REAL WORLD USAGE OF AN ADAPTIVE TESTING ALGORITHM TO UNCOVER LATENT KNOWLEDGE

Danny Lynch and Colm P. Howlin

Research and Development Laboratory, CCKF Limited (IRELAND)

Abstract

At the beginning of every course, it can be expected that several students already have some knowledge of the curriculum. For learning efficiency it is important to quickly identify this latent knowledge. This enables students to begin new learning immediately which in turn prevents the frustration or boredom that might be caused otherwise. A component of the Realize[it] learning environment that uncovers this latent knowledge is an algorithm called *Determine Knowledge*. This is offered as an adaptive pretest at the beginning of each course. It allows new students to be automatically placed at the appropriate position within the course. In this paper, we detail the algorithm and review its relative strengths and weaknesses. In particular we discuss the real-world usage of Determine Knowledge run in a higher educational institute in the United States over a two year period. In total this encompasses over 455,000 Determine Knowledge operations performed by over 48,500 unique students. We finish the paper by outlining some of the enhancements made to the algorithm based on the analysis of this data.

Keywords: Latent Knowledge, Technology Enhanced Learning, Intelligent Learning Systems.

1 INTRODUCTION

There are now many intelligent learning environments that can guide students through the learning materials of a course or curriculum. These systems typically involve some form of curriculum sequencing, where learning paths are defined through a space of learning objectives and learning materials [1]. On top of this curricular guidance, these systems often strive to identify the preferences, knowledge and skills of each student. These characteristics can assist in providing a more personalized and adaptive learning experience for each individual [2].

Before students begin a new course in such a learning environment, it is reasonable to expect they might already have some knowledge of the associated curriculum. We say that each student has some *latent knowledge*. This could have been acquired through a combination of external sources such as taking related courses elsewhere or even by accessing the wealth of educational materials online. In all cases, the learning environment should strive to quickly uncover the latent knowledge of each individual. This promotes learning efficiency as each student is able to begin learning new material immediately and so is not forced to repeat materials already known to them. By respecting what each individual already knows, the learning system avoids much potential frustration and boredom on the part of the student.

In this paper we will discuss the approach to uncovering latent knowledge as implemented in the Realize[it] learning environment. This takes the form of an algorithm known as Determine Knowledge (DK) which is detailed in Section 2. The theoretical performance of this algorithm along with its real-world usage over a two year period will be investigated in Section 3. From this data, we consider how this algorithm was utilized in reality and investigate where additional performance enhancements can be made for future iterations of the algorithm.

1.1 The Realize[it] System

Realize[it] is a comprehensive learning environment that unifies a multitude of aspects of the learning process [3]. This includes tools for importing, designing and managing the curriculum and the learning content. There is a sophisticated instructional delivery mechanism that personalizes and adapts to each individual while also capturing the stream of highly granular data generated by each student. These feed into several institutional reporting and analytics tools. Realize[it] supports all of the typical roles found in the educational sector; this includes the learner, instructor, course developer, the business intelligence teams and the administration users.

Note that a fundamental approach taken by Realize[it] is the separation of the curriculum from the content. The curriculum is represented as a set of connected and related concepts (see Section 2.1 for details) and is used to drive the direction of the learning. The content on the other hand delivers the learning to the individual. Just as a teacher can teach the same concept in many different ways, Realize[it] can have multiple pieces and types of content available for each concept in the curriculum.

