

A GRAPH-BASED TAXONOMIC INTELLIGENT TUTORING SYSTEM UTILIZING
BLOOM'S TAXONOMY AND ITEM-RESPONSE THEORETIC ASSESSMENT

A Thesis/Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Ph.D.

in

Computer Science

by
Dennis Castleberry
B.S., LSU, 2009
May 2017

Acknowledgments

I would like to thank my research group, my committee, and the Department of Computer Science for supporting my teaching and research efforts throughout my career.

Table of Contents

ACKNOWLEDGMENTS	ii
LIST OF TABLES	v
LIST OF FIGURES	vi
ABSTRACT	viii
CHAPTER	
1 BACKGROUND INFORMATION	1
1.1 Introduction	1
1.1.1 Problem Definition	1
1.1.2 Novel Contributions	2
1.2 Bloom’s Taxonomy	2
1.2.1 The Interpretation in Computer Science	4
1.2.2 Assumptions of this Work	5
1.3 Item Response Theory	6
1.3.1 Evaluation of Trait Ability	12
1.4 Previous Literature	15
1.4.1 Works Mentioning Bloom’s Taxonomy and Item Response Theory	15
1.4.2 Categories of Systems	16
1.4.3 Memory and Forgetting	16
1.4.4 Executability	17
1.4.5 Reconciling Bloom’s Taxonomy with Item Re- sponse Theory	18
2 ITEMS AND SETS	21
2.1 Representing Items	21
2.1.1 Taxonomic Information	21
2.1.2 Difficulty vs. Bloom Level	23
3 TRAIT ABILITY	25
3.1 Representing Ability	25
3.2 Proximal Zone of Development	26
3.3 Updating Ability	31
3.3.1 Short Answer	31
3.3.2 Coding	31
4 ITEM DEPENDENCIES	33
4.1 Dependency Information	33
4.1.1 The Dependency Graph	35
4.1.2 Probability Estimates Using Dependees	38

5	MEMORY	44
5.1	Models of Memory	44
5.2	ACT-R	46
5.3	Alterations to Memory Model	47
5.4	Re-Activation	49
5.4.1	Spacing Effect	49
6	SCHEDULING	52
6.1	Item Scheduling	52
6.1.1	Preliminaries	52
6.1.2	Subsequent Assessments	53
6.1.3	Finding High-Impact Items	54
6.1.4	Finding the Whole Schedule	56
7	EXPERIMENTAL RESULTS	57
7.1	Experiments Supporting the Utility of the Taxonomy	57
7.1.1	Experiment 1 Design	57
7.1.2	Experiment 2 Design	58
7.1.3	Experiment 3 Design	59
7.2	Analysis and Results	60
7.2.1	Experiment 1 Analysis	60
7.2.2	Experiment 2 Analysis	60
7.2.3	Experiment 3 Analysis	60
7.2.4	Limitations of These Experiments	61
7.3	Experiments Supporting the Reconciliation of the Taxonomy with IRT	61
7.3.1	Experiment 1 Design	61
7.3.2	Experiment 2 Design	62
7.3.3	Experiment 3 Design	62
7.4	Analysis and Results	64
7.4.1	Experiment 1 Analysis	64
7.4.2	Experiment 2 Analysis	66
7.4.3	Experiment 3 Analysis	66
	REFERENCES	68
	APPENDIX	
A	APPENDIX A: IRB DOCUMENTS	72
	VITA	80

List of Tables

1.1	The levels of Bloom’s taxonomy defined	3
1.2	Grades mapped to trait ability levels.....	3
4.1	Notation for counts of joint observations	34
7.1	Proportions of variance due to questions for Experiment 1.....	64
7.2	Item means per Bloom category for Experiment 1 (M_1) and Experiment 2 (M_2)	64
7.3	The percent of students who answered the depender correctly given a particular set of questions (left), as well as the percent of students who answered the depender correctly given exposure to a given question (right).	67

List of Figures

1.1	A probability curve in Item Response Theory.....	7
1.2	A maximum-likelihood estimation for IRT	7
1.3	The standard normal curve; the distribution of test scores as modeled by Classical Test Theory	9
1.4	The PL1 Item Response Theory model, where $\beta=1$	9
1.5	The PL2 Item Response Theory model, where the item discrimination $\alpha = .5$ and $\beta = 0$	11
1.6	The PL3 Item Response Theory model, where $\alpha = 1$ and $\beta = 0$, and probability of guessing $\gamma = .25$	11
2.1	The database layout.	22
3.1	The trait ability lattice, which accounts for domain-specific interests	30
4.1	A forest obtained from an item set, where each tree is a subset of the item set which has dependency relationships	36
4.2	The severance of dependency relationships based upon low α values	36
4.3	The base item set graph, which includes item-specific parameters	37
4.4	The student-specific item set graph, which includes the list of student responses and timestamps for each response	37
4.5	A perceptron; a single-layer neural network, where the inputs a, b, and c are multiplied by weights, summed, and applied to a sigmoid squash function	42
4.6	A view of the dependency graph and weights used in the logistic regression, including the item parameters	42
5.1	The Ebbinghaus curve of forgetting, which features an exponential dropoff of memory strength over time	45
5.2	The modified curve of forgetting, which resembles the reverse sigmoid function	45
5.3	Forgetting with re-activation; each spike in the graph is an additional trial where the student is exposed to the item again	48

5.4	The learning curve, which indicates the extent of memory lifespan increase given a time between trials	48
-----	--	----

Abstract

This dissertation details an intelligent tutoring system which aims to raise student trait abilities across concepts and levels of Bloom's taxonomy within an educational program.

The system utilizes Item Response Theory, a probabilistic theory of assessment which accounts for item discrimination, difficulty, and probability of guessing in its assessment of student trait ability.

The intelligent tutoring system features a graph-based model of dependency relationships among questions, models of memory and forgetting, and fine-grained characterization-based models of student ability. In conjunction, these lend to efforts to schedule problems for the particular student which have the highest payoff.

Chapter 1

Background Information

1.1 Introduction

1.1.1 Problem Definition

A *content item* is some unit of information, such as a statement, graph, table, example, and so forth. An educational television program, interactive tutorial or university lecture (generally, these may all be called programs) can be broken down into these units of information. Call an item χ ; then the i^{th} item to be seen is χ_i , and the time at which it is seen t_i . Then (χ_i, t_i) represents the i^{th} item and its scheduled time.

For that matter, any program can be thought of as a sequence of these tuples, thereby forming a schedule of items:

$$X = \langle (\chi_1, t_1), (\chi_2, t_2), \dots (\chi_n, t_n) \rangle. \quad (1.1)$$

Furthermore χ could be a question, which could be thought of as an item which accepts a response back that has a correct answer. Items and questions share many similar properties which help to distinguish them, such as the ones in Fig ??.

The main question this body of work seeks to answer is this: *based on input from the student on any subset of questions in an item schedule, how should the remaining questions be scheduled?*

1.1.2 Novel Contributions

The answer to this question required a series of novel contributions to the fields of intelligent tutoring systems (ITS) and computer-aided assessment (CAT). Specifically:

- The creation of a data structure in the form of a graph, whose nodes contain information relevant to the assessment process, and whose edges capture dependency relationships among questions;
- A modification to an existing assessment theory known as Item Response Theory, which however providing a mature means of assessing student ability, benefited from an account of dependency relationships;
- A scheduler, or algorithm whose purpose was to determine what the questions should be given the item parameters and the students' response sets;
- An addendum to an existing theory of memory, forgetting, and practice, which could then be integrated into the scheduler to provide a fuller-featured system.

In addition to this, the body of work rests upon other incidental novel contributions, such as the separation of Bloom level and difficulty [?].

1.2 Bloom's Taxonomy

Bloom's cognitive taxonomy organizes questions into levels depending on the cognitive functions required of the answerer. The levels are: knowledge, comprehension, application, analysis, evaluation, and synthesis. A brief overview is given in Table 1.2, with definitions and examples of questions covering the concept of for-loops:

Table 1.1: The levels of Bloom’s taxonomy defined

Level	Explanation	Example Question
Knowledge	Recalling factual information.	What is a for-loop?
Comprehension	Assigning meaning to information.	What does the example for-loop output? (Give example.)
Application	Applying a rule to a specific instance.	How can the update statement of the loop be changed to print only even numbers?
Analysis	Breaking information into parts and exploring relationships.	What would happen if the update statement decremented instead of incremented the counter?
Evaluation	Judging the use of knowledge or the validity of an argument.	Which is better for reading user input: a for-loop or a while-loop? Why?
Synthesis	Utilizing knowledge to create a new solution to satisfy a goal.	Write a for-loop to print only even numbers up to ten.

Table 1.2: Grades mapped to trait ability levels

Letter	θ	Explanation
F	-3.0	unsatisfactory
D-	-2.5	minimal
D	-2.0	
D+	-1.5	
C-	-1.0	acceptable
C	-0.5	
C+	0.0	
B-	0.5	good
B	1.0	
B+	1.5	
A-	2.0	distinguished
A	2.5	
A+	3.0	

Each category depends on the cognitive functions used in the previous category. That is, comprehension requires knowledge, application requires comprehension, and so forth. Furthermore the mastery of one of the levels is with respect to a given concept. A student may be able to synthesize solutions to problems dealing with expressions, but may not possess knowledge of equations, and thus could not solve problems involving equations.

The utility of Bloom’s taxonomy lies in its ability to pinpoint the underlying cause of the student’s problem-solving impasses [24]. There is even evidence to suggest that Bloom levels correspond to different electroencephalographic (EEG) signatures [9], implicating distinct brain functions in their use.

Suppose a test of mastery of loops is given with the comprehension question “What does such-and-such loop output?” is given, and the student reaches an impasse. If the question “What are the three expressions of the loop and what do they do?” is asked and the student does not know, the impasse can be attributed to a lack of knowledge about loops. If the student does know about the loop expressions but still cannot answer, one might instead attribute it to a comprehension difficulty *as such*; which might be remedied by giving some examples to build intuition, then continuing to test at the comprehension level.

Educators may have an intuitive notion of how to do this, but Bloom’s taxonomy gives the ability to examine the impact of questions scientifically. By identifying the tested concept and the Bloom level of exercises, one can then form hypotheses about student responses to questions.

1.2.1 The Interpretation in Computer Science

Bloom’s taxonomy has been proven to be useful at the undergraduate level, and particularly in the field of software engineering [5, 21]. It has seen success in program comprehension [6], where the asking of comprehension questions fosters code reading [15]. In addition,

it has been useful for pinpointing the difficulties of novice programmers in a guided learning approach [24]. It been used to identify a marked preference for higher-level problems for those able to solve them [11] [14]. At least two experiments have shown the effect of item ordering on performance [20, ?], and the taxonomy has even been applied to create ratings of courses based on the average Bloom levels of tasks and questions in the course [22].

In spite of all this, there is an ongoing debate regarding the applicability of Bloom’s taxonomy to computer science [17, 12, 26]. The crux of this debate centers around the interpretation of Bloom levels: not only how questions map to Bloom levels, but also regarding the progression of Bloom levels over the span of a course or curriculum.

A tacit assumption in much of the research is that Bloom levels equate to difficulty levels. While there is certainly a relationship between Bloom level and difficulty in the “ordinary course” of devising problems, a distinction can be made between the two.

1.2.2 Assumptions of this Work

Prior research by the author supports the view that Bloom level and difficulty are separable parameters of a content item. This work will proceed on the assumption to that effect.

In addition, some of the components of this body of work, particularly those pertaining to memory and recall, are supported by many decades of empirical research [?]. While the study of intelligent tutoring systems has seen attempts to validate addendums to these theories which accommodate re-activation of forgotten knowledge, none have undergone extensive empirical testing as is typical for psychological models. The components of this body of work regarding re-activation introduce hypotheses.

1.3 Item Response Theory

Here is introduced a mature assessment theory known as Item Response Theory, an alternative to Classical Test Theory (CCT). Whereas Classical Test Theory assigns a student a grade based on the student's position in a distribution of composite test scores, Item Response Theory accounts for item difficulty, item discrimination, the probability of guessing the question correctly.

One of the main appeals of IRT is its incorporation of item difficulty. As will be shown later, this is pertinent to the interpretation of Bloom levels.

According to Item Response Theory (IRT), the probability p_i that a student answers correctly the i^{th} question on a test, is given by:

$$p_i(\theta_s) = \gamma_i + \frac{1 - \gamma_i}{1 + e^{\alpha_i(\theta - \beta_i)}} \quad (1.2)$$

where:

- α is the item discrimination, or how well the item can distinguish students of varying trait ability;
- β is the question difficulty,
- γ is the probability of guessing the answer correctly,
- and θ is the *trait ability* of the student, or the student's particular ability to answer that question correctly.

A graph of the IRT curve for the parameters $\alpha = 1, \beta = 0, \gamma = 0$ is shown [1.3](#).

Note that α_i, β_i , and γ_i are parameters of the item i ; however θ_s is particular to the student s .

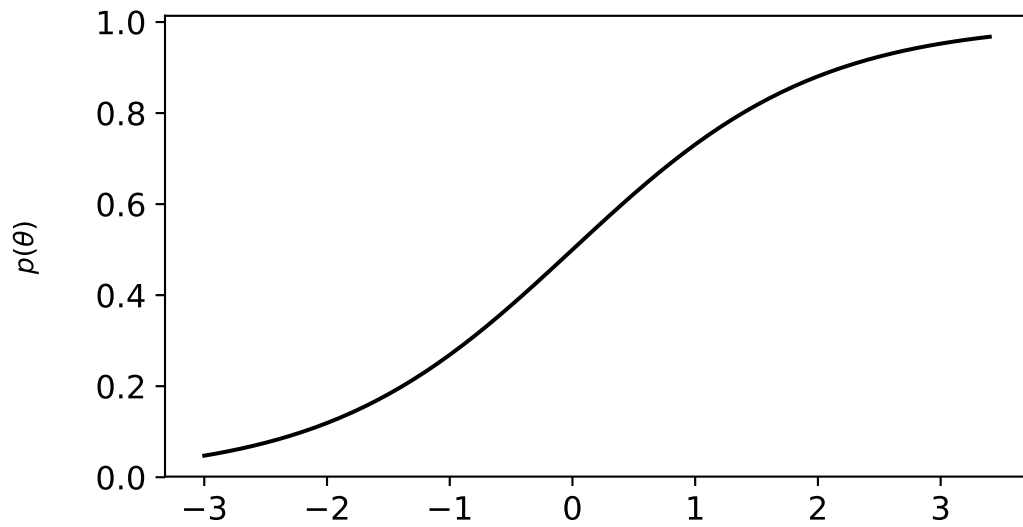


Figure 1.1: A probability curve in Item Response Theory.

