Visualization, the graphical representation of data by computer, is believed to be an aid to cognitive processes. Algorithm animation, a type of visualization that displays a dynamic visual representation of the state and behavior of algorithms, is aimed to help students learn and understand complex algorithms better and more easily. Previous evaluations of algorithm animations have been performed. However, the results of these studies were mixed and often inconclusive. The visualization community has called for further studies to answer the question of whether algorithm animation is beneficial. This thesis presents such a study. Three-dimensional graphics were used to implement algorithm animations for distributed algorithms. Two studies were conducted to evaluate whether these visualizations helped students to learn about the distributed algorithms.

INDEX WORDS: Visualization, Algorithm visualization, Algorithm animation, Evaluation, Java 3D, Distributed algorithms, Distributed computations.
VISUALIZATION AS AN AID FOR UNDERSTANDING DISTRIBUTED
ALGORITHMS: AN EVALUATION

by

RONG WU

B.E., Zhengjiang University, P. R. China, 1997
M.S., The University of Georgia, 1999

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial
Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2001
VISUALIZATION AS AN AID FOR UNDERSTANDING DISTRIBUTED ALGORITHMS: AN EVALUATION

by

RONG WU

Approved:

Major Professor: Eileen T. Kraemer
Committee: Daniel M. Everett
Walter D. Potter

Electronic Version Approved:

Gordhan L. Patel
Dean of the Graduate School
The University of Georgia
December 2001
ACKNOWLEDGMENTS

I would like to thank my major professor, Dr. Eileen T. Kraemer, for her patience in guiding me, and encouraging me throughout this study. This project can not be done without her invaluable advice.

I also would like to express my appreciation to Dr. Daniel Everett and Dr. Don Potter for being on my advisory committee, and for opening up new fields of knowledge to me through their classes.

My special thanks go to Mihail E. Tudoreanu for developing the visualizations, for always giving invaluable suggestions, and for giving me generous help during the whole study.

I also would like to thank other team members, Ritu Dhawan and Ashley Taylor, for their collaborating work and invaluable suggestions.

I also wish to thank Jinhua Guo, Renyi Liu, David Miller, Jian Wang and all my friends in Athens for their kind help and encouragement.

My sincere thanks go to my husband, Xiaoyu Zhang, and my parents for their unconditional love and support.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2 LITERATURE REVIEW</td>
<td>3</td>
</tr>
<tr>
<td>Visualization as an Aid to Learning</td>
<td>3</td>
</tr>
<tr>
<td>Software Visualization Classifications</td>
<td>5</td>
</tr>
<tr>
<td>Algorithm Animation</td>
<td>7</td>
</tr>
<tr>
<td>Lack of Solid Proof of Visualization’s Efficacy</td>
<td>8</td>
</tr>
<tr>
<td>Lack of Solid Proof of Algorithm Animation’s Efficacy</td>
<td>9</td>
</tr>
<tr>
<td>About Our Study</td>
<td>11</td>
</tr>
<tr>
<td>3 FIRST STUDY</td>
<td>14</td>
</tr>
<tr>
<td>Materials and Methods</td>
<td>14</td>
</tr>
<tr>
<td>Data and Results</td>
<td>22</td>
</tr>
<tr>
<td>Experimental Observations and Discussions</td>
<td>28</td>
</tr>
<tr>
<td>4 SECOND STUDY</td>
<td>34</td>
</tr>
<tr>
<td>Materials and Methods</td>
<td>34</td>
</tr>
<tr>
<td>Data and Results</td>
<td>41</td>
</tr>
<tr>
<td>Experimental Observations and Discussions</td>
<td>44</td>
</tr>
<tr>
<td>5 CONCLUSIONS AND FUTURE WORK</td>
<td>47</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>49</td>
</tr>
<tr>
<td>APPENDICES</td>
<td>53</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

Visualization uses computers as a medium to render visual representations of data, provides real-time interactivity, and can automatically and quickly transform the underlying data into pictures and animations, and reveal hidden patterns [Card, et al. 1999]. These features have the potential to amplify cognition.

