actual data, this provided some validation that the underlying models are well designed.

Considering all the comparisons, the mixed linear model ranks the highest, followed by the CPT with grade buckets model. The differences in the scores for many of the comparisons were very small between these two models, while the difference between these two and the lowest ranked model, CPT with percentile buckets, was large.

6.3.9 Evaluation of Behavior Modeling and Bayesian Analysis

In this section the evaluation criteria for Section 1.7.4 is considered. Several Bayesian inference networks have been created based on combinations of the *precedes* and *prerequisite* relations from the course maps. These relations are used to determine parents of each node in the CPT tables. The values in the Bayesian networks are based on the actual data from the sample courses. In addition, several linear probability models were created.

Sampling of each of the Bayesian networks and linear probability models was used to produce 1430 sample sets of data for each model. Final scores were computed for each sample set. These final scores were compared to the final scores of the existing course using a variety of measures, including score prediction (see Section 6.3.3), L1 errors (see Section 6.3.4), L2 errors (see Section 6.3.5), log-likelihood (see Section 6.3.6), and final grade distribution (see Section 6.3.1).

CHAPTER 7

TRACKING MASTERY OF LEARNING ITEMS

Given a course map and a method to accurately assess a student's mastery level of a specific learning item (e.g., assign grades to graded learning items), we now describe a technical approach to model and track student mastery of the learning items during a traversal of the course map. That is, at each step in the traversal a mastery level can be determined for each learning item for the particular student. First, we define a linear learning model, which provides a straightforward way to estimate the student's mastery of each learning item. Then, we demonstrate how to estimate learning coefficients (variables related to the student's abilities and the difficulty of the material) using a nonlinear model. Such information is crucial in defining an interactive strategy to facilitate student learning.

7.1 Linear Learning Model with Mastery

Let $\mathcal{M} = (\mathcal{N}, \mathcal{E})$ be a course map with nodes $\mathcal{N}$ and edges $\mathcal{E}$. Let $\mathcal{C} = (\mathcal{L}, \mathcal{R})$ be the corresponding class map for $\mathcal{M}$, where $\mathcal{L}$ is the set of learning items and $\mathcal{R}$ is the set of relations on $\mathcal{L}$. Construct a vector, $x = [x_1, x_2, ..., x_n]^T$, where x_i represents the mastery level of the student for learning item $\mathcal{L}_i$; we have x_i is in the range $[0, 1]$, where $x_i = 0$ means no mastery and $x_i = 1$ means full mastery.

Furthermore, let x^t represent the mastery state at step t. If the student knows nothing at all at the start, then $x^0 = 0$; however, if the student has some background knowledge concerning a learning item $\mathcal{L}_i$, then x_i^0 can be set to the appropriate amount.

We assume that the *prerequisite* relation entails some amount of causal relation between the mastery of the respective learning items. I.e., if A *prerequisite* B, then the probability that the student masters learning item B depends on the mastery of learning item A. We further propose as a starting point a linear function to describe the dynamic learning process (also called the *transition model*):

$$x^{t+1} = Ax^t + Bu_t + \epsilon$$

where the matrix A describes the relation between learning item mastery and the matrix B describes the impact of the control variable, u_t, at time t. The control variable describes what learning items the student works on at time t as well as the amount of work, while the first term in the transition model (i.e., Ax^t) characterizes the learning impact of the mastery of the previous learning items.

The transition model also includes a characterization of the noise of the learning process by means of the random variable $\epsilon \sim \mathcal{N}(0, \sigma_{p_i}))$, where σ_{p_i} is the variance in the learning process for each individual learning item, $\mathcal{L}_i$. The covariance matrix for the full vector, x, is called R.

It is also necessary to have a model of the observation process. Mastery will be measured by means of graded learning items, and the *measurement model* is

$$z^t = Cx^t + \delta$$

where C is a $k \times n$ matrix providing observations of the graded learning items. Furthermore, $\delta \sim \mathcal{N}(0, \sigma_z)$ where σ_z characterizes the noise in the measurement (testing) method. For the full z vector, this is given by the covariance matrix Q.

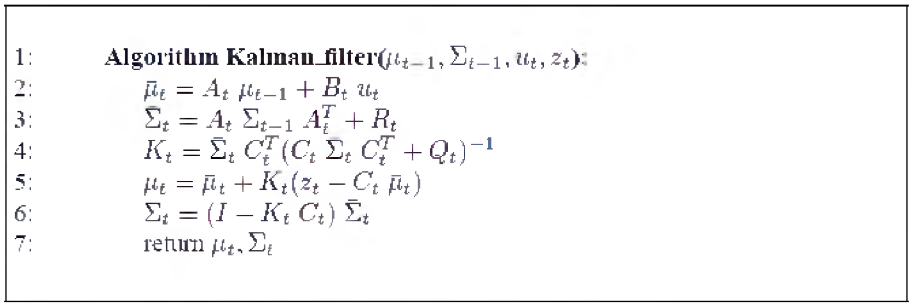
Given these transition and measurement models, it is appropriate to use a Kalman Filter [3] to track the mastery level during a traversal. Given such a model, control vectors can be selected to maximize mastery of the learning items while minimizing needless repetition and effort. The Kalman Filter algorithm is shown in Figure 7.1.

In order to exploit this dynamic Bayesian network approach, it is necessary to specify the matrices A, B, and C; the covariance matrices can be set based on actual class data or based on data generated by the artificial agents previously described. As a first cut at a learning material mastery model, let

$$x = \begin{bmatrix} 1 & 0 & \dots & 0 \\ a_{21} & a_{22} & 0 & \dots \\ \dots & & & \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} + \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & & & \\ 0 & 0 & \dots & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ u_n \end{bmatrix} \tag{7.1}$$

where we assume u_t has one element set to 1 (or some amount of effort between 0 and 1), meaning that only one learning item is worked on at a time. For A, $a_{i,j} = 0$ for $j > i$; for $j \leq i$, $a_{i,j}$ is set to 0 if $\neg(x_i \textit{ prerequisite } x_j)$; otherwise, $a_{i,j} = \frac{1}{n_{ij}}$, where n_{ij} is one plus the number of learning items that are prerequisites for x_j.

Consider the simple class map, $\mathcal{C}$, shown in Figure 7.2.



```
1:  Algorithm Kalman_filter($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):
2:      $\bar{\mu}_t = A_t\,\mu_{t-1} + B_t\,u_t$
3:      $\bar{\Sigma}_t = A_t\,\Sigma_{t-1}\,A_t^T + R_t$
4:      $K_t = \bar{\Sigma}_t\,C_t^T(C_t\,\bar{\Sigma}_t\,C_t^T + Q_t)^{-1}$
5:      $\mu_t = \bar{\mu}_t + K_t(z_t - C_t\,\bar{\mu}_t)$
6:      $\Sigma_t = (I - K_t\,C_t)\,\bar{\Sigma}_t$
7:      return $\mu_t, \Sigma_t$
```

Figure 7.1. Kalman Filter Algorithm as shown in [3].

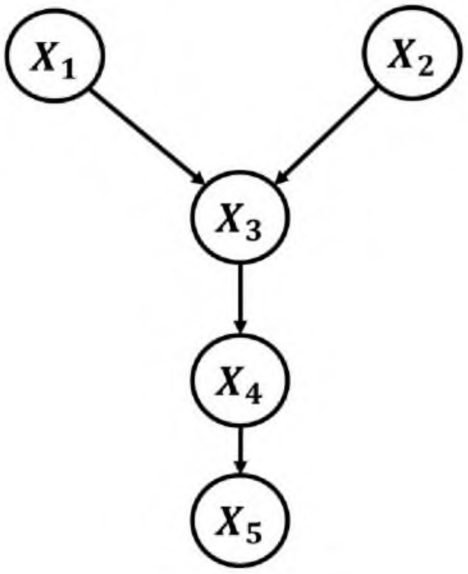


Figure 7.2. Class Map for Simple 5-Learning Item Class.

Then

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{bmatrix}$$

and

$$x^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Then a simple traversal such as $[1, 2, 3, 4, 5]$ yields the mastery level estimates shown in Figure 7.3. This allows us to track in more detail how a student is moving through the

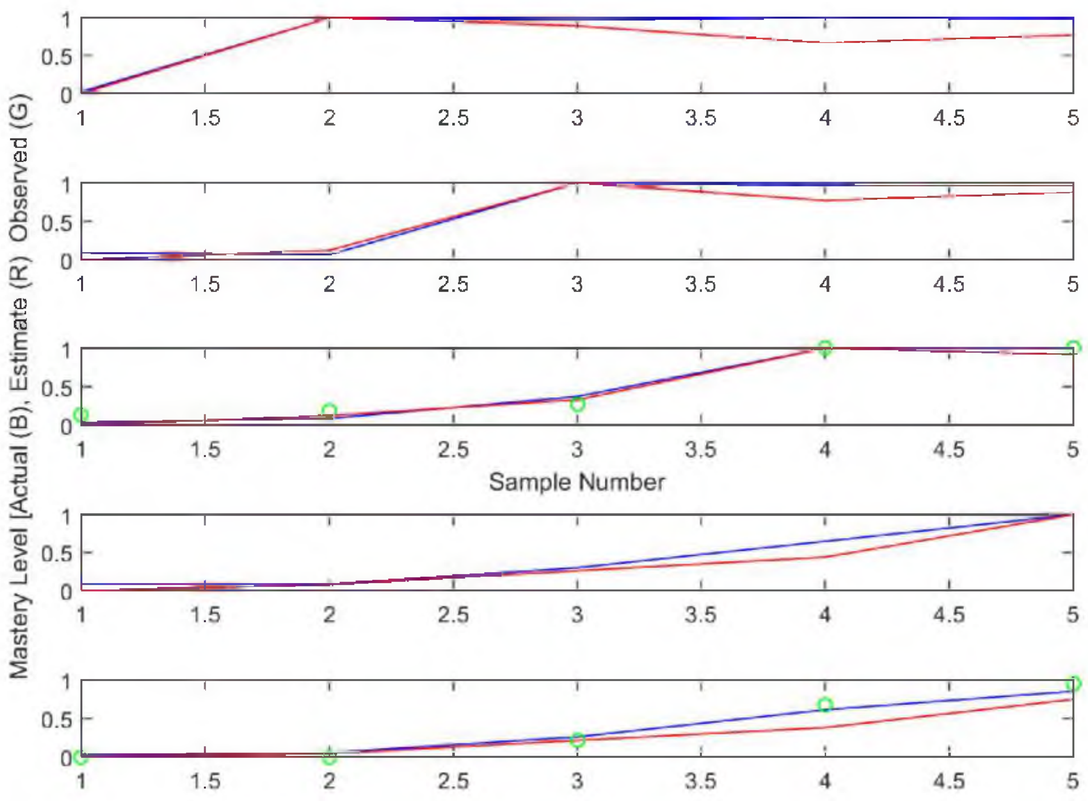


Figure 7.3. Mastery Estimates for Five Learning Item Traversal.

learning material. By including the concept of mastery, we introduce a more in-depth view of how well the student is mastering the learning items. Currently students move through learning items based on which day it is. Mastery could provide a different gauge by which to determine when and in what order a student moves through the material. Also, because this process can include ungraded learning items, we are able to get some perspective on items without scores.

7.2 Non-Linear Learning Model with Mastery

We now introduce a more complex learning framework which utilizes the direct interaction of an artificial student agent. A learning item, $l \in \mathcal{L}$, also has an associated difficulty level, $d(l) \in \Re^{\geq 0}$, where $\Re^{\geq 0}$ is the non-negative real numbers. The class graph imposes traversal constraints on a student; namely, every prerequisite of a node l must be mastered to an acceptable level before node l can be mastered.