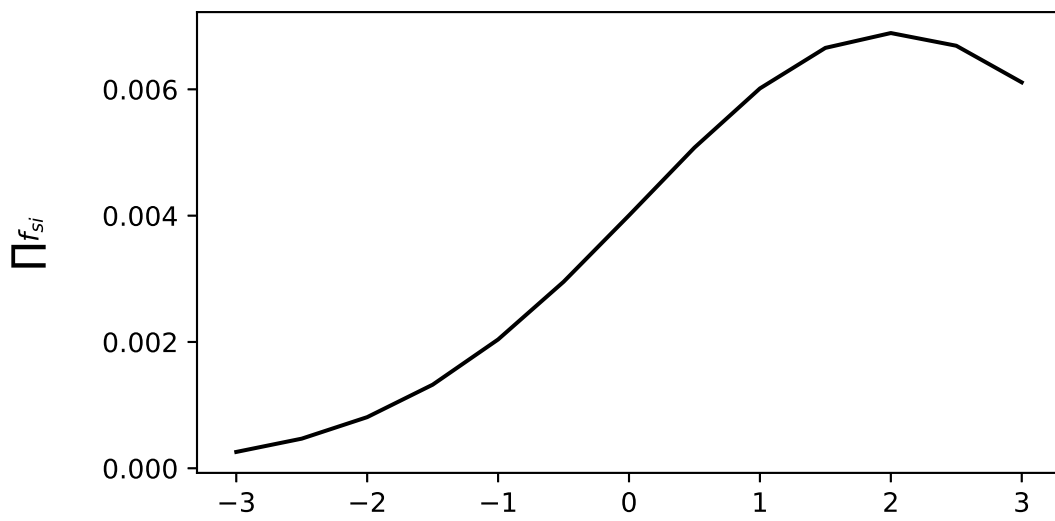


Figure 1.2: A maximum-likelihood estimation for IRT

The question difficulty β can be estimated by the proportion of students who pass the question. The value β is analogous to standard deviation in classical test theory. If a model of student ability based upon the number of standard deviations from the mean is desired, and the scores are normally distributed, then the β values can simply be obtained from the probability density function for a Student's t distribution.

Supposing ϕ is the proportion of students who pass the question, $f(v, x)$ is the probability density function for the Student's t-distribution with v degrees of freedom then the relationship between ϕ and β in such an assignment would be given by

$$\phi = \int_{\beta}^{\infty} f(v, x) dx \quad (1.3)$$

That is, β is the number of standard deviations such that the area from β to ∞ under the curve is equal to the proportion of students who pass the question. The value can also be obtained from a t-table.

Insofar as the intelligent tutoring system is concerned, the instructor or user is free to assign β . Varying assignments of the β scores will slightly alter the behavior of the system; if a given question is tagged as easier (that is, assigned a lower β value), it will have a greater chance of appearing earlier in any given student's schedule.

The item discrimination, α_i , is defined as the Pearson product-moment correlation coefficient between the item responses and the composite score on the test. Since the item responses are dichotomous, a correlation known as the point-biserial correlation is computed. It is defined as:

$$\alpha_i = \frac{M_1 - M_0}{s_n} \sqrt{\frac{n_0 n_1}{n^2}} \quad (1.4)$$

where M_1 is the mean composite score for all correct responses; M_0 is the mean composite score for incorrect responses; n_0 , n_1 , and n are number of incorrect, correct, and

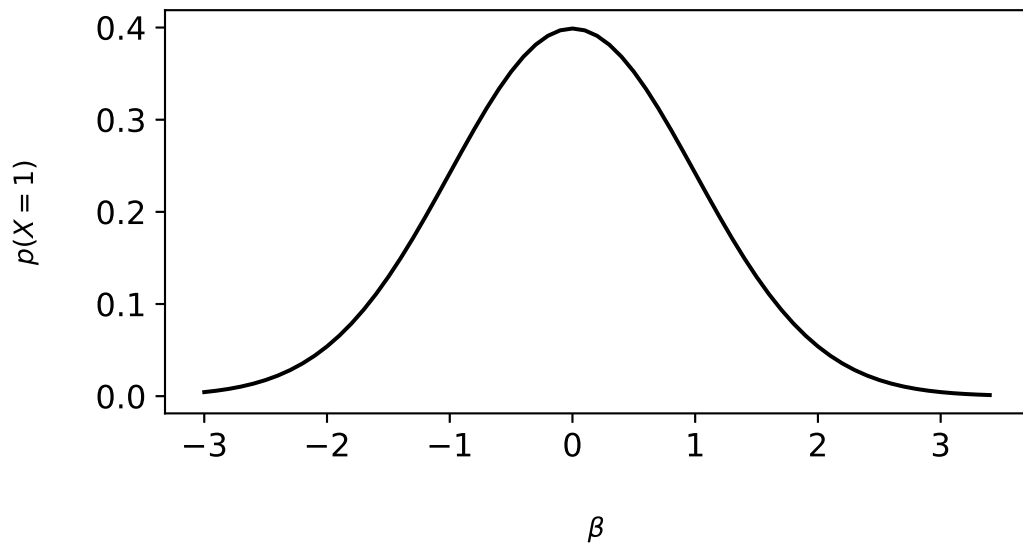


Figure 1.3: The standard normal curve; the distribution of test scores as modeled by Classical Test Theory

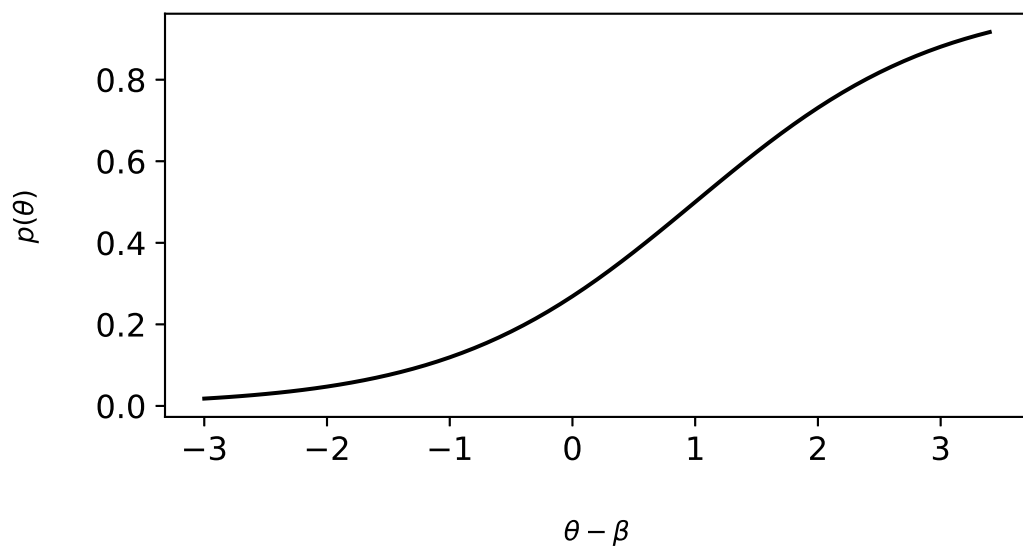


Figure 1.4: The PL1 Item Response Theory model, where $\beta=1$

total responses, respectively; and the standard deviation s_n is defined as:

$$s_n = \sqrt{\frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2} \quad (1.5)$$

where X_i is the i th response and \bar{X} is the mean of the responses. α values have range $[-1, 1]$. Positive values indicate that the question is positively discriminating; it is a question which is useful for gauging the constructs which the composite score seeks to measure. Students who succeed on a question with a higher α are more likely to have higher composite scores. Likewise, negative values indicate that the question is inversely related to composite score. Values near zero indicate that the question is unrelated.

The probability of guessing is defined as

$$\gamma_i = \frac{1}{m_i} \quad (1.6)$$

where m_i is the number of options in a multiple choice question. It is assumed that each option is equally likely to be chosen if a student with very low trait ability were to answer the question; that is, there can be no allowance for logical process-of-elimination tactics.

If the trait ability of the student is known in addition to the item parameters, then the probability of a correct response can be calculated. However, it is more often than not the case that the response set of the student is known, and trait ability is unknown. In such a case, a variety of techniques are deployable.

Trait ability in IRT can be obtained using a maximum likelihood estimation (MLE) method, which finds a maximum-likelihood estimate of θ by testing a range of θ values with the IRT formula [2]. Later it will be shown how θ can be estimated. Another potential

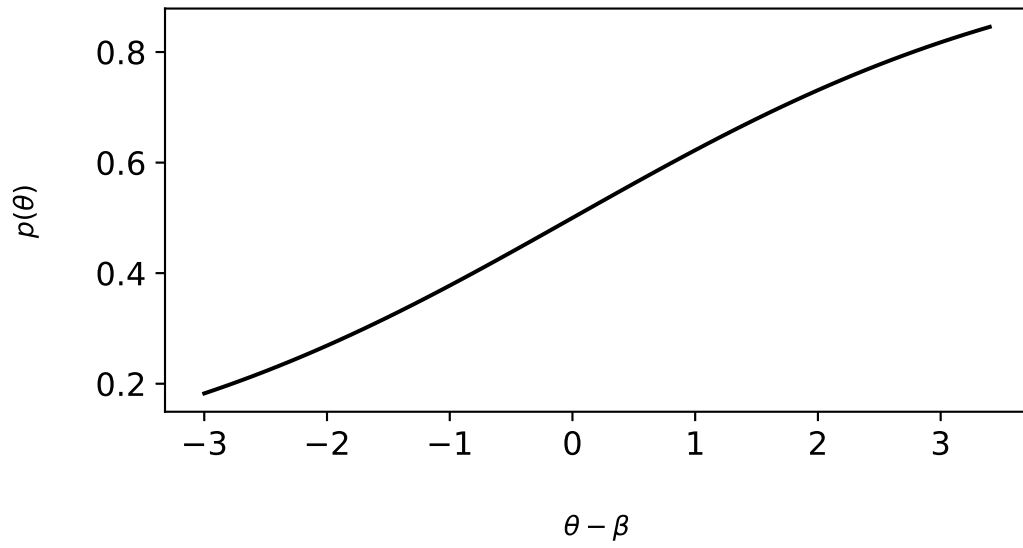


Figure 1.5: The PL2 Item Response Theory model, where the item discrimination $\alpha = .5$ and $\beta = 0$

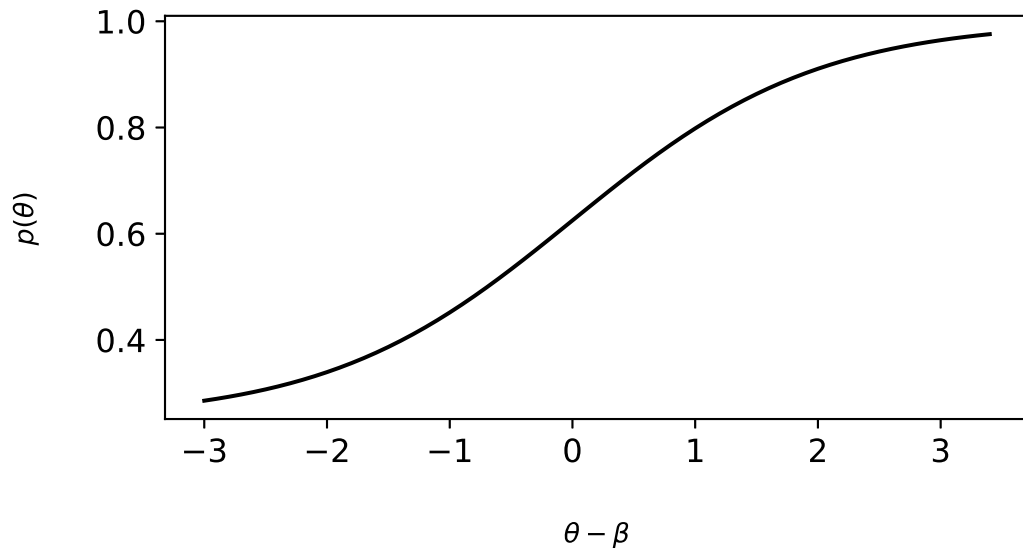


Figure 1.6: The PL3 Item Response Theory model, where $\alpha = 1$ and $\beta = 0$, and probability of guessing $\gamma = .25$

technique is Newton-Raphson, which in most other applications would be the more efficient and desirable method [2]. It will later be discussed why the MLE method is preferred.

Until now, the fusion of IRT and Bloom’s taxonomy has only existed in the literature as a possibility [25]. This work seeks to reconcile the two by offering a compatible interpretation of Bloom’s taxonomy.

1.3.1 Evaluation of Trait Ability

Consider a set of content items or questions for which the answer is either incorrect or correct. This is true in the case of multiple choice questions (as well as true-false, which is a subset of multiple-choice questions).

Suppose the student has a response set for content items:

$$X_s = x_{s1}, x_{s2}, \dots, x_{si}, \dots x_{sn} \tag{1.7}$$

Here X_s is the response vector of student s , and x_{si} is the correctness of the response to content item i by student s ; it is zero if the response is incorrect, or 1 if it is correct.

These questions may be of various and sundry discriminations, difficulties, and probabilities of guessing. The first question may have $\alpha_1 = .5$, $\beta_1 = 1$, and $\gamma_1 = .25$; the second may only differ in its difficulty, for example $\beta_2 = -1$. However, it shall be assumed that the set of content items for which the student has provided responses are for the same concept and the same Bloom taxonomic level. The reason for this assumption will become apparent later.

Recall that the probability p_{si} of student s answering the i^{th} question correctly is:

$$p_{si}(\theta_s) = \gamma_i + \frac{1 - \gamma_i}{1 + e^{\alpha_i(\theta_s - \beta_i)}} \quad (1.2)$$

Since θ_s is unknown but the response set is known, one method for determining θ_s is by guessing a range of possible values. First, one can define a function f_{si} :

$$f_{si}(\theta_s) = \begin{cases} p_{si}(\theta_s) & \text{if } x_{si} = 1 \\ q_{si}(\theta_s) & \text{otherwise} \end{cases} \quad (1.8)$$

where

$$q_{si}(\theta_s) = 1 - p_{si}(\theta_s). \quad (1.9)$$

That is, f_{si} assumes the probability p_{si} if answered correctly and q_{si} if not answered correctly. Proceeding on the assumption that each of the observations (that is, responses in the response set) is independent, the probability of observing a total response set given a particular θ_s value is the product of the probabilities f_{si} for all i , $1 \leq i \leq n$, or

$$\prod_{i=1}^n f_{si}(\theta_s). \quad (1.10)$$

Supposing that there exists some θ_s which maximizes this product, the most likely value for the student's true trait ability θ_s is defined by:

$$\theta_s = \underset{\theta}{\operatorname{argmax}} \left[\prod_{i=1}^n f_{si}(\theta) \right] \quad (1.11)$$

that is, that value of θ_s which maximizes the product which gives the probability of all the observations occurring together, given θ . To obtain this, products for a range of possible θ values are calculated.

In the intelligent tutoring system which has been constructed, there are only thirteen such values, drawing from the +/- system. The mapping of grades to trait ability levels is given in Table 1.2. This mapping could also be applied to difficulty levels of questions. A “C question” is one which students of average trait ability (and perhaps just more than half the class) may be expected to answer; an “A+ question” is a very difficult question which students of only A+ ability at the time of asking may be expected to answer; and an “F question” is one which may be used to determine if a student’s trait ability is minimally satisfactory. This mapping lends to a intuitive understanding of trait ability—as the familiar letter grade. An MLE may thereby effectively grade the student.

To that end, rather than using a fine-grained MLE, it makes practical sense to calculate products for these thirteen values of θ , since higher granularity than the +/- system is not useful for final grade assignment, nor is necessarily more intuitive for the student or instructor. Restricting the products calculated to this set of values also makes sense from an efficiency standpoint. The total time complexity of the MLE is linear in the size of the response set, regardless of the number of guesses in θ ; while not asymptotically reduced from using a smaller step size, it is five times faster than the typical choice of 10^{-1} , and fifty times faster than the fine-grained 10^{-2} . A graph calculating likelihoods for MLE is depicted in Figure 1.3.

The total number of MLE estimations for trait ability throughout a course is equal to the total number of responses for all questions throughout a course, which can be quite large; it is proportional to the number of students times the number of items.

