# Modeling, Visualizing, and Analyzing Student Progress on Learning Maps

By

Dain Vermaak

Submitted to the graduate degree program in Department of Electrical Engineering and Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Committee members

James Miller, Chair

Man Kong

Guanghui Wang

Suzanne Shontz

Bruce Frey

Date defended: _____

The Thesis Committee for Dain Vermaak cerpngies
that this is the approved version of the following thesis :

Modeling, Visualizing, and Analyzing Student Progress on Learning Maps

James Miller, Chair

Date approved: _____

# Abstract

We present a design in which data visualization techniques are applied to meet the needs of teachers and education researchers in analyzing and responding to student learning. Developed using the iterative process fundamental to visualization research and building on established research in the fields of Computer Science and Psychometrics, we present an account of the experimental approaches developed to better understand the project requirements. This will include rationale, observations, and conclusions drawn for each approach. Also presented is the process used to synthesize, from these early efforts, a single visualization tool capable of meeting both predictive and validation requirements as well as the methods used to measure the effectiveness and correctness of the final design.

In addition to visualization schemes and methods, we present contributions made to the field of Computer Science in the form of algorithms developed over the course of the research project in response to gaps in prior art. These include novel approaches to simulation of student response patterns, ranked layout of weighted directed graphs with variable edge widths, and enclosing certain groups of graph nodes in envelopes.

# Contents


**1 Introduction** **1**

1.1 Goals . . . . . 2

1.2 Terminology . . . . . 3

**2 Prior Art** **5**

**3 Simulation** **11**

**4 Early Development** **14**

4.1 Augmented Learning Map . . . . . 15

4.2 Icon Graph . . . . . 16

4.3 Focus on Edges . . . . . 20

4.4 Node and Edge Scores . . . . . 21

4.5 Edge Centric Design . . . . . 25

4.6 Tower Design . . . . . 27

4.7 Validation Graph . . . . . 31

**5 Final Design Synthesis** **35**

5.1 Envelopes . . . . . 38

5.2 Icons . . . . . 39

5.3 Variable Width Edges . . . . . 40

**6 Layout Algorithm** **41**

6.1 Steps . . . . . 43

6.2 Compute all Paths . . . . . 44

6.3 Determine Node Tiers . . . 44
6.4 (Optional) Account for Bypass Edges . . . 45
6.5 Calculate Node Placement Order . . . 49
6.6 Place Nodes . . . 50
6.7 Apply Physics System . . . 53
6.8 (Optional) Bypass Gates . . . 53
6.9 Arcs . . . 56
6.10 Collision Detection and Correction . . . 58
6.11 (Optional) Custom Changes . . . 59

**7 Envelopes** **60**
7.1 Steps . . . 64
7.2 Define the Grid . . . 65
7.3 Apply Seed . . . 66
7.4 Restricted Flood . . . 67
7.5 Build Edges . . . 68
7.6 Edge Operations . . . 72

**8 Contours** **76**

**9 Final Design Phased Evaluation and Refinement Process** **79**
9.1 Types of Students . . . 80
9.2 Interview Tool . . . 81
9.2.1 Phase 1 . . . 82
9.2.2 Phase 2 . . . 90
9.2.3 Phase 3 . . . 92
9.3 Data Evaluation . . . 95
9.3.1 Research Goal 1 - Progress Assessment and Prediction . . . 98
9.3.2 Research Goal 2 - Validation . . . 101

9.4 Evaluation of Design Elements . . . . . 102

**10 Conclusions** **105**

**11 Future Work** **107**

**A Participant Data** **112**

**B Phase 1 Data** **113**

**C Phase 2 Data** **114**

**D Phase 3 Data** **115**

**E Classroom Progress Assessment** **116**

**F Interview Transcript Summaries** **117**

F.1 Participant A Summary . . . . . 117

F.2 Participant B Summary . . . . . 118

F.3 Participant C Summary . . . . . 119

F.4 Participant D Summary . . . . . 120

F.5 Participant E Summary . . . . . 122

F.6 Participant F Summary . . . . . 122

F.7 Participant G Summary . . . . . 123

F.8 Participant H Summary . . . . . 124

F.9 Participant I Summary . . . . . 126

**G Interview Outline** **128**

# List of Figures


1.1 Unit Map . . . . . 2

1.2 Mapping Test to Learning Map . . . . . 3

2.1 Sankey Diagram *Google Charts. Accessed 25 Aug. 2018.* . . . . . 8

2.2 Color Maps *Color Brewer 2.0. Accessed 14 Oct. 2018.* . . . . . 9

3.1 Unique Paths . . . . . 12

4.1 Augmented Learning Map . . . . . 17

4.2 Student Icons . . . . . 19

4.3 Local Percent . . . . . 21

4.4 Edge bands . . . . . 22

4.5 Node and Edge Scores for Students A,B,C,D (Cutoff:70) . . . . . 24

4.6 Aggregate Node and Edge Scores (Cutoff:70) . . . . . 25

4.7 Edge Centric Design . . . . . 26

4.8 Tower Design . . . . . 28

4.9 Tower Design All Layers . . . . . 28

4.10 Tower Design Layer Breakdown (top to bottom: layer 0, layer 1, layer 2, layer 3) . 29

4.11 Tower Design Using Only Original Edges . . . . . 31

4.12 Validation Graph . . . . . 32

4.13 Validate Graph Highlighting Flaws . . . . . 34

5.1 Design Progression . . . . . 36

6.1 Motivation for Layout Algorithm . . . . . 41

6.2 Layout Example using Layered Algorithm . . . . . 43
6.3 Paths Through Graph . . . . . 44
6.4 Tier State . . . . . 46
6.5 Paths Scores . . . . . 49
6.6 Build Grid For Nodes A and C . . . . . 50
6.7 Build Grid For Nodes F and G . . . . . 52
6.8 Example after Initial Node Placement . . . . . 52
6.9 Example After Physics System . . . . . 54
6.10 Position of Gates . . . . . 54
6.11 Arrow/Gate Interaction . . . . . 55
6.12 Arrow . . . . . 57
6.13 Final Result . . . . . 59

7.1 Spring Approach . . . . . 61
7.2 Walking Approach for Connected Envelope . . . . . 63
7.3 Walking Approach for Connected Envelope . . . . . 63
7.4 Walking Approach for Unconnected Envelope . . . . . 64
7.5 Example Configuration . . . . . 64
7.6 Example Grid Space . . . . . 65
7.7 Steps in the Flood Process . . . . . 68
7.8 Computing Concave Hulls . . . . . 69
7.9 Pixel Corner . . . . . 70
7.10 Envelope Operation Examples . . . . . 73
7.11 All Operations . . . . . 75

9.1 Test Students . . . . . 81
9.2 Phase 1 - Standard Graph Features . . . . . 83
9.3 Phase 1 - Flow Graph Features . . . . . 86

9.4 Phase 1 Fred Results . . . . . 89
9.5 Phase 2 Features . . . . . 90
9.6 Phase 3 Features . . . . . 93
9.7 Contour Types . . . . . 94
9.8 Teacher Choices . . . . . 97
9.9 Classroom Assessment . . . . . 100

# List of Tables

A.1 Participant Backgrounds . . . . . 112

A.2 Validation Inference . . . . . 112

B.1 Phase 1 Choices . . . . . 113

B.2 Phase 1 Choices . . . . . 113

B.3 Phase 1 Choices . . . . . 113

C.1 Phase 2 Choices . . . . . 114

D.1 Phase 3 Choices . . . . . 115

E.1 Classroom Assessment . . . . . 116

# Chapter 1

# Introduction

A learning map is an unweighted directed graph containing relationships between discrete skills and concepts with edges defining the prerequisite hierarchy. Learning maps arose to connect student instruction directly to standards and curriculum and are designed to assist teachers in lesson planning and evaluating student response. In particular, learning maps provide a means by which teachers can diagnose the strengths and weakness of individual students and quickly tailor lessons accordingly.

The Enhanced Learning Maps Project designed learning maps for the Math and ELA subjects. These comprehensive master maps included material from kindergarten through grade twelve and at the time of writing contained 4220 nodes and 11323 edges across both subjects. The large Subject Maps were broken down into smaller, more manageable subsets called Unit Maps (Fig. 1.1). Instead of wading through thousands of nodes, teachers were able to select a Unit Map specifically aligned to a topic. Teaching resources had been developed for many Unit Maps, relying on the structure of the map to guide teachers through a set of student exercises. These resources often provided ways to evaluate and correct common student misconceptions. By understanding their student's mistakes, a teacher could measure their progress and indicate which nodes on the unit map a student or class understood. The question of how to model and provide an intuitive visualization of that understanding is the inspiration for this research.

A first step in providing a visualization of student's knowledge was to evaluate their understanding of a set of nodes. Short assessments composed of 10-15 multiple-choice questions were developed to gain insight. Each answer choice on the assessment was linked to one or more nodes within the learning map. Fig. 1.2 shows an example of a multiple choice item taken directly from

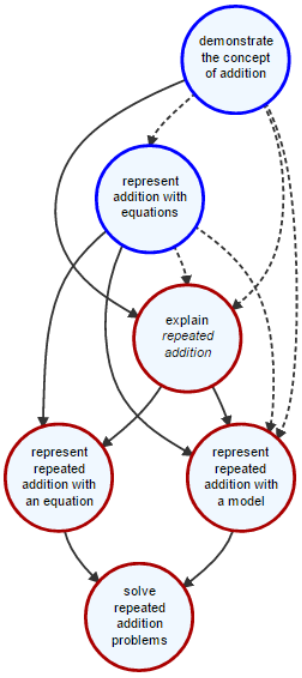


Figure 1.1: Unit Map

an Enhanced Learning Maps test. Red and blue arrows indicate a mapping between incorrect and correct answers respectively. The results of these assessments were then processed using data mining and psychometric techniques to determine the degree to which students understood each node in the corresponding Unit Map.

## 1.1 Goals

The goal of this research project was to develop visualization capabilities to meet the needs of two distinct audiences:

- **Teachers:** (Research Goal 1) Allow rapid assessment of student or class progress with respect to a given learning goal (the Target Node of a Unit Map as defined in section 1.2) and provide insight into what should be taught next to further their understanding.
- **Researchers:** (Research Goal 2) Allow for validation of the underlying learning maps by providing indications of where student learning patterns differ from those predicted by the maps.

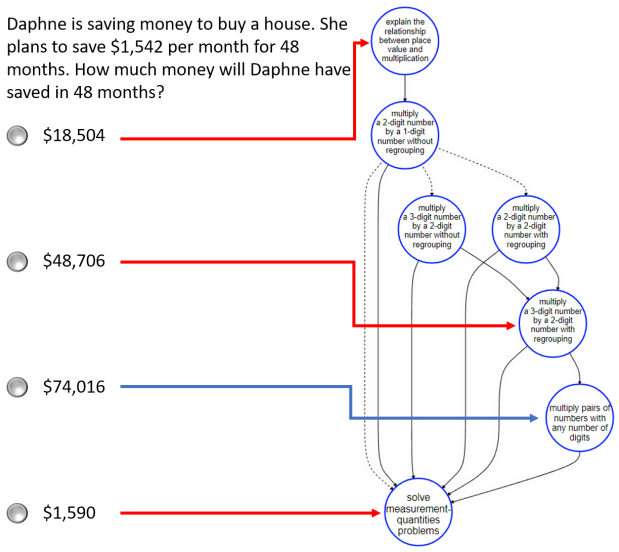


Figure 1.2: Mapping Test to Learning Map

While initially unsure whether a single interactive visualization capable of meeting the requirements of both groups could be designed, we did discover a scheme that worked well for both audiences.

## 1.2 Terminology

Given the interdisciplinary nature of the project, it is important to clearly define some terminology:

**Destination Node**: (of an edge) The node to which an edge points.

**Learning Map**: A directed acyclic graph defining the relationships between discrete skills and concepts.

**Mastery**: (of a node) The degree to which a student understands a node. Mastery is measured in this paper as a numerical value between zero and one hundred and can also be considered the likelihood (or percent chance) that a student has mastery of a node and will answer questions associated with the node correctly.

**Root Node**: A node in a unit map without a parent node. Root nodes appear above their children in a unit map.

**Source Node**: (of an edge) The node from which an edge emerges.

**Subject Map**: The underlying master graph for a subject. Subject Maps were developed by content experts for the Enhanced Learning Maps project for Math and ELA subjects, covering concepts between kindergarten and grade twelve.

**Target Node**: A node in a Unit Map without a child node. Target nodes appear below their parents in a Unit Map.

**Test Map**: A subset of a Subject Map containing nodes referenced by a given test.

**Unit Map**: A subset of a Subject Map containing nodes related to a particular unit or teaching goal.

Note: A node's type is context specific and based on the map being considered. A given node may be a destination node, a source node, both or neither depending on which additional nodes are shown. A node may also be a target node of one Unit Map and a root node of another.

# Chapter 2

# Prior Art

Analyzing relevant prior art must include both the use of graphs and similar visualizations to help teachers and education researchers as outlined in chapter 1 as well as known specific techniques for selecting and rendering subsets of large graphs (e.g., the Subject Maps of our application), including augmenting them with various types of auxiliary information. Searches for prior art of the first type revealed very little. Before the inception of the Enhanced Learning Maps project, learning maps appear solely as a data construct driving assessment suites such as Dynamic Learning Maps (DLM). One focus of the Enhanced Learning Maps project was determining whether putting the learning maps in the hands of classroom teachers would improve their ability to teach effectively, making them pioneers in the field of visualizing student understanding (Kingston & Broaddus (2017)). Given the relative infancy of learning maps as a wide-spread organizing structure for learning, no relevant prior art was found which contributed directly to either of the two research goals addressed in this paper. Some prior art indirectly related, such as work done by Broaddus et al. (2015) in map validation, involves efforts aimed at refining the learning map structure from an educational context but does not present new methods for visualization.

Graphs, graph theory, and graph visualizations, on the other hand, do have a rich literature. It was immediately clear that we needed fast and effective algorithms for automatic layout and rendering of small and medium graphs as well as the ability to augment these basic renderings in several ways. The remainder of this chapter focuses on the prior art in this area and exposes several significant holes corresponding to unique aspects of our application that required the development of several sophisticated new techniques relating to graph rendering.

Classic topics in effective graph drawing and interpretation generally begin with a survey of

quality metrics. Studies into objective measures of usability have, over time, resulted in a set of guidelines for improving their readability. A commonly referenced example of a study in this area was done by Purchase et al. (1997) who were able to confirm the negative effects of edge crossings and edge bends on the understandability of graph drawings. Additional metrics include those relating to consistency of edge lengths and slopes as well as more general rules such as those relating to the overall graph area (Eades & Tamassia (1988)). Certain metrics have more or less impact in different contexts and a number of different layout algorithms have been developed which provide optimizations to achieve some metrics at the expense of others. In their paper, von Landesberger et al. (2011) provide an extensive overview of the state of the art for visual analysis of large graphs, providing an introduction to all major approaches and techniques. Among these is the layered approach used by the Enhanced Learning Maps project. The library used by the project is called *dagre* and is built on well-known and accepted research (Gansner et al. (1993),Brandes & Köpf (2001),Jünger & Mutzel (1997), Barth et al. (2002)). While numerous improvements have been made to the original version developed by Sugiyama et al. (1981), the layered approach maintains adherence to the same basic process of first layering nodes, reducing edge crossings, and finally placing nodes. The layered approach tends to perform extremely well, even for very large graphs, and their design is very well suited to directed graphs.

Another common approach, known as force-directed, uses a physical system to balance node and edge placement. The original work done by Eades (1984) used a spring-electron approach and assigned physical forces modeled as springs or charged particles to nodes based on their connections within the graph. The system of forces is then released and allowed to settle to a state of equilibrium before being rendered. Force-directed approaches are known for clean, symmetrical graphs but tend to have a high run-time cost associated with the physics system used to drive them. In many cases an alternate layout approach is used before application of a force-directed algorithm. This often mitigates much of the cost, especially for large graphs.

The typical node-edge representation of graphs is not the only way to view hierarchical data. One large family of visualization approaches involve matrix-based layouts (Burch et al. (2013)),

which attempt to solve edge problems by removing edges entirely. Reading a graph without its edges may seem difficult but a study by Ghoniem et al. (2004) which focused on undirected graphs concluded that, while in general most tasks are more easily performed on node-link diagrams for smaller graphs, matrix-based layouts perform better for large, dense graphs. While promising, further work by Didimo et al. (2014) narrowed the focus to directed graphs and showed a number of visualization approaches to node-link diagrams that outperformed matrix-based diagrams in all areas. Despite this, matrix-based approaches offer a unique perspective and clear advantages in many graphical applications. Using a strong layout method does not by itself guarantee a useful graph but it often provides a framework on top of which can be presented much more information than was previously possible.

In addition to fast, high quality, layout algorithms, there exist many techniques for augmenting graphs to extend the amount or type of information displayed. Of particular interest to this project were weighted directed graphs featuring variable edge widths. The body of literature relating to graphs of variable edge width was found to be extremely lacking and prompted investigation into closely related fields to find practices that would translate to node-link diagrams. Sankey Diagrams (Fig. 2.1) represent an approach closely related to variable edge width directed graphs and many concepts are applicable to both visualizations. Sankey Diagrams are known for providing an effective means of communicating flow data between entities over time. Originally used in the context of energy or material flow, they are an old concept which seems to be gaining popularity in areas such as web traffic visualization. A key property of Sankey Diagrams is conservation of data, meaning that no information within the diagram is created or destroyed. This property, therefore, guarantees that all data can be traced to and from its final destination.

Fortunately, many of the same heuristic methods that already exist for managing layered graphs have, over time, been adapted for Sankey diagrams. Zarate et al. (2018) provides an excellent summary of recent approaches but points out that recent works have focused on heuristic methods without consideration for the edge width. They go on to present a layout approach using an ILP model which considers the edge widths of the diagram to provide an optimal layout in terms of

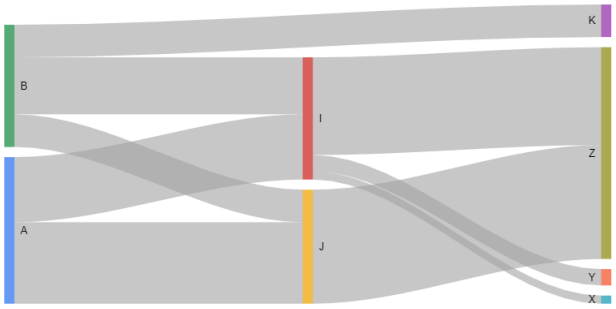


Figure 2.1: Sankey Diagram *Google Charts. Accessed 25 Aug. 2018.*

crossing area reduction. Other sources also contribute by drawing attention to features which are unique to this class of graph. One example of this is the problem noted by Riehmann et al. (2005) who observed that Sankey diagrams tend to suffer from the vertical-horizontal illusion (Mamassian & de Montalembert (2010)). This is the phenomenon whereby humans viewing vertical and horizontal lines of the same length always perceive the vertical line as longer.

In addition to augmenting the graph through edge width manipulation, our visualizations needed to incorporate various forms of auxiliary information outside the field of graph theory. To ensure effective display of the data we also relied heavily on foundational literature in the field of Visualization. Chief among these is the concept of visual variables (Bertin (2011)). There are believed to be eight ways to encode information in graphical components. Though occasionally seen with different names, they are commonly referred to as brightness, color, orientation, position, shape, size, texture and motion. While effective alone, visual variables are often more so when combined. For example, a square icon used to represent a bird could be combined with color to indicate the type of bird, simultaneously encoding multiple pieces of information. Testing has shown that certain variables are well suited to viewing different types of information and so are also better suited to performing certain tasks. When selecting similar objects from a set, for example, evidence suggests that for this task humans are most readily able to group objects by size, brightness, texture, orientation and sometimes color. In contrast, when trying to *order* those objects based on some sort of comparative measure, humans are best able to use texture, size and brightness but struggle to use orientation or color. The most effective visualizations are those in which the visual vari-

ables are chosen to naturally draw the viewer to complete a prescribed task which then seamlessly communicates the intended information.

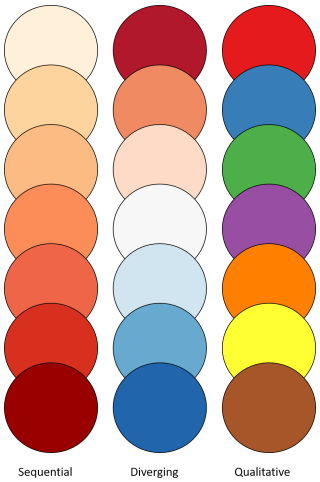


Figure 2.2: Color Maps *Color Brewer 2.0. Accessed 14 Oct. 2018.*

Conversely, the poor use of visual variables can easily make a task harder. Color is a variable that is often misused. Fortunately, the results of significant studies published by several groups provide insight into how effective color maps can be designed for common tasks. Color maps define a linear range of numeric values by assigning each to a separate color. They rely on the user's ability to idenpngy and decode the color. While people sometimes struggle with this task, and numerous examples exist of modern visualizations sabotaged by color (Borland & Ii (2007)), the proper use of color is a powerful tool made even more effective by the work done by those such as Harrower & Brewer (2011) in compiling and disseminating effective color maps. In their work they break color maps into the three categories shown in fig. 2.2. While any type of color map can be used for any data set, there are many types of data which are better shown with certain color maps. Sequential color maps can be used to present ordered data. Though possible to display continuous numerical data, they typically use discrete data partitioned into a number of buckets, each assigned to a fixed color. While this mapping has been effectively used for ordering, it is has not been found useful when performing strict quantitative visual comparisons such as determining whether one value is twice that of another. Furthermore, with the exception of diverging color maps, multi-hued color maps have been found to be more confusing than helpful for ordering

purposes. Diverging color maps draw attention to a break or focus point in the data. They are always multi-hued, values appearing above the breakpoint are assigned one hue (e.g. red) while those below are assigned another (e.g. blue). Within these groupings the data points tend to form a sequential color sub-map, becoming darker the further they are from the break point. While both diverging and sequential color maps are used to convey ordered data, the final type, Qualitative, is best used for categorical data which is not sortable. Qualitative color maps employ different hues with common lightness and saturation levels. While originally designed for use in cartographic applications, color maps provide a structure of differentiation useful among a wide array of fields.

# Chapter 3

# Simulation

The nature of our project was such that real student data was produced very slowly as teachers signed on to participate. Even were that not the case, the amount of data required to fully explore the visualizations would have been years away from being accumulated. With so little data, development of the visualization tools became extremely difficult. In the absence of real data we turned to techniques frequently used by our Education collaborators and simulated student response data. The design of a simulation process being tangential to our research focus, we needed a way to provide student response patterns feasible within the context of a particular learning map yet not necessarily representative of real world data. Functionally, the simulation process needed to construct a measure of mastery for a given student relative to a particular node. In other words: a mapping function of the form $Score(Student, Node)$.

Naive attempts to simply apply a uniform distribution to the mastery of nodes for a student revealed additional requirements:

- (Simulation requirement 1) Consistent mastery. Students should tend to follow paths of understanding and should only rarely skip large sections.
- (Simulation requirement 2) Increased difficulty for topics lower in the map.

To address the problem of consistency, we loosely applied a psychometric technique in which we defined a student's performance as a culmination of their ability to perform specific tasks. For example, to answer a math story problem would require that a student be able to read and understand English, interpret the English into a mathematical domain, and then apply the correct operations to solve the problem. Each of these skills is represented by a unique $\theta$. In this context

$\theta$ describes the probability that the student will answer a question relating to the skill correctly. Different ways may also exist to solve a math story problem, leading to cases in which different combinations of $\theta$s can be used equally well to solve the same problem. In the simulation below, each $\theta$ was assumed to be described by a connected path. Fig. 3.1 below shows an example graph and the three unique paths from the single root node (node V) to the single target node (node Z). Each path was assumed to be one dimension of a student's innate ability to learn. Since the same student may learn some things more easily in certain ways, a student may have more ability in one path than another. A student with a high ability to learn concepts in path 2 would more easily understand node X while a student with a high ability to learn concepts in path 1 would more readily understand node W than node X.

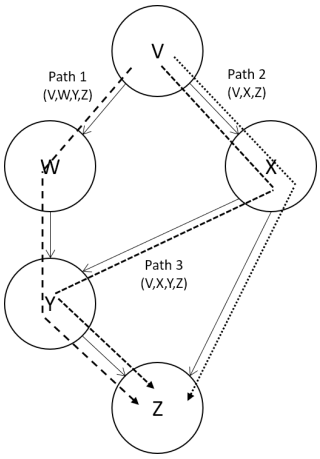


Figure 3.1: Unique Paths

Having defined a student as the combination of their innate ability, random numbers were generated for each ability using a uniform distribution after which the ability scores were normalized to a range between fifty and one hundred. Having defined each student's abilities, the next step was to use those abilities to determine whether a particular student is capable of mastering a given node in the learning map.

Algorithm 1 details the process used to derive node scores from the individual path scores representing a student's ability. At each node the student's path scores are computed individually. When all path scores have been computed the two highest are averaged to determine the final node

score for the student.

**Algorithm 1** Simulated node score for a given student

1: **procedure** SIMULATEDNODESCORE(Graph $G$, Student $s$, Node $n$)
2: let $level(n_i, p_j) = 1+$ the number of edges between node $n_i$ and the root of $p_j$
3: **if** $n \notin G.nodes$ **then**
4: **return** 0
5: **end if**
6: let $P$ be the set of all paths in $G$
7: let $score(p_i)$ be the score of the student $s$ for the path $p_i$
8: $P_n \leftarrow \{p | p \in P, n \in p\}$ ▷ All paths containing n
9: $Q \leftarrow \emptyset$
10: **for all** $p \in P_n$ **do**
11: **for all** $m \in p$ **do**
12: let $l$ be the node appearing in $p$ immediately before $n$
13: **if** $n = m$ **then**
14: $q \leftarrow score(p) * (1 - \frac{(0.1 * (level(l,p)))}{|p|-1})$
15: $Q \leftarrow Q \cup \{q\}$
16: **end if**
17: **end for**
18: **end for**
19: $q_0 \leftarrow max(Q)$ ▷ Get largest score
20: $q_1 \leftarrow max(Q - \{x_0\})$ ▷ Get second largest score
21: return $\frac{(q_0+q_1)}{2}$
22: **end procedure**

The final simulation step involves addressing the second requirement. In a learning map the difficulty of the concepts increases the farther down the student progresses. Therefore, a student with a path score of $q$ should have lower scores the further down the path they travel. Linear interpolation was used to simulate an increase in difficulty for concepts farther down the graph such that the first node on a path would receive a score of $q$ while the last node on a path would receive a score of $0.9q$ (Algorithm 1 Line 14).

# Chapter 4

# Early Development

At this stage, it is important to clarify a few details regarding the rendering and interpretation of maps in the Enhanced Learning Maps project.

The first are dashed edges which appear in many Unit and Test Maps. They are used to reduce map complexity by indicating the existence of pathways which are present in the underlying Subject Map but not included in the visible node subset. The number of nodes represented by one dashed edge can range anywhere from a single node to the contents an entire Subject Map. As applied to our research goals, dashed edges generally represented untested nodes for which we had no evidence of mastery. For simplicity we chose to ignore them entirely, leaving the possibility of extrapolating information onto them for future work.

Another defining attribute of the learning map was the concept of multiple pathways used to model the fact that a person can learn the same thing in different ways. While conceptually sound, this concept introduced extreme complexity into the system. Two polarizing logical relationships could be brought to bear on the link between a node and its parents. The AND relationship implied that every parent of a node had to be mastered in order to learn the node itself. In contrast, the OR relationship suggested that it was sufficient for any one of the parents to be mastered. Between these extremes lay every imaginable combination, in which certain sets of parents were sufficient under particular conditions. Rather than enforcing a logical paradigm, the meaning of each edge was left to the discretion of the Subject Map's author. This worked surprisingly well as the primary users of the map were teachers who generally had sufficient background knowledge of the map's subject matter to understand the author's intent. Naturally, designing a recommendation system for such a structure took what was a simple logic diagram and moved it into the domain of statistical

analysis. Combined, these two facets of the learning map made it an interesting problem requiring a unique approach.

Crucial to the success of any interdisciplinary research effort is the ability of the computer scientist to see the problem through the eyes of the other researchers who are opening themselves to the possibilities enabled by exploiting fundamental principles of visualization technology. In the following section we document a series of visualization approaches developed in the early stages of the project and which eventually formed the basis of the final design. Documentation will include rationale for each experimental approach and will attempt to explain how our understanding of both the visualization and education aspects evolved in response. Throughout this process we drew on the expertise of staff members from the Enhanced Learning Maps project who would critique each visualization from the perspectives of both target audiences: teachers and education researchers. All Enhanced Learning Maps staff members had backgrounds in both classroom teaching and educational research, making them uniquely suited to consider both angles. In addition to Enhanced Learning Maps staff, we were able to regularly gain feedback directly from classroom teachers who attended yearly conferences for the learning map software. This feedback, while informal and entirely subjective, provided a lot of insight into the initial reactions of educators to the developing models. The combination of these teachers and the Enhanced Learning Maps staff evaluators are hereafter referred to as "reviewers".

Finally, bear in mind that the goal underlying all these designs was to understand how best to move students from the Root Node of a Unit Map to its Target Node. The fact that we seemingly ignored some nodes by suggesting that one path or another should be taken does not imply that the skipped nodes were any less important. Rather it suggests that progress from Root Node to Target Node in the map of interest was likely not furthered by focusing on the skipped nodes.

## 4.1 Augmented Learning Map

To better understand the problem requirements, the first visualization augmented a basic learning map (Fig. 4.1). Colors were used to indicate the likelihood of mastery beginning from a dark green

indicating a likelihood of at least 90% to a dark red implying a likelihood of at most 10%. Around the edge of the main central map are six smaller maps, each representing an individual student. The central map depicts an aggregate view of the six students in which the mastery of each node is computed by taking the average of the six student masteries. Gray nodes are those for which there was not enough data to make an estimate of the likelihood and are ignored when computing the aggregate. The use of a diverging color map was a natural choice as we wanted to clearly separate nodes mastered by most students from nodes mastered by few students.

Observations revealed that the main benefit to this initial visualization was consistency. Having already been exposed to the structure of learning maps, users required little additional instruction in its use. Users found grouping nodes by whether they were mastered to be natural given the break in color but had difficulty distinguishing the difference in color between similar shades, particularly around the data set's breakpoint and when comparing non-adjacent nodes. This difficulty prevented grouping within a color, often requiring each node to be individually examined, quickly becoming tedious on larger graphs. A final problem related to the color scale was the tendency for reviewers to interpret it as an indicator for the degree of mastery rather than the likelihood. While this distinction may seem trivial, it could be argued that the discrete nature of the nodes precludes partial understanding, making this implication a misrepresentation of the fundamental structure. Another downside was scalability. The visualization quickly became overwhelming as more students were added. Student graphs lost all meaning as they became too small to make out details. Additionally, while the aggregate proved a useful tool for placing a set of students on the map, it lacked any means of idenpngying the next recommended step for teachers and so failed to fulfill either of the research goals.