At each time step, the agent specifies how much time of the total alloted is to be spent on each accessible learning item; this constitutes an *action*. Agents may implement different learning tactics and their respective learning performance traces may then be compared.

For example, an agent that spends equal time on newly available nodes or equal time on all previous nodes if there are no new nodes, is specified as:

```
On input: open (accessible) nodes
Local: visited nodes
On output: relative percent of time on all nodes

new_nodes <-- open and not visited

if no new_nodes
  if open_nodes not empty
    action <-- open/|open|
  end
else
  action <-- new_nodes/|new_nodes|
end
```

To demonstrate the performance of an artificial student agent on the class map, $\mathcal{C}$ (shown in Figure 7.4), suppose the agent has characteristics $W = 3$, $I = 3$, $B = 0$, and $D = 1$. Then, a learning curve plot for $Agent_1$ on the 10-node class graph is shown in Figure 7.5 (left), while an $Agent_2$ with $W = 1$, $I = 1$, $B = 0$ and $D = 3$ is shown on the right side of Figure 7.5 (right). Note that $Agent_1$ has achieved almost perfect mastery of all ten learning items by step 80, whereas $Agent_2$ has only mastered a few items in the same time.

Note that learning curves are also a function of the learning tactics of the agent. Suppose that $Agent_1$ modifies its approach so as to focus on individual items until they are mastered before moving on to the next available item. The resulting learning curve is shown in Figure 7.6, which illustrates that items are mastered sequentially, and that it takes longer to learn all ten items than the equal time strategy. [Note that this may also provide evidence that a linear organization of the course material slows learning!]

7.2.1 A Learning Model

In 7.1, we defined the notion of *mastery* of a learning item as a random variable ranging from 0 to 1 and demonstrated the use of a linear learning model combined with a Kalman Filter to obtain an optimal estimate of student mastery of the learning items in a course graph based on combining the model prediction with a measurement (i.e., a grade) correction.

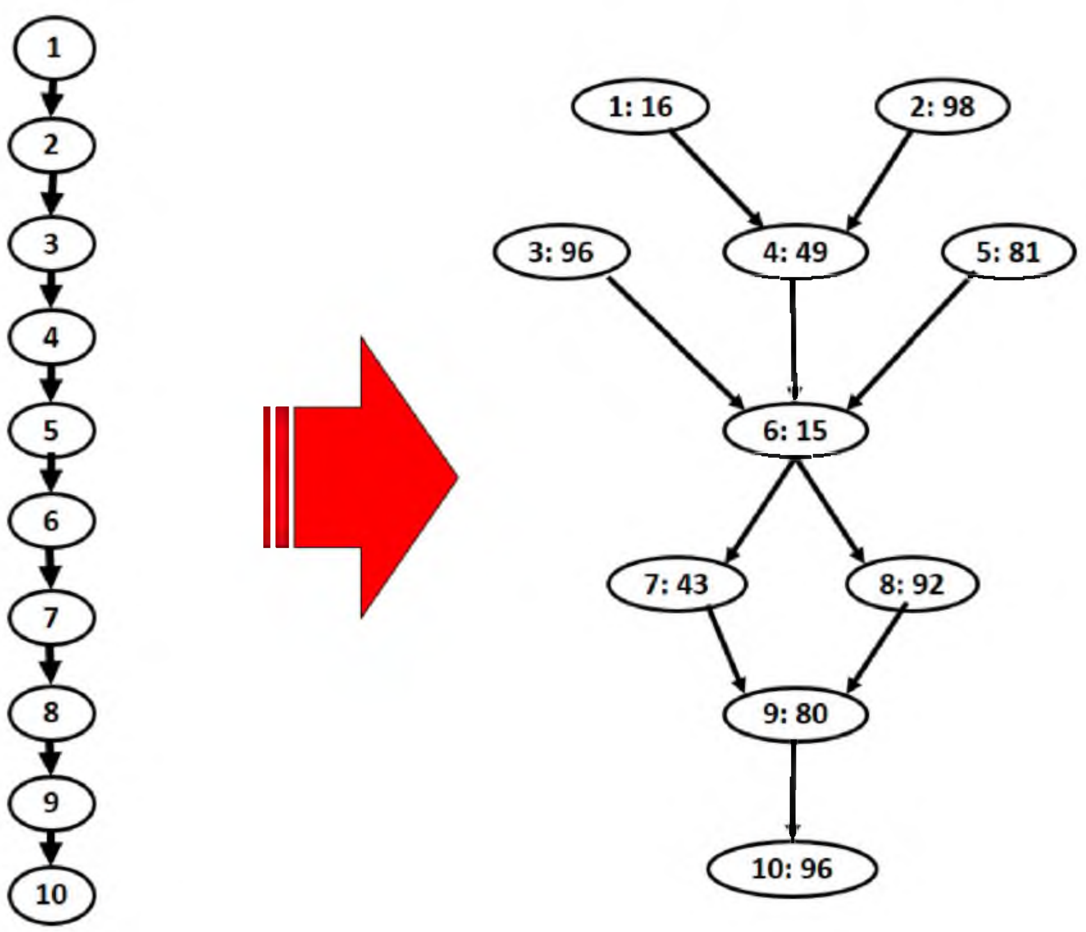


Figure 7.4. Example of Converting a Standard Synchronous Class to an On-line Class With Computed Values For Difficulty Level (the first number is the Learning Item number, the second is the Difficulty).

In this section we propose a more refined nonlinear student learning model which includes a parameter – the *learning coefficient* – and compare three ways to estimate it: (1) direct inverse, (2) iterative least squares, and (3) the Extended Kalman Filter. This is called either *model parameter calibration* or *parameter estimation.*

The estimation method is based on the use of a class graph that describes the organization of the learning material, a set of artificial student agents with an associated learning model, and a mechanism for the class graph traversal. A wide variety of user models have been proposed for interactive learning environments, e.g., see [26, 111, 122, 123]. We have opted to use a more basic and general model of learning as described in [124]:

$$x_i^{t+1} = M - (M - x_i^t)e^{-k_i s_i^t} + \epsilon \tag{7.2}$$

where x_i^t is the mastery level of learning item i at time t, M is the maximal mastery level

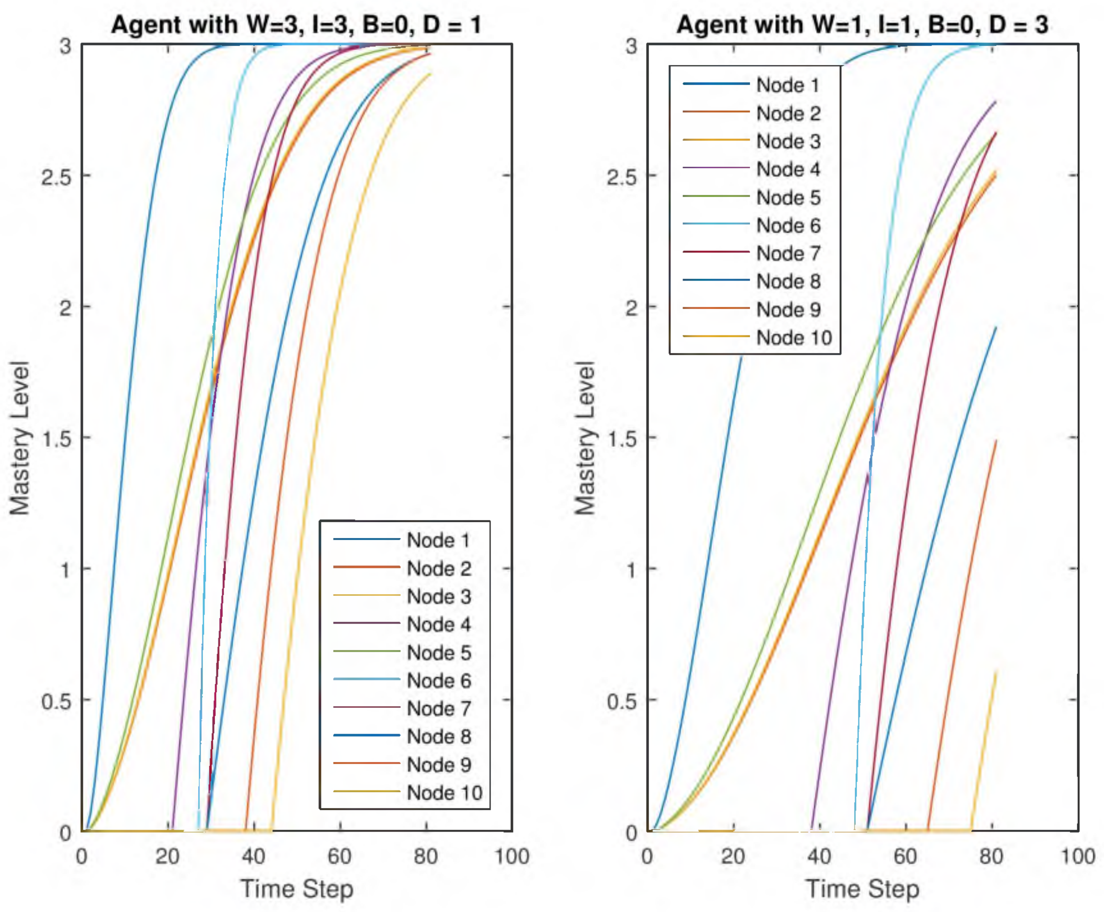


Figure 7.5. Learning Curves for Agents with Different Abilities.

(which we set to 3 in these experiments), k_i is the learning coefficient for the student on learning item i, s_i^t is the cumulative time spent on learning item i, and $\epsilon \sim \mathcal{N}(0, \sigma^2)$, where σ^2 is the variance in the learning model process. The learning coefficient k_i is a function of the agent and the learning item:

$$k_i = \frac{\frac{W+I+B}{D}}{\alpha_i} \tag{7.3}$$

where α_i is the difficulty of learning item i (and is in the range [0,100] in these experiments).

The learning coefficient provides a way to compute the relative difficulty of each learning item. This can provide insight to help the instructor better balance the student work load.

In our initial use the formula, 7.2 is linear. However, using this formula allows us to work with situations that will produce a nonlinear function, such as if the learning coefficient is itself included in the state or if learning item mastery levels are multiplied by each other.