1.4 Previous Literature

1.4.1 Works Mentioning Bloom’s Taxonomy and Item Response Theory

There has been one suggestion by Sitthisak et al. linking Item Response Theory with Bloom’s taxonomy which interprets Bloom levels as difficulty levels. That is, the values of β from Item Response Theory are mapped directly to Bloom levels [25]. This suggestion exists in the form of a proposal to implement a computer-aided tester using Item Response Theory.

Another study identifies difficulty by levels of Bloom’s taxonomic levels, and evidently applies Item Response Theory to scoring of the same questions; however it is unclear how the Bloom levels are mapped to Item Response Theory β values, or whether or not the mentioned difficulties were hypothesized or actual difficulties [23].

It is discussed in Sec 1.4.5 why it is desirable for the current intelligent tutoring system to main difficulty and Bloom levels as separate entities.

Ghulman et al. may be the closest to a full separation of Bloom’s taxonomy and difficulty, as well as establishing the connection between Bloom’s taxonomy and Item Response Theory, as it categorizes questions by Bloom level before applying a Rasch model to score them [13]. The Rasch model is a model simpler in nature than Item Response Theory which uses one parameter and is derived from the data set itself.

1.4.2 Categories of Systems

Conole and Warbuton have developed a classification system for such computer-aided testing tools. Web-based systems [27] fall under networked systems, which in turn fall under computer-based assessment utilities, which in turn falls under computer-assisted assessment [10]. Conole and Warbuton identify that Bloom’s taxonomy is often used as an outcomes-based categorization tool for questions, and that Classical Test Theory and Item Response Theory are the two main ways in which computer-aided systems score questions; however these are mentioned in isolation.

Bejar distinguishes between two types of systems: deficit assessment and error analysis [3]. Deficit assessment is concerned with finding where a student is lacking (such as in trait ability), and error analysis with what steps in the problem-solving process the student is making errors in. This distinction is similar to Anderson’s characterization-based models versus process-based models in intelligent tutoring systems; where characterization-based models attempt to create a representation of the student’s strengths, process-based models attempt to simulate the student’s problem-solving processes [?]. The current work would fall squarely within the realm of Bejar’s deficit assessment-type system, or Anderson’s characterization-based model.

1.4.3 Memory and Forgetting

Ebbinghaus is credited with having created one of the first mathematical memories of remembrance and forgetting [?]. In particular, he identified that recall drops off exponentially with time. Empirical studies since then have supported his original finding.

John Anderson developed a model for process-based learning which could provide the foundation for an intelligent tutoring system [?]. He called this Adaptive Control

of Thought-Rational (ACT-R). In ACT-R, there are goals, akin to problem statements; and rules, or processes used to solve problems; and finally facts, or knowledge utilized in the course of applying rules. In this regard, the structure of an ACT-R model resembles a logic program. Anderson also developed a memory model for ACT-R fact and rule recall loosely based on Ebbinghaus' recall functions, as well as models of re-activation of knowledge generalized from his own empirical research.

[1]

1.4.4 Executability

Finally, some supplementary work on the executability component of the intelligent tutoring system has been done [7]. One component of the intelligent tutoring system allows for arbitrary code to execute before a question or item is displayed to the student. This feature was inspired by earlier work on what are known as executable papers. Executable papers are academic publications residing inside of a virtual (or other) machine which have editable codes on the front-end and compilers or interpreters on the backend. The front-end codes may be edited, then the paper may be re-compiled so that results from the codes are substituted back into the paper.

This idea was originally intended for use with computational science codes, but was adapted for statistics codes [?]. It has been integrated into the present work to allow for sophisticated random-number generation, and for the nature of the questions and items to change in response to the student or other contextual data.

1.4.5 Reconciling Bloom’s Taxonomy with Item Response Theory

A popular interpretation of Bloom’s cognitive levels is that they are difficulty levels, and that these difficulty levels are fixed [20, 22, 19, 17, 12]. Knowledge is easy; synthesis is hard. We will call this the Bloom-equals-difficulty hypothesis.

A creative interpretation of this hypothesis is that per-student difficulty can be explained in terms of the demotion of Bloom levels. The synthesis *and therefore difficult* questions, once the solution is obtained, are reduced to knowledge *and therefore easy* questions. Certainly, this is one possible scenario. However, it does not explain a student’s ability to answer altogether distinct synthesis problems; this is to suggest students engage the cognitive functions of synthesis, presumably as an effect of learning the use of those functions. To maintain that Bloom equals difficulty would either require sustaining the view that students are able to pass the questions because of a demotion of the cognitive functions required (which poses a rather cynical view of cognition and of education); or else some notion of trait ability—in particular one which offsets difficulty, allowing an increase in the probability of passing the question.

The latter option takes the shape of a primeval form of Item Response Theory, mapping Bloom levels to β . This is highly convenient for the hypothesis that Bloom equals difficulty, as it becomes agreeable to evaluation with Item Response Theory. Nonetheless, there are still several issues with the hypothesis.

Consider first that exposure to knowledge and knowledge questions causes an increase in θ , provided the student acquires the knowledge. According to Bloom’s theory, this would increase the probability of answering comprehension questions correctly. This agrees with the theory. However, according to Item Response Theory, it would also increase the probability of answering application questions correctly even if no comprehension-level

questions are asked. In fact, it is theoretically possible to design tests with sufficient numbers of knowledge questions to place, for any β , $\theta - \beta$ much greater than zero—that is for example to say, asking a sufficient number of knowledge questions should give the reasonable assurance that application-level ability is high enough not to bother testing it.

As many educators will no doubt agree, this is not the case. Even if a student scores exceptionally well on knowledge questions, it gives no such assurance that the student will score similarly on application-level questions. Even if knowledge and application scores are found to be correlated, there may be an additional underlying factor contributing to both scores (such as intelligence). The application level is *qualitatively* different; hence it needs to be tested in spite of the value of θ .

Second, if the hypotheses is strictly true, then there can exist no counterexample to the claim. That is to say that there exists no problem on a lower Bloom level which is more difficult than a problem on a higher Bloom level. To take an example, it would be mistaken to call an application problem easier than any comprehension problem.

Counterexamples with intuitive appeal can be constructed. Take for-loops for example. To ask a student to print out the result of a simple loop which prints numbers 1 to 10 should present an easy enough task. This is an application of the many rules of for-loops, but which does not require creative synthesis or critical thinking on the part of the student (hence it is an application problem).

Consider then asking the student to impose a flowchart over the same code, including the initialization, condition and update statements and the body of the loop, and to indicate the start and stop of the loop. Suppose the student has seen flowcharts over similar loops before. This is a test of comprehension—in particular, comprehension of the internal workings (the control flow) of the loop.

The application problem above requires only an intuitive comprehension of the loop: “*It prints the sequence from 1 up to 10 in steps of 1*”, the student may realize. It demands only shallow comprehension and the rule to be applied is simple. In the comprehension problem,

on the other hand, a higher degree of precision in comprehension of the loop is demanded to solve the problem. The comprehension problem also depends on mastery of the knowledge, comprehension, and application of flowchart symbols. Regardless, explaining the control flow of the loop in full detail is a more difficult undertaking than simply printing its output.

Likewise, it is possible to ask the student to synthesize a loop printing the first ten powers of two, then ask a comparatively difficult analysis question in the form of an obfuscated loop code. Impasses in this situation may be attributed to a lack of knowledge or comprehension of the constructs used in the presented code. In the synthesis problem, the student has the advantage of using known syntax; it is a “constructed-response” type of question, rather than a closed-form question [18].

Third, by Bloom’s own admission, it is possible to skip levels for certain concepts [4]. In this case, the ordinality of Bloom levels breaks down. A plausible example is given above. It is possible to teach how to obtain the numeric sequences printed by for-loops before covering in full detail the control flow of the loop. In that particular case, application of the rule does not depend on comprehension of the code construct.

Alternative interpretations of Bloom levels distinguish between facility or difficulty and complexity of a problem [16, 26]. In such interpretations, the cognitive complexity has an orthogonal relationship to the item difficulty. The problem with such interpretations, however, is that they do not readily explain the moderate correlation that does exist between mean performance and Bloom level [16].

Chapter 2

Items and Sets

2.1 Representing Items

A database was used to represent items, item sets, students, and student responses. A full diagrammatic representation of the database is shown in Figure 2.1.

The database contains items of the following nature:

- *Content*. These include facts (which may be arranged into paragraphs, subsections, and sections), definitions, diagrams, and source codes.
- *Assessment*. These include questions, which may be true/false, multiple-choice, code writing, code simulation, short answer, and freewriting; also Likert scale items.

2.1.1 Taxonomic Information

Any output from this database to the student which is intended to solicit input from the student has the following ordinal dimensions:

- *Difficulty*. Following the +/- grading system, difficulties range from -3 (very easy) to +3 (very hard), with 0 being medium. As alluded to earlier, difficulty determines the probability that a given student in the class will answer the item correctly.
- *Bloom level*. The Bloom levels of cognitive functioning are Knowledge, Comprehension, Application, Analysis, Evaluation, and Synthesis.
- *Concept*. Concepts covered in computer science courses; for example in a programming course, these may include: Variables, Expressions, Control Structures, etc.

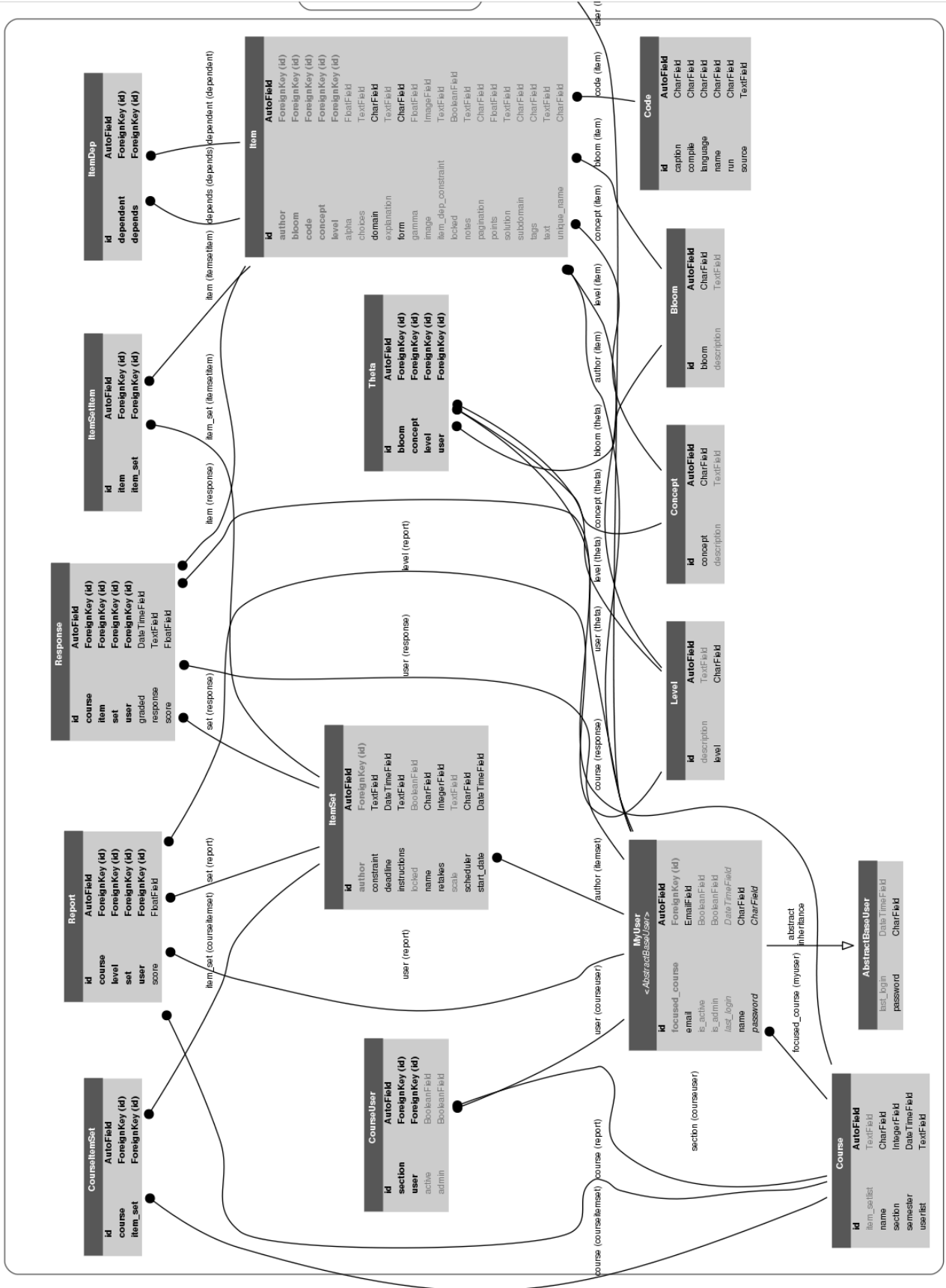


Figure 2.1: The database layout.

Any output to the student also has the following categorial dimensions:

- *Context.* A problem may have a domain-specific context, e.g. it may be a problem relevant to biology, chemistry, physics, mathematics, etc.
- *Type.* The problem may be true/false, multiple-choice, short answer.

Each output may also have a dependency list, that is a list of IDs of, or a rule describing, other output entries which the student should be exposed to prior to that output. For this system, Bloom level, subject domain, concept, and difficulty are dimensions of a test question.

It is important to note that the dimensions of the data are not necessarily limited to the above; an exploratory factor analysis (EFA) could be used to extend the dimensionality of the data semi-automatically. This functionality has not been implemented, but there exists a potential for it.

2.1.2 Difficulty vs. Bloom Level

With this, an alternative hypothesis is introduced: the Bloom cognitive taxonomic level and the difficulty (β value) are two separate properties of items. Rather than being an indicator of difficulty, the Bloom level indicates a cognitive dependency relationship to items at lower Bloom levels.

This is made clearer by the fact that Bloom levels are hierarchical: knowledge is required for comprehension, comprehension is required for application, and so forth. At any level, there exists the potential to create items of a range of difficulties.

However, the application of Bloom's taxonomy to problems is specific to the concept being analyzed, and moreover specific to the questions being asked. The application of mathematical expressions (such as being able to evaluate or reduce expressions) is different from the application of programmatic expressions (being able to evaluate or reduce such

expressions, bearing in mind the language-specific rules). The application of mathematical expressions depends on the comprehension of mathematical expressions, however the application problems may be easier once the dependencies are satisfied.

Thus for every concept, and for every Bloom level, a student may possess a different trait ability (θ value). This explains why students would be able to skip Bloom levels; in particular this would occur if a question of a higher Bloom level (e.g. Evaluation) did not depend on the Bloom level immediately preceding it (e.g. Analysis) but rather some lower level (e.g. Application).

A series of experiments have been conducted by the author toward the end of supporting the hypothesis that Bloom levels represent dependency relationships rather than difficulty levels [8]. These experiments consisted of demonstrating various ordering effects. In the experiments, questions of different Bloom levels (Knowledge through Evaluation) were asked in forward and reverse order. It was shown that forward-order schedules resulted in statistically significantly greater performance than reverse-order schedules [8]. These experiments are detailed in Sec 7.1.

