Interactive Multimedia in Digital Courses: Design and Evaluation of Concept Maps Glossary and Narration Support

Ehsan S. Elgendi

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Science and Applications

Clifford A. Shaffer, Chair

Edward A. Fox

Mohammed Seyam

May 13, 2019

Blacksburg, Virginia

Keywords: Computer Science Education, Concept Maps, Narration, Multimedia

Interactive Multimedia in Digital Courses: Design and Evaluation of Concept Maps Glossary and Narration Support

Ehsan S. Elgendi

(ABSTRACT)

Multimedia content, e.g., sound files, interactive demos, and video files, has been widely used in digital courses to provide an easy to use format and to emphasize the ideas. In this work, we address aspects of generating multimedia contents automatically in digital courses. In particular, we focus on two types of automatically generated multimedia: interactive glossaries and sound files. Glossaries play a major role in enhancing students' comprehension of the course core concepts. Glossary terms have complex interrelationship that cannot be fully illustrated by standard approaches, e.g., including all the terms as a linear, alphabetized list. To overcome this limitation, we introduce an interactive design for the glossary terms using concept maps. Glossary terms are visualized as nodes in graphs and their relationships are included on the edges. We implement these concept maps within the OpenDSA e-textbook system. A concept map associated with the selected term is generated on demand. We evaluate the effectiveness of our design by comparing student use of our concept-map based glossary to the traditional alphabetized list. We have designed new exercises that target the comprehension of the glossary terms to make students familiar with the concept maps. Our other work generates sound files automatically to supplement text narration in slide shows. This is made feasible by the widespread availability of text-to-speech generators in web browsers. To this end, we designed an interactive narration tool and integrated it into the OpenDSA library. In this way, all slide shows automatically have their text augmented with narration.

Interactive Multimedia in Digital Courses: Design and Evaluation of Concept Maps Glossary and Narration Support

Ehsan S. Elgendi

(GENERAL AUDIENCE ABSTRACT)

Recently, there has been an increase in the use of multimedia contents in digital courses. Multimedia files, e.g., sound files, interactive demos, and video files, are used in digital courses to provide an easy to use format and to emphasize the ideas. In this work, we address aspects of generating multimedia contents in digital courses. In particular, we focus on two types of automatically generated multimedia: interactive glossaries and sound files. Glossaries play a major role in enhancing students' comprehension of the core concepts in the courses. In general, glossary terms have complex interrelationship that cannot be fully illustrated by standard approaches, such as the alphabetized list. To overcome this limitation, we introduce an interactive design for the glossary terms using concept maps. In this design, glossary terms are visualized as nodes in graphs and their relationships are included on the edges. We implement these concept maps within the OpenDSA e-textbook system to be generated on demand. We evaluate the effectiveness of our design by comparing student use of our concept-map based glossary to the traditional alphabetized list. We have designed new exercises to make students familiar with the concept maps. Our other work generates sound files automatically to supplement text narration in slide shows. This was motivated by the widespread use of text-to-speech generators in web browsers. To this end, we designed an interactive narration tool and integrated it into the OpenDSA library so that all OpenDSA slide shows can benefit from the narration tool.

Dedication

To my beloved husband AbdelRahman,

and my precious children Omar, Adel, and Ali

Acknowledgments

First, all thanks due to ALLAH, may His peace and blessings be upon his prophet, for granting me the chance and strength to accomplish my Master's work including this thesis.

My gratitude to my advisor, Professor Clifford A. Shaffer for his inspiration, invaluable guidance, and patience. He has taught me many things, and this work would not have been possible without his encouragement and support.

I would also like to thank Dr. Edward A. Fox and Dr. Mohammed Seyam for serving on my thesis committee.

Special thanks to my parents who inspired me with their love and blessings throughout my life. I give my deepest expression of love to them. I also want to thank my lovely sisters Sara and Arwa for their encouragement, and constant love which has sustained me all the time.

I would like to thank my dearest husband, AbdelRahman, for his love and support. Without his encourage and support, I could have never been able to finish this thesis. Really, No words can express my gratitude to you. I thank my precious kids, Omar, Adel and Ali, for being patient with a super busy mum and for keeping wish me the best during my Master's degree journey.

I also want to thank all my wonderful friends in Blacksburg who made me feel at home and supported me during hard times. Also, I want to thank my friends in Egypt, who kept supporting and encouraging me.

Again, thanks to ALLAH All mighty God for giving me the ability, mindset, and perseverance to be where I am now.

Contents

List of Figures ix

List of Tables xii

1 Introduction 1

1.1 Concept Maps for Glossaries 4

1.1.1 Motivation for Concept Maps 5

1.2 Narration Support 6

1.2.1 Motivation 6

1.3 Major Contributions 7

1.4 Structure of the Thesis 8

2 Concept Maps in Education 9

2.1 Background 9

2.2 Concept Maps in Education 10

2.2.1 Concept Maps in Educational Disciplines 12

2.2.2 Concept Maps for Assessment 13

2.3 Concept Maps Tools 14

2.4 Conclusions 15

3 Concept Maps Implementation **17**

3.1 Proposed Design 17

3.2 Implementation 20

3.2.1 Platform and Planned Features 20

3.3 Creating Concept Maps from Glossary 22

3.4 Converting RST files into JSON files 24

3.5 Concept Maps Implementation using D3 25

3.5.1 Force graph actions 27

3.5.2 Concept map hierarchical levels 28

3.5.3 Backward levels 29

4 Concept Maps Evaluation **32**

4.1 Concept map Exercise Design 33

4.2 Data Collection 35

4.3 Results 36

4.4 Statistical Analysis: ANOVA Test 42

4.5 Statistical Analysis: Multi-comparison 46

5 Narration Support **51**

5.1 Introduction 51

5.2 JavaScript Algorithm Visualization Library: JSAV 53

5.3 Narration Support Implementation 55

5.4 Results 57

5.5 Open Problems with Narration Support 61

6 Conclusions and Future Work **63**

Bibliography **65**

List of Figures

3.1 An abstract example of a concept map. 18

3.2 The concept map generated for the term “graph”. 19

3.3 A concept map with the concept definition shown. 21

3.4 The proposed framework 22

3.5 Part of the glossary RST file showing the term “Acyclic Graph” and its related terms. 23

3.6 Part of the glossary file showing the “Data Type” term. 23

3.7 Part of the glossary file showing the “Graph” term. 24

3.8 Python code to convert RST files into JSON files 25

3.9 Part of the JSON file with the connections specified 26

3.10 The generated concept map for the glossary term “data type”. 27

3.11 Backward level for the concept “primary storage”. 29

3.12 Backward level for the concept “pointee”. 30

4.1 An example of a multiple choice involving a concept map 34

4.2 An example of a True/False including concept maps. 35

4.3 Student use of the glossary in CS2114 in the Spring 2019 semester (the median is 7). 37

4.4 Student use of the glossary in CS3114 in the Spring 2019 semester. 38

4.5 Student use of the glossary in CS2114 in the Spring 2019 semester after excluding the exercises direct effect. 40

4.6 Percentage of students who have used the glossary multiple times in CS2114. Notice that the last two bars show use in Spring 2019; the yellow bar includes uses to complete exercises while the blue bar shows only uses unconnected with the exercises. 41

4.7 Percentage of students who have used the glossary multiple times in CS3114. 42

4.8 Interval plot for the means and variances in CS2114. 43

4.9 Interval plot for the means and variances in C2114 after excluding direct exercises effect. 44

4.10 Interval plot for the means and variances in C3114. 45

4.11 Tukey multi-comparison test for CS2114 complete data. 47

4.12 Dunnett multi-comparison test for CS2114 complete data. 48

4.13 Tukey multi-comparison test for CS2114 after excluding the exercises direct effect. 49

4.14 Dunnett multi-comparison test for CS2114 after excluding the exercises direct effect. 50

5.1 A typical interactive algorithm visualization created using JSAV. 53

5.2 The next slide after the one shown in Figure 5.1. 53

5.3 Another format for JSAV algorithms. 54

5.4 JSAV example after including narration support with the default selection to be OFF. 56

5.5 JSAV example after changing the narration support to be ON. 57

5.6 Student use of the narration feature in CS2114 in the Spring 2019 semester (11 students). 59

5.7 Student use of the narration feature in CS2114 in the Fall 2018 semester (19 students). 59

5.8 Student use of the narration feature in CS3114 in the Spring 2019 semester (17 students). 61

5.9 Student use of the narration feature in CS3114 in the Fall 2018 semester (16 students). 62

List of Tables

4.1 Students' use of the glossary compared to the total number of students in CS2114. 37

4.2 Students' use of the glossary compared to the total number of students in C32114. 38

4.3 One-way ANOVA test results for CS2114 complete data. 44

4.4 One-way ANOVA test results for CS2114 after excluding the exercises direct effect. 45

4.5 One-way ANOVA test results for CS3114. 46

5.1 Students' use of the narration tool in CS2114 for both Fall 2018 and Spring 2019. 60

5.2 Students' use of the narration tool in CS3114 for both Fall 2018 and Spring 2019. 60

Chapter 1

Introduction

Multimedia is widely used in digital education to enhance students experience and to facilitate delivering information. The goal of using multimedia is to increase the number of communication channels such as audio, visual, or interaction used by students, by integrating more than one communication channel into a presentation. This has been shown to be more beneficial in delivering information than a single communication channel such as written text [1]. Multimedia can also help students to focus more, so instead of just reading text, they can watch videos that explain some topic or listen to a recorded lecture which should be more appealing for them.

Mayer [2] defined multimedia learning as learning from words and pictures. Words can come in multiple formats such as printed text or audio narrated text. Pictures, on the other hand, refer to static images, such as illustrations, charts, diagrams, and dynamic content, such as animations or video. Using these two formats, words and pictures, helps to stimulate more learning channels in the brain as it was shown that human brains possess separate channels for processing visual and verbal material. Given that each channel can process only a small amount of information at a time, it will be better to use more than one channel simultaneously.

In [3], it was shown that multimedia has the ability to stimulate learner motivation by its multiple representational modalities which, in turn, can improve transfer the information in an attractive way. In particular, combining text, animations, graphics, audio, and video can

help students to better comprehend the course materials and to understand the relevance of various contextual elements. Other research has studied the effect of using multimedia on the education process such as [4, 5, 6].

Mishra and Sharma [4] covered concepts related to using multimedia in digital education, especially interactive multimedia. They started by defining multimedia in light of its use in digital education, showing that multimedia has multiple definitions that share combining multiple communication channels or media files into a single output that is more attractive to the learner. Mishra and Sharma discuss aspects of effective multimedia design and what good and effective interactive multimedia should be. In [5], Reddi describes the use of multimedia as a power education technology. He also discussed the potential use of multimedia as a pedagogical tool. In [6], Low *et al.* designed and deployed a multimedia learning system that uses commercial software platforms in building an interactive learning system. The importance of multimedia is also highlighted in their work.

Multimedia is as effective in computer science courses as it is in other disciplines. Naps *et al.* [7], presented a framework to experimentally study the effectiveness of visualizations in computer science courses. Their main finding was that to effectively use visualization, an active learning environment should be created to engage the students in the learning process. A similar framework was presented in [8] to evaluate the effectiveness of algorithm visualizations through a meta-study of 24 other experimental studies that evaluate algorithm visualizations. The most important finding of this study is that the way students use these visualizations has a greater impact on effectiveness of these visualization, which corroborate the findings of using interactive multimedia.

Although multimedia is powerful in digital courses, its use in online courses is limited by many factors. According to Reddi and Mishra [9], digital courses distributed on physical means like CDs are limited only by the storage space, however, web-based courses are limited

by both the server's storage space and the connection bandwidth. The latter is especially important because a student browsing an online course will not like to wait long for every page to load. Reddi and Mishra [9] suggested some ways to avoid these drawbacks, such as using low quality multimedia files and distributing the media files on different web-pages.

In this work, we design and develop multimedia objects that are generated automatically when a student selects these objects. Since these files are generated automatically, they will require less storage space on the server side than storing, e.g., video files. They will also require less communication bandwidth as they are generated only when being selected. Therefore, these files will comply with the findings of Reddi and Mishra [9] about the limited resources in online courses.

In this work, we focus on two applications of these automatically generated multimedia files. The first is creating interactive concept maps for the glossary terms, which should provide an interactive environment in which students can draw connections between different glossary terms and navigate through different terms. The second application pertains to narration support files in which audio files are generated automatically for the selected sections in the courses. We also integrated the sound control in an interactive tool in which students can control the features.

We have implemented these tools, i.e., concept maps and narration support, in the context of OpenDSA [10]. OpenDSA is an open source infrastructure for digital textbooks that is currently used by many Computer Science courses such as data structures and algorithms, and programming languages in both CS2 and CS3 levels. OpenDSA integrates multiple features to the ordinary textbooks such as interactive visualizations and auto-graded exercises. Combined with its easy to use clutter-free interface, OpenDSA provides a rich experience to the end users (students).

Next, we give an introduction for each application and the motivation for selecting such applications.

1.1 Concept Maps for Glossaries

Books have a long tradition of including glossaries and this has carried over to digital media [11] and [12]. Glossaries, or glossary terms, are collections of the main terminologies and jargon used in the associated text, along with their meanings or definitions. Traditionally, a glossary is displayed using a list that is sorted either alphabetically or grouped by chapters.

Computer science courses are not different from other digital courses when it comes to glossaries. Glossaries play a significant role in providing definitions for the many concepts and terms found in computer science courses. This holds true especially in the core courses in Computer Science such as Data Structures and Algorithms, which usually have hundreds of glossary terms [13].

Many Computer Science instructors like to include algorithm visualizations int their courses to convey the dynamic nature of the algorithms [14]. Algorithm visualization is the use of graphical tools to visualize the steps of an algorithm and to illustrate how it works. Examples of these visualizations include sorting algorithms, in which the algorithm is illustrated by showing each record move as the algorithm is executed on an input, typically an array. Algorithm visualizations can be either interactive, where students can manipulate the input and test the effect on the output, or non-interactive, in which students can only navigate through the algorithm steps. Interactive algorithms are considered to be an active method of instruction and, thus, preferred over the other passive techniques [7], [8], and [15].

1.1.1 Motivation for Concept Maps

Given the complex interrelationships of the glossary terms in Computer Science courses, the traditional alphabetical list method does not provide a suitable means for students to make the best use of the glossaries. In particular, we have identified the following shortcomings in traditional glossary lists:

- Students find it hard to navigate through the long list of glossary terms to find terms of concern. Moreover, this long list does not represent an appealing format for the students.
- The list does not help students to draw connections between related terms.
- Usually no or minimal interaction is available with the terms.

These shortcomings are believed to affect and limit the value students get from using the glossaries. Therefore, we propose to implement a new design for the glossary terms that can overcome these shortcomings. The main motivations for our implementation are as follows:

- We want the design to be more appealing for students. This will encourage students to use the glossaries more frequently.
- In this design, we need to focus on the term of concern along with its relationship to other terms.
- The visualization tool used to implement our design needs to be compatible with the rest of our infrastructure.
- The visualization tool used in the implementation should provide a means of interaction for the students.

For this reasons, we adopt an implementation of the glossary based on concept maps. Concept maps are discussed in Chapter 2, and our implementation and evaluation in Chapters 3 and 4.

1.2 Narration Support

Narration support, within our context, is the feature of including automatically generated sound files to supplement course content. Specifically, for us this means audio narration of the text in slide show captions. This allows students to both read and listen to the course materials. According to Mayer [2], spoken and written text represent the same format of multimedia content. Therefore, we include this feature within interactive algorithm visualization tools which already utilize two cognitive brain channels. The sound, here, will help the brain to better process the information already found in the visualizations. To this end, we designed our narration support tool to be compatible and integrated with JSAV [16], the graphics library used by OpenDSA. Moreover, other digital courses that use JSAV will be able to make use of this tool. The motivations for this work are discussed next.

1.2.1 Motivation

We have identified the following motivations to direct our design.

- We want to stimulate more learning channels in students' brains and provide a more attractive format for the algorithm visualization slide shows through narration support.
- The narration support needs to focus on the important parts only to encourage students to use the feature.

- The narration support needs to be used in an interactive tool in which students can have the basic controls over the audio files, e.g., play, pause, forward, and backward.
- The narration support needs to integrate on a natural way with the context, allowing students a choice of whether to use it or not.
- The narration support needs to avoid creating an additional burden to content developers.

Narration for slideshows has been a desired feature in the OpenDSA e-Textbook system for years. But the time and expense that would be required to manually create narrations has deterred implementing this feature. With the recent availability of automated text-to-speech support in modern browsers, an automated approach to solving the problem has become practical. This is therefore the approach that we have used.

Our implementation is discussed in Chapter 5.

1.3 Major Contributions

The major contributions of this thesis can be summarized as follows.

- Design a system that automatically generates interactive concept maps for the glossary terms, that help to visualize the relations between different glossary terms.
- Provide an easy-to-use implementation for the glossary maps that is compatible with most web browsers and technologies.
- Collect and analyze data about the student use of the new concept maps implementation.

- Design exercises tailored to concepts maps to measure students' understanding of the basic concepts after using the newly designed concept maps.
- Design an interactive tool to support narration in digital courses and integrate it into, JSAV in Computer Science courses.
- Collect and analyze data related to student use of the narration tool to see if narration is a popular feature.

1.4 Structure of the Thesis

In Chapter 2, we give background about concept maps and their use in digital education, followed by our proposed design to address the previously listed shortcomings and motivations in Chapter 3. We then evaluate our design and analyze the usage data in Chapter 4. Narration support is introduced in Chapter 5, and finally conclusions and future work are given in Chapter 6.

Chapter 2

Concept Maps in Education

2.1 Background

Concept maps have been widely used in many applications to represent the relation between a number of related items or terms. According to [17], concept maps are graphical tools that can organize and represent knowledge. A concept is designated by a label which is displayed in a box or circle and connected using lines to other concepts. Thus, the concept map can be considered to be a graph with concepts as the nodes and relationships as the edges. Concept maps were developed in 1972 [18] by Joseph D. Novak, to follow and understand changes in children's knowledge of science. Novak found that representing the information in this type of map was helpful to better understand and relate the different changes related to children.

This ability of concept maps to deliver better information was later addressed by psychological research that interpreted the relation between concept map design and our brains. The basic relationship was concluded from the work in [19], which models human memory not as a single "vessel" that can be filled sequentially, but rather as a complex set of interrelated memory systems which can be fed with parallel information at once. That is why viewing the same information represented in a concept map with the concepts displayed side by side is easier to comprehend than reading a long paragraph with the same information.