## 4.2 Icon Graph

In response to insights gained into requirements from the previous iteration we turned our attention to the use of visual variables (Bertin (2011)). By overlaying a learning map with colored icons as shown in fig 4.2 we increased the amount of information present while maintaining high visual

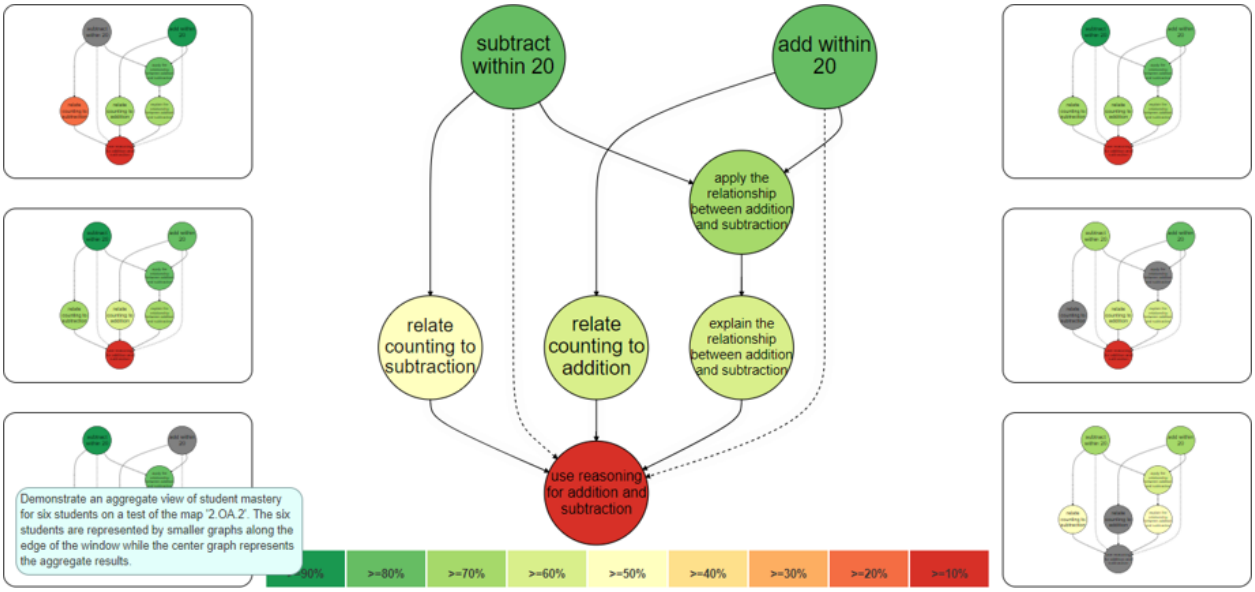


Figure 4.1: Augmented Learning Map

fidelity. Each student was represented by an icon and so needed to exhibit multiple properties. These included the student's mastery level of the nodes, personal demographic information, arbitrary skill, and the source of data (e.i. their classroom). Mastery was already indicated by the position of the student icon within the graph and the student's arbitrary skill level was discretized before being included as a demographic trait. To further simplify the problem, we assumed that only one demographic trait needed to be displayed, the viewer being able to specify the target trait as needed. The remaining attributes (classroom and single demographic trait) could be then defined using the following visual variables:

- Icon position within a node.
- Icon color.
- Icon shape.
- Icon texture.
- Icon brightness.
- Icon size.

- Icon orientation.
- Icon motion.

Before proceeding, we presented the concept to reviewers to ascertain the manner in which a teacher might use the graphic. Reviewers indicated that the first action taken by a teacher would likely be to survey the graph, noting the relative distribution of icons across the nodes. Having done that they would then isolate their own classroom and search for patterns. Demographic information may lastly be used to evaluate the behavior of certain individuals. This progression from all to classroom to demographic trait drove the choice of visual encoding. For the first operation teachers would need to quickly group icons into classrooms. Size, position, texture, brightness, color, and orientation tend to excel in selective operations and were the obvious candidates. Of these, size, texture, and brightness also tend to encourage ordering and were discounted to avoid the implication of relative importance between classrooms. Of the attributes remaining (color, orientation, and position) color was chosen to represent the classroom of the student. Shape was then selected to represent the demographic trait as, being highly associative, different data values would not draw attention away from the grouping task. To further encourage classification and pull slightly more attention to the differences between the icon shapes, icons from the same classroom were grouped together within each node.

In the resulting visualization shown in Fig. 4.2 the icon color denotes the student's classroom while its shape indicates a particular demographic attribute. Student icons are present on a node if their mastery is equal to or greater than a chosen cutoff (in this case 70%). Students are present on edges only if they are also present on both nodes connected by the edge.

The icon driven graph proved superior to the initial iteration in many ways. It was more scalable, able to encode multiple discrete attributes for many students without loss of information or visual overload. Additionally, it was able to display derived edge traffic making simple next step prediction possible, and so addressing the first research goal. For example: a teacher trying to move a group of students to understand node F with the knowledge that they had just grasped the concepts at node B may choose to skip node D and instead teach only the concepts at node F.

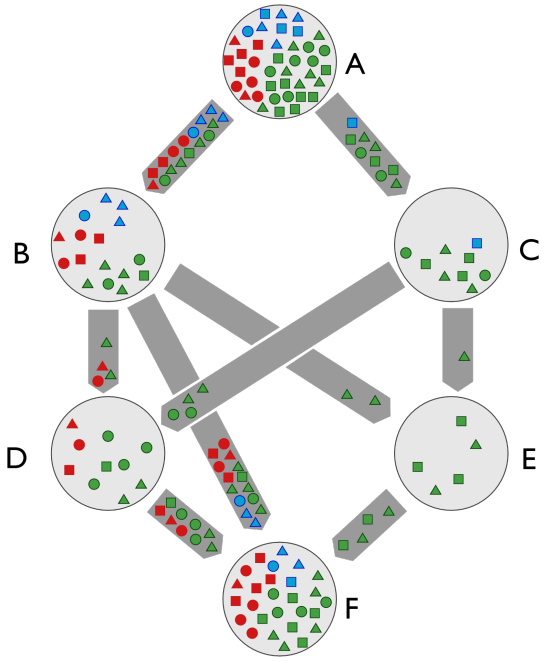


Figure 4.2: Student Icons

Such a course could easily be plotted from the graph's indication that most students who master the concepts at node B tend to bypass node D, and instead move directly to node F.

While better, the icon graph also had its weaknesses. Firstly, there was a clear limitation in scalability due to the icon size and detail being inversely proportional to the icon count. Grouping icons into denominations would have alleviated the problem at the cost of clarity since most grouping strategies involve shape or color manipulation and so would begin to obscure the information encoded by those variables. A more pressing concern was the inability of the visualization to show student transitions across edges not present in the original graph. A student could, for example, show significant mastery of node A and node F without ever showing mastery of nodes B, C, D, or E. These types of events speak to the second research goal and serve as a validation tool, indicating a weakness in either the underlying learning map, the questions used to test mastery, the curriculum, or the visualization algorithm. The act of bypassing a set of nodes may also represent the recommended path, a critical piece of information.

During sessions with reviewers it became clear that they were easily able to understand the placement of students but, while they tended to prioritize edge operations, they struggled to inter-

pret and compare edge information. In an attempt to clarify this data, edge icons were removed and the edges themselves separated into distinct bands.

In Fig. 4.3 the total width of the edge is proportional to the ratio of the number of students on the edge's source node to the number of students on the edge. Each edge is then subdivided based on the relative number of students from each class traveling along the edge. For example, the total width of the edge between nodes A and C in fig. 4.3 is determined by dividing the forty students on node A by the nine students that traveled along the edge. The width of the green band is then computed as eighty-nine percent of the total width given that eight of the nine total travelers belong in the green classroom.

This approach proved useful when comparing paths from a single node. For example, of the three paths leaving node B it is clear that the majority of students travel to node F while fewer travel to node D and almost none travel to node E. However, the visualization is misleading when edges from different nodes are compared. The edge between nodes A and C has nine travelers but is half the width of the edge between nodes C and D which has only four. The localized nature of the graph proved unintuitive to researchers, making it difficult to quickly assess the best path of travel through the graph.

## 4.3 Focus on Edges

While the edge-band approach provided unique information, a key concern was the difficulty in easily recognizing the difference between edge band widths. Further discussion revealed that using the band length rather than the band width provided a much more usable tool for comparison. Fig. 4.4 shows five edge representations for the edge between nodes A and B in Fig. 4.3. The left-most edge serves as an unscaled reference while edge (I) is the same edge as shown in Fig. 4.3. Edge (II) defines the total width using the source node in the same manner as edge (I) but uses the band length to show the same information as was previously encoded in the band width.

Edges (III) and (IV) forgo use of the total edge width as an encoding and instead provide another approach. Edge (III) displays each band width as equal and then each uses a darker inner

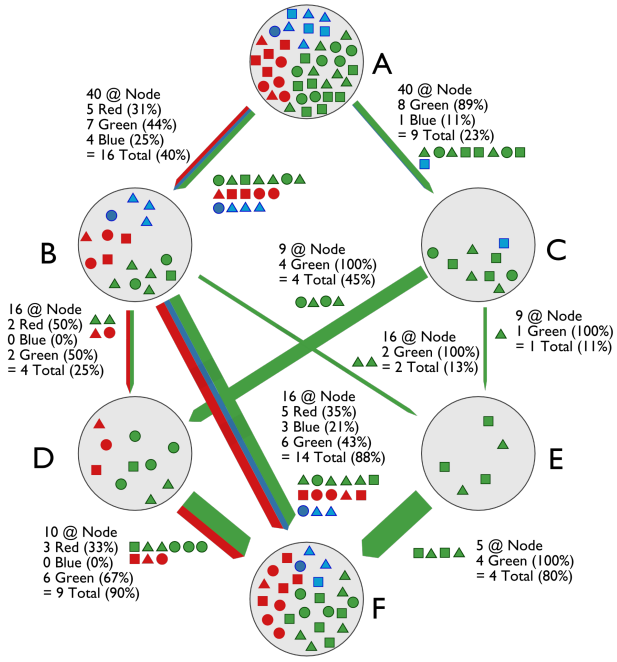


Figure 4.3: Local Percent

band to show the ratios within the lighter bounded area. Edge (IV) shows the same information as edge (II) minus that given by the total width. Reviewers ranked the edges based on how quickly they were able to evaluate the relative differences between colors and found edge (II) the most useful followed by edges (IV) and (III), with edge (I) the least usable.

## 4.4 Node and Edge Scores

During development of designs following the icon graph in section 4.2 it became necessary to determine the edges traversed by a particular student. The relevant data associated with each student was the mastery obtained for each node represented as a number between zero and one hundred. For simplicity a cutoff score was applied such that a score above the cutoff was considered evidence of mastery while one below was considered evidence of misunderstanding. Additionally, for the remainder of this chapter an edge is considered mastered by a student if the student has obtained mastery of both the source and destination nodes.

Determining the score for a specific node relative to a student can be conceptualized by imag-

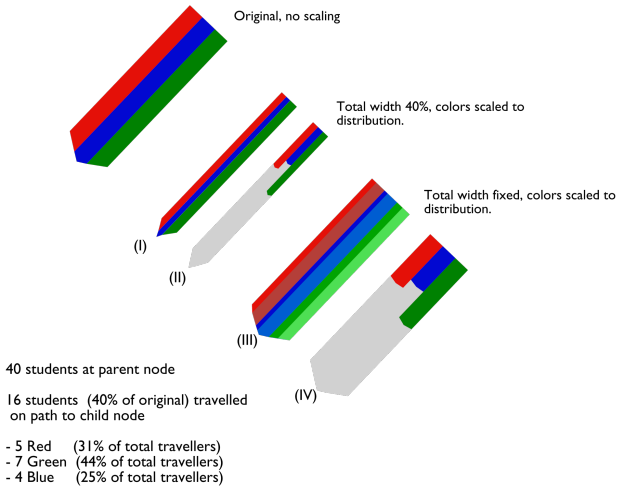


Figure 4.4: Edge bands

ining an icon of the student placed at the root node of the graph. The icon is copied to form a set of smaller icons, one for each edge leaving the root node for which the student has gained mastery. Each icon is of equal size and together their area is equal to that of the original icon. One icon is placed on each child of the root node mastered by the student, after which the process repeats. As the icon moves down the graph it continues to divide, becoming smaller with each step. Algorithms 2 and 3 detail the recursive process used.

**Algorithm 2** Student node score

**procedure** NODESCORE(Graph $G$, Student $s$, Node $n$, Cutoff $\alpha$)
  let $s(n_i)$ be the score obtained by student $s$ at node $n_i$
  **if** $n$ is the root node **then**
    **return** 1.0
  **else if** $s(n) < \alpha$ **then**
    **return** 0.0
  **else**
    $E_{in} \leftarrow \{e | e \in G.edges, e.destination = n\}$ ▷ Edges entering n
    **return** $\sum_{t \in E_{in}}$ EDGESCORE$(G, s, t, \alpha)$
  **end if**
**end procedure**

Learning maps for four students, each in a separate panel, are shown in Fig. 4.5. Nodes mastered by the student are shaded gray. Within each node is given the student's level of mastery for that node, the name of the node, and the node score obtained by the student. For instance: Student

**Algorithm 3** Student edge score

**procedure** EDGESCORE(Graph $G$, Student $s$, Node $n_{from}$, Node $n_{to}$, Cutoff $\alpha$)
  let $s(n_i)$ be the score obtained by student $s$ at node $n_i$
  **if** $s(n_{from}) < \alpha$ **then**
    **return** 0.0
  **else if** $s(n_{to}) < \alpha$ **then**
    **return** 0.0
  **end if**
  $E_{out} \leftarrow \{e | e \in G.edges, e.source = n_{from}\}$ ▷ Edges leaving 'from' node
  $E \leftarrow \{e | e \in E_{out}, s(e.source) >= \alpha, s(e.destination) >= \alpha\}$
  $ns \leftarrow$NODESCORE$(G, s, n_{from})$
  **return** $\frac{ns}{|E|}$
**end procedure**

1's graph shows that they have achieved 90% mastery over node V.

To better understand the algorithms, consider student 3. To find the node score for node Z relative to student 3 begin at node V for which student 3 received a node score of 1. Since student 3 has mastered both node W and node X, node V's score is divided by two and applied to each path. This trickle-down effect continues until node Y at which point the input edge scores are recombined and their sum is defined as the node score of Y.

Our goal in this was to derive the number of students that travel each edge. The final step in producing an aggregate graph of student travel was to sum the node and edge scores for each student. The result of this operation is shown in Fig. 4.6. A key difference between the aggregate graph and the individual student graphs is the inclusion of a gray edge from node V to node Z. Consideration of the following points is helpful in understanding why that edge is necessary:

- All four students have mastered node V.
- Three of four students have mastered node Z.
- Summing the edge scores entering node Z we see that $0.75 + 1.25 = 2$.

Since we know that three students mastered node Z but can only see two students entering the node, we infer that a student entered node Z on a path not shown in the original graph. This can be confirmed by examining the graph for student 4 in fig. 4.5. Student 4 was somehow able to master

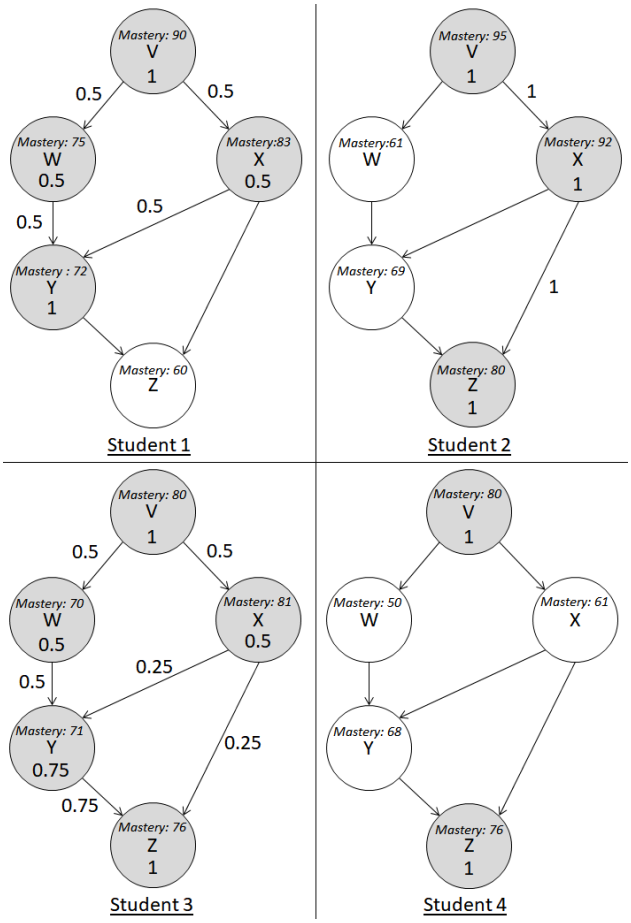


Figure 4.5: Node and Edge Scores for Students A,B,C,D (Cutoff:70)

node V and node Z without mastering nodes W, X, or Y. Rather than ignore the path of student 4, we add it to the map as a gray edge to indicate passage along an unknown path. Doing so ensures a conservation of information within the graph and more completely indicates the path of students. Gray lines represent the start of a framework needed to begin addressing the second research goal. A map which perfectly modeled the behavior of students would contain no gray lines. The presence of gray lines indicates that the underlying structure of the map may be compromised.

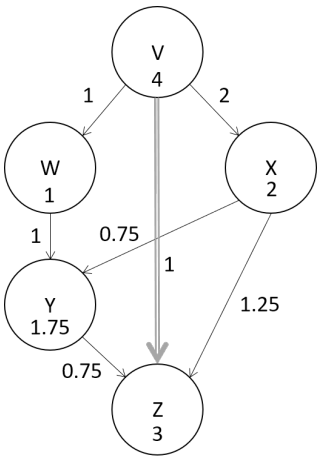


Figure 4.6: Aggregate Node and Edge Scores (Cutoff:70)

## 4.5 Edge Centric Design

Working with the edges in Section 4.4 demonstrated that, under certain conditions, most of the aggregate information about a node could be derived through its edge data. Up to this point graph edges were only shown if they existed in the original Subject Map. This policy failed to account for the situation in which a student, given three sequentially connected nodes *A*,*B*, and *C* somehow mastered *A* and *C* without mastering *B*. One requirement for transferring node information to the edges was to ensure representation of all data, including edges apparently not represented in the original Subject Map. Fig. 4.7 depicts the same graph as used in Fig. 4.2 and was generated in the following manner:

1. Nodes were grouped into tiers. A node's tier was defined as the minimum number of edges

between itself and the last node in the graph. This was the only impact the original graph structure had on the visualization.

2. A "gate" was defined between each tier and edges were drawn between each node and the entrance of the exit gate to the tier. The thickness of each edge represented a percentage of the total students that traveled from the node further into the graph.

3. Edges were drawn to each node from the exit of the entrance gate to the tier. The thickness of each edge represented a percentage of the total students that traveled to the node from elsewhere in the graph.

4. Edges were drawn between gates to represent students that bypassed all the nodes in a given tier. For example: a student may have significant likelihood of mastery of nodes A and D with very low likelihood of mastery of B or C.

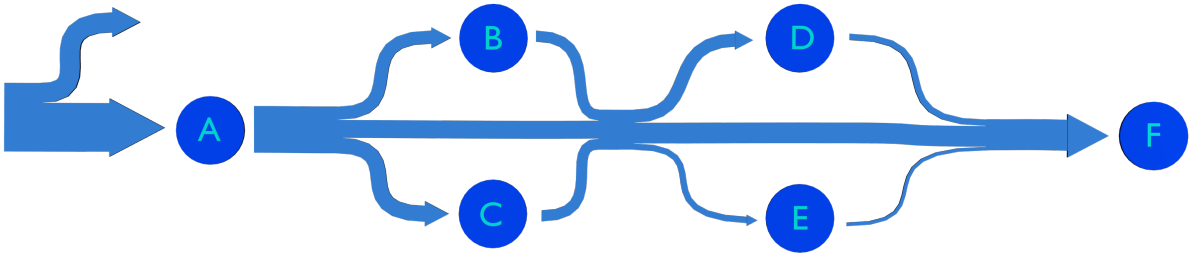


Figure 4.7: Edge Centric Design

The resulting visualization provided several insights. To begin with it guaranteed that all travel between nodes was clearly displayed, allowing a shift of information from nodes to edges. It also proved intuitive, reviewers requiring little explanation before being able to use it. Reviewers felt confident that they could determine whether the original learning map was flawed based on the relative number of students that bypassed prerequisite nodes. The tiered approach to the edge centric design also removed edge crossings entirely, drawing attention to key relationships between nodes. However, the benefits of the sleek edge centric design came with a few significant costs.

Chief among these costs were a lack of graph structure, limited connective data and a restriction in the number of source and target nodes. As the original Unit Map served only to categorize nodes into tiers, there was an overall lack of cohesion when trying to interpret results of the edge centric map against the Unit Map. Connections such as existed between node B and node F were not represented in the visualization, the students on that edge instead represented by the bypass between the second and third gates. The missing edges made it difficult to confidently plan future lessons. This design also suffered when used with maps containing multiple root or target nodes. Multiple target nodes in particular caused problems as nodes could easily belong to many nonadjacent tiers.

Overall, the edge centric design represented progress. While it failed to provide a system of recommendation to meet the first research goal, it did hint at attributes necessary to perform validation of the underlying map as required by the second.

## 4.6 Tower Design

Encouraged by the promise shown in the edge centric design, a more complex graph was used to further investigate the properties that made validation appear possible in the previous iteration. Fig. 4.8 therefore, shows the visualization of a much more complicated Test Map, including eighteen nodes and nineteen edges. The visualization was constructed using the same process as that used for the edge centric design with one minor difference. Whereas in the edge centric design the edges between tiers pass through a common gate, the tower design featured direct connections between nodes in adjacent tiers regardless of whether the nodes were connected in the original learning map. The result was a graph with edges that, given that the source node was mastered, described the probability that the destination node was also mastered. This method preserved some of the data lost to the central gates of the edge centric design.

The immediate problem with this approach was the high number of edge crossings, particularly in the area shown by Fig. 4.9. A vertical "layered" scheme was used to break apart the edge crossings. In trying to solve this problem, consideration was given to what edge information was most important to teachers. It was determined that the distribution of child nodes was the most

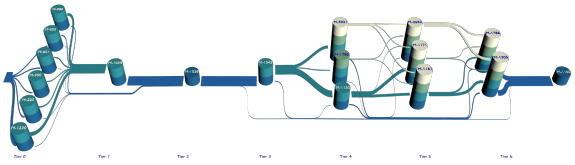

Figure 4.8: Tower Design

crucial aspect since it would indicate what to teach next. Each layer, therefore, contains a node and its children. To minimize the number of layers, a node may exist in multiple layers but no two nodes in the same tier can exist as a source node in the same layer. In fig. 4.9 node M-5021 is a source node in layer three with child nodes M-2650, M-1171, and M-1161 and is also a child to the source node M-1045 in layer one. This "layered" approach ensured that no edges could cross within the same layer, indirectly providing a means of isolating nodes for examination. To view node M-5021, for example, a user could ignore all but layer three which is shown in the last frame of fig. 4.10. The result would eliminate the web of edges and show only children of M-5021 along with other nodes in that layer.

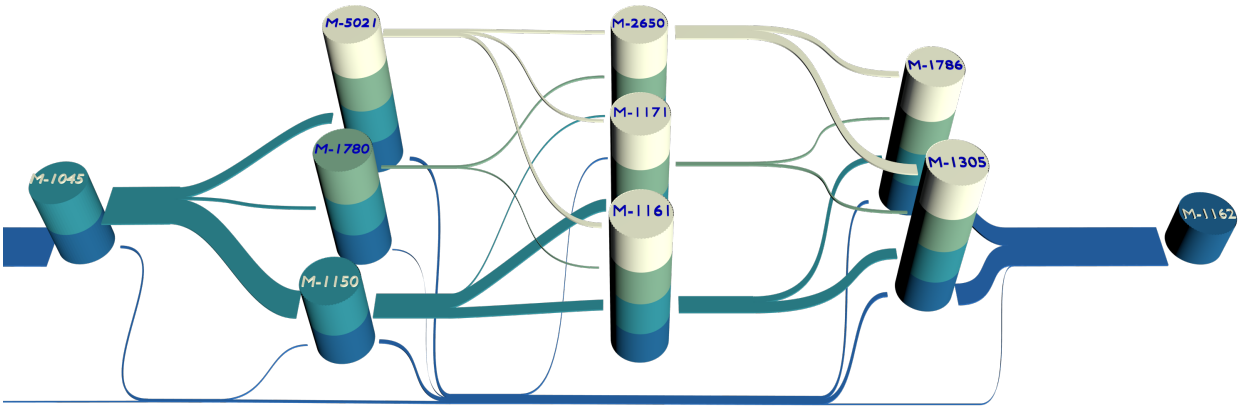


Figure 4.9: Tower Design All Layers

Response from reviewers was mixed. On the one hand, the connections directly between tiers provided a convenient means of determining the next concept to teach. On the other hand, the addition of a third dimension was difficult for reviewers to use without spending time acclimatizing.

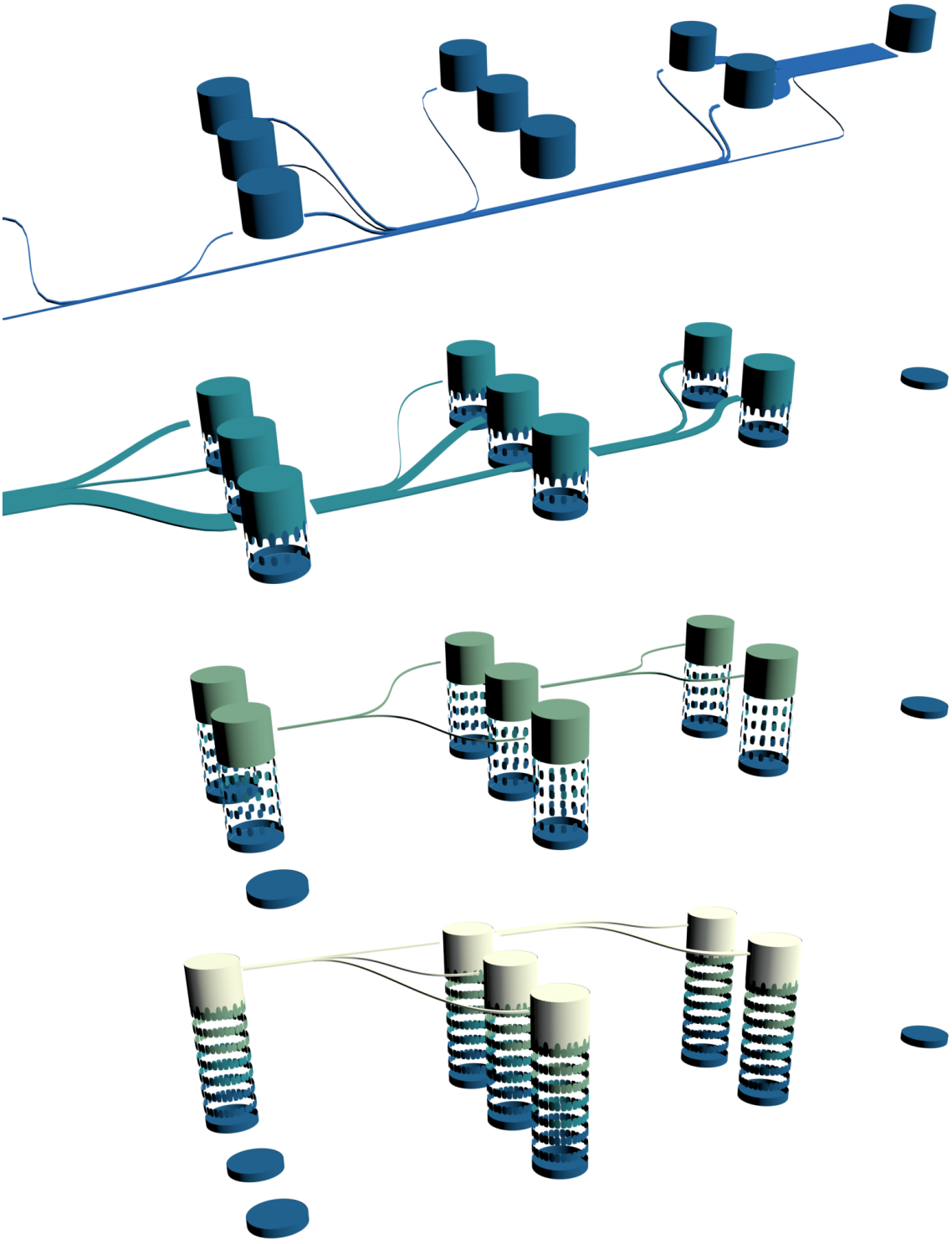

Figure 4.10: Tower Design Layer Breakdown (top to bottom: layer 0, layer 1, layer 2, layer 3)

The difficulty was made more so by areas densely packed with interwoven edges. This could be mitigated by isolating specific layers but doing so often removed other contextual information that the user would have preferred to keep visible. Despite the difficulty in getting at the information, once they did reviewers were able to interpret the meaning of connections and easily recognize ways in which it could be used for both validation and planning future lessons.

Having been developed to investigate validation using correlation, conclusions drawn using the tower design were difficult to map onto the original learning map. A user may determine the best route from a node only to find it missing from the original map. A variant of the tower design shown in fig 4.11 represents one of the earliest attempts to merge validation back into the original learning map. In fig 4.11 the same graph is used as was used in fig 4.8 with four unconnected nodes removed from the left side to reduce clutter. Separation of nodes into tiers and calculation of the bypass edges shown running across the green plane follow the same process used in both precursor versions and continue to represent pathways taken between non-adjacent tiers for which no path exists in the original graph. What makes the variant different is that it includes no edges except those present in the original graph. Fortunately, the tiered version of the graph contained a single edge crossing, making it very clean in comparison to other graphs of similar size.