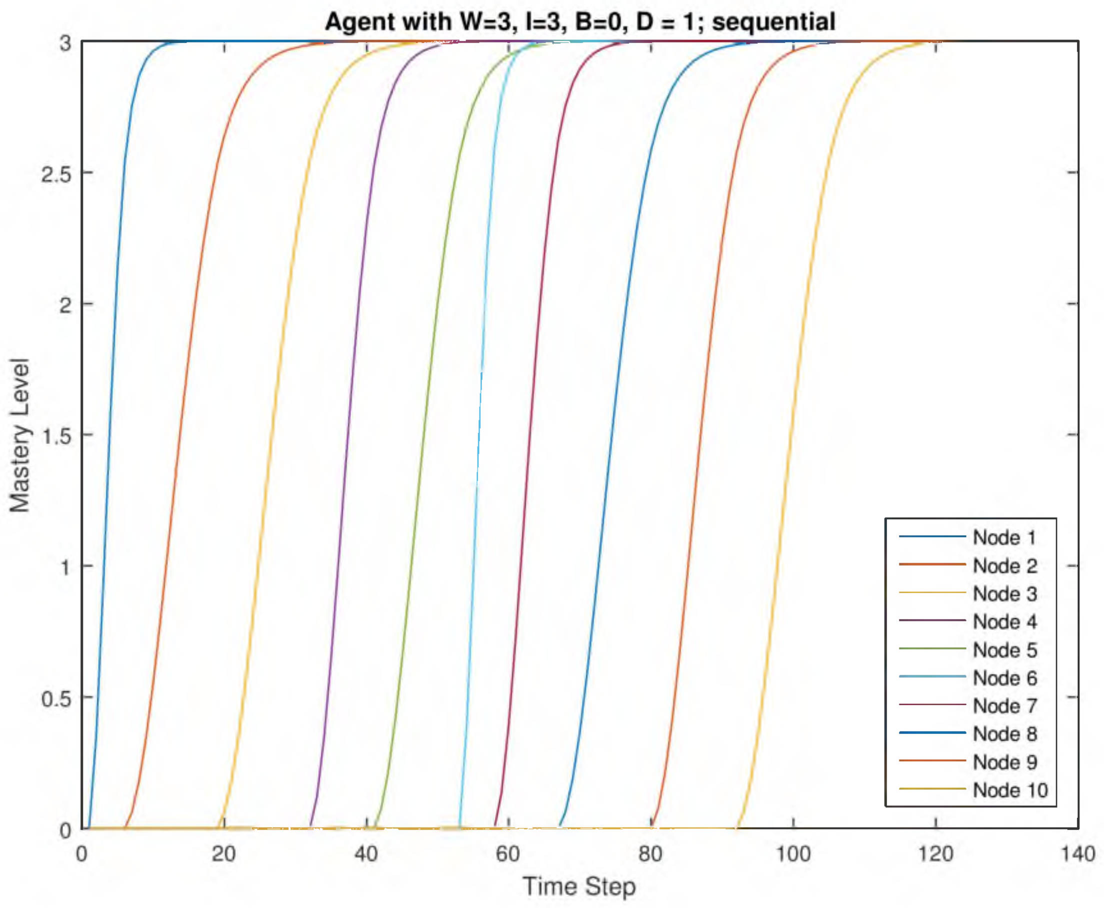


Figure 7.6. Learning Curves for Agent with Two Different Learning Tactics.

7.2.1.1 Learning Model Parameter Calibration

Given the ENABLE framework, it is possible to use the information from a student's interaction with the class map to estimate the particular learning coefficient for each learning item. This allows active modification of the graph traversal in order to facilitate learning by the student. It also makes it possible to estimate the difficulty of each learning item (by using ratios of learning coefficients) so that the instructor is better informed about the nature of the presentation of the learning items (see Figure 7.4).

7.2.1.2 Direct Inverse Method

Given the learning model in Equation 7.2, then for every step at a learning item in which learning takes place (i.e., $x_i^{t+1} > x_i^t$), and which had time allocated to the item (i.e., $s_i^{t+1} > s_i^t$), then k_i can be found as:

$$k_i = \frac{-ln(\frac{-(x_i^{t+1}-M)}{(M-x_i^t)})}{s_i^t} \tag{7.4}$$

Since there is noise in the learning process, the following algorithm is applied:

```
for every learning item i
  for every step t that meets conditions
    calculate k_i,t
  end
  k_i estimate is median of k_i,t
end
```

For $\sigma^2 = 0.001$, with $Agent_1$ and class map $\mathcal{C}$, the learning curves for one trial are shown in Figure 7.7. The actual learning coefficients are [0.1250, 0.0204, 0.0208, 0.0408, 0.0247, 0.1333, 0.0465, 0.0217, 0.0250, 0.0208] the estimates found using the inverse method are [0.1257, 0.0205, 0.0210, 0.0409, 0.0252, 0.1340, 0.0466, 0.0217, 0.0249, 0.0208], and the RMSE is 0.0123. Figure 7.8 shows the RMSE on this for σ^2 ranging from 10^{-5} to 1, with 10 trial samples per variance value.

7.2.1.3 Least Squares Method

Least squares is a standard method for the determination of a best solution to an over-constrained problem (see [125] for an introduction). We follow here the method described in [126]. The least squares estimate is arrived at by iterating:

$$k_i^{t+1} = k_i^t + (J^T J)^{-1} J^T (Y - V|_{k_i^t}) \tag{7.5}$$

where k_i^t is the estimate of the actual learning coefficient k^* at step t, J is the Jacobian of the learning process model, Y is the observed mastery values from the trace of a student traversal of the class graph, and $V|_{k_i^t}$ is the predicted mastery values for a student traversal of the class graph using the current learning coefficient estimate. Note that for the full graph, k^* is a vector, and $|k^*| = n$, where n is the number of nodes in the class graph.

Since the process model is

$$f(k, s, x) = M - (M - x)e^{-ks} \tag{7.6}$$

the Jacobian is

$$J = \frac{\partial f}{\partial k} = s(M - x)e^{-ks} \tag{7.7}$$

The least squares iteration is continued until convergence criteria are satisfied. Figure 7.8 shows the RMSE values achieved by the least squares method.

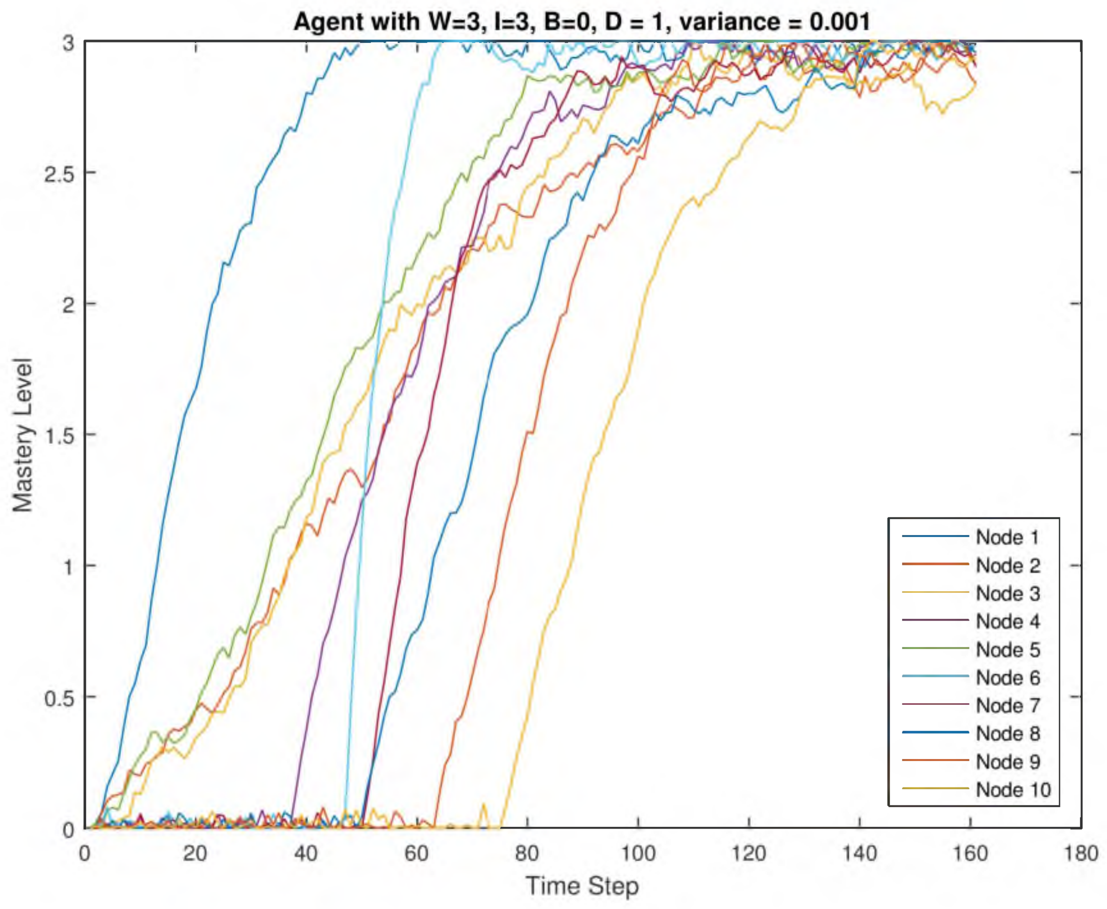


Figure 7.7. Learning Curves for Trial (with noise).

7.2.1.4 Extended Kalman Filter Method

The Kalman Filter is a state estimation technique that seeks to optimally combine a process model prediction of the state with a measurement of the state, where both have an associated uncertainty. Here we use the Extended Kalman Filter since it applies to nonlinear models (see [3] for a detailed introduction to Kalman Filter methods). We apply the algorithm to each nonzero mastery level update in the student's traversal of the class graph, as described earlier. The algorithm is then repeated until convergence:

1. $\bar{k}_t = a(k_{t-1})$
2. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t$
3. $K_t = \bar{\Sigma}_t B_t^T (B_t \bar{\Sigma}_t B_t^T + R_t)^{-1}$
4. $k_t = \bar{k}_t + K_t(Z_t - b(\bar{k}_t))$
5. $\Sigma_t = (I - K_t B_t)\bar{\Sigma}_t$

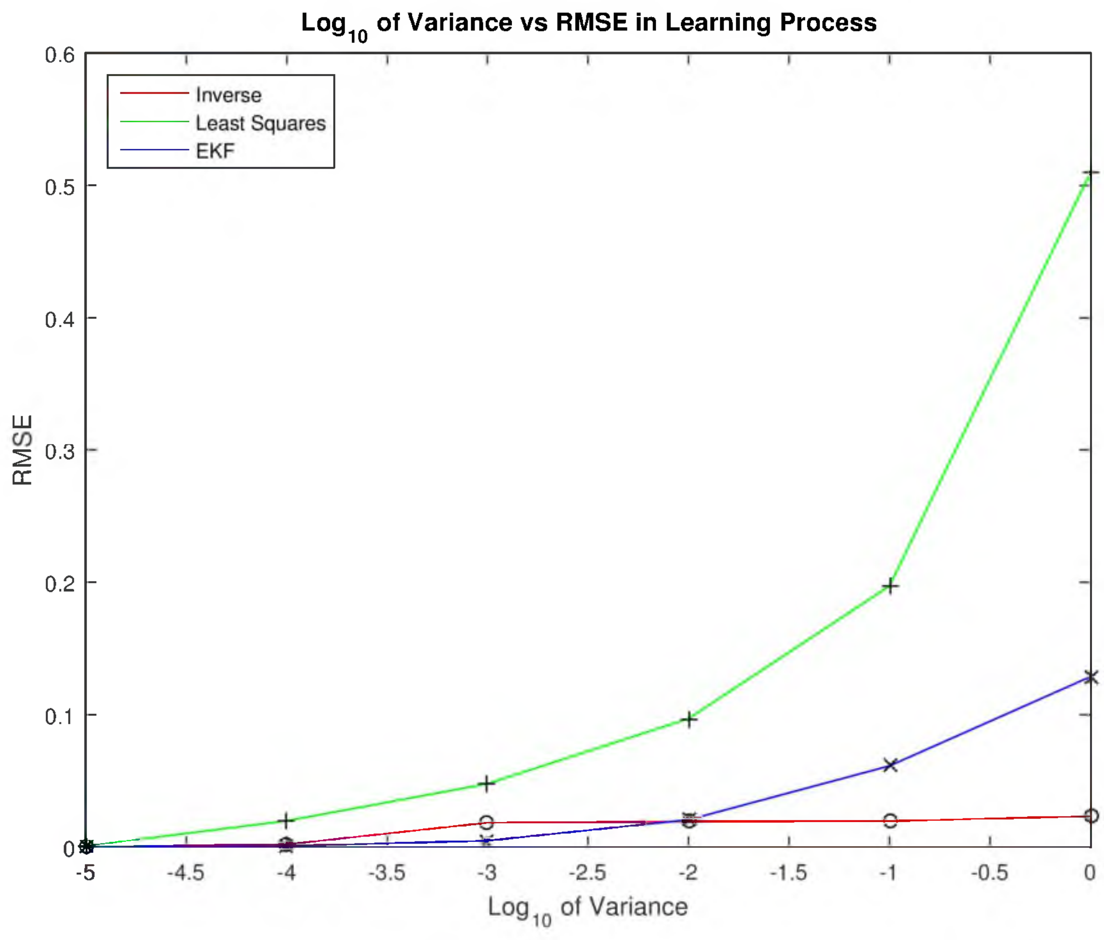


Figure 7.8. RMSE Values for Learning Coefficient Estimate for the Three Methods.

where k_t is the learning coefficient estimate at time t, a is the process model for how k evolves, A_t is the Jacobian of the process model for k, Q_t is the covariance for the process model for k, b is the measurement model (in our case, this is the learning update function), B_t is the Jacobian of the measurement model, K_t is the Kalman Filter gain, R_t is the covariance of the measurement model, and Z_t is the observed student performance. In particular, these variables are

$$\begin{aligned}
&a(k) = k \\
&A_t(k) = 1 \\
&b(k, s, x) = M - (M - x)e^{-ks} \\
&B(k, s, x) = s(M - x)e^{-ks}
\end{aligned}$$

and Q_t and R_t are assigned specific variances. Applying this method to the student learning traces yields the learning coefficient estimates shown in Figure 7.8. Figure 7.8 compares the three methods directly. As can be seen, the inverse method works best over all and the least squares method performs the worst, while the EKF works slightly better in lower noise than the inverse method.