Concept maps are among a group of tools that can be used to construct and share information

in a meaningful way. Eppler [20] discusses concept maps, mind maps, conceptual diagrams, and visual metaphors. Concept maps, according to Eppler, represent a top-down diagram that shows the relationships between different concepts and their interrelationships. A mind map, on the other hand, is a radial diagram that is centered around an image and that represents the hierarchical connections between different portions of the learned materials. A conceptual diagram uses boxes to represent abstract concepts in a systematic illustration and it also highlights the relationship between these concepts. Finally, a visual metaphor is a graphical structure created based on the shape of a popular natural or man-made artifact that can be easily recognizable. It can model an activity or a story to organize contents meaningfully.

These different graphical representation tools are used to deliver different types of information. Of these different tools, the concept map was shown to be the best tool that can be used to deliver information for students and to summarize the key topics in courses [20]. They can also help to clarify the elements of a big topic and to give examples of specific concepts.

These findings helped to draw attention to the importance of using concept maps in education to highlight the main ideas and facilitate the learning for the students.

2.2 Concept Maps in Education

The possible use of concept maps in education was discussed in [21] The initial focus was on teaching science, however, Novak determined that concept maps were useful to represent knowledge and information from many disciplines. The main result of the research was that concept maps can be beneficial to students, however, they cannot be the only source of information. Instructors still need to develop methods for integrating these concept maps

into their courses to maximize their benefits.

In a recent follow-up study [22], Novak suggested using "expert skeleton" concept maps for educational purposes. Expert skeleton concept maps are those maps prepared by an expert in the knowledge domain and are used to represent the basic information in the field in the form of scaffolding. This was proven to facilitate meaningful learning and to help to represent the general view of the ideas in a way that removes misconceptions.

In [23], Stewart *et al* suggest three different uses of concept maps in the educational process: curricular tools, instructional tools, and a means of evaluation. The curricular aspect of the concept maps is related to designing the curricula of the subject by helping the designer to determine the main ideas in the subject that need to be covered, and, hence, improve the intended learning outcomes. The instructional aspect of the concept maps deals with teaching the concepts and the ideas to the students, which is the most obvious use of concept maps in education. Finally, the evaluation aspect suggests using concept maps in students' assessment.

The trade-off between introducing an effective method of learning, using concept maps, and the cognitive load they cause to students was discussed in [24]. The authors introduced a web-based problem-solving environment built on concept maps to summarize information for students. The framework was tested on a social studies course for elementary school students with the goal to determine the effectiveness of concept maps on student comprehension and also on their satisfaction. Statistical results showed that students were able to better perceive the core concepts, however, they indicated that this method of teaching had too much cognitive load. Based on these important findings, we note that concept maps should not be the only source of information for students, rather, they need to be used with other formats in order to better present the material.

Concept maps are also used to facilitate other services related to education. In [25], a framework is presented to model courses prerequisites using concept maps. This was shown to provide teachers with the ability to analyze and refine their teaching strategies based on examining the relations of their courses to other courses. In particular, a three-phase framework is used to automatically generate the concept maps by analyzing the students performance and classifying them into groups.

In [26], the effect of generating concept maps on students was discussed. Students in a middle school were divided into three groups: those who individually generated concept maps for some course concepts, those who created these concept maps in groups, and those who did not use concept maps to learn the material. The results of the study have shown that students who worked on concept maps, in general, had more positive attitudes towards concepts than those who did not. However, students who worked individually on generating concept maps had better understanding of the concepts, and they also liked generating these concept maps more than those who worked in groups.

2.2.1 Concept Maps in Educational Disciplines

Concept maps have been used in many disciplines to facilitate the learning of the main ideas and concepts. In [23], the use of concept maps in teaching biology was discussed. Stewart *et al* explained how to extract the main concepts in biology and how to define relations between these concepts to be used in concept maps. The work presents the basics of designing concept maps and suggests performing empirical experiments to study the effectiveness of applying the proposed approaches on students cognition.

Similarly, Lloyd [27] has studied the elaboration of concepts in biology. In particular, three text books in biology were chosen that target different audiences, and their presentation of

a specific concept was investigated. The author created three different concept maps for the concepts of interest, one from the information introduced in each book. The goal of the study was to evaluate the different levels of presenting related concepts with a specific concept. Similar to [24], Lloyd has shown that there is a trade-off between the amount of information represented in each of these concept maps and the cognitive load on students.

Bon-Marten *et al* in [28] explored the applicability of using concept maps to deliver scientific knowledge used for practical decision-making situations. They performed five studies to cover five different fields in public health. Results of the studies showed that concept maps were effective in highlighting the key issues and delivering the required medical information. The authors suggest to use concept maps widely to improve theory development, leading to making decisions, over formulating new theories.

In [29], the use of concept maps in engineering education was discussed. The work related the most important concepts in engineering and identified their relationship. These concepts included experimentation, research, analysis, and modeling. Turns *et al* have also shown the connection between these concepts and the impact of their implementation on the environmental, ethical, and social aspects.

2.2.2 Concept Maps for Assessment

Ruiz-Primo and Shavelson [30] present the possible ways to use concept maps in the assessment process. One method is to give students a partially filled concept map with multiple possible choices to be used to complete the concept map. Here, the student tries to create the concept map based on the knowledge he has gained from the course. Another approach is to give students a concept map with missing relations, and the student should again choose from the given relations to complete the diagram. The work evaluated these methods and

they were shown to be accurate and consistent ways to assess the students.

Turns, Atman, and Adams in [29] presented a complete framework for student assessment in engineering education. The framework considers using concept maps for both course-level and program-level assessments. Course-level assessment describe the assessment within specific courses of the level of learning associated with it. The purpose of this kind of assessment can vary from quickly measuring students' understanding to exploring what students have perceived in the learning process. It can also help to assign grades to students. Program-level assessment refers to the assessment performed to evaluate the overall knowledge of a group of students. This type of evaluation needs to consider a student's level of expertise in this specific domain, to identify the student's knowledge about the whole discipline, and to explore the student's comprehension of the basic engineering concepts. The authors suggest that engineering schools must adopt both course-level and program-level assessments.

2.3 Concept Maps Tools

In this section we consider both commercial (whether free or not) tools that can be used to generate concept maps, and also tools that were proposed in literature. First, we consider the commercial tools that are electronically available and that have been developed to help design and create concept maps. Examples of these tools include Mindmup [31], Cmap [32], Lucidchart [33], and Mindmeister [34]. All these tools have many features that enable users to create powerful concept maps. They differ in the features they provide as some allow users to create a concept map online and store it in the cloud. Others require use of their software which can be downloaded based on the user's operating system. The output of these tools can be saved in multiple formats, such as images or portable documents.

However, the main drawback of these tools that makes them unsuitable for our implemen-

tation is that they all produce static output. Given the number of glossary terms found in our target digital courses, it would not be feasible to show the entire course concept map, nor, is it feasible to create a map for each term. Another drawback is that generating a map as a static image will not allow students to have interaction with the map.

The research literature has many proposed tools to automatically generate concept maps. In [35], a tool known as Concept Map Miner (CMM) was introduced to automatically generate concept maps. The first step of CMM involves identifying the main concepts and their relationships. In particular, the grammatical tree of each sentence is exploited to extract the compound nouns. The relationships between these nouns are, then, identified through a semantic layer. Finally, the output of the first step is represented as a terminological map and is then transformed into a reduced version with no grammatical dependencies.

However, one issue with CMM and similar tools is that they depend on the structure of the given sentences to extract the concepts that will be represented in the map. Thus, the extracted concepts represent the most common words in these sentences, which are not always the core concepts of the course. Therefore, such tools will not to be suitable to generate concept maps for the glossaries in a digital course.

2.4 Conclusions

In this section, the importance of using concept maps in education was highlighted. The use of concept maps in specific disciplines and the various methods of using them were discussed. In our work, we build on these previous studies by introducing a tool to generate concept maps for courses glossaries. The main challenges in this step is the huge number of terms that can be found in glossaries, in a typical course. This huge number of terms allows to create more concepts maps, but on the other hand is challenging in making the students

focus on the main concept itself which requires choosing the relationships carefully so that the concept maps remains useful for the students. We address this in later chapters.

Another issue related to the automatically generated concept maps, as discussed earlier, is their dependency on the sentences structure to extract the concepts. This, in turn, generates many irrelevant concepts to the course core concepts. To overcome this limitation, core concepts and their relationships will be provided to our concept map generation tool. This will help to eliminate any misconception within the generated concepts maps and will help to represent only the core concepts in a given course. Our tool will then process the provided information to generate the actual concept map. Finally, we also consider the static nature of the commercial tools and overcome this by creating dynamic concept maps.

Chapter 3

Concept Maps Implementation

In this chapter, we present our proposed tool to automatically generate concept maps. This tool is used to automatically generate interactive concept maps for glossaries from a collection of terms and references to related terms. The concepts and their relationships will be prepared by humans to ensure that only the core concepts of the courses are presented. The tool was developed and tested on the courses CS2114 and CS3114 from Virginia Tech. These courses mainly focus on data structures and algorithms.

3.1 Proposed Design

Recall that concept maps are a type of graph that links concepts (nodes/nouns) with relationships (edges/verbs). If we wish to focus on the relationships for a particular concept, then the graph can put that node at the center. Besides the central node, concept maps include a number of links or branches connecting this main concept to other concepts or nodes.

In the research literature, concept maps have many designs according to the application. For instance, some concept maps use a horizontal hierarchical view for the nodes. Others use tree designs, which represent a vertical hierarchy. General graphs do not follow a special structure. Links are usually directional, pointing from one node to another, but they can also specify bi-directional relations between the nodes and in this cases different relations

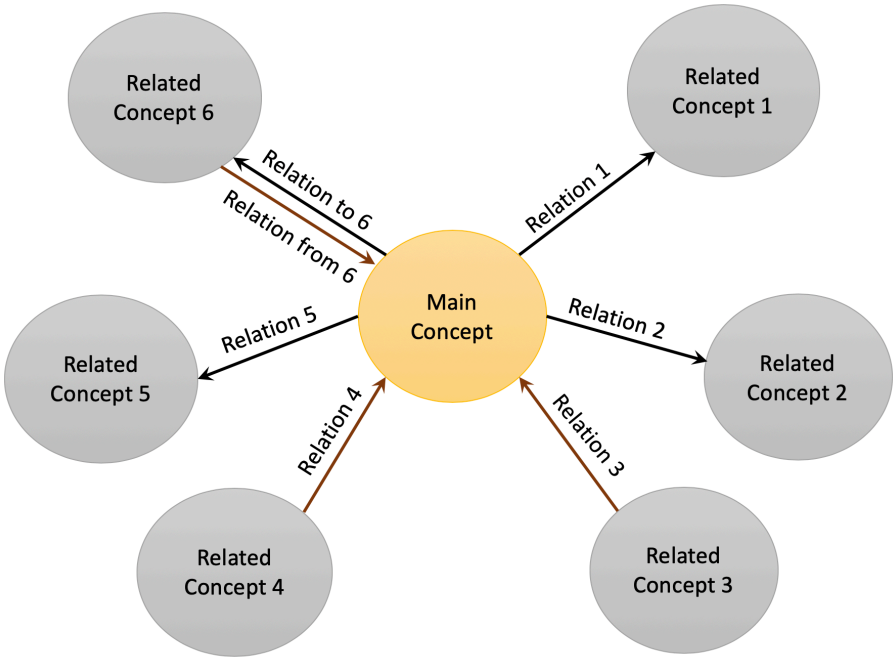

Figure 3.1: An abstract example of a concept map.

are given for different directions. In either case, the links are typically labeled with the type of relation between these nodes. Figure 3.1 shows an abstract example of a concept map.

This flexibility in designing concept maps makes them a good fit to model glossaries. In our proposed design, we consider each glossary term separately. This glossary term is represented as the main concept in its local region of the main graph of the concepts for the course. We manually identify the related terms to this central concept. The related terms are then presented in the concept map as nodes connected to the main term. Here, we use directional links pointing out from the main concept toward these terms. The links used in our concept map design will be labeled according to the relation between the terms. Example relations include part of, example of, implemented in, consists of, synonym, etc. Figure 3.2 shows a sample concept maps that includes one main concept and multiple related concepts.

The concept map of Figure 3.2 can be further extended by including terms from other

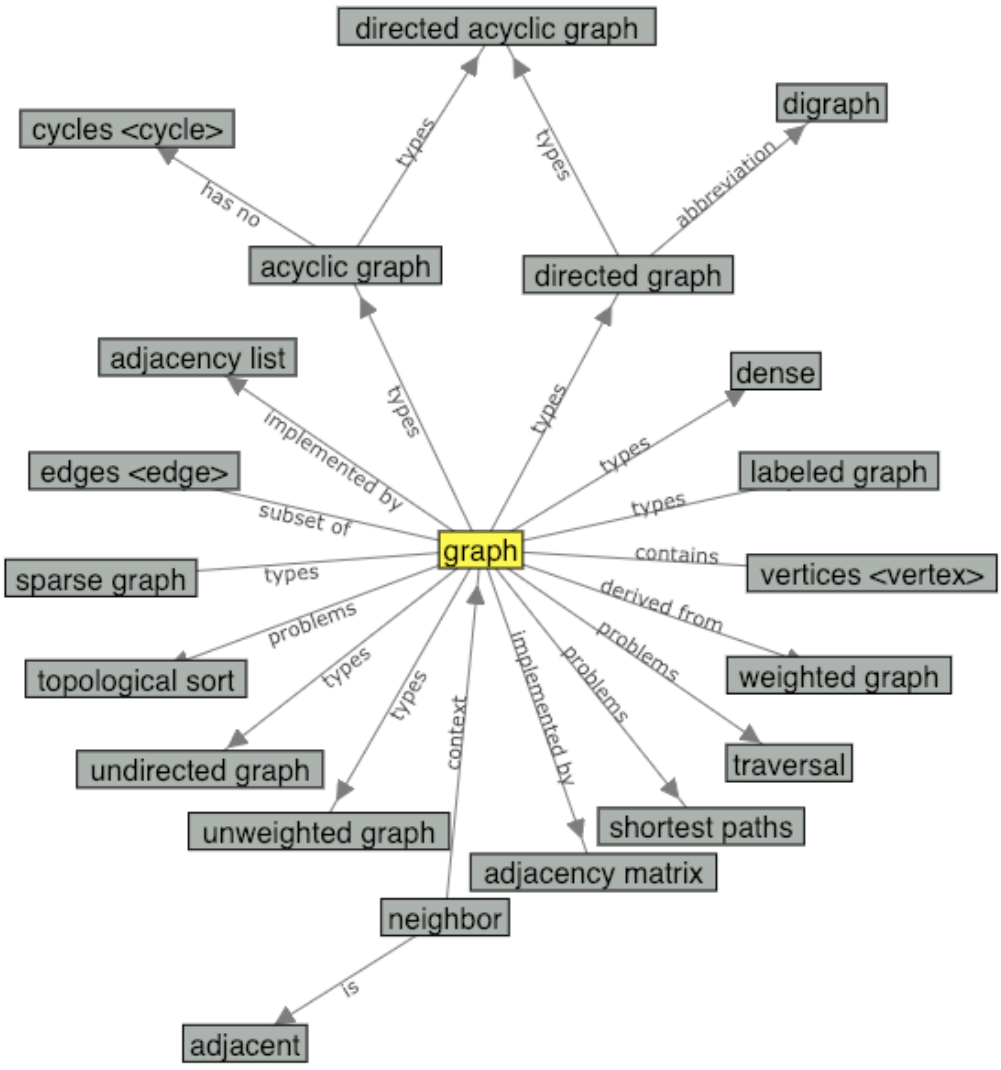

Figure 3.2: The concept map generated for the term “graph”.

levels, i.e., concepts that are related to these concepts that have a direct relationship to the central concept. However, there are usability concerns with making the concept map extend too far. There are also performance concerns with how we store and locate the various relationship. Next, we will consider the practical implementation of our concept map design and the guidelines we followed to address the motivations and concerns related to the implementation.

3.2 Implementation

We define the following goals, to be included as features, in our implementation:

- The concept map needs to be interactive so that the output is a dynamic graph that students can manipulate.
- The concept map should be accessible from the course contents. However, we will display both the original glossary list as well as the generated concept map graph when students click on a term in the course contents, so that students can check both.
- Interactive actions need to be implemented on the dynamic graph to enhance the user experience when using the concept map.
- Although the concept map can theoretically support any number of hierarchical levels, we will limit the number of hierarchical levels so as not to clutter the display to the point where it distracts users from the concepts relations of the selected term.

3.2.1 Platform and Planned Features

In this section, we present our implementation of the desired features, along with the tools used in the implementation. We use D3 [36], a JavaScript visualization library that helps to create visualizations from data. D3 uses HTML, SVG, and CSS, so it is compatible with most web browsers.

We implemented the concept map to be generated on-the-fly when a concept term is selected by the user, either from the module text or from the glossary list. The concept map is automatically generated every time so we do not need to store the map for every concept, nor spend a lot of time loading them from the server.

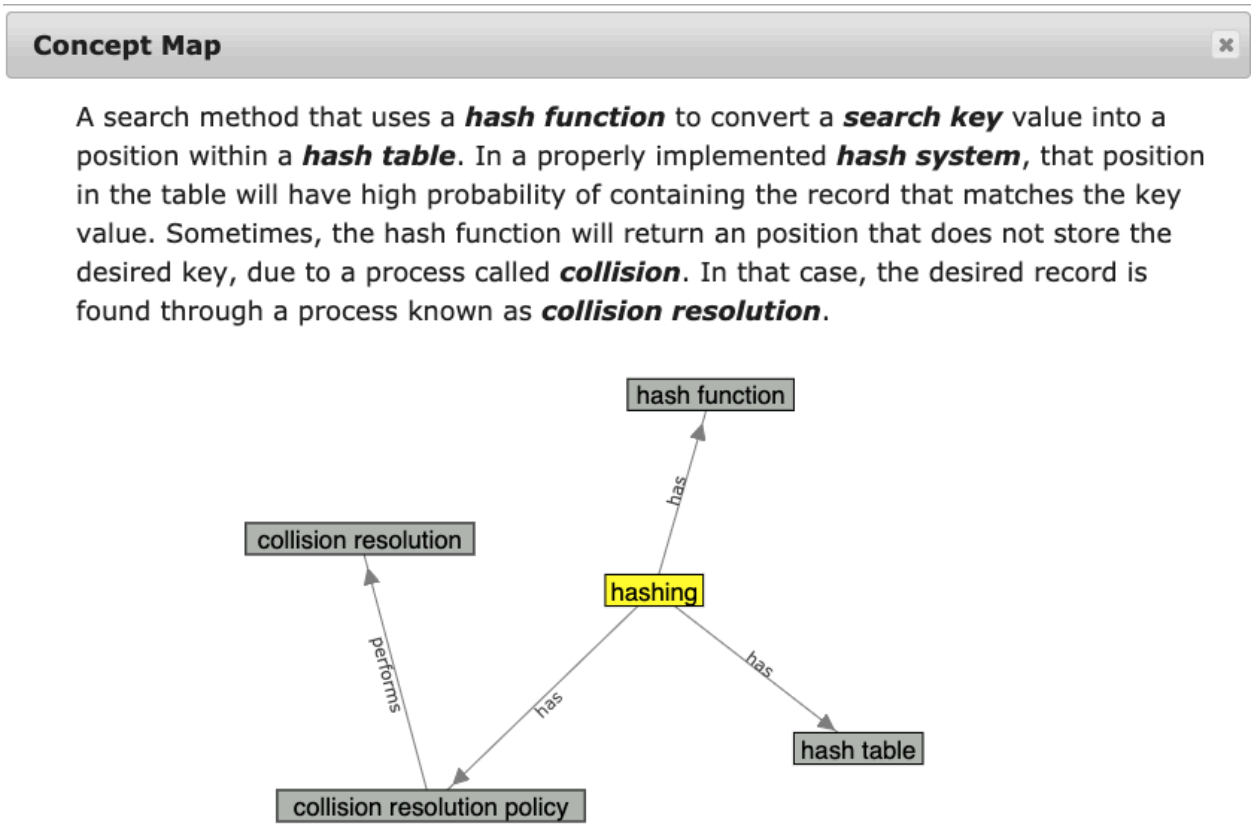

Figure 3.3: A concept map with the concept definition shown.