While still able to isolate layers in the graph, users found doing so unnecessary given the much lower number of edge crossing obscuring the graph. Additionally, reviewers found that when simply viewing the graph from directly above they were easily able to separate bypass edges from normal edges. The major observation from this design was that reviewers were able to accept the difference in edges. Though curious about the way the white bypass edges had been determined, they were willing to accept the explanation and then apply them, making simple observations on the degree to which the data contradicted the map. In a similar manner, the way in which edge data was obscured when passed through gates elicited questions, the answers to which were accepted without apparent reservation. The lack of negative response to this version in some cases may well have been its relative simplicity to the much more complex version shown above. Whatever the case, reviewers were able to perform all expected operations involving high level validation and

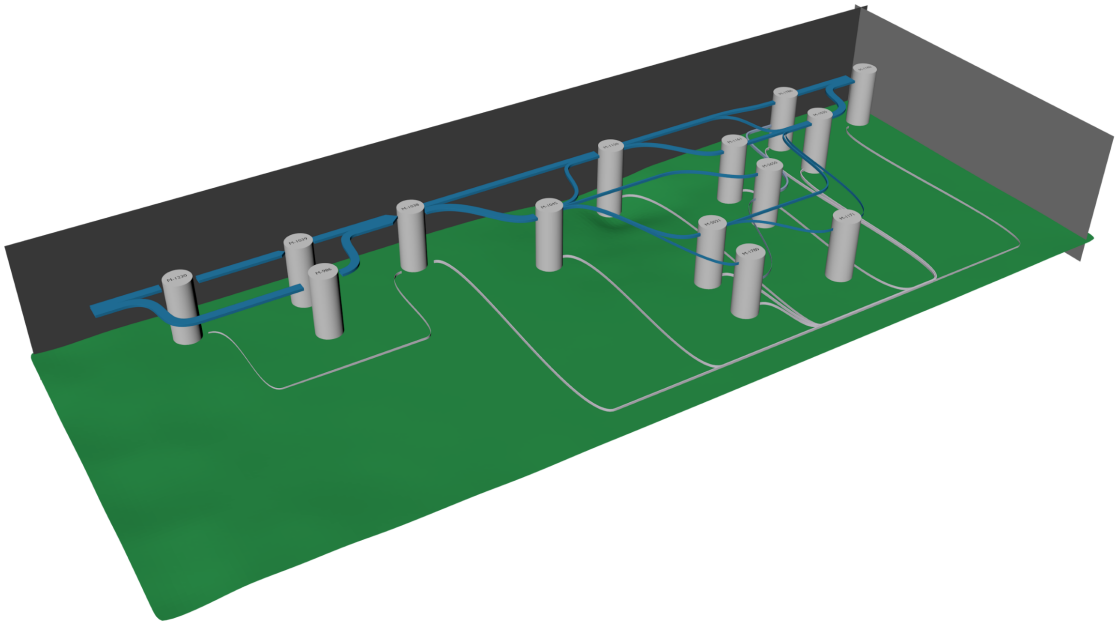

Figure 4.11: Tower Design Using Only Original Edges

choosing next concepts for individual cases.

## 4.7 Validation Graph

Observations made in the latter stages of the tower design prompted a shift from three dimensional graphing back to flat graph displays. Building on the idea of using correlation for bypass edges while letting normal edges adhere to the original graph, we developed a visualization focused entirely on addressing the second research goal of validation. The end result was the graph shown in fig 4.12 which combined concepts found in both the tower and edge-centric designs. Orange edges represent bypass edges, generated using a method similar to that described later in chapter 6. As in the tower design, these bypass edges represented connections that did not exist in the original learning map. Conversely, the blue edges represent those that were present in the original map. Edge width was determined after all edges were placed and used methods presented in section 4.4 such that thicker edges were representative of higher traffic. Edge color in the validation graph

was used as an additional indication of traffic, with darker blue hues indicating areas of higher traffic. The intent was to more clearly differentiate the edges in areas such as that around M-471 in which many edges merge as they arrive at the node.

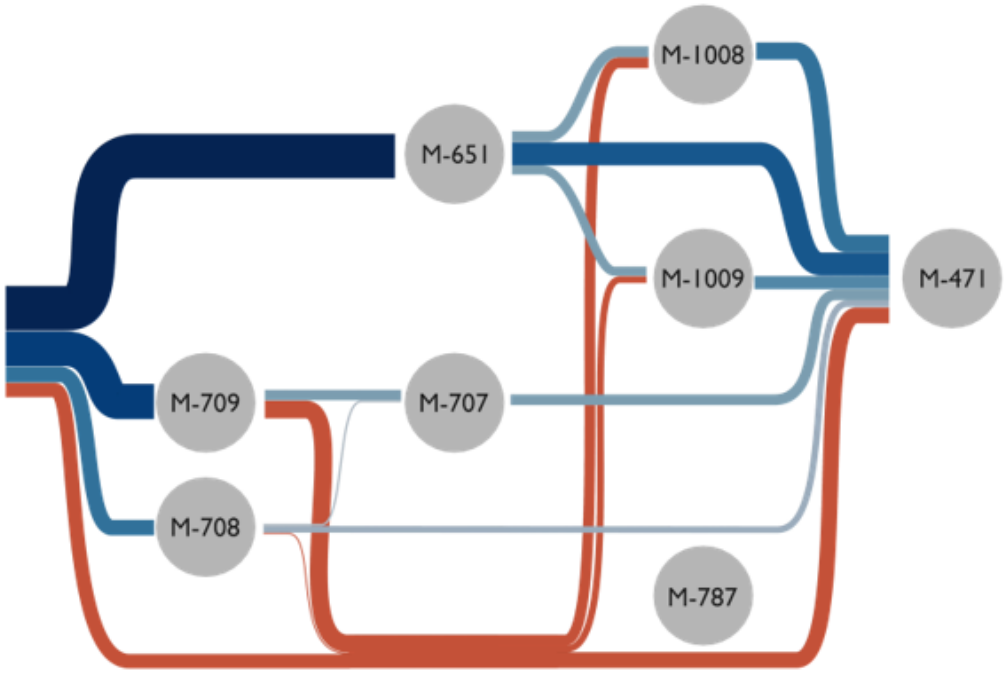


Figure 4.12: Validation Graph

In addition to our normal reviewers, the design was also presented to the governance board of the Enhanced Learning Maps project. The members of the board were considered experts in their respective fields and were at the forefront of educational research. While the graph did initially include icons to indicate student progress, to capitalize on the research experience of the board we tailored the design such that attention was instead drawn to the existence of anomalies. After showing the map, we asked the board members present to answer the following questions:

1. What can be inferred about the original learning map from the flow map?
2. Do you feel that given the flow map it would be possible to validate the original learning map? If so, which nodes or connections would you consider reviewing?

Reviewers were able to grasp the general use of the design immediately. They found using the edge width as a representation of the probability very natural, allowing for a clear system of recommendation. They were also able to extend the concept to consider conditional probability. For example, they recognized that when given a student who had mastered M-471 it was possible to estimate from the edges entering the node the probability that they had also mastered M-651. The only notable concern raised by reviewers was the potential for teachers outside the research field to be overwhelmed by the high complexity of the flow graph. A number of suggestions were made on how to reduce the complexity without loss of data, many of which were later used to inform the final design.

Results pertaining directly to validation, however, were less clear. While reviewers were easily able to accept the concept of the bypass edges, they did not immediately recognize their role in validation. We found that the questions listed above initially resulted in confusion, reviewers unable to understand the context of the questions. Switching the direction and presenting the questions from the point of view of everyday teaching was what finally communicated the potential for validation. For example, when reviewers were asked what they would teach a student who had mastered M-709 they would first answer with the only connected node: M-707. After a moment's hesitation their follow-up question would be: "But why is the orange line out of M-709 thicker?". It was as though a switch had been flipped. Once reviewers arrived at the point where they asked that question, the potential for validation seemed to become clear. While the different background of reviewers present meant that there were a number of opinions regarding the reliability of the bypass edges, they were easily able to argue and compare the various merits and causes of orange edge widths relative to blue. Fig 4.13 details the possible flaws indicated by the graph and observed by the group. Each flaw is marked with a yellow box. In most cases the presence of a flaw could be deduced from the high width of an orange edge relative to blue, similar to the previously given example involving M-709.

The key point taken from this was that validation seems to require a different mindset from that required in everyday teaching. Even when presented to a room full of researchers, many of

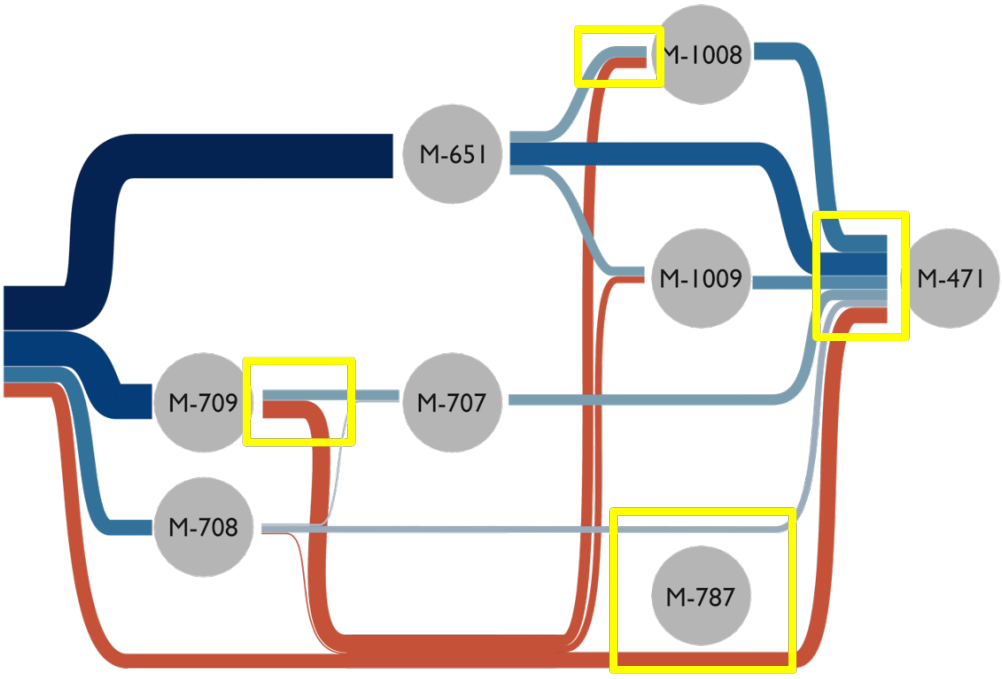


Figure 4.13: Validate Graph Highlighting Flaws

whom had extensive prior experience with the maps, most were unable to immediately recognize the potential for validation. When finally, through exposure or focused instruction, reviewers were able to recognize the role of validation then the graph became an effective tool for detecting the presence of anomalies. This ultimately raised questions of whether a better way existed to more naturally encourage validation or, alternately, whether the act of suspending trust in the original map may be unnatural in the absence of context.

# Chapter 5

# Final Design Synthesis

Fig 5.1 shows the path taken through the various early designs to arrive at a final visual solution able to fulfill the needs of both target audiences. Evaluation by reviewers revealed that the Icon Graph (Section 4.2) and, to a lesser extent, the Augmented Learning Map (Section 4.1) provided the most information in terms of analyzing the current state of students in a classroom. Using these tools teachers were able to assess the performance of a classroom or single student and quickly isolate the information needed to begin making decisions regarding future lesson planning. While clearly meeting the requirements of the first research goal, education researchers were unable to use these early visualizations for map validation. In contrast, the last approaches presented: The Tower (Section 4.6), Edge-Centric (Section 4.5), and Validation Graph (Section 4.7) designs were well suited for the map validation described in the second research goal, yet provided limited information for classroom use. Using the feedback gathered from these existing designs we isolated the key features and synthesized a visualization tool we felt would meet the requirements of both goals.

Designs developed prior to this point represent effort focused on using a single graphic to simultaneously view multiple pieces of information. Using an iterative process of presentation, feedback gathering and redesign we were able to narrow the initial list of requirements, idenpngy the effectiveness of different visualization combinations, and determine which approaches were best suited to understanding each piece. Moving forward we shifted our focus to design a tool rather than a single image. In much the same way that a city map gives context to different types of information such as roads and districts, our goal was to provide a foundational visualization capable of providing context to any applicable data set within the research focus.

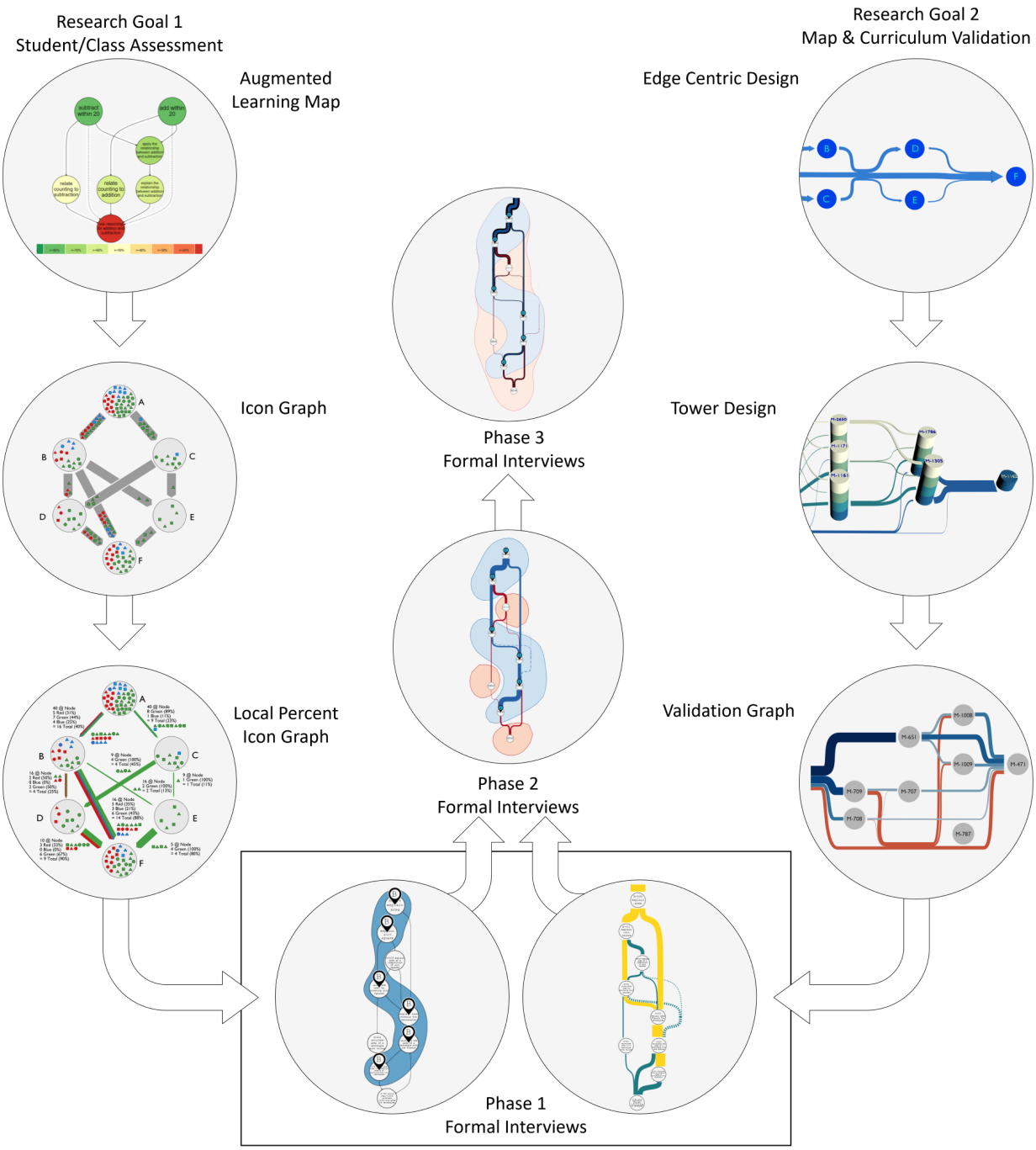


Figure 5.1: Design Progression

We began by revisiting the original two research goals (Section 1.1), using knowledge gained through reviewer interactions to isolate the set of essential operations associated with each goal.

The first research goal required that the visualization "**Allow rapid assessment of student or class progress with respect to a given learning goal and provide insight into what should be taught next to further their understanding.**" When separated by task, it was ultimately implemented as three operations such that any visualization tool capable of fulfilling the research goals:

1. Must be able to evaluate mastery of a single student and, by extension, be able to idenpngy areas in which further teaching is required.

2. Must be able to evaluate mastery of a classroom and, by extension, be able to idenpngy areas in which further teaching is required.

3. Must be able to use the visualization to inform decisions regarding future teaching. In other words: when given the current level of mastery attributed to a student or classroom the visualization should provide a indication of typical learning patterns ranked by frequency.

The second research goal required that the visualization "**Allow for validation of the underlying learning maps by providing indications of where student learning patterns differed from those predicted by the maps.**". The single task we redefined in terms of the reasons that student learning patterns might differ from those predicted by the maps. In that sense, any visualization tool capable of fulfilling the research goals:

4. Must be able to detect problems in alignment between test, curriculum and learning map. This detection would ideally include some suggestions as to the cause and recommend strategies for realignment.

Having reduced the granularity of the research goals to a set of operations, we began examining the features of previous designs, selecting those that were well suited to performing the required operations.

## 5.1 Envelopes

Evaluation of understanding for either a single student or a classroom necessitated rapid, instinctive node grouping. As a starting point we revisited earlier work done using the augmented learning map in section 4.1. Using the augmented learning map, reviewers found grouping nodes into mastered and un-mastered sets to be simple given the differentiation in color above and below the breakpoint. Unfortunately they found discerning individual shades within each color challenging, particularly when comparing shades between non-adjacent nodes. They also found that coloring the nodes often obscured the text. Of greater concern was the susceptibility of the color scale to misinterpretation. While variation in color were intended to indicate the confidence of the data, many users repeatedly interpreted darker green shades to indicate a higher level of mastery and darker red colors to imply a deeper level of misunderstanding. As previously explained in section 4.1, this represented a possible misrepresentation of the fundamental principles regarding learning maps.

Recognizing valuable features within the design, we began to address some of the problems, starting by applying a certainty threshold such that mastery of a node was ignored unless the confidence level of the data exceeded the threshold. The addition of a threshold reduced the gradient scale to a binary one: mastered or not. Mastered nodes with a confidence level above the threshold were shown as green while those below were shown as red. Users would then be able to set what they considered an acceptable level of confidence, observing the effect as changes in which nodes were mastered. By making the act of adjusting the threshold a separate task, we hoped to prevent typical users from misinterpreting the gradient while still permitting advanced users to have control of the confidence level.

Another change was to move the indication of mastery off the node and onto an enclosing envelope. Our goal was to prevent the color from obscuring the text and also to provide a superior way for users to maintain a mental grouping of the nodes while examining edge information. Envelopes were expected to provide a clear indication of holes in individual student learning and so enable the first essential operation. They would also contribute in some part to the third operation

which involved choosing the next topic for a student by eliminating those already mastered.

Notably, the decision was initially made to exclude envelopes from the flow graph visualization. While this decision was later repealed, reviewers felt that the combination of the envelopes and the variable width edges would prove too overwhelming to participants who, prior to this, had been exposed to nothing more complicated than the comparatively simple learning maps. Exclusion of the envelopes from the flow graph ultimately led to the phase 1 interview tool having its functionality split between the two graphs.

## 5.2 Icons

In addition to the augmented learning map, we also borrowed inspiration from the icon graphs presented in section 4.2. Unsurprisingly, icon graphs had proven to be ineffective when used to isolate and evaluate individual students. Instead, they brought insight into classroom evaluation and comparison. Hoping to simplify the design, we initially focused on the demographic information unique to the icon graph. While the concept of demographic grouping had been well received by reviewers, encoding the information within the shape of the icon had intentionally limited utility and prevented demographic grouping from interfering with that being done for classrooms. The question raised in response to that decision was whether viewing classroom and demographic information simultaneously was useful. The only obvious reason to do so was to enable comparison of classroom performance with respect to certain specific demographics. Subsequent discussion with reviewers indicated that, as well as being a rarely needed operation, the same effect could be achieved more clearly by adding a means of filtering students shown on the graph based on specific demographic classifications. Rather than attempting to decipher the difference between icon shapes, a user would see only those students within the demographic of interest, entirely removing the need for an additional visual variable.

Having already idenpngied more effective ways of using graph edges, the only remaining features of interest in the original icon graph were the number of icons per node and the classroom distribution. Not being among the set of essential operations, we chose to temporarily ignore class

comparison. Having not yet determined an alternate way of indicating the number of students, we replaced the multiple icons present on each node with single map icons. A number within the map icon indicated the quantity of students able to claim mastery of the associated node. Alone, these map markers were not expected to contribute significant understanding on distribution of mastery, but it was hoped that they would provide a means of quanpngying the information encoded by the envelopes.

## 5.3 Variable Width Edges

Variable width edges had, by this point, become a primary visual tool. The question, therefore, wasn't if they would be included in the final design, but how. The bypass edges used in the validation graph (Section 4.7) had proven the most effective way to perform the fourth essential operation relating to validation, while the probabilistic nature of the edge width had enormous potential for choosing the next steps in student learning. The same probabilities when viewed as the baseline performance for a classroom suggested that variable width edges may also be well suited to comparing student performance.

Naturally, the choice of baseline would impact the effectiveness of this feature. Brief consideration was given to making the edge width dependent on the number of students currently shown on the map as was done in variations of the icon graph. While doing so would have provided a clear indication of the path taken by the current students for individual comparison, preliminary work on the envelopes already hinted at more effective ways of communicating this information. Although locking the edge width would still have had value when used to compare entire classrooms, it was decided in the final design to allow users to select the students used to compute the edge width. Doing so gave users a means of comparing their students to others in the same classroom, other classrooms, or larger external sets. Additionally, by using the students of one set to determine the edge width and then applying a different method to overlay the nodes mastered by a second set, users would be able to compare groups of students.

# Chapter 6

# Layout Algorithm

Often, the biggest impediment to obtaining usable graph drawings is the confusion caused by edge intersection. As it is frequently impossible to completely remove all edge crossing from a graph, the primary goal of most graph layout algorithms is to meet an objective set of quality metrics. These metrics range in type from reducing edge crossings and corners to minimizing graph area. Many layout algorithms are designed to optimize certain of these metrics at the expense of others. Fig 6.1 depicts a learning map drawn using a typical graph layering algorithm (Sugiyama et al. (1981)). A single edge connecting nodes M-1289 and M-2738 has been enlarged to demonstrate one of the major difficulties faced when attempting to adapt variable edge widths to traditional graph layout approaches.

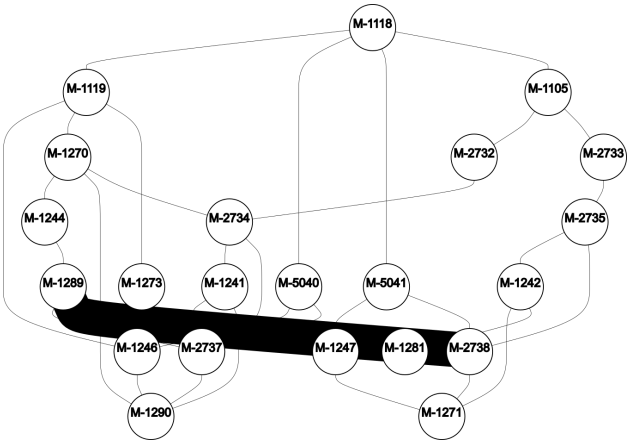


Figure 6.1: Motivation for Layout Algorithm

Effectively managing edge width meant that, along with commonly accepted metrics, we needed a layout algorithm that met the following additional requirements:

1. Performance should be on par with existing algorithms and therefore capable of real-time

rendering for small and medium sized graphs.

2. When possible, the algorithm should account for, and indicate, the relative importance between pathways.

3. Priority should be given to reducing crossings between edges of higher importance. These edges are generally thicker and, while reducing their edge intersections helps ensure clear visibility of important information, it also prevents them from completely obscuring thinner edges as shown above.

4. A layered vertical approach similar to that used in normal learning maps should be used to indicate a clear progression of learning with no edge able to connect nodes within a layer.

5. A primary use for the resulting graph will be determining the next topic from a given node. The layout algorithm should, therefore, prioritize clarity of edges leaving a node over those entering.

6. To further mitigate the problems caused by crossings, edges should have a consistent shape to assist in predicting where it will emerge.

In the absence of an existing approach matching the requirements above, and realizing from work done by Zarate et al. (2018) that the possibility of finding applications in parallel domains was unlikely, we used a combination of existing techniques to design one. The resulting algorithm used a combination of layering techniques (Sugiyama et al. (1981)) to prioritize high traffic paths followed by a modified forced-directed (Hu (2006)) approach to space out nodes within their layers. To more clearly explain the following concepts, the graph shown in fig 6.2 has been rendered using a normal layering algorithm and has been augmented by the addition of small diamonds on each edge to indicate weight. This graph will be used to demonstrate the different steps in the remainder of this chapter.

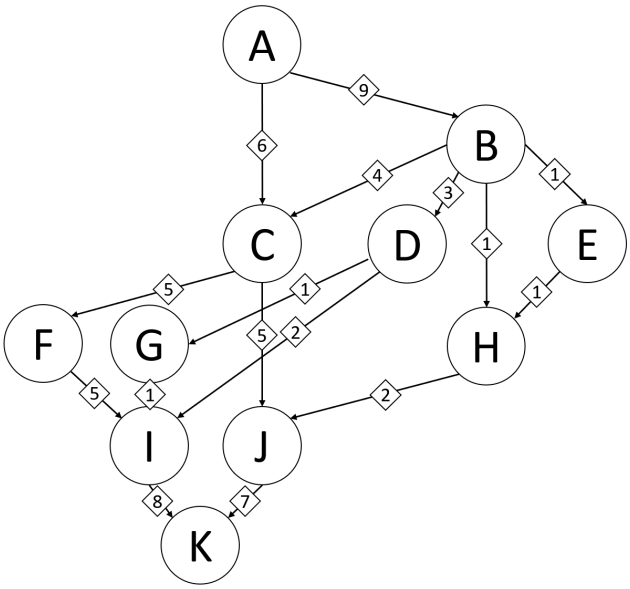


Figure 6.2: Layout Example using Layered Algorithm

## 6.1 Steps

We present this algorithm as a general layout solution for weighted directed graphs of variable edge width. Steps which are unique to the learning map context are marked as optional and can be ignored for the general case. Additionally, any input graph must be acyclic.

Section 6.2 - Compute all Paths

Section 6.3 - Determine Node Tiers

Section 6.4 **(Optional)** Account for Bypass Edges

Section 6.5 Calculate Node Placement Order

Section 6.6 - Place Nodes

Section 6.7 - Apply Physics System

Section 6.8 - **(Optional)** Bypass Gates

Section 6.9 - Arcs

Section 6.10 - Collision Detection and Correction

Section 6.11 - (Optional) Custom Changes

## 6.2 Compute all Paths

A first step is the calculation of every path through the graph from every root node to every target node using a modified breadth first search. Fig 6.3 shows the resulting path set which will be used multiple times in subsequent steps. Each column in the table represents a path between root and target nodes. Each row marks the tier in which a node falls relative to the path.

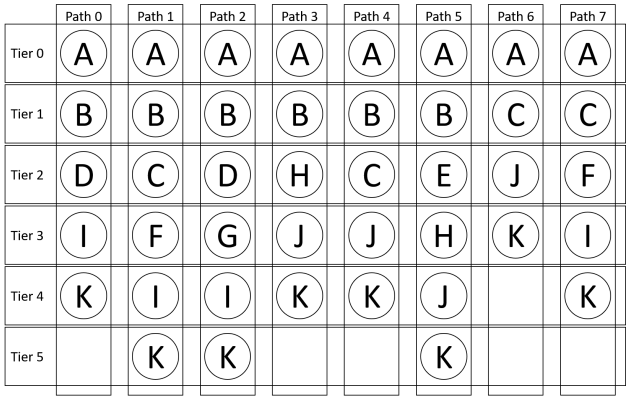


Figure 6.3: Paths Through Graph

## 6.3 Determine Node Tiers

After separating out the distinct paths, our goal is to first sort the paths by some level of importance before merging them back together. Imagine sliding one path over another such that the nodes common to both are superimposed while nodes unique to each path are left side-by-side. One glance at fig 6.3 shows that such an operation would encounter immediate complications. Attempting to align $Path0$ with $Path1$ would lead to duplicate versions of nodes $I$ and $K$ unless they were somehow shifted vertically so as to appear in the same tier for both paths. To that end, the next step involves aligning nodes such that they share a common tier across all paths.

Every path is first shifted down such that missing nodes are moved to the "top" of the path. In the first frame of fig 6.4 the blank spaces formerly in tiers 4 and 5 have been moved to the front of each path and are now shown as black circles. Each node is then shifted individually back into place, the process detailed in algorithm 4. Before shifting a node $n$, the lowest tier at which it currently appears in any path is determined and becomes the target tier for $n$. Each path containing $n$ is then checked in turn. If there is already a different node present in the target tier for $n$ then that node becomes the focus and is positioned before returning to finish shifting $n$. If there is instead a space in the target tier for $n$ then $n$ is moved, and its original space is left empty. The middle frame in fig 6.4 shows nodes $A$ and $B$ being repositioned so that they are in the same tier for every path. The last frame shows the same set of paths after all nodes have been correctly repositioned.

**Algorithm 4** Reposition node $n$ so that it appears in the same tier (index) for every path in $P$

```
1:  procedure REPOSITION(Graph G, Paths P, Node n)
2:      for all p ∈ P do
3:          let t_n be the current tier of n ∈ p
4:          let t_min be the lowest tier that n appears at in all paths.
5:          if t_n > t_min then
6:              let m be the node at position t_n − 1 ∈ p
7:              if m is empty then
8:                  Move n into the lower tier, leaving its old position empty.
9:              else
10:                 REPOSITION(G, P, m)
11:                 REPOSITION(G, P, n)
12:             end if
13:         end if
14:     end for
15: end procedure
```

## 6.4 (Optional) Account for Bypass Edges

One step often required in the context of this project, but which is not a requirement of general weighted directed graphs with variable edge widths is to account for the existence of bypass edges. A Bypass edge (often referred to simply as a bypass) occurs in cases where student test data indicates the existence of a connection between nodes not present on the original learning map.