7.3 Conclusion

These algorithms apply known techniques in a novel way. They add the concept of mastery to the learning models. These models are then calibrated with three different approaches. This ability to track student mastery allows the educator to regulate movement through the learning material based on mastery of the material. It also provides relative difficulty values that can provide insight to help the instructor better balance the student work load.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

8.1.1 Implementing a Graphical Course Map

Currently, learning items are presented to both faculty and students through a learning management system (LMS) in a linear, textual way. This presentation method has limited capacity to convey information about the organization of the learning material and the relations between learning items. Its primary focus is on the *precedes* relation, which identifies which learning item comes directly before another learning item. But there are much richer relations that can be expressed in a visual, graphical representation of a course like topical and prerequisite relations. Presenting information graphically provides an opportunity to convey a greater amount of information in a concise way. Using visual representations to increase the amount of information delivered and enhance the meaning of that information is not new. This technique is used in a variety of applications across many different fields; however, it is not yet available for the representation of course organization.

An automated system that constructs an initial course organization graph based on information provided by Canvas, a standard LMS, has been developed. It was found that the data in the LMS contained much information that could be used to create a graphical course map. A variety of types of material are represented in the nodes of the graph and initially only their chronology is known. A detailed analysis of the materials based on the text contained within each learning item allows a more informed representation which captures the topic relations among the items. A set of graph transformations is then defined which convert the (basically) linear structure of the course to a graph structure, which makes evident the dependencies and independencies of the learning items. Three specific test courses were transformed in this way to demonstrate the power of the method.

The system is able to exactly reproduce the temporal *precedes* relations of all dated learning items. For those learning items that do not have a due date, a date is used that is related to the earliest date in the unit in which the undated item is included. It is also possible to extract and identify the Unit or *includes* relations in all cases where the modules tool in Canvas is used. No testing was done with courses that did not use the modules tool. It is anticipated that input from the instructor would be required to identify unit relations if they are not identified in the LMS. Topical *occurs in* relations are produced from the text. In one of the example courses the automated assignment of *occurs in* relations was correct in 472 of the 490 topic to learning item combinations. In eight cases none of the topic word variations were found in the learning item, but they were, in fact, related to the topic. For nine combinations a topic word was encountered in the description but the learning item was not related to the topic. In one case the topic word was used to instruct the students to not use the topic on this specific assignment. These discrepancies were corrected with input from the instructor.

Using the topic relations determined by the detailed analysis, the *precedes* relations are combined with the topic relations to identify the relations between learning items that include both the precedes relation and the topic relation. Combining these two relations identifies a stronger relation between learning items. Another informative relation is the *prerequisite* relation that expresses a recommendation that one learning item be completed before another learning item. This relations is not available in the LMS but is derived through the combination of text analysis and input from the instructor.

Involving the instructor in the analysis process is one of the strengths of the ENABLE system. It not only improves the validity of the results but also improves the possibility of increasing the instructor's potential for understanding the possibilities for change. To facilitate this involvement an interface is provided that allows the addition of expert knowledge. This combination of data and expert knowledge is used to create a variety of course maps that display information about the learning material and the relations between learning items in a visual, graphical way. Currently, ENABLE produces eleven different course maps for each course, based on the relations between learning items. Each of these graphs can be rearranged by the instructor to produce many organizations. The course maps are designed to minimize the congestion of learning items and relations. The system does not, however, guarantee a minimum number of edge crossings. There is congestion of undated learning items as they are all positioned temporally at the beginning of a unit. Another congested area is produced when exams are split. The due date and unit assignments are the same for each of the split exams. This locates them close together. These congested areas are easy to resolve by dragging the nodes to new locations. When the nodes are moved, the connecting edges automatically move with them, keeping the visual representation of relations intact.

Course mapping has the potential to change how teachers and students see a course, giving teachers new insights and students a better understanding of how to negotiate the course materials.

8.1.2 Extending Accessibility

The availability and accessibility of education over the Web has increased, but barriers remain [2, 4, 127, 128]. The current presentation of material in a textual, linear format primarily based on chronology provides a limited view of the course and reaches only some of the potential users of online educational tools. Expanding the delivery of learning material to include a graphical course map can increase the information that is available and make that information accessible to a larger number of diverse consumers. Making educational information available to a broader spectrum of users has the potential to include more people in the educational process and improve their opportunities for inclusion and success.

As more and more educational opportunities are being made available on the Web, there is a greater need for tools that can present those materials in a more accessible way. An online course is not limited to a linear, chronological organization that has been the preferred presentation of the traditional classroom. This research demonstrates the possibilities of presenting learning materials in a graphical course map. This has led to many discoveries about the opportunities for enhancing the information available to students and educators. The development of a variety of course maps has identified new ways to organize and present learning materials and restructure their delivery to exploit the flexibility of the online setting.

Improving accommodation for people with different abilities and a wide range of circumstances can be augmented by removing the temporal limitations of the traditional text-based, linear presentation of course materials. To facilitate a different presentation of the learning material, there is a need to focus on more functional relations between learning items and presenting those relations in a graphical way. The ENABLE system provides interactive tools that allow an instructor to discover important relations between the learning items and manipulate a variety of graphical course maps that maintain those relations.

8.1.3 Facilitating Change

Presenting the course materials in a new way facilitates the consideration by an instructor of different ways to organize and deliver the material. Many instructors have been teaching a course for multiple semesters. When a change is needed or desirable, it is sometimes difficult to see new possibilities. Seeing the course materials in a graphical way provides a view of the course from a different perspective. This novel view can inspire the educator to consider new possibilities.

The functionality is provided to manipulate the graphical course maps produced by this process. This ability to move learning items around and make different connections between them allows the instructor to see the relations between learning items and how the organization of the learning items can be changed, while maintaining these relations. This process of analysis and transformation provides insight for the instructor about a variety of ways the course materials can be organized. As the instructor sees existing course materials presented in a variety of ways, their perception of the possibilities for making innovative changes is increased. This increase in possibilities facilitates change.

Information about the existing course is provided by each step in this process. The ENABLE system gathers existing information from what is available about the course in the LMS and represents it in a visual way. This sheds light on what students currently have available through their access to the learning materials in the LMS. A significant finding is how entrenched the *precedes* relation is in the presentation of course material. This relation often adds little meaning to how learning items are related, and yet it is the predominant organization strategy used when displaying information to students. When comparing the visual representation of that organization to the alternative organizations produced by ENABLE, it is clear that there is significant room for improvement in how the educational community presents learning material to students (see Figure 1.2). Although this research is designed to inform instructors about the many organization options available when making changes, the perception of the authors is that the effort to develop a graphical, nonlinear representation of a course could have significant impact on how students perceive and interact with course materials.

8.1.4 Student Agents

Artificial student agents have been developed that traverse the course map and implement decision-making. These agents have four characteristics: intelligence, work ethic, background, and distractibility. Each agent is assigned a value for each of these characteristics. The agent makes decisions about what learning item to consider and how much effort to exert on each learning item based on the values of theses characteristics. The agents are able to traverse the course maps in a variety of node sequences. They are limited only by prerequisite relations. A learning item cannot be attempted until the agent has visited all the prerequisite learning items.

A trace of the order the learning items are visited is recorded as each agent moves through the learning items, also referred to as traversing the graph. These agents can be used to traverse the graph over and over, producing a new trace for each traversal. These traces demonstrate a broad variety in the order in which the learning items can be attempted. This demonstrates the possibility of creating many different organizations and the potential of allowing students to choose the order in which they move through the learning items. This shows a variety of changes that are possible.

The score data from the sample courses is used to inform the characteristics and actions of these artificial student agents. Using these agents with the model provided by the course map provides a means for comparing traversals of the learning materials. The student agents can also be used to produce large sets of score data.

8.1.5 Probability Models

A probability model gives us a way to make predictions. These predictions can be used to inform students and educators about possible outcomes. Several probability models that predict grades on learning items were created. These models are trained using the existing score data. What was discovered is that probability models can indeed be generated from existing data. Many of those models are able to predict individual scores with over 70% accuracy. These models can be sampled to produce data that has a distribution similar to the original data.

This predictive capability can be used to inform educators and students about the likely results of a variety of choices. One of the clear messages from this work is the significant impact of zeros. The occurrence of a zero reflects that the student failed to work on a learning item. This choice has real implications for the overall success of the student. If a student is better informed of the effect of skipping an assignment, they may choose to engage rather than skip a learning item.

Several Bayesian inference networks were developed that incorporate the interdependencies between learning items. The score data from the sample courses and the course maps produced were used to create the conditional probability tables (CPTs). In addition, a variety of linear probability models were created. These models use a linear function and a variance based on the error model. The error model considers the difference between the actual values and the values computed by the linear function. These differences are used as the variance. Both that Bayesian models and the linear models are used for both prediction and sampling.

These predictive models were tested and compared to the actual data from the sample courses. Several measures were used to identify the validity of the resulting predictions. This comparison identified the models that produced the highest quality predictions.

8.1.6 Tracking Mastery of Learning Items

Both linear and nonlinear learning models were used to identify the mastery level of specific learning items. This technical approach is used to model student mastery of the learning items during a traversal of the course map. Calibration techniques were used to calibrate the learning coefficients of the individual student agent. Using this approach, relative difficulty for each learning item can also be determined. This provides more information to the instructor about the nature of the course and its organization.

8.1.7 Combining Complementary Techniques

Both innovative and established methods have been combined to analyze current course organization and inform course transformation. Automated analysis and instructor engagement are combined to augment broader solutions. Teacher experience, existing structure, and new views are assembled to generate new insights. This connecting of novel and well-known provides a solid foundation for the introduction of nonlinear, graphical course mapping.

8.2 Future Work

8.2.1 Enhancing the User Interface

This work has provided a solid foundation for the creation of graphical course mapping systems. For such a system to become widely useful an interface is needed that incorporates the relations discovered and the recommendations available through the predictive models. A meaningful next step would be to create a rich graphical user interface that improves both the quality and quantity of student and teacher interaction with the learning material. Conducting user testing at all stages of the system design, development, and testing would be used to identify the usability of the interface and make revisions based on the results. Such an interface could then be embedded in the LMS for student and faculty use on the web or mobile devices.