We set the concept map to be displayed in a separate window, while the original glossary list is displayed in the original window and the selected term is highlighted. In this window, the term definition is also included on top of the graph as shown in Figure 3.3. This allows students to read the concept definition while viewing its relations to other concepts in order to better understand the relations.

Next, we discuss the stages of converting the glossary terms into concept maps and adding interactive actions to them using D3. Figure 3.4 shows an overview of the stages of our framework.

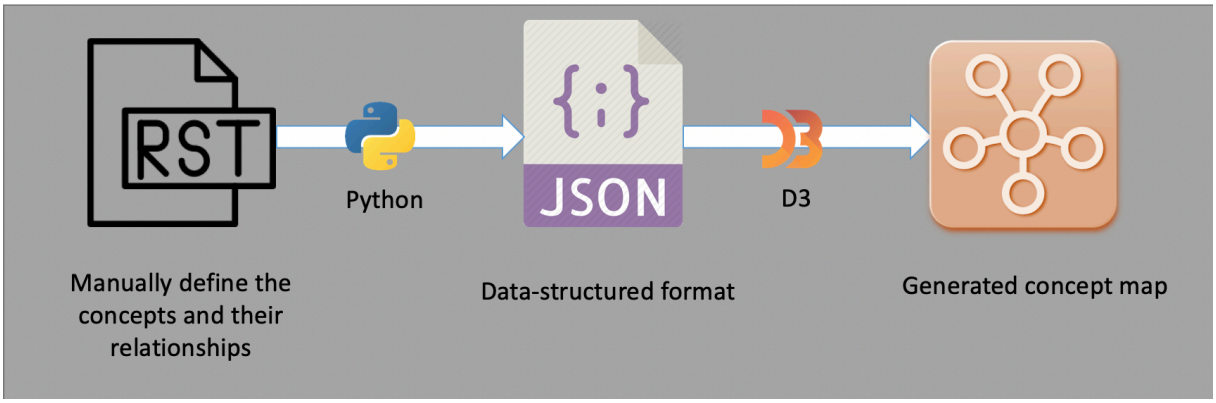

Figure 3.4: The proposed framework

3.3 Creating Concept Maps from Glossary

Our implementation requires that the list of concepts and their relationships to be predefined. Since we are dealing with glossary terms, we already have the list of concepts. The goal now is to define the relationships between the different concepts. These relations are defined as one-way (directed) relations from a concept towards another concept. If there can be a relation in the opposite direction, it needs to be defined as a separate relation from the second concept to the first.

OpenDSA uses ReStructredText (RST) as its authoring language for content. RST is lightweight markup language widely used by the Python community. It includes a number of features for creating digital book-like artifacts, including glossary support. OpenDSA uses RST (and the Sphinx compiler for converting RST to HTML) because it is extensible. Thus it is easy for us to take the standard glossary support mechanism (a list of terms with definitions) and annotate those entries with other attributes.

The starting point for our concept map implementation is the RST glossary file. Each term has a separate entry in the file, along with its definition. Figure 3.5 shows part of the glossary RST file, specifically, for the “acyclic graph” term. We can see that the term is displayed

```
acyclic graph
   :to-term: directed acyclic graph :label: types
   :to-term: cycles <cycle> :label: has no

   In :term:`graph` terminology, a graph that contains no
   :term:`cycles <cycle>`.
```

Figure 3.5: Part of the glossary RST file showing the term “Acyclic Graph” and its related terms.

```
data type
   :to-term: abstract data type :label: implemented by
   :to-term: aggregate type :label: type
   :to-term: simple type :label: type
   :to-term: list :label: example
   :to-term: array :label: example
```

Figure 3.6: Part of the glossary file showing the “Data Type” term.

on the first line, and the related terms are displayed on the following two lines after the keyword “to-term” which is an attribute that we have added to glossary terms. This means there will be a relation pointing out from the term “acyclic graph” towards“directed acyclic graph” and “cycles” terms. The label for each relation is also shown in Figure 3.5. The labels used here are “types” and “has no” to indicate the relation type between the concept and its connected concepts.

Other examples are presented in Figures 3.6 and 3.7. Figure 3.6, shows the connections created for the concept “Data Type”. There are five connected terms with new relationships shown as “implemented by” and “example”.

In Figure 3.7, we show the connections for one of the most important concepts in data structures, i.e., “Graph”. Here, we were have defined 15 different connections with “Graph”. For example, the relation “contains” connects “Graph” to “edges”. The remaining relation is “problems” which represent problems that are built on graphs.

```
graph
   :to-term: edges <edge> :label: contains
   :to-term: vertices <vertex> :label: contains
   :to-term: adjacency matrix :label: implemented by
   :to-term: adjacency list :label: implemented by
   :to-term: traversal :label: problems
   :to-term: topological sort :label: problems
   :to-term: shortest paths :label: problems
   :to-term: dense graph :label: types
   :to-term: sparse graph :label: types
   :to-term: directed graph :label: types
   :to-term: acyclic graph :label: types
   :to-term: labeled graph :label: types
   :to-term: undirected graph :label: types
   :to-term: weighted graph :label: types
   :to-term: unweighted graph :label: types
```

Figure 3.7: Part of the glossary file showing the “Graph” term.

The process of editing the RST file to add the connections that build the concept map is actually the most time-consuming process in our implementation. It needs only to be done once at the beginning, but it requires examining the concepts manually, defining their relations, and entering them into the RST file. After storing these relations in the RST file, the remaining stages are automated to generate the concept maps automatically.

3.4 Converting RST files into JSON files

The next stage, after defining the connections that define the concept map in the RST file, is to convert it into a structured data format that can be processed in other programming languages effectively. Here we choose JSON [37] as our data-structured format. JSON stands for “JavaScript Object Notation” which is a lightweight data-interchange format that can be used with JavaScript. This allows later for this JSON file to be easily processed using standard JavaScript tools.

The process for converting the RST file into JSON is implemented by a python script that

```
#parses the glossary terms relationships. prints error message if the format is not correct
#format    :to-term: term1 :lable: label :alt-text: alternate text in case the to-term is not a glossary term
def parse_term_relationship(line, term, line_num, cmap_dict, console_msg_prefix = ''):
  if line.strip().startswith(':to-term:') and ':label:' in line.lower():
    args = re.split(':to-term:|:label:', line)
    term = term.strip().rstrip('\n')
    cmap_dict['concepts'][term] = ''

    if (args[1] not in cmap_dict['concepts']):
      if len(args) == 3:
        args[1] = args[1].strip().rstrip('\n')
        cmap_dict['concepts'][args[1]] = ''
    if args[2].replace(" ", "").strip().strip('\n') not in cmap_dict['linking_phrase']:
      #size_lp = len(cmap_dict['linking_phrase'])
      #cmap_dict['linking_phrase']['lp-' + str(size_lp + 1)] = args[2]
      cmap_dict['linking_phrase'][args[2].replace(" ", "").strip().rstrip('\n')] = args[2]
    if args[1].replace(" ", "").strip().strip('\n') not in cmap_dict['linking_phrase']:
      size_c = len(cmap_dict['connections'])
      #cmap_dict['linking_phrase']['lp-' + str(size_lp + 1)] = args[2]
      cmap_dict['connections']['con-' + str(size_c + 1)] = {'from': term, 'to': args[1], 'label': args[2]}
      #cmap_dict['connections']['con-' + str(size_c + 2)] = {'from': args[2].replace(" ",
      "").strip().rstrip('\n'), 'to': args[1]}
  else:
    print_err("%sWARNING: Glossary terms relationship declaration on line %d" % (console_msg_prefix, line_num))
```

Figure 3.8: Python code to convert RST files into JSON files

reads the RST file lines, adds more information to these lines, and stores them into the JSON file. Part of the python script is shown in Figure 3.8, in which we can see how the JSON structure is created using Python commands. For example, the string "con-" is added to the beginning of each relationship to indicate that there is a connection between these terms.

Executing the Python script creates the JSON file. Figure 3.9 shows part of the resulting JSON file. We can see that every connection in the JSON file is represented as a tuple in the format ($``con-i"$: {$``to"$: to-label, $``from"$: from-label, $``label"$: label-name}). Where "con-i" is the connection number, "to-label" is the main concept, "from-label" is its related concept, and "label-name" is the type of connection.

3.5 Concept Maps Implementation using D3

The JSON file is then fed to the last stage, which generates the concept map. We have used JavaScript for the actual implementation of the concept maps. The implementation

```
{"connections": {"con-106": {"to": "labeled graph", "from": "graph", "label": " types\n"}, "con-101": {"to":
"shortest paths", "from": "graph", "label": " problems\n"}, "con-100": {"to": "topological sort", "from":
"graph", "label": " problems\n"}, "con-29": {"to": "double rotation", "from": "AVL Tree", "label": "
operations"}, "con-28": {"to": "asymptotic algorithm analysis", "from": "asymptotic analysis", "label": "
formal synonym\n"}, "con-25": {"to": "queue", "from": "array-based queue", "label": " implementing\n"}, "con-
24": {"to": "array-based list", "from": "array-based queue", "label": " analogous to"}, "con-27": {"to":
"algorithm analysis", "from": "asymptotic analysis", "label": " synonym"}, "con-26": {"to": "intermediate
code", "from": "assembly code", "label": " form of"}, "con-21": {"to": "array", "from": "array-based list",
"label": " uses"}, "con-20": {"to": "amortized analysis", "from": "amortized cost", "label": " used in"}, "con-
23": {"to": "stack", "from": "array-based stack", "label": " implementing\n"}, "con-22": {"to": "array-based
list", "from": "array-based stack", "label": " analogous to"}, "con-123": {"to": "replacement selection",
"from": "heapsort", "label": " variant"}, "con-122": {"to": "min heap", "from": "heap", "label": " example\n"},
"con-121": {"to": "max heap", "from": "heap", "label": " example\n"}, "con-120": {"to": "priority queue",
"from": "heap", "label": " used in\n"}, "con-127": {"to": "Parse tree", "from": "intermediate code generation",
"label": " walks through"}, "con-126": {"to": "subclass", "from": "inherit", "label": " has\n"}, "con-125":
{"to": "base class", "from": "inherit", "label": " has"}, "con-124": {"to": "heap", "from": "heapsort",
"label": " concept of\n"}, "con-242": {"to": "node", "from": "vertex", "label": " synonym"}, "con-129": {"to":
"discriminator", "from": "kd tree", "label": " uses"}, "con-128": {"to": "intermediate code", "from":
"intermediate code generation", "label": " produces\n"}, "con-228": {"to": "PR quadtree", "from": "spatial data
structure", "label": " example\n"}, "con-229": {"to": "splaying", "from": "Splay Tree", "label": " operation"},
"con-130": {"to": "key space", "from": "key", "label": " has"}, "con-131": {"to": "key", "from": "key sort",
"label": " uses"}, "con-132": {"to": "object-space decomposition", "from": "key-space decomposition", "label":
" type"}, "con-18": {"to": "Floyd's algorithm", "from": "all-pairs shortest paths problem", "label": " solved
by"}, "con-19": {"to": "algorithm analysis", "from": "amortized analysis", "label": " is"}, "con-133": {"to":
"image-space decomposition", "from": "key-space decomposition", "label": " type\n"}, "con-10": {"to": "member",
"from": "aggregate type", "label": " has\n"}, "con-11": {"to": "problem", "from": "algorithm", "label": "
```


Figure 3.9: Part of the JSON file with the connections specified

is developed using D3, as discussed in Chapter 1, which makes it compatible with many visualization tools implemented using JavaScript.

We have implemented the concept maps to be generated every time a user selects a glossary term, from either the glossary page or from the course contents. The D3 procedure is executed each time that a glossary term is clicked, to generate the concept map automatically. We choose to display the generated concept map in a separate window while the original glossary list is displayed in the original window and the selected term is highlighted. The term definition is also included in the concept map new window.

D3 provides many graph layout algorithms. We have chosen to use the *graph force* layout algorithm. This allows students to drag and drop nodes within the window, allowing them to be moved while the nodes remain connected to each other. This, in turn, will allow students to better examine their concepts of interest in more detail, especially if the graph has a large number of nodes. The main concept is excluded from this drag-and-drop action so it remains

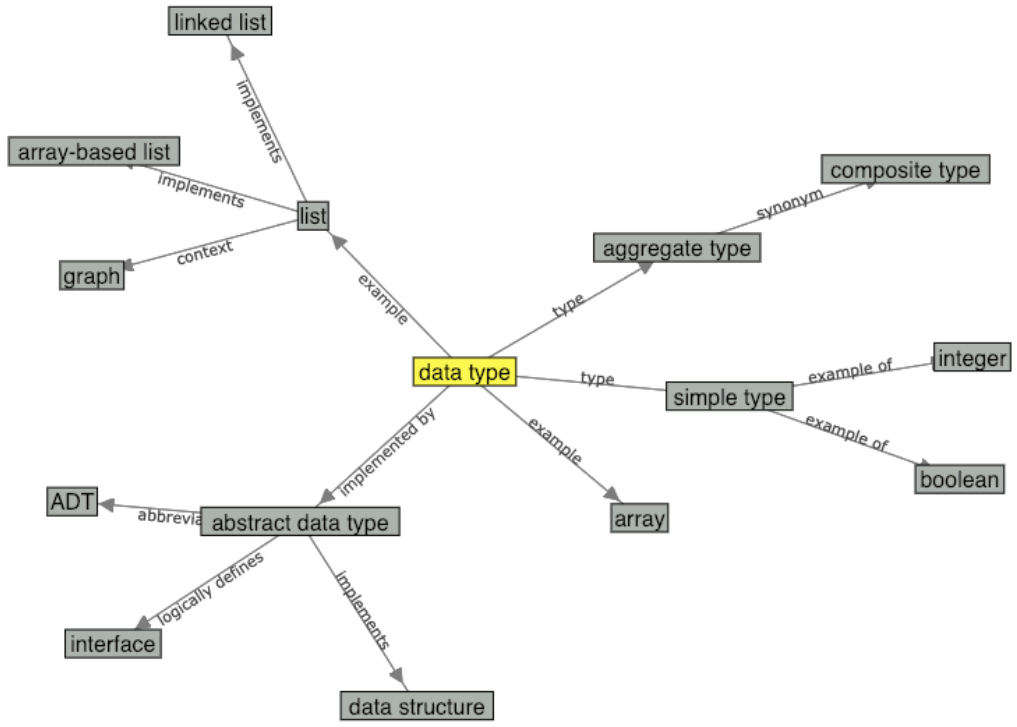

Figure 3.10: The generated concept map for the glossary term “data type”.

in the center of the window. We choose to display the main concept in a different color to distinguish it from the other nodes. Figure 3.10 shows the generated concept map for the glossary term “data type”. Notice that the main concept has a different color. Links are directed from the main concept to the other concepts, with the edges having labels. The *force graph* layout generates a visually appealing format.

3.5.1 Force graph actions

We take advantage of several features provided by D3. The first is zoom, which allows students to zoom in or out in the graph. To provide smooth zoom, we implemented the graph to zoom in or out as a whole, like a static image. The graph returns to its dynamic

nature after the student releases the mouse and stops zooming. This allows fast zoom in or out.

Our implementation uses click actions on the nodes. Students can select any node in the concept map by clicking on it. This node will highlight when the mouse hovers over it, to indicate that it is clickable. If this node has its own relationships, then its concept map will be generated and displayed in the current concept map window.

3.5.2 Concept map hierarchical levels

An important design choice for concept maps is selecting the number of hierarchical levels to be displayed for each node. This is complicated be the fact that the number of nodes connected to each concept varies drastically. Therefore, there is a trade-off to consider. On one hand, increasing the number of levels will allow more nodes to be displayed, and, hence, more information. On the other hand, if the number of nodes become too large, it will hinder understanding the relationships.

For example, if a node has a lot of directly related nodes, it might be enough to display only its first level of connections. But if the number of immediate neighbors is relatively small, we can then explore more by displaying concepts further away in the map.

In the current implementation, we have adjusted the number of levels under each node to be two. This means the connections coming out from a specific node are limited to two levels. We have selected this number of levels since usually it displays enough nodes that are related to each concept without causing a lot of visual noise to the students. A future direction we want to explore is to automatically adjust the number of the displayed nodes in the graph by allowing the implementation to decide the number of levels to be displayed. In this sense, more levels can be generated depending on the whole number of nodes displayed.

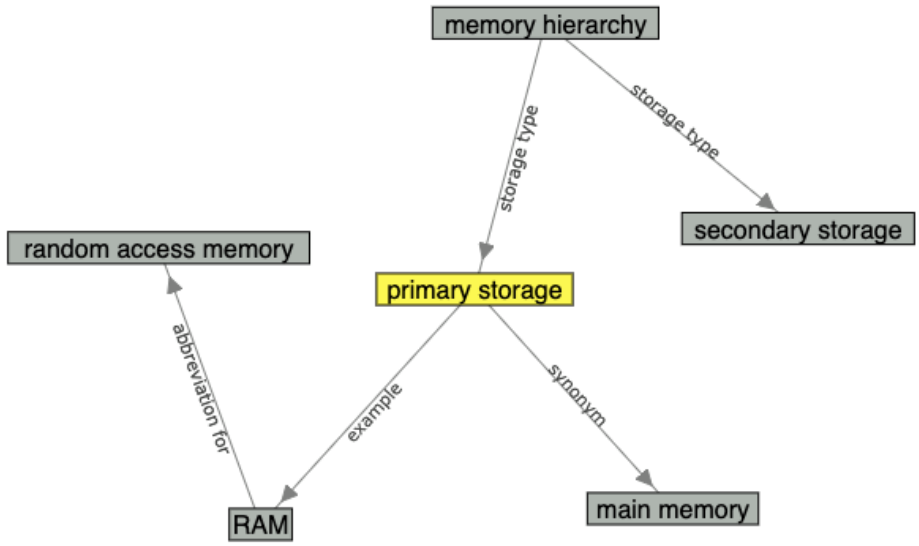

Figure 3.11: Backward level for the concept “primary storage”.