The strong belief that visual representations can facilitate the cognitive process has driven many researchers in computer science to develop tools utilizing visualization, hoping to help users better learn about, understand, debug, and develop programs. Such visualizations are known as software visualizations. The field of software visualization can be further categorized into sub-fields such as program visualization and algorithm visualization.

A subset of algorithm visualization, algorithm animation, or dynamic algorithm visualization, presents dynamic representations of the algorithm’s progress, as opposed to simple static displays. These visualizations aim to help students learn and understand complex algorithms better and more easily. Two aspects of algorithm animation are believed to be primarily responsible for its purported benefit to the cognitive process. First, it visualizes abstract concepts, which are usually difficult for students to learn. Second, algorithm animation displays the evolving process of the algorithm, which is the essence of computer algorithms [Byrne, et al. 1999].
Some formal studies have been conducted to evaluate algorithm animation as a teaching tool, but the objective measures were mixed and inconclusive, although students gave positive feedback on the usefulness of algorithm animation when subjective comments were collected [Stasko and Lawrence 1998; Byrne, et al. 1999].

The purpose of this study is to evaluate the efficacy of algorithm animation for helping students to learn about distributed algorithms. The graphical representation used in this study was three-dimensional animations implemented in Java 3D.

The first study was conducted in Spring, 2001. The quantitative results from the first study did not give any conclusive results. However, some problems with the materials and the practice of the study were identified. After improving the materials according to the findings from the first study, a second study was conducted in Fall, 2001. The quantitative results from the second study showed that the group that learned about the algorithm with the visualization outperformed the group that learned about the algorithm without the visualization. Subjective comments given by the participants also supported the use of the algorithm animation to facilitate the learning process.

Literature research and study motivations are described in Chapter 2. The details and results of the two evaluation studies are explained in Chapter 3 and 4, respectively. The findings from those two studies are also discussed in Chapter 3 and 4. Chapter 5 summarizes the findings in the studies and proposes possible future research.
Visualization as an aid to learning

Definition of Visualization

Graphical representations have long been an important tool for helping users learn about and understand complex problems or situations. It has been concluded that two ways in which graphical representations are helpful in the learning and cognitive processes: to communicate ideas and to create or discover the ideas [Card, et al. 1999]. Visualization, defined as “the use of computer-supported, interactive, visual representations of data to amplify cognition” [Card, et al. 1999], has a very good quality of rendering, and real-time interactivity. It can automatically and quickly transform the underlying data into pictures and animations, and reveal hidden patterns. These features have the potential to amplify cognition.

Visualization is not merely the pictorial representation of physical objects but more importantly, it serves to provide insight to help discover, make decisions and explain. In addition to visualizations that convey visible properties of objects that exist in the physical space, another type of visualization maps abstract data, nonphysical information, into effective visual form to aid cognition. This kind of visualization, defined as Information Visualization by Card, et al. [1999], takes advantages of both the proper visual representation of the problem and the interactive manipulation of those representations.
Proposed Ways to Amplify Cognition by Visualization

Six major ways in which visualization can amplify cognition have been stated by Card, et al. [1999].

• Visualization increases the memory and processing resources. It can expand processing capability either by using the resources of the visual system directly [Resnikoff 1987], or by offloading work from the cognitive to the perceptual system [Larkin and Simon 1987]. It also expands the working memory, and the storage of the information [Norman 1993].

• Visualization can reduce the effort involved in search for information by grouping or visually relating information. Compacting information into small space [Tufte 1983], hierarchical search, and spatially indexed addressing [Larkin and Simon 1987] are some of the functionalities that visualizations may possess and that can reduce the effort involved in search.

• Visualization can enhance the recognition of patterns in the data. By simplifying and organizing data into a more aggregated form, visualization can aid in a higher level of understanding [Card, et al. 1991; Resnikoff 1987].

• Visualization makes some inferences very easy for humans [Larkin and Simon 1987]. One demonstration of this is that all physics students are taught to start with a diagram of a problem [Card, et al. 1999]. Some visualizations even can enable complex specialized graphical computations [Hutchins 1995]. For example, the graphing calculators used in high school math classes use this kind of visualizations. [Card, et al. 1999]
• Visualization has the potential to monitor a large number of events by using perceptual attention mechanisms, if the display is designed such that the events to be monitored stand out by appearance or motion [Card, et al. 1999].
• Unlike static diagrams, visualization can encode information in an interactive medium, which allows exploration of a space of parameter values and can amplify user operations [Card, et al. 1999].