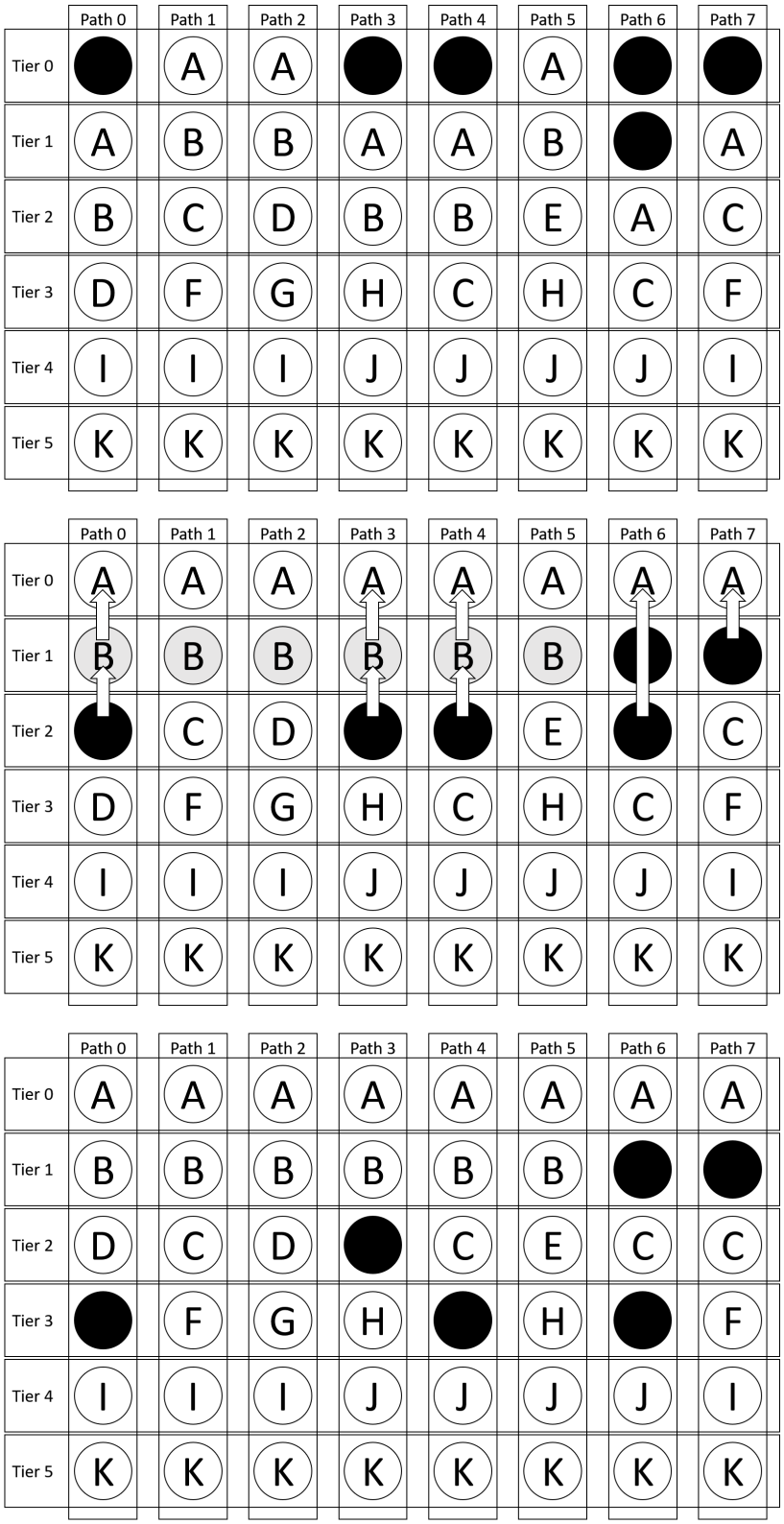


Figure 6.4: Tier State

They often indicate a lack of alignment between the test, curriculum, and original learning map. For example in the graph shown in fig 6.2 a student's test results for the map may indicate mastery of nodes $B$ and $J$ while failing to master nodes $C$ and $H$. According to the original learning map there is no path between node $B$ and node $J$ that does not pass through either node $H$ or node $C$, implying that the student has somehow "bypassed" nodes $H$ and $C$.

Not every pair of nodes can be connected. The term *bypass* carries with it the idea of going around an obstacle on the path to somewhere else. In the same way, a bypass cannot join two nodes unless there exists a way to traverse them without the bypass. For example, a student could conceivably use their mastery of the concepts at node $D$ in fig 6.2 to intuit understanding of node $I$. On the other hand, it is unlikely that they could use knowledge of node $D$ to infer anything all about node $F$. It could be argued that, were such a connection to appear in the data, it would almost certainly have to draw from, or simultaneously prompt mastery of, the concept or concepts common to both paths. The concepts common to both in this case are nodes $A$ and $B$. This would then imply a bypass, not between nodes $D$ and $F$, but instead between nodes $B$ and $F$. Bypass edges, by their nature, represent data that is often not wholly understood - even by those with intimate knowledge of the concepts represented by each node. It is important, therefore, that the visualization refrain from implying the existence of connections without sufficient evidence. It is reasonable to infer from a student's data that, while traveling along a specific path, the student skipped mastering one concept before continuing on to master another on the same path. It is less reasonable to infer such a connection between two different paths on opposite sides of the learning map.

The first step in accounting for bypass edges lies in idenpngying them. Algorithm 5 presents a brute-force style algorithm in which every pair of nodes in a graph is examined to determine whether a bypass edge should connect them. For each pair, the node's tiers are used to ensure that the direction of flow is preserved. Additional checks ensure that there is not already a normal edge connecting the nodes, and that there is a valid path that can be formed between them without the use of a bypass.

After determining whether a bypass edge is possible, the next loop (lines 9-19) check to see whether it is required. Students are filtered to select only those with mastery of both nodes. From there, every path between the nodes is computed and subsequently compared to every student in the filtered set. If every node in the path has been mastered by the student, then they are considered to have mastered the path. If any student exists for which no path has been mastered, then the bypass is considered required.

Once all required bypass edges are idenpngied, they are briefly added to the graph as regular edges. All edge scores are then computed using algorithm 3 after which the bypass edges are removed. Note that the steps in this section can, and probably should, be performed as a separate preprocessing step to avoid repeatedly paying the exponential time cost of the functions required to compute edge scores in real time.

**Algorithm 5** Determine whether a pair of nodes form a Bypass Edge

```
1:  procedure IsValidBypass(Graph G, Node n_from, Node n_to, Students S)
2:      let tier_from be the highest tier in which n_from appears.
3:      let tier_to be the highest tier in which n_to appears.
4:      let edgeExists be true if G contains an edge connecting n_from and n_to.
5:      let pathExists be true if a path exists in G between n_to and n_from.
6:      if (tier_from + 1 ≤ tier_to) ∧¬edgeExists ∧ pathExists then
7:          let P be the list of all paths between n_to and n_from in G
8:          let S_Q be all students who have mastered both n_from and n_to
9:          for all s ∈ S_Q do
10:             γ ← 0
11:             for all p ∈ P do
12:                 if Student s has mastered all nodes in path p then
13:                     γ ← 1
14:                 end if
15:             end for
16:             if γ = 0 then                    ▷ Student failed to find a connected path.
17:                 return True
18:             end if
19:         end for
20:     end if
21:     return False
22: end procedure
```

## 6.5 Calculate Node Placement Order

To fulfill the second layout requirement and provide an indication of a path's level of importance, paths with higher traffic, and therefore more importance, are placed to the left of the graph, making them the first to be seen by a user who naturally reads from left to right. Numerous metrics exist to determine the relative importance of pathways. In this paper, the minimum weight of all edges in a path becomes the path score with higher scores indicating paths of greater importance. In the case of ties between path scores when sorting, those with greater path length are given priority. Paths placed first would have far fewer edge collisions, making it advantageous to prioritize longer paths and so protect more edges. Fig 6.5 shows each path and includes the weight of the edges from the original graph. Below each column is its path score.

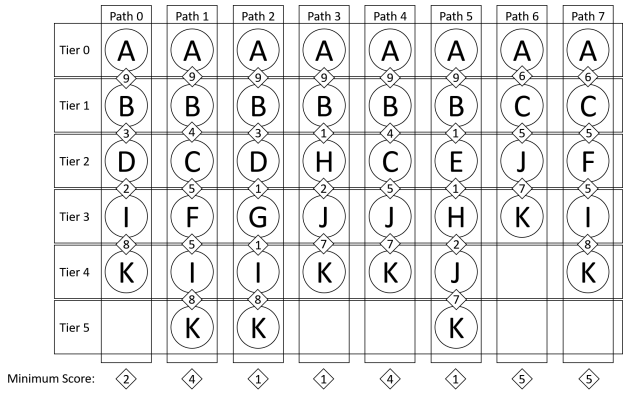


Figure 6.5: Paths Scores

Having sorted the paths by importance, the strategy going forward was to first place the most important, ensuring that it would be rendered as a clear vertical path from top to bottom. Each subsequent path would be placed to the right in order of importance and then shifted left so that it merged with those already placed. This would naturally lead to edge crossings on the right of the graph as edges in less important paths began to conflict, while also keeping the more important paths on the left comparatively clear of obstruction.

To obtain the node placement order, the paths were traversed in order of importance and each node in the path was recorded as it was encountered. In the example, $Path7$ as the most important, first recorded nodes $A$,$C$,$F$,$I$,$K$. This was followed by $Path6$ which skipped nodes $A$ and $C$ as they

were already recorded and instead recored only node $J$. A possible final node order which will be used for following steps is A,C,F,I,K,J,B,D,E,H,G.

## 6.6 Place Nodes

Nodes were placed onto the graph using a grid and were added in the order determined in the previous step. Algorithm 6 details the process used to add each node. Key to this approach is the requirement that every node be placed as far to the left of the graph as possible. The first frame of fig 6.6 shows the first node in the sequence (node $A$) being placed in the grid. Above each column is a set of one or more characters which indicates the tiers in which the node can appear. Referring back to fig 6.2, node $A$ is a parent of nodes $B$ and $C$ which are clearly in different tiers. For an edge to be drawn from node $A$ to node $C$ will require that the edge pass through tier 1. In response to this prediction, the algorithm leaves a space in tier 1 below node $A$ to permit passage of that edge. Hence, node $A$ is placed in the first column of the grid and occupies the first two rows. The second table shows a similar process being applied to node $C$ in which a space will be left in the fourth row to accommodate the edge connecting it to node $J$.

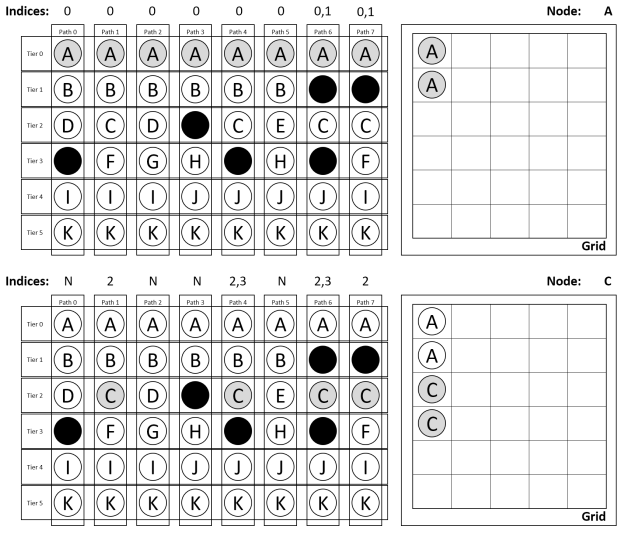


Figure 6.6: Build Grid For Nodes A and C

Figure 6.7 shows node $F$ being added to the grid, and represents the first conflict to occur in

**Algorithm 6** Add Node to Grid

```
1:  procedure AddToGrid(Grid grid, Node n)
2:      col ← −1
3:      let R be the list of tiers in which n can appear.
4:      let grid_{r,c} be the cell at the c^t h column of the r^{th} row of the grid.
5:      a ← 0
6:      while a = 0 do
7:          col ← col + 1
8:          a ← 1
9:          for all r ∈ R do                          ▷ Check each row for availability
10:             if grid_{r,col} ≠ ∅ then
11:                 a ← 0                             ▷ Row not empty in the current column
12:             end if
13:         end for
14:     end while
15:     for r ∈ R do                                  ▷ Add node to rows in the current column.
16:         gridr, col = n
17:     end for
18: end procedure
```

the placement process. By the time node $F$ is placed in the fourth row, the first column is already occupied by node $C$. As a result, node $F$ is placed in the next available column to the right of node $C$. The process continues until node $G$ which is shown in the second frame of fig 6.7. Node $G$ appears only in tier 3. However, as all columns already contain a node in the third row, node $G$ is instead placed alone in the last row.

When all nodes have been placed, all copies of each node except those in the highest tier are removed. Fig 6.8 shows the result of the final layout after all nodes have been placed. While edges have not yet been added to the graph, they are included in the figure to show the passage of edges through the openings in tiers. Were spaces not left below certain nodes, then the nodes $A$, $B$, and $C$ would be directly above one another, making it difficult to follow the edge between nodes $A$ and $C$. It should also be clear from the figure that wider paths have been shifted to the left of the graph.

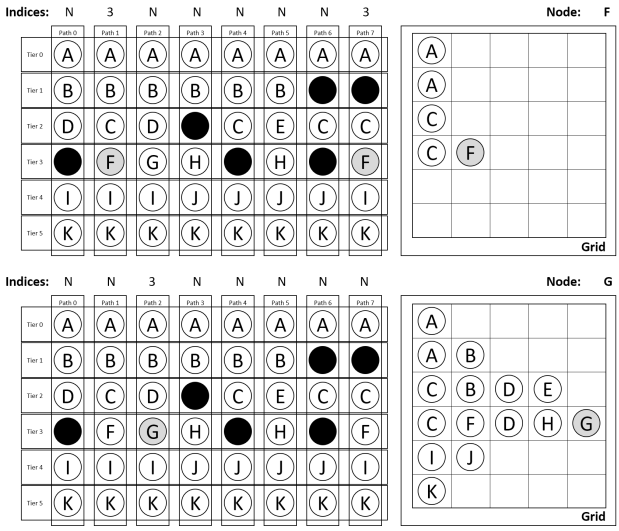


Figure 6.7: Build Grid For Nodes F and G

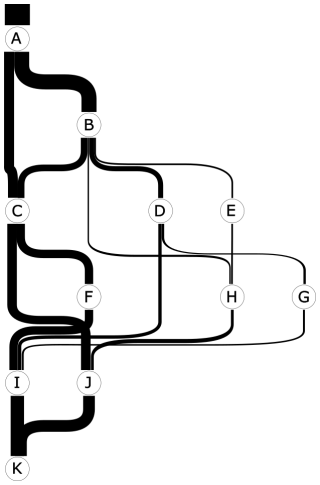


Figure 6.8: Example after Initial Node Placement

## 6.7 Apply Physics System

The graph constructed in the previous section defines the sequence of the nodes within each layer, ensuring that paths are placed left to right in the order of importance. The effect is a very rigid graph with unnecessary crossings due to inconvenient node placement.

The next step taken was the introduction of a physics system incorporating a similar adaptive cooling scheme to that presented by Hu (2006). In such a system, attractive and repulsive forces are applied to all nodes in the graph. Attractive forces are modeled between connected nodes as springs, drawing the nodes towards one another. At the same time, repulsive forces modeled as electrical charges exist between every pair of nodes, preventing them from overlapping one another. Once modeled, the system is released and allowed to settle to a state of equilibrium in which the net force of the system is equal to zero.

One key way in which our approach differed to that of general force-directed approaches was the requirement that the vertical placement of the nodes in the graph remain the same. In each step of the physics loop, the forces acting on each node was calculated, and each force then combined with the current position of the node to determine the next offset. Before updating the node's offset, the y-value of the future offset was changed to match that of the current offset. As the forces themselves were not modified, the effect was of nodes encountering glass panes above and below which allowed only horizontal change. If instead of changing the final offset, the force on the node had been changed, then additional steps would have been needed to ensure that the unmodified force was used for calculation of the net force. Failure to do so would find the system reaching equilibrium prematurely.

Fig 6.9 shows the example graph after the horizontally constrained system has settled.

## 6.8 (Optional) Bypass Gates

Gates, which are used to route bypass edges, are placed to the left of the main graph area between every tier of nodes as shown in fig 6.10.

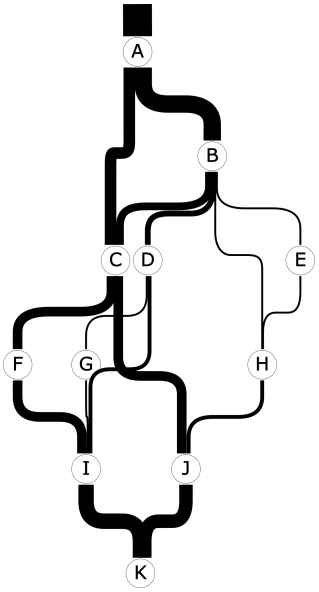


Figure 6.9: Example After Physics System

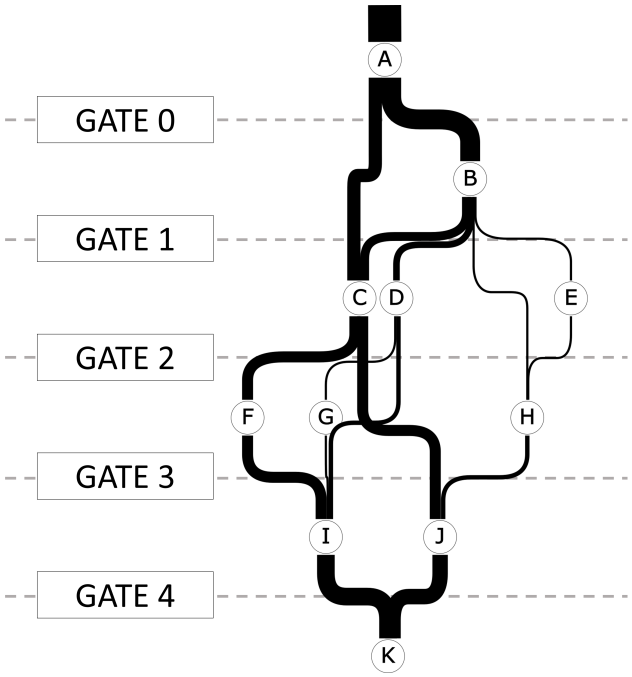


Figure 6.10: Position of Gates

Bypass edges, as a rule, do not pass between adjacent tiers of a graph but rather connect nodes separated by one or more intervening tiers. Instead of passing directly between nodes as is done by normal edges, bypass edges extending from a node in tier $i$ to a node in tier $j$ must first travel through gates $i, i+1, ..., (j-1)$ before reaching the destination node. For example, a bypass edge between the unconnected nodes $B$ and $F$ would be drawn from node $B$ to $Gate1$ then down to $Gate2$ before finally being drawn to node $F$.

Each gate position is calculated and added to the graph as a node object. With the gates in place, bypass edges are returned to the graph. During this process they are first broken up such that they follow a path through the correct gates from source node to destination node. The left frame in fig 6.11, in which bypass edges drawn as dashed arrows, show them connecting node $T$ to nodes $V$ and $W$ and node $U$ to node $W$. Instead of the direct connections, the bypass edges must traverse the gates as shown in the middle frame. In doing so the gates for which each bypass edge must traverse are determined. In the case of the edge between nodes $T$ and $W$, the edge must travel from tier 0 to tier 3, passing through gates 0, 1, and 2 for a total of three gates. If $G$ represents the number of gates traversed by a bypass edge, then the bypass edge is replaced with a set of $G+1$ smaller edges, all with an edge score equal to that of the original. The first of these smaller edges is used to join the source node to the first gate. One edge is then used between each pair of gates, with the last forming the connection between the final gate and the destination node.

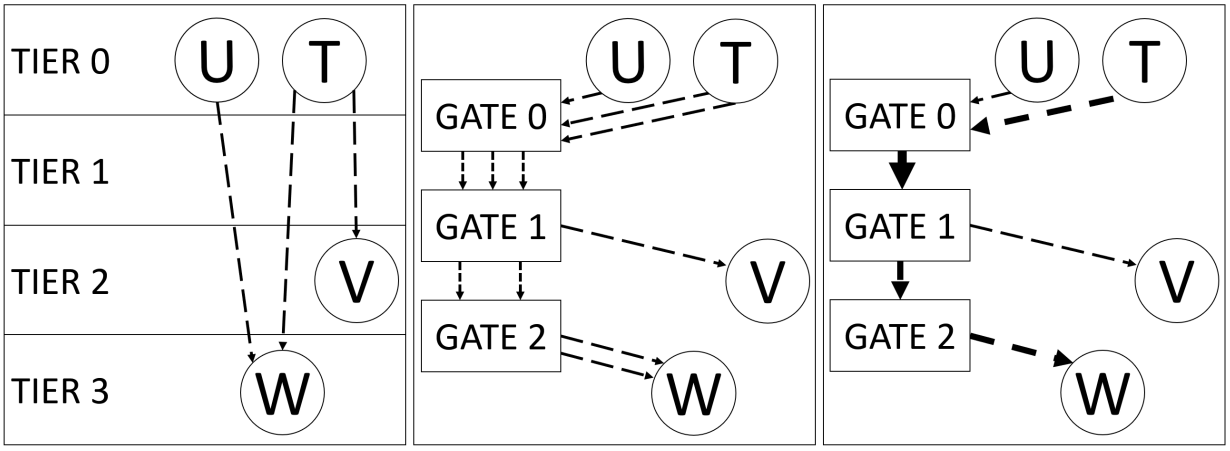


Figure 6.11: Arrow/Gate Interaction

The last step is, for each gate, to replace all edges entering the gate with a single edge. The new edge score is equal to the sum of the scores of the replaced edges. Doing this merges bypass edges as shown in the right hand frame of fig 6.11. When necessary, the process is repeated for edges leaving each gate.

One side effect of this process, as shown by the example in fig 6.11, is the inability to determine the exact destination of edges leaving nodes in the top tier. This is intentional. Bypass edges represent data that is not wholly understood. Knowing that a student mastered node $T$ and node $V$ does not guarantee that a connection exists between the nodes. All it guarantees is that the student in question did not follow the expected path between nodes $T$ and $V$. In this example there could be a node connecting $T$ and $V$ which is not included in the original learning map. The student could also have mastered node $V$ using an entirely different sequence of nodes, making the fact that they also mastered $T$ entirely coincidental. Bypass lines represent an educated guess based on the structure of the surrounding graph. Running bypass edges through gates obscures the information, making it harder for teachers to use when choosing next steps.

Passing the information through gates does not, however, hide the key information presented by bypass edges. While a user may be unable to tell whether a student traveled from node $U$ to node $V$ or node $W$, they can tell from the width of the edges that a number of students appeared to move from node $U$ to some node in a lower tier. Similarly they can discern that a number of students achieved mastery of node $W$ using a path not present in the original learning map. The importance of bypass edge details pales in comparison to the fact of their existence, which often indicates some form of disconnect between the curriculum, test, and original learning map.

## 6.9 Arcs

While arcs are shown in previous figures for clarity, they are only truly added to the graph at this stage of the algorithm. As shown in fig 6.1, edges defined in other graph layout algorithms were generally unsuited to the needs of our visualization. To meet requirement 6, edges had to have a consistent shape. They also had to be strictly controlled so as to follow carefully defined paths and

avoid edge crossings.

Fig 6.12 contains a sketch of the arc used in this algorithm. Each arc is defined using six control points marked as $CP0$ - $CP6$. The curve between $CP1$ and $CP2$ is formed using a quadratic Bezier curve with control points at $CP1$, $QCPA$, and $CP2$. It is mirrored by the curve between $CP3$, $QCPB$, and $CP4$. Each sequential pair of control points is locked on a given axis according to the following constraints:

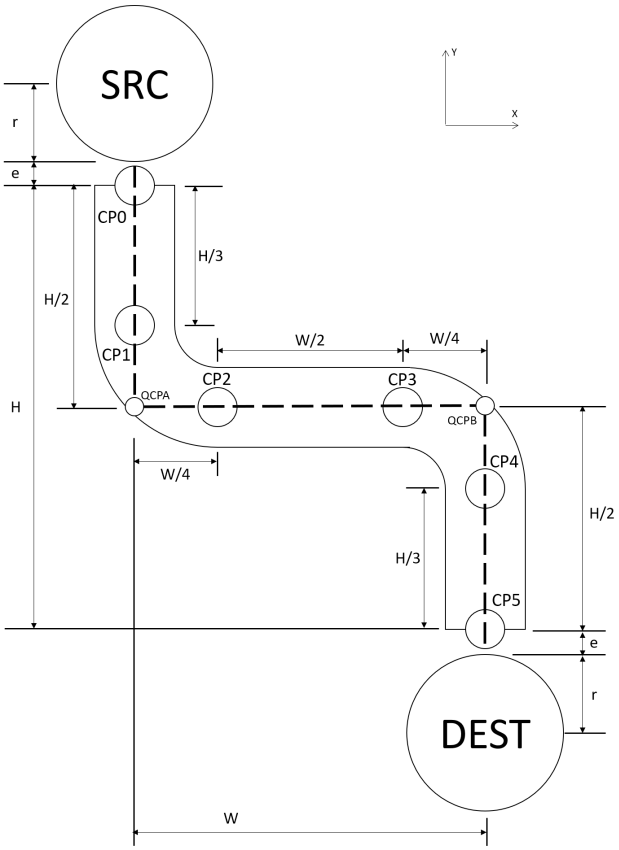


Figure 6.12: Arrow

$$CP0.x = CP1.x \tag{6.1}$$

$$CP2.y = CP3.y \tag{6.2}$$

$$CP4.x = CP5.x \tag{6.3}$$

$$|QCPA - CP2| + |CP2 - CP3| + |CP3 - QCPB| \equiv |QCPA - QCPB| \tag{6.4}$$

Every edge in the graph is replaced by an arc with the center y-axis through $CP2$ and $CP3$ initially set to lie at the vertical center of the arc. The width of the arc is initially set to equal the score of the edge it replaces. After all arcs are placed, their widths are normalized such that the maximum combined width of all arcs entering or leaving any node is equal to the minimum node diameter.

## 6.10 Collision Detection and Correction

Replacing edges with arcs occasionally results in overlap between arcs and nodes such as occurs at node *G* in fig 6.9. This step of the process involves detecting and correcting such overlap.

1. For each node, the x-values of the $CP0$ and $CP1$ control points are reordered such that edge crossings do not occur between arcs leaving the same node.

2. Arcs are realigned so that for all arcs leaving a given node, the $CP2$ and $CP3$ control points lie on the y-axis at a point between the tier of the node and the tier below the node. This is done to meet the fifth layout requirement, which states that edges out of a node be given priority over those leaving. For example, the horizontal piece of the arc connecting *A* and *C* is raised to the same level as the horizontal portion of the arc connecting nodes *A* and *B*.

3. Collision detection is performed between all nodes within a tier. For each collision a small horizontal force is applied to each node, pushing it away from the collision point.

4. Collisions are then detected between nodes and arcs. Since the horizontal segments of the arcs between $CP2$ and $CP3$ are placed between tiers in step 2, they will never intersect a node. The same is true for the vertical arc segments between $CP0$ and $CP1$. Therefore, the only pieces of an arc able to intercept a node are the segments between $CP4$ and $CP5$. Collision detection is performed for each node *n* against any arc which connects a node in a tier above *n* to a node in a tier below *n*. In each case, a ray-circle test is performed between the node and the edges of the line formed by the $CP4$ and $CP5$ control points of the arc. For

every detected collision, a small horizontal force oriented away from the point of collision is applied to the node.

5. For each node, if the sum of the forces acting on it is greater than zero, then the node is moved in the direction of the total force and the control points of arcs entering and leaving the node are recalculated.

6. The steps above are repeated until the sum of all forces acting on any node is zero or until an equilibrium condition is met. The result of these steps on the example graph is shown in fig 6.13.

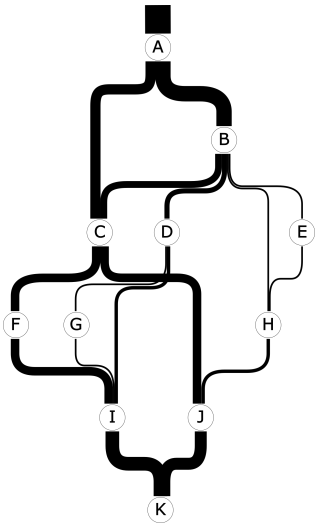


Figure 6.13: Final Result

## 6.11 (Optional) Custom Changes

An additional step can be performed at this stage of the algorithm to apply custom user changes. When building the graph used for interviews in chapter 9, centering and alignment operations were applied to the nodes which produced a cleaner, more streamlined graph. It is recommended that the steps in Section 6.10 be performed after any custom changes to ensure that arcs are correctly aligned.

# Chapter 7

# Envelopes

To avoid confusion between definitions in other sections of this paper, the following terminology is used for this section:

**Anchor** - A circle in Cartesian space with radius, offset, group index, and color.
**Color** - An integer value greater than zero.
**Pixel** - A square in grid space with a row index, column index, and color.
**PixelGrid** - A container with an offset in Cartesian space marking the lower left corner of the lower left pixel and a two-dimensional array of Pixels.

A number of alternatives were considered when determining how best to wrap an envelope around a set of points. Having already spent time working with the spring-electron physics system, it was natural to fist consider a system of springs as a possible option. Such an approach would begin by modeling each node as a point fixed in space. Around each point would be placed a ring of smaller points, each with a small mass and attached to the original via a spring. Each point in the ring would also be joined by springs to its neighboring points, forming a shell able to adapt to external forces acting on any of its corners.

With the rings set in place, the resting length of the springs could be increased incrementally, pushing each point mass outward from their central anchor. After each increase, the system could be allowed to go to equilibrium, the process to repeat until either all spaces between the anchors were enclosed within a spring-bounded shell or the spring length at rest reached a maximum. As each ring slowly expanded, they would encounter other rings, at which point they would deform to

accommodate the external forces acting between them while still forming a clear envelope around their anchor. Fig 7.1 is a rough sketch of what such a system might look like. The four larger circles represent the fixed anchors while the smaller circles indicate the presence of point masses. All circles are joined by heavy lines representing springs. As shown, the rings have expanded until making contact with one another at which point they deform, the forces being redirected, to form envelopes around the anchors.

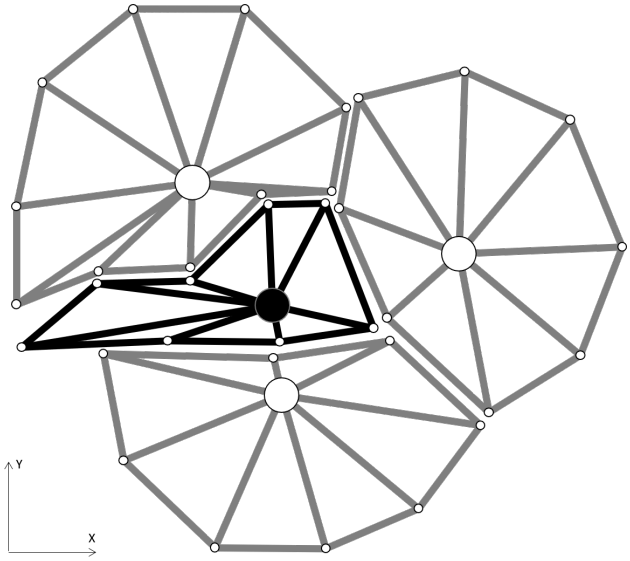


Figure 7.1: Spring Approach

Preliminary testing of the approach suggested it would provide clear, symmetrical shells regardless of the original graph shape. Unfortunately, it also became apparent that such an approach lacked any way of effectively handling the interaction between rings. The small size of the point masses relative to the overall graph area make it likely that the points of two rings would simply slip past one another rather than make contact in such a way as to exert external force. A mechanism was needed to detect and react to collisions between expanding rings.