In Figure 3.10, two layers of hierarchical nodes are displayed for the concept “data type”. Another, more dense example, is the concept “graph”, which was shown in Figure 3.2. We can see that at one level, more than one node can be connected to a single node in the next level, e.g., the node “directed acyclic graph” has two nodes pointing towards it from its previous level. This helps students to better relate the information between different concepts.

3.5.3 Backward levels

We notice that some concepts are better understood when their previous level nodes, i.e., nodes that point towards them can also be seen.

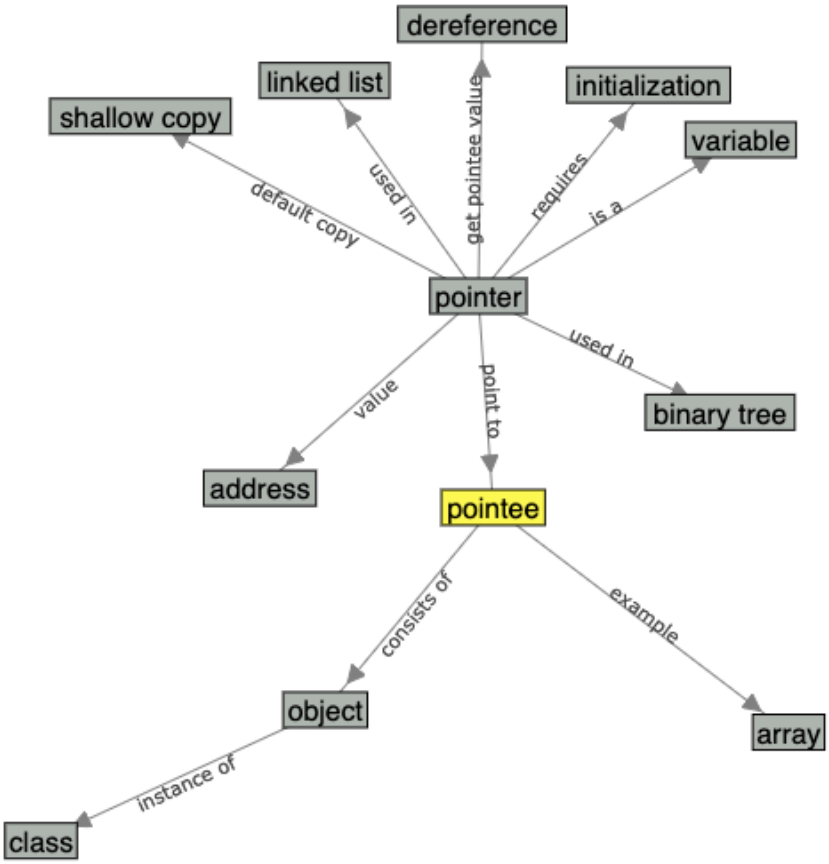

Figure 3.12: Backward level for the concept “pointee”.

In the current version of our implementation, we use two levels of connected nodes with one backward level. The backward level is chosen to be displayed in certain graphs to support the idea behind their main concepts, or when the total number of nodes in the graph is below a threshold. In Figure 3.11, the number of directly connected nodes to “primary storage” is just two nodes, one with connection to a node in the next level. In this case, the backward level is displayed and can be distinguished by the arrows pointing towards the main concept.

In Figure 3.12, it was important to display the parent node of the concept “pointee”, which is “pointer”. We think this will help students to better understand the relationship between

"pointee" and "pointer", and not to mix them with each other. Note that when "pointer" is displayed, all its first level neighbors will also be displayed.

Chapter 4

Concept Maps Evaluation

After designing and implementing the concept maps for glossaries, we tested and evaluated the effectiveness of our work by measuring students interactions with the new glossaries after including the new design. To this end, we defined the following research questions to help determine our evaluation approach:

- How can we effectively make the concept maps noticeable for students? In some sense, how can we advertise the implementation?
- What data should we collect to represent effective student interactions with the glossaries?
- Based on the data collected, how effective are concept maps versus standard glossaries?

In the following sections, we will answer these questions and present the corresponding results. We start with the first question, which is our approach to advertise our work for students using the digital courses. As mentioned earlier, our concept maps were integrated into the CS2114 and CS3114 courses at Virginia Tech. However, if a student is not used to checking the glossary of a digital textbook, he/she will not notice the newly included concept maps. Therefore, we decided to design special exercises that refer to the concept maps within the course contents in order to make the students familiar with the available concept maps while also making them more familiar with these specific concepts. These exercises are discussed in detail in the next section.

4.1 Concept map Exercise Design

The following exercises were included in the CS2114 courses but they were not used to evaluate the students and so did not affect the course grade. The functional purpose for these exercises was to make students aware of the available concept maps, and hopefully to ignite their curiosity to explore more about the concepts in the questions, and, hence, use the concept maps. Students could tell that these exercises were not graded, however, almost all students tried to solve these exercises.

To build the exercises, we used the Khan Academy infrastructure [38] that is integrated with the OpenDSA [10] system. The Khan Academy exercise framework provides a robust infrastructure that can be used to create questions including ones of the types are True/False, multiple choice, and Fill-in-the-Blank.

In order to make the exercises quick for students, we used mainly True/False questions and multiple choice questions. The designed exercises include a static image of the concept maps for the glossary term found in the exercise. For example, Figure 4.1 shows a multiple choice question.

Another example for a True/False question can be seen in Figure 4.2. We notice that in both figures, the same question header is used to refer to the concepts maps that can be found in the glossary page. After selecting the answer, the student needs to click on “Check Answer” on the right side of the question.

In case the student made a wrong selection, a message is displayed that the answer is not correct and advises the student to try again. Meanwhile, the button “Show hints” changes its color to let the student know that hints are available for this question. In this case, we chose a fixed message to be displayed that asks the student to check the full concept map to get more information.

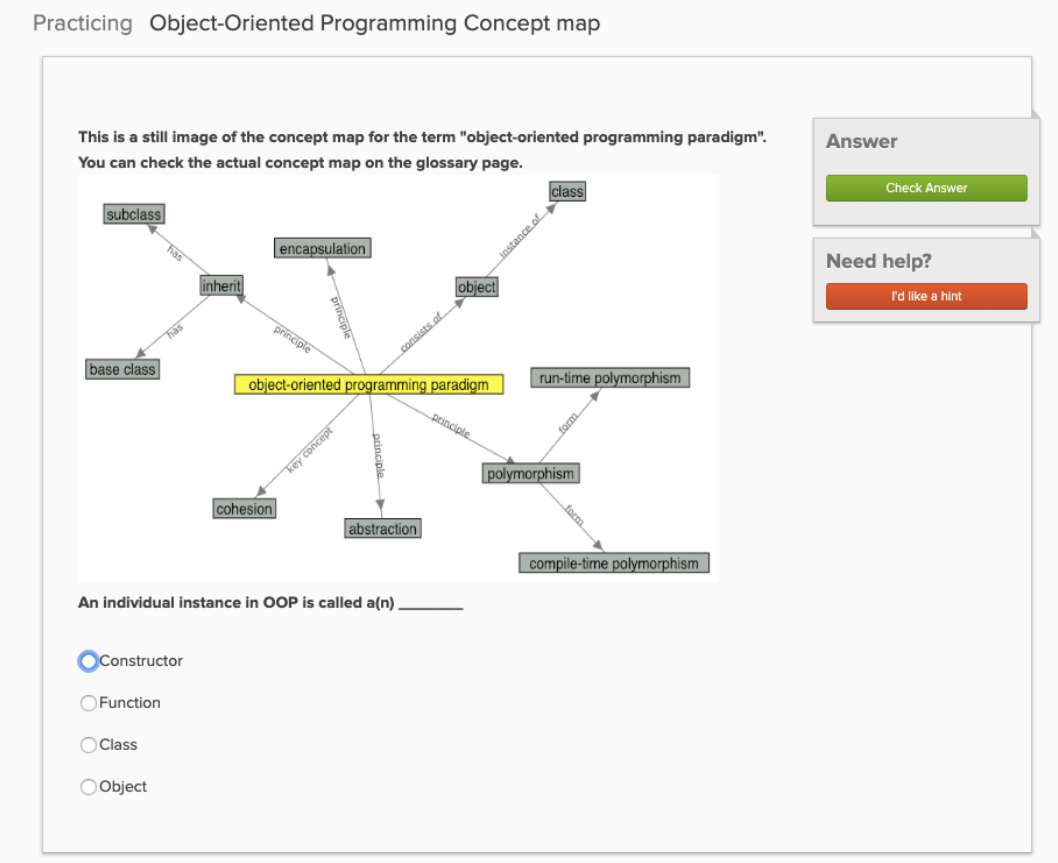

Figure 4.1: An example of a multiple choice involving a concept map

In designing these exercises, we took into consideration that these exercises are not meant to evaluate the students. Hence, the questions were not designed to be complicated or to require further exploration from the students. The answers of these questions only require the students to study the picture to identify the correct choice among the given answers. We believe that this method of representation and interaction with the glossary terms will help students to be familiar with the concept maps, and hence, they will start using the feature.

Finally, we note that we have implanted these exercises in CS2114 only and not in CS3114. This lets us to measure whether these exercises were helpful in directing the students towards exploring more about the concept maps by visiting the glossary page.

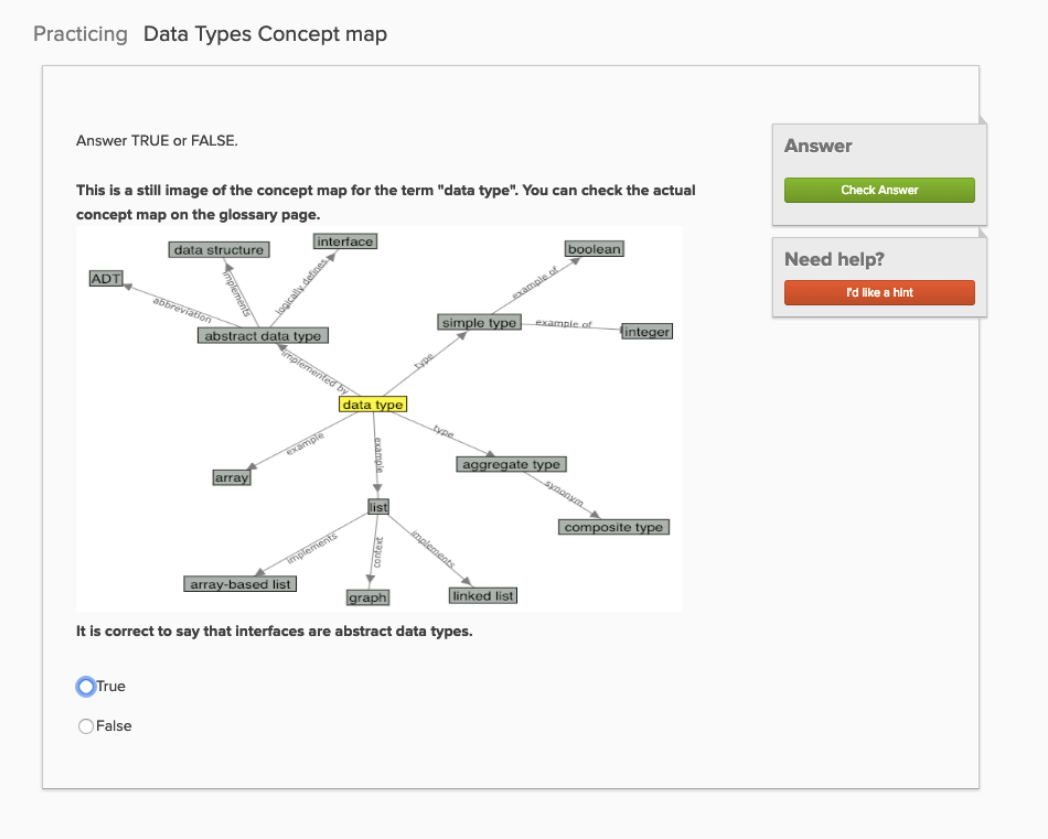

Figure 4.2: An example of a True/False including concept maps.

4.2 Data Collection

Both CS2114 and CS3114 make use of OpenDSA content. Besides the client-side content that students use and interact with, OpenDSA uses a server side implementation to register and store any events or logs of interest. The OpenDSA server [39] is implemented using Ruby on Rails.

OpenDSA logs student use of many features, with all interactions logged into a database. This allows us to study student use of specific features within the system, such as the number of interactions with the glossary, visualizations, exercises, etc. This was helpful in collecting data about student use of the original glossaries in the previous semesters, as this

was already stored in the database. The same was applied for this semester and also the students' actions were stored in the database. We used simple SQL queries to retrieve from the database information about students' use of the glossary.

Next, we preform statistical analyses on these collected data to measure the effect of implementing the concept maps on the students' use of the glossaries.

4.3 Results

We now present the data on student use of the glossary before and after activating the concept maps. In particular, the previous three semesters (Fall 2017, Spring 2018, and Fall 2018) were used to compare with the current semester (Spring 2019). First, we start with a visual display for the students' use of glossary in CS2114 in the Spring 2019 semester. Figure 4.3 shows the number of times each student, defined by user id, has used the glossary page so far. The numbers of uses are sorted from the smallest to the largest. Student use varies, with the maximum number of glossary accesses around 173 times. The second highest count for glossary use is around 150 times and the minimum is 1. Figure 4.3 displays only the students who have used the glossary (84.4% of students in the class).

Figure 4.4 shows the number of times each student in CS3114 in Spring 2019 has used the glossary page. Similar to Figure 4.3, students are identified by user id. The maximum number of glossary uses is around 340 times (two students had this level of use). The next highest count for students' use is around 160 times. Since we only show data for students who used the glossary (62.3% of students use), the minimum number of uses is 1. Information about students who have and have not used the glossaries is discussed next.

The fraction of students who have used the glossary in Spring 2019 semester is shown in

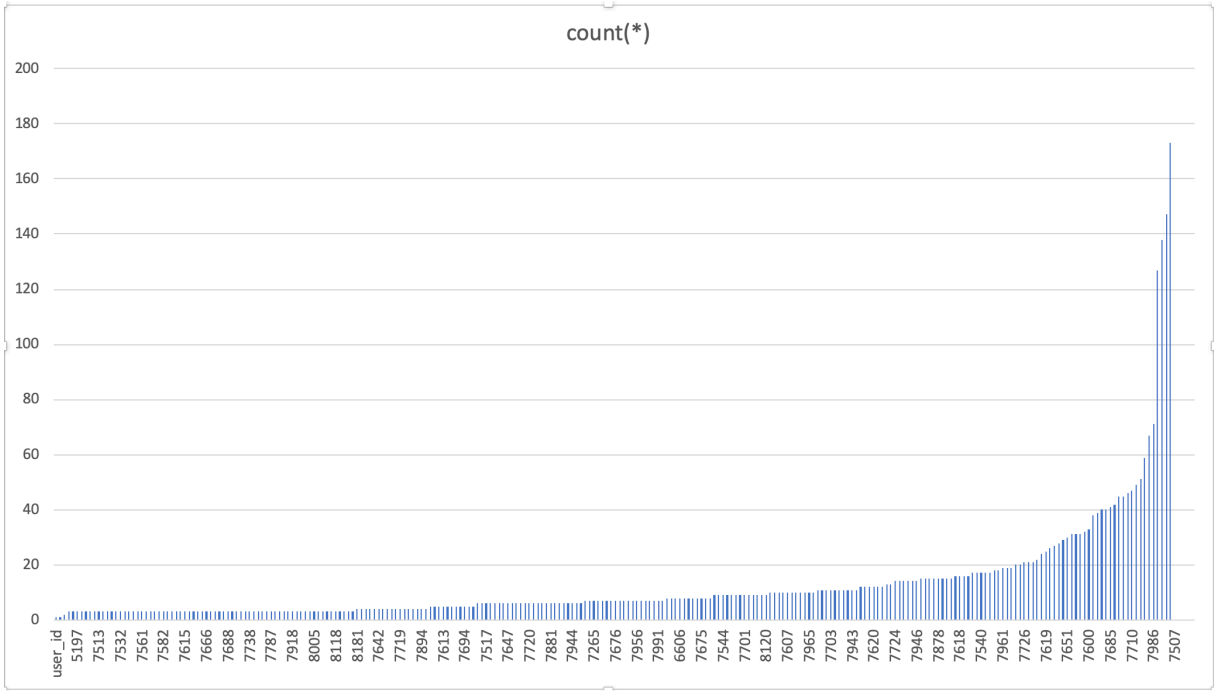

Figure 4.3: Student use of the glossary in CS2114 in the Spring 2019 semester (the median is 7).

CS2114	Fall 2017	Spring 2018	Fall 2018	Spring 2019
Total number of students	363	435	380	308
Count of students who used the glossary	121	179	223	260
Percentage	33.33%	41.1%	58.6%	84.4%

Table 4.1: Students' use of the glossary compared to the total number of students in CS2114.

Table 4.1. The percentages for the three previous semesters, i.e., Fall 2017, Spring 2018, and Fall 2018, are also shown in Table 4.1.

We can notice that the percentage of students who have used the glossary in Spring 2019 semester, i.e. after implementing the concept maps, is higher than the previous three semesters. In Spring 2019, about 84% of students have used the glossary compared to an average of 44% in the previous semesters.