2 UNCOVERING LATENT KNOWLEDGE

2.1 The Curriculum Structure

A curriculum is traditionally thought of as a hierarchical representation of all concepts and knowledge within a particular domain. The items at the bottom of the hierarchy represent the finest grained pieces of knowledge in the curriculum definition. While the hierarchy does signify part of the relationship that exists between these granular knowledge items, it fails to capture the full picture. Realize[it] supplements this hierarchical representation with a second structure known as the Curriculum Prerequisite Network (CPN). This is a directed acyclic graph that describes the relationships between the most fundamental pieces of knowledge in a curriculum, known as knowledge items. These knowledge items are represented as nodes on the graph and the edges represent the relationships between the items. The CPN is structured so that knowledge items are preceded in the graph by their prerequisite items. These prerequisite items are the ones that should be mastered before the current knowledge item can be studied. Learning in the Realize[it] system is driven by the CPN. In this way it provides a map to aid a learner in navigating their way through a curriculum. An example of such a network is shown in Fig. 1. This is the structure of a real network for a Grade 6 Probability objective in the Realize[it] learning environment. This particular network has 23 nodes, 27 prerequisites links and is itself a subset of a much larger mathematics network.

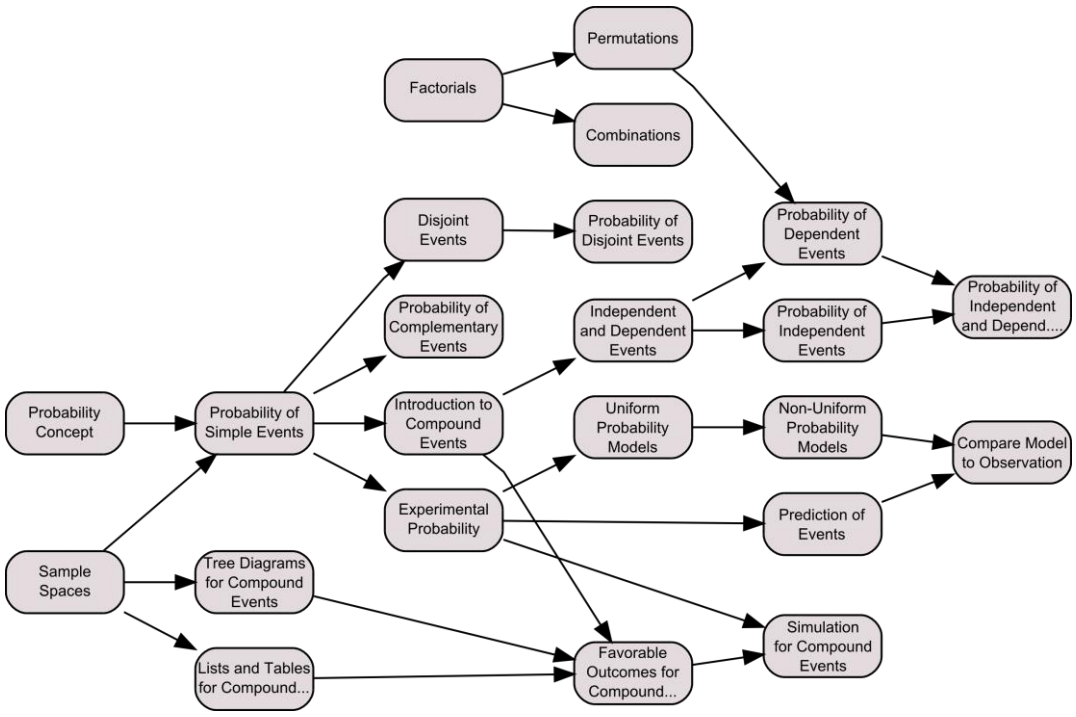

Figure 1: A 23 node curriculum prerequisite network for Grade 6 Probability

2.2 The Latent Knowledge State

2.2.1 Knowledge Spaces

The theory of Knowledge Spaces, first introduced by Doigon and Falmagne in 1985, is a combinatorial structure that describes the possible states of knowledge that a learner may have [4-5]. In this framework, a graph structure is defined where each node represents a question and a directed edge between two nodes represents a prerequisite connection. Mastery of certain nodes is necessary in order to begin others. A *knowledge state* is then defined as a feasible subset of nodes that a person could be capable of answering. Given a graph structure, the list of possible knowledge states can be combinatorially calculated. The *knowledge space* is then the collection of all these knowledge states.