8.2.2 Implementation in Ongoing Courses

In the future, we intend to study the exploitation of ENABLE in actual on-line versions of computer science courses in order to validate the approach by using it with human instructors and students; in the present work, we have developed new algorithms and showed their correctness in simulation experiments based on data from actual classes. Future work will be undertaken both from the instructor's perspective; e.g., using the system to detect content or structural weaknesses in the course as well as the student's perspective; e.g., projected grades on future items as well as advice on what to do to improve mastery of the learning material – this would be accomplished through the control vector in the Kalman Filter approach.

8.2.3 Making Recommendations

In the future we hope to transform learning outcomes by (1) facilitating deep student learning in science and engineering by providing the student feedback resulting from behavior models based on monitoring paths taken through the online course graph and linking that to performance in the class, and (2) providing effective tools for the instructor to monitor the effectiveness of the course material and its organization. Using the predictive computational models for individual learner's and educator's recommendations can be added to the LMS. These recommendations can be used to guide the student as they navigate the learning materials in a course. They can also be used to inform educators as they engage in activities to support student success in the course.

By identifying operational student learning processes it may be possible to detect how knowledge gaps are a consequence of less successful learning strategies and tactics. Developing learning strategies can be challenging, thus an effective learning environment to support this must be designed and developed.

APPENDIX A

ENABLE USER MANUAL

ENABLE includes a graphical user interface. The instructions for using the interface are contained in the ENABLE User Manual. This manual contains instructions and images that explain how to interface with the ENABLE system. The user manual can be found in the repository at `http://enableThinking.com/CourseTransformation/repository.html`.

APPENDIX B

TRACES OF STUDENT AGENT TRAVERSALS OF THE COURSE MAP

As a student agent traverses the course map, it keeps track of each learning item as it encounters it. This trace can be displayed. The printed trace includes a list of the labels for each learning item. For the learning items that the agent did not complete, the label is in parentheses. Traces of 10 traversals for each of the seven student agents have been recorded. These traversal streams demonstrate the variety of ways the course map can be traversed. Although an agent has a set of characteristics and a specified approach to learning, they traverse the course map in a variety of ways. This variation is apparent in these traces. The traversal data can be found in the repository at `http://enableThinking.com/CourseTransformation/repository.html`.

APPENDIX C

COMPARISONS OF MULTIPLE STUDENT AGENT TRAVERSALS

At the end of a series of runs, the mean, max, min, and standard deviation of the final scores are computed. We can compare these values to agents with different characteristics. When the agent does only one traversal per run it is easy to see the variation from one traversal to another. When comparing the mean, max, min, and standard deviation of these runs, we see that running the agent 1000 times produces steady mean values. This is just the result of averaging together 1000 different final scores. The variation is still apparent in the max and min scores. With this many traversals you see the highest highs and lowest lows. The topical cohesion also becomes steady when many traversals are averaged. It is varying only by a single point from run to run for any given agent.

Data has been compiled that includes the following tables:

- Table 1 shows the results of doing 10 runs for each agent. The agent does only one traversal per run.
- Table 2 shows the results of doing 10 runs for each agent. The agent does 10 traversals per run.
- Table 3 shows the results of doing 10 runs for each agent. The agent does 100 traversals per run.
- Table 4 shows the results of doing 10 runs for each agent. The agent does 1000 traversals per run.

These tables of data can be found in the repository at `http://enableThinking.com/CourseTransformation/repository.html`.

APPENDIX D

INDIVIDUAL SCORE REPORTS CPT MODEL

Four sample sets of data have been compiled that contain the individual score data for the first 143 sample runs for the CPT model. Every sample set includes a score for each learning item. This sampling was done by taking each learning item in the order it was presented in the original course. A score was generated for the learning item by sampling from the score probability distribution created by the individual model for that item. The individual scores produced by this sampling were used to compute a final score.

The first model presented is a CPT model with five buckets, four grade buckets, one each for A, B, C, and DF. The fifth bucket is for zero scores. This particular CPT model uses both the immediately precedes and the pruned prerequisite relations. It combines both the very current information by including the immediately preceding item, as well as a more extended view of the preceding scores by including the pruned prerequisite information.

The data is broken up into assignment groups. First are the columns that contain the exercises, exams, and the final score. Right after these scores are the homework scores. Finally, the activity scores. There are too many activity scores to fit across one page, so first come activities 1 to 10 followed by activities 11 to 20. This data can be found in the repository at `http://enableThinking.com/CourseTransformation/repository.html`.

APPENDIX E

INDIVIDUAL SCORE REPORTS LINEAR MODEL

Four sample sets of data have been compiled that contain the individual score data for the first 143 sample runs for the linear model. Every sample set included a score for each learning item. This sampling was done by taking each learning item in the order it was presented in the original course. A score was generated for the learning item by sampling from the score probability distribution created by the individual model for that item. The individual scores produced by this sampling were used to compute a final score. The model is a linear model. This particular linear model uses all the preceding learning items in the feature list.

The data is broken up into assignment groups. First are the columns that contain the exercises, exams, and the final score. Right after these scores are the homework scores. Finally, the activity scores. There are too many activity scores to fit across one page, so first come activities 1 to 10 followed by activities 11 to 20. This data can be found in the repository at `http://enableThinking.com/CourseTransformation/repository.html`.

APPENDIX F

INDIVIDUAL SCORE REPORTS MIXED MODEL

Four sample sets of data have been compiled that contain the individual score data for the first 143 sample runs for the linear model. Every sample set included a score for each learning item. This sampling was done by taking each learning item in the order it was presented in the original course. A score was generated for the learning item by sampling from the score probability distribution created by the individual model for that item. The individual scores produced by this sampling were used to compute a final score. The mixed linear model uses all the preceding learning items in the feature list.

The data are broken up into assignment groups. First are the columns that contain the exercises, exams, and the final score. Right after these scores are the homework scores. Finally, the activity scores. There are too many activity scores to fit across one page so first come activities 1 to 10 followed by activities 11 to 20. These data can be found in the repository at `http://enableThinking.com/CourseTransformation/repository.html`.

APPENDIX G

COMPARING MODELS BY ASSIGNMENT GROUP

The following four tables compare 38 different models. Each table uses a different value for the comparison. The comparisons are also broken down based on learning item type. ENABLE identifies four distinct types of learning items: exercises, exams, homework, and activities. The first value in each table is the overall value. This column is followed by a column for each learning item type.

Table G.1 compares grade accuracy. Table G.2 compares L1 errors. Table G.3 compares L2 errors. Table G.4 compares log-likelihood scores.

Table G.1. Comparison of Grade Accuracy Between Models and Learning Activity Types.

Model Type	**Dependencies**	Grade Accuracy	Exercise	Exam	HW	Activity
Mixed Linear	PrecedesThree	77%	72%	49%	71%	85%
Mixed Linear	P_3/PrunedPrereqs	77%	72%	49%	71%	85%
Mixed Linear	PrecedesFive	76%	71%	49%	69%	85%
Mixed Linear	PrecedesTwo	76%	72%	45%	70%	84%
Mixed Linear	PrecedesNine	76%	70%	46%	70%	84%
Mixed Linear	PrecedesOne	75%	68%	42%	71%	83%
Mixed Linear	P_1/PrunedPrereqs	75%	68%	42%	71%	83%
Mixed Linear	PrecedesThirteen	75%	69%	49%	69%	84%
CPT Grades	PrecedesOne	75%	69%	39%	71%	83%
CPT Grades	P_1/PrunedPrereqs	75%	69%	39%	71%	83%
CPT Grades	PrecedesTwo	75%	72%	46%	69%	84%
Mixed Linear	Precedes	73%	67%	46%	68%	81%
CPT Grades	Empty	73%	68%	39%	69%	82%
CPT Grades	PrunedPrereqs	73%	68%	39%	69%	82%
CPT Grades	Prereqs	73%	68%	39%	69%	82%
CPT Grades	PrecedesThree	73%	68%	46%	68%	81%
Mixed Linear	PrunedPrereqs	72%	68%	26%	69%	82%
Mixed Linear	Prereqs	72%	68%	26%	69%	82%
Mixed Linear	Empty	72%	68%	26%	69%	82%
Linear	PrecedesThree	72%	62%	47%	68%	81%
Linear	P_3/PrunedPrereqs	72%	62%	47%	68%	81%
Linear	PrecedesFive	72%	60%	47%	68%	81%
Linear	PrecedesTwo	72%	61%	45%	68%	81%
Linear	PrecedesOne	72%	61%	42%	67%	82%
Linear	P_1/PrunedPrereqs	72%	61%	42%	67%	82%
Linear	PrecedesNine	72%	61%	47%	68%	80%
Linear	PrecedesThirteen	72%	60%	49%	67%	80%
Linear	Precedes	70%	59%	47%	67%	78%
Linear	Empty	67%	60%	29%	66%	74%
Linear	Prereqs	67%	60%	29%	66%	74%
Linear	PrunedPrereqs	67%	60%	29%	66%	74%
CPT Percentiles	PrecedesThree	32%	34%	48%	50%	19%
CPT Percentiles	PrecedesOne	32%	34%	38%	52%	19%
CPT Percentiles	P_1/PrunedPrereqs	32%	34%	38%	52%	19%
CPT Percentiles	PrecedesTwo	31%	34%	36%	52%	19%
CPT Percentiles	Empty	31%	33%	36%	50%	19%
CPT Percentiles	PrunedPrereqs	31%	33%	36%	50%	19%
CPT Percentiles	Prereqs	31%	33%	36%	50%	19%

Table G.2. Comparison of L1 Errors Between Models and Learning Activity Types.

Model Type	**Dependencies**	L1 Errors	Exercise	Exam	HW	Activity
Mixed Linear	PrecedesFive	4.8	4.2	11.6	7.26	2.54
Mixed Linear	PrecedesThree	4.9	4.0	12.3	7.34	2.62
Mixed Linear	P_3/PrunedPrereqs	4.9	4.0	12.3	7.34	2.62
Mixed Linear	PrecedesNine	5.0	4.3	12.1	7.48	2.73
Mixed Linear	PrecedesTwo	5.0	4.2	13.2	7.37	2.68
Mixed Linear	PrecedesThirteen	5.1	4.4	11.7	7.81	2.77
CPT Grades	PrecedesTwo	5.4	4.6	13.2	7.77	3.17
CPT Grades	PrecedesThree	5.5	4.6	12.5	8.00	3.32
Linear	PrecedesThirteen	5.6	4.8	11.0	8.21	3.43
Mixed Linear	Precedes	5.6	4.6	12.9	8.08	3.30
Linear	PrecedesNine	5.6	4.8	12.5	7.99	3.41
Linear	PrecedesFive	5.6	4.8	12.6	8.08	3.43
Linear	Precedes	5.8	4.9	12.2	8.42	3.57
Mixed Linear	PrecedesOne	5.8	4.9	15.5	8.72	2.92
Mixed Linear	P_1/PrunedPrereqs	5.8	4.9	15.5	8.72	2.92
Linear	PrecedesThree	5.9	4.9	13.0	8.64	3.48
Linear	P_3/PrunedPrereqs	5.9	4.9	13.0	8.64	3.48
CPT Grades	PrecedesOne	6.1	5.1	15.4	8.95	3.40
CPT Grades	P_1/PrunedPrereqs	6.1	5.1	15.4	8.95	3.40
Linear	PrecedesTwo	6.2	5.2	14.1	8.84	3.67
Mixed Linear	PrunedPrereqs	6.2	5.1	15.9	9.51	3.20
Mixed Linear	Prereqs	6.2	5.1	15.9	9.51	3.20
Mixed Linear	Empty	6.2	5.1	15.9	9.51	3.20
CPT Grades	Empty	6.3	5.0	15.0	9.31	3.62
CPT Grades	PrunedPrereqs	6.3	5.0	15.0	9.31	3.62
CPT Grades	Prereqs	6.3	5.0	15.0	9.31	3.62
Linear	PrecedesOne	6.7	5.7	15.9	9.57	3.96
Linear	P_1/PrunedPrereqs	6.7	5.7	15.9	9.57	3.96
Linear	Empty	7.6	6.2	15.8	11.19	4.61
Linear	Prereqs	7.6	6.2	15.8	11.19	4.61
Linear	PrunedPrereqs	7.6	6.2	15.8	11.19	4.61
CPT Percentiles	PrecedesTwo	9.5	8.4	13.4	8.92	9.42
CPT Percentiles	PrecedesThree	9.5	8.4	13.7	8.94	9.40
CPT Percentiles	PrecedesOne	10.0	8.7	15.8	9.75	9.46
CPT Percentiles	P_1/PrunedPrereqs	10.0	8.7	15.8	9.75	9.46
CPT Percentiles	Empty	10.2	9.0	15.2	10.02	9.88
CPT Percentiles	PrunedPrereqs	10.2	9.0	15.2	10.02	9.88
CPT Percentiles	Prereqs	10.2	9.0	15.2	10.02	9.88

Table G.3. Comparison of l2 Errors Between Models and Learning Activity Types.