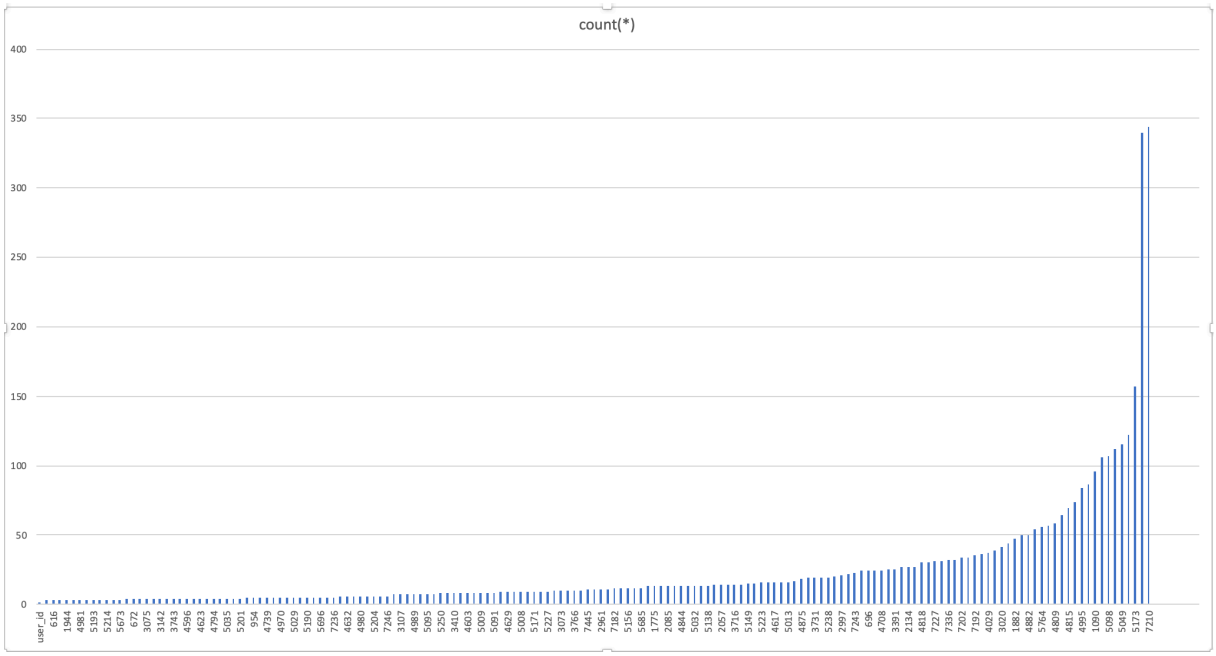

Figure 4.4: Student use of the glossary in CS3114 in the Spring 2019 semester.

CS3114	Fall 2017	Fall 2018	Spring 2019
Total number of students	295	387	268
Count of students who used the glossary	150	209	167
Percentage	50.8%	54%	62.3%

Table 4.2: Students' use of the glossary compared to the total number of students in C32114.

Similarly, in Table 4.2, we show the same percentages for CS3114. Note, in this case the students' use in Spring 2018 was not available, and, hence, the comparison used only the other two semesters. Similar to Table 4.1, that the percentage of students who have used the glossary has increased in the Spring 2019 semester, i.e., after implementing the concept maps, compared to all the previous semesters. In Spring 2019, about 62% of students have used the glossary compared to an average of 52% in the previous semesters.

We notice that the student use of the glossary in CS2114 is higher than that in CS3114. However, we must consider the possible effect of the concept map exercises on student use

of the glossary in CS2114. The key question then becomes: If the exercises related to an increase in use of the glossary, does this mean that (a) the increase in glossary use is mostly related directly to completing the exercises themselves, and otherwise there was no real increase in glossary use, or (b) the exercises trained the students to use the glossary, after which they continued to do so?

We found that about 18% of the total number of glossary uses happened while students were solving the exercises. Since this percentage represents a significant part of the total use, we show the results of CS2114 for the total use, and also after excluding the exercises effect.

We start by showing, in Figure 4.5, the number of times each student has used the glossary page; after excluding the concurrent use of the exercises. We notice that the maximum number of glossary uses has dropped to around 160 times, compared to 173 before excluding the exercises effect. The minimum in this case has dropped to 0 which happened for 10 students. This means the number of students who have used the glossary only while solving the exercises is 10.

Next, we study the number of students who have used the glossary more than once. The previous results show absolute percentage even if a student used the glossary only once. When a student uses the glossary more than once, this can be interpreted as the glossary was useful for him, so he decided to use the feature again. In the following, we counted the number of students who have used the glossary more than 5 times, more than 10 times, more than 20 times, and more than 30 times.

Figure 4.6, shows a comparison between the four semesters for CS2114. Each column in the Figure 4.6 shows the percentage of students who have used the glossary, for the number of times shown, to the total number of students who have used it in this semester. Each semester is represented with a column with the Spring 2019 represented twice, once with the

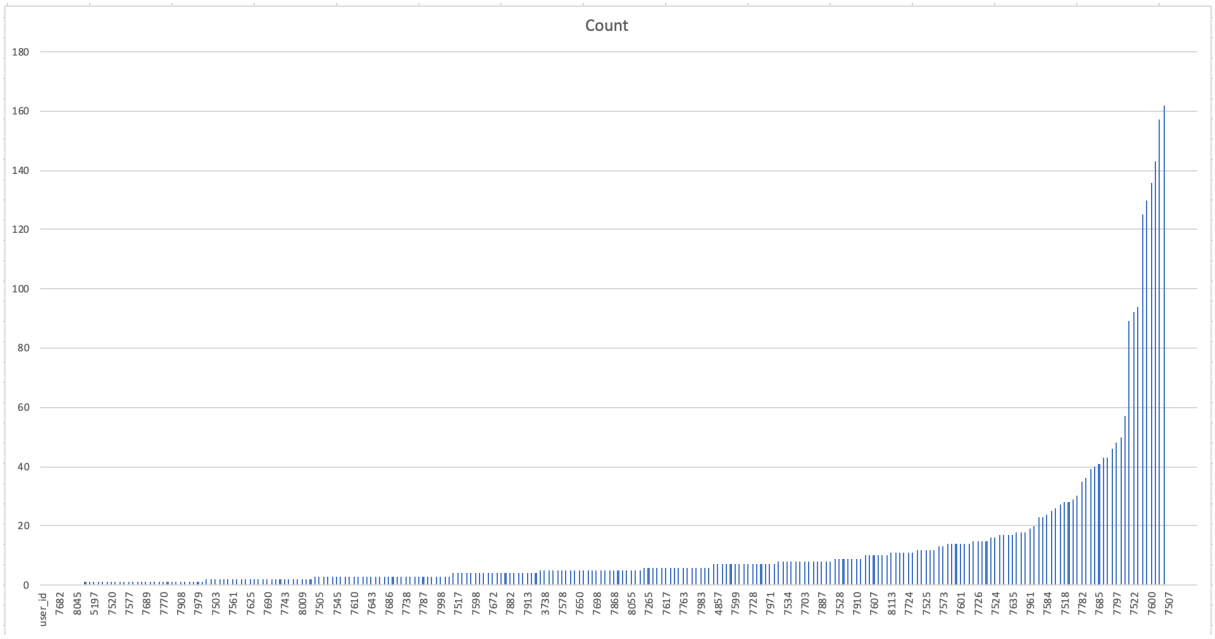

Figure 4.5: Student use of the glossary in CS2114 in the Spring 2019 semester after excluding the exercises direct effect.

complete data, and the other after excluding the direct exercise effects. We will discuss the results after excluding the exercises effect, however, the complete data are also shown as a reference.

In Figure 4.6, we can notice that out of the 260 students who have used the glossary in CS2114, 145 students have used the glossary more than 5 times (after excluding the exercises effect). This makes the percentage about 55%. The remaining numbers of use are calculated similarly. In 4.6, we notice that the percentages of use in Spring 2019, after excluding the exercises effect, are higher than the percentages in Fall 2017 and Spring 2018, for all numbers of use. However, in Fall 2018 the percentages are higher for 20 and 30 times of use and are comparable for 5 and 10 times of use.

Similarly, in Figure 4.7, we study the percentage of students who used the glossary multiple

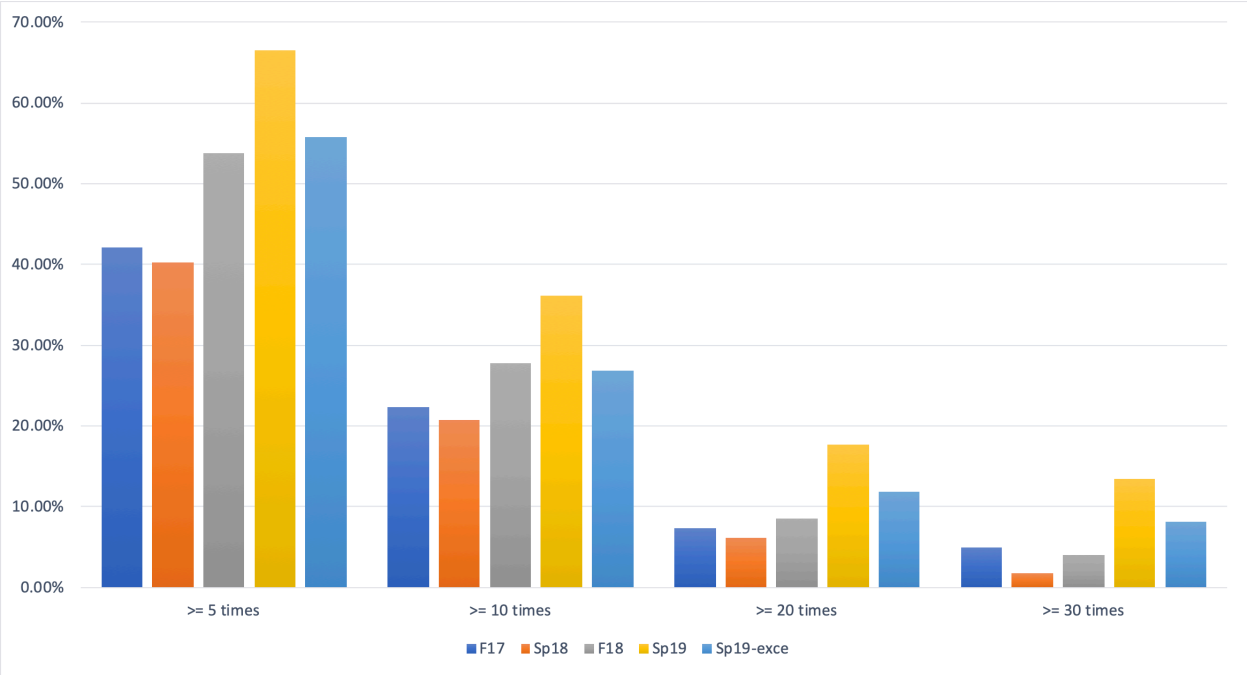

Figure 4.6: Percentage of students who have used the glossary multiple times in CS2114. Notice that the last two bars show use in Spring 2019; the yellow bar includes uses to complete exercises while the blue bar shows only uses unconnected with the exercises.

times between Spring 2019 and the previous two semesters for CS3114. In this case, the percentages in Spring 2019 are also higher than the percentages in the previous semesters, for different users' counts. However, the percentages of Spring 2019 are comparable to the percentages of Fall 2018.

For both CS2114 and CS3114, these results give strong indication that students who have tried the new concept map glossary decided to the use the glossary more than the previous semesters. However, we note that the percentage increase of students who have used the glossary in general and those who have used it multiple times, is higher in CS2114 than CS3114, with respect to the previous semester. One difference between CS2114 and CS3114 is the included concept map exercises, so it could be that using our exercises has affected the results positively.

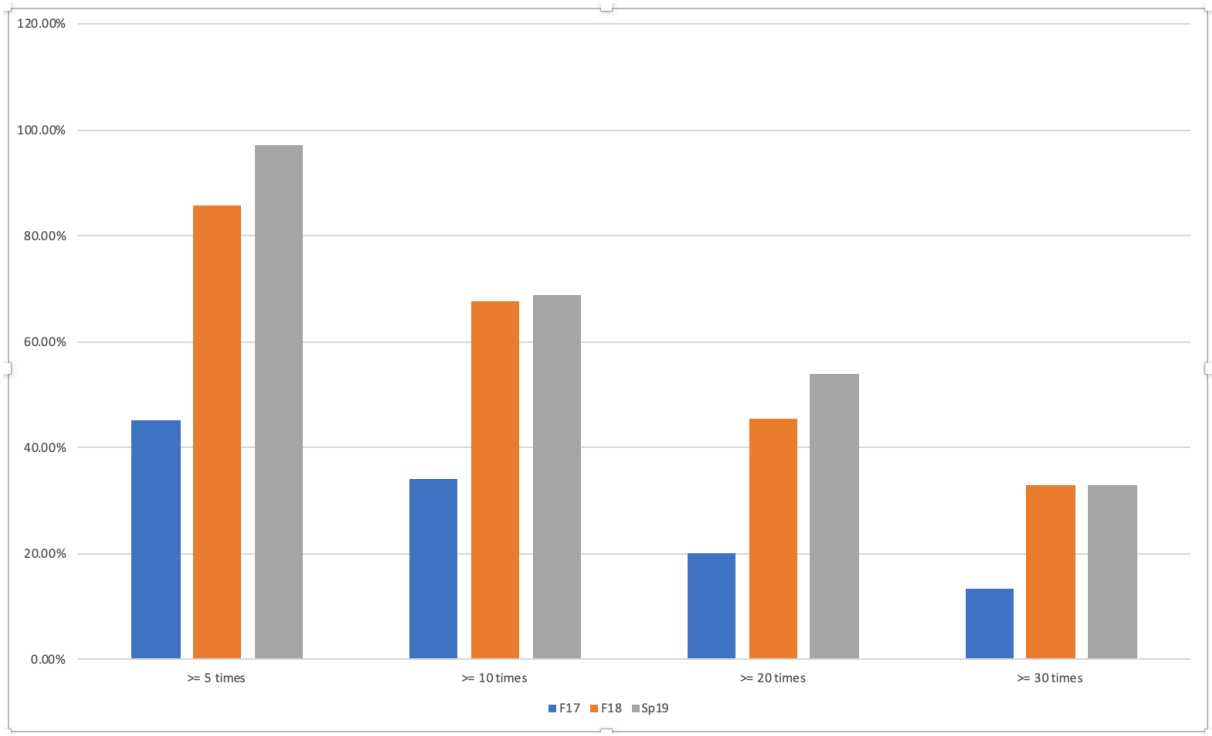

Figure 4.7: Percentage of students who have used the glossary multiple times in CS3114.

Next, we perform in-depth statistical analysis of the previous data to find if there are significant changes after implementing the concept maps.

4.4 Statistical Analysis: ANOVA Test

The goal of this section is to provide an in-depth comparison though statistical analysis of the students' use of the glossary between the different semesters. We start with the relations between the means and variances in each semester.

Figure 4.8 shows the interval plot for CS2114 complete usage data. There are four groups representing the four semesters under test. Figure 4.8 shows that the means are not equal. Figure 4.9 shows the interval plot for CS2114 using the updated students' use after excluding

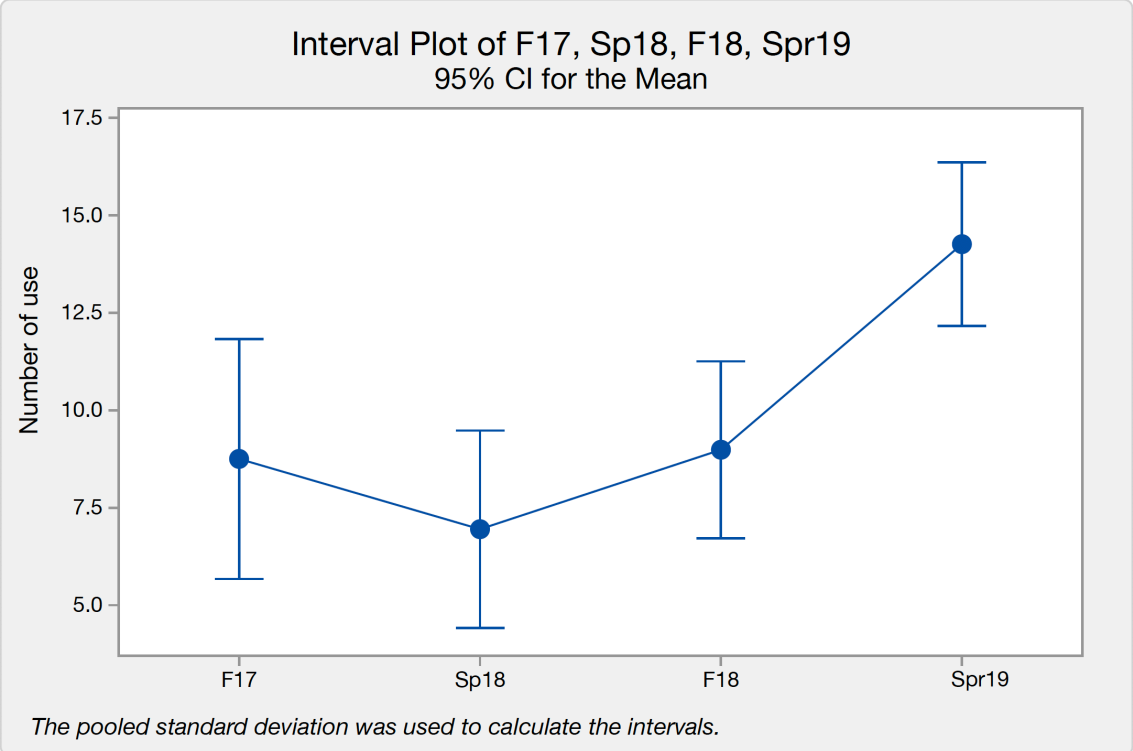

Figure 4.8: Interval plot for the means and variances in CS2114.

the exercises direct effect. We notice that the means again are not the same. The mean of the updated use in Spring 2019 is still higher than the previous semesters (but of course it is lower than the complete data case).

Similar results are shown in Figure 4.10 for CS3114. In this case, there are only three groups representing the data from the available semesters.

More importantly, we perform a statistical analysis to check whether the differences in the means are significant. Since the collected data represent the student use of the glossary in different semesters, we want to use a statistical test that can compare the different semesters and determine if the results of Spring 2019 are significantly different from the other semesters. Therefore, we use one-way ANOVA [40] and [41] on the CS2114 and CS3114 data.