Chapter 3

Trait Ability

3.1 Representing Ability

To form a statistical basis for content scheduling, the measure of trait ability should be done per-concept and Bloom level. Therefore if there are m concepts and n levels, there are then nm number of θ values. This shall be called Θ , the trait ability matrix; and Θ_s will denote the trait ability matrix of student s . Let j be the index of a Bloom level and k be the index of a concept, then:

$$\Theta_s = \begin{bmatrix} \theta_{s11} & \dots & \dots & \dots & \theta_{sn1} \\ \vdots & & \ddots & & \\ \vdots & & & \theta_{sjk} & \\ \vdots & & & & \ddots \\ \theta_{s1m} & & & & \theta_{snm} \end{bmatrix} \quad (3.1)$$

While not strict, there is certainly an ordering about Θ . Lower-level concepts come before higher-level concepts, and lower Bloom levels come before higher Bloom levels. Conceivably a Θ may look like the following:

$$\Theta_1 = \begin{bmatrix} 3 & 2.5 & 1 & 0 & -1 & -2 \\ 2 & 1.5 & 0 & -0.5 & -1.5 & -2.5 \\ 1 & .5 & 0 & -1 & -2 & -3 \\ 1 & 0 & -0.5 & -1 & -2.5 & -3 \end{bmatrix} \quad (3.2)$$

Highlighted are areas where $\theta_{sjk} = 0$. These are the areas where the student has roughly

.5 probability of answering a question at difficulty $\beta = 0$ correctly. Now the question arises: given a rich content item set with questions in all (Bloom \times concept \times difficulty) categories, which categories should be selected? Many factors are taken into account.

3.2 Proximal Zone of Development

Clearly the higher θ_{sjk} values should be left alone, particularly those nearing 3, since this demonstrates exceptional mastery of that (Bloom \times concept) category. In particular, if $\theta_{sjk} = 3$, there is no sense in asking at all since trait abilities are capped at 3. In probabilistic terms and relative to questions, asking questions for which the estimated probability is overwhelmingly high, for example $p > .9$, serves no purpose, since probability estimates of that degree require $\theta - \beta$ significantly greater than 0.

If $\theta_{sjk} = x$ then it would be “unfair” to ask questions in that category which have $\beta > x$. According to Equation 1.2, if $\beta > x$ and $\theta_s < x$, then $\theta - \beta < 0$ and therefore $p(\theta_s) < .5$, which means the student has less than .5 probability to answer correctly. Asking questions for which $p(\theta_s) < .5$ has psychological ramifications, and potential problems for the updated MLE of θ_s .

It is true that correct answers of more challenging questions raise the ability returned by a maximum likelihood estimate, however if the probability of answering them is consistently less than chance and the student responds accordingly, it is unlikely the estimate will return any $\theta_{sjk} > x$ for those problems. Also, given the information Θ_s about the student, if it is known that the particular student will more than likely fail a particular question, it would not make sense to ask it from a psychological standpoint, provided that the intention of asking is to raise trait ability levels. If the student consistently experiences more failures than successes, the student is more likely to be discouraged by the testing.

There is another consideration: the concept tier and Bloom levels. The course is a

progression of concepts across Bloom levels. There is evidence to suggest that students prefer problems with Bloom levels as high as they are able to solve [14]; and ideally, the student should see steady progress in the concepts of the course. The set of questions asked at any given time in a course of study typically range over a subset of concepts. Testing for all concepts begins at the Knowledge level; as new concepts are introduced over time, the tested Bloom level for earlier-introduced concepts rises. In a “perfect” situation, we might observe a trait ability matrix as in Equation 3.3, which shows a clear diagonal reflecting a progression in both concepts and Bloom levels.

$$\Theta_2 = \begin{bmatrix} 3 & 3 & 2 & 1 & 0 \\ 3 & 2 & 1 & 0 & -1 \\ 2 & 1 & 0 & -1 & -2 \\ 1 & 0 & -1 & -2 & -3 \\ 0 & -1 & -2 & -3 & -3 \end{bmatrix} \quad (3.3)$$

With all this in mind, the most logical subset of (Bloom \times concept) categories to select questions from are those in the neighborhood of $\theta_{sjk} = 0$. One interpretation of this subset is that it is the student’s proximal zone of development. In the psychology of learning, the proximal zone of development is the area or areas in which a student can perform a task with assistance, but could not perform the task without assistance. This is consistent with $p \approx .5$.

However, any question can potentially be asked for any (Bloom \times concept) category while still placing p at or just slightly above .5 by manipulating difficulty. In fact, a matrix of difficulties for desired target questions could be calculated from the ability matrix:

$$B_s = \Theta_s - \delta \quad (3.4)$$

where δ is a sufficiently low value, such as .5. In the case of student $s = 1$, B would then equal:

$$B_1 = \begin{bmatrix} 2.5 & 2 & 1 & -.5 & -1.5 & -2.5 \\ 1.5 & 1.5 & -.5 & -1 & -2 & -3 \\ .5 & 0 & -.5 & -1.5 & -2.5 & -3.5 \\ .5 & -.5 & -1 & -1.5 & -3 & -3.5 \end{bmatrix} \quad (3.5)$$

Then, it is possible to threshold the matrix so as to eliminate (Bloom \times concept) categories after this stepladder, but include some lag up to and not including those β_{sjk} values for which the student has θ_{sjk} indicating distinguished mastery:

$$B_1 = \begin{bmatrix} & 2 & 1 & -.5 \\ 1.5 & 1.5 & -.5 \\ .5 & 0 & -.5 \\ .5 & -.5 \end{bmatrix} \quad (3.6)$$

From here, the density of questions asked may be in proportion to the distance from -.5. Eventually, either the student will reach $\theta_{sjk} = 3$ or else the tutoring system will run out of questions for that (Bloom \times concept), in which event the trait ability level will stay.

Sometimes the trait ability matrix may be jagged:

$$\Theta_1 = \begin{bmatrix} 3 & 3 & 2 & 0 & -1 & -2 \\ 3 & 2.5 & 1 & -0.5 & -1.5 & -2.5 \\ 3 & 3 & 2.5 & -1 & -2 & -3 \\ 2 & 1.5 & -0.5 & -1 & -2.5 & -3 \end{bmatrix} \quad (3.7)$$

in which event, the scheduler discussed in Sec 6.1 will attempt to correct the trait ability matrix by asking questions in such a manner as to have the proximal zone of development conform to a diagonal shape.

It is possible, in fact, to have a three-dimensional structure as depicted in Fig 3.2. In the original conception of this work, the scheduler was intended to support not only progressions along the (concept \times Bloom) categories, but also to identify subdomains of interest. Introductory-level computer science invites many creative opportunities for assigning interdisciplinary problems; for example coding problems related to biology, chemistry, physics, mathematics, and even digital art. It was assumed that assigning problems catering to the students' interests would raise the incentive to learn the concept.

While the features of the scheduler were not extended to support the capability to “zero in” on the student’s domain-specific interests, the database problem taxonomy supports tagging problems by subdomain. The trait ability matrix may be extruded to include this interest dimension.

This structure, Θ , therefore shows where to begin asking questions. It also helps to define the goal of the intelligent tutoring system: to increase values in Θ successively along its diagonal, thereby engineering an experience similar to course progression.

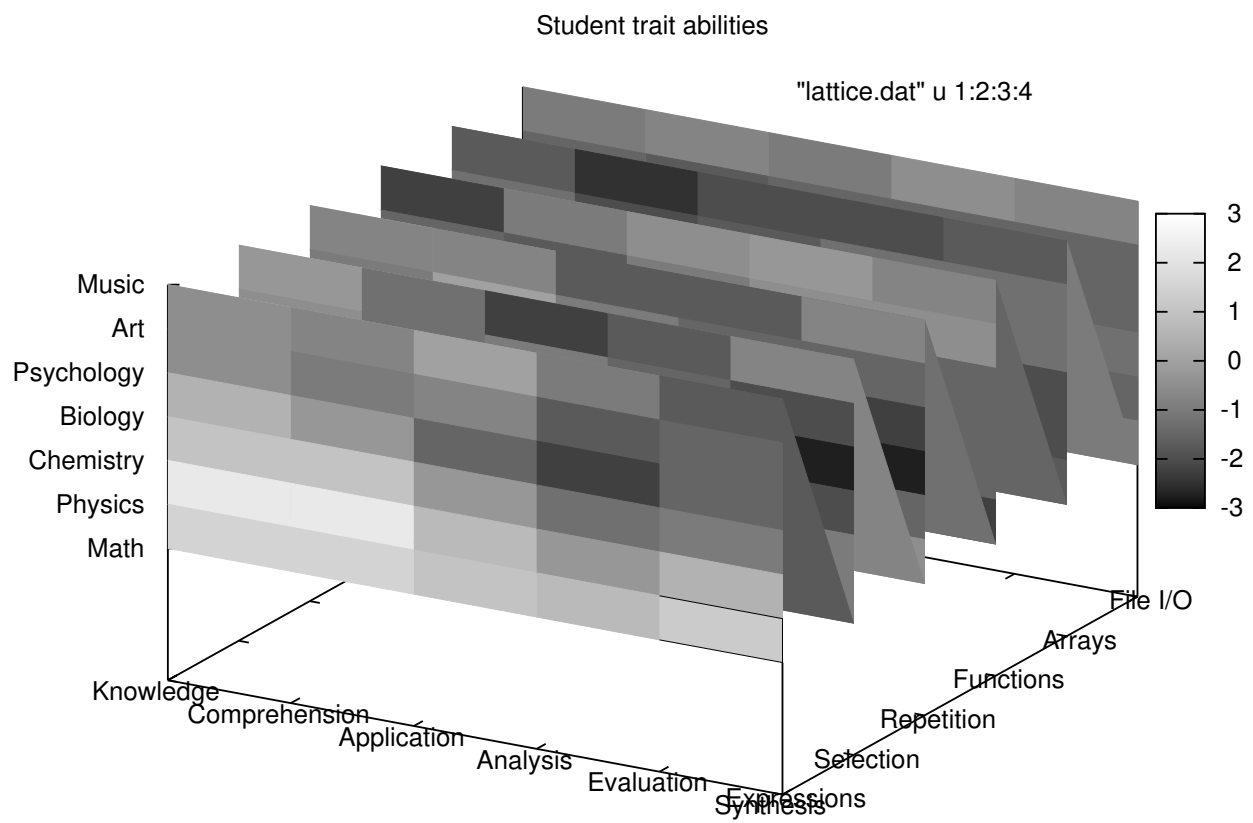


Figure 3.1: The trait ability lattice, which accounts for domain-specific interests

3.3 Updating Ability

The trait ability matrix must be refreshed (either after a student response, for a more fine-grained scheduling, or after a total assessment for course-grained scheduling). Therefore the student responses must be auto-graded, and an MLE for each element in Θ must be performed.

3.3.1 Short Answer

For short answer questions, γ is set to 0, since it is assumed that there are infinitely many possible responses to the short answer question, unless otherwise indicated in the problem specification.

Support for auto-grading short answer questions was easily obtained. Short answer questions can be graded using the Levenshtein distance, also known as the edit distance. Essentially, the Levenshtein distance gives the number of edits (insertions, substitutions and deletions) required to arrive from the student input to the solution. If the Levenshtein distance is below a threshold, the answer is marked correct; otherwise it is marked incorrect.

3.3.2 Coding

An independent component for auto-grading code does exist, but it has not been integrated into the system. The code auto-grader uses a humble approach; it associates regular expressions with point values:

$$\langle expr \rangle_1 \rightarrow x_1$$

$$\langle expr \rangle_2 \rightarrow x_2$$

$$\vdots$$

$$\langle expr \rangle_i \rightarrow x_i$$

$$\vdots$$

$$\langle expr \rangle_n \rightarrow x_n$$

where $\langle expr \rangle_i$ is some regular expression, and x_i is the point value assigned, which may be negative. This is to dock for the appearance of certain constructs.

The problem is scored based upon the expressions which were parsed, allowing a minimum score no greater than zero:

$$\max \left\{ \sum_{i=1}^n x_i, 0 \right\} \tag{3.8}$$

Chapter 4

Item Dependencies

4.1 Dependency Information

No representation of trait ability mentioned so far specifically accounts for dependency relationships between questions. There are certainly dependency relationships between whole (Bloom \times concept) categories. For example, one must be able to execute a for-loop (Comprehension of Loops) well before gaining the ability to write one which satisfies an intended goal (Synthesis of Loops). This is a course-grained dependency.

Other course-grained dependencies might be less obvious; questions in categories for which concept is high-level and Bloom is low-level, depending on questions in categories for which concept is lower-level and Bloom is high-level. For example, understanding a for-loop (Comprehension of Loops) is predicated on being able to evaluate expressions (Application of Expressions). In a trait ability matrix, these categories might even lie on a diagonal, which would otherwise seem to suggest their independence.

Consider finer-grained dependency relationships among the following specific questions regarding expressions:

- (a) What is $(5 \% 2)$?
- (b) What is $(5 / 2)$?
- (c) What is $(5 \% (5 / 2))$?

The intuition captured by this example is that Part (c) could not be answered correctly, at least not by any logical chain of reasoning, without possessing the specific application ability that Part (a) and Part (b) test for.

It is probably true that Part (c) is more difficult than Part (a) or Part (b), but this is not the reason for the dependency; rather it would appear the dependency is the reason for

Table 4.1: Notation for counts of joint observations

	$y = 0$	$y = 1$	total
$x = 0$	n_{00}	n_{01}	$n_{0\bullet}$
$x = 1$	n_{10}	n_{11}	$n_{1\bullet}$
total	$n_{\bullet 0}$	$n_{\bullet 1}$	n

the higher difficulty. In terms of questions which test application ability for expressions, all of the questions are in the same neighborhood of difficulty; it is even possible albeit unlikely for the β values for all of these to be equal.

However, if dependencies of the form $c \rightarrow a$ and $c \rightarrow b$ exist (that is, c depends on a and b), they may be indicated by a high simple matching coefficient:

$$\frac{n_{00} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}} \quad (4.1)$$

where n_{00} is the number of observations in which both questions are not passed, n_{01} and n_{10} are the number of observations for which either but not both questions are passed, and n_{11} is the number of observations for which both questions are passed. This is assuming that the dependee is scheduled prior to the depender question. The notion expressed by a high simple matching coefficient is that the depender is not passed if the dependee is not passed; and the depender is passed if the dependee is passed.

Alternatively, the phi coefficient, or mean square contingency coefficient, can be used to identify the degree of association between two questions. It is defined as:

$$\phi = \frac{n_{11}n_{00} - n_{01}n_{10}}{\sqrt{n_{\bullet 0}n_{\bullet 1}n_{0\bullet}n_{1\bullet}}} \quad (4.2)$$

This is the binary analogue to the Pearson correlation coefficient; but its range is only $[-1, 1]$ if there is a fifty-fifty split.

4.1.1 The Dependency Graph

Items are organized into sets, called item sets. Alternatively they may be called assessments. These are groups of items used for testing; practically speaking, they may be tests, worksheets, homework assignments, and so forth.

Subsets containing items which have dependency relationships form a directed graph, specifically in form of a tree. In such a tree graph, $x \rightarrow y$ indicates that x depends on y . In addition, as will be discussed later, these edges have weights which indicate the extent of the dependency relationship. The root of each such tree is a depender. Not all items in the set are part of the same tree; there may thus be a forest for an item set.

There are two incarnations of each graph: one is a base dependency graph whose nodes contain the item-specific information. The node i for the i th item contains item response theory parameters α_i , β_i , and γ_i ; as well as a memorability parameter μ_i , which is discussed in Sec ??.

The other is a student graph, which is a clone of the base dependency graph but whose nodes contain the response information from the student. The response information is a list of 2-tuples containing the correctness of the response and the timestamp of that response:

$$\langle (x_1, t_1), (x_2, t_2), \dots (x_n, t_n) \rangle \quad (4.3)$$

An example of a base dependency graph is depicted in Fig 4.1.1; and a student-specific dependency graph is depicted in Fig 4.1.1.