Model Type	**Dependencies**	L2 Errors	Exercise	Exam	HW	Activity
Linear	PrecedesFive	0.1411	0.2833	0.8372	0.3690	0.1082
Linear	PrecedesThirteen	0.1419	0.2867	0.7777	0.3801	0.1097
Linear	PrecedesNine	0.1423	0.2855	0.8504	0.3707	0.1093
Linear	Precedes	0.1466	0.2906	0.8580	0.3839	0.1141
Mixed Linear	PrecedesFive	0.1468	0.3115	0.8056	0.3899	0.1155
Linear	PrecedesThree	0.1478	0.2820	0.8723	0.3928	0.1090
Linear	P_3/PrunedPrereqs	0.1478	0.2820	0.8723	0.3928	0.1090
Mixed Linear	PrecedesThree	0.1492	0.2997	0.8766	0.3893	0.1170
Mixed Linear	P_3/PrunedPrereqs	0.1492	0.2997	0.8766	0.3893	0.1170
Mixed Linear	PrecedesNine	0.1523	0.3219	0.8634	0.3990	0.1210
Linear	PrecedesTwo	0.1536	0.2931	0.9348	0.4052	0.1116
Mixed Linear	PrecedesTwo	0.1549	0.3086	0.9349	0.4024	0.1188
Mixed Linear	PrecedesThirteen	0.1569	0.3229	0.8909	0.4146	0.1219
CPT Grades	PrecedesThree	0.1572	0.3247	0.9093	0.4186	0.1157
CPT Grades	PrecedesTwo	0.1583	0.3225	0.9656	0.4133	0.1170
Linear	PrecedesOne	0.1638	0.3057	1.0402	0.4281	0.1155
Linear	P_1/PrunedPrereqs	0.1638	0.3057	1.0402	0.4281	0.1155
Mixed Linear	Precedes	0.1675	0.3330	0.9671	0.4340	0.1372
Mixed Linear	PrecedesOne	0.1711	0.3435	1.0473	0.4475	0.1252
Mixed Linear	P_1/PrunedPrereqs	0.1711	0.3435	1.0473	0.4475	0.1252
CPT Grades	PrecedesOne	0.1767	0.3553	1.1189	0.4599	0.1245
CPT Grades	P_1/PrunedPrereqs	0.1767	0.3553	1.1189	0.4599	0.1245
Linear	Empty	0.1798	0.3238	1.0955	0.4803	0.1252
Linear	Prereqs	0.1798	0.3238	1.0955	0.4803	0.1252
Linear	PrunedPrereqs	0.1798	0.3238	1.0955	0.4803	0.1252
CPT Percentiles	PrecedesThree	0.1814	0.4051	0.9440	0.4320	0.1890
CPT Percentiles	PrecedesTwo	0.1816	0.4066	0.9659	0.4304	0.1882
Mixed Linear	PrunedPrereqs	0.1836	0.3391	1.1089	0.4888	0.1310
Mixed Linear	Prereqs	0.1836	0.3391	1.1089	0.4888	0.1310
Mixed Linear	Empty	0.1836	0.3391	1.1089	0.4888	0.1310
CPT Grades	Empty	0.1878	0.3413	1.1438	0.5028	0.1293
CPT Grades	PrunedPrereqs	0.1878	0.3413	1.1438	0.5028	0.1293
CPT Grades	Prereqs	0.1878	0.3413	1.1438	0.5028	0.1293
CPT Percentiles	PrecedesOne	0.1947	0.4256	1.0852	0.4707	0.1898
CPT Percentiles	P_1/PrunedPrereqs	0.1947	0.4256	1.0852	0.4707	0.1898
CPT Percentiles	Empty	0.2058	0.4217	1.1273	0.5143	0.1928
CPT Percentiles	PrunedPrereqs	0.2058	0.4217	1.1273	0.5143	0.1928
CPT Percentiles	Prereqs	0.2058	0.4217	1.1273	0.5143	0.1928

Table G.4. Comparison of Log-likelihood Scores Between Models and Learning Activity Types.

Model Type	**Dependencies**	Log Likelihood	Exercise	Exam	HW	Activity
Linear	PrecedesFive	0.1411	0.2833	0.8372	0.3690	0.1082
Linear	PrecedesThirteen	0.1419	0.2867	0.7777	0.3801	0.1097
Linear	PrecedesNine	0.1423	0.2855	0.8504	0.3707	0.1093
Linear	Precedes	0.1466	0.2906	0.8580	0.3839	0.1141
Mixed Linear	PrecedesFive	0.1468	0.3115	0.8056	0.3899	0.1155
Linear	PrecedesThree	0.1478	0.2820	0.8723	0.3928	0.1090
Linear	P_3/PrunedPrereqs	0.1478	0.2820	0.8723	0.3928	0.1090
Mixed Linear	PrecedesThree	0.1492	0.2997	0.8766	0.3893	0.1170
Mixed Linear	P_3/PrunedPrereqs	0.1492	0.2997	0.8766	0.3893	0.1170
Mixed Linear	PrecedesNine	0.1523	0.3219	0.8634	0.3990	0.1210
Linear	PrecedesTwo	0.1536	0.2931	0.9348	0.4052	0.1116
Mixed Linear	PrecedesTwo	0.1549	0.3086	0.9349	0.4024	0.1188
Mixed Linear	PrecedesThirteen	0.1569	0.3229	0.8909	0.4146	0.1219
CPT Grades	PrecedesThree	0.1572	0.3247	0.9093	0.4186	0.1157
CPT Grades	PrecedesTwo	0.1583	0.3225	0.9656	0.4133	0.1170
Linear	PrecedesOne	0.1638	0.3057	1.0402	0.4281	0.1155
Linear	P_1/PrunedPrereqs	0.1638	0.3057	1.0402	0.4281	0.1155
Mixed Linear	Precedes	0.1675	0.3330	0.9671	0.4340	0.1372
Mixed Linear	PrecedesOne	0.1711	0.3435	1.0473	0.4475	0.1252
Mixed Linear	P_1/PrunedPrereqs	0.1711	0.3435	1.0473	0.4475	0.1252
CPT Grades	PrecedesOne	0.1767	0.3553	1.1189	0.4599	0.1245
CPT Grades	P_1/PrunedPrereqs	0.1767	0.3553	1.1189	0.4599	0.1245
Linear	Empty	0.1798	0.3238	1.0955	0.4803	0.1252
Linear	Prereqs	0.1798	0.3238	1.0955	0.4803	0.1252
Linear	PrunedPrereqs	0.1798	0.3238	1.0955	0.4803	0.1252
CPT Percentiles	PrecedesThree	0.1814	0.4051	0.9440	0.4320	0.1890
CPT Percentiles	PrecedesTwo	0.1816	0.4066	0.9659	0.4304	0.1882
Mixed Linear	PrunedPrereqs	0.1836	0.3391	1.1089	0.4888	0.1310
Mixed Linear	Prereqs	0.1836	0.3391	1.1089	0.4888	0.1310
Mixed Linear	Empty	0.1836	0.3391	1.1089	0.4888	0.1310
CPT Grades	Empty	0.1878	0.3413	1.1438	0.5028	0.1293
CPT Grades	PrunedPrereqs	0.1878	0.3413	1.1438	0.5028	0.1293
CPT Grades	Prereqs	0.1878	0.3413	1.1438	0.5028	0.1293
CPT Percentiles	PrecedesOne	0.1947	0.4256	1.0852	0.4707	0.1898
CPT Percentiles	P_1/PrunedPrereqs	0.1947	0.4256	1.0852	0.4707	0.1898
CPT Percentiles	Empty	0.2058	0.4217	1.1273	0.5143	0.1928
CPT Percentiles	PrunedPrereqs	0.2058	0.4217	1.1273	0.5143	0.1928
CPT Percentiles	Prereqs	0.2058	0.4217	1.1273	0.5143	0.1928

REFERENCES

[1] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2009.

[2] I. E. Allen and J. Seaman, "Grade change: Tracking online education in the united states, 2013," *Babson Survey Research Group and Quahog Research Group, LLC. Retrieved on*, vol. 3, no. 5, p. 2014, 2014.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics.* MIT press, 2005.

[4] D. Rover, Y. Astatke, S. Bakshi, and F. Vahid, "An online revolution in learning and teaching," in *Proc. Front. Educ. Conf.* Oklahoma City, Oklahoma: IEEE, 10 2013.

[5] S. Cutler, M. Borrego, M. P. C. Henderson, and J. Froyd, "A comparison of electrical, computer, and chemical engineering faculty progressions through the innovation-decision process," in *Proc. Front. Educ. Conf.* Seattle, Washington: IEEE, 10 2012.

[6] I. Allen and J. Seaman, *Changing Course: Ten Years of Tracking Online Education in the United States.* Newburyport, MA: Sloan Consortium, 2013.

[7] T. Willett, "Current status of curriculum mapping in canada and the uk," *Med. Educ.*, vol. 42, no. 8, pp. 786–793, 2008.

[8] S. Kome, C. Booth, D. Brecher, M. Lowe, S. Stone, and N. Tagge, "Computer science curriculum map 2013-2014," http://scholarship.claremont.edu/ccct_cmaps/36, 2014, accessed: 2015-05-21.

[9] S. Stone, C. Booth, D. Brecher, M. Lowe, and N. Tagge, "Engineering curriculum map 2013-2014," http://scholarship.claremont.edu/ccct_cmaps/37, 2014, accessed: 2015-05-21.

[10] R. Harden, "Amee guide no. 21: Curriculum mapping: A tool for transparent and authentic teaching and learning," *Med. Teach.*, vol. 23, no. 2, pp. 123–137, 2001.

[11] K. Uchiyama and J. Radin, "Curriculum mapping in higher education: A vehicle for collaboration," *Innov. High. Educ.*, vol. 33, no. 4, pp. 271–280, 2009.

[12] D. Curtis and D. Moss, "Curriculum mapping: Bringing evidence-based frameworks to legal education," *Nova Law Rev.*, vol. 34, no. 473, 2010.

[13] L. DuHadway and T. Henderson, "Informing change: Course content analysis and organization," School of Computing, University of Utah, Salt Lake City, UT, Tech. Rep. Tech. Rep. UUCS 15-002, 2015.

[14] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, no. 3, pp. 75–174, 2010.

[15] S. Bridgeman, “Graph drawing in education,” in *Handbook of Graph Drawing and Visualization*, R. Tamassia, Ed. Chicago, IL: CNC Press, 2013, pp. 737–762.