Possible Reasons That Explain Visualization’s Cognitive Benefits

Roman and Cox [1992] listed several factors that could contribute to the effectiveness of visualization in aiding cognitive processes. Such factors include the importance of visualization in human communication, the extraordinarily high bandwidth of light recognized by the human visual system, the high speed at which humans track and detect visual patterns, the potential of visualization to exploit multiple dimensions, and finally, the ability of visualization to abstract.

More and More Effort to Develop Visualization in Computer Science

With advances in graphical technology, exploring information by visualization is gaining more and more interest. Many computer scientists believe that visual representations can help users better and easily learn about, understand, debug, and develop programs. Therefore, many tools utilizing visualization have been developed or are being developed.

Software Visualization Classifications

Definition of Software Visualization

Though “visualization” includes the word “visual”, and is usually considered as restricted to visible images, Price, et al. [1993] cited from “The Oxford English
Dictionary” [Simpson and Weiner 1989] that the primary meaning of visualization is “the power or process of forming a mental picture or vision of something not actually present to the sight”. Therefore, visualization can result from sources other than visible objects. Price, et al. [1993] proposed “software visualization” as the term for the “use of the crafts of typography, graphic design, animation and cinematography with modern human-computer interaction technology to facilitate both the human understanding and effective use of computer software”, which includes all of the software design tasks from planning to implementation [Price, et al. 1993].

Classifications of Software Visualization

According to the taxonomy of software visualization by Price, et al. [1993], software visualization can be divided into two categories: program and algorithm visualization. Program visualization uses various techniques to enhance the understanding of computer programs. A large number of visualizations in this group visualize the code or data structures that are critical for understanding and debugging programs. On the other hand, algorithm visualization visualizes the high-level descriptions of the software, the more abstract information expressed by the program.

Some Published Software Visualization Applications

Many software visualization tools, covering a broad range of application areas, have been developed and studied. These systems include visualizations to format or pretty-print source code [Burd, et al. 1996; Baecker and Marcus 1986; Knuth 1984], to view statistics associated with code [Baker and Eick 1994], to automatically create graphical displays of program data structures [Mendelzon and Sametinger 1995; Myers 1983; Brayshaw and Eisenstadt 1991], to display software structural changes in large

**Algorithm Animation**

**Learning Benefits of Algorithm Animation**

As mentioned in the Introduction, a subset of algorithm visualization, algorithm animation, presents dynamic visual representations of algorithms’ progress. These visualizations are aimed to help students learn about and understand complex algorithms better and more easily. Two aspects of algorithm animation have been considered the main reasons that it can benefit the cognitive process. First, it visualizes abstract concepts such that they are easier for students to learn. The other aspect is that algorithm animation displays the essential evolving progress of the algorithm [Byrne, et al. 1999]. Other pedagogical benefits that Byrne, et al. claim may be gained from algorithm animations are dual representations (visual and textual) and the simulation of mental models. Some of the learning benefits of algorithm animation claimed by Gloor [1998] are: the enhanced learning motivation provided by more appealing presentations, extra practice opportunities other than traditional paper exercises and writing programs, assistance in the development of analytic skills, and the provision of additional context knowledge by the integrated hypermedia learning environment.

**Different Uses of Algorithm Animation**
Algorithm animation could be used in various learning environments: in the classroom as an aid to the lecture; after class as a study tool; or even as a tool for code reading and grading [Gurka and Citrin 1996].

**Some Published Algorithm Animation Systems**

Some algorithm animation systems, either animations for a specific algorithm or general algorithm animations, have been developed to help students learn about computer algorithms. Sorting Out Sorting, [Baecker 1981], the Brown University Algorithm Simulator and Animator (BALSA) [Brown and Sedgewick 1984], TANGO [Stasko 1990], Zeus [Brown 1992], Animated Algorithms [Gloor 1998], Pavane [Roman, *et al.* 1992], and POLKA [Stasko and Kraemer 1993] are some of the published algorithm animation systems.