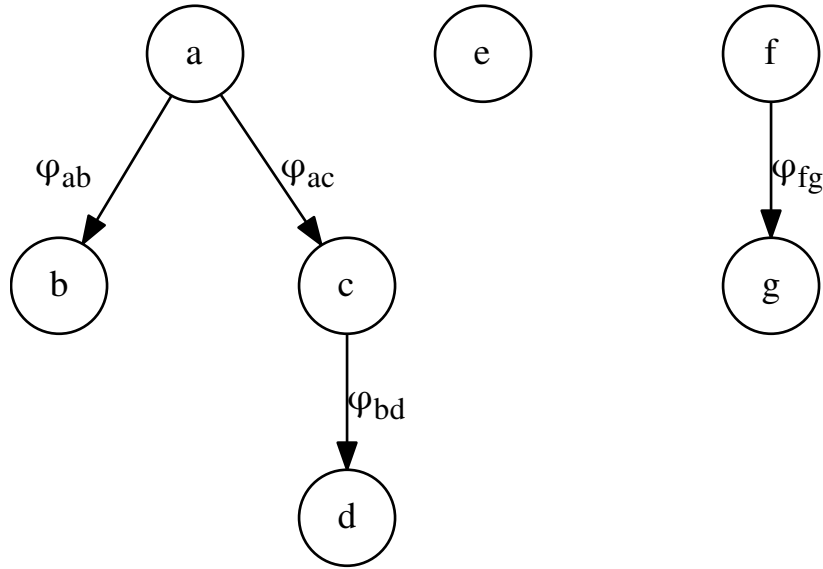


Figure 4.1: A forest obtained from an item set, where each tree is a subset of the item set which has dependency relationships

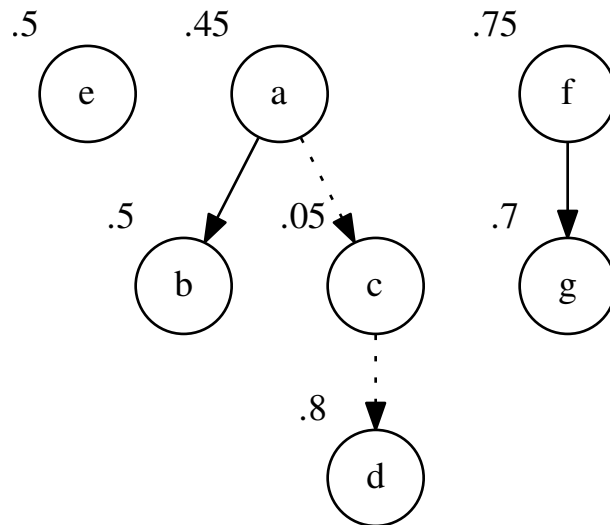


Figure 4.2: The severance of dependency relationships based upon low α values

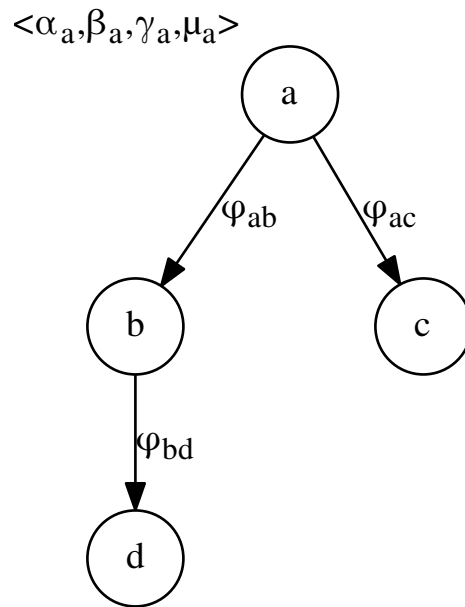


Figure 4.3: The base item set graph, which includes item-specific paramters

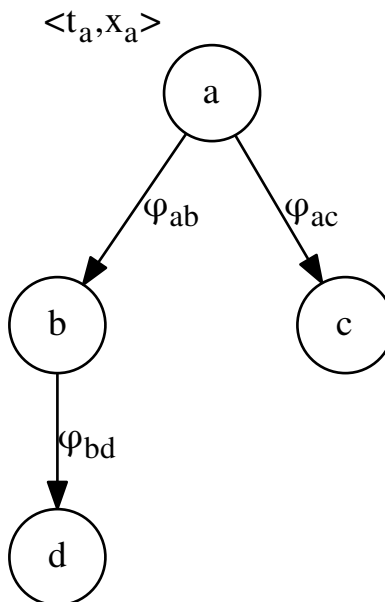


Figure 4.4: The student-specific item set graph, which includes the list of student responses and timestamps for each response

4.1.2 Probability Estimates Using Dependees

Logistic regression (pioneered by David Cox [?]) is an ideal tool for predicting success or failure given prior information about dependencies. This is due to the fact that the dependent variable in question (the success of answering a depender) is binary.

One may wonder why any other form of regression technique is preferred over linear regression. Consider the linear equation in which a binary dependent variable Y depends on some response X :

$$Y = a + bX + \epsilon \tag{4.4}$$

where a is some constant, b is a coefficient, ϵ is the error, and Y assumes one of two possible values, 0 for failure or 1 for success. Linear regression has five assumptions, three of which are violated in this situation:

- Linear relationship. Linear regressions require that linear relationships exist between the dependent variable and independent variables; however since the dependent variable assumes one of two possible values, the relationship is non-linear.
- Multivariate normality. This requirement holds that every linear combination of k components has a univariate normal distribution. This is not the case since each k component assumes one of two possible values.
- Homoscedasticity. This requires that error terms along the regression are equal, but in the above situation, the variance of the error is dependent on the probability. In particular $\text{var}(\epsilon) = p(1 - p)$.

In addition, linear regressions require little to no multicollinearity; that is, independent variables should be independent from one another; this may be established by com-

puting the correlation matrix for the independent variables. Also, there should be no auto-correlation: y_2 , the second observation, should not depend on y_1 . This assumption is typically violated by time series, but it is assumed to hold here because the dependent variables are measured across students rather than time.

Binary logistic regression assumes that:

- The dependent variable is binary,
- $P(Y=1)$ is the probability that the event Y occurs,
- The model is fitted correctly, which means that there are no extraneous variables used in the regression, but that all variables are meaningful,
- That error terms should be independent; each observation should be independent, and little to no multicollinearity should exist.
- Independent variables should be linearly related to the log odds of the event.

It is important to note that the third assumption may or may not hold depending on the correlations between the dependees. The nature of the problem is such that one expects dependee questions to be intercorrelated. One possible remedy is to group the questions to avoid intercorrelations.

Logistic regression is a regression of what are known as log odds. The odds are defined as:

$$\text{odds} = \frac{p}{1 - p} \tag{4.5}$$

And the log odds, or logit, is defined as:

$$\text{logit}(y) = \ln(\text{odds}) = \ln\left(\frac{p}{1 - p}\right) = a + bX \tag{4.6}$$

Correspondingly, odds may be defined as follows:

$$\frac{p}{1-p} = e^{a+bx} \quad (4.7)$$

Solving this equation for p yields

$$p = \frac{e^{a+bx}}{1 + e^{a+bx}} \quad (4.8)$$

which can be reduced to

$$p = \frac{1}{1 + e^{-(a+bx)}} \quad (4.9)$$

This is called the logistic curve, which is one in the family of sigmoid curves. It is identical to the type of curve used in Item Response Theory. Compare the exponent in the Item Response Theory probability formula with the right-hand-side of the logit function:

$$a + bx = \alpha(\beta - 1)\theta \quad (4.10)$$

In the 2PL model, in which γ_i is absent, the Item Response Theory calculation of θ_s can be interpreted as a logistic regression in which the unknown coefficient is θ_s , the item-dependent inputs are given by $\alpha(\beta - 1)$, and the constant $a = 0$.

Thus, a modified form of logistic regression may be sought. It is desirable to keep in account the item discrimination as well as the trait ability of the student.

$$Y = b_1X_1 + b_2X_2 + \dots + b_{n-1}X_{n-1} + b_n\alpha_i(\theta_s - \beta_i) \quad (4.11)$$

In this expression, the term $\alpha_i(\theta_s - \beta_i)$ is retained, however it is multiplied by a coefficient b_n to determine its weight. Likewise coefficients $b_1, b_2, \dots b_{n-1}$ determine weights of the dependee items. This results in a probability function for answering the depender question correctly given responses for the dependee questions:

$$p = \frac{1}{1 + e^{b_1X_1 + b_2X_2 + \dots + b_{n-1}X_{n-1} + b_n\alpha_i(\theta_s - \beta_i)}} \quad (4.12)$$

To determine the individual contribution of each predictor (dependee), the Wald statistic may be used. It is defined as:

$$W_i = \frac{b_i^2}{SE_{b_j}^2} \quad (4.13)$$

If the statistic is sufficiently low, then the supposed dependee is likely not a dependee at all, in which case the edge indicating the dependency relationship may be pruned from the tree. The logistic regression model will naturally assign a low weight to such a dependee, and its use in the scheduling algorithm will be limited by its relative weight.

The use of the logistic regression model in the intelligent tutoring system suggests that the student would need to have answered all the dependencies (that values of 0 or 1 are available). In many instances where it is desirable to use such a predictive model, this is not the case; instead what is available is a probability that the student will be able to answer the question correctly. In such cases, the model is used with the probabilities calculated using Item Response Theory.

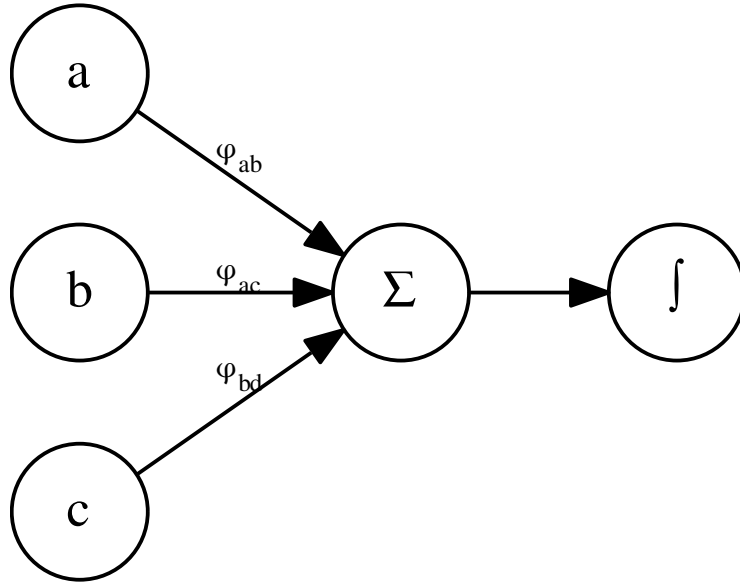


Figure 4.5: A perceptron; a single-layer neural network, where the inputs a , b , and c are multiplied by weights, summed, and applied to a sigmoid squash function

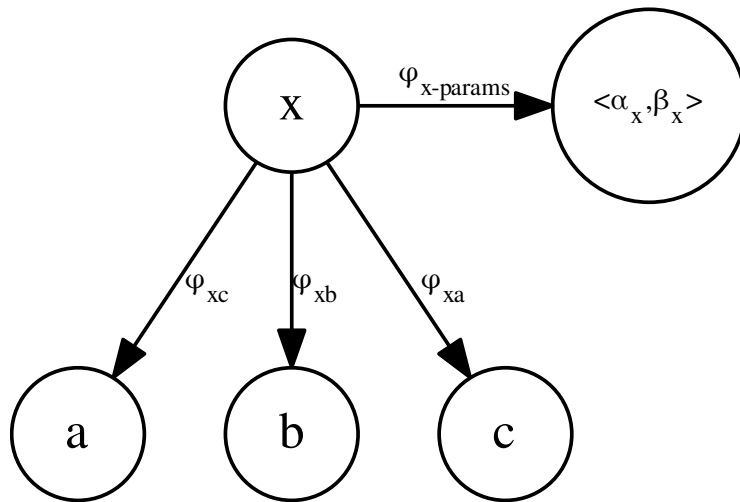


Figure 4.6: A view of the dependency graph and weights used in the logistic regression, including the item parameters

The above probability estimates also assume an atemporal view of questions: if the student has answered a question, the student remembers the answer to the question permanently; however this is evidently not the case.

What remains is a temporal account of questions, in particular the role of memory and forgetting.

Chapter 5

Memory

5.1 Models of Memory

Ebbinghaus is credited with a theory of memory and forgetting which has withstood empirical study for over a century [?]. It is known as the power law of forgetting. According to the power law of forgetting, the strength of a memory after a time t falls off exponentially:

$$S(t) = ae^{-bt} \tag{5.1}$$

In this model, a is the initial strength of the memory, and b^{-1} is a decay rate. The curve drawn by this function is known as the curve of forgetting, which is depicted in Figure 5.1. If $a = 1$, the function may be interpreted as a probability function:

$$p_{recall}(t) = e^{-bt} \tag{5.2}$$

According to the theory, if the memory strength falls below a certain threshold, then in the absence of any intervening information (that is, information which re-activates the memory via association), the individual will be unable to spontaneously recall the information. In a probabilistic model, one may set the threshold at .5 probability—the probability below which the individual has more than likely forgotten the information.

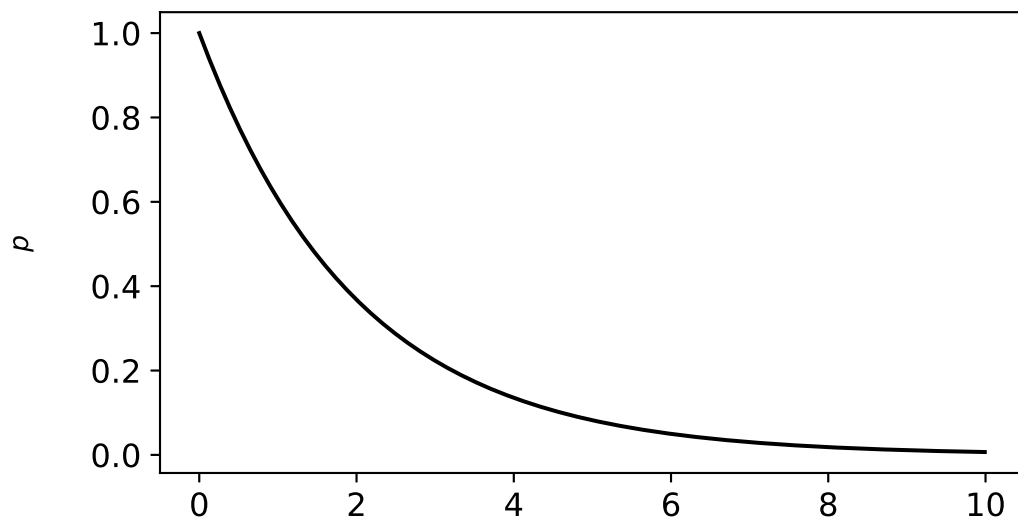


Figure 5.1: The Ebbinghaus curve of forgetting, which features an exponential dropoff of memory strength over time

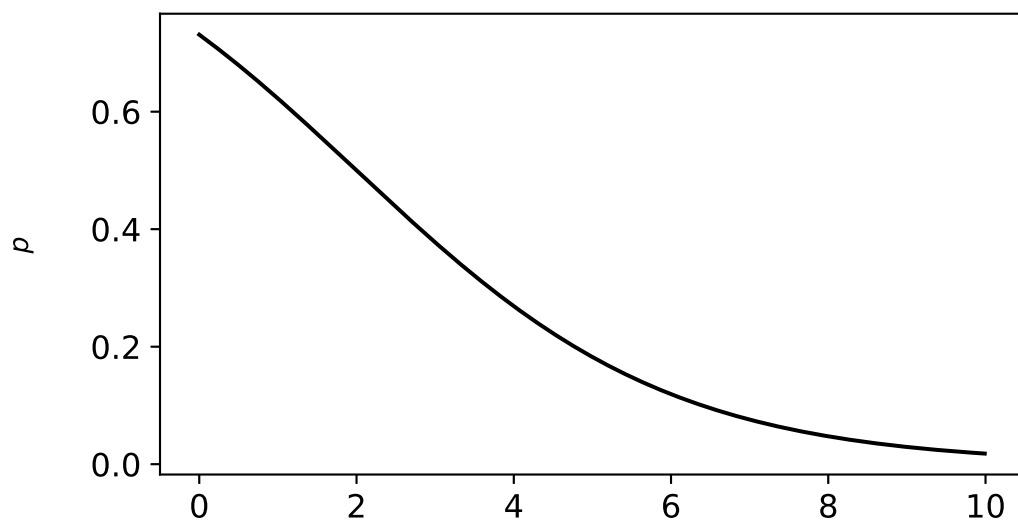


Figure 5.2: The modified curve of forgetting, which resembles the reverse sigmoid function

5.2 ACT-R

As has been mentioned in Sec ??, John Anderson is credited with having developed Adaptive Control of Thought-Rational (ACT-R), a process-based model which simulates the solving of problems. In ACT-R, there are goals, akin to problem statements; and rules, or processes used to solve problems; and finally facts, or knowledge utilized in the course of applying rules.