The most obvious solution was to simply increase the number of point masses in each ring, reducing the chance of a mass slipping between those of other rings. When taken to its extreme, this solution would result in a solid barrier of point masses, entirely preventing any interleaving of rings. However, physics systems of a similar nature used in graph layout applications tend to have a much higher run time cost than other layout algorithms. Similarly, the sheer number of point

masses sufficient to prevent interleaving would, despite the simplicity of the calculations in this approach, incur a run time cost too high to be feasible without the use of parallel architecture.

Another possible solution was to use the same method as that used by force-directed algorithms and avoid overlap by assigning a small electrical charge to each point mass. Application of Coulomb's law between every pair of points would then add a small repulsive force based on the intervening distance. As the points draw close, the repulsive force would offset the force exerted by connected springs, eventually causing the point to stop as the forces reach equilibrium. Though far less than that of the previous solution, this approach faces similar scaling problems before enough points are added to make it viable. It would also need to undergo extensive tuning to ensure a balance between the two sets of forces. Furthermore, it is unclear whether a system, once tuned, could be applied to any shape of graph without requiring additional calibration.

Other potential workarounds included modeling the response of springs with point masses pushing diagonally against them and performing collision detection by applying ray-circle intersections between springs and point masses of other nodes after each iteration. Associated with each solution was the potential for high run times using an increasingly complex physical model, making the use of a physics system infeasible.

Investigations into physics systems having failed to provide a viable approach, our attention turned to geometric modeling. Conceptually, this involved the series of steps shown in fig 7.2. Large circles were first drawn around each anchor, after which lines were drawn tangential to the circular outlines between pairs of connected anchors, with the radius of corner arcs matching those of the original anchors.

This approach proved to be both extremely fast and to produce very clean, precise envelopes. The cost to this approach was discovered in the edge cases, some of which are shown in fig 7.3. Many of these required individual consideration and unique responses. The most basic case is shown in the first frame and is encountered when a set of nodes is connected in a way that forms a hollow center as indicated in the figure by a gray background. Such a case would need additional checks to detect and remove the anomaly. While not an unsolvable problem, most solutions which

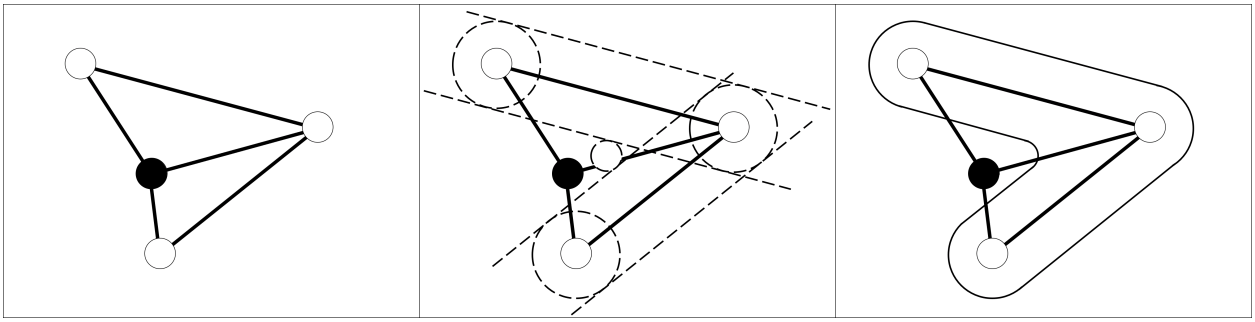

Figure 7.2: Walking Approach for Connected Envelope

would prevent the formation of hollow areas would fail when encountering the edge case shown in the middle frame where the holes technically form outside the hourglass boundaries. The third frame shows anchors which, while not connected, are made to appear so by the overlapping envelopes. This edge case would need to first decide whether a complete envelope was formed before either separating the nodes, deforming the circle outlines, or handling the edge case caused by the hollow center.

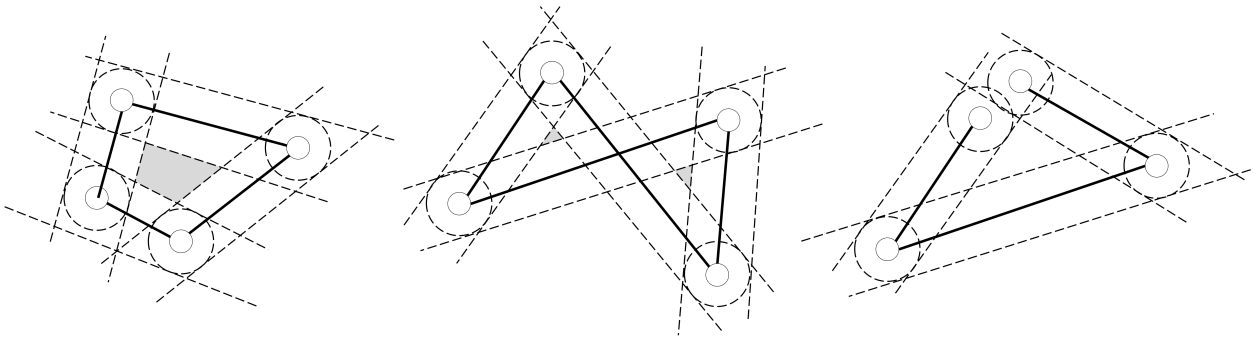

Figure 7.3: Walking Approach for Connected Envelope

The most troubling edge case can be seen in fig 7.4 which features two separate envelopes around the set of white nodes. Splitting the envelope in such a way undermines its purpose in grouping the nodes which would, ideally, enclose all the white nodes to produce a shape similar to that shown in fig 7.2. A solution to this sort of problem would likely employ ray casting to first idenpngy clear avenues and so would have to consider the different edge cases caused by the position of two separate node groups.

Though none were individually unsolvable, the combined weight of edge cases encountered

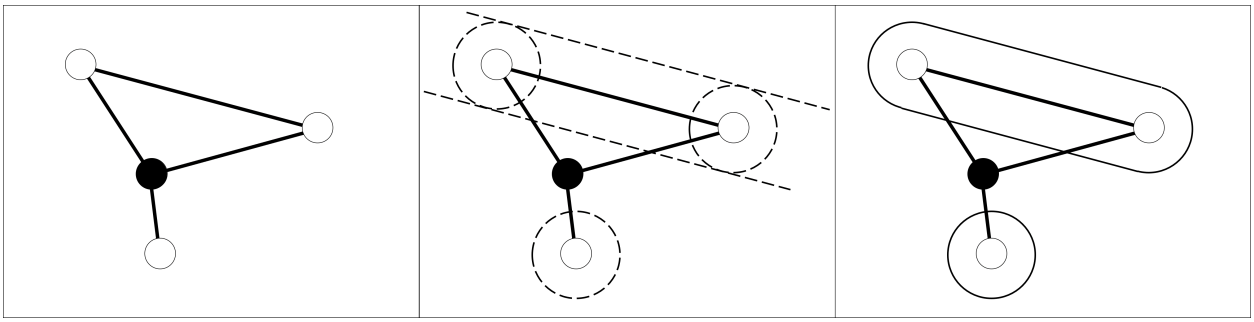

Figure 7.4: Walking Approach for Unconnected Envelope

prior to serious testing prompted a reevaluation of the approach. It was ultimately felt that, if the total overhead of the edge cases already exceeded that of the regular execution for such straightforward situations, then proving consistent operation for larger graphs of unpredictable shape would be challenging.

We finally settled on an approach which employed a flood-fill algorithm in combination with band-pass filtering to wrap certain nodes in smooth envelopes. Throughout this chapter we will be referring to fig 7.5 as an example.

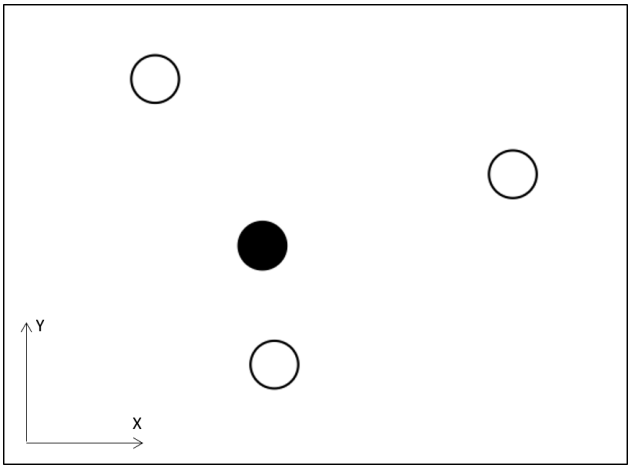


Figure 7.5: Example Configuration

## 7.1 Steps

Section 7.2 - Define the Grid

Section 7.3 - Apply Seed

Section 7.4 - Restricted Flood

Section 7.5 - Build Edges

Section 7.6 - Edge Operations

## 7.2 Define the Grid

During execution of the flood behavior, it is not important where in real space the pixels are located. Of greater importance is a clear indication of which pixels are adjacent to one another. As such, it makes sense to begin by shifting operation to a separate coordinate space. The following equations define the bounds of what will be referred to to as "Grid Space" and is based on the set of anchors provided. Fig 7.6 shows the grid corresponding to our example with the Cartesian origin marked by green arrows.

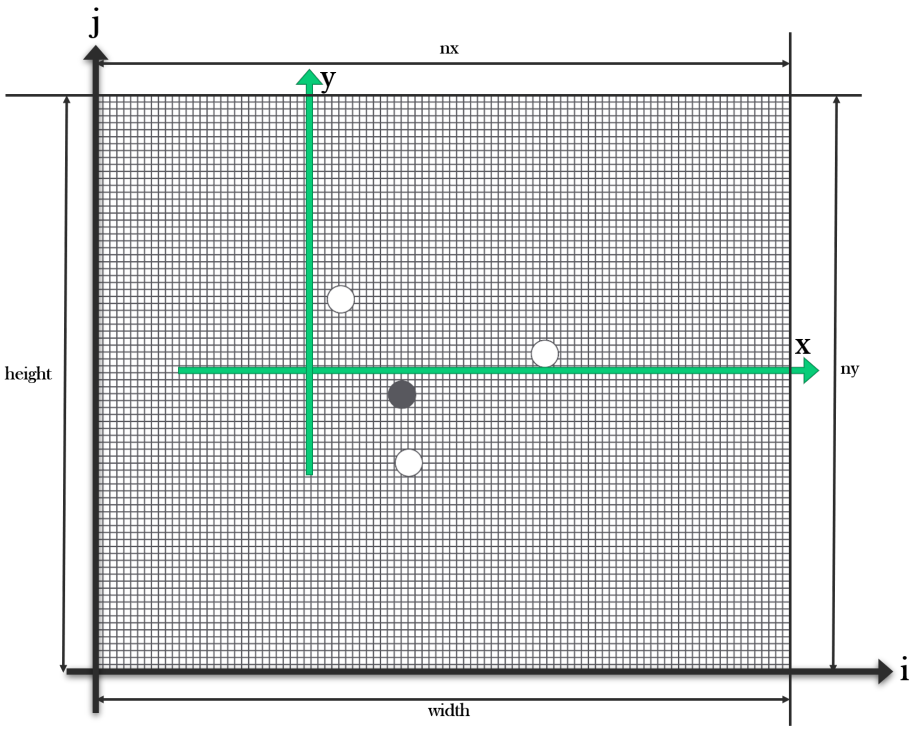


Figure 7.6: Example Grid Space

Grid space is defined as the following:

$$width = 2r + (max(\{a_0.x, ..., a_n.x\}) - min(\{a_0.x, ..., a_n.x\})) \tag{7.1}$$

$$height = 2r + (max(\{a_0.y, ..., a_n.y\}) - min(\{a_0.y, ..., a_n.y\})) \tag{7.2}$$

$$startX = min(\{a_0.x, ..., a_n.x\}) - R \tag{7.3}$$

$$startY = min(\{a_0.y, ..., a_n.y\}) - R \tag{7.4}$$

$$pixelSize = width/nx \tag{7.5}$$

$$ny = height/pixelSize \tag{7.6}$$

where $a_i$ is the $i^{th}$ anchor and $R$ is the maximum flood radius.

Using the resulting variables, the position of a point in Cartesian space at $(x, y)$ can be converted to and from grid space using the following formulae.

$$(i, j) = (\lfloor \frac{x - startX - 0.5 pixelSize}{pixelSize} \rfloor, \lfloor \frac{y - startY - 0.5 pixelSize}{pixelSize} \rfloor) \tag{7.7}$$

$$(x, y) = (startX + pixelSize * i + 0.5 pixelSize, startY + pixelSize * j + 0.5 pixelSize) \tag{7.8}$$

where $i$ and $j$ are the row and column of the pixel containing the point.

## 7.3 Apply Seed

With the grid defined, the points from which to begin the flood are determined. This set of points is referred to as the seed, and each point corresponds to the center of an anchor. The color of every pixel is initially set to zero. Every anchor is then assigned a unique color regardless of their original group. Finally, the seed pixel for every anchor is computed using the anchor's offset, and the pixel's color set to the same color as that of the anchor.

## 7.4 Restricted Flood

Algorithm 7 outlines the process for flooding the grid. Seed pixels are initially placed in an active set defined in the algorithm as $D_A$ where they are then removed individually from the set. As each pixel $p$ is taken from the set, its 4-connected neighbors are each checked to see whether they are a candidate for flooding. To qualify, a 4-connected neighbor must not yet have been assigned a color and must be within a specified distance ($R$) from the anchor whose color matches that of $p$. Upon meeting these conditions, the color of the 4-connected neighbor is changed to match that of $p$ before being added to the next active set. When the active set is empty, it retrieves the contents of the next active set and the process repeats. Flooding is complete when both the active and next active sets are empty.

**Algorithm 7** Restricted Flood

```
1:  procedure RESTRICTEDFLOOD(PixelGrid D, Radius R, Anchors A)
2:      let D_A be the set of all pixels in D with color ≠ 0.
3:      while |D_A| > 0 do
4:          D_N ← {}
5:          for all p ∈ D_A do
6:              let N be all 4-connected neighbors of p with color = 0.
7:              let d_{a,b} be the Cartesian distance between two pixels a and b.
8:              let p_a be the pixel directly below the anchor a ∈ A such that p.color = a.color.
9:
10:             N_R ← {n | n ∈ N, d_{p_a,n} ≤ r}
11:             for all n ∈ N_R do
12:                 n.color = p.color
13:                 D_N ← D_N ∪ n
14:             end for
15:         end for
16:         D_A ← D_N
17:     end while
18: end procedure
```

Fig 7.7 shows the example after 3, 10, and 20 steps into the flood process. The final frame shows the same example after step 27, at which point all pixels within the flood radius have been reached, marking an end to the flood loop.

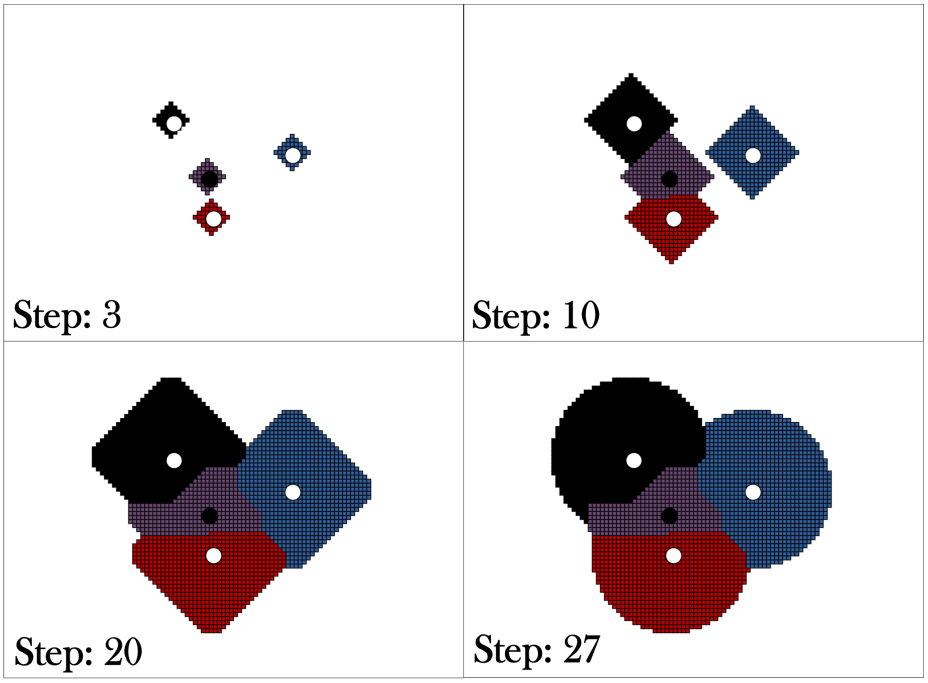


Figure 7.7: Steps in the Flood Process

## 7.5 Build Edges

The results of the flood loop alone, while accurate, are not particularly useful. A method is needed to process the pixelated shells formed by the flood and produce clean envelopes that correctly group nodes. The steps below summarize the process.

1. All shells containing anchors of the same group are merged into a single object.
2. The merged shell is hollowed out to form a set of adjacent pixels.
3. The pixels forming the hollow shell are traversed, building a concave hull surrounding the enclosed anchors.

To improve clarity of the following explanation we refer to the anchor whose color matches that of a pixel as belonging to the pixel. Algorithm 8 details the process used to compute the concave hulls. In the algorithm, pixels are first divided in into groups based on the group index of their anchor. Next, the color of each pixel within the same group is changed to the group index of their anchor, thereby merging them into a single object as shown in the first frame of fig 7.8. Each group

is then filtered, as shown in the middle frame, to remove any pixel whose 4-connected neighbors all have the same color as the pixel itself. The final, more complicated step begins on line 10 and is detailed in algorithm 9.

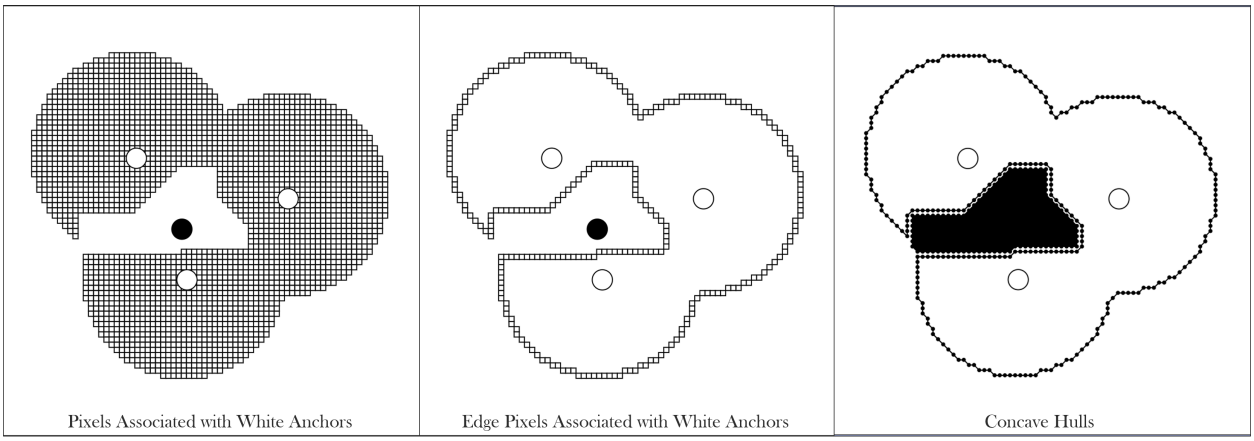


Figure 7.8: Computing Concave Hulls

**Algorithm 8** Compute the concave hull(s) associated with each anchor group for PixelGrid $D$

1: **procedure** CONCAVEHULLS(PixelGrid $D$, Anchors $A$)
2: $H \leftarrow \{\}$
3: let $I_A$ be the set of unique group indices associated with the anchors in $A$
4: let $group(p)$ be the group index of the anchor $a \in A$ such that $a.color = p.color$
5: let $p$ be an *edge pixel* if $|\{x|x \in EdgeNeighbors(p), group(p) \neq group(x)\}| > 0$
6: **for all** $i \in I_A$ **do**
7: $D_a \leftarrow \{p|p \in D, group(p) = i\}$
8: $D_i \leftarrow \{p|p \in D_a, p$ is an edge pixel$\}$
9: **while** $|D_i| > 0$ **do**
10: $P_i \leftarrow$ ALLPATHS($D_i$)
11: let $P_iL$ be the longest path in $P_i$
12: $D_i \leftarrow D_i - P_iL$
13: $H \leftarrow H \cup P_iL$
14: **end while**
15: **end for**
16: **return** $H$
17: **end procedure**

Algorithm 9 begins by selecting a pixel then forming ordered paths from that pixel through as many other pixels in the list as possible and back to itself. The first part of the algorithm (line 5 to line 14) searches through the list of pixels to find a suitable start pixel. For an effective trace, the

start pixel must have exactly two 8-connected neighbors also present within the set. Choosing a pixel such as $P_B$ means that a path can be traced from it to $P_C$ and onward through the list. When it eventually returns via $P_A$, it will have traversed a significant number of the available pixels. While a pixel such as $P_X$ in fig 7.9 may have a single 8-connected neighbor, it is not the main reason to require a start pixel to have exactly two 8-connected neighbors. Instead the requirement is meant to prevent pixels such as $P_F$ or even worse $P_C$ being chosen. Were $P_F$ used as a start pixel then the algorithm would have to account for paths such [$P_F$,$P_C$,$P_D$,$P_E$] in which most pixels are ignored by the trace. This would likely require the algorithm to be run on every neighbor, significantly impacting performance.

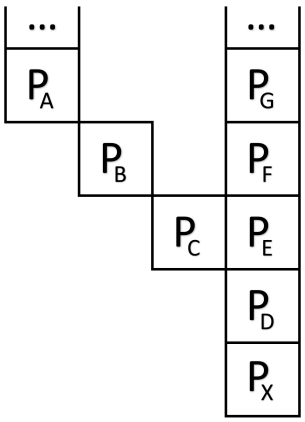


Figure 7.9: Pixel Corner

With a suitable start pixel selected, all paths from one 8-connected neighbor of the start pixel to the other 8-connected neighbor are recovered. For example: starting from $P_B$ to $P_C$ in fig 7.9, the paths recorded would include [$P_B$, $P_C$, $P_D$, $P_E$, $P_F$, $P_G$, ... , $P_A$] as well as the paths [$P_B$, $P_C$, $P_E$, $P_F$, $P_G$, ... , $P_A$] and [$P_B$, $P_C$, $P_F$, $P_G$, ... , $P_A$]. Having gathered the set of paths, the algorithm returns them and the process continues after line 10 in algorithm 8.

In the last part of algorithm 8 the longest path is selected as a piece of the concave hull. At this stage there is a strong likelihood that only one of multiple envelopes enclosed by a given set of pixels has been found. To confirm this, all pixels spanned by the newly added piece of the hull are removed from the original pixel set. If no pixels remain in the set then all pieces of the concave

**Algorithm 9** Find all paths through a set of pixels $P$

```
1: procedure ALLPATHS(Pixels P)
2:     let state be an object with sets: (open, closed) and array: (path)
3:     target ← null
4:     paths ← {}
5:     for all p ∈ P do                          ▷ Find a pixel with two valid neighbors.
6:         if |state.open| = 0 then
7:             N ← {n | n ∈ Neighbors(p), n is the same color as p}
8:             if |N| = 2 then
9:                 Make first neighbor the target to find
10:                Add second neighbor to state.open
11:                Add p to state.closed and state.path
12:            end if
13:        end if
14:    end for
15:    let stack be an ordered FILO array
16:    Push state onto the stack
17:    while |stack| > 0 do
18:        currState ← Pop the last state from the end of the stack array
19:        if |currState.open| > 0 then
20:            currPix ← any pixel in currState.open
21:            Remove the currPix from currState.open
22:            Add currPix to currState.path
23:            if currPix = target then
24:                paths ← paths ∪ currState.path
25:            else
26:                N ← {n | n ∈ Neighbors(p), n ∉ currState.closed, n is the same color as p}
27:                if |N| > 0 then
28:                    for n ∈ N do
29:                        let stateCopy be a copy of currState
30:                        Add n to stateCopy.open and stateCopy.closed
31:                        Push stateCopy onto the stack
32:                    end for
33:                else                          ▷ Reached the end of the path so record it
34:                    paths ← paths ∪ currState.path
35:                end if
36:            end if
37:        end if
38:    end while
39:    return paths
40: end procedure
```

hull have been accounted for. Otherwise the process repeats until each of the envelopes is mapped. The last frame of fig 7.8 shows the final result of this last step.

The performance of this algorithm varies based on the relationship between the number and placement of anchor groups and the grid resolution. A large number of anchors with different groups in close proximity, when combined with insufficient resolution, can result in numerous imperfections. These include hull fragmentation prompting the creation of envelopes without corresponding nodes and isolated pixels caused by sharp corners, such as that shown in fig 7.9, where the longest path in the set cannot pass through $P_X$. Most such imperfections can be removed by raising the grid resolution. Alternatively, algorithm 9 could be modified to include additional checks after line 11 to compare the selected path against a set of conditions and, when necessary, either repairing/removing the invalid paths or dynamically increasing the resolution.

## 7.6 Edge Operations

Numerous operations were applied to the list of 2D points forming the envelope edge that were generated in the previous step. Analogous to electronic filters used to modulate analog signals, these operations transform envelope shape by reducing noise, represented in the geometry as sharp edges and corners, to produce smooth, rounded shapes with high visual fidelity. The relevant operations are described in the following list and include examples applied to lists of numbers in lieu of the more complicated list of vectors.

**Double** - The double operation first buffers the list by adding the first element to the end of the list. For every pair of points the average is then computed and the resulting value is inserted into the list between the original points. As a last step, the buffer is removed by dropping the last point from the end of the list. For example: [0,1,2,3] becomes [0,0.5,1,1.5,2,2.5,3,1.5].

**Halve** - The halve operation removes every second point from the list.

**Moving Average** - When applying the moving average operation, the list was first buffered by adding copies of the first two elements to the end of the list. For every three points, the average of the first and third was computed and the result used to replace the value of the second element. For

example: [0,1,2,3] becomes [0,1,2,1].

**Rotate**: The rotation operation simply applied a shift in the position of all points within the list. By moving the first point to the end of the list, the position of each point in the list was shifted by one to the left.

**Smooth**: When applying the smooth operation, the list was first buffered by adding the last element to the front of the list and the first element to the end, after which the average of every three points in the list was computed. For example, the list of numbers: [0,1,2,3] when smoothed would become [4/3,1,2,5/3].

While a few of the operations above could be repeated to produce a smooth envelope, combining them in different ways often provided more interesting results. Fig 7.10 shows the effect of applying different operations to a list of points. In each frame, the series of connected blue dots indicates the original list of points while the connected black dots represent the same series after application of one or more operations. The specific operations applied in each case are displayed beneath the corresponding frame. Lastly, the arrow running below the frames mark how closely the resulting envelopes conform to the original shape.

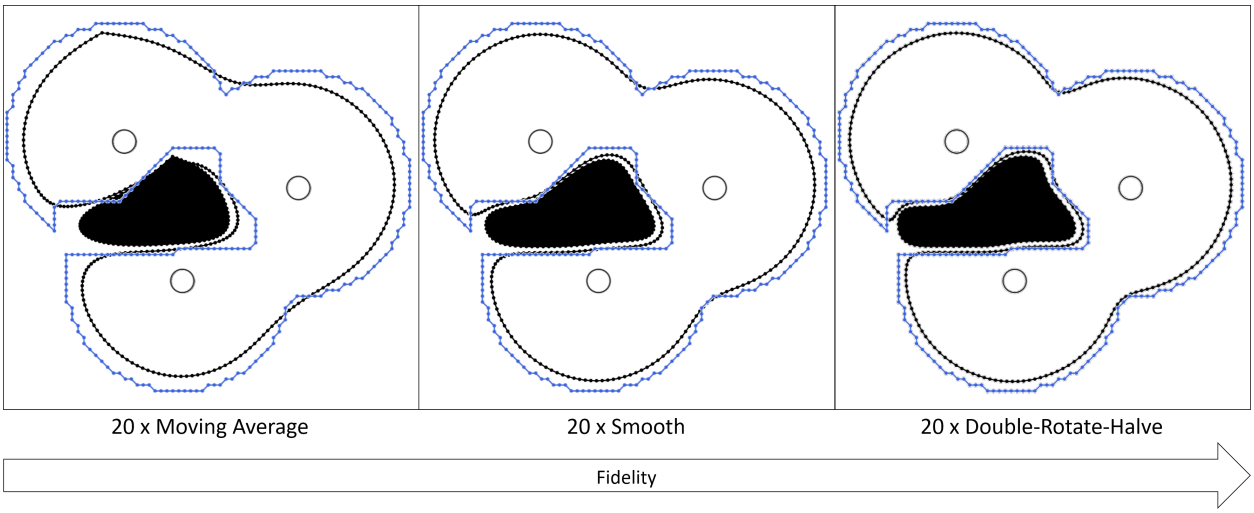


Figure 7.10: Envelope Operation Examples

Envelopes formed by repeatedly applying the moving average operation, as shown in the left frame, are characterized by a sharp point which forms at the start of the list. They also tend to

collapse inward from the original shape more rapidly than other operations, preserving the broad shape while losing many of the fine details. More moderate, the center frame features repeated application of the smooth operation. Though also an based on averaging, smoothing waits until all points have been traversed before applying all averaging operations at once. As a result, the shape conforms more closely to the original than does the moving average. On the other end of the scale is a multi-step operation which employs doubling followed by a single rotation before finally halving the number of points. Were the rotation omitted, the sequence would simply add and remove the same points. The inclusion of the rotation operation means that in each step the entire set of points is, instead, replaced by a set created by taking the average of every sequential pair. This is fundamentally performing the same function as the smooth operation but does so with two points rather than three. As shown in the right frame of fig 7.10, the double-rotate-halve operation preserves the fine details in the original and filters out only the most severe outliers.