**Lack of Solid Proof of Visualization’s Efficacy**

Despite the increasing interest and effort in software visualization, software visualization technology has not been widely used. Price, *et al.* [1993] stated that a possible reason is that the computer science community has not seen any solid study results that demonstrate the benefits of software visualization, though some empirical studies have been carried out. In other words, the usefulness of software visualization has not been conclusively demonstrated. Studies to evaluate the benefits of visualization are needed.

For some of the visualization tools, studies were conducted to measure the efficacy of these tools. Storey *et al.* [1997] conducted a study to compare three visualization systems for browsing program source code and exploring software structures. From detailed observations and interviews with the study participants, they
stated that in general, the visualization tools enhanced the users’ preferred comprehension strategies while solving the tasks. However, in some cases the tools hindered users’ progress in performing the tasks. An empirical investigation was carried out by Mulholland [1998] to study the effectiveness of four Prolog program visualization systems (tracers). Detailed observations of information access, use of strategies, and the misunderstandings of the traces were made. Although some of the features of those systems found to have a significant effect on the usability for Prolog beginners, a number of misunderstandings of the traces were identified to be caused by the tracers’ failure to convey sufficient information or relationship. Overall, the results were mixed and inconclusive.

**Lack of Solid Proof of Algorithm Animation’s Efficacy**

**Inconclusive Results from Previous Studies**

In the algorithm animation field, some algorithm animation tools were tested in small groups of volunteers or even used in real classes.

Animated Algorithms version 1.0 was tested in the lab where the animations were developed and had been used in the MIT “Introduction to Algorithms” course on a voluntary basis. The main benefit reported in voluntarily completed questionnaires was improved student’s motivation [Gloor 1998]. Bazik, et al. [1998] discussed the software visualizations used in teaching at Brown University and positive effects on students were found.

In addition, more formal studies have been conducted to evaluate algorithm animation as a teaching and learning tool. Two studies were discussed by Stasko and Lawrence [1998]. The first study examined the efficacy of algorithm animations created
by XTango as learning aids for computer science graduate students. Although the group provided with the animations performed slightly better than the text-only group, the results were not statistically significant. The second study also used an algorithm animation created by XTango to evaluate whether the animation helped undergraduate students learn about the algorithm in different learning environments. They did not obtain statistically significant results favoring the group provided with the animation although advantage was shown for the animation group and the students enjoyed having the animation as a learning tool. However, one finding confirmed was that the more relaxing learning environment, the interactive laboratory session, was beneficial in the learning process.

Two experiments were conducted by Byrne, et al. [1999] to evaluate the efficacy of algorithm animations created by POLKA [Stasko and Kraemer, 1993]. Some evidence of the benefits of animation in the learning process was provided. However, no statistically significant proof of animations’ usefulness was found.

**Difficulty to Define Efficacy of Algorithm Animation**

Stasko and Lawrence [1998] pointed out that one fact that may have contributed to those inclusive results is that it is very difficult to determine the effectiveness of an algorithm animation in an easily quantifiable way. They stated further that even to define what “learning an algorithm” means is not easy.

**Relaxed Learning Environment Found Helpful**

Motivated to see whether algorithm animation is a useful teaching tool in a more relaxing environment, Kehoe, et al. [2000] performed a study in an educational environment instead of the traditional exam-like settings, and detailed observations of
participants were made. This study led to the conclusion that algorithm animation is more beneficial in open, interactive learning situations and also enhances student motivation, which could consequently assist learning.

More Studies Needed in Evaluating Algorithm Animations’ Efficacy

Although algorithm animation has the potential of cognitive benefits, no solid proof has been found from previous studies to support this. The software visualization community has called for further studies in this area (Toronto workshop, May 2001).

About Our Study

Study Purpose

As described above, although algorithm animation appears very promising as a valuable teaching/learning tool, it has not gained any strong support from previous studies. The purpose of this study is to provide such an evaluation of the benefits of algorithm animation. In particular, we focus on an evaluation of three-dimensional animations of distributed algorithms.