In addition to this, however, Anderson added models for memory and forgetting to support realistic recall probabilities and latencies. The memory component is based on Ebbinghaus' model of memory retrieval. Anderson added a component to explain memory re-activation of a memory. According to Anderson's model, a chunk of memory i is re-activated (or additionally activated) to the extent that other chunks of information (related concepts, words, ideas, etc.) which have some association to i are attended to. This notion is captured in the following equation:

$$a_i = b_i + \sum_{j=1}^n w_j s_{ji} \quad (5.3)$$

In this equation, known as the activation equation, the activation of a chunk i is equal to its base activation b_i , plus the products of the attentional weights w_j by the associative strength of s_{ji} to other chunks. This provides an intuitive explanation for the manner in which recall of a target chunk can be stimulated by dropping hints, using certain key words or phrases, or mentioning related material.

Practice has the effect of causing the base strength of the memory to increase, and delays cause the strength of the memory to drop off:

$$b_i = \ln \left(\sum_{j=1}^n t_j^{-d} \right) \quad (5.4)$$

Here, t_j is the time since the j th practice of an item, and d is a decay rate. . . .

Some concepts, particularly the notion of re-activation of memories, have been borrowed from ACT-R and modified to fit the more coarse-grained intelligent tutoring system presented in this work. In particular, total problems rather than individual processes will have probabilities of recall associated with them. Also, associative strengths are established using a factor analysis.

5.3 Alterations to Memory Model

A slight modification to this theory accounts for short-term memory and short-term memorization, which allows for a small time window for the student to enjoy a high probability of recollection before dropping off sharply, as in the original curve:

$$p_{recall}(t) = \frac{1}{1 + e^{m(t-\lambda)}} \quad (5.5)$$

In this equation, λ is the lifespan of the memory; or, the amount of time that passes until there remains only a .5 probability that the student recalls the information. The value m is a parameter which controls the rate of dropoff, much like the decay rate in Ebbinghaus' model. An example curve for this equation is given in Figure 5.1.

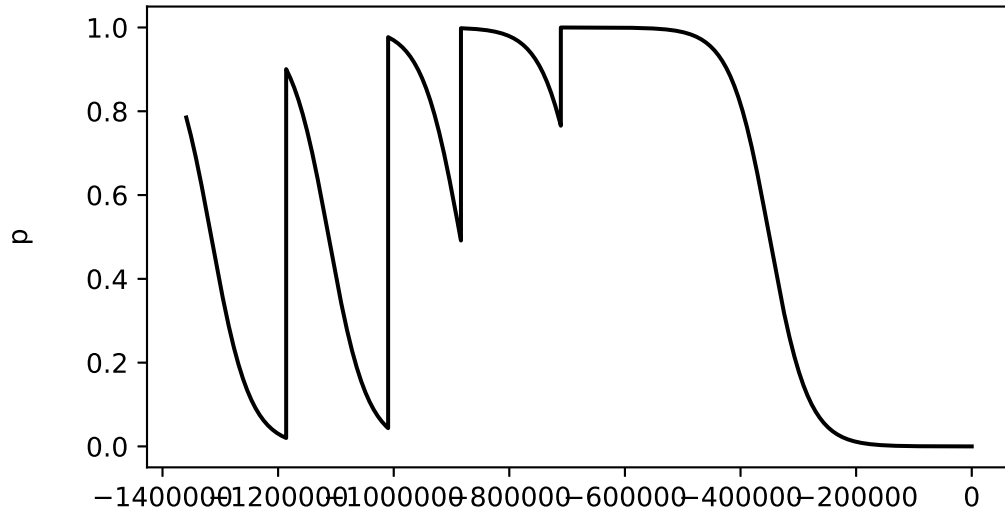


Figure 5.3: Forgetting with re-activation; each spike in the graph is an additional trial where the student is exposed to the item again

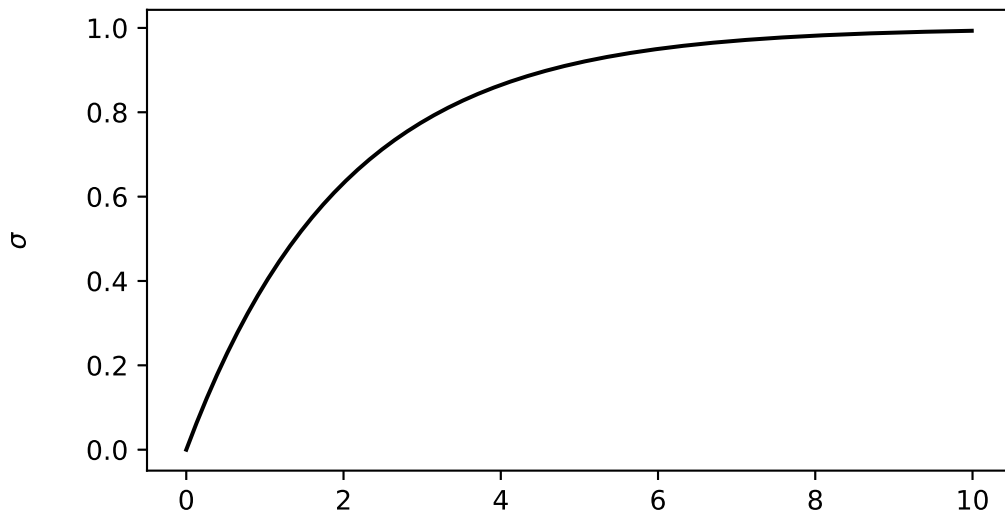


Figure 5.4: The learning curve, which indicates the extent of memory lifespan increase given a time between trials

5.4 Re-Activation

??

To account for re-activation, a simple model for the extension of half-life may be used:

$$\lambda_n = \rho_s \lambda_{n-1} \quad (5.6)$$

Here, n refers to exposure or trial number n . In the intelligent tutoring system, this is the n th time that the student has seen the problem. λ_{n-1} is the former lifespan of the memory. ρ_s is a learning rate, which is a parameter particular to the student; its domain is $(1, \infty]$. The intuition captured by this formula is that with an increased number of trials, the lifespan of the memory increases.

Apparently there is a difference in problems in the ease with which they are learned. An addendum to this can be used to account for individual differences in problems:

$$\lambda_n = \mu_i \rho_s \lambda_{n-1} \quad (5.7)$$

Here, μ_i represents the memorability of the problem, or the ease with which the problem solution can be committed to memory.

5.4.1 Spacing Effect

The spacing effect is the effect that the amount of time in between trials has on the memorization of a chunk of memory. In the above model, memorization is interpreted as

an increase in the lifespan of a memory. If only a short amount of time passes between the last trial, the effect will not be as great as if a longer time has passed. One consequence of this is that, according to the spacing effect hypothesis, cramming is ineffective (where cramming is namely repeating trials in short bursts).

The spacing effect can be accommodated in the memory model used by the intelligent tutoring system. We define a function for the dropoff:

$$\sigma_t = (1 - e^{-at}) \quad (5.8)$$

This indicates the extent to which the spacing from the time the item was last seen influences the increase in the lifespan of the memory:

$$\lambda_n = (1 + \sigma_t \mu_i \rho_s) \lambda_{n-1} \quad (5.9)$$

The utility of this model is in assessing the probability with which a student answers a question; not only based on trait ability and dependency relationships, but also on the inherent tendency to forget information with the passage of time.

It will be assumed that, if a student has been exposed to a item before, then the probability of being able to answer the item correctly may assume one of two values. The first is based upon recollection; it is the probability of recalling the facts, processes, and so forth required to produce the solution for an item. The second is based upon derivation of the solution from known facts, processes, and so forth in the dependencies, as if the student were answering the question for the first time. That is, if a student does not recall the process for solving a problem, the probability defaults to the probability based upon item parameters, trait ability and dependency relationships:

$$p = \begin{cases} p_{recall}(t) & \text{if } p_{recall} > p(\theta_s, x_1, \dots) \\ p(\theta_s, x_1, \dots) & \text{otherwise} \end{cases} \quad (5.10)$$

Now the intelligent tutoring system has models of dependency relationships among questions and probability estimates for questions, and all other mathematical equipment required to schedule problems.

Chapter 6

Scheduling

6.1 Item Scheduling

The general procedure for scheduling content and assessments can be stated as follows.

6.1.1 Preliminaries

First, at the very beginning of the program, the trait ability matrix for the student is initialized:

$$\Theta \leftarrow -3 \tag{6.1}$$

That is, the algorithm initially proceeds on the assumption that it is unlikely for any student to answer any question. The initial estimate for the α values may be 1, such that the Item Response Theory 3PL model defaults to 2PL in the absence of any data. The initial difficulty estimates may be determined by the instructor, or they may be initialized set to 0. The μ values may be initialized to 1.

The first iteration schedules a preliminary test consisting of a diagonal block of the trait ability matrix, for example:

$$\Theta_s = \begin{bmatrix} \theta_{s11} & \theta_{s21} & \theta_{s31} & \dots \\ \theta_{s12} & \theta_{s22} & & \\ \theta_{s13} & & \ddots & \\ \vdots & & & \end{bmatrix} \tag{6.2}$$

The test should be small enough to be feasible while having enough questions to support an MLE. At least two questions are needed per (Bloom \times concept) for an MLE. Asking the questions from a single category is sufficient to jumpstart the process, but the larger the triangle, the richer the initial information (α , β values). Once the data is collected, it is then possible to re-calculate α_i and β_i for all items.

With respect to α_i , it is desirable to discard any question i for which $\alpha_i \leq 0$. Recall that negative values of α indicate a negative biserial correlation with the composite score, which means that the question asked is an indicator of whether or not the student will perform poorly on the overall measure.

If α_i is close to -1, then the question may have predictive power, in which case it should be analyzed for its properties. It may be desirable to retain such questions for the purpose of predicting success. Regardless, if $\alpha_i < 0$, then it along with its subtree should be severed from the dependency graph.

With the response set, Θ_s can be constructed and the proximal zone of development can be identified.

6.1.2 Subsequent Assessments

When a student is assigned an assessment, first the probabilities that the student will be able to answer each question are recursively calculated. They must be calculated from the bottom-up, since internal nodes require probability estimates from their children. The leaves are the only nodes that do not require logistic regression to obtain a probability estimate, since the leaves have no dependencies. Probability estimates for the children are obtained using Item Response Theory.

For each node which has been answered, the probability of recall can be calculated

using Eq 5.5. In the presence of a probability of recall, the final probability of answering the question can be calculated using Eq 5.10.

Then, a recursive traversal begins at the root of each tree. If the probability of answering the question at the root is within the neighborhood of $[\delta, .5+\delta]$, it is asked. Recall that low probabilities correspond to higher differences in $\beta - \theta$; thus if a probability is very low, yet the student answers the question correctly, it will indicate a corresponding rise in trait ability. Such items have higher impact on the MLE estimate of trait ability. Yet if the probability is below .5, there is no reasonable assurance the student will be able to answer it to begin with.

If the probability to answer the question correctly is in the neighborhood of $[\delta, .5+\delta]$, the question is skipped. The algorithm may recursively descend into its dependencies to find lower- p items.

If the probability is less than .5, then the system will seek to raise the probability by targetting the dependees, in order of the highest-impact dependees to the lowest-impact.

6.1.3 Finding High-Impact Items

The dependee which controls the highest amount of variance in the parent probability is that dependee which has the highest coefficient the logistic regression model. However the student may already have a high probability of answering that dependee, such that there is little room for increases in the probability in the parent node due to answering the child node question correctly.

Supposing p_i is the probability of answering the i th dependee correctly, then the amount which can be gained in the dependee is

$$\Delta_i = 1 - p_i \tag{6.3}$$

If Δ_i assumes a value close to zero, it implies two things: first, the student's trait ability for the dependee is high enough not to bother testing it, and second, raising it to 1 would likely have little effect on the probability estimate of the parent.

Thus the highest impact item is determined not only by the coefficient, but also by the gain in probability for the dependee. Supposing that the odds estimate for the parent j were given by

$$\text{logit}_j = e^{b_1p_1+b_2p_2+\dots+b_ip_i+\dots+b_np_n} \quad (6.4)$$

Then the odds estimate for the parent j if the probability were to rise to 1 would be

$$\text{logit}_j = e^{b_1p_1+b_2p_2+\dots+b_ip_i+b_i\Delta_i+\dots+b_np_n} \quad (6.5)$$

then the odds ratio would be equal to

$$\frac{e^{b_1p_1+b_2p_2+\dots+b_ip_i+\dots+b_np_n}}{e^{b_1p_1+b_2p_2+\dots+b_ip_i+b_i\Delta_i+\dots+b_np_n}} \quad (6.6)$$

which when reduced is simply

$$e^{b_i\Delta_i} \quad (6.7)$$

Thus that dependency which has the highest increase in the odds ratio is the item i such that

$$\operatorname{argmax}_i \left[\left(e^{b_i} \right)^{\Delta_i} \right] \quad (6.8)$$

If that dependency has a probability within the window $[.5, .5+\delta]$, it is asked; otherwise if it has probability $[0, .5)$, the algorithm is recursively applied to the dependency.

6.1.4 Finding the Whole Schedule

The above algorithm may find a complete schedule by iteratively finding the next item in the schedule, then proceeding on the assumption that the item is answered correctly and is re-activated. In order to preserve the data in the course of iteratively finding the next item, the student's trait ability matrix and the student dependency graph may first be copied. Any changes to trait ability estimates or to the student's responses set are reflected in the copies.

The algorithm comes to a stopping point when there exists no question for which the probability of answering falls below $.5+\delta$.

Chapter 7

Experimental Results

7.1 Experiments Supporting the Utility of the Taxonomy

7.1.1 Experiment 1 Design

Our first experiment tested to see if there is a performance difference between computer-based assessment and paper-based assessment when questions are ordered by Bloom level. Our hypothesis was that students taking the computer-based assessment would fare better than those taking the paper-based assessment because of the immediate feedback offered by the computer-based assessment.