[16] H. Ehrig, H. Kreowski, U. U. Montanari, and G. Rozenberg, *Handbook of Graph Grammars and Computing by Graph Transformation: vol. 3: Concurrency, Parallelism, and Distribution*. Hackensack, NJ: World Scientific Publishing Co., Inc, 1999, vol. 3.

[17] J. Bzivin, F. Bttner, M. Gogolla, F. Jouault, I. Kurtev, and A. Lindow, “Model transformations? transformation models!” in *In Model Driven Engineering Languages and Systems*. Berlin Heidelberg, Germany: Springer, 2006, pp. 440–453.

[18] G. Mezei, L. Lengyel, T. Meszaros, and T. Vajk, “Metamodeling, model processing and simulation - putting the puzzle pieces together based on industrial experiences,” in *Model-Driven Engineering Languages and Systems*, J. Dingel, W. Schulte, I. Ramos, S. Abrahao, and E. Insfran, Eds. Switzerland: Springer International Publishing, 2014.

[19] G. Taentzer, “Agg: A graph transformation environment for modeling and validation of software,” in *Lect. Notes Comput. Sc.*, J. Pfaltz, M. Nagl, and B. Bohlen, Eds. Berlin Heidelberg, Germany: Springer, 2004, pp. 446–453.

[20] G. Taentzer, K. Ehrig, E. Guerra, J. de Lara, L. Lengyel, T. Levendovszky, U. Prange, D. Varro, and S. Varro-Gyapay, “Model transformation by graph transformation: A comparative study,” in *Worksh. Model Transform. Pract.*, Montego Bay, Jamaica, 10 2005.

[21] D. Varr and A. Balogh, “The model transformation language of the viatra2 framework,” *Sci. of Comput. Program.*, vol. 68, no. 3, pp. 214–234, 2007.

[22] D. Varr and A. Pataricza, “Vpm: A visual, precise and multilevel metamodeling framework for describing mathematical domains and uml (the mathematics of metamodeling is metamodeling mathematics,” *Softw. Syst. Model.*, vol. 2, no. 3, pp. 187–210, 2003.

[23] K. Czarnecki and S. Helsen, “Feature-based survey of model transformation approaches,” *IBM Syst. J.*, vol. 45, no. 3, pp. 621–645, 2006.

[24] M. Desmarais and R. d Baker, “A review of recent advances in learner and skill modeling in intelligent learning environments,” *User Model. User-Adapt.*, vol. 22, no. 1–2, pp. 9–38, 2012.

[25] Z. Pardos and N. Heffernan, “Modeling individualization in a bayesian networks implementation of knowledge tracing,” in *P. 18th Int. Conf. User Model. Adaptat. Personalization*. Big Island, HI: ACM, 6 2010, pp. 255–266.

[26] C. Carmona, G. Castillo, and E. Milln, “Designing a dynamic bayesian network for modeling students’ learning styles,” in *IEEE Adv. Learn. Technol.* Santander, Cantabria, Spain: IEEE, 7 2008, pp. 346–350.

[27] C. Conati, A. Gertner, and K. Vanlehn, “Using bayesian networks to manage uncertainty in student modeling,” *User Model. User-Adap.*, vol. 12, no. 4, pp. 371–417, 2002.

[28] P. Garca, A. Amandi, S. Schiaffino, and M. Campo, "Evaluating bayesian networks' precision for detecting students' learning styles," *Comput. Educ.*, vol. 49, no. 3, pp. 794–808, 2007.

[29] A. Anthony and M. Raney, "Bayesian network analysis of computer science grade distributions," in *P. Comput. Sci. Educ.* Raleigh, NC: ACM, 2 2012, pp. 649–654.

[30] M. Prensky, "Digital natives, digital immigrants part one," *On the Horiz.*, vol. 9, no. 5, pp. 1–6, 2001.

[31] S. Bennett, K. Maton, and L. Kervin, "The 'digital natives' debate: a critical review of the evidence," *Brit. J. Educ. Technol.*, vol. 39, no. 5, pp. 775–786, 2008.

[32] E. Smith, "The digital native debate in higher education: A comparative analysis of recent literature/le débat sur les natifs du numérique dans l'enseignement supérieur: une analyse comparative de la littérature récente," *Can. J. Learn. Technol.*, vol. 38, no. 3, 2012.

[33] A. F. Mayadas, J. Bourne, and P. Bacsich, "Online education today," *Science*, vol. 323, no. 5910, pp. 85–89, 2009.

[34] Y. Zhao, J. Lei, B. Yan, C. Lai, and S. Tan, "What makes the difference? a practical analysis of research on the effectiveness of distance education," *Teach. Coll. Rec.*, vol. 107, no. 8, pp. 1836–1884, 2005.

[35] A. N. Nichols, D. Steele, and E. Washburn, "The details make a difference when delivering technology courses via web-based distant education," in *Proc. Front. Educ. Conf.* IEEE, 2010, pp. T4C–1.

[36] J. Lockyer, J. Sargeant, V. Curran, and L. Fleet, "The transition from face-to-face to online cme facilitation," *Med. Teach.*, vol. 28, no. 7, pp. 625–630, 2006.

[37] A. Gordillo, E. Barra, D. Gallego, and J. Quemada, "An online e-learning authoring tool to create interactive multidevice learning objects using e- infrastructure resources," in *Proc. Front. Educ. Conf.* IEEE, 2013, pp. 1914–1920.

[38] B. King, H. McCauslan, and T. Nunan, "Converting to online course and program delivery: the university of south australia case study," *The Int. Rev. Res. Open Distr. Learn.*, vol. 1, no. 2, 2001.

[39] A. W. Chickering and Z. F. Gamson, "Seven principles for good practice in undergraduate education," *AAHE Bull.*, vol. 3, p. 7, 1987.

[40] P. Rangachari, "Back to the future? active learning of medical physiology in the 1900s," *Adv. Physiol. Educ.*, vol. 31, no. 4, pp. 283–287, 2007.

[41] D. Grant, A. Malloy, and G. Hollowell, "Enhancing students interest in science and technology through cross-disciplinary collaboration and active learning techniques," in *P. Inform. Sci. Inform. Technol. Educ.*, vol. 2013, no. 1, 2013, pp. 101–112.

[42] B. Settles, "Active learning literature survey," *University of Wisconsin, Madison*, vol. 52, no. 55-66, p. 11, 2010.

[43] T. Andrews, M. Leonard, C. Colgrove, and S. Kalinowski, "Active learning not associated with student learning in a random sample of college biology courses," *CBE-Life Sci. Educ.*, vol. 10, no. 4, pp. 394–405, 2011.

[44] V. Drew and L. Mackie, "Extending the constructs of active learning: Implications for teachers' pedagogy and practice," *Curriculum J.*, vol. 22, no. 4, pp. 451–467, 2011.

[45] M. Prince, "Does active learning work? a review of the research," *J. Eng. Educ. Wash.*, vol. 93, pp. 223–232, 2004.

[46] L. Blasco-Arcas, I. Buil, B. Hernández-Ortega, and F. J. Sese, "Using clickers in class. the role of interactivity, active collaborative learning and engagement in learning performance," *Comput. Educ.*, vol. 62, pp. 102–110, 2013.

[47] J. R. Drake, "A critical analysis of active learning and an alternative pedagogical framework for introductory information systems courses," *J. Inform. Technol. Educ.*, vol. 11, pp. 39–52, 2012.

[48] C. J. Miller and M. J. Metz, "A comparison of professional level faculty and student perceptions of active learning: Its current use, effectiveness, and barriers," *Adv. in Physiol. Educ.*, vol. 38, no. 3, pp. 246–252, 2014.

[49] C. B. Lee, S. Garcia, and L. Porter, "Can peer instruction be effective in upper division computer science courses?" *ACM T. Comput. Educ.*, vol. 13, no. 3, p. 12, 2013.

[50] S. Kotru, S. L. Burkett, and D. J. Jackson, "Active and collaborative learning in an introductory electrical and computer engineering course," *J. of Gen. Educ.*, vol. 59, no. 4, pp. 264–272, 2010.

[51] J. Gardner and B. R. Belland, "A conceptual framework for organizing active learning experiences in biology instruction," *J. Sci. Educ. Technol.*, vol. 21, no. 4, pp. 465–475, 2012.

[52] H. H. Hu and T. D. Shepherd, "Teaching cs one with pogil activities and roles," in *SIGCSE Bull.* ACM, 2014, pp. 127–132.

[53] Z. Zayapragassarazan and S. Kumar, "Active learning methods," *Online Sub.*, vol. 19, no. 1, pp. 3–5, 2012.

[54] C. J. Miller, J. McNear, and M. J. Metz, "A comparison of traditional and engaging lecture methods in a large, professional level course," *Adv. Physiol. Educ.*, vol. 37, no. 4, pp. 347–355, 2013.

[55] W. Cashin, "Idea paper# 50 effective lecturing," 2010.

[56] "Process oriented guided inquiry learning classroom activities," http://pogil.org/resources/curriculum-materials/classroom-activities, 2015, accessed: 2015-09-21.

[57] I. D. Cherney, "The effects of active learning on students' memories for course content," *Active Learn. High. Educ.*, vol. 9, no. 2, pp. 152–171, 2008.

[58] A. S. Richmond and L. K. Hagan, "Promoting higher level thinking in psychology is active learning the answer?" *Teach. Psychol.*, vol. 38, no. 2, pp. 102–105, 2011.

[59] M. Cavanagh, "Students' experiences of active engagement through cooperative learning activities in lectures," *Active Learn. High. Educ.*, vol. 12, no. 1, pp. 23–33, 2011.

[60] B. Detlor, L. Booker, A. Serenko, and H. Julien, "Student perceptions of information literacy instruction: the importance of active learning," *Educ. Inform.*, vol. 29, no. 2, pp. 147–161, 2012.

[61] D. L. Linton, J. K. Farmer, and E. Peterson, "Is peer interaction necessary for optimal active learning?" *CBE-Life Sci. Educ.*, vol. 13, no. 2, pp. 243–252, 2014.

[62] B. S. Bloom, *Taxonomy of educational objectives: The classification of education goals by a committee of college and university examiners.* David McKay, 1956.

[63] J. R. Drake, M. OHara, and E. Seeman, "Five principles for mooc design: With a case study." *J. Inform. Technol. Educ.*, vol. 14, pp. 125–143, 2015.

[64] M. Liu, D. Wrobbel, and I. Blankson, "Rethinking program assessment through the use of program alignment mapping technique," *Communicat. Teach.*, vol. 24, no. 4, pp. 238–246, 2010.

[65] B. B. Marshall, H. Chen, R. Shen, and E. A. Fox, "Moving digital libraries into the student learning space: the getsmart experience," *J. Educ. Resour. Comput.*, vol. 6, no. 1, p. 2, 2006.

[66] B. Marshall, Y. Zhang, H. Chen, A. Lally, R. Shen, E. Fox, and L. N. Cassel, "Convergence of knowledge management and e-learning: the getsmart experience," in *Lect. Notes Comput. Sc.* IEEE, 2003, pp. 135–146.

[67] D. Hay, I. Kinchin, and S. Lygo-Baker, "Making learning visible: the role of concept mapping in higher education," *Stud. High. Educ.*, vol. 33, no. 3, pp. 295–311, 2008.

[68] M. Davies, "Concept mapping, mind mapping and argument mapping: what are the differences and do they matter?" *High. Educ.*, vol. 62, no. 3, pp. 279–301, 2011.

[69] J. D. Novak and A. J. Cañas, "The origins of the concept mapping tool and the continuing evolution of the tool," *Infor. Vis.*, vol. 5, no. 3, pp. 175–184, 2006.