The Applications

Understanding distributed computations is considered more challenging than understanding sequential computations because distributed programs are often large and complex and create a large amount of data [Kraemer 1998]. Textual representations of the data from distributed systems are by nature sequential and can be difficult for understanding the behavior of distributed system [Kraemer and Stasko 1993]. Visualization on the other hand can be a powerful tool for this task due to our highly developed imaging system, which helps us track multiple complex visual patterns and to
easily differentiate different visual patterns. Therefore, distributed computations were chosen as this study’s application area.

Two distributed algorithms were used in the first study: the “Dijkstra-Scholten Distributed Diffusing Termination Detection” and the “Distance Vector Routing”. The “Dijkstra-Scholten Distributed Diffusing Termination Detection” Algorithm applies to a distributed network environment in which nodes communicate with each other only by message passing. The algorithm allows a node to determine the termination of a task that has been computed by multiple nodes in the network. The “Distance Vector Routing” Algorithm also applies to a network environment in which nodes can exchange information only by message passing. The algorithm solves the problem of finding a minimal path to send packets among reachable nodes in the network. The details of these two algorithms are explained in Appendices A and B.

These algorithms were chosen because they are classical and interesting distributed algorithms and have the appropriate complexity for the participating students, who were senior and graduate Computer Science students. In the second study, only the “Dijkstra-Scholten Distributed Diffusing Termination Detection” algorithm was used.

Techniques of Visualization in this Study

The graphical representation used in this study was three-dimensional graphics. In addition to its advantage in the representation of naturally three-dimensional objects, it has another important benefit: the increased quality and quantity of information it expresses [Brown and Najork 1998]. Some algorithm animation systems utilizing interactive three-dimensional graphics were described and three distinct uses of three-dimensional graphics were explained: “expressing fundamental information about
structures that are inherently two-dimensional, uniting multiple views of an object, and capturing a history of a two-dimensional view” [Brown and Najork, 1998]. Brown and Najork [1998] also indicate that the use of three-dimensional graphics in visualization has tremendous potential, but most of it is yet to be discovered. The authors also state that more experimental studies are necessary in order to explore this field.
CHAPTER 3
FIRST STUDY

The first part of the evaluation was a study to evaluate the efficacy of algorithm animation in helping students learn about two distributed algorithms. The experiment was designed in such a way that each participant was asked to learn both of the algorithms, with visualization provided for one algorithm, and for each algorithm, about half of the participants learned about it with visualization. The details of the experiment are explained in the following sections.

Materials and Methods

Learning Materials

The materials provided to help the participants understand these algorithms included a hard copy of the algorithm description (Appendices A and B), a Java-style pseudocode display (Appendices C, D and E) and a textual output display (Appendix E) corresponding to the pseudocode. One group, consisting of half the participants, was also provided a visualization of the algorithm, seen in Figure 1 and Figure 2.

Implementations

Both algorithms were implemented in Java. Threads were used to simulate the distributed systems.

Visualization

A visualization implemented in Java 3D was designed for each of the algorithms by Mihail E. Tudoreanu (Ph.D. student in Computer Science Department of Washington
Univerisity). Java 3D API provides a set of interfaces that support a simple, high-level programming model, which allows program developers to build, render, and control the behavior of three-dimensional objects and visual environments. It is a standard extension to the Java 2 SDK [Java 3D API homepage]. Screenshots from these visualizations are seen in Figure 1 and Figure 2.

Figure 1 Screenshot of the visualization of the Termination Detection Algorithm The upper layout indicates the virtual tree view (parent-child relationship) in the network, and the bottom layout displays the network’s basic layout and each node’s status and attributes. The messages among nodes are represented by the small spheres on the connections between nodes, and different colors of the spheres indicate different types of messages. The numbers indicate the individual nodes represented by cylinders in different gray shade.
Figure 2 Screenshot of the visualization of Distance Vector Routing Algorithm The network layout is displayed by the disks and the connections in two dimensions. The pink spheres represent the messages passed among nodes. The bars in the third dimension indicate paths to other reachable nodes. The slope of the bar shows the distance to other reachable nodes. The steeper the slope, the longer the distance between the two nodes.