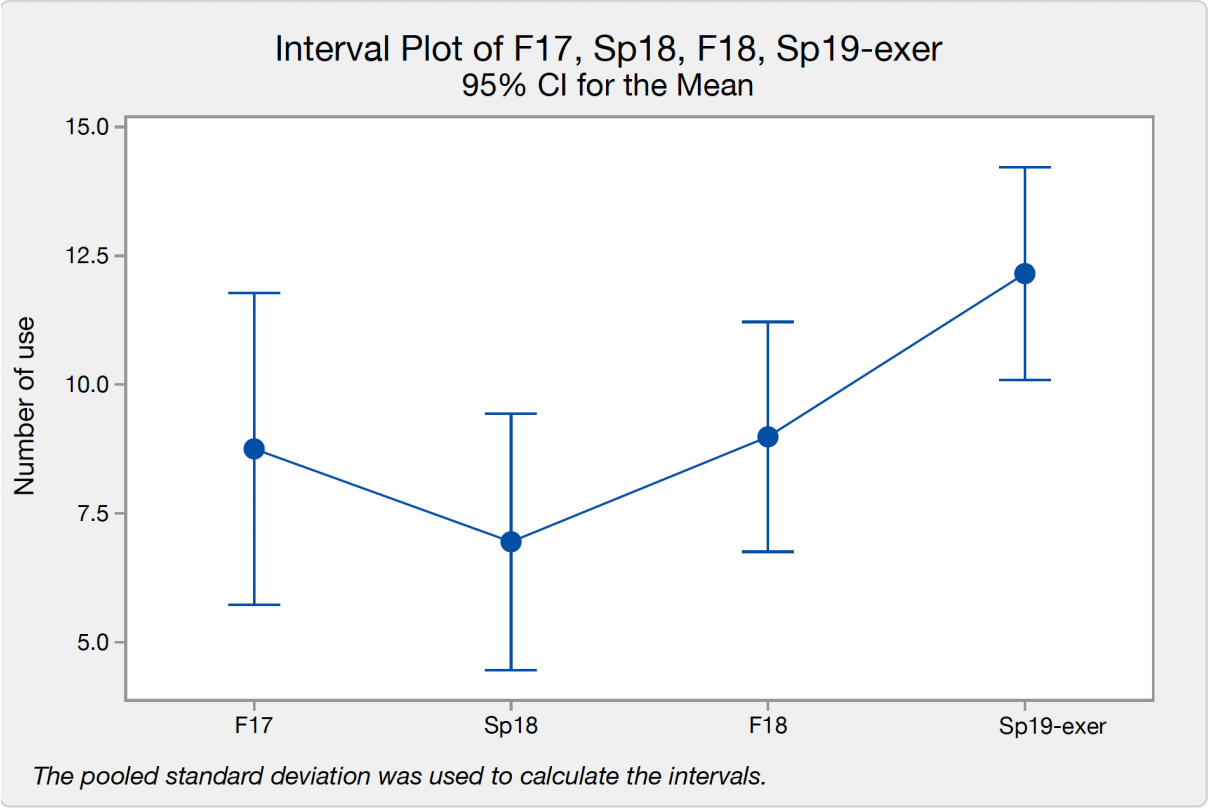

Figure 4.9: Interval plot for the means and variances in C2114 after excluding direct exercises effect.

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Factor	3	6753	2250.99	7.57	0.0001
Error	779	231566	297.26		
Total	782	238319			

Table 4.3: One-way ANOVA test results for CS2114 complete data.

In the following tests, we use the NULL hypothesis as H_0 : All means are equal and the alternative hypothesis H_{alt} : At least one mean is different. The significance level used in the following experiments is $\alpha = 0.05$, which is the probability of rejecting the null hypothesis when it is true.

We show the results of the one-way ANOVA for CS2114 complete data in Table 4.3. We notice that the P-Value = 0.0001 which is much less than the significance level α. Therefore,

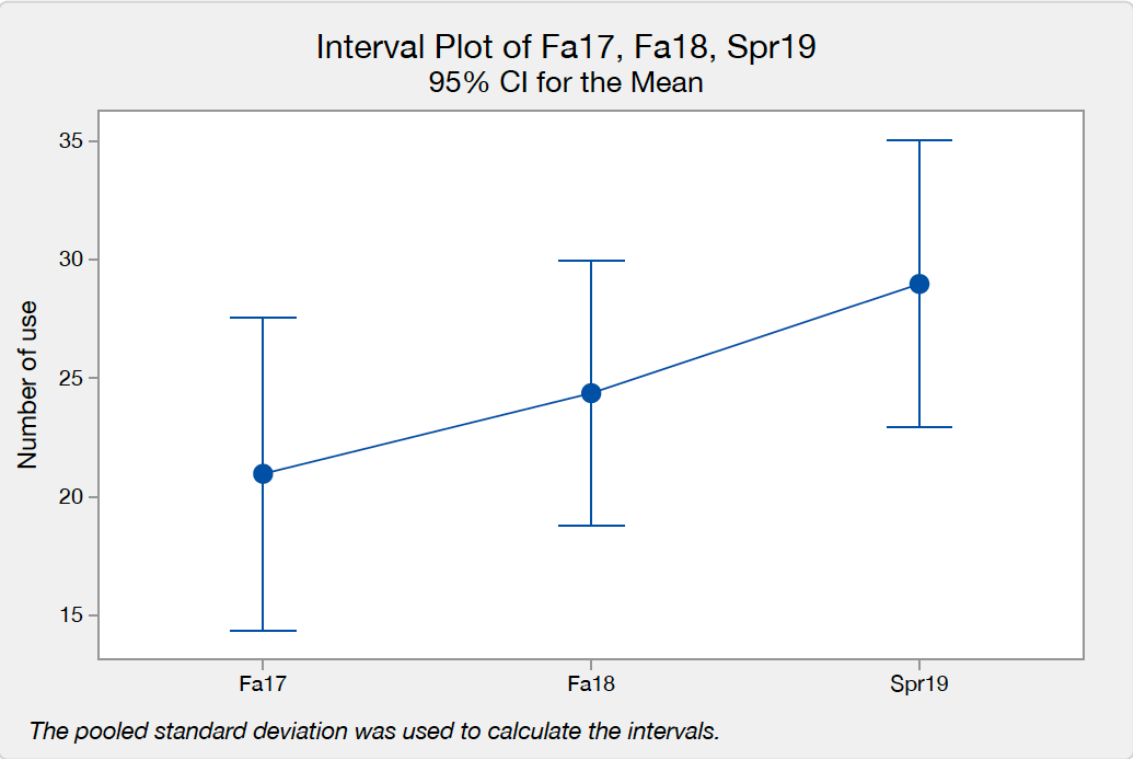

Figure 4.10: Interval plot for the means and variances in C3114.

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Factor	3	3121	1040.27	3.61	0.0131
Error	779	224392	288.05		
Total	782	227513			

Table 4.4: One-way ANOVA test results for CS2114 after excluding the exercises direct effect.

we can reject the null hypothesis (that all means are equal) and we conclude that a significant difference exists between the different semesters of CS2114 course.

Next, we perform the one-way ANOVA test on the updated students' use, after excluding the exercises, in Spring 2019 compared to the previous semesters. The results of the ANOVA test are shown in Table 4.4 where we can see that the P-Value in this case = 0.01314 which is also less than the significance level $\alpha = 0.05$. Therefore, we can also reject the null hypothesis

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Factor	2	5329	2664.47	1.58	0.2077
Error	534	902755	1690.55		
Total	536	908084			

Table 4.5: One-way ANOVA test results for CS3114.

and we conclude that a significant difference exists between the different semesters of CS2114 course after excluding the exercises direct effect.

Similarly, in Table 4.5 , we show the results of the one-way ANOVA test for CS3114. The P-Value in this case = 0.2077 which is higher than the significance level α. Therefore, we cannot reject the null hypothesis (that all means are equal) and we conclude that there is no significant evidence that the means are different.

One difference between CS2114 and CS3114 is that the exercises were only given to CS2114 students. So the exercises probably helped to increase the level of glossary use in CS2114. Although the percentages of use in CS3114 are higher after implementing the concepts maps, as was shown in Figure 4.7, these differences were not significant.

After determining that there is a significant difference between the means in CS2114, we need to perform follow-up tests on the means to check which semesters actually differ from the others. In the next section, we perform *multi comparison* tests to check whether the concept maps in Spring 2019 makes a significant differences from the other semesters.

4.5 Statistical Analysis: Multi-comparison

The multiple comparison test can be used only when the ANOVA test shows that there is a significant difference between the groups. Therefore, we apply these only to the CS2114

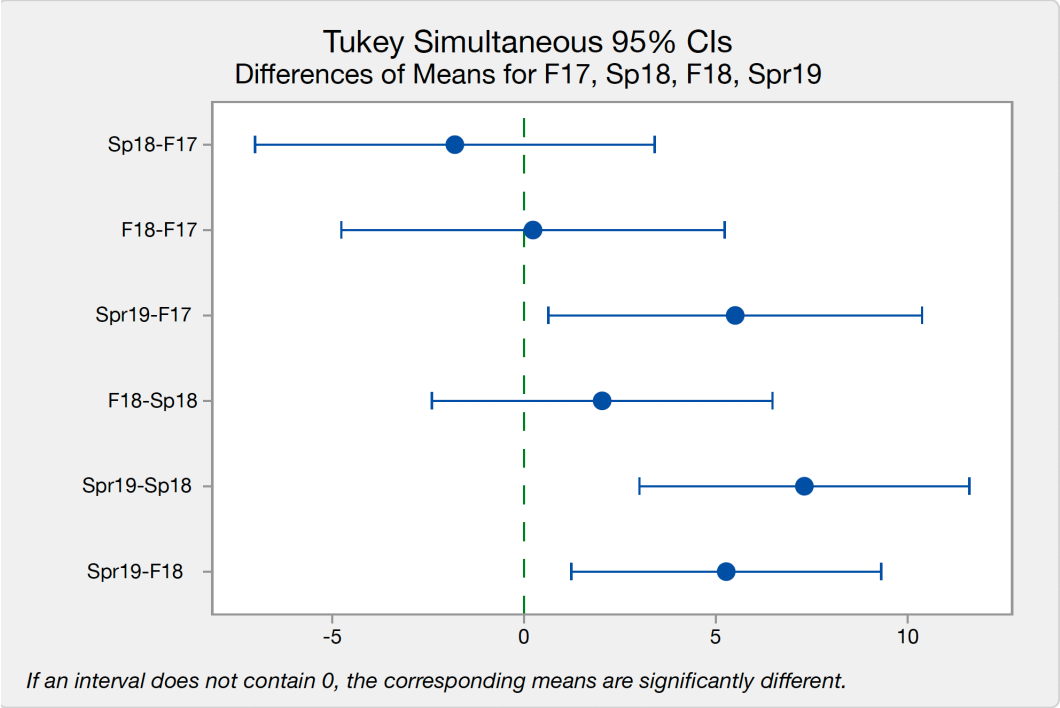

Figure 4.11: Tukey multi-comparison test for CS2114 complete data.

course for both the cases of complete data and after excluding the exercises direct effect. In general, there are two types of multiple comparison tests: pair wise comparisons such as Tukey test, and non-pairwise comparisons. Moreover, the non-pairwise comparisons can be done with a control group, such as the Dunnett test, and without a control group, like the Bonferroni test.

In the following, we will perform both the Tukey test (pairwise comparisons) and the Dunnett test (non-pairwise with a control) for CS2114 course. We start by showing the Tukey test results in Figure 4.11 for the complete data case. We notice that there are, in total, six pairwise comparisons between the four groups (semesters). All the intervals that contain zero are assumed not to be significantly different. We notice that three intervals do not include zero, and, hence, their groups have significant differences in their number of use. The differences are between the Spring 2019 semester and all the previous three semesters.

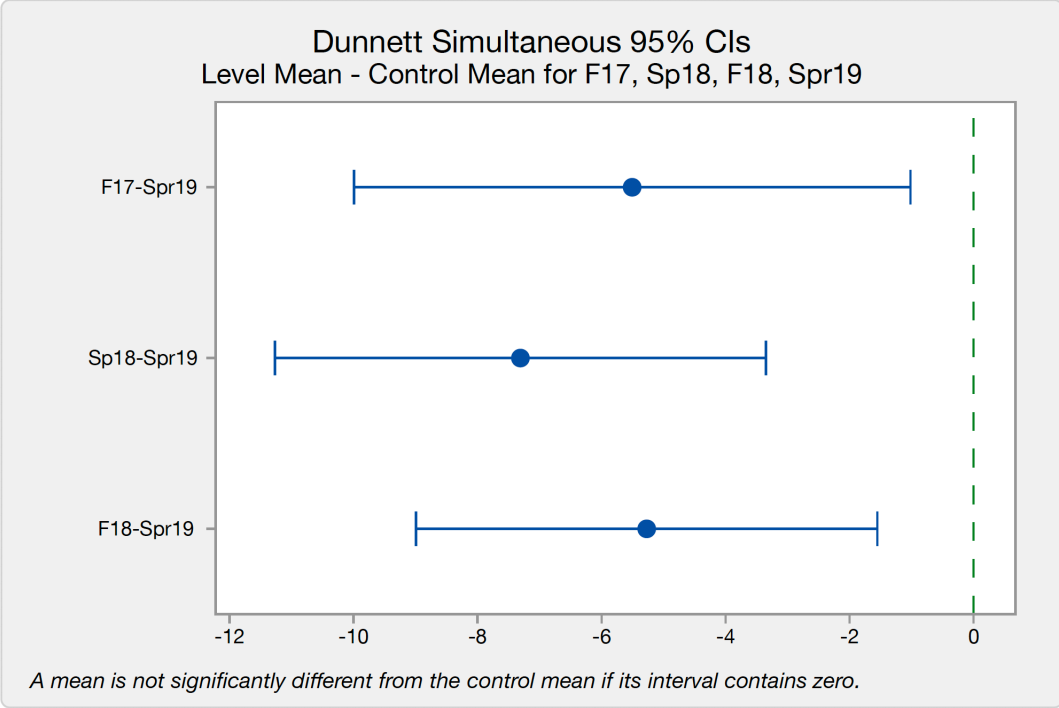

Figure 4.12: Dunnett multi-comparison test for CS2114 complete data.

These results corroborate the ANOVA results that there is a significant difference between the means.

Next, we show the results of Dunnett test with the Spring 2019 semester is selected as the control group. Figure 4.12 shows the results of the Dunnett test for CS2114 complete data. The results are similar to the Tukey test. Spring 2019 has a significant difference from all the previous three semesters.

Next, we perform the multi-comparison for the updated CS2114 results, after excluding the exercises effect. The Tukey test is shown in Figure 4.13. We notice, in this case, only two groups have a significant difference. These groups are the updated Spring 2019 and Spring 2018. We notice that, Spring 2019 is now significant to only one previous semester compared to three semesters when considering the complete data.

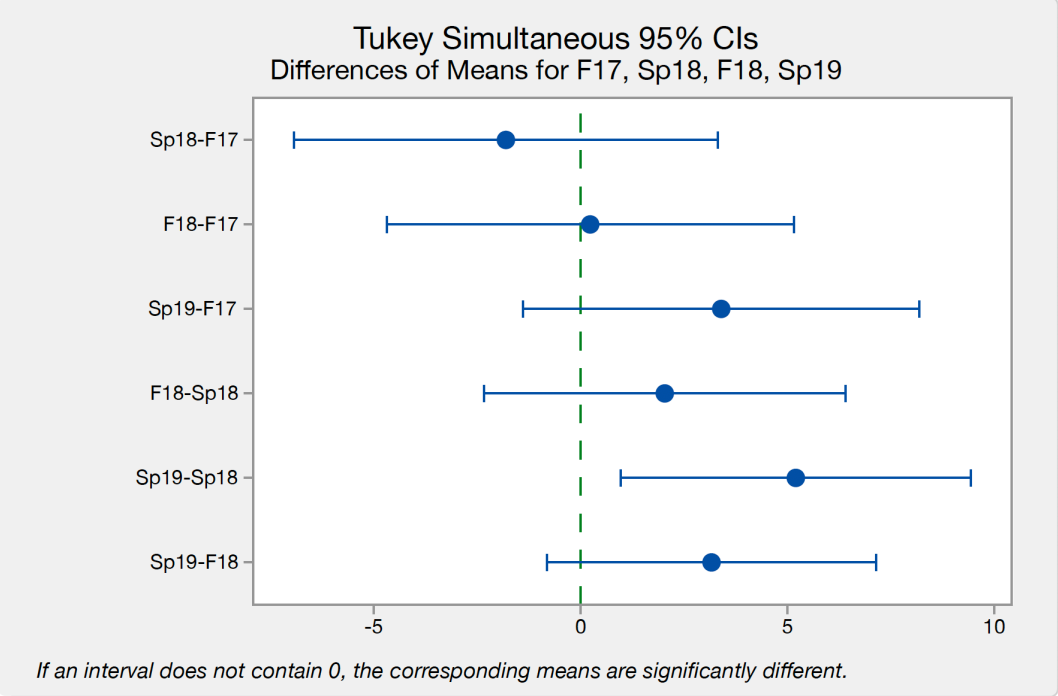

Figure 4.13: Tukey multi-comparison test for CS2114 after excluding the exercises direct effect.

Next, we show the results of Dunnett test with the updated Spring 2019 semester is selected as the control group. Figure 4.14 shows the results of the Dunnett test for CS2114 complete data. The results are similar to the Tukey test. Spring 2019 has a significant difference from Spring 2018 semester.

Given the significant differences found by the multi-comparison tests, and the results in Figure 4.6 that shows higher means percentage use in Spring 2019, we can conclude that students use of the glossary has increased significantly in the Spring 2019 semester. Since this increase happened immediately after implementing our concept maps and nothing else changed that we are aware of to explain this, it is plausible that the concept maps helped to increase the students' use of the glossary in CS2114 course.

We have already noted that there was no significant increase in glossary use for CS3114,

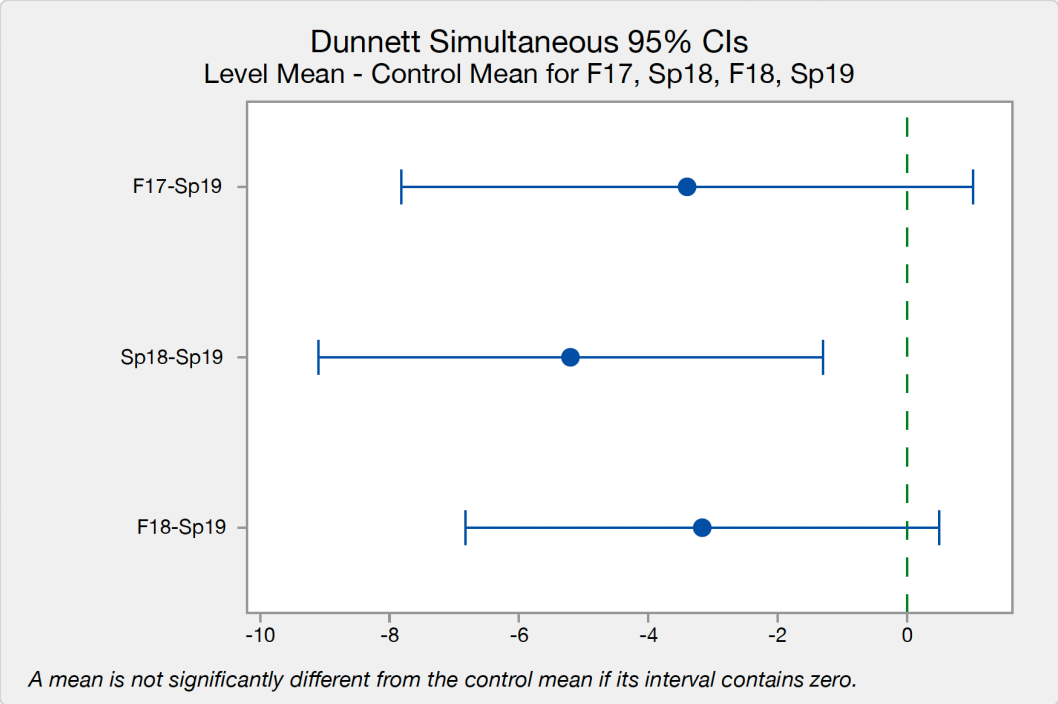

Figure 4.14: Dunnett multi-comparison test for CS2114 after excluding the exercises direct effect.

where there was no use of the exercises, and so no ”advertising” of the glossary. So it seems likely that the exercises was successful in drawing the students’ attention to the concept maps, and, hence, the glossary.