Though possible to achieve very detailed envelopes, we found that in practice there was little purpose for such high granularity. The layout algorithms used for node placement generally minimized areas in which such minute fluctuations in envelope shape might be useful. Rather, reviewers found that extra space between lower detail envelopes helped to keep them visually separate. Testing eventually led to a multi-step operation consisting of a single moving average operation followed by two rotations to prevent the formation of the sharp corner shown in fig 7.10. The result of repeated application of this sequence on the example is shown in fig 7.11.

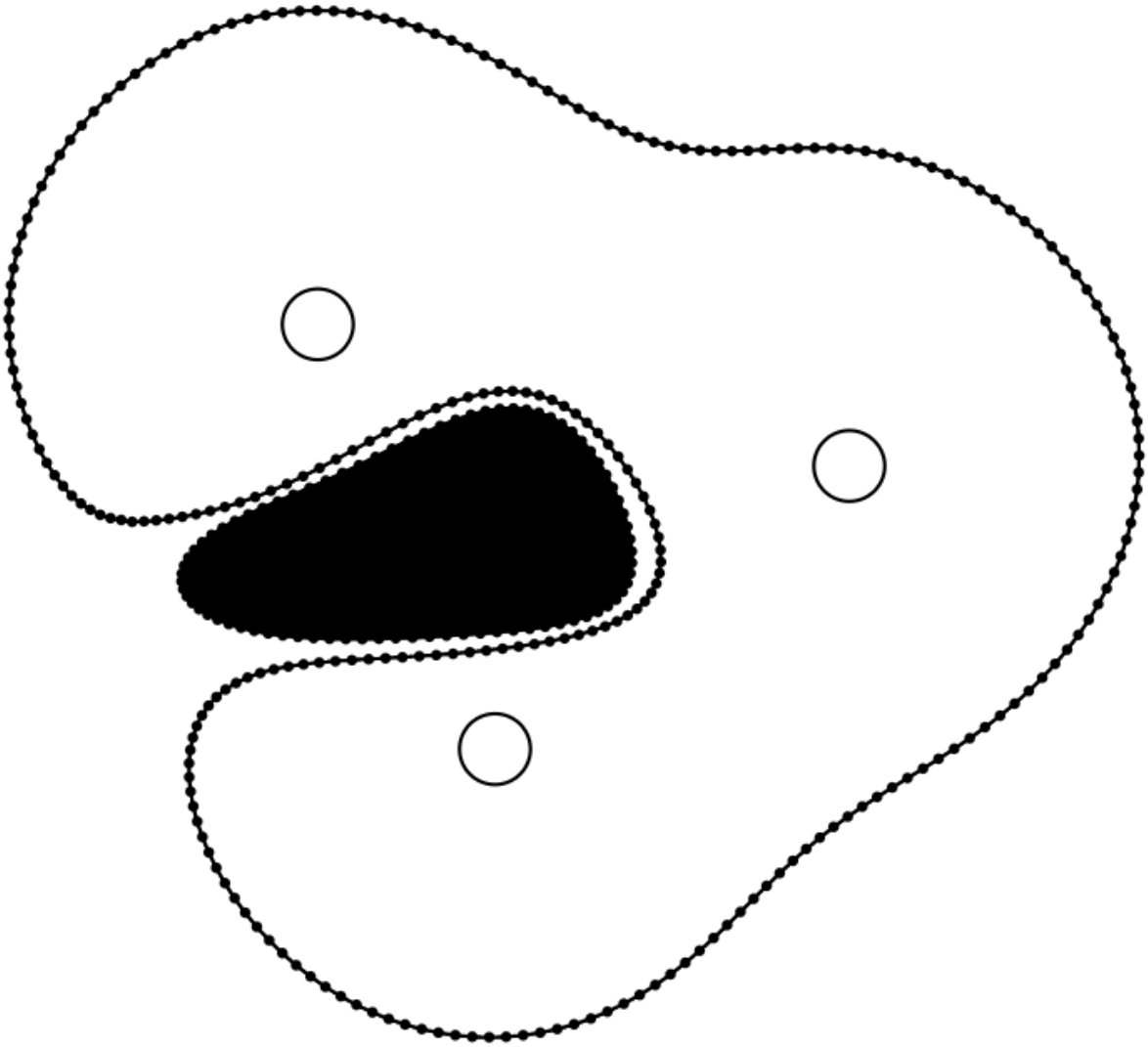

Figure 7.11: All Operations

# Chapter 8

# Contours

By extending the envelopes developed in chapter 7 we were able build a contour map which was then overlaid on the flow graph. At a high level, each node was grouped based on the number of students who had mastered the associated concepts. Nodes with the same number of students were then considered to belong to the same contour level. Each level also had to be placed upon the levels below, meaning that each contour level was also assigned all nodes from the contours at higher levels. Algorithm 10 lays out the key details used to derive the contours through composition of layered envelopes. The algorithm accepts as its inputs a set of nodes, a set of students and a contour type. The contour type describes the location of the peaks and can be set to either *PeakHigh* or *PeakLow* to indicate whether the peaks of the contour map are drawn around nodes for which few students mastered the concepts or nodes for which the majority of students mastered the concepts.

The algorithm determines the number of contours drawn as equal to the maximum number of students on any node plus an additional contour for nodes mastered by no student. For each contour level, the algorithm defines a render set ($R$) and a conflict set ($C$). The envelope drawn around nodes in the render set will be displayed, while the purpose of the envelope drawn around nodes in the conflict set is to prevent the flooding of higher contour levels. For each node in the render set, the algorithm assigns a group index based on the contour level. Each node in the conflict set is assigned a group index of zero. The two sets of nodes are then merged to form the final seed set for the envelope algorithm.

Each of the node sets is individually passed to the envelope algorithm as an anchor set, resulting an a set of envelopes for each of the contour levels. Each envelope is returned with an associated group index matching that of the anchors it contains. Algorithm 11 describes the full process, in

**Algorithm 10** Get flood seed sets for each level of the contour set

```
1:  procedure ContourSet(Nodes N, Students S, ContourType y)
2:      let N_i be the set of students that have mastered i nodes in N
3:      let n be the maximum number of students that have mastered a node in N
4:      E ← {}
5:      for i ← 0, n do
6:          R ← {}
7:          C ← {}
8:          if y = PeakHigh then
9:              R ← N_i ∪ N_{i+1} ∪ ... ∪ N_n
10:             if i > 0 then
11:                 C ← N_0 ∪ N_1 ∪ ... ∪ N_{i-1}
12:             end if
13:         else if y = PeakLow then
14:             if i = 0 then
15:                 R ← N_i ∪ N_{i+1} ∪ ... ∪ N_n
16:             else
17:                 R ← N_1 ∪ N_2 ∪ ... ∪ N_{n-i+1}
18:                 C ← N_0 ∪ N_{n-i+2} ∪ N_{n-i+3} ∪ ... ∪ N_n
19:             end if
20:         end if
21:         let R_c and C_c be copies of the nodes in R and C respectively.
22:         Set the group index of nodes in R_c to i
23:         Set the group index of nodes in C_c to 0
24:         E ← E ∪ {R_c ∪ C_c}
25:     end for
26:     return E
27: end procedure
```

which contour sets are assembled and then drawn. As shown, before any of the envelopes in the set are drawn, they are first filtered to remove any with a group index of zero. These envelopes correspond to anchors in the conflict sets defined by algorithm 10.

**Algorithm 11** Draw a contour map

```
 1: procedure CONTOURMAP(Nodes N, Students S, ContourType y)
 2:     let n be the maximum number of students that have mastered a node in N
 3:     CS ← CONTOURSETS(N,S,y)
 4:     E_0 ← {}
 5:     for all e ∈ CS do
 6:         A ← CONVERTTOANCHORS(e)
 7:         E_0 ← E_0 ∪ ENVELOPE(A)
 8:     end for
 9:     E_f ← {e | e ∈ E_0, e.groupindex > 0}
10:     for i = 1, n do
11:         E_r ← {e | e ∈ E_f, e.groupindex = i}
12:         Draw all envelopes in E_r
13:     end for
14: end procedure
```

Finally, the envelopes are sorted such that those with the lowest group index are first and then drawn in order. Given the high probability that one or envelopes will overlap, the order that they are drawn is important. When drawn in ascending order, the larger envelopes will be drawn first, followed by progressively smaller envelopes drawn one atop the other.

# Chapter 9

# Final Design Phased Evaluation and Refinement Process

Evaluation of the final design synthesized in chapter 5 was done through the use of interviews performed between myself and educators associated with the Enhanced Learning Maps project. Of the five hundred teachers involved in the Enhanced Learning Maps project, forty-two were considered most likely to both have sufficient experience working with the learning maps and be willing to participate in an interview. Nine of those contacted responded, and the results of their interviews are presented throughout this chapter as well as chapter 10.

The iterative process of designing, presenting, and refining had so far been a central component of this research effort. While confident that it would meet the research goal requirements, we nevertheless planned to employ a similar iterative process to refine the final design. To that end, we took into account the time available and determined that there would be sufficient for three iteration cycles. Using a rough estimate of the total number of participants, we divided them in to three groups referred to as phases. Participants within each phase would be shown the same interview tool. After each phase, the responses from participants would be reviewed and changes would be made to the interview tool. Changes made during the development portion of a phase had to follow a set of constraints to either maintain or improve access to information. Participants using the second phase interview tool had to be able, at minimum, to access the same information as those using the tool available during the first phase.

## 9.1 Types of Students

To drive the interviews, a group of students was needed for whom questions could be posed to the participants. Perfect students who had mastered all concepts offered little in the way of useful questions, so to thoroughly test the final visualization we wanted students with differing degrees of misunderstanding. After some consideration, the following set of requirements was generated to describe the students featured in the interviews.

- **Student 0**: Given multiple paths from root to target, the first student should exhibit natural progression along a single well traveled path. The student should not have mastered nodes that deviate from the path but should be able to achieve mastery of the target node in the next step through a high traffic edge. The purpose of this student's inclusion is to challenge the participant's opinion of multiple pathways by presenting a typical student who, having a sound understanding of the material, has not yet explored optional alternatives.

- **Student 1**: The second student should also suffer from holes in their understanding but, unlike the first, they should not yet be at the point where they are able to master the final concept through any well traveled edge. The second student should be representative of someone who struggles to think in the same manner as their peers. Despite that, the student has managed to progress in their understanding to a point where they must either backtrack to more commonly used methods or attempt to learn a concept that most others fail to master entirely. The question raised is which option is more acceptable to the participant.

- **Student 2**: The mastery of the final student should be split such that there are no paths connecting the two pieces for which the student has mastery. At the same time, this student should be able to progress immediately to the target node using a high traffic edge. The purpose of this student is to investigate participant response to disconnected mastery.

A set of one hundred students was simulated using methods defined in chapter 3. From those, a small set of students was extracted to be used as the interview set. Of these, three were selected and small adjustments made to their node mastery, resulting in the three students whose

mastery is shown in fig 9.1. For each case, the blue envelopes enclose nodes for which the student has obtained mastery, while those outside the envelope indicate those which the student failed to master.

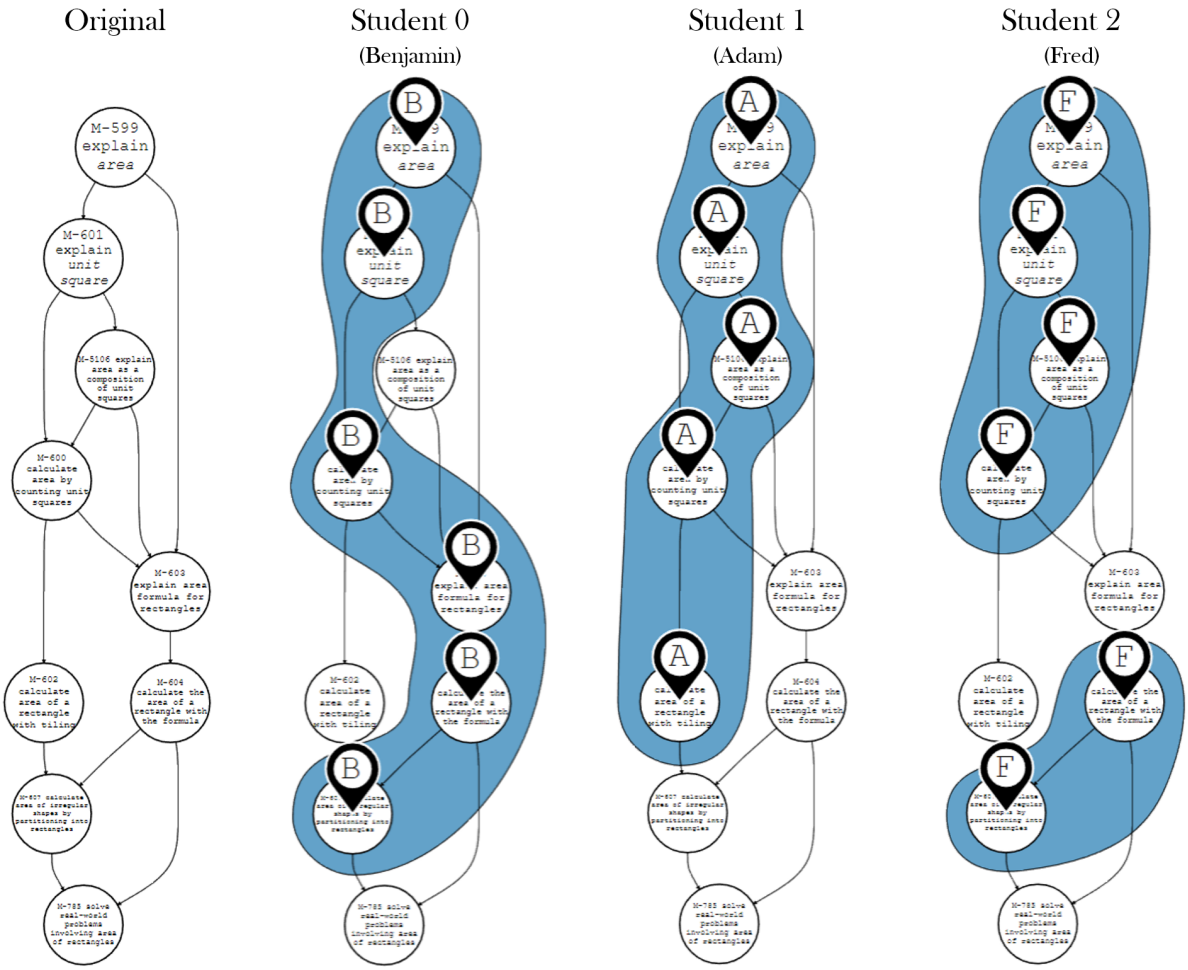


Figure 9.1: Test Students

## 9.2 Interview Tool

The purpose of the interview tool was to enable participants to access and manipulate the various facets of the final design during an interview. As participants would be uncompensated for the time they donated, our focus was in making sure every second of that time was used effectively. Extensive preprocessing was performed to ensure maximum runtime performance, with all student

simulations and graph layouts built and cached before the first interview of each phase.

Development of the tool's user interface used the Elm programming language, a purely functional language which compiles directly to javascript, html and css. While not always as flexible as we wanted, the language was reliable, easy to use and included welcome run-time performance gains during render. Functional programming languages such as Elm have many benefits, both during design and at run time, but they aren't the best tool in all cases. Immutability of most data types in functional languages makes operations involving pointer-like behavior, such as is often seen in flood fill algorithms, difficult to achieve without a loss of performance. When such operations were required in the interview tool, the data was passed from the Elm framework to be handled using mutable javascript, with the result returned to the Elm framework for rendering. Client-side memoization was performed on any run-time operation which occurred outside of the functional Elm framework. Additionally, before being returned to the client application, the results were sent to the server, allowing them to be accessed during future page loads to significantly reduce run-time delay in later interviews.

In all cases, Scalable Vector Graphics (SVG) was used for graph display and interaction. Horak et al. (2018) demonstrated that SVG performance was comparable to that of Canvas for large graphs and we found SVG well suited to the nature of our visualization as well as being easy to integrate within the Elm framework.

### 9.2.1 Phase 1

The focus of development in the first phase of the interview tool was on layout and interaction of the various components. Instead of implementing the algorithms for layout and envelopes presented in chapters 6 and 7 respectively, all visual assets were hand-drawn using Inkscape. These were then loaded to the Elm framework as SVG objects to be manipulated at run time using javascript. These preloaded visual assets included both the standard and flow graphs, all student envelopes, an overlay of the map icons positioned over the nodes and all legend assets.

Due to reasons detailed in chapter 5, the first phase interview tool separated functionality be-

tween the two different graphs. Fig 9.2 and fig 9.3 show respectively the interview tool set to display the standard graph, as developed by the Enhanced Learning Maps project, and the derived flow graph. Each numbered circle in the figure correspond to a number appearing in the feature list below.

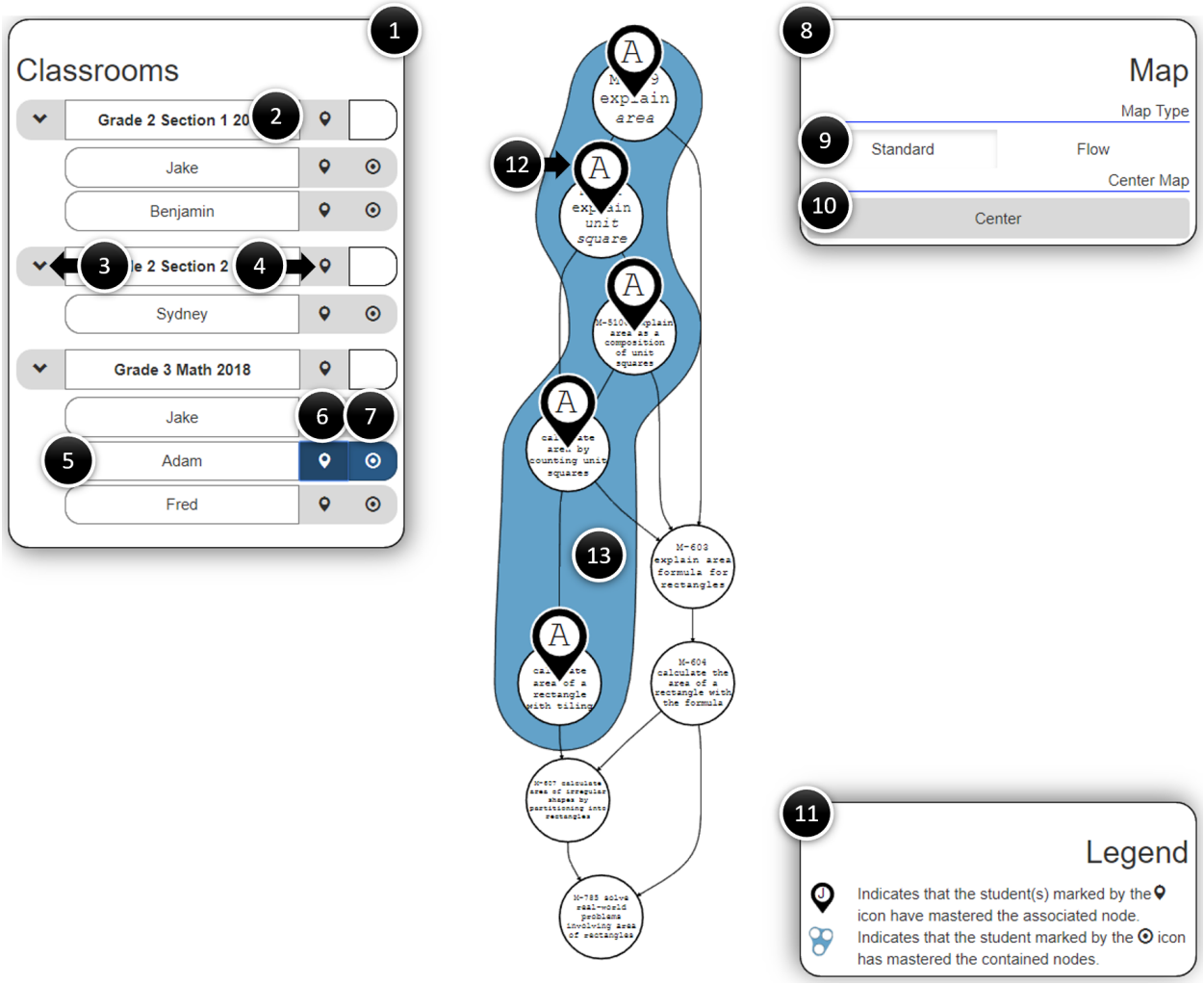


Figure 9.2: Phase 1 - Standard Graph Features

1. Classroom Box - The Classroom Box controls how student and classroom data associated with the user is displayed.

2. Classroom Name - The name of the classroom.

3. Classroom Drop-Down Button - Visually, the Classroom Drop-Down Button indicates whether

the associated classroom section is visible to the user. Clicking on the button shows and hides the associated classroom section.

4. Classroom Icon Button - Visually, the Classroom Icon Button indicates whether all students in the classroom are being displayed using map icons. The button background turns blue when the data of all students in the classroom is displayed. Clicking on the button when it is inactive (gray) sets the data of all students in the classroom to appear in the map icons. Clicking on the button when it is active (blue) removes the data of all students in the classroom from the map icon display.

5. Student Name - The name of a student. In some cases, such as that of "Jake" in the figure, a student may appear in multiple classrooms.

6. Student Icon Button - The Student Icon Button indicates whether the associated student data is being displayed with the map icons. The button background turns blue (active) when the associated student's data is being displayed. Clicking on a student's icon button when active prevents the associated student's data from being displayed with the map icons. Clicking on a student's icon button when it is inactive (gray) first prevents data from all students from being displayed with map icons before allowing the data of the student associated with the clicked button to be displayed. Icons associated with one student could be shown at the same time as the envelopes associated with another, allowing for a comparison of mastery.

7. Student Envelope Button - The Student Envelope Button indicates whether the associated student data is being displayed with an envelope. The button background turns blue (active) when the associated student's data is being displayed. Clicking on a student's envelope button when active prevents the associated student's data from being displayed with an envelope. Clicking on a student's envelope button when it is inactive (gray) first prevents data from all students from being displayed with envelopes before allowing the data of the student associated with the clicked button to be displayed. Icons associated with one student

could be shown at the same time as the envelopes associated with another, allowing for a comparison of mastery.

8. Map Box - The Map Box controls how the map is displayed.

9. Map Type - The Map Type switch toggles between the Standard View (Fig 9.2) and Flow View (Fig 9.3).

10. Center Map Button - Clicking the Center Map Button moves and scales the map so that it appears in the middle of, and fits entirely within, the viewing area.

11. Legend - The Legend provides an explanation of the visual variables currently being used in the application.

12. Map Icon - Map icons indicate mastery of the associated node for all students whose student icon button (see #6) is set to active (blue). The text within the icon shows the first character of the student who has mastered the node. If more than one of the selected students has mastered the associated node, the text changes to a number indicating the quantity of students with mastery. Nodes without map icons represent concepts for which the selected students tried and failed to obtain mastery. Hovering over a node with a visible map icon caused the map icon to become semi-transparent, making any obscured text beneath visible.

13. Envelope - All nodes within the blue envelope have been mastered by the student whose envelope button (see #7) is set to active. Nodes that fall outside the envelope represent concepts for which the selected students tried and failed to obtain mastery.

14. Highlighted Flow Edge - The yellow edge indicates a path that was traveled by the student whose data is being shown by an envelope (see 7). An edge is considered to be traveled by a student if both the source and target nodes of the edge have been mastered by the student.

15. Flow Edge - Flow edges represent edges that appear in the original learning map and therefore have a research-based juspngication for their placement. An edge is considered to be

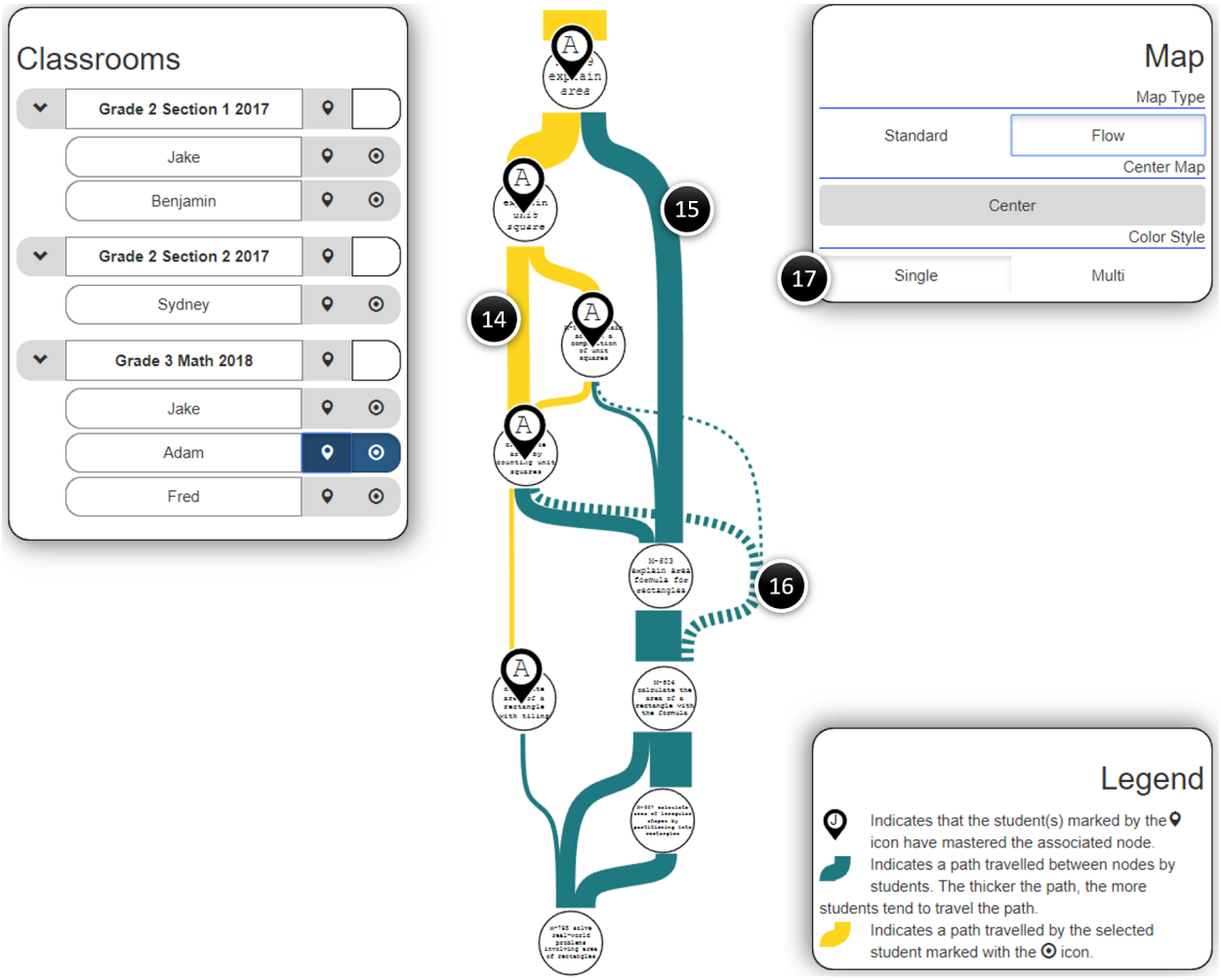


Figure 9.3: Phase 1 - Flow Graph Features

traveled by a student if both the source and target nodes of the edge have been mastered by the student.

16. Bypass Edge - Bypass edges represent connections that do not appear in the original learning map and so indicate events for which there is no explanation. They often point to areas in which either the original map, placement test, or curriculum fail to align.

17. Color Style - Color style buttons toggle the color of the un-highlighted flow and bypass edges between a single blue hue and a multi-color scheme that sets the hue based on the relative width of the edge (see 4.7).

Many lessons were learned over the course of the initial phase of interviews. The first was that, in many cases, teachers would naturally fall back on their own experience rather than using the visualization. This was especially true of math teachers, many of whom were already familiar with the concepts covered by the interview map. Instead of relying on the visual variables, they would read the text within the node and use it to make decisions. In response to this observation, subsequent versions of the interview tool included the ability to modify (or hide) the text shown within each node (see 9.2.2 #18).

While participants found the map icons essential to viewing classroom data, the icons proved less useful when used to view a single student. In some cases, participants who used only the map icons missed spotting them on certain nodes. The color of the map icons was, therefore, changed to a blue color in future versions of the tool to improve visibility when used without envelopes (see 9.2.2 #22). Reviewers also expressed concern regarding the use of the first character of a student's name in icons where a single student had mastered the associated node. Participants were able to instantly recognize the connection between character and name without explanation. The problem they saw was scalability as eventually two students will exist whose names share the same first letter. To address this, the map icons were changed such that in all cases the text within the icon would indicate the number of students currently selected who had mastered the associated node.

While nodes within blue envelopes were intuitively thought to be mastered, participants often

had difficulty classifying nodes that appeared outside the envelope. Nodes outside an envelope can be thought of as

- Untested - the test did not sufficiently examine the node and so no determination could be made about mastery.
- Not Mastered - the test was written to sufficiently test the node but the results were not sufficient to indicate mastery. While it is clear that they did not achieve mastery, there is also no indication of whether they have misunderstood the concepts.
- Misunderstanding - the results of the test contain clear evidence of the student attempting and failing to achieve mastery of the associated concepts and indicates that the student has a flaw in their understanding.

While seemingly identical on the surface, to an educator each of these situations represent a unique set of challenges and entail different responses. For example, a teacher who knows a student has misunderstood a concept may choose to take a different approach than with one who has never seen the material and doesn't have to unlearn habits. To clearly indicate that nodes outside the blue envelopes were those for which the student had demonstrated misunderstanding, red envelopes were added to the visualization tool (see 9.2.2 #19).

Another lesson which prompted a dramatic shift in design was that participants were observed to be more comfortable using the envelopes for generalization but found it easier to make critical decisions with the flow graph. Fig 9.4 shows the decisions made by participants in phase 1. Participants were first shown the left graph and struggled, when asked, to choose the next concept to teach the student. They did not consider any of the options inferior but simply had no way of evaluating which was a better choice. However, when shown the flow graph on the right, they each selected the same node. While time is not a perfect measure in this case, it is useful to note that any decisions made using the left graph generally required 5-10 minutes, during which the participant would explain and consider the merits of each choice, often going back and forth between them. In contrast, decisions made using the right hand graph were typically made within seconds.