Testing Environment

In order to automate the data collection procedure and to avoid distracting or making the study participants uncomfortable, a testing environment was developed in Java to collect data and integrate all learning materials, except for the hard copy of the algorithm description, into one environment. The testing environment consisted of several screens. Screenshots of all screens of the testing environment are listed in Appendix E. A screenshot taken from the page that links to all the supporting learning materials is displayed in Figure 3.
The first two buttons link to the algorithm pseudocode and the visualization. The “Run Randomly” button allows the user to run the program with randomly chosen network configurations.

The algorithm visualization was based on the output of the program, which was executed in advance by the researchers. The participants were also given the opportunity to run the program with randomly chosen network configurations during the evaluation session, with the output and the visualization changing accordingly. This feature was added with the idea that different outputs and visualizations emphasize specific features of the algorithm, which might help subjects better understand the algorithms.

Subsequently, the questions were displayed page by page in the same layout as Figure 4. A complete list of questions for both algorithms can be found in Appendices F
and G. The page for question one is displayed in Figure 4. The participants were allowed to go back and forth between the question pages, and to answer the questions in any order.

![Screenshot of a question page from the testing environment](image)

Figure 4 **Screenshot of a question page from the testing environment** The list on the left is an index to the question pages. The question content and the multiple-choice answers are displayed at right. Each page displays one question. Clicking on the index item or the buttons at the bottom will change the question page accordingly.

**Evaluation Method and Data Collected**

The “Use Data Collection” method defined by Karat [1997] was employed in this study. This method collects some objective measures during the course of the system use, such as errors, task completion time, general logging data, etc. This method is more objective, more easily quantified than verbal reports and questionnaires, and is often used in more formal quantitative analysis.
Each user’s understanding of the algorithms was evaluated based on the number of questions they answered correctly. Time spent in the testing environment was also recorded.

All the questions were multiple-choice questions and designed to evaluate participants’ procedural and conceptual knowledge of the algorithms. The questions covered basic concepts, operations, limitations and simple improvements to the algorithms. All the answers could be found from the algorithm description, code, and output. The nine questions for each algorithm of the first study are listed in Appendices F and G.

The time-recording schemes were embedded in the program of the testing environment to collect the time usage automatically during the study. The time recorded included the time the participant spent on viewing the code, output, individual question pages, and the total time spent on answering all the questions. However, the time recorded was based on how long those windows were open. In addition, the participants were allowed to keep the code, output and visualization window open while answering the questions or to reopen them at any time. Thus, the time usage recorded was only approximate, but still showed some interesting trends.

**Study Subjects**

Thirty-nine students enrolled in the “Human Computer Interaction” class (a senior/graduate level course of Computer Science Department at UGA) participated in the study to earn class credit and compete for cash prizes. The top three performers (tied performers considered as in the same place) earned prizes of $20, $10, and $10 respectively according to the correctness of their answers to the questions.
Procedure

Each participant was required to learn the two algorithms during the evaluation process. The algorithm description, the Java-style pseudocode and the corresponding output displayed in the testing environment were provided. The study was designed such that for each algorithm, about half of the participants learned about it with visualization, and the other half learned about it without visualization (called Viz and Non-Viz groups, respectively). Also, for each participant only one algorithm is coupled with the visualization.

In addition, the participants who were in the same combination group of the algorithm and visualization were further divided into two sub-groups: one sub-group learned one algorithm with visualization first, and the other sub-group learned the other algorithm without visualization first. How the participants were grouped and the number of participants in each group is demonstrated in Table 1.

Table 1 Demonstration of different groups of participants  First indicates that the algorithms were learned in the first sub-session. TD represents the Termination Detection Algorithm; VR represents the Distance Vector Routing Algorithm.

<table>
<thead>
<tr>
<th></th>
<th>Viz</th>
<th>Non-Viz</th>
</tr>
</thead>
<tbody>
<tr>
<td>TD - First</td>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>VR - First</td>
<td>10</td>
<td>8</td>
</tr>
</tbody>
</table>

The participants, regardless of gender, age, or computer experience, were randomly assigned to one of the four groups. This was designed to eliminate the factor that the order of the encountering of the algorithms and the visualizations might have in the participants’ ability to understand the algorithms.