On the outset, this appears similar in design to the curriculum structure described above in Section 2.1. The fundamental difference relates to what a node signifies in each construct. Rather than being a single question, a node in Realize[it] represents a granular piece of knowledge. Due to the separation of curriculum and content, there can be various learning materials available to teach any specific node. Moreover each one of these typically includes a repertoire of questions that the system can ask the student on. Hence while a node in the theory of Knowledge Spaces represents a single question, in Realize[it] it can represent a multitude along with associated learning material.

In any case, the theory of Knowledge Spaces can be leveraged due to the set-theoretic similarity between these two structures. We find that the graph structure defined in Fig. 1 allows for a total of 6,932 knowledge states that a student may have in this curriculum. The distribution of these states by size, which represents the number of nodes a student could know, is shown in Fig. 2. For example, we find there are only 3 knowledge states with one node while there are 850 states with thirteen nodes.

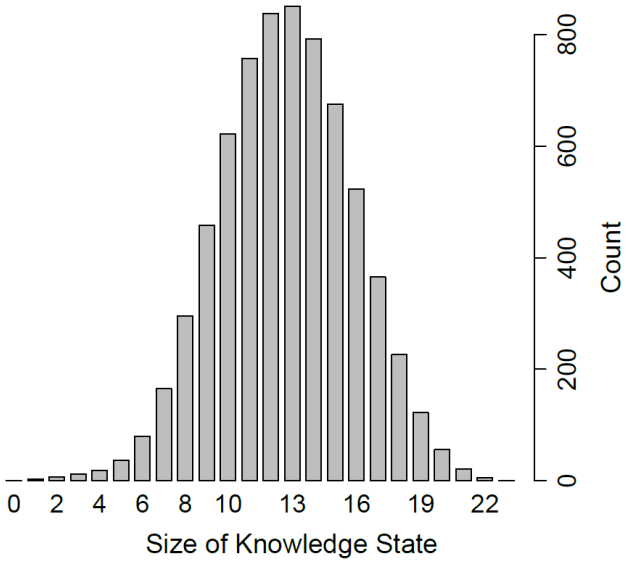

Figure 2: The distribution of knowledge states by size

2.2.2 The Latent Knowledge State

Via this Knowledge Space representation, we can appreciate how students will have different knowledge states at different points during their learning process. The overarching goal of the learning process is to bring students from their initial state to the single state where they know all nodes. However this initial state can be different for each individual, and in fact can be any one of the feasible knowledge states. We label this as the *latent knowledge state*.

When a new student is introduced to Realize[it], the system aims to identify their latent knowledge state; the subset of nodes from the curriculum to which the student can already answer. One naïve approach might be to test the student on each node from the curriculum. This is obviously impractical due to time constraints and the cognitive burden on the student. More intelligent methods of selecting knowledge items and interpreting the results are needed. This is akin to Computerized Adaptive Testing where a pool of items or questions is optimally searched [6].

2.3 The Determine Knowledge Algorithm

The Determine Knowledge (DK) algorithm was developed for Realize[it] to efficiently and accurately identify the latent state of each student. This process begins with all nodes in the curriculum network having the potential to be part of the latent state. The algorithm selects the node which will tell the system the most information about that student's knowledge. The student is tested on this node and their response dictates which nodes should be included or excluded as part of the latent state. This process makes use of the prerequisite definition of the curriculum. That is:

- If a student knows a node, then they must also know all its prerequisites,
- If a student does not know a node, then they cannot know its post-requisites.

Hence based on the student response, the algorithm can narrow down the feasible nodes the student might know. This process is repeated until no nodes are left. A step-by-step overview of the algorithm is supplied below.