We designed a test of 10 questions (2 concepts, each concept having questions over 5 Bloom levels) to give to students¹. The questions were of multiple-choice and short-answer format. There were two knowledge, comprehension, application, analysis, and evaluation questions. No synthesis questions were asked because of the constrained formats allowed by the computer-based testing framework. An interrater reliability of 90% was determined by two independent raters (the authors), both computer science educators, who assessed the Bloom levels of the questions. After a point of disagreement about an evaluation-level problem, the test was adjusted to yield an 100% interrater reliability.

We designed a test of 10 questions (with 2 concepts, each concept having questions over 5 Bloom levels) to give to students. The concepts tested were on recursion and binary trees, and were written to be language-independent. The questions were of multiple-choice

¹All questions may be viewed at: <https://steam.cct.lsu.edu/assessment/>

and short-answer format. There were two knowledge, comprehension, application, analysis, and evaluation questions. No synthesis questions were asked because of the constrained formats allowed by the computer-based testing framework. An interrater reliability of 90% was determined by two independent raters, both computer science educators, who assessed the Bloom levels of the questions. After a point of disagreement about an evaluation-level problem, the test was adjusted to yield an 100% interrater reliability.

Evaluation-level problems are resistant to multiple-choice and short-answer formats because of the nature of the category. Our approach was to use multiple-choice questions of the “choose the best answer” format, in which there are many proposed uses of a concept or language construct, but one stands out as the most sensible from the standpoint of experts.

The test resembled a quiz that might be given in the normal course of teaching the class. At the end of the quiz, the question “how satisfied were you with the (paper/computer)-based medium?” was asked to gauge satisfaction differences as well.

For this experiment, volunteers were recruited from a Java programming class for introductory computer science students. All students volunteered; candy was offered as an incentive for all the experiments. We split the classroom into two groups, matched based on their current grade in the course. We gave the control group the paper quiz, and the experimental group the computer-based quiz.

An answer was scored as totally correct if it coincided exactly with the solution, and otherwise scored as incorrect. Correct answers were encoded with 1, incorrect with 0. A composite score was derived by summing these scores per-student.

7.1.2 Experiment 2 Design

The second experiment tested the effect of ordering the questions by Bloom level. For this experiment we designed another test of 10 questions (2 concepts, each concept having questions over 5 Bloom levels). This test also covered recursion and binary trees. In the

control condition, questions were given in forward Bloom-level order. In the experimental condition, they were given in reverse order.

For both Experiments 2 and 3, participants were recruited in the same manner; however for this experiment a C++ class which followed the same conceptual track was also added to the pool. Matched-pairs were assigned to each group based on their current grade in each course. The test was constructed and interrater reliability gauged in the same manner, and the test was also scored in the same manner.

7.1.3 Experiment 3 Design

The third experiment tested the effect of intervening questions on the performance of later questions in the assessment. Our hypothesis was that overall performance would be improved if incorrect answers triggered the addition of new *intervention* questions from a lower Bloom level.

Experiment 3 participants were recruited from a different class. The test was this time language-dependent (MATLAB) and tested mastery of control structures, in particular for-loops. In the control condition, the control group was given an assessment of 10 questions, with 2 questions per the first five Bloom levels. The experimental group was given an adaptive measure. If at any point a student answered a question incorrectly, then a question at the next lowest level was given. This applied to all levels except knowledge. So for example if a student answered an application-level question incorrectly, a comprehension-level question (related to the application-level question) was scheduled before another application-level question of the same type. The experimental group thus had a maximum of 4 additional questions asked for a total possible 14-question test.

7.2 Analysis and Results

7.2.1 Experiment 1 Analysis

The experimental condition ($N=27$ $M=6.21$) did in fact show a higher mean score than the control condition ($N=27$ $M=5.23$) in overall performance. Statistical significance was tested with a one-tailed two-sample matched-pairs Student's t-test on the composite score. The result indicated a statistically significant difference ($t=2.024$, $p=0.048$). The experimental condition ($M=4.93$) showed a higher mean score than the control condition ($M=4.38$) in satisfaction as well; a similar t-test was done and was marginally statistically significant ($t=1.7753$, $p=0.082$).

7.2.2 Experiment 2 Analysis

The experimental condition ($N=48$, $M=4.94$) showed a higher mean score than the control condition ($N=48$, $M=4.31$). Statistical significance was tested with a one-tailed parametric Student's t-test on the composite score. The result indicated a statistically significant difference ($t=2.13$, $p=0.036$).

7.2.3 Experiment 3 Analysis

To tell the immediate effect of the intervention questions, one-tailed parametric Student's t-test on the composite score of questions starting after the first intervention question was done. It was hypothesized that the experimental condition would perform better on the remainder of the test. The experimental group ($N=45$, $M=6.98$) outperformed the

control group ($N=45$, $M=6.23$). The result indicated a marginally statistically significant difference ($t=1.7082$, $p=0.092$).

7.2.4 Limitations of These Experiments

A few validity concerns are to be pointed out. No random order was given in Experiment 2 because of a lack of available subjects; hence it cannot be inferred that forward order is no different from random order. The experimenters (paper authors) designed the tests. The number of items per test was small to allow for a conservative testing time. For Experiment 1, confounding variables (those other than immediate feedback) may have played a role in test-taking because the test-taking media were different.

7.3 Experiments Supporting the Reconciliation of the Taxonomy with IRT

7.3.1 Experiment 1 Design

In the first experiment, a selection of 16 multiple-choice questions on topics in operating systems concepts were asked to a sample group of students ($N=54$) in an undergraduate-level operating systems class. Each question had 5 choices ($\gamma=.2$). These questions were tagged with Bloom levels, and interrater reliability was calculated between two independent raters with experience in curricular and course design. Questions were tagged with hypothesized difficulty levels, which were targeted to be inversely proportional to the Bloom levels. Hence the question author intended for the knowledge questions to be difficult relative to analysis questions, while still maintaining the appropriate Bloom category.

7.3.2 Experiment 2 Design

To test this modified theory, another experiment was conducted. The purpose of this experiment was to explore the relationship among the probability predicted by unmodified IRT, the probability predicted by modified IRT, and the correctness of the actual response. It was hypothesized that modified IRT should produce a more accurate output than unmodified due to its account of dependency relationships.

As in the first experiment, a selection of 16 multiple-choice questions on topics in operating systems concepts were asked to the same sample group of students ($N=54$) in the same manner ($\gamma=.2$). The questions covered 4 concepts, and each concept had 4 questions, each of different Bloom levels (Knowledge, Comprehension, Application, Analysis), where each question of a Bloom level higher than Knowledge was dependent on a question of an immediately-lower Bloom level (e.g. the Application question was dependent on the Comprehension question of the same concept). These questions were asked in forward Bloom order. The hypothesized difficulties were held equal across Bloom levels.

7.3.3 Experiment 3 Design

One question remains: can p_i^{mod} be used in some meaningful way? Suppose we wish to maximize the probability that a student answers an Analysis question correctly, and that this Analysis question depends on an Application question. For that matter, it may depend on more questions, each of which having its own degree of relatedness to the depender.

We thus posit a more general form of modified IRT which takes into account multiple dependencies using factor analysis [?]. From a factor analysis, it is possible to obtain the proportion of variance explained by each dependency by squaring the factor loading. A factor analysis was performed on the dependent question using dependees as factors to

Those proportions of variance could then be used as part of a more generalized modification to IRT:

$$p_i(\theta) = (1 - (\sum_j v_j)) \left(\gamma_i + \frac{1 - \gamma_i}{1 + e^{\alpha_i(\theta - \beta_i)}} \right) + \sum_j (v_j) \left(\text{sgn}(r) x_{sj} + \frac{1 - \text{sgn}(r)}{2} \right)$$

In our experiment to test the utility of this formula, we first required a small problem set with one depender and multiple dependees. The experiment was intended to test whether or not the asking of highly-dependent questions increased mean scores.

The others would receive a combination of questions. Each student was randomly assigned to one of eight groups. Four students received each combination of three questions,

Table 7.1: Proportions of variance due to questions for Experiment 1

	q1	q2	q3	q4
v_j	.02	.23	.30	
M	.67	.66	.43	.34

Table 7.2: Item means per Bloom category for Experiment 1 (M_1) and Experiment 2 (M_2)

	Knowledge	Comprehension	Appplication	Analysis
M_1	.42	.53	.67	.82
M_2	.78	.62	.48	.41

such that 4 received the depender outright, 12 received one of the three dependees (4 each) before the depender, 12 received two of the three dependees (6 each) prior, and 4 received all three dependees. The purpose of this was to test the effect of exposure of each dependee. Thus each dependee question appeared in half of the students' assessments. The total number of respondents for the depender was 32, and the total number of respondents for each dependee was 16.

7.4 Analysis and Results

7.4.1 Experiment 1 Analysis

Interrater reliability on Bloom levels was calculated to be 93%. Means for each Bloom level are reported below.

One-tailed Student t-tests were performed on the three consecutive pairs of means. The mean for comprehension was statistically significantly greater than the mean for knowledge

($t = 2.00$), the same held true for application and comprehension ($t = 2.33$), and analysis and application ($t = 2.64$). These statistics, in conjunction with the interrater reliability for the assignment of Bloom levels, support the view that Bloom level and difficulty are distinguishable.

In light of this data, a modification of IRT was sought in order to account for the dependency relationships that existed among the questions. Intuitively, this modification of p_i should possess the properties that: (a) if a question x_j is unrelated, p_i should be unaffected; and (b) if x_j is fully related, p_i should be equal to the response value for x_j .

The degree of relationship may be expressed with Pearson's r , where $r_{ij} = 0$ indicates no relationship between x_i and x_j and $r = 1$ indicates a full positive correlation. An even more viable metric is the coefficient of determination r^2 , which indicates the proportion of variance in the depender which is attributable to the dependee. Intuitively, it is this proportion of the probability which should be associated with the correctness of j . The remainder may be left to item response theory. In the event of a negative correlation, the response should be opposite.

$$p_i(\theta) = (1 - r^2) \left(\gamma_i + \frac{1 - \gamma_i}{1 + e^{\alpha_i(\theta - \beta_i)}} \right) + (r^2) \left(\mathbf{sgn}(r)x_{sj} + \frac{1 - \mathbf{sgn}(r)}{2} \right)$$

The above formula satisfies our needs. The **sgn** function gives the sign of r . For $r = 1$, p_i defaults to the correctness of the dependee's answer (0 or 1, depending on whether or not the dependee was answered correctly); and for $r = -1$, p_i assumes the flip of the dependee's answer. For $r = 0$, p_i assumes the form of original IRT.

In the MLE estimation of θ , we would omit question j , since it is included already in the above formula.

7.4.2 Experiment 2 Analysis

Interrater reliability on Bloom levels was once again calculated to be 93%. To test the hypothesis that modified IRT leads to an increase in accuracy, a t-test on the increase in accuracy of probability was conducted at the .05 significance level.

First the probabilities $p_i^{irt}(\theta)$ (for original IRT) and $p_i^{mod}(\theta)$ (for modified IRT) were calculated. Note that each of these formulas require θ . The value for θ was calculated per-student using the MLE method on the first data set. To determine whether or not $p_i^{mod}(\theta)$ was on the whole more accurate than p_i^{irt} , the result

$$\Delta p_{si} = (2x_{si} - 1)(p_i^{mod} - p_i^{irt})$$

was calculated. Here p_{si} represents the probability that student s will answer item i correctly. Then Δp_{si} represents the increase in accuracy of the probability; for example, it is positive if $p_i^{mod} > p_i^{irt}$ and $x_{si} = 1$.

Finally, to test the effectiveness of p_i^{mod} , a one-tailed Student's t-test was calculated on Δp_{si} for the depensee questions (N=648). The test revealed that Δp_{si} is statistically significantly greater than zero ($t = 2.05$), implying that p_i^{mod} indeed produces a more accurate estimate than IRT alone.

As the data reveals, the proportion of students who passed a question did not agree with the hypothesized difficulties. It was suspected that this may be due to the effect of item dependencies, which the next experiment investigates in further detail.

7.4.3 Experiment 3 Analysis

The analysis of this data required examining the relationship between dependent response and depensee exposure.

Table 7.3: The percent of students who answered the depender correctly given a particular set of questions (left), as well as the percent of students who answered the depender correctly given exposure to a given question (right).

dependees	#correct	percent	#correct	percent
none	1	.25	1	.06
q1	1	.25	8	.50
q2	2	.50	10	.62
q3	3	.75	11	.69
q1, q2	2	.50		
q1, q3	2	.50		
q2, q3	3	.75		
q1, q2, q3	3	.75		

Let E_{sj} be 1 if student s was exposed to j , 0 otherwise. Then, our hypothesis was that exposure of dependee questions should result in an increase in the probability that the student answers the depender correctly, in proportion to the amount of variance explained by the dependee. Below in the table is the number of times the depender was answered correctly given exposure to a set of dependees. Though the data set is small, there does appear to be a trend in the data. A clearer trend can be observed by summing the number of correct responses per exposure to a given question.

It is interesting to note that to an extent, the proportions of variance explained by a given question appear to be co-related to the percentage of students who answered the depender correctly. However, the numbers are too small to draw a definitive conclusion.

Due to the low N in Experiment 3, the results do not present *conclusive* evidence that the intervention of the dependee questions was responsible. However, there is enough preliminary evidence to warrant further investigation. If there is an effect, it is suspected that exposure to the dependee problem solution by the ITS plays a larger role than simply the question itself.

References

- [1] Donald R Bacon. Assessing learning outcomes: A comparison of multiple-choice and short-answer questions in a marketing context. *Journal of Marketing Education*, 25(1):31–36, 2003.
- [2] Frank B Baker and Seock-Ho Kim. *Item response theory: Parameter estimation techniques*. CRC Press, 2004.
- [3] Isaac I Bejar. Educational diagnostic assessment. *Journal of educational measurement*, pages 175–189, 1984.
- [4] Benjamin Samuel et al. Bloom. *Taxonomy of educational objectives*. David McKay, 1956.
- [5] Ricardo Britto and Muhammad Usman. Blooms taxonomy in software engineering education: A systematic mapping study. *Frontiers in Education Conference*, 2015.
- [6] Jim Buckley and Chris Exton. Blooms taxonomy: A framework for assessing programmers knowledge of software systems. *International Workshop on Program Comprehension*, 2003.
- [7] Dennis Castleberry. The prickly pear archive. *ICCS*, 2011.
- [8] Dennis Castleberry and Steven R Brandt. The effect of question ordering using bloom’s taxonomy in an e-learning environment. In *International Conference on Computer Science Education Innovation & Technology (CSEIT). Proceedings*, page 22. Global Science and Technology Forum, 2016.
- [9] Debatri Chatterjee, rajat Das, Anirhudda Sinha, and Shreyasi Datta. Analyzing elementary cognitive tasks with bloom’s taxonomy using low cost commercial eeg device. *International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2015.