[70] J. C. Nesbit and O. O. Adesope, "Learning with concept and knowledge maps: a meta-analysis," *Rev. Educ. Res.*, vol. 76, no. 3, pp. 413–448, 2006.

[71] E. N. Biktimirov and L. B. Nilson, "Mapping your course: Designing a graphic syllabus for introductory finance," *J. Educ. Bus.*, vol. 78, no. 6, pp. 308–312, 2003.

[72] L. B. Nilson, *The Graphic Syllabus and the Outcomes Map: Communicating Your Course.* John Wiley & Sons, 2009, vol. 137.

[73] W. Zhang, T. Yoshida, and X. Tang, "A comparative study of tf*idf, lsi and multi-words for text classification," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 2758–2765, 2011.

[74] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 1999, pp. 50–57.

[75] ——, "Unsupervised learning by probabilistic latent semantic analysis," *Mach. Learn.*, vol. 42, no. 1-2, pp. 177–196, 2001.

[76] J.-T. Chien and M.-S. Wu, "Adaptive bayesian latent semantic analysis," *IEEE T. Audio Speech*, vol. 16, no. 1, pp. 198–207, 2008.

[77] A. Lancichinetti, M. I. Sirer, J. X. Wang, D. Acuna, K. Körding, and L. A. N. Amaral, "High-reproducibility and high-accuracy method for automated topic classification," *Phys. Rev. X*, vol. 5, no. 1, p. 011007, 2015.

[78] W. Song and S. C. Park, "A novel document clustering model based on latent semantic analysis," in *Semantics Knowl. Grid Third Int. Conf.* IEEE, 2007, pp. 539–542.

[79] K. Toutanova, F. Chen, K. Popat, and T. Hofmann, "Text classification in a hierarchical mixture model for small training sets," in *Proc. 10th Int. Conf. Info. Knowl. Manag.* ACM, 2001, pp. 105–113.

[80] X. Phan, L. Nguyen, and S. Horiguchi, "Learning to classify short and sparse text & web with hidden topics from large-scale data collections," in *17th Int. Conf. WWW*. Beijing, China: ACM, 4 2008, pp. 91–100.

[81] G. Taentzer, O. Runge, E. Biermann, J. Schneider, M. Matz, B. Melamed, M. Rudolf, T. Schultzke, and S. Gruner, "The attributed graph grammar system: A development environment for attributed graph transformation systems," http://user.cs.tu-berlin.de/~gragra/agg/, 2015, accessed: 2015-03-13.

[82] J. D. Lara, H. Vangheluwe, E. Posse, I. A.V., M. Provost, and W. Liang, "A tool for multi-formalism meta-modelling," http://atom3.cs.mcgill.ca, 2015, accessed: 2015-03-13.

[83] W. Herzner, A. Balogh, and G. Csertán, "Design patterns for domain-specific application modeling," in *DECOS/ERCIM W. Dependable Emb. Syst. at EUROMICRO*, vol. 190, 2006.

[84] S. Sendall and W. Kozaczynski, "Model transformation the heart and soul of model driven software development," Swiss Federal Institute of Technology in Lausanne, Tech. Rep., 2003.

[85] N. R. Jennings and M. Wooldridge, "Applications of intelligent agents," in *Agent Technology*. Springer, 1998, pp. 3–28.

[86] C.-Y. Chou, T.-W. Chan, and C.-J. Lin, "Redefining the learning companion: the past, present, and future of educational agents," *Comput. Educ.*, vol. 40, no. 3, pp. 255–269, 2003.

[87] D. Sleeman and J. S. Brown, *Intelligent Tutoring Systems*. London : Academic Press, 1982. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00702997

[88] A. De Antonio, J. Ramírez, R. Imbert, and G. Méndez, "Intelligent virtual environments for training: an agent-based approach," in *Multi-Agent Systems and Applications IV*. Springer, 2005, pp. 82–91.

[89] N. Capuano, M. Marsella, and S. Salerno, "Abits: An agent based intelligent tutoring system for distance learning," in *P. Adapt. Intell. Web-Based Educ. Syst.*, 2000.

[90] L. M. M. Giraffa and R. M. Viccari, "The use of agents techniques on intelligent tutoring systems," in *Lect. Notes Comput. Sc.* IEEE, 1998, pp. 76–83.

[91] M. Hospers, E. Kroezen, A. Nijholt, R. op den Akker, and D. Heylen, *An Agent-based Intelligent Tutoring System for Nurse Education*. Springer, 2003.

[92] M. Moundridou and M. Virvou, "Evaluating the persona effect of an interface agent in a tutoring system," *J. Comput. Assist. Learn.*, vol. 18, no. 3, pp. 253–261, 2002.

[93] E. Shaw, W. L. Johnson, and R. Ganeshan, "Pedagogical agents on the web," in *Proc. Third Conf. on Auton. Agent.* ACM, 1999, pp. 283–290.

[94] W. L. Johnson, J. W. Rickel, and J. C. Lester, "Animated pedagogical agents: Face-to-face interaction in interactive learning environments," *FR Art. Int.*, vol. 11, no. 1, pp. 47–78, 2000.

[95] Y. Kim and A. L. Baylor, "A social cognitive framework for pedagogical agents as learning companions," *ETR&D-Educ. Technol. Res.*, vol. 54, no. 6, pp. 569–596, 2006.

[96] M. Soliman and C. Guetl, "Implementing intelligent pedagogical agents in virtual worlds: Tutoring natural science experiments in openwonderland," in *EDUCON 2013 IEEE*, 3 2013, pp. 782–789.

[97] G. Clarebout, J. Elen, W. L. Johnson, and E. Shaw, "Animated pedagogical agents: an opportunity to be grasped?" *J. Educ. Multimedia Hypermedia*, vol. 11, no. 3, pp. 267–286, 2002.

[98] R. Moreno, R. Mayer, and J. Lester, "Lifelike pedagogical agents in constructivist multimedia environments: Cognitive consequences of their interaction," in *World Conf. Educ. Media Technol.*, vol. 2000, no. 1, 2000, pp. 776–781.

[99] E. André, T. Rist, and J. Muller, "Employing ai methods to control the behavior of animated interface agents," *Appl. Artif. Intell.*, vol. 13, no. 4-5, pp. 415–448, 1999.

[100] S. Domagk, "Do pedagogical agents facilitate learner motivation and learning outcomes? the role of the appeal of agents appearance and voice," *Media Psychol.*, 2010.

[101] G. Clarebout and J. Elen, "Open learning environments and the impact of a pedagogical agent," *J. Educ. Comput. Res.*, vol. 35, no. 3, pp. 211–226, 2006.

[102] S. Heidig and G. Clarebout, "Do pedagogical agents make a difference to student motivation and learning?" *Educ. Res. Rev.*, vol. 6, no. 1, pp. 27–54, 2011.

[103] K. Leelawong and G. Biswas, "Designing learning by teaching agents: the betty's brain system," *FR Art. Int.*, vol. 18, no. 3, pp. 181–208, 2008.

[104] J. Kaplan and N. Yankelovich, "Open wonderland: an extensible virtual world architecture," *IEEE Internet Comput.*, vol. 15, no. 5, pp. 38–45, 2011.

[105] M. Gardner, A. Gánem-Gutiérrez, J. Scott, B. Horan, and V. Callaghan, "Immersive education spaces using open wonderland from pedagogy through to practice," *Multiuser Virtual Environments for the Classroom: Practical Approaches to Teaching in Virtual Worlds*, pp. 190–205, 2011.

[106] M. Ibanez, C. Kloos, D. Leony, J. Rueda, and D. Maroto, "Learning a foreign language in a mixed-reality environment," *IEEE Internet Comput.*, vol. 15, no. 6, pp. 44–47, 11 2011.

[107] I. Branovic, R. Popovic, N. Jovanovic, R. Giorgi, B. Nikolic, and M. Zivkovic, "Integration of simulators in virtual 3d computer science classroom," in *Global Engineering Education Conference (EDUCON), 2014 IEEE*, 4 2014, pp. 1164–1167.

[108] D. Parsons and R. Stockdale, "Cloud as context: Virtual world learning with open wonderland," in *P. .9th World Conf. Mobile Context. Learn., Valetta, Malta*, 2010, pp. 123–130.

[109] J. Blair and F. Lin, "An approach for integrating 3d virtual worlds with multiagent systems," in *Advanced Information Networking and Applications (WAINA), 2011 IEEE Workshops of International Conference on*, 3 2011, pp. 580–585.

[110] A. Kobsa, "Generic user modeling systems," in *The Adaptive Web.* Springer, 2007, pp. 136–154.

[111] P. Brusilovsky, S. Sosnovsky, and O. Shcherbinina, "User modeling in a distributed e-learning architecture," in *Lect. Notes Artif. Int.* Springer, 2005, pp. 387–391.

[112] A. Mitrovic, "15 years of constraint-based tutors: What we have achieved and where we are going," *User Model. User-Adap.*, vol. 22, no. 1-2, pp. 39–72, 2012.

[113] P. Brusilovsky and E. Millán, "User models for adaptive hypermedia and adaptive educational systems," in *The Adaptive Web.* Springer-Verlag, 2007, pp. 3–53.

[114] E. Rich, *Stereotypes and User Modeling.* Springer, 1989.

[115] N. Li, W. Cohen, K. R. Koedinger, and N. Matsuda, "A machine learning approach for automatic student model discovery," in *Educ. Data Min.*, 2010.

[116] E. Frias-Martinez, S. Y. Chen, and X. Liu, "Survey of data mining approaches to user modeling for adaptive hypermedia," *IEEE Sys. Man. Cybern.*, vol. 36, no. 6, pp. 734–749, 2006.

[117] R. S. d Baker, A. T. Corbett, and V. Aleven, "More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing," in *Intelligent Tutoring Systems.* Springer, 2008, pp. 406–415.

[118] J. Vomlel, "Bayesian networks in educational testing," *Int. J. Uncertain. Fuzz.*, vol. 12, no. supp01, pp. 83–100, 2004.

[119] J. Xu and W. Croft, "Corpus-based stemming using cooccurrence of word variants," *ACM T. .Inform. Syst.*, vol. 16, no. 1, pp. 61–81, 1998.

[120] C. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval.* Cambridge, UK: Cambridge University Press, 2008.

[121] I. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *Lect. Notes Artif. Int.* ACM, 8 2001, pp. 269–274.

[122] C. Carmona and R. Conejo, "A learner model in a distributed environment," in *P. Adapt. Hypermedia Adapt. Web. Syst.*, Eindhoven, The Netherlands, 8 2004.

[123] N. Henze and W. Nejdl, "Student modeling in an active learning environment using bayesian networks," in *Lect. Notes Artif. Int.* Springer, 1999.

[124] W. Zangwill and P. Kantor, "Toward a theory of continuous improvement and the learning curve," *Manage. Sci.*, vol. 44, no. 7, pp. 910–920, 7 1998.

[125] S. Chapra and R. Canale, *Numerical Methods for Engineers.* Boston, MA: McGraw Hill, 2002.

[126] M. Myers, A. Jorge, and D. G. Walker, "A comparison of extended kalman filter approaches using non-linear temperature and ultrasound time-of-flight measurement models for heating source localization of a transient heat transfer problem," in *Inverse Probl., Design Optim.*, Joao Pessoa, Brazil, 8 2010.

[127] C. S. Fichten, V. Ferraro, J. V. Asuncion, C. Chwojka, M. Barile, M. N. Nguyen, R. Klomp, and J. Wolforth, “Disabilities and e-learning problems and solutions: An exploratory study,” *Educ. Technol. Soc.*, vol. 12, no. 4, pp. 241–256, 2009.

[128] T. Müller, “Persistence of women in online degree-completion programs,” *Int. Rev. Res. Open Distrib. Learn.*, vol. 9, no. 2, 2008.