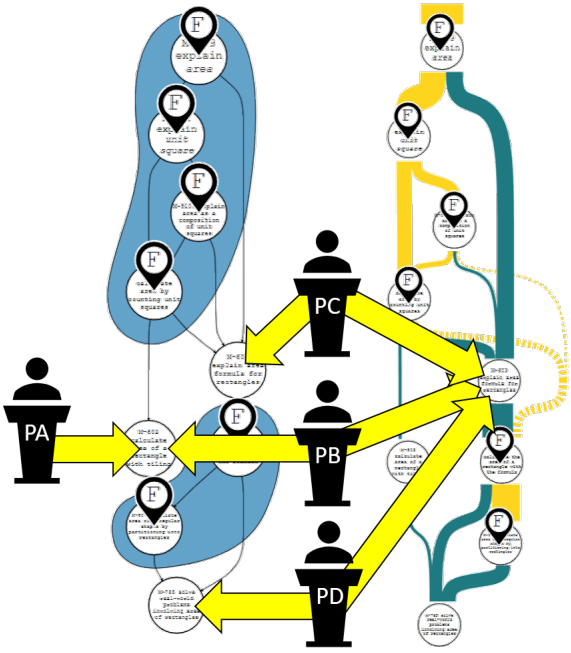


Figure 9.4: Phase 1 Fred Results

Participants appeared comfortable with the familiarity of the standard graph and so were hesitant to switch to the unknown flow graph. So much so that, in most cases, a prompt was required for them to switch. However, after first using the flow graph, participants would almost never require a second prompt when faced with a similar situation. The results shown in fig 9.4 beg the question of why the two graphs were not originally merged. As stated in chapter 5, reviewers originally felt that merging the graphs would prove overwhelming to participants. Confronted by the evidence above, we judged the potential benefits of merging to outweigh the risk, and moved envelopes onto the flow graph for following interview phases.

One surprising observation was the overwhelmingly positive reaction received from participants in response to the classroom map icons for both graphs. Though extremely primitive, participants were eager to offer suggestions and present use cases well beyond anything we had considered prior to this. A feature which was frequently requested was the addition of some means of idenpngying which students were included in the icon numbers. In response, a feature was added in phase 2 in which a participant could hover the mouse over an icon, causing the names of represented students to be highlighted in the left hand menu. The addition of this feature also supported

the earlier decision to remove the first character of a selected student's name from the map icons.

### 9.2.2 Phase 2

Apart from those detailed at the end of section 9.2.1, major changes were made during phase 2 development to the back-end architecture. Evidence collected during phase 1 interviews strongly suggested that the final visualization was meeting the expected goals. This meant that, while the visuals in the first phase had been hand drawn, those in the second were precomputed using the algorithms presented in previous chapters.

Fig 9.5 highlights the changes made to features between the first and second phases.

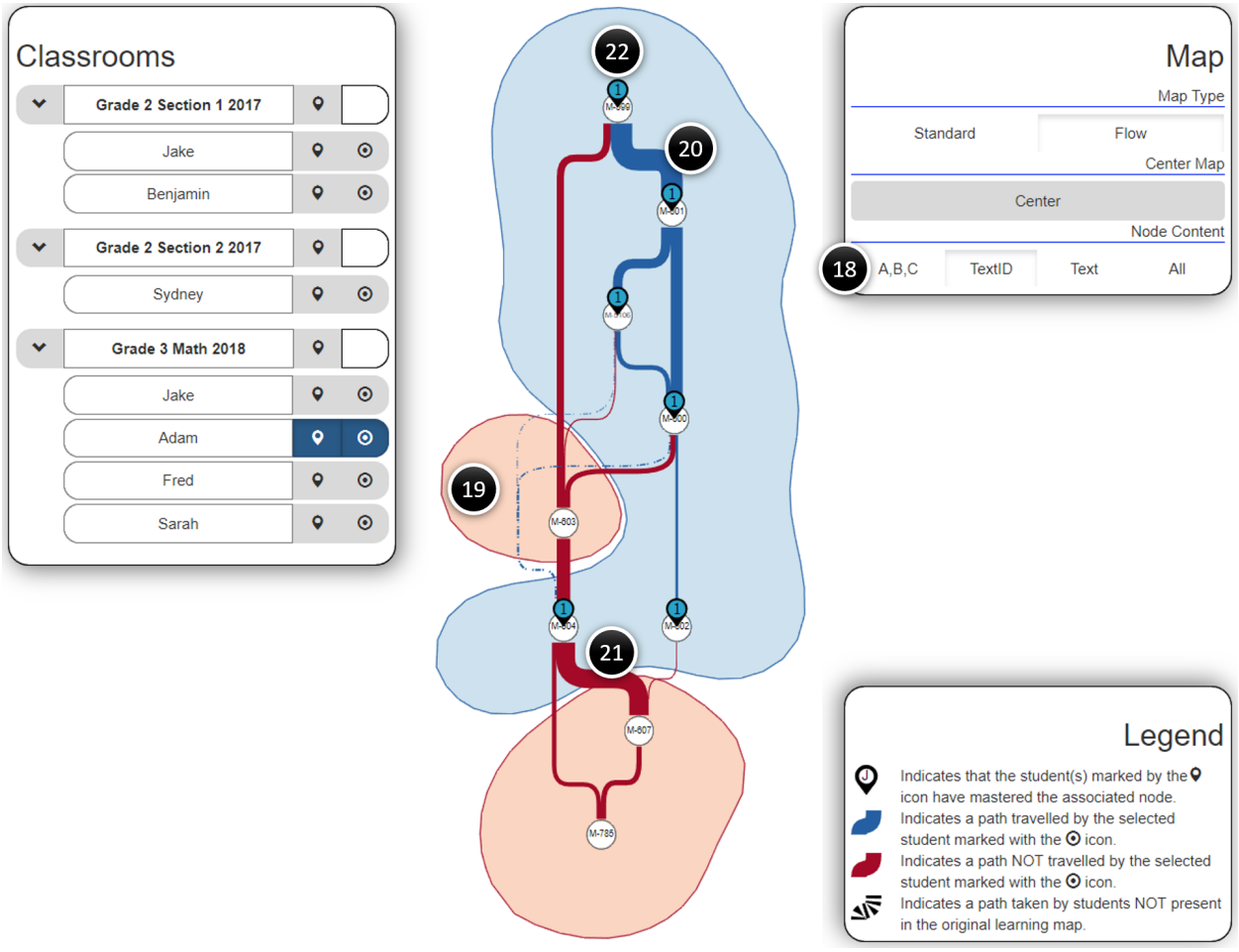


Figure 9.5: Phase 2 Features

18. Node Content - The node content buttons allowed control over the text displayed within each node. Selecting "A,B,C" shows a unique character in each node. Choosing "TextID" displays the id of the node as shown in the original learning map (e.g. M-600) but does not display the node text. The "Text" option conversely shows the text associated with node from the original learning map but does not show the id. "All" shows both the node's id and its associated text separated by a dash.

19. Red Envelope - All nodes within the red envelope represent concepts for which the student whose envelope button (see 9.2.1 #7) is set to active tried and failed to obtain mastery.

20. Blue Edge - Blue edges indicate a path that was traveled by the student whose data is being shown by an envelope (see 9.2.1 #7) and are represented by yellow edges (see 9.2.1 #14) in phase 1. Hovering over an edge causes it to highlight, making it easier to trace.

21. Red Edge - Red edges indicate a path for which the student whose data is being shown by an envelope (see 9.2.1 #7) tried and failed to travel. Hovering over an edge causes it to highlight, making it easier to trace.

22. Map Icon - Map icons in phase 2 differ from those in phase 1 (9.2.1 #12) in a number of ways. The most obvious change is the difference in background color from white to light blue, improving visibility when used without envelopes. Additionally, map icons no longer show the first character of the student's name for nodes in which a single student has gained mastery. Instead the icon always represents the number of students whose data is being shown with map icons who have mastered the associated node. Hovering the mouse cursor over a map icon highlights selected students who have mastered the corresponding node.

As with the first phase of interviews, lessons were learned during interviews with participants in phase 2. The most extreme change was the addition of the contours presented in chapter 8. While teachers in the first two phases found the idea of performing comparisons between two students using envelopes and icons independently appealing, in practice the process proved unintuitive. Additionally, while viewing icons for a class was very well received, it was hard to convey

meaningful comparative data using numbers, even on such a small map. Contours were expected to provide solutions to both problems.

The second, less dramatic change was to remove the envelope buttons from the side menu. Observations of teachers in the first two phases showed that they almost never displayed icons without also showing the envelopes and flow graph highlights. Moreover, as no way existed at the time to use envelopes for multi-student analysis, showing them often hindered participants in cases when the icons alone would have been preferable. The addition of contours meant that, in every case both icons and envelope would provide useful information. Removal of envelope buttons prompted the addition of filter buttons below the classroom areas in the side menu. These allowed components of the visualization to be hidden, preserving the former functionality of the envelope and icon buttons.

The final change was to provide an indication of the ratio of students in a class traveling along an edge. In phase 3, the edge starts with a thin strip of color showing the percentage of students that travel, out of the total students in the available classrooms. This percentage is fixed above a minimum of 50% to allow for readability when few students are selected. For example, if ten out of ten students are selected then the blue edge width is one hundred percent of the total edge. If six of ten are selected then the width of the blue edge is sixty percent of the total edge width and a black border appears. Prior to release, we determined that the addition of such edge information would complicate understanding so, while the feature is included in phase 3, it is never explained. Rather, the edges are treated as normal, with the black borders serving as a means of more clearly defining the edge.

### 9.2.3 Phase 3

The addition of contours to the third phase of the interview tool necessitated a few major changes to a design already focused on high performance. The flood algorithm used in the second phase was fast, but not fast enough that it could be used at runtime without the delay being noticeable. These delays were prevented by pre-computing the student envelopes. However, to fully account for

every possibility of contours in the same manner would have required that $|student|!$ contours be precomputed. The envelope algorithm was instead optimized and moved from the Elm framework to imperative javascript. Memoization which persisted between the page refresh also reduced delay on common contour operations. While not seamless, the changes allowed for contours to be computed fast enough to be useful in real time. As with the first two phases, fig 9.6 provides a labeled list of features unique to the third phase of interviews.

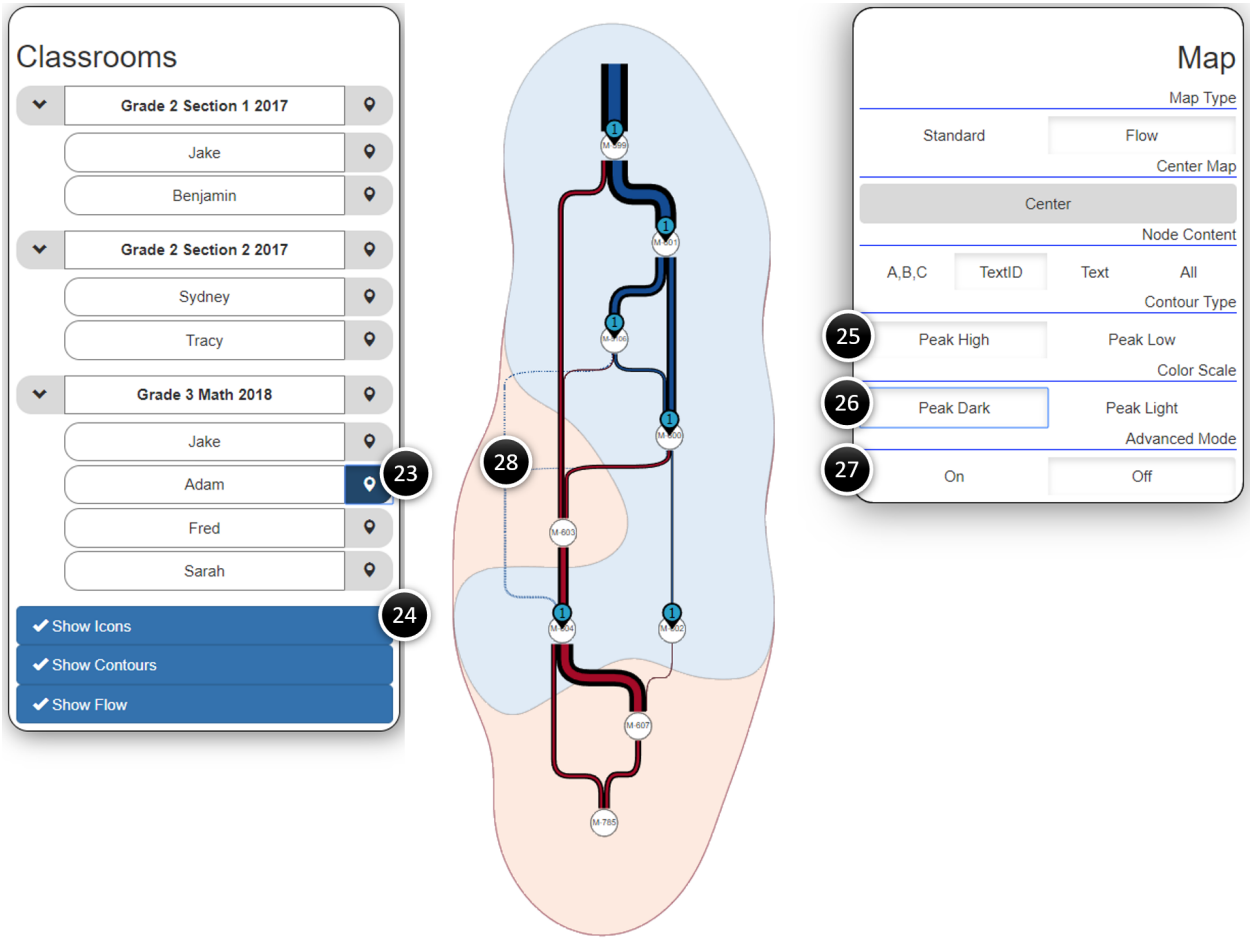


Figure 9.6: Phase 3 Features

23. Student Icon Button - Rather than two buttons to independently control use of student data in map icons and envelope as shown in phases 1 and 2, phase 3 uses a single button to indicate whether a student's data is shown on the map. The icon button turns blue (active) when a student's data is shown on the map in any form. Clicking on a student's icon button shows

or hides the selected student's data on the map. Data from multiple students can be shown by clicking on their icons.

24. Filter Buttons - Filter buttons appear below classrooms when one or more student icon buttons are active (blue). Clicking on the "Show Icons", "Show Contours" and "Show Flow" buttons shows or hides the map icons, contour envelopes, and flow edge/bypass colors respectively.

25. Contour Type - The contour type controls where the peak of the contour map is drawn. When set to "Peak High" the peaks of the contour map are drawn around nodes mastered by the highest number of students as shown in the first and second graphs in fig 9.7. "Peak Low" causes the peaks to be drawn around nodes mastered by the lowest number of students as shown in the third and fourth graphs.

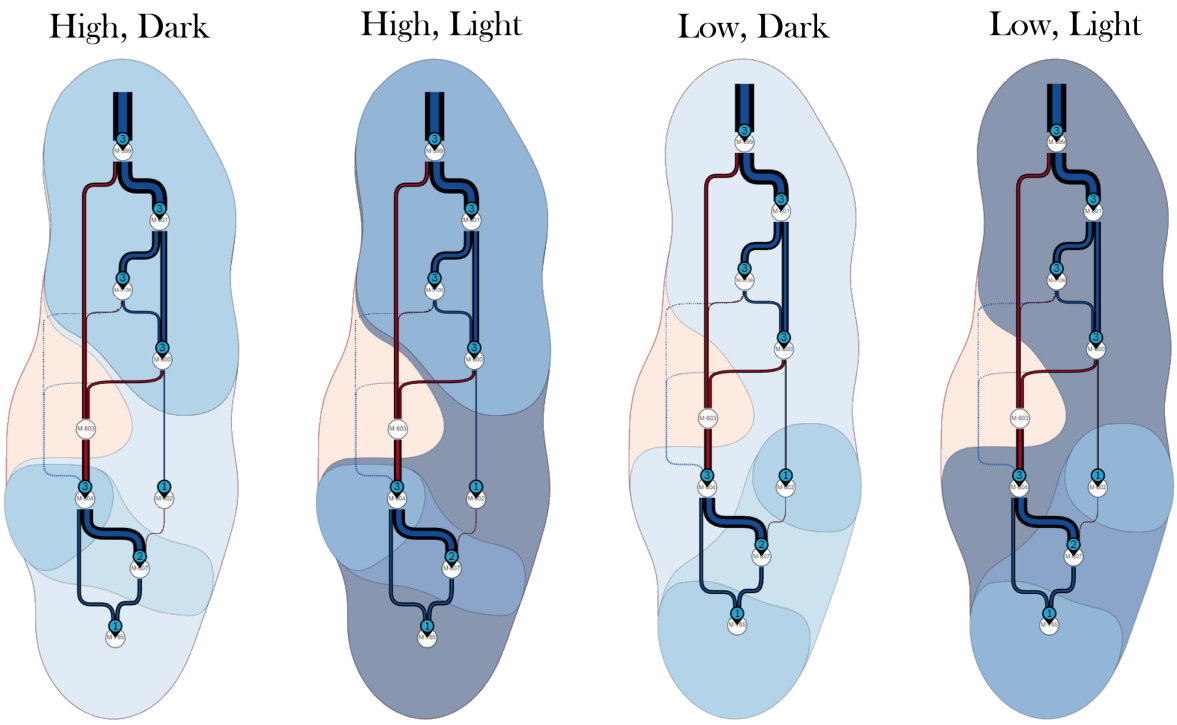


Figure 9.7: Contour Types

26. Color Scale - The color scale controls the direction of the color scale across the contour levels. The first and third graphs of fig 9.7 show cases in which the color scale is set to "Peak

Dark". When in this mode, the peak is a dark color that gets progressively lighter as the contour level decreases. The second and fourth graphs show the alternate option. When set to "Peak Light" the colors are reversed and the higher level contours are drawn with a lighter hue, getting progressively darker as they descend.

27. Advanced Mode - Advanced options were used primarily for development and include:

    - Preventing the use of existing cached components.
    - Changing the flow graph visualization in real time to show the effect of different post-processing operations.
    - Controlling the flow graph orientation (Vertical vs Horizontal).
    - Controlling the minimum score required for a student to master a node.

## 9.3 Data Evaluation

The phased development and evaluation methodology described in the previous section left us confident that our final design was sound. Here we turn our attention to a different, but equally important, evaluation: assessing how teachers used visualizations produced by the tool to make basic decisions for typical situations. This evaluation involved measure both ]] the consistency with which different teachers made decisions about what to teach next for our given situations, and comparing how education researchers familiar with Learning Maps interpreted the visualizations in terms of forming opinions as to whether certain Unit Maps were valid. Interviews conducted in each phase were recorded and later transcribed in full. Using the transcripts, key decisions made by the participants were carefully summarized (appendix F). The data contained in the summary files was then gathered into tables. Among these tables are logistical details such as participant information which includes experience, subject and grade (appendix A) as well as decisions made by each participant in response to interviewer questions. These responses are broken apart by phase in appendix B, appendix C, and appendix D. Additional data, gathered indirectly, was that

pertaining to unprompted recognition of validation. Table A.2 records instances in which the participant offered unprompted observations or comments which demonstrated their recognition of the potential for detection of errors in alignment between the test, learning map or curriculum. For example, a participant stating that they would want to examine the test more closely in order to determine the reason that such a large percentage of users tended to bypass a certain node was taken as evidence that the participant was capable of using the design to detect the existence errors in the original learning map.

To provide context, the tabulated data was then overlaid onto maps of the test students as shown in fig 9.8. Each participant is represented as a person icon and shown adjacent to the nodes they chose as the next skill or concept to teach the corresponding student. In certain cases, a participant was unwilling to choose between two alternatives. In other cases they felt that the best way to proceed was to simultaneously teach the concepts of two different nodes. These cases are marked by yellow arrows between the participant and the nodes in question. Keen readers will also notice that not all nine of the participants interviewed are accounted for in each of the graphs. In some cases, a participant did not provide a clear indication of their next steps and in these instances their icon is omitted from the graph. Decisions about the placement of the icons were made using the following set of rules:

1. In instances where a participant's choice changed after being shown previously hidden node text, the original decision was used. The accuracy of the visualization is based on the random nature of simulated students and, while deemed sufficiently realistic, is not aligned to real learning pathways. While arguably more correct, the decision made using the node text is not a fair assessment of the visualization.

2. In the case of Phase 1, in which the flow and standard layouts were separated, participant decisions made using the standard graph were considered to be overwritten by subsequent decisions made after viewing the flow graph. The reason for this is that many participants, comfortable with the standard graph, were unwilling to switch to the flow graph. As such they attempted to make decisions without benefit of having all the data. Later choices made

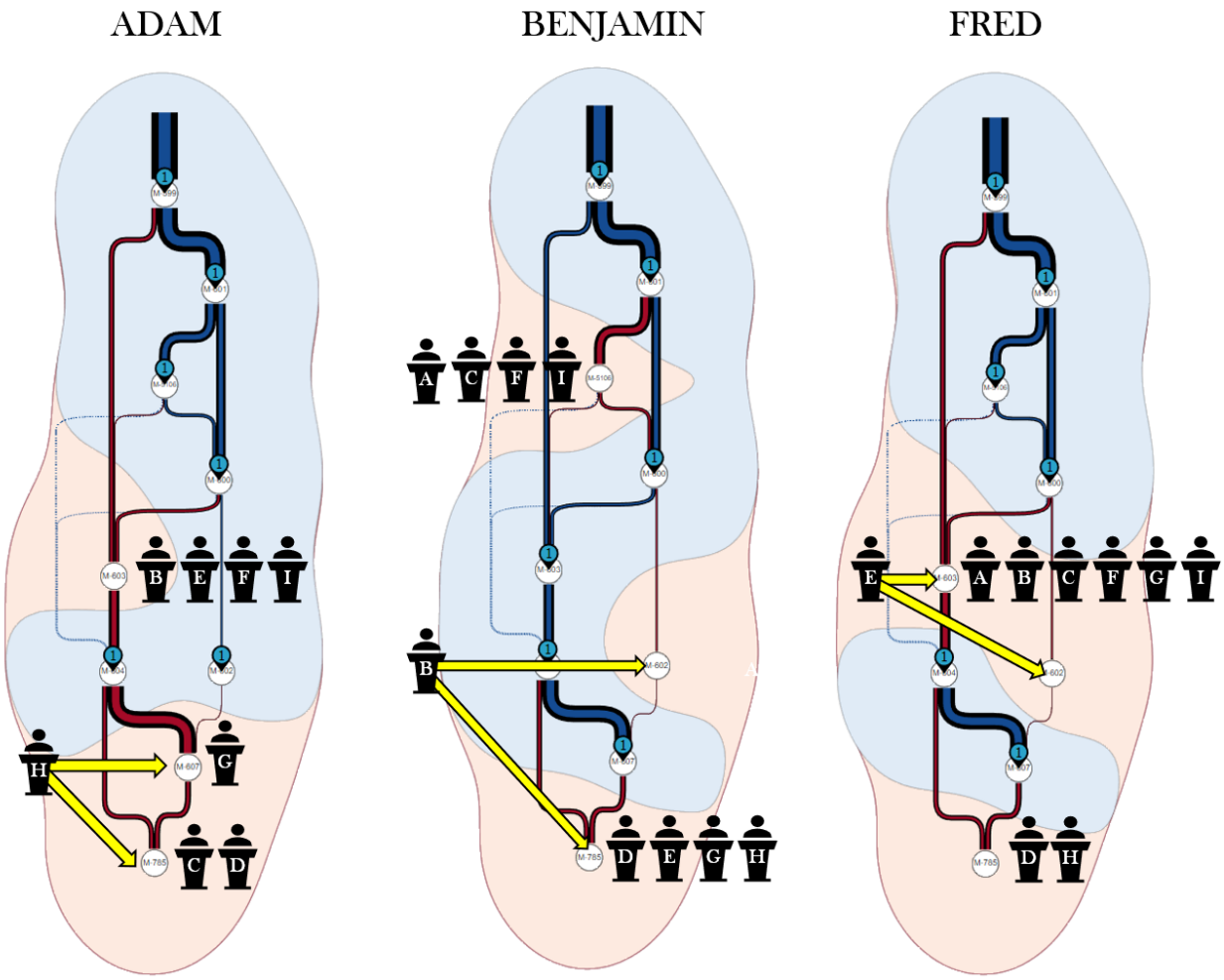


Figure 9.8: Teacher Choices

by participants using all available data are, therefore, given priority over those made before that point by the same participant. Phase 2 and 3 design changes prevent this being a concern following phase 1.

### 9.3.1 Research Goal 1 - Progress Assessment and Prediction

Using gathered data, we were able to draw conclusions regarding the effectiveness of the visualizations in meeting the research goals. Starting with the first research goal, the question was a) whether the final design allowed for rapid assessment of student or class progress with respect to a given learning goal and b) whether the final design provided insight into what should be taught next to further their understanding.

In the case of individual student progress assessment, conclusions regarding the second part of this question provided the evidence needed to answer the first. While not every participant was able to agree on what to teach next, no participant ever expressed a desire to teach a concept (though some said they may review them) already mastered by the student and so falling within the blue envelope. Neither did any participant require correction in their understanding of student progress in conversation or while reasoning about future lessons. All evidence seems to support the claim that use of envelopes in combination with map icons are an effective representation of individual student progress.

In a similar way, the act of participants choosing to teach already mastered topics is evidence that the visualization is able, at minimum, to reduce the choice of options from all nodes to the subset of those not yet mastered. The initial hypothesis going into the interviews was that teachers would always choose to teach the concept farthest down the learning map for which their student had mastered an adjacent concept. It was further expected that when presented with a choice between two paths, the participant would select the path with the higher traffic and therefore greater probability of success. In practice, we observed that, in general, participants tend to fall into two categories. The first category of participants tries first to bridge holes in the student's understanding before progressing to concepts lower in the map. Conversely, those in the second category,

exhibiting greater trust in the concept of multiple pathways, prefer to push forward towards to the target node of the map.

Evidence for this conclusion can be seen in the maps for Benjamin and Adam and is supported by the transcripts in which most participants stated clearly their intent to either go back and fill in gaps or push forward. The graph for Fred demonstrates a deviation from this categorization in that participants in the second category, when confronted with a situation in which no discernible path juspngied the progress of the student, chose to backtrack rather than push forward.

While the first part of the hypothesis regarding the specific choice of node proved to be flawed, the second held true. Participants, when presented with a choice between two paths, would select the path with the greater probability of success. Evidence for this is most clearly provided in fig 9.4. Deprived on the relative measure of probability between paths, participants fell back on their past experience and, in extreme cases, a coin toss. In contrast, when those same participants were shown the flow graph they all made the same decision in a fraction of the time. Though it is difficult to provide a qualitative measure, it seems clear that the variable edge widths defined by the flow graph are an effective means of displaying typical learning patterns and, when used in combination with the envelopes and map markers to indicate student progress, become an intuitive way of informing decisions about next steps.

The tools provided for Phase 1 interviews allowed for a limited view of classroom progress by using the map icons to display the number of students having mastered each node. Despite such a poor visual indicator, participants were universally delighted by the feature and expressed a number of unforeseen uses including grouping students for classroom exercises and assessing weaknesses in the curriculum based on the position of low numbered markers. In response to repeated requests by participants, Phase 2 included a feature in which a user hovering over a map icon would cause the names of all selected students having mastered the associated node to glow. Again, the response was overwhelmingly positive, with participants eager to contribute ways in which it could be improved further. Phase 2 also included a change in which red envelopes were added around misunderstood nodes. This prompted discussions both during and apart from interviews about

cases in which the envelopes of two students were laid one atop the other. The contours presented in chapter 8 and added to the interview tool in Phase 3 were the natural extension of the concept.

Being added to the interview tool in this fragmented manner means that results from participants were only sought directly in the later interview stages and so limited data exists. Participants were asked a) what they would teach next to the Grade 3 classroom and b) what they perceived to be the weakness in the curriculum or teaching for the classroom. Responses to these questions can be found in table E.1. The tabled results have been projected onto the graphs in fig 9.9. Limited data means limited conclusions, but participants again appear to fall into the categories of those that go back to bridge gaps in understanding and those that push forward. Interestingly, it also seems that a participant can appear in different categories when making decisions regarding individual students opposed to groups of students. While it is clear that the combined contour map and icon visualization meet the requirements of the first research goal in providing a means of visualizing classroom progress, given the much higher number of variables considered by participants, its unclear whether it will scale well given larger maps or more students.

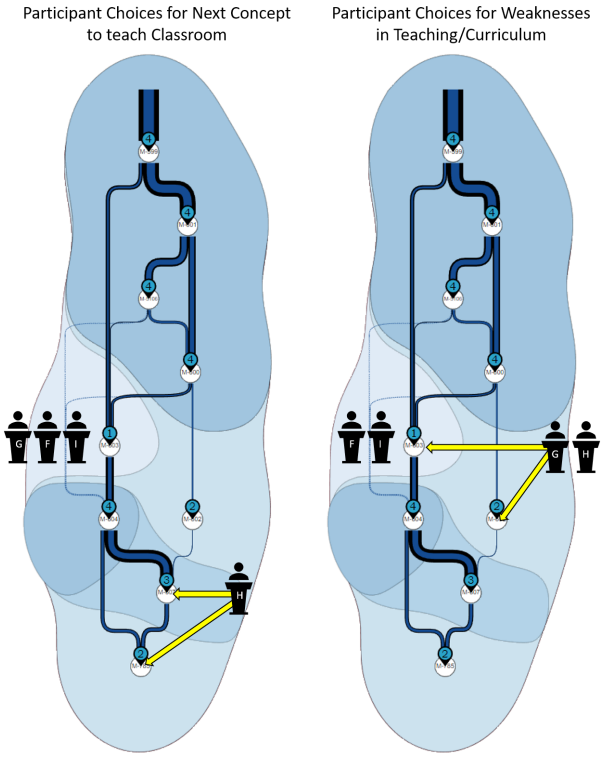


Figure 9.9: Classroom Assessment

### 9.3.2 Research Goal 2 - Validation

To meet the requirements of the second research goal, the visualization needed to provide indications of where student learning patterns differed from those predicted by the original learning maps. Throughout the project, validation has always been a goal with a very loose definition. Our education sponsors, while able to explain how a learning map may be invalid, were never able to provide a clear indication of how they felt such information should be presented. Over time it became clear that the goal was not to provide a detailed explanation of the precise errors, which would have been well beyond the scope of this research, but to make clear that such errors did in fact exist.

Even this much proved challenging. The most effective tool for performing validation was found to be the bypass edges introduced in section 4.4 and later refined in chapter 6. When presented with a flow graph that had been derived from a purposefully flawed learning map, reviewers were able to recognize the presence of flaws. In some cases reviewers could even provide counter explanations as to why the student test data might not follow the learning map due to flaws in the curriculum. What stood out during the work on validation is the shift in mindset required before an individual can recognize it.