- [10] Gráinne Conole and Bill Warburton. A review of computer-assisted assessment. *ALT-J*, 13(1):17–31, 2005.
- [11] E de Bruyn, E Mostert, and A van Schoor. Computer-based testing—the ideal tool to assess on the different levels of blooms taxonomy. *Interactive Collaborative Learning*, 2011.
- [12] Ursula Fuller, Colin G Johnson, Tuukka Ahoniemi, Diana Cukierman, Isidoro Hernán-Losada, Jana Jackova, Essi Lahtinen, Tracy L Lewis, Donna McGee Thompson, and Charles et al. Riedesel. Developing a computer science-specific learning taxonomy. In *ACM SIGCSE Bulletin*, volume 39:4, pages 152–170. ACM, 2007.
- [13] Hamzah A Ghulman and Mohd Saidfudin Mas’odi. Modern measurement paradigm in engineering education: Easier to read and better analysis using rasch-based approach. In *Engineering Education (ICEED), 2009 International Conference on*, pages 1–6. IEEE, 2009.
- [14] Sanjay Goel and Nalin Sharda. What do engineers want? examining engineering education through bloom’s taxonomy. *Australasian Association for Engineering Education*, 2004.
- [15] Isidoro Hernán-Losada. Testing-based automatic grading: A proposal from blooms taxonomy. *International Conference on Advanced Learning Technologies*, 2008.
- [16] PW Hill and B McGaw. Testing the simplex assumption underlying bloom’s taxonomy. *American Educational Research Journal*, 18(1):93–101, 1981.
- [17] Colin G Johnson and Ursula Fuller. Is bloom’s taxonomy appropriate for computer science? In *Proceedings of the 6th Baltic Sea conference on Computing education research: Koli Calling 2006*, pages 120–123. ACM, 2006.

- [18] William L Kuechler and Mark G Simkin. Why is performance on multiple-choice tests and constructed-response tests not more closely related? theory and an empirical test. *Decision Sciences Journal of Innovative Education*, 8(1):55–73, 2010.
- [19] Thomas Lord and Sandhya Baviskar. Moving students from information recitation to information understanding: exploiting bloom’s taxonomy in creating science questions. *Journal of College Science Teaching*, 36(5):40, 2007.
- [20] Dianna L Newman, Deborah K Kundert, David S Lane Jr, and Kay Sather Bull. Effect of varying item order on multiple-choice test scores: Importance of statistical and cognitive difficulty. *Applied Measurement in education*, 1(1):89–97, 1988.
- [21] Mahmood Niazi. Teaching global software engineering: experiences and lessons learned. *IET Software*, 2014.
- [22] Dave Oliver, Tony Dobeles, Myles Greber, and Tim Roberts. This course has a bloom rating of 3.9. In *Proceedings of the Sixth Australasian Conference on Computing Education-Volume 30*, pages 227–231. Australian Computer Society, Inc., 2004.
- [23] Richard H Osborne, Roy W Batterham, Gerald R Elsworth, Melanie Hawkins, and Rachelle Buchbinder. The grounded psychometric development and initial validation of the health literacy questionnaire (hlq). *BMC public health*, 13(1):658, 2013.
- [24] Shuhaida Shuhidan, Margaret Hamilton, and Daryl D’Souza. Understanding novice programmer difficulties via guided learning. *ITiCSE*, 2011.
- [25] Onjira Sitthisak, Tasanawan Soonklang, and Lester Gilbert. Cognitive assessment applying with item response theory. *19th Annual International Conference on Computers in Education*, 2011.
- [26] Errol Thompson, Andrew Luxton-Reilly, Jacqueline L Whalley, Minjie Hu, and Phil Robbins. Bloom’s taxonomy for cs assessment. In *Proceedings of the tenth conference*

on Australasian computing education-Volume 78, pages 155–161. Australian Computer Society, Inc., 2008.

- [27] Tzu-Hua Wang, Kuo-Hua Wang, Wei-Lung Wang, Shih-Chieh Huang, and Sherry Y Chen. Web-based assessment and test analyses (wata) system: development and evaluation. *Journal of Computer Assisted Learning*, 20(1):59–71, 2004.

Appendix A

Appendix A: IRB Documents

Application for Exemption from Institutional Oversight

Unless qualified as meeting the specific criteria for exemption from Institutional Review Board (IRB) oversight, ALL LSU research/ projects using living humans as subjects, or samples, or data obtained from humans, directly or indirectly, with or without their consent, must be approved or exempted in advance by the LSU IRB. This form helps the PI determine if a project may be exempted, and is used to request an exemption.



Institutional Review Board
Dr. Dennis Landin, Chair
130 David Boyd Hall
Baton Rouge, LA 70803
P: 225.578.8692
F: 225.578.5983
irb@lsu.edu | lsu.edu/irb

- Applicant, Please fill out the application in its entirety and include the completed application as well as parts B-F, listed below, when submitting to the IRB. Once the application is completed, please submit the completed application to the IRB Office by e-mail (irb@lsu.edu) for review. If you would like to have your application reviewed by a member of the Human Subjects Screening Committee before submitting it to the IRB office, you can find the list of committee members at <http://sites01.lsu.edu/wp/ored/human-subjects-screening-committee-members/>.
- A Complete Application Includes All of the Following:
- (A) This completed form
 - (B) A brief project description (adequate to evaluate risks to subjects and to explain your responses to Parts 1&2)
 - (C) Copies of all instruments to be used.
*If this proposal is part of a grant proposal, include a copy of the proposal and all recruitment material.
 - (D) The consent form that you will use in the study (see part 3 for more information.)
 - (E) Certificate of Completion of Human Subjects Protection Training for all personnel involved in the project, including students who are involved with testing or handling data, unless already on file with the IRB. Training link: (<http://phrp.nihtraining.com/users/login.php>)
 - (F) Signed copy of the IRB Security of Data Agreement: (<https://sites01.lsu.edu/wp/ored/files/2013/07/IRB-Security-of-Data.pdf>)

1) Principal Investigator: Rank:

Dept: Ph: E-mail:

2) Co Investigator(s): please include department, rank, phone and e-mail for each
*If the Principal Investigator is a student, identify and name supervising professor in this space

3) Project Title:

4) Proposal? (yes or no) ☒ yes If Yes, LSU Proposal Number

Also, if YES, either ☒ This application completely matches the scope of work in the grant
OR ☐ More IRB Applications will be filed later

5) Subject pool (e.g. Psychology students)

*Indicate any "vulnerable populations" to be used: (children <18 the mentally impaired, pregnant women, the ages, other). Projects with incarcerated persons cannot be exempted.

6) PI Signature  Date (no per signatures)

**** I certify my responses are accurate and complete.** If the project scope or design is later changes, I will resubmit for review. I will obtain written approval from the Authorized Representative of all non-LSU institutions in which the study is conducted. I also understand that it is my responsibility to maintain copies of all consent forms at LSU for three years after completion of the study. If I leave LSU before that time the consent forms should be preserved in the Departmental Office.

Screening Committee Action:	<input type="radio"/> Exempted	<input type="radio"/> Not Exempted	Category/Paragraph	<input type="text"/>
Signed Consent Waived?:	<input type="radio"/> Yes or <input type="radio"/> No			
Reviewer	Signature		Date	

Continue on the next page

Abstract. The purpose of this study is to design a metric of success for an Intelligent Tutoring System (ITS) which gives programming problems for students to solve.

Description. We have partially constructed an ITS (intelligent tutoring system) which can issue problems for the student to solve. We would like to examine the differences (particularly in performance) between solving the problems through an ITS versus a conventional paper-based approach.

After we seat the student in a private and distraction-free conference room, the student will solve problems on the ITS or on paper for up to an hour, after which the system will cue the student that the trial is concluded. Once the trial has concluded, the test administrator will stop the ITS (if applicable) for the debriefing phase.

We will conduct at least one experiment, which is to test mean performance differences between the paper-based and computer-based contexts. Further experiments may test differences between two distinct problem sets given by the ITS; for example, performance differences due to differing progressions of Bloom taxonomic levels, problem difficulties, or concepts. All further experiments will test performance differences due to manipulations on the problems (rather than the interface, the test administrator's instructions, or other potential variables).

Consent Form

Study Title: EDUCAT: Educational Data-Modelling for Unified Computerized Assessment and Teaching

Performance Site: Louisiana State University and Agricultural Mechanical College

Investigators:

- PI: Steven R. Brandt, sbrandt@cct.lsu.edu, (225) 287-8548
- Co-PI: Dennis Castleberry, dcastl2@tigers.lsu.edu, (225) 578-5912

Purpose of Study: This study seeks to measure the success of an intelligent tutoring system (ITS).

Subject Inclusion: Individuals between the ages of 18 and 65 with normal or corrected to normal vision, who are enrolled in introductory-level programming courses.

Number of subjects: 300

Study Procedures: The study will take approximately 1 hour to complete. Participants will be asked to perform a programming-related tasks.

Benefits: This study will advance knowledge in intelligent tutoring systems and their effects on learning problem-solving techniques in STEM courses.

Risks: There are no known risks.

Right to Refuse: Subjects may choose not to participate or to withdraw from the study at any time without penalty or loss of any benefit to which they might otherwise be entitled.

Privacy: Results of the study may be published, but no names or identifying information will be included in the publication. Subject identity will remain confidential unless disclosure is required by law.

Compensation: Subjects will be given extra credit simply for showing up to their appointment. As incentive for solving problems, they will be given additional extra credit for problems they solve correctly.

****Please sign and submit this document with your IRB application****

Security of Data

Number: PS06.20

SECURITY OF DATA

PURPOSE

I certify that I have read and will follow LSU's policy on security of data – PS06.20 (<http://sites01.lsu.edu/wp/policiesprocedures/policies-procedures/6-20/>) and will follow best practices for security of confidential data (http://www.lsu.edu/it_services/its_security/best-practices/sensitive-data.php)

This Policy Statement outlines the responsibilities of all *users* in supporting and upholding the security of *data* at Louisiana State University regardless of *user's* affiliation or relation with the University, and irrespective of where the *data* is located, utilized, or accessed. All members of the University community have a responsibility to protect the confidentiality, integrity, and availability of *data* from unauthorized generation, access, modification, disclosure, transmission, or destruction. Specifically, this Policy Statement establishes important guidelines and restrictions regarding any and all use of *data* at, for, or through Louisiana State University. This policy is not exhaustive of all *user* responsibilities, but is intended to outline certain specific responsibilities that each *user* acknowledges, accepts, and agrees to follow when using *data* provided at, for, by and/or through the University. Violations of this policy may lead to disciplinary action up to and including dismissal, expulsion, and/or legal action. It is recommended that all personnel on your project be familiar with these policies and requirements for security of your data.

In addition it is recommended that PIs review any grant, non-disclosure/confidentiality agreement, or restricted data agreements before publishing articles using the data.

I certify that I have read and understand these policies

Name: Dennis G. LeBlanc
Date: 12/14/15

****Please sign and submit this document with your IRB application****

Security of Data

Number: PS06.20

SECURITY OF DATA

PURPOSE

I certify that I have read and will follow LSU's policy on security of data – PS06.20 (<http://sites01.lsu.edu/wp/policiesprocedures/policies-procedures/6-20/>) and will follow best practices for security of confidential data

(http://www.lsu.edu/it_services/its_security/best-practices/sensitive-data.php)

This Policy Statement outlines the responsibilities of all *users* in supporting and upholding the security of *data* at Louisiana State University regardless of *user's* affiliation or relation with the University, and irrespective of where the *data* is located, utilized, or accessed. All members of the University community have a responsibility to protect the confidentiality, integrity, and availability of *data* from unauthorized generation, access, modification, disclosure, transmission, or destruction. Specifically, this Policy Statement establishes important guidelines and restrictions regarding any and all use of *data* at, for, or through Louisiana State University. This policy is not exhaustive of all *user* responsibilities, but is intended to outline certain specific responsibilities that each *user* acknowledges, accepts, and agrees to follow when using *data* provided at, for, by and/or through the University. Violations of this policy may lead to disciplinary action up to and including dismissal, expulsion, and/or legal action. It is recommended that all personnel on your project be familiar with these policies and requirements for security of your data.

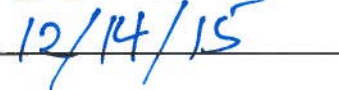
In addition it is recommended that PIs review any grant, non-disclosure/confidentiality agreement, or restricted data agreements before publishing articles using the data.

I certify that I have read and understand these policies

Name:



Date:





Certificate of Completion

The National Institutes of Health (NIH) Office of Extramural Research certifies that **Dennis Castleberry** successfully completed the NIH Web-based training course "Protecting Human Research Participants".

Date of completion: 03/06/2009

Certification Number: 184387



Certificate of Completion

The National Institutes of Health (NIH) Office of Extramural Research certifies that **Steven Brandt** successfully completed the NIH Web-based training course "Protecting Human Research Participants".

Date of completion: 08/12/2015

Certification Number: 1812669

Vita

Dennis Castleberry

Curriculum Vitae

1077 Louisiana Emerging Technology Center
110 Union Square, Baton Rouge LA 70802
dcastl2@tigers.lsu.edu
(255) 337-0226

EDUCATION

Ph.D., Computer Science <i>Louisiana State University</i> , Baton Rouge, LA	<i>In Progress</i>
B.S., Psychology <i>Louisiana State University</i> , Baton Rouge, LA Minor: Mathematics	<i>May 2008</i>
B.A., Philosophy <i>Louisiana State University</i> , Baton Rouge, LA Minor: Physics	<i>May 2007</i>

EXPERIENCE

Instructor , <i>Louisiana State University</i>	<i>08/2016 - Present</i>
<ul style="list-style-type: none">• Taught CSC 1254 (Advanced Programming with C++) and CSC 4103 (Operating Systems)• Scheduled to teach CSC 4101 (Programming Languages) in Spring 2017	
Teaching Assistant , <i>Louisiana State University</i>	<i>08/2011 - 08/2016</i>
<ul style="list-style-type: none">• Taught CSC 1253 and CSC 1254 (Intro to Programming with C/C++) as instructor of record since 08/2014• Developed automated intelligent tutoring system (ITS) and autograder for programming courses	
Research Assistant , <i>Center for Computation & Tech</i>	<i>06/2011 - 09/2014</i>
<ul style="list-style-type: none">• Developed executable paper frameworks to facilitate publication of computational scientific results	
IT Analyst , <i>Center for Computation & Tech</i>	<i>09/2007 - 00/2010</i>
<ul style="list-style-type: none">• Performed maintenance and repair on Linux systems	

RESEARCH PUBLICATIONS

Castleberry, Dennis and Brandt, Steven. "The Effect of Question Ordering Using Bloom's Taxonomy in an e-Learning Environment." *CSEIT 2016*. Singapore, 2016.

Castleberry, Dennis. "Inkling: An Executable Paper System for Scientific Applications." *SocialCom 2013*. Washington, D.C, 2013.

Castleberry, Dennis. "RChive: An Executable Paper System for R." *SCALA: Scientific Computing Around Louisiana*. New Orleans, Louisiana, 2013.

Castleberry et al. "The Prickly Pear Archive: A Hypermedia for Scholarly Publication." *XSEDE 2012*. Chicago: ACM. Print.

Krishnan et al. “Mojave: a Development Environment for the Cactus Computational Framework.” *XSEDE 2012*. Chicago: ACM. Print.

Brandt et al. “The Prickly Pear Archive.” *ICCS 2011*. Singapore: Elsevier. Print.

POSTERS & PRESENTATIONS

Castleberry, Dennis. “Educational Data Modelling for Unified Computerized Assessment and Tutoring.” *SCALA 2016*. Baton Rouge, LA, 2016.

Castleberry et al. “Automated Classification of Sleep Stages using EEG/EOG and R&K Rules.” Baton Rouge, LA, 2012.

SKILLS

- **Programming:** C/C++, Java, Haskell, R, MATLAB, Shell (bash), L^AT_EX, PHP, JavaScript/jQuery, HTML, CSS, MySQL.