The participants were allowed to use all supporting materials when answering the questions. In addition, they were allowed to answer the questions in any order. Therefore,
they could learn the algorithm and answer the questions in any way and in any order they chose, thus providing a more relaxing learning environment as Kehoe, et al. [2000] suggested.

Only after answering all the questions were the participants allowed to exit the evaluation system. Immediately before exiting, they were asked whether they had enough time doing the evaluation and whether they knew the algorithm prior to the study. They were also asked to give any comments and suggestions they had.

Each participant attended a session ranging from fifty to seventy-five minutes in duration. Fifty minutes was the required minimum participation time to ensure that the participants took enough time to utilize all the supporting materials. Seventy-five minutes was the maximum participation time to avoid participant fatigue and to comply with the class schedule.

One week prior to the study, a pilot study was conducted so that researchers could experience the study procedure and to provide the opportunity to find any problems or overlooked details. The participants of the pilot study were two students from the same “Human Computer Interaction” class, excluded from the thirty-nine students attending the formal study. The time of the pilot study was chosen to ensure enough time for modification if problems emerged.

Analytical Method

After the study, the quantitative data, the correct responses of the answers and the time spent on answering questions were analyzed using a t-test [Bartz 1999] to compare the means of the data from the two groups (Viz and Non-Viz groups). The t-test uses the Student’s t distribution to analyze the difference between sample means when the
standard deviation of the population is not known [Bartz 1999]. The t value is calculated as Formula 1.

\[
t = \frac{(\bar{X}_1 - \bar{X}_2) - 0}{S_{\text{diff}}}
\]

(1)

where

\[
S_{\text{diff}} = \sqrt{\frac{(N_1 - 1)s_1^2 + (N_2 - 1)s_2^2}{N_1 + N_2 - 2} \left(\frac{1}{N_1} + \frac{1}{N_2}\right)}
\]

- \(s_1\) = the standard deviation of the first sample
- \(s_2\) = the standard deviation of the second sample
- \(N_1\) and \(N_2\) = the sizes of the respective samples

The hypothesis used in the study was the null hypothesis: the means of the two populations from which the sample are drawn are equal (\(H_0: \mu_1 - \mu_2 = 0\)). According to convention, the null hypothesis was rejected at the significance level of 0.05. This means that the difference observed in the samples would happen 5% of the time or less by sampling error [Bartz 1999].

**Data and Results**

**Correct Responses of the Questions**

The most important parameter in these two studies was the number of correct responses. Table 2 and Table 3 show the numbers of participants in different groups, and the means and the standard deviations of the number of questions answered correctly (Each participant answered nine questions for each algorithm). The complete data are listed in Appendices H and I.

**Table 2: Number of Correctly Answered Questions: Termination Detection Algorithm**

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>19</td>
<td>3.26</td>
<td>1.95</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>19</td>
<td>3.92</td>
<td>1.75</td>
</tr>
</tbody>
</table>
T-tests were carried out to determine whether there was a statistically significant difference between the means of the Viz and Non-Viz groups from the algorithms. The result showed that there was no statistically significant difference between the means of the two groups for either of the algorithms ($t = 1.111, \text{df} = 36, p>0.05; t = 0.358, \text{df} = 37, p>0.05$).

Each participant learned about both algorithms, one with, and one without the visualization. About half of the group learned about the one with the visualization first, and the other half learned about the one without the visualization first. The data from those two subgroups were also analyzed to see if there was an effect of the order by which the algorithms were learned. The results are listed in Table 4, Table 5, Table 6, and Table 7.

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>20</td>
<td>3.95</td>
<td>1.76</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>19</td>
<td>3.77</td>
<td>1.28</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>11</td>
<td>3.64</td>
<td>2.16</td>
</tr>
<tr>
<td>Second</td>
<td>8</td>
<td>2.88</td>
<td>1.64</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>9</td>
<td>3.44</td>
<td>1.81</td>
</tr>
<tr>
<td>Second</td>
<td>10</td>
<td>4.4</td>
<td>1.65</td>
</tr>
</tbody>
</table>
Table 6: Number of Correctly Answered Questions: Distance Vector Routing Algorithm - Viz Group