Chapter 5

Narration Support

5.1 Introduction

In this chapter, we present our work on narration support. In particular, we have designed and implemented a tool to automatically generate audible content from the textual content associated with slideshows. The main goal behind this approach is to stimulate more learning channels in students' brains by combining both visual and audio contents. This combination is one of the basic pillars of the the cognitive theory by Mayer [1], which is known as the "Dual Channel Assumption".

According to Mayer, the cognitive system in humans consists of two different channels that are responsible for representing and processing the knowledge. The first channel deals with visuals and pictures and represents any input to the cognitive systems that comes through the eyes. The second channel deals with audio and verbal formats and represents any input to the cognitive system through the ears. Thus, adding narration support to the digital courses is expected to help students' brains to better process the information, and, hence, it will increase students' comprehension. To this end, we defined the following research questions to determine our design and implementation approach:

- What contents, in a digital course, can make use of the narration support? In other words, should the narration be added to all the course materials?

- How can student control the narration? What are the basic controls that will be used?

- How can the developed tool integrate smoothly with our current eTextbook frameworks?

In the following, we will answer these research questions. We start with the first question, which pertains to the contents that will be converted into audible format. Recently, there have been many platforms that support audio books such amazon audio books [42] which integrates the service from Audible. The idea of these audio books is to provide a visual-free format that is easier to follow when a person cannot or does not like to read from a regular book. This can be useful, for example, if someone is driving a car. However, in digital courses, the goal is completely different. Our goal is not to replace the textual format with an audible format, instead, the audible contents should be used in addition to the visual contents. Hence, the decision was to implement the narration support tool only where it can support the goal of increasing students' comprehension of mixed text and visualization material, and not on primarily textual course contents.

To this end, we decided to implement the narration support tool only within the OpenDSA slideshows. These require a visual interaction with the contents in order to perform tasks such as make selections, click objects, start/ stop animations, or answer questions. Thus, the narration support, when added to such content, will not be a substitute format. Instead, it will be used simultaneously with the visual contents.

Most interactive content in OpenDSA is designed using the JSAV graphics library. In particular, the OpenDSA eTextbooks for data structure and algorithms courses such as CS2114 and CS3114 have many interactive algorithm visualizations using JSAV. More about JSAV and its basic functionalists is discussed next.

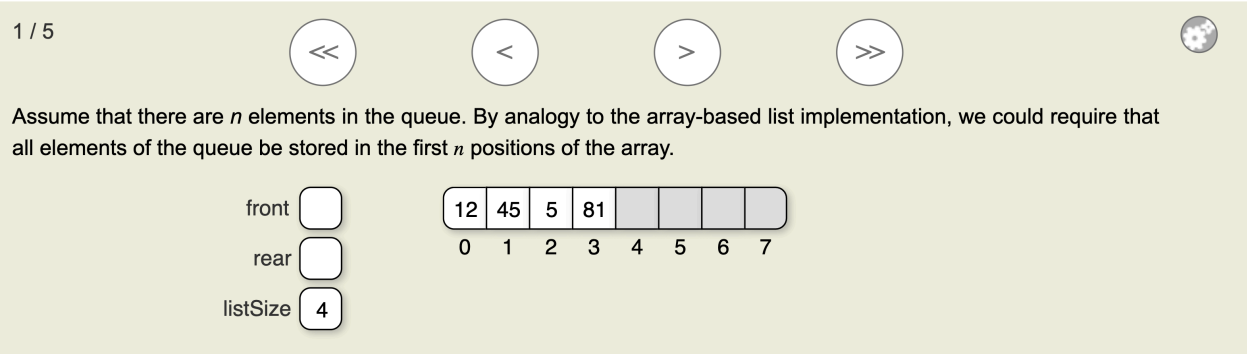

Figure 5.1: A typical interactive algorithm visualization created using JSAV.

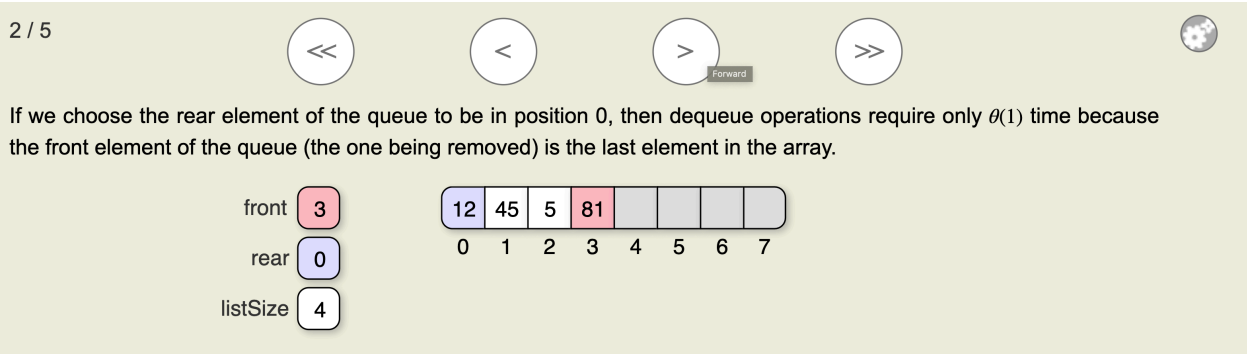

Figure 5.2: The next slide after the one shown in Figure 5.1.

5.2 JavaScript Algorithm Visualization Library: JSAV

As discussed earlier, JSAV is a development library that is used for creating Algorithm Visualizations using JavaScript. JSAV is part of the OpenDSA infrastructure, which is used to create interactive digital books. JSAV helps to show how an algorithm works by displaying the changes that happen to a certain input during the algorithm execution.

For example, Figure 5.1 shows the first step of creating an array-based implementation for the queue. We can see that navigation buttons are used to allow students to navigate through the algorithm steps. These navigation buttons are forward, backward, first slide, and last slide. For example, when a student clicks the forward button, the next step of the algorithm

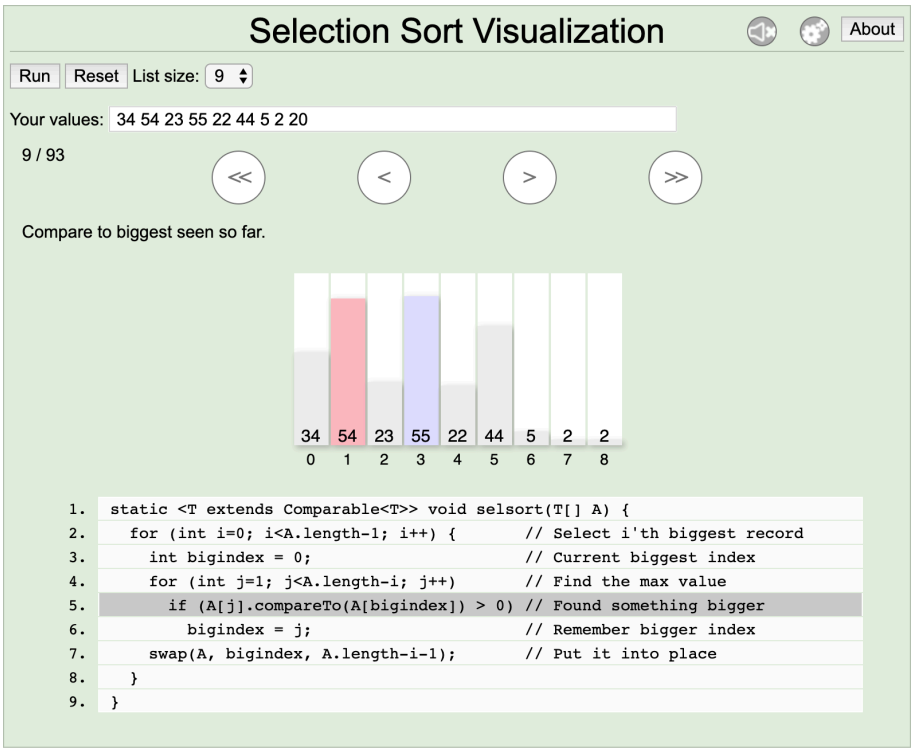

Figure 5.3: Another format for JSAV algorithms.

will be displayed as shown in Figure 5.2.

Another format for JSAV slides is shown in Figure 5.3. This format allows for more interactions as students can input their own data to the algorithm. The algorithm will, then, be applied on the input array and the visualization will show the changes to the array after each step of iteration of the algorithm.

We can notice here that, in these visualizations, a student needs to read the description first and then watch for the changes in the visualization. This means that these two things (textual explanation and their visual actions) are competing for student attention. Hence, students do not make full use of the interactive slide show. Now consider the effect if a student can listen to this description while watching the changes simultaneously. In this case, the information will be delivered to the student's cognitive systems using two cognitive

channels at the same time, which better matches the dual channel assumption of Mayer.

To this end, we have answered the first research question by identifying precisely what textual content will be converted into an audible format, and why it is not as likely beneficial for students to broaden the use of narration to the whole course.

Now that we have determined which parts of the eTextbook system are most likely to benefit from audio narration, we must find an implementation mechanism. One implement to adding audio narration is that it would take a long time for human to record the verbal track, and to integrate these into the system. It has been a goal of the OpenDSA project for several years to add audio material in this way, but so far the effort required has been too much for this to be implemented.

Fortunately, there have recently been great advances in automatic text-to-speech conversion. This approach to automatically create audio narration from our existing text makes the task far more practical.

5.3 Narration Support Implementation

We use the the Google text-to-Speech API [43] to convert the explanatory captions in JSAV slide shows into audible format. Google text-to-Speech API uses a process known as speech synthesis in which the written text is translated into an audio file and then spoken using a human voice. The API provides a set of voices to select from, and if no selection is made, a default voice for each browser is selected.

The next step after developing the narrations tool was to integrate it with JSAV. This step was challenging because of the way JSAV visualizations are generated. JSAV uses two different formats for its visualizations as was shown earlier. These slide shows are generated

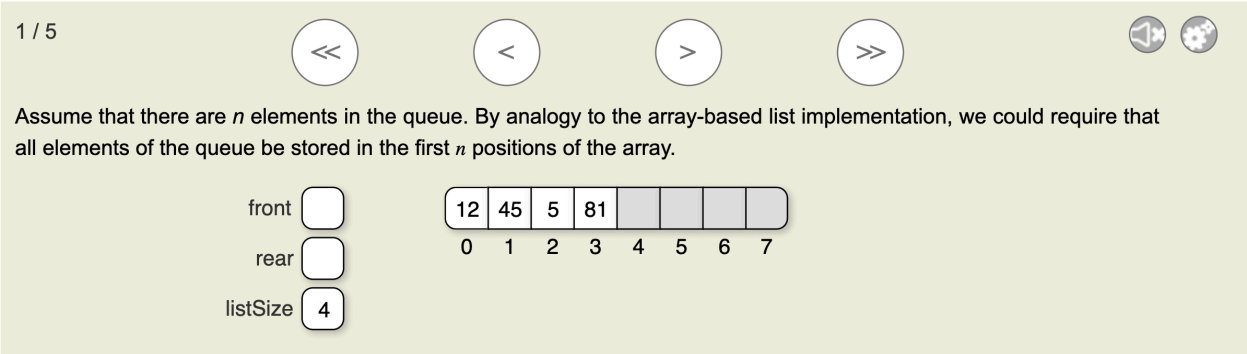

Figure 5.4: JSAV example after including narration support with the default selection to be OFF.

automatically and their content is loaded from the HTML page of each module. To create a stable implementation of the narration support tool, it had to be implemented in the JSAV library itself and to be executed when a JSAV slide show is loaded. However, this caused another issue of obtaining the text from the HTML page to be fed to the Google Text-to-Speech API, as JSAV just creates the slide show container and provides the necessary functionality.

To this end, we have implemented another module within JSAV to identify each JSAV container and retrieve the current text within this container to be used in the narration support tool. The final outcome of the tool can be seen in Figure 5.4.

The tool provides as a button in the JSAV slide show container, close to the settings button. This button has a default selection to be OFF, which means narration will not start automatically when the slide show is loaded. When a user clicks the sound button, it changes to be ON as shown in Figure 5.5.

Since JSAV provides navigation controls, we implemented our sound controls to be executed with the slide navigation. When a student selects to navigate to another slide, its contents will be narrated whenever the sound control is ON. If necessary the narration of the previous

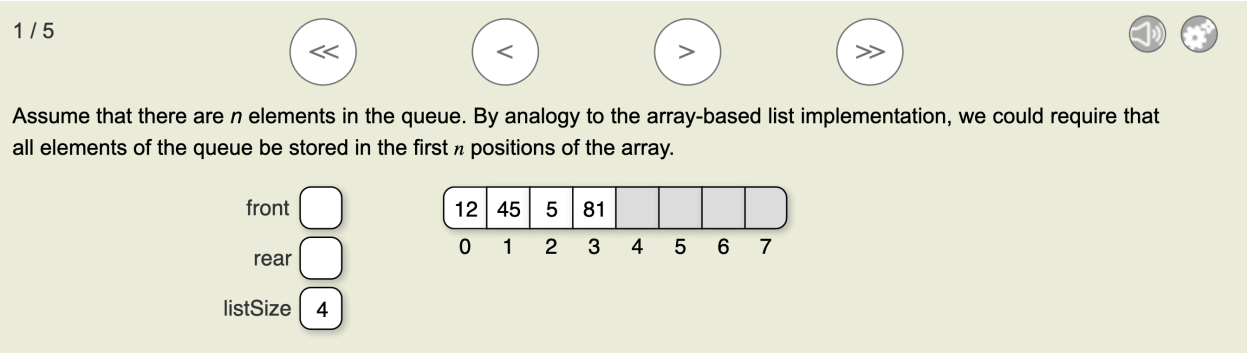

Figure 5.5: JSAV example after changing the narration support to be ON.

slide will stop when the user advances to a new slide, so that there will be no interference.

A similar option was implemented when there is more than one slide show in the same web page. If a student starts the narration of one slide show, and then clicks on a navigation button in any other slide show, the narration of the first slide show will stop immediately. In this case, the sound button of the original slide show till toggle to the OFF position.

To this end, we have answered the questions on how students can control the spoken text and the narration support tool. We also note that, since the narration tool is implemented in JSAV and uses our module that extracts text from the HTML, our narration support tool can automatically be used with any new JSAV slideshows created for OpenDSA. This requires almost no set-up from the developers as the framework automatically provides audio narration support.

5.4 Results

Now, we discuss the students' use of the narration feature through analyzing their usage data collected from OpenDSA. The data were collected similarly to the concept maps by logging the use events in the OpenDSA database. Since this is a new feature, the first semester that

we were able to collect the data is Fall 2018. Therefore, we will discuss the results in both the Fall 2018 and Spring 2019 semesters.

After implementing the narration feature and integrating it into JSAV, it was offered to students as part of their courses without explicitly notifying them about the new feature. The sound button was simply added to JSAV slideshows. To study the students' use of the new feature, we logged students' use of the narration tool in both CS2114 and CS3114. This was done for both Spring 2019 and Fall 2018 semesters.

Figure 5.6 shows the number of times each student, defined by user ID, has used the narration feature so far in CS2114 in Spring 2019. The numbers of use are sorted from the smallest to the largest. Student use varies, with the maximum number of narration use was 25 times and the second highest was 23. Similarly, Figure 5.7 displays the students use of the narration in CS2114 in Fall 2018. We can notice that the maximum number of narration uses was 53 and the second highest was 48.

Table 5.1 summarizes the use in both semesters. We notice that the percentage of students who have used the narration tool was small compared to the class sizes, about 5% (11 students) and 3.5% (19 students) in the Fall 2018 and Spring 2019 semesters, respectively. The rest of the students did not try the feature even once. Since those students did not try the feature, there is not enough information to decide if they were not interested in the feature itself, or they did not notice its presence.

A single use happened for two students in Fall 2018 and three students in Spring 2019. These numbers reflect students who tried the feature and did not use it again. Finally, the average of students' use (for those who used it at least once) was 7.8 times in Fall 2018 and 5.9 times in Spring 2019.

Similarly, Figure 5.8 shows the sorted number of times each student, defined by user id,

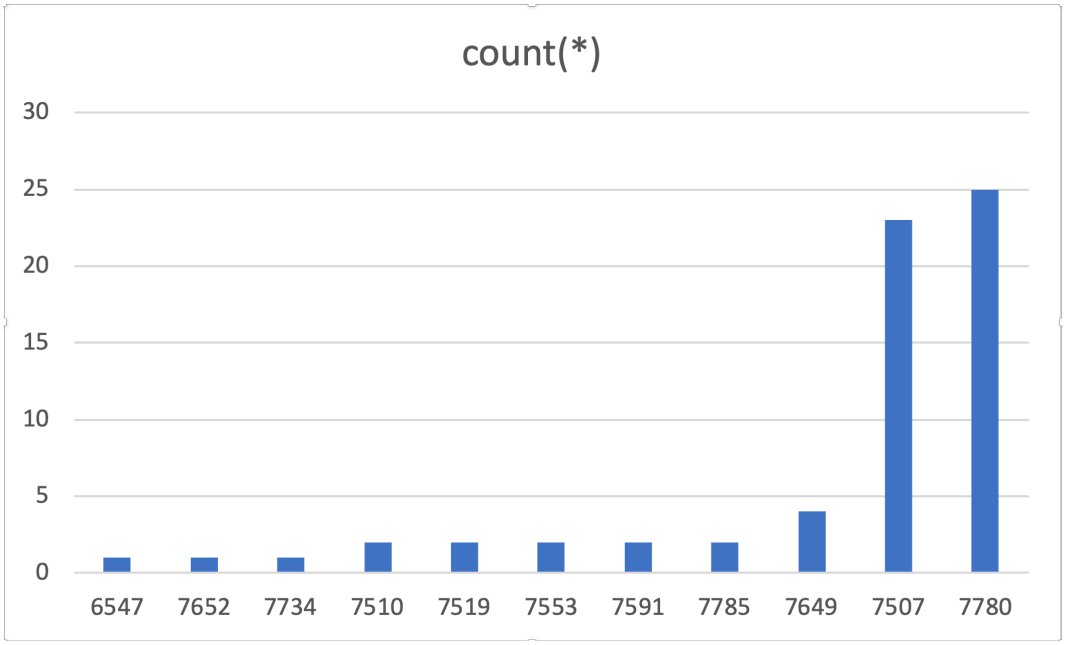

Figure 5.6: Student use of the narration feature in CS2114 in the Spring 2019 semester (11 students).