In section 4.7 a design was presented to a group of education researchers and we found that most, despite being told that validation was the goal of the visualization, immediately went into a mindset aimed towards using the visualization for prediction. As they became more comfortable with the design, they would eventually make the connection to validation, after which they were able to recognize and reason about it. This pattern was also frequently encountered during interviews. A participant, already having demonstrated a thorough understanding of the entire design, when asked whether they felt able to point to possible flaws in the learning map would be unable to do so. In many cases that same participant would reach a point while working through a different operation in which they questioned the reasoning behind certain edges. Once at that point, the participant would be able to point to the potential flaws encountered later in similar situations. Additional support for validation in the final design can be drawn from table A.2 which details

observations made during interviews and summarizing instances where participants verbally expressed their recognition of the design as a tool for validation.

The final visualization design was able to simultaneously meet the requirements for validation, progress assessment, and prediction. The cost for doing so may well have been the tendency for users to get stuck in the "teacher" mindset and require a conscious shift before being able to make use of the design for validation. Once in that mindset, the majority of reviewers had little difficulty in detecting flaws in the learning map design in a manner consistent with expectations.

## 9.4 Evaluation of Design Elements

A common problem with developing software is the tendency for scope creep. Rather than first idenpngying the best solution to meet a fixed set of requirements, features are added in response to changing requirements, often in ways that integrate poorly or conflict outright with existing components. Our goal was to provide a targeted solution which performed the tasks it was designed for in the most effective way possible. Our final design incorporate different visual mechanics and, while some excel at meeting a specific requirement or are genuinely irreplaceable, the functionality of most could, often with extreme effort, be done using other components. However, we feel that when combined they form a set of tools which any user can quickly understand and apply, along with an interface that guides users intuitively to the best way to achieve their goals. To this end we close by briefly considering each of the visual mechanics present by Phase 3 and provide evidence for their inclusion in the final design.

**Flow Graph:** The key attributes of the flow graph are the representations of typical learning paths described by the variable edge widths and the inclusion of bypass edges. Evidence strongly suggests (fig 9.4) that in the absence of some way to break ties between nodes of apparently equal importance, users will first try to fall back on prior experience and then simple chance to make decisions. In contrast, when provided with an indication of which of the nodes were more commonly chosen by previous students, they were able to make rapid, confident decisions absent the series of second guesses that previously accompanied the choice. Bypass edges, which indicate the devia-

tion of students from expected patterns, are one of the only viable ways we found of facilitating validation of the learning map, test, and curriculum in a way that can be readily understood by those outside the field of Psychometrics. Since the information shown by the edges is derived from the distribution of student mastery, it is conceivable that map icons and envelopes could be used to gain access to the information encoded in edge width. However, the process for doing so visually is arduous and scales extremely poorly with larger numbers of students, making the flow graph a necessary component of the final design.

**Map Icons:** Icons, while redundant when used with envelopes for individual students, have proven to be critical when assessing the understanding of a group of students. In those cases, the map icons provide a means of labeling the level of each contour. While such indication of level could also be done using a single label within the contour's envelope, the appearance of the map icon in many widely used applications makes them a comfortable starting point for participants who have been conditioned to recognize the shape as marking a position in space.

**Envelopes:** Envelopes as they relate to individual students are designed to aide users in grouping. Phase 1 and 2 interviews, in which icons could initially be shown without accompanying envelopes, showed that even when colored, users struggled to keep track of which nodes were mastered, needing to examine each node individually then try to mentally maintain the grouping while moving their focus to the edges. Envelopes solve this problem by providing a clear visual grouping of the nodes for which a student has mastery, making them overwhelmingly superior to the alternative.

**Contours:** Contours are an extension of envelopes used for individual students and so could similarly be replaced by icons. Indeed, before Phase 3 icons were the only way of obtaining information about groups of students. However, no participant in Phase 3, after being shown the contour map, felt that the task could be better done with the icons alone. The problems faced grouping the mastery of a single student were magnified exponentially for a set of students, making anything more complex than finding the maximum or minimum number a major undertaking when using icons alone. By grouping nodes into layers and linking the levels of a contour map to the over-

all group mastery, contours provides a way for users to instantly perform such simple operations, before moving on to analyze patterns between groups.

# Chapter 10

# Conclusions

The purpose of this research was to address the two research goals defined in section 1.1. During synthesis of the final design in chapter 5 we further broke apart the first research goal pertaining to progress assessment of students into a set of fundamental operations of which a solution to the research goal must be capable. This set of operations, combined with the second research goal involving validation of the original learning map, provided the basis for development of the final design. Using observations and reviewer feedback gathered from the earlier designs in sections 4.1, 4.2, 4.5, 4.6, and 4.7, a final design was synthesized using those visual strategies found most effective at performing the required fundamental operations.

Chapter 9 details the interview process used to refine the interview tools while also gathering data to evaluate the effectiveness of the final design. As explained in section 9.3.1, evidence strongly suggests that the final design was able to fully meet the requirements of the first research goal, providing mechanisms for assessing student and classroom progress while informing future lesson planning. Designing a visualization to meet the requirements of the second research goal proved far simpler than determining a means of evaluating it. Section 9.3.2 explains the difficulties faced in communicating the relevance of visualization components to validation. Despite the difficulty in overcoming the initial learning curve, evidence collected through interviews and interaction with reviewers also indicates that the final design was able to meet the criteria described by the second research goal.

Additional contributions made to the field in response to gaps in prior art include a method for simulating realistic student response data for tests aligned to a given learning map (Chapter 3), a graph layout algorithm designed for weighted directed graphs employing variable width edges

with ranked pathways (Chapter 6), an approach to wrapping certain nodes of a graph in envelopes to promote grouping (Chapter 7) and an extension of this envelope algorithm to overlay a contour map onto a graph in order to compare attributes between groups of students (Chapter 8).

# Chapter 11

# Future Work

- Explore the possibility of extrapolating information onto dashed edges of the Enhanced Learning Maps map views.
- The layout algorithm presented in chapter 6 is the bare minimum required to achieve our goals. There are many ways it could be improved in future work. A few thoughts on these are listed below:
  - Instead of removing duplicate nodes after placement, affix duplicate nodes together as a rigid body using them to forcibly keep the space below the nodes free during application of the physics system.
  - Add arcs prior to application of physics system and then treat the arc control points similar to nodes. Decisions would need to made about whether control point forces could affect nodes and vice versa but such an approach may result in a reduction in edge crossings.
- Additional features for the interview tool requested by participants.
  - Hovering over an edge highlights students who have mastered the edge.
  - Hovering over a node highlights students who have mastered the node as well as highlighting all edges into and out of the node.
  - To avoid large classrooms forcing the filter buttons off the bottom of the screen, move them to a separate pane.

  - Hide the Flow Layout and Flow Direction Advanced Options when the Standard Graph is Selected.
  - Hide the Color Type and Color Scale Options when the Standard Graph is Selected.

- Investigate extension of the visualization to fields such as marketing or network analysis.

# References


Barth, W., Jünger, M., & Mutzel, P. (2002). Simple and efficient bilayer cross counting. In M. T. Goodrich & S. G. Kobourov (Eds.), *Graph Drawing* (pp. 130–141). Berlin, Heidelberg: Springer Berlin Heidelberg.

Bertin, J. t. b. W. J. B. (2011). *Semiology of graphics : diagrams, networks, maps*. Redlands, Calif: ESRI Press Distributed by Ingram Publisher Services.

Borland, D. & Ii, R. M. T. (2007). Rainbow color map (still) considered harmful. *IEEE Computer Graphics and Applications*, 27(2), 14–17.

Brandes, U. & Köpf, B. (2001). Fast and simple horizontal coordinate assignment.

Broaddus, A., Sharma, A., & Adjei, S. (2015). Using test responses to validate a learning map of integer understanding. `https://nctm.confex.com/nctm/2015RP/webprogram/Manuscript/Paper2181/BroaddusSharmaAdjei_LM-Integer_NCTM%202015.pdf`.

Burch, M., Schmidt, B., & Weiskopf, D. (2013). A matrix-based visualization for exploring dynamic compound digraphs. In *Proceedings of the International Conference on Information Visualisation* (pp. 66–73).

Didimo, W., Montecchiani, F., Pallas, E., & Tollis, I. (2014). How to visualize directed graphs: A user study. In *IISA 2014 - 5th International Conference on Information, Intelligence, Systems and Applications* (pp. 152–157).

Eades, P. (1984). A heuristic for graph drawing. *Congressus Numerantium*, 42, 149–160.

Eades, P. & Tamassia, R. (1988). *Algorithms for Drawing Graphs: An Annotated Bibliography*. Technical report, Providence, RI, USA.

Gansner, E. R., Koutsofios, E., North, S. C., & Vo, K. . (1993). A technique for drawing directed graphs. *IEEE Transactions on Software Engineering*, 19(3), 214–230.

Ghoniem, M., Fekete, J., & Castagliola, P. (2004). A comparison of the readability of graphs using node-link and matrix-based representations. In *IEEE Symposium on Information Visualization* (pp. 17–24).

Harrower, M. & Brewer, C. A. (2011). *ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps*, chapter 3.8, (pp. 261–268). Wiley-Blackwell.

Horak, T., Kister, U., & Dachselt, R. (2018). Comparing rendering performance of common web technologies for large graphs. In *Poster Program of the 2018 IEEE VIS Conference*, VIS '18.

Hu, Y. (2006). Efficient and high quality force-directed graph drawing. *Mathematica Journal*, 10, 37–71.

Jünger, M. & Mutzel, P. (1997). 2-layer straightline crossing minimization: Performance of exact and heuristic algorithms. 1.

Kingston, N. M. & Broaddus, A. (2017). The use of learning map systems to support the formative assessment in mathematics. *Education Sciences*, 7(1).

Mamassian, P. & de Montalembert, M. (2010). A simple model of the vertical–horizontal illusion. *Vision Research*, 50(10), 956 – 962.

Purchase, H. C., Cohen, R. F., & James, M. I. (1997). An experimental study of the basis for graph drawing algorithms. *J. Exp. Algorithmics*, 2.

Riehmann, P., Hanfler, M., & Froehlich, B. (2005). Interactive sankey diagrams. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.* (pp. 233–240).

Sugiyama, K., Tagawa, S., & Toda, M. (1981). Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2), 109–125.

von Landesberger, T., Kuijper, A., Schreck, T., Kohlhammer, J., van Wijk, J., Fekete, J.-D., & Fellner, D. (2011). Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum*, 30(6), 1719–1749.

Zarate, D. C., Bodic, P. L., Dwyer, T., Gange, G., & Stuckey, P. (2018). Optimal sankey diagrams via integer programming. In *2018 IEEE Pacific Visualization Symposium (PacificVis)* (pp. 135–139).

# Appendix A

# Participant Data

Table A.1: Participant Backgrounds

| | Experience | | | |
|---|---|---|---|---|
| | Teaching | Research | Grade(s) | Subject(s) |
| A | Y | | 3rd,4th,5th Grade | All |
| B | Y | | 2nd Grade | All |
| C | Y | | | |
| D | Y | | | ELA |
| E | Y | Y | | Math |
| F | Y | | | Math |
| G | Y | Y | | ELA |
| H | Y | Y | | ELA and Math |
| I | Y | Y | | Math |

Table A.2: Validation Inference

| | Experience | | Expressed Recognition of Use in Validating | | |
|---|---|---|---|---|---|
| | Teaching | Research | Map | Test | Curriculum |
| A | Y | | | | Y |
| B | Y | | | | |
| C | Y | | Y | Y | Y |
| D | Y | | | | |
| E | Y | Y | | | |
| F | Y | | | | Y |
| G | Y | Y | Y | | |
| H | Y | Y | Y | | Y |
| I | Y | Y | ? | ? | ? |

# Appendix B

# Phase 1 Data

Table B.1: Phase 1 Choices

| | Benjamin | | | |
|---|---|---|---|---|
| | Icons | +Envelope | Flow | **Final** |
| A | N/A | M-602 & M-5106 | M-5106 | **M-5106** |
| B | M-602 & M-785 | M-602 & M-785 | N/A | **M-602 & M-785** |
| C | N/A | M-5106 | N/A | **M-5106** |
| D | M-602 | M-785 | N/A | **M-785** |

Table B.2: Phase 1 Choices

| | Fred | | | |
|---|---|---|---|---|
| | Icons | +Envelope | Flow | **Final** |
| A | N/A | M-603 | N/A | **M-603** |
| B | N/A | M-602 | M-603 | **M-602 or M-603** |
| C | N/A | M-603 | M-603 | **M-603** |
| D | N/A | M-785 | M-603 | **M-785** |

Table B.3: Phase 1 Choices

| | Adam | | | |
|---|---|---|---|---|
| | Icons | +Envelope | Flow | **Final** |
| A | N/A | N/A | N/A | **N/A** |
| B | N/A | M-604 | M-603 | **M-603** |
| C | N/A | M-603 | M-603 | **M-607 or M-785** |
| D | N/A | M-607 | M-785 | **M-785** |

# Appendix C

# Phase 2 Data

Table C.1: Phase 2 Choices

| | Benjamin | | Adam | | Fred | |
|---|---|---|---|---|---|---|
| | No Text | Text | No Text | Text | No Text | Text |
| E | M-785 | N/A | M-603 | M-603 | N/A | M-603 or M-602 |
| F | M-5106 | M-5106 | M-603 | M-607 | M-603 | M-602 |

# Appendix D

# Phase 3 Data

Table D.1: Phase 3 Choices

| | Benjamin | | Adam | | Fred | |
|---|---|---|---|---|---|---|
| | No Text | Text | No Text | Text | No Text | Text |
| G | M-785 | N/A | M-607 | M-607 | M-603 | N/A |
| H | M-785 | N/A | M-785 or M-607 | M-603 or M-607 or M-785 | M-785 | M-785 |
| I | M-5106 | M-5106 | M-603 | M-603 | M-603 | M-603 |

# Appendix E

# Classroom Progress Assessment

Table E.1: Classroom Assessment

| | Teach Next | Perceived Weakness in Curriculum/Teaching |
|---|---|---|
| F | M-603 | M-603 |
| G | M-603 | M-603 and M-602 |
| H | M-607 and M-785 | M-603 and M-602 |
| I | M-603 | M-603 |

# Appendix F

# Interview Transcript Summaries

## F.1 Participant A Summary

**Background:** 3rd,4th,5th Grade All Subjects

**Edge Width Understanding:** Able to understand flow, able to understand the use in validating curriculum.

**Requests:** Mechanism for manually assigning student mastery.

**Benjamin:**

First showed icons then envelopes. Participant claimed they would help Benjamin make the connection between area and unit square (M-5106) before using the real-world problems(M-785) to assess whether the teaching was successful. They also claimed that the the top two nodes missed by the student (M-5106,M-602) presented opportunities for them to use manipulatives to help them make those two connections. Then they planned to take all those manipulatives and apply them to a real world problem (M-785). When asked if the map supported their choice they explained that it did and also raised an interesting point: if the bottom node was mastered but not all precursors were mastered, they said it was possible that they would push on, making a note to return and address the precursors at a later date, possibly in a different subject or context. When asked which precursor they would teach next they switched to the flow chart then stated that, because fewer students understood "calculating the area of a rectangle with tiling" (M-602) that:

1. if many students don't understand it then maybe there needs to be more emphasis on it and
2. explaining area of composition of unit squares seems to be a path that helps students progress.

Given this they chose M-5106 as the next node.

**Fred:**

Using the standard graph, the participant first showed icons then envelopes. They said that if Fred had mastered M-785 they would move on but make M-603 a part of his daily work until he had mastered it. Since he had not mastered M-785 they said they would focus next on M-603.

## F.2 Participant B Summary

**Background:** 2nd Grade – All subjects

**Source of Data:** Relied a lot on personal knowledge of the nodes.

**Edge Width Understanding:** Unprompted and prompted evidence of flow mechanics as used in tracking student mastery.

**Requests:** Mechanism for viewing the entire class before subsequently breaking down the results on a student-by-student basis.

**Benjamin:**

When using icons alone they chose to teach M-785 and a little of M-602. Participant did not feel that they needed M-5106. Choices did not change with the addition of envelopes.

**Adam:**

When using the standard graph the participant used envelopes alone, entirely bypassing icons. Chose M-603 and M-604 followed by M-607. Their goal was to convey understanding of the formula (M-604) before moving on to irregular shapes (M-607), since tiling was believed insufficient to model things like triangles.

When using the flow graph they selected M-603 as the next node to teach. They recognized that they could theoretically go from M-602 to M-785 but preferred to stick to the "thick" paths.

**Fred:**

Participant ignored icons when using the standard graph and used only envelopes. Their initial thought was to teach M-785. When asked why they would skip M-602 and M-603, they revised

the answer. When asked whether they would teach M-602 or M-603 they said M-602 based on experience with the node content and because Fred had demonstrated mastery of M-604.

When using the flow graph they chose to teach M-603 first since "most students went that way". As a result they felt that it would be easier for the class to keep moving. This appears to disagree with their previous response but does not take into account Fred's mastery of 604 as a condition which is unlikely to be shared by the rest of the class.

## F.3 Participant C Summary

**Background:** Teaching

**Source of Data:** Experience and Graphs Evenly Split

**Edge Width Understanding:** Unprompted & prompted understanding of flow mechanics. Unprompted understanding of use in validation of map and test.

**Requests:** Map configuration based on Depth of Knowledge (DoK) principles. **Issues:** Participant struggled to gain initial understanding of flow graph. They were very uncomfortable with the discrepancy between flow graph and standard map and claimed that being a linear thinker ofen made the organic nature of the maps hard to use intuitively.

**Benjamin:**

Participant ignored icons, choosing to first open envelopes. Chose M-5106. Their reasoning was that misunderstanding M-5016 may well be the reason that the student hadn't yet mastered M-602. Rather than waste time trying to teach M-602, only to backtrack later, they felt they would rather just start at M-5106.

**Fred:**

Participant started with the standard graph and again ignored icons. Initially chose M-603. When asked why they chose M-603 instead of M-602 they stated that more edges entered M-603 making it the better choice.

When switching to the flow graph the participant initially misread the flow diagram, thinking that the width represented the number of students taking the associated assessment. After explaining

the flow graph again they chose M-603 as a third of those mastering M-603 had also mastered M-607.

**Adam:**

Participant started with the flow graph on and initially claimed that Adam had mastered everything, demonstrating further misunderstanding. Explaining the flow graph at this point made them cognizant of its use in validation, which they found extremely interesting. After turning on the icons they asked what the yellow lines meant. They immediately chose M-785 as the target.

Switching to the standard graph caused a lot of problems. The participant struggled with the missing connections in the flow graph, eventually stating that the answer to what to teach would depend on the graph being used. They claimed that using the standard graph, they would choose M-607 and using the flow graph they would choose M-785, pushing as far down in the graph as it would allow them. If they had to choose which one they would trust, they claimed they would use the standard graph. After explaining that the standard map is our best guess at what the progressions are, while the flow diagram is based on the data given to us by the students they quickly raised questions about whether M-607 was a true precursor or whether there had been a problem in assessing the mastery, moving the discussion away from the question and into validation.

## F.4 Participant D Summary

**Background:** Teacher, ELA.

**Source of Data:** Visualization. No prior knowledge of node concepts.

**Edge Width Understanding:** Demonstrated understanding of flow mechanics and how they related to the path most traveled by students.

**Requests:** Red envelops around misunderstood nodes to indicate a failure to understand. Red flow paths to indicate a failure to travel.

**Issues:** Participant believed that unmarked nodes were those that had not yet been tested for the given student. Assumed that either the student had easily understood them or had not yet been taught the concepts. This only became clear at the end of the interview. I believe this also led to a

misunderstanding of the dashed lines.

**Benjamin:**

Started with icons on. Participant initially stated that they would teach M-602 before moving forward. When asked why they would teach M-602 and not M-5106 they were surprised and said they hadn't seen M-5106. After a short pause I turned on the envelopes and the participant then claimed that they would teach M-785 next given that there was already a path through the map taken by the student. Their reasoning was that even though it is not the most direct route, it is still a viable alternative.

**Adam:**

When using the standard graph the participant relied on icons and envelopes, they said they would teach M-607 as the next skill using reasoning similar to that used for Benjamin.

After turning on the flow diagram the participant said they would instead choose to go directly through to M-785 as the pathway existed to do so. At this point they defended their answer by saying that, while they realized that the majority of students traveled through M-603, they felt that Adam had found a way he was comfortable with and should continue that way if possible. This was confirmed by asking what they would teach if the student failed to learn M-785. They responded immediately with M-603.

**Fred:**

When using the standard graph the participant said they would likely teach M-785 while doing tests to investigate the large gap in the middle of the graph. When told they could only teach M-602 or M-603 and asked which they would choose, they were unable to answer. So much so that they eventually chose M-602 based on it being numerically smaller than M-603.

After turning on the flow graph they immediately selected M-602 stating that, while they recognized the majority went through M-603, the student had already demonstrated mastery on that path by learning the nodes following M-603 but had not yet demonstrated mastery on the left hand path.

## F.5 Participant E Summary

**Background:** Teacher, Math. Research Experience.

**Source of Data:** Prior experience and visualization. Fairly even split.

**Edge Width Understanding:** Understood how students flowed through the map.

**Issues:** Interview interrupted by maintenance in the interview room. Prior (and different) use of dashed lines seemed to prevent recognizing their value in terms of validation.

**Benjamin:**

Participant said they would likely bypass M-5106 (C) depending on what it was. Given the progress on the alternate route they were likely to choose M-785 (I) as the next concept to teach.

**Adam:**

Chose to teach M-603. Initially the choice was made using prior knowledge of the node text but when pressed they were able to work through the visualization without the text and determine that, even without the text, they would still choose M-603 given its relative width.

**Fred:**

Chose to start with a review M-600 as both connections leading from it were missed. They then said they would go to M-603 or M-602 depending on the content. They were not comfortable considering teaching M-785 as they didn't feel that the student had a complete understanding of the material required to progress. This opinion was based on prior knowledge of the missed nodes.

## F.6 Participant F Summary

**Background:** Teacher, Math

**Edge Width Understanding:** Curriculum validation, Flow mechanics.

**Requests:** Mechanism to set thickness of lines to reflect only their class(es)

**Benjamin:**

Started by using icons and info. Initially chose M-5106 as they felt Benjamin's depth of understanding may not be sufficient to continue, despite having a path through M-603. They also stated

that this was only if they were working individually. If working with the whole class, they said they would likely push on to M-785. Showing them the text only made them more determined to teach M-5106.

**Adam:**

Turned on icons and info. Participant initially chose M-603 (E) with the hope that understanding that may help unlock the remaining un-mastered nodes. They weren't confident that he understood M-604 (G) without having mastered M-603. When given the node text, they changed their opinion and instead claimed they would perhaps try to build on M-602 (tiling) and follow the thinner path. They felt that doing so would help them fill in M-603 at the same time.

**Fred:**

Chose M-603 (E) and M-602 (F). They claimed that because he wasn't "solid" on either concept, that they would be very hesitant going forward. They would still build on tiling (M-602) as they did with Adam and then use that go down the thinner path. When asked whether they would teach M-603 or M-602 based solely on the graph, they said they would go with M-603 given the flow traffic.

**Grade 3 Math 2018:**

Initially said M-607 given that most of the class was already there but switched to M-603 when they realized only one student had mastered it.

**Grade 3 Math Weakness in Teaching:**

M-603

## F.7 Participant G Summary

**Background:** Teacher, ELA. Research Experience.

**Edge Width Understanding:** Able to use the flow graph to track student progress.

**Notes:** Participant really struggled to make choices about what to teach next and continually second guessed previous choices.

**Benjamin:** Chose M-785. They recognized that Benjamin had found a second (non-mainstream)

path so they would keep going forward. Noted repeatedly that they were concerned because the majority of students went a different path.

**Adam:** Confusion about the area around E (M-603) as there is a blue line through a red area. Became much happier when they realized it was an anomaly. After turning on the text, they decided that they would teach H (M-607) using the same logic as before: push onward. When the text was removed and they were asked again, they determined that they may have said E (M-603) but after thinking it through they would probably have gone with H (M-607) anyway.

**Fred** Saw a lot of parallels to Adam but was much more concerned with the gap between the blue areas. Eventually decided to teach E (M-603) or F (M-602) and then selected E (M-603) given the higher flow traffic.

**Grade 3 Math 2018** Teach E (M-603) and F (M-602) next and would prioritize E (M-603) given that it had one fewer student with mastery.

**Grade 3 Math Weakness in Teaching:** E (M-603) and F (M-602)

## F.8 Participant H Summary

**Background:** Teacher, Math and ELA. Research Experience.

**Issues:** Expressed concern about the double lines. Found them very hard to differentiate from the solid lines. At one point confused a solid line for a double line. On the other hand they also recognized, when shown an earlier version of the tool, that they would have had difficulty interpreting the dashed lines correctly.

**Edge Width Understanding:** No evidence that they used the proportions to make decision about future planning. However, they were able to recognize the use of the bypass lines in map validation.

**Requests:** Mechanism to view node description (a longer passage that more fully describes the node)

**Note:** Preferred the contour to the old envelope view for individual students when given a side by side comparison.

**Benjamin:**

Initially chose M-5106. When pushed, they claimed that they planned to teach them all so they would start with the one at the top. When told they could only teach one, then they immediately switched to M-785, since Benjamin had followed a pathway down to it but had not yet mastered it. They felt that the other nodes could be worked around using alternate pathways but that not knowing M-785 would prevent him from moving on to other things.

**Adam:**

Mistook the line from M-600 to M-602 as an anomaly. Once they realized it wasn't, then they struggled to choose between M-785 and M-607. Turning on the text caused them to revise their opinion and also led them to more clearly understand the bypass lines. After a while they settled on teaching M-603 if they wanted the student to actually understand what was happening for later lessons, otherwise M-607 or M-785.

**Fred:**

Participant struggled to choose. Claimed they wouldn't teach E (M-603) because Fred had already mastered the two right after E. **Note:** This is pretty major. Most people struggle with the fact that E (M-603) and F (M-602) are equidistant from D (M-600) so don't really have a basis to choose the better. This participant looked further down and noted that G (M-604) and H (M-607) are both decedents of E (M-603) while F (M-602) only has H (M-607) as a decedent. They eventually settled on I (M-785) since it's the end goal but mentioned again that they would teach all if they could. The participant did not seem to have used the flow graph in any meaningful way.

**Grade 3 Math:**

Their initial reaction was to view darker blue as areas that everyone had mastered. They decided that they would teach a little of I (M-785) and H (M-607). "A brief review of H to get the last student caught up and then on to I (M-785)".

**Grade 3 Math Weakness in Teaching:**

E (M-603) and F (M-602). "Without a doubt."

## F.9 Participant I Summary

**Background:** Teacher, Math. Research Experience.

**Edge Width Understanding:** Able to use the flow graph to track students. Able to validate the learning map.

**Requests:** Make clicking on the student name toggle the icon. Return to solid lines instead of lines with black edges. The participant struggled with the edged lines as they felt the width to be significant.

**Notes:** Participant used icons in multi-student view to pair students in the hope that by pairing stronger and weaker students they would both benefit.

**Benjamin:**

Chose to teach C (M-5106) with the understanding that it is so far back that it's a concern that the student hasn't already figured it out. While they also said they would teach F (M-602), they claimed to be less concerned as the student appeared to be around that area in terms of current learning and would likely pick up on the concept soon. A unique point they raised is that, while there is an alternate path through the current map there are other maps that contain C (M-5106) and by not learning it now could easily affect the student's ability to progress later. Seeing the text made the participant more certain of their choice.

**Adam:**

E (M-603). The act of skipping E (M-603) raised a red flag for the participant who felt they should form a connection between D (M-600) and G (M-604)

**Fred:**

The gap made the participant feel that the student had not, in fact, achieved complete mastery of G (M-604) and H (M-607). They stated that their next step would be to go back and teach C (M-5106) and D (M-600) before moving through E (M-603) and F (M-602) to G (M-604) and H (M-607). The idea being to build connections through the middle. When asked to choose E (M-603) or F (M-602) using the visualization they said they would choose E (M-603) as the path through it was clearly thicker and therefore stronger.

**Grade 3 Math 2018:** Teach E (M-603) next. Their reasoning was: "I've got to focus on the thing that more students haven't mastered and seems to be more critical". Since most students hadn't mastered E (M-603) that was their next step.

**Grade 3 Math Weakness in Teaching:** E (603)

# Appendix G

# Interview Outline

1. Connect using KU Zoom Client
2. Display oral consent form on shared screen. Read through and answer questions.
3. Inquire after participant background information (e.g. Grade and Subject).
4. Switch display to show interview tool.
5. Explain classroom box
6. Demonstrate and explain icon button.
7. Demonstrate and explain envelope button (when applicable).
8. Explain how to switch map type when applicable.
9. Answer participant questions.
10. (Phase 1 Only) Switch view back to standard graph.
11. Deselect all students.
12. Set up screen sharing so that the participant is able to operate the mouse from their device. In the case where this is not possible (e.g. SmartPhone) instruct the participant to clearly specify any manipulation of the tool in future operations.
13. Ask participant to determine the next node/concept to teach to Benjamin.

14. Ask participant to explain the reasoning behind their decision. If the participant says they would focus on more than one node then accept the answer and follow-up by asking them to select only one and explain their reasoning. This type of situation is common in cases when the teacher wants to go back and fill in gaps rather than go forward as there are two possible gaps.

15. Ask participant to determine the next node/concept to teach to Adam. If short on time then proceed instead to Fred.

    - **Phase 1**

        (a) Adam is unique in that there are only two nodes (M-603, and M-607) that appear viable given the envelope view. Once the participant has come to that realization ask them to choose one.

        (b) After they have either answered the question or some time has passed without any indication of a choice switch to the flow diagram.

        (c) Ask the participant to idenpngy using the flow diagram the next to teach.

        (d) Due to the flawed flow graph the participant will likely choose either the target node (M-785) or M-603. Regardless of the choice ask for an explanation.

    - **Phase 2**

        – Repeat 14 for Adam

16. Ask participant to determine the next node/concept to teach to Fred.

17. Fred is a difficult choice as the participant is forced to make a decision to either push forward to the target node or go back and fill in gaps. Ask participant to explain decision.

18. Ask participant whether they would choose to teach M-603 or M-602 given the visualization.

    - **Phase 1**

(a) After they have either answered the question or some time has passed without any indication of a choice switch to the flow diagram.

(b) Ask the participant to idenpngy using the flow diagram whether they would choose to teach M-603 or M-602 given the visualization. **Note:** The flaw in the flow graph will not impact the results of this step but seeing the differences between the graph may help prompt understanding of validation or of the data-driven nature of the flow graph in general.

19. **Phase 2 and 3 Only** - Ask participant what they would teach "Grade 3 Math 2018". Ask participant to explain reasoning.

20. **Phase 2 and 3 Only** - Ask participant to idenpngy the weaknesses of the teaching in "Grade 3 Math 2018".

21. Ask participant to idenpngy any features that they felt were misleading or unclear. Also any features and changes they felt would improve the visualization.

22. Ask participant to idenpngy any features that they felt were especially useful or interesting.