Determine Knowledge Algorithm

1. Start with an empty knowledge state and an item pool containing all nodes.
2. Calculate the information content for each item in the pool. This is based on the structure of the network and defined using Shannon Entropy.
3. Choose the item to be tested with the maximum information content.
4. Based on the student response, remove appropriate items from the pool and update the knowledge state accordingly.
    a. If correct, add items and prerequisites to knowledge state as known
    b. If incorrect, add items and post-requisites to knowledge state as unknown
5. Repeat steps 2 through 4 until the item pool is empty

At the completion of the Determine Knowledge algorithm, there is a set of nodes from the curriculum that are marked as known. These form the latent knowledge state for that student. As a result, their personalized map of the curriculum will be updated to reflect this knowledge. This becomes their starting point on the curriculum. They are not forced to cover the material already known and can begin new learning immediately.

3 ALGORITHM PERFORMANCE

For the Determine Knowledge algorithm to be considered effective, it has to be both efficient and accurate. In other words the whole process should be quick to complete and it should result in students arriving at an appropriate knowledge state. In the following section we investigate this performance at two levels; a review of the theoretical work previously published in [7] and an examination of the real world usage of the algorithm.

3.1 Theoretical Performance

From a theoretical standpoint, the DK algorithm was evaluated with simulated students which is a common technique found in education research [8]. For each possible knowledge state in a given network, a simulated student was created with that state defining their latent knowledge. When these students were asked on nodes from the domain, they would respond appropriately based on their latent state. To emulate more realistic behavior, a probabilistic response rate parameter was also introduced. This resulted in cases where a simulated student would not respond appropriately. For instance, a response rate of 0.9 signifies the student responds accurately every 9 out of 10 questions. This is commonly referred to in the literature as the probability of a lucky guess or careless error, and they are treated as identical for this study. For comparison purposes, the same simulated students were also evaluated using a second algorithm that can also uncover latent knowledge. Specifically this algorithm originates from Knowledge Space theory (KST); further details can be found in [5].

Overall, this process took simulated students with pre-defined latent states and applied two uncovering algorithms. Two performance factors were measured from the results; the total number of nodes asked (to measure algorithm efficiency) and the count of differences between the defined and the uncovered latent knowledge state (to measure algorithm error rate). Detailed results of this work along with comprehensive explanations can be found in [7].

The primary visualization used to represent these results is shown below in Fig. 3 for two separate curriculum networks. We see that each network has two associated heat maps; one for efficiency based on the nodes asked, and the second for the accuracy. The coordinates of these heat maps represent the two simulation parameters: the size of the latent knowledge state and the student response rate. For all parameter pairings, the data for each algorithm was averaged and compared to determine if the results were significantly different (at a 95% confidence level). The coloring at each coordinate represents which algorithm performed better for the simulations run with those parameters.