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>10</td>
<td>4.50</td>
<td>1.90</td>
</tr>
<tr>
<td>Second</td>
<td>10</td>
<td>3.4</td>
<td>1.51</td>
</tr>
</tbody>
</table>

Table 7: Number of Correctly Answered Questions: Distance Vector Routing Algorithm - Non-Viz Group

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>8</td>
<td>4</td>
<td>1.41</td>
</tr>
<tr>
<td>Second</td>
<td>11</td>
<td>3.55</td>
<td>1.21</td>
</tr>
</tbody>
</table>

Interestingly, in three of those four tables, the sub-groups that learned about the algorithm in the first half of the session had a higher average correct response than the sub-group that learned about the algorithm in the second half session. However, the t-test of those groups did not show any statistically significant difference ($t = 0.835, df = 17, p>0.05$; $t = 1.205, df = 17, p>0.05$; $t = 1.435, df = 18, p>0.05$; $t = 0.753, df = 17, p>0.05$).

Although the statistical test did not show any significant difference between the average correct responses of the sub-group that learned about the algorithm in the first half session and the sub-group that learned about it in the second half session, the means in those different groups still showed differences in some degree. Therefore, to avoid the impact of the order in which the two algorithms were learned, the number of correct responses of the Viz and Non-Viz groups that learned about the algorithm in the first half session was compared by t-tests with that of those who learned about the algorithm in the second half session. The data are listed in the Table 8, Table 9, Table 10, and Table 11.

Table 8: Number of Correctly Answered Questions: Termination Detection Algorithm - Group That Learned the Algorithm in the First Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>11</td>
<td>3.64</td>
<td>2.16</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>9</td>
<td>3.44</td>
<td>1.81</td>
</tr>
</tbody>
</table>
Table 9: Number of Correctly Answered Questions: Termination Detection Algorithm - Group That Learned the Algorithm in the Second Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>8</td>
<td>2.88</td>
<td>1.64</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>10</td>
<td>4.4</td>
<td>1.65</td>
</tr>
</tbody>
</table>

Table 10: Number of Correctly Answered Questions: Distance Vector Routing Algorithm - Group That Learned the Algorithm in the First Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>10</td>
<td>4.5</td>
<td>1.90</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>8</td>
<td>4</td>
<td>1.41</td>
</tr>
</tbody>
</table>

Table 11: Number of Correctly Answered Questions: Distance Vector Routing Algorithm - Group That Learned the Algorithm in the Second Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>11</td>
<td>3.4</td>
<td>1.51</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>10</td>
<td>3.55</td>
<td>1.21</td>
</tr>
</tbody>
</table>

The Viz groups did not show any significantly higher correct responses than the Non-Viz groups. The t-tests did not show any statistically significant difference between the means of respective Viz and Non-Viz groups either (t = 0.212, df = 18, p>0.05; t = 1.955, df = 16, p>0.05; t = 0.618, df = 16, p>0.05; t = 0.245, df = 19, p>0.05).

Participation Time

As mentioned in the previous chapter, the time recorded in the study was approximate. Nevertheless, statistical analysis was performed on the time that the participants spent answering the questions regarding each algorithm. Table 12 and Table 13 show the time spent by different groups.

Table 12. Time Spent on Answering Questions: Termination Detection Algorithm

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>19</td>
<td>13.06</td>
<td>8.75</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>19</td>
<td>17.48</td>
<td>8.98</td>
</tr>
</tbody>
</table>
Table 13. Time Spent on Answering Questions: Distance Vector Routing Algorithm

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>20</td>
<td>18.67</td>
<td>8.71</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>19</td>
<td>16.13</td>
<td>7.67</td>
</tr>
</tbody>
</table>

In answering the questions regarding the “Termination Detection” Algorithm, the participants in the Non-Viz group spent more time on average. The opposite was observed for the “Distance Vector Routing” Algorithm: the Viz group spent more time. However, the t-test did not show a statistically significant difference between the means of the Viz and Non-Viz groups for either of the algorithms (t = 1.536, df = 36, p>0.05; t = 0.965, df = 37, p>0.05).

As we will discuss in the next section, the time spent on the two sub-sessions by the same participant was affected by the restricted participation time and the limited lab facilities. In order to determine the time-using trend in the two sub-sessions