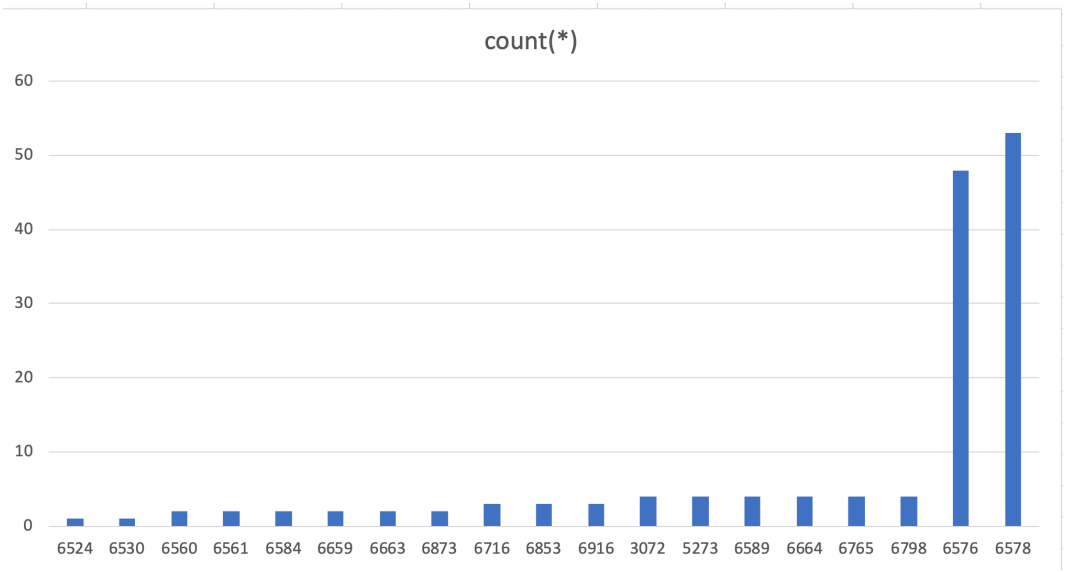

Figure 5.7: Student use of the narration feature in CS2114 in the Fall 2018 semester (19 students).

CS2114	Fall 2018	Spring 2019
Number of use	19	11
Total number of students	380	308
Percentage of use	5%	3.5%
Minimum number of use	1	1
Maximum number of use	53	25
Mean number of use	7.8	5.9

Table 5.1: Students' use of the narration tool in CS2114 for both Fall 2018 and Spring 2019.

CS3114	Fall 2018	Spring 2019
Number of use	16	17
Total number of students	387	268
Percentage of use	4%	6.3%
Minimum number of use	1	1
Maximum number of use	6	156
Mean number of use	2.75	12.35

Table 5.2: Students' use of the narration tool in CS3114 for both Fall 2018 and Spring 2019.

has used the narration feature so far in CS3114 in Spring 2019. The maximum number of narration use was 156 times which represents a strong interest in using the narration feature by that user. The second highest in the same semester was 21.

Figure 5.9 displays the students use of the narration in CS3114 in Fall 2018. We can notice that the maximum number of narration uses was 6, which occurred for two students. The second highest was 4.

Table 5.2 summarizes the students' use of the narration feature in CS3114 for both semesters. The percentage of students who have used the narration tool was also small. Three students of 16 and 17 used it only once in both Fall 2018 and Spring 2019, respectively. The average number of uses was 2.75 and 12.35, respectively.

In general, we think that the results presented in this section are affected by the lack of

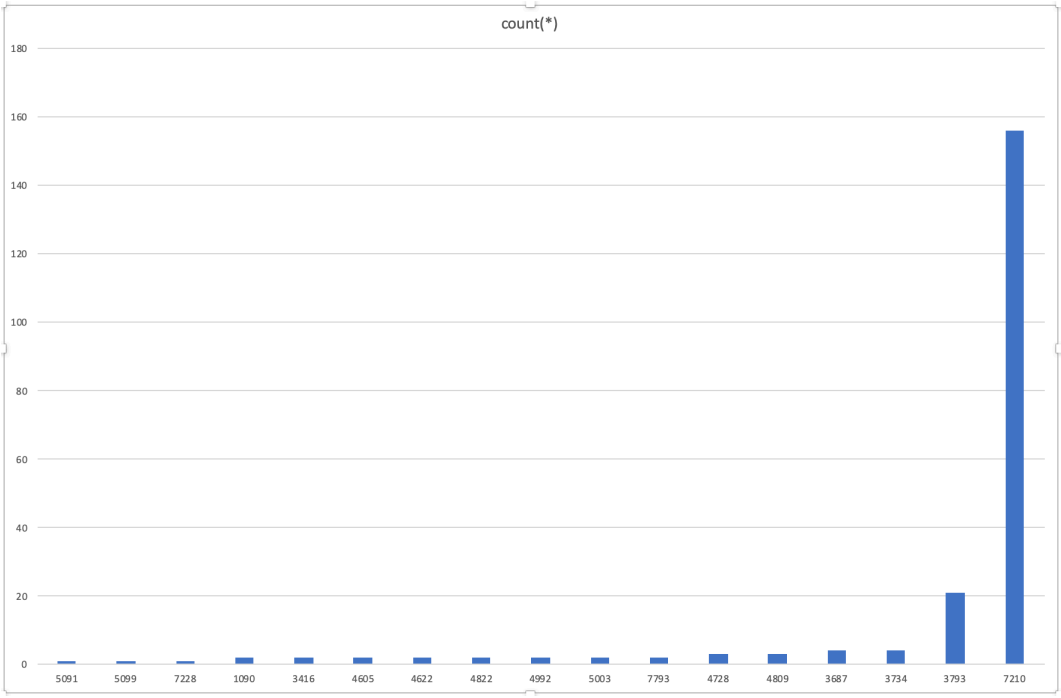

Figure 5.8: Student use of the narration feature in CS3114 in the Spring 2019 semester (17 students).

advertisement of the narration feature. Many students did not use the feature at least once to decide if it is useful or not. On the other hand, some students found the feature very helpful and used it for many slideshows.

5.5 Open Problems with Narration Support

The main technical challenge with narration support was the way the mathematical formulas are spoken. As we are dealing with algorithms, many of the slides have some math, such as recurrence relations, algorithm complexity in terms of asymptotic growth rate and even sometimes arithmetic operations and symbols. We have addressed some of these challenges in our implementation by replacing theses formulas with written English words before feeding

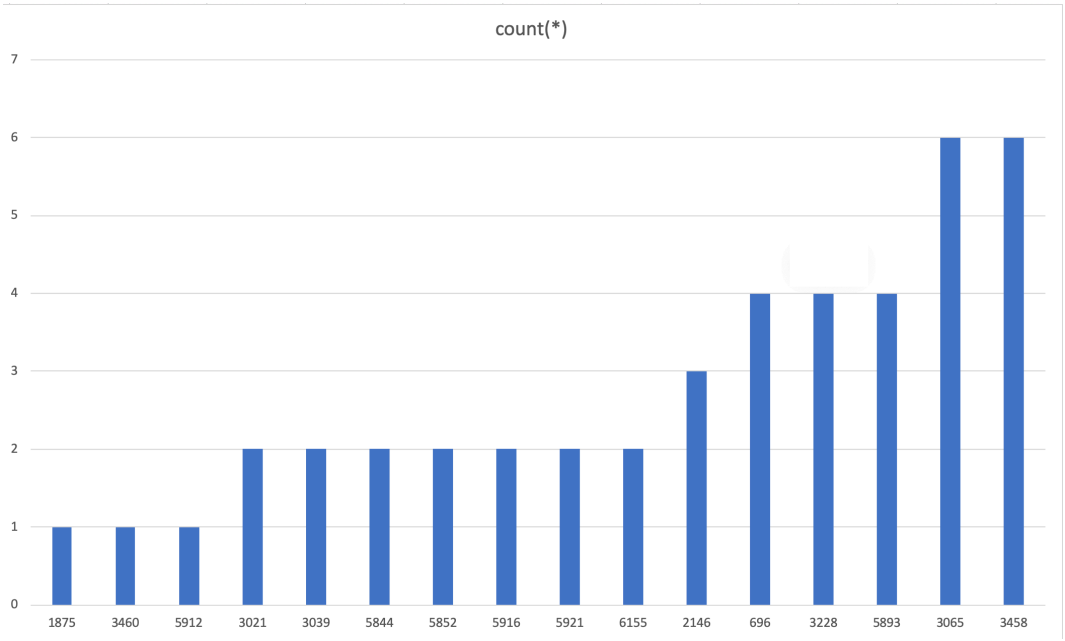

Figure 5.9: Student use of the narration feature in CS3114 in the Fall 2018 semester (16 students).

it the text-to-Speech API. For example, Google text-to-Speech API ignores the dashes in its spoken text. If the formula has a subtraction operator, written as a dash, it will be completely ignored by the API. Therefore, we generated some test cases to replace this dash, if it means subtraction, with the word "minus". Then in the spoken version of the text, the whole sentence makes sense. Given the variety of mathematical formulas that are used in algorithms, this remains an open problem that needs to be addressed carefully in order to develop effective solutions for the formulas to be spoken in a natural way.

Chapter 6

Conclusions and Future Work

In this work, we have addressed the problem of automatically generating multimedia content in digital courses. In particular, two types of automatically generated multimedia were considered. First, we have designed and implemented an interactive visualization tool for the glossary terms using concept maps. In this implementation, glossary terms were visualized as nodes in graphs, while their relationships were associated with the edges. Our concept maps tool was implemented within the OpenDSA e-textbook system. The concept map associated with a glossary term is automatically generated when that term is clicked in the text. We have then evaluated the effectiveness of our concept maps by performing statistical analysis to measure the effect of our concept maps on the students' use of the glossary. We used ANOVA tests followed by multi-comparison tests to compare the average students' use after and before implementing the concept maps. Results have shown that our concept maps design has significantly increased students' use of the glossaries when they are made aware of their existence.

A future direction that can be explored in the concept map implementation is to automatically adjust the number of the displayed nodes in the graph. This can be achieved by determining the optimal number of levels to be displayed for each node based on the whole number of related nodes retrieved after each level.

Our other work pertains to automatically generating sound files in digital courses. We have implemented a narration support tool in JSAV, the graphics library used by OpenDSA. This

allows students to listen to the algorithm description while watching the visualization. In our implementation, the narration tool is included automatically every time a JSAV slide show is generated. In this way, all slide shows generated with JSAV will have access to our narration support. Finally, we have discussed an open problem with the narration support in general and with the Google text-to-speech API, in particular. The problem concerns finding an effective solution to the way mathematical formulas are spoken, and trying to find a natural way to narrate these formulas. Another promising direction for the narration tool emerges from the fact that narration can be useful for visually impaired users. Hence, our narration tool can be modified and enhanced to suit such students and ease using the course materials for them.

Finally, for both tools, a survey needs to be designed and given to the students to obtain their feedback about the tools. In particular, students' feedback about the available features can help to determine which features need more development or which features are not used by students, and, hence, can be eliminated. It is also useful to ask students if they have used concept maps before this class. This will help to analyze the students' use of concept map glossaries in light of their previous experience with concept maps.

Bibliography

[1] R. E. Mayer, "Cognitive theory and the design of multimedia instruction: an example of the two-way street between cognition and instruction," *New Directions for Teaching and Learning*, vol. 2002, no. 89, pp. 55–71, 2002.

[2] ——, "Applying the science of learning: Evidence-based principles for the design of multimedia instruction." *American psychologist*, vol. 63, no. 8, p. 760, 2008.

[3] B. Hoffmann and D. Ritchie, "Using multimedia to overcome the problems with problem based learning," *Instructional Science*, vol. 25, no. 2, pp. 97–115, 1997.

[4] S. Mishra and R. C. Sharma, *Interactive multimedia in education and training*. IGI Global, 2004.

[5] U. V. Reddi, "Multimedia as an educational tool," *Educational multimedia: A handbook for teacher-developers*, pp. 3–7, 2003.

[6] A. L. Y. Low, K. L. T. Low, and V. C. Koo, "Multimedia learning systems: a future interactive educational tool," *The Internet and Higher Education*, vol. 6, no. 1, pp. 25–40, 2003.

[7] T. L. Naps, G. Rößling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger *et al.*, "Exploring the role of visualization and engagement in computer science education," in *ACM Sigcse Bulletin*, vol. 35, no. 2. ACM, 2002, pp. 131–152.

[8] C. D. Hundhausen, S. A. Douglas, and J. T. Stasko, "A meta-study of algorithm visu-

alization effectiveness," *Journal of Visual Languages & Computing*, vol. 13, no. 3, pp. 259–290, 2002.

[9] U. Reddi and S. Mishra, "Educational multimedia a handbook for teacher-developers commonwealth educational media centre for asia," *New Delhi*, 2003.

[10] OpenDSA Project. OpenDSA system documentation. [Online]. Available: https://opendsa.readthedocs.io/en/latest/index.html

[11] R. Smith, "The purpose, design, and evolution of online interactive textbooks: the digital learning interactive model," *History Computer Review*, vol. 16, no. 2, pp. 43–43, 2000.

[12] H. Beetham and R. Sharpe, *Rethinking pedagogy for a digital age: Designing for 21st century learning*. Routledge, 2013.

[13] Virginia Tech. CS2 software design & data structures. [Online]. Available: http://lti.cs.vt.edu/LTI_ruby/Books/CS2/html/Glossary

[14] C. A. Shaffer, M. L. Cooper, A. J. D. Alon, M. Akbar, M. Stewart, S. Ponce, and S. H. Edwards, "Algorithm visualization: The state of the field," *ACM Transactions on Computing Education (TOCE)*, vol. 10, no. 3, p. 9, 2010.

[15] E. Fouh, M. Akbar, and C. A. Shaffer, "The role of visualization in computer science education," *Computers in the Schools*, vol. 29, no. 1-2, pp. 95–117, 2012.

[16] V. Karavirta and C. A. Shaffer, "Creating engaging online learning material with the jsav javascript algorithm visualization library," *IEEE Transactions on Learning Technologies*, vol. 9, no. 2, pp. 171–183, 2016.

[17] J. D. Novak and A. J. Cañas, "The theory underlying concept maps and how to construct and use them," Institute for Human and Machine Cognition, Pensacola, Tech. Rep., 2008.

[18] J. D. Novak and D. Musonda, "A twelve-year longitudinal study of science concept learning," *American Educational Research Journal*, vol. 28, no. 1, pp. 117–153, 1991.

[19] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information." *Psychological Review*, vol. 63, no. 2, p. 81, 1956.

[20] M. J. Eppler, "A comparison between concept maps, mind maps, conceptual diagrams, and visual metaphors as complementary tools for knowledge construction and sharing," *Information Visualization*, vol. 5, no. 3, pp. 202–210, 2006.

[21] J. D. Novak, "Concept mapping: A useful tool for science education," *Journal of Research in Science Teaching*, vol. 27, no. 10, pp. 937–949, 1990.

[22] ——, *Learning, creating, and using knowledge: Concept maps as facilitative tools in schools and corporations*. Routledge, 2010.

[23] J. Stewart *et al.*, "Concept maps: A tool for use in biology teaching." *American Biology Teacher*, vol. 41, no. 3, pp. 171–75, 1979.

[24] G.-J. Hwang, F.-R. Kuo, N.-S. Chen, and H.-J. Ho, "Effects of an integrated concept mapping and web-based problem-solving approach on students' learning achievements, perceptions and cognitive loads," *Computers & Education*, vol. 71, pp. 77–86, 2014.

[25] R. Kavitha, A. Vijaya, and D. Saraswathi, "An augmented prerequisite concept relation map design to improve adaptivity in e-learning," in *International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME-2012)*. IEEE, 2012, pp. 8–13.

[26] S. Y. Kwon and L. Cifuentes, "Using computers to individually-generate vs. collaboratively-generate concept maps," *Journal of Educational Technology & Society*, vol. 10, no. 4, pp. 269–280, 2007.

[27] C. V. Lloyd, "The elaboration of concepts in three biology textbooks: Facilitating student learning," *Journal of Research in Science Teaching*, vol. 27, no. 10, pp. 1019–1032, 1990.

[28] M. van Bon-Martens, L. van de Goor, J. Holsappel, T. Kuunders, M. Jacobs-van der Bruggen, J. Te Brake, and J. van Oers, "Concept mapping as a promising method to bring practice into science," *Public Health*, vol. 128, no. 6, pp. 504–514, 2014.

[29] J. Turns, C. J. Atman, and R. Adams, "Concept maps for engineering education: A cognitively motivated tool supporting varied assessment functions," *IEEE Transactions on Education*, vol. 43, no. 2, pp. 164–173, 2000.

[30] M. A. Ruiz-Primo and R. J. Shavelson, "Problems and issues in the use of concept maps in science assessment," *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching*, vol. 33, no. 6, pp. 569–600, 1996.

[31] Mindmup. [Online]. Available: https://www.mindmup.com/

[32] ihmc. Cmap. [Online]. Available: https://cmap.ihmc.us/

[33] Lucidchart. [Online]. Available: https://www.lucidchart.com/pages/landing/concept_map_maker

[34] mindmeister. [Online]. Available: https://www.mindmeister.com/

[35] J. Villalon and R. A. Calvo, "Concept maps as cognitive visualizations of writing assignments," *Journal of Educational Technology & Society*, vol. 14, no. 3, pp. 16–27, 2011.

[36] N. Q. Zhu, *Data visualization with D3. js cookbook*. Packt Publishing Ltd, 2013.

[37] Json. [Online]. Available: https://www.json.org/

[38] Khan Academy. Khan academy exercises wiki. [Online]. Available: https://github.com/Khan/khan-exercises/wiki

[39] OpenDSA Project. OpenDSA-LTI implementation. [Online]. Available: https://opendsa.readthedocs.io/en/latest/OpenDSA-LTI-Implementation.html

[40] Statistics How To. ANOVA test: Definition, types, examples. [Online]. Available: https://www.statisticshowto.datasciencecentral.com/probability-and-statistics/hypothesis-testing/anova/

[41] Leard Statistics. One-way ANOVA. [Online]. Available: https://statistics.laerd.com/statistical-guides/one-way-anova-statistical-guide.php

[42] Amazon Inc. Audible Audiobooks. [Online]. Available: https://www.amazon.com/Audible-Audiobooks/

[43] Google.com. Cloud Text-to-Speech API basics. [Online]. Available: https://cloud.google.com/text-to-speech/docs/basics