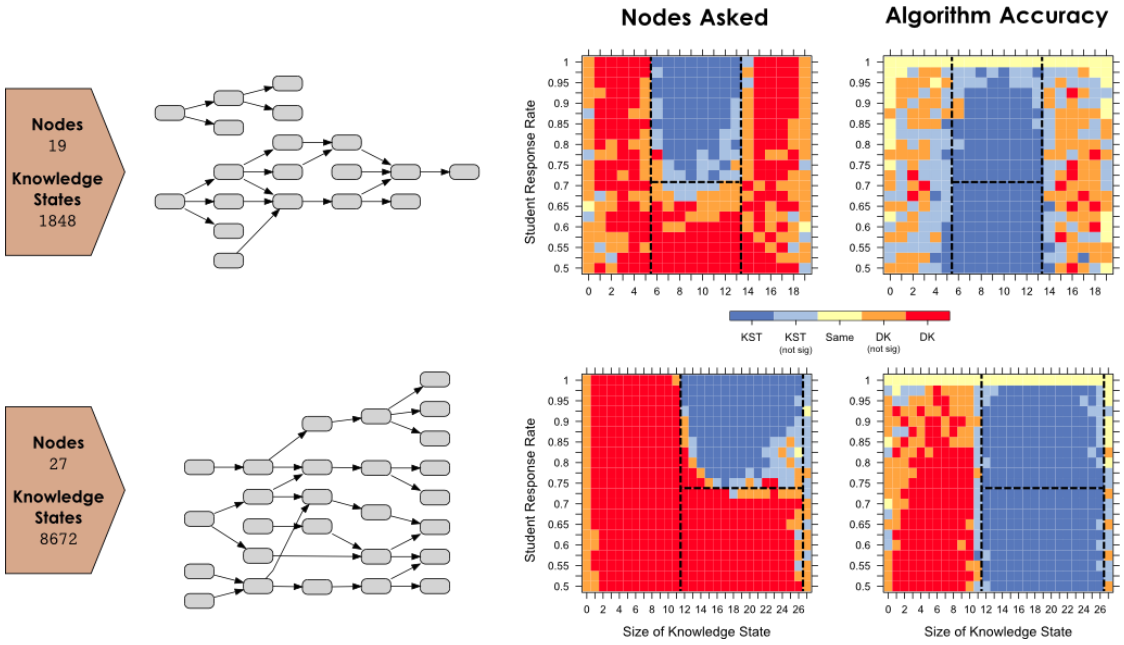

Figure 3: Heat maps signifying the dominant algorithm in terms of number of nodes asked and algorithm accuracy run on two networks of different characteristics

The simulation process was run on many networks of various sizes and complexities. Similar outcome areas were obtained irrespective of the network. Three areas of similarity can be seen on the heat maps of Fig. 3 as highlighted by the overlaid black 'H' shape.

- For smaller knowledge state sizes (to the left of the 'H' shape), the DK algorithm in general asks on fewer nodes and has a higher or similar accuracy.
- For knowledge states of middling size (within the 'H' shape), both algorithms have areas of strength. For high student response rates, the KST algorithm is both more accurate and efficient. However DK becomes more efficient as the student response rate lowers. The KST algorithm does remain more accurate but it requires more nodes to maintain this accuracy.
- For larger knowledge state sizes (to the right of the 'H' shape), the DK algorithm tends to ask on fewer nodes. In this region both algorithms have similar accuracy.

Overall we find that the Determine Knowledge algorithm is in general more efficient by asking students on fewer nodes. However the Knowledge Spaces algorithm typically has a higher accuracy. As concluded in [7], there is a clear trade-off between the number of nodes asked upon and the accuracy of the state. Yet as this trade off depends on the values of the parameters, a natural question arises as to what values, if any, are most typical for real students.

3.2 Real World Usage

As part of the performance research on Determine Knowledge, anonymized data from a higher educational institution based in the United States was supplied for analysis. This specific institution has been running Realize[it] for over two years and in doing so has generated a particularly large dataset. It comprises of 455,660 Determine Knowledge operations which were initiated by 48,498 unique students. Overall there was a median of 4 DK operations per student per course; this value is to be expected as students are able to do a DK on each objective that they are assigned within a course. There were a total of 89 different courses run over the two years. Broadly speaking, these covered subject domains such as English, Math, Business, Humanities, Law, Computing and Science. Moreover as these courses were run multiple times (typically once every six weeks), the dataset actually covers 571 course occurrences.

As per the DK procedure, each running of the algorithm resulted in the system identifying one of the possible knowledge states as being the latent one. However this is not necessarily the same as the student's actual latent state. Slips and guesses during the course of the DK process could cause the algorithm to select an incorrect state. For this reason, this data cannot be used to determine the accuracy of the algorithm. Nevertheless the simulation research of [7] has shown that the differences between the identified and actual latent states are typically quite small, and so we can consider the size of the identified states to be an accurate representation. This allows us to gauge the real values of the parameter *Size of Knowledge State* used in the simulations and heat maps of Fig. 3. Rather than using magnitudes, we consider this size as a percentage of the total number of nodes in the network. This facilitates a direct comparison between objectives of various sizes. For this dataset, the distribution of these DK operations by percentage size is shown in Fig. 4.

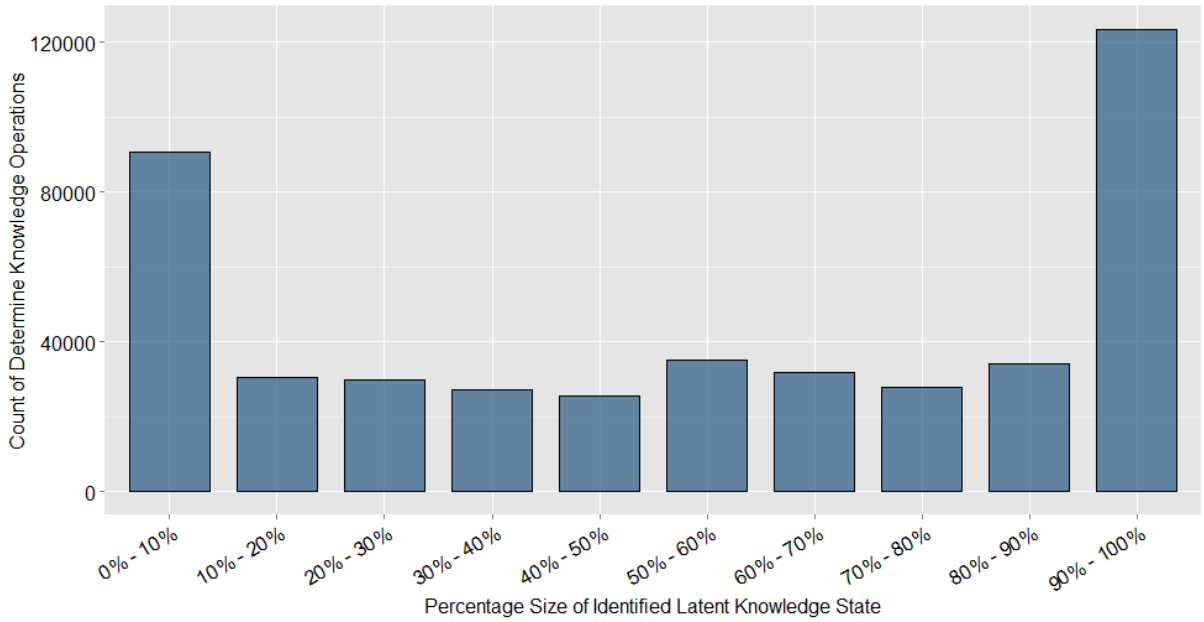

Figure 4: Distribution of identified latent knowledge states by percentage size

The most striking feature of this figure is the bimodal nature of the distribution; most students either know very little of the objective or know most of it. We note that similar distributions were also found across individual courses and subject domains. When limiting to the extreme values, we find that 18.9% of all DK operations resulted in the empty state and 24.2% of them resulted in the complete state. Therefore a significant proportion of students had latent states that correspond with regions of the simulation where the DK algorithm has been shown to be more efficient and have greater or similar accuracy. The remaining 56.9% of DK operations occurred when the students had partial knowledge of the objective. Compared to the extreme categories, we find that these DK operations are relatively evenly distributed by percentage size. Hence the correspondence between these operations and the simulation is spread across all identified regions.

Beyond the size of the latent state, this dataset also permits us to determine the number of nodes that were asked during each running of the DK algorithm. This is the same metric that was used to gauge the efficiency of the algorithms in the simulations of Section 3.1. For comparison reasons, we also deal with this number as a percentage out of the total number of nodes in the objective. We find that the median DK algorithm asked the student on 21.4% of the nodes. This can be further separated out by the percentage size of the identified latent knowledge state. These results can be seen by the multiple box-and-whisker plots shown in Fig. 5.

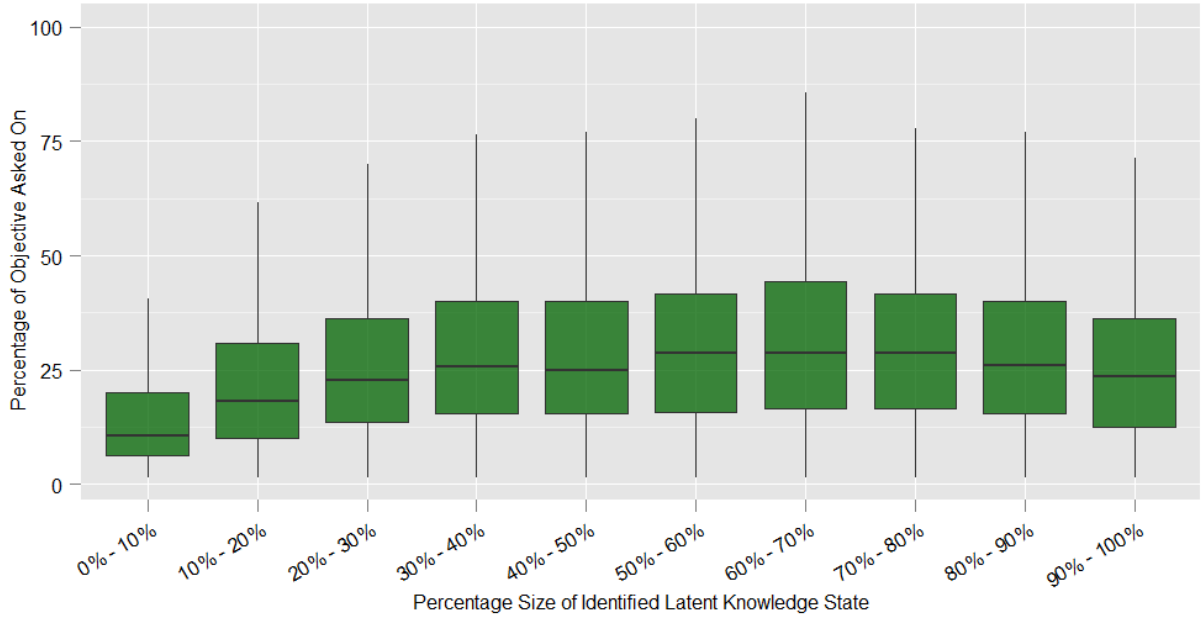

Figure 5: Percentage of curriculum asked on by latent knowledge state size

We immediately notice how the median percentage number of nodes asked varies depending on the size of the underlying latent knowledge state. The DK algorithm was found to be most efficient for latent states in the category 0%-10%, where the median student was only asked on 10.7% of the nodes. This median then gradually increases for every successive category of latent knowledge state size. This continues to a peak at the 60%-70% category where the median student is asked on 28.6% of the nodes. The remaining categories then see decreases in this median. Similarly we find that the range of the whiskers in the box-and-whisker plots follows a related pattern. Overall we note that these real-world observations closely reflect the simulations. In particular, the simulated number of nodes asked exhibited similar dependencies on the latent knowledge state size. We conclude that the DK algorithm is more efficient for latent states of lower and higher sizes compared to those in-between. Nonetheless having the worst case median below 29% is still quite respectable.

4 CONCLUSIONS

When a student begins a new course, the Determine Knowledge algorithm can be used to identify any relevant knowledge the student already has of the curriculum. This provides a mechanism for students to uncover or verify their latent knowledge and be allowed to work on new material immediately. The simulations which examined the performance of the Determine Knowledge algorithm had the net effect of identifying areas of relative strength and weakness. Specifically these areas have a clear dependency on the value of two parameters: the size of the latent knowledge state and the student response rate. One natural question that arose is what would be considered typical values. Perhaps if these were known prior to running the algorithm, then it could be tailored accordingly. To investigate this, a large set of real-world data of students using the DK algorithm was sourced and analyzed. This demonstrated how DK was being used in reality and what the median student experienced. It further showed how the efficiency of the algorithm varied across the student population. The real-world data was found to agree with the simulations in terms of efficiency. Moreover, insights into realistic values for some of the discussed parameters were identified. Although these values would not typically by known for any individual ahead of running Determine Knowledge, improvements to the algorithm can be made to take advantage of this general knowledge. We conclude this paper with a brief discussion on how this has been applied in the Realize^{it} system.

4.1 Algorithm Improvements

Knowing the size of the latent knowledge state before running DK is rather unlikely given that the aim of the algorithm is to identify this unknown latent state. However an estimation of this size could still be beneficial in speeding up the algorithm. One method of estimation can be achieved by simply asking the students directly. As shown in Fig. 6, this now manifests itself in Realize[it] through the use of a slider before starting the Determine Knowledge process. In particular the real-world data has shown that a high proportion of students often fall into the two extreme cases: not knowing any of the material or knowing all of the material. These two cases are now catered for on the slider and given a selection the DK algorithm is tailored appropriately. Further research will investigate if these selections actually cause greater efficiency for those students that can indeed demonstrate their selection.

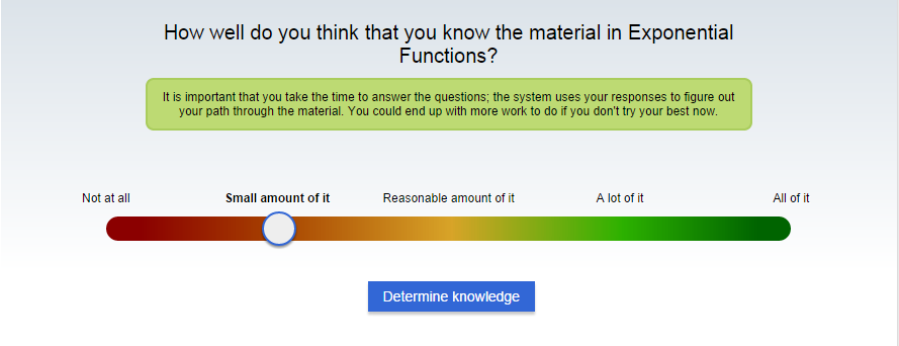

Figure 6: Slider to help tailor the Determine Knowledge algorithm for each student

Obtaining the accuracy of the student response rate beforehand is also difficult. Further this is not a parameter that can easily be estimated by either the students themselves or the facility. So rather than adjusting the DK algorithm based on the response rate, separate functionality has been implemented in the Realize[it] system to recognize that slips and guesses are possible. This is achieved through the systems use of a probabilistic measure of ability, and the application of Item Response Theory [9] to all questions answered by the students. As students continue working through the system, they can be directed back to completed nodes if regressions in knowledge are detected. This includes nodes that were skipped as part of the DK procedure. Overall the system will use all evidence it collects to build its confidence that the student has mastery on all the curricular concepts.

REFERENCES

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

[2] Brusilovsky, P., & Peylo, C. (2003). Adaptive and intelligent web-based educational systems. *International Journal of Artificial Intelligence in Education*,*13*(2), 159-172.

[3] Howlin, C.P. (2013). The Realize[it] System

[4] Falmagne, J. C., Cosyn, E., Doignon, J. P., & Thiéry, N. (2006). The assessment of knowledge, in theory and in practice. In *Formal concept analysis*(pp. 61-79). Springer Berlin Heidelberg.

[5] Falmagne, J. C., & Doignon, J. P. (2011). Learning spaces. Springer.

[6] Wainer, H., Dorans, N. J., Flaugher, R., Green, B. F., & Mislevy, R. J. (2000). Computerized adaptive testing: A primer. Routledge.

[7] Lynch, D. J., & Howlin, C. P. (2014) Uncovering Latent Knowledge: A Comparison of Two Algorithms. User Modeling Adaptation and Personalization

[8] Abdullah, S. C., & Cooley, R. E. (2002). Using simulated students to evaluate an adaptive testing system. In *Computers in Education, 2002. Proceedings. International Conference on* (pp. 614-618). IEEE.

[9] Drasgow, F., & Hulin, C. L. (1990). Item response theory.