Refining Learning Maps with Data Fitting Techniques: Searching for Better Fitting Learning Maps

Abstract: Learning sciences needs quantitative methods for comparing alternative theories of what students are learning. This study investigated the accuracy of a learning map and its utility to predict student responses. Our data included a learning map detailing a hierarchical prerequisite skill graph and student responses to questions developed specifically to assess the concepts and skills represented in the map. Each question aligned to one skill in the map, and each skill had one or more prerequisite skills. Our research goal was to seek improvements to the knowledge representation in the map using an iterative process. We applied a greedy iterative search algorithm to simplify the learning map by merging nodes together. Each successive merge resulted in a model with one skill less than the previous model. We share the results of the revised model, its reliability and reproducibility, and discuss the face validity of the most significant merges.

Introduction
Cognitive models are used to represent how one’s knowledge may be organized (Gierl, Wang, & Zhou, 2008). As such, they contain descriptions of component pieces of knowledge and connections among the components to indicate how understanding develops in a specified domain (Gierl, Wang, & Zhou, 2008). Different authors have described various cognitive models, including learning maps (Popham, 2011), learning trajectories (Clements & Sarama, 2004), and learning hierarchies (Gagné, 1968). Learning maps use linear sequences of learning goals and are useful for instructional planning (Popham, 2011). A learning trajectory includes a learning goal, a developmental progression defining the levels of thinking students pass through as they work toward the defined goal, and a set of learning activities or experiences that assist students in reaching the defined goal (Clements & Sarama, 2004). As their name implies, learning hierarchies model prerequisite knowledge components in hierarchies, allowing multiple pathways to extend from one prerequisite skill to multiple learning goals (Gagné, 1968).

The learning map extends the notion of a learning hierarchy by representing domain knowledge as a network of component skills and connections, allowing for multiple paths from prerequisites to learning goals. While multiple paths add complexity to the cognitive model, they allow the learning map to represent the potential learning of a broad range of individuals who may experience difficulties traversing certain pathways due to disabilities or particular learning preferences. As such, the learning map provides a flexible model of learning that is consistent with recent advances in universal design for learning (Center for Applied Special Technology [CAST], 2012).

In the present study we examine a small section of the learning map and investigate the effects of permuting the topology of the hierarchy. Skills and concepts are represented by latent nodes in the learning map. Directed edges represent the prerequisite relationship among latent nodes and also represent the relationship between those nodes and their associated test items. We present a simple method for improving the predictive power of the learning map by combining latent nodes. We report our initial results on the fit improvement, stability of the resulting map, and interpretation of the algorithms chosen node combinations.

This work connects with literature on searching for better fitting cognitive models. Several non-hierarchical cognitive models have been developed to represent the relationship between knowledge components (KCs) in the form of prerequisite skill maps. These cognitive models have been developed to help intelligent tutors, as well as experts, determine student mastery of KCs. A number of technical approaches have been developed to evaluate cognitive models developed by domain experts. One approach is Learning Factors Analysis (LFA), developed by Cen, Koedinger and Junker (2006) to help the Educational Data Mining (EDM) community evaluate different cognitive models.

There are several different methods for analyzing skills. Tatsuoka (1983) introduced the rule space method for representing and determining how well students understood the underlying skills (or rules as the paper calls it) for test items. Additionally, the method is used to identify any erroneous classification or misconceptions of students in responding to test items. Barnes (2005) utilized the Q-matrix method from Tatsuoka’s rule space method to organize combinations of skills into distinct latent classes and assign students to latent classes based on level of mastery. Additive Factor Models (AFM) also utilize the Q-matrix but with a multiple logistic regression model which predicts student performance based on a number of factors, primarily the number of opportunities a students has to demonstrate a particular skill. Cen (2009) reported that AFM did not accurately predict items involving conjunctive skills and hence introduced the Conjunctive Factor Model (CFM) to improve predictions in this area. In addition to latent skill cognitive models, item to item knowledge structures have also been learned from empirical data using Bayesian Network structure learning and partial order knowledge structures (Desmarais, Gagnon 2006).
Our approach to simple merging of skills was inspired by Learning Factors Analysis, which uses a combinatorial search to determine which model best fits student data. The combinatorial search consists of three different types of operations: splitting, merging or adding existing KCs. Splits occur when a knowledge component is determined to be composed of more than one skill, and hence splits into multiple skills. One or more skills are merged if they are determined to be inseparable skills, given student data. The add operation involves the inclusion of a completely new skill to the original map (Cen, 2009).

Other researchers have tried to extend LFA to other subject domains. Leszczenski and Beck (2007) introduced a scalable application of the LFA framework in the context of reading knowledge transfer. The problem with this approach is that the search was unstable and could give different results each time the search was run. Instead of determining a student model given an initial human generated model, Li, Cohen, Noboru, and Koedinger (2011) proposed a method for automatically generating the KCs from student responses to individual items. Although their method resulted in the best fit among the other candidates, it may not generalize for models with less coarse grained KCs.

Other models have focused on the determination of a student’s knowledge of certain skills. Logistic regression has been used to trace multiple sub-skills of a given skill (Xu and Mostow, 2011). Pavlik, Cen, and Koedinger (2009) proposed a method for automatically deriving a cognitive model by generating a Q-matrix, which provides a representation of the KCs required for each test item.

In this work we follow the process described by Cen, Koedinger and Junker (2006). This technique can be used to analyze hypothesized learning maps and consider whether small improvements to the model result in a better fit to the data. In this method two different approaches were studied to determine the best skill map from an initial graph. Cen, Koedinger, and Junker (2006) suggested three types of operations, i.e., merges, splits, and adds. However, in this study, we used only merge operations given the already highly granular quality of our initial, subject matter expert derived learning map.

**Initial Learning Map**

This study examined a section of the learning map containing 15 concepts and skills related to understanding integers. The map was developed using mathematics educational literature describing how students learn to understand and operate with integers. The set of integers includes the whole numbers and their opposites, presenting many students their first exposure to negative numbers (Van de Walle, Bay-Williams, Karp, & Lovin, 2014). Although many students have prior knowledge of negative values within contexts such as debt or temperatures below freezing, they often struggle when first learning to work with negative numbers. Proficiency with integers includes understanding opposite numbers, comparing integers, representing integers on number lines and graphs, and using integers in real world problem contexts. The learning map shown in Figure 1 illustrates the component concepts and skills that comprise such understanding. This map suggests that students should learn to identify opposite numbers (M-1104) and integers (M-1289) in preparation for comparing and ordering integers (M-1133, M-1135, M-1140) as well as representing integers on number lines (M-1118, M-1120, M-1108, M-1126) and coordinate planes (M-1122, M-1124). Because integers challenge the initial counting strategies students learned for positive numbers, it is beneficial for students to work with integers in real-world contexts (M-1106, M-1105, M-1127, M-1128) (Van de Walle, Bay-Williams, Lovin, & Karp, 2014).

The data for this study was gathered from student responses to 25 test items aligned to the 15 skills shown in the learning map in Figure 1. All of the test items were multiple choice questions, with four answer options per question. Each skill was assessed by one or more items. As part of the test development process, subject matter experts confirmed the alignment of each item to its associated skill, meaning that the item was judged by experts to evoke the intended skill. Therefore, when a student answered a test item correctly, we assumed in this study that the student had mastered the skill associated with that test item. Furthermore, due to the hierarchical structure of the learning map, items associated with skills lower in the learning map were assumed to be more difficult, i.e., require more skills, than items associated with skills higher in the learning map.

In addition to the graph, we utilized a data-set containing the responses of 2,846 students answering the same sequence of 25 items in the learning map. All the students were chosen from middle schools in a midwestern state from grades 6 (8%), 7 (49%), 8 (39%) and 10 (4%). The students’ responses were dichotomous, ‘1’ for correct and ‘0’ otherwise.
Methodology

Merge Operation
In all of the experiments our sole manipulation of the map was to merge latent nodes. A merge operation occurred when two skills adjacent to each other in the map were combined into one skill. Items from both skills that were merged were reattached to the new single skill. The prerequisites of the constituent skills became prerequisites of the merged skill and the same applied to the post-requisites. An example merge operation on a section of the skill map is shown below. The skill maps before and after the merge operation are shown in Figure 2. M-1289 and M-1133 are the skills that were merged into a single skill, named “M1289xM1133”. Note that the names of the skill hold no meaning of their own, just as the labels of the arcs between the skills.

Evaluation Procedure
For evaluating the models we used per student per item cross validation with 5 student folds and 3 item folds. Our student and item folds were chosen randomly for our evaluation. More details about how the cross-validation was done can be found in the technical document (1). We used the Root Mean Squared Error (RMSE) metric to evaluate the results of the experiments. RMSE is calculated by squaring the differences between each actual value and predicted value and then finding the average value of the differences. Taking the square root of the average will give the RMSE value for the model.
Experiment 1: Iterative Search

The purpose of this experiment was to take the original learning map and to create and run a search algorithm to find a better, more predictive, learning map. This experiment uses a greedy search algorithm to generate the new models.

In this experiment we started with the initial learning map shown in Figure 1 and created a Bayesian network to represent this map. Starting with the original map we programmatically found all possible skill pairs that could be merged. The algorithm only considered merging adjacent skills, or skills that shared an edge between them. Each possible merge was evaluated using the procedure previously described, and the best merge was chosen based on the map with the lowest cross-validated prediction error. We applied the best possible merge to the map and this resulted in a map with one less skill. The new map was used as the input to the next iteration of the algorithm. This technique was iteratively applied until all the skills were merged into a single skill. Further details of the iterative search algorithm can be found in the technical document.

Results and Analysis

Figure 3 shows a table and a graph of the results from the iterative search. The search started at iteration 0, which was the initial skill map consisting of 15 skills before any merges were applied to it. The search ended at iteration 14, which is a graph consisting of just one skill with all the items attached to that one skill. The best models from each iteration are shown below. We recorded AUC, RMSE, accuracy, AIC, and BIC metrics, although we only used RMSE to choose the best models at each iteration and to guide our search. Ultimately, we chose RMSE as the deciding metric.

![Skills + RMSE](image)

(a) Skill Accuracy

![Best Model Skill Map (11 skills at iteration 4)](image)

(b) Best Model Skill Map (11 skills at iteration 4)

Figure 3. a) The chart of results and b) the graph of the best skill model.
The results show that the best RMSE obtained was from the 11-skill map at iteration 4 with an RMSE of 0.37238. This is slightly better than the original skill map with RMSE of 0.37451. The 11-skill map has a small but significant improvement \((p = 6.22624E-68)\) from the original skill map. The graph shown in Figure 3b also shows that models consisting of between 9 and 12 skills have similar RMSE values and are alternative choices for a best model depending on the level of skill granularity desired. Those models are also significantly better than the original model. A significance table can be found in the technical report.

In addition to looking at which model best predicted actual responses, we examined which skills were being merged throughout our iterative search to see if we could find any general trends. A list of the merges can be found in the technical report. The individual skills are represented by their original numbers and a merged skill is represented by the numbers of each skill concatenated with an ‘x’. The numbering is in topological order, meaning that the skill highest up on the skill map was listed first for a merged skill. The first merge occurred for skills M-1128 and M-1127. Since skill M-1128 was a parent of skill M-1127, it is listed first in the combined skill name M-1128xM-1127. (See Figure 3b)

**Experiment 2: Stability Experiment**

In the previous experiment, every model was evaluated once and only once, which lead to the question of whether or not our results were stable. Our model evaluation used the Expectation Maximization (EM) algorithm, which is known to be affected by the starting value. For our original experiment we chose our starting points for EM randomly and only evaluated each model once. The authors, in earlier research, found that the starting point of the EM algorithm could make a difference in the converged value. In general the EM algorithm does converge to the correct value, but there are cases where it can converge to incorrect values or to the “opposite” value. Considering the range to be between 0-1, if the actual true value of a parameter was 0.3, EM could converge to \((1 - 0.3) = 0.7\) instead if the initial starting point was too far from the true value.

Our question was: if we were to run the iterative search experiment several times would we end up with the same results using different starting values for EM. Since it takes several hours just to evaluate a single model, running the entire search consisting of over 100 models to evaluate would take too long. The purpose of this experiment was to evaluate just the first iteration of the search ten times to see if the results converged to a single best graph.

For the first iteration of the algorithm there were sixteen possible merges that could happen. For each of these possible merges we evaluated the resulting model ten times. The evaluation used was the same evaluation as the iterative search experiment for which we tested stability. For each of the ten runs we set the random seed in MatLab to correspond to the run number. This gave us a different set of random numbers for each run of the 16 possible merges, where each merge got the same random seed within a run. Manually setting the random seed also meant our results for the stability experiment would be reproducible.

**Results and Analysis**

After evaluating all sixteen models from the first iteration ten times we kept a count of how many times a model was the best model and how many times a model was in the top 3 best models. RMSE was used to choose the best models since it was used to determine the best model in the iterative search experiment. The results are shown in the table below in Figure 4.

![Figure 4: Stability Graph](image)

Merge ‘g’ was in the top 3 the most times (6) and was also the best model the most times (3). Merge ‘x’ and merge ‘q’ also did well. Merge ‘x’ was in the top 3, five times and was the best model two times. Merge ‘q’ was in the top 3, three times and was the best model three times. Merges ‘t’ and ‘o’ also did well. The general
observation was that there was separation between good and bad merges but the best merge was not stable and did not converge.

We compared the graphs to our original iterative search experiment. In the original iterative search, the first two skills that were merged were skills M-1128 and M-1127, corresponding to merge ‘x’ in our stability experiment. The second two skills that were merged in the iterative search experiment were skills M-1140 and M-1118, corresponding to merge ‘g’ in the stability experiment. Both merges ‘x’ and ‘g’ were the best two graphs in the stability experiment. Although merge ‘g’ did slightly better in the stability experiment, the order in which the merges took place did not matter. The best model in the iterative search took place after 4 merges, which included merges ‘x’ and ‘g’. Although we could not run the stability experiment 10 times for all possible merges and merge paths, we believe that it has a decent chance to converge to the same best model, which occurred after the fourth merge in the iterative search.

**Discussion**

When analyzing each merge, we considered the skills or concepts described by the affected skills as well as the test items associated to those skills. The descriptions below discuss the three groups of skills merged in experiment 1 and shown in the Best Model Skill Map (Figure 3b). The two additional pairs of skills merged in experiment 2 are also discussed. In each case, the merges point to commonalities in the skills themselves or among the test items used to assess different skills.

Merge ‘x’ affected skills M-1127 and M-1128. These skills represent “the abilities to represent inequalities from real world contexts” and “explain inequalities from real-world contexts”, respectively. The test items associated with these skills required students to read problems and identify inequality statements that matched the problems. In this case the test items did not distinguish between two unique skills, i.e., representing a problem or explaining a problem, as was suggested by the two skills.

Merges ‘g’ and ‘k’ affected skills M-1118, M-1140 and M-1120. These skills represent the abilities to “locate integers on a number line”, “represent integers on a number line”, and “order integers from least to greatest”, respectively. The test items associated with these skills required students to select lists of correctly ordered integers or identify the correct number line graph of a particular integer. In this case the test items did not adequately distinguish between locating and representing integers on a number line (i.e., M-1118 and M-1120) because all of the items were multiple-choice, and none provided students the opportunity to construct their own number line representations of integers. The inclusion of ordering integers from least to greatest (i.e., M-1140) with the other two skills is possibly due to the fact that using a number line is inherently, cognitively connected to ordering numbers from least to greatest.

Merge ‘t’ affected skills M-1105 and M-1106. These skills represent the abilities to “use positive and negative numbers in real-world contexts” and “relate the meaning of zero to positive and negative numbers in real-world contexts”, respectively. The test items associated with these skills required students to interpret problems involving integers and choose integer answers or verbal statements about integers. Two of the four test items included references to zero either as freezing point or sea level. In this case the items were designed to distinguish between the two skills, i.e., using integers and relating integers to zero. However, the relationship between zero and positive or negative numbers is so critical for understanding integers, that it is likely one cannot compare integers without considering their values in relation to zero.

Merge ‘q’ affected skills M-1120 and M-1108. These skills represent the abilities to “represent integers on a number line” and “recognize opposite numbers on a number line”, respectively. The test items associated with these skills required students to identify the correct number line graph of a particular integer or the opposite of a given integer. In this case, the two skills are inherently connected by the very definition of an integer as the opposite of a whole number. Consequently, it is likely that once students understand the definitions of integers and opposites and can use a number line, the act of graphing an integer is the same as graphing an opposite.

Merge ‘o’ affected skills M-1122 and M-1124. These skills represent the abilities to “recognize integer coordinate pairs” and “graph integer coordinate pairs”, respectively. The test items associated with these skills required students to identify the graph of a given integer ordered pair or to select the description of how to graph a given ordered pair on a coordinate plane. In this case, the items did not clearly distinguish between the two skills because the items associated with recognizing integer coordinate pairs included graphs. Furthermore, the skills themselves are difficult to distinguish in a practical sense because when students learn to graph integer ordered pairs, they routinely associate the numerical representation (i.e., the ordered pair) with its graphical representation (i.e., the point graphed in the coordinate plane).

An additional observation is that some of the skills tended to merge by pairing up with one and only one adjacent skill before RMSE started to decline. Before merge ‘t’, the merges were all pairwise with the exception of merge ‘m’. After merge ‘t’, the skills tended to keep merging into the same skill. The best skill map was generated after merge ‘r’, suggesting that adjacent skills tended to be similar skills and skills M-1140 and M-1120 were similar although they were not adjacent. This was a stronger relationship for several reasons. Firstly, the merges that culminated in the merger of M-1140, M-1118, and M-1120 all took place before the best skill map
was reached. This indicated that those three skills give better predictive performance when represented as one skill. Secondly, this was the first and only 3-skill group to be merged in the best model before RMSE declines. Lastly the three skills took two iterations of the search algorithm to merge together because skills M-1140 and M-1120 were not adjacent skills. Despite the initial graph topology, our search decided to merge these three skills. The combination of all these factors provided strong reasoning that the three skills M-1140, M-1118, and M-1120 were not really distinct skills.

Contributions, Conclusions and Future Work
In this work we provided a search algorithm to reduce the complexity of a given learning map, while improving its fit to real student data. Since merging skills increased accuracy, these results suggest that the original skill map was too fine-grained (given the number of questions per skill and the number of students who took the test.). In some cases the test items did not adequately distinguish between the skills that were merged; hence such skills were merged. The results of algorithms like this can help the content experts who are creating skill maps and test items to either reconsider thinking of two skills as separate, or prompt them to write different test items to better distinguish between students that have mastered one of the skills but not the other skill. In this work, the team that created the learning map expected item 11 was a prerequisite for items 12 and 13, but our stability results suggested that of all the arcs, this arc was the least supported by the data (see Figure 4, arc “g”). In fact, due to this work, we asked an unbiased teacher who did know what our mapping was, to create a hierarchy between items 11, 12 and 13. Surprisingly, she suggested that 12 and 13 were prerequisites to item 11, suggesting that the arc should point in the exact opposite direction. This may indicate that our method may be helpful in using the data to suggest places in the skill graph that need more attention and refinement.

We can relate this work to other work. Heffernan’s ASSISTments project is a project that is attempting to track and improve students’ knowledge across middle school mathematics. About a decade ago we had a learning map with over 300 skills but we now have reduced that complexity to 147 skills. Curriculum designers will correctly be thinking about the subtle ways in which problems are different from one another, which cause them to want to add skills to the skill maps to make more subtle distinctions between questions. However, if you also want to use the hierarchy to track knowledge, having more skills creates complexity, as few questions for each skill make fitting quantitative models harder.

All of the work we have done in this paper has a very small number of questions per skill. This naturally would cause us to think that many merges would be necessary, but if we had a large number of questions, and added all those students’ responses to that large number of questions, we could probably justify more complicated models.

In our experiments we examined the effects of merging skills on an existing learning map. There are many other ways we could have used the existing map to create alternatives. For instance, Cen, Koedinger and Junker (2006) have explored ways of splitting skills or adding new skills, but all of those make more complicated models. What was not examined were the split, and add operations. Possible future work could examine those operations to see if a better model can be obtained with them. Additionally, to validate our algorithm, applying it to synthetic learning maps and synthetic data could be useful to determine if our algorithm does converge to a true learning map.

End Notes
(1) The dataset, evaluation algorithm, and a technical report describing the algorithm in detail can be found at https://sites.google.com/site/assistmentsdata/kansas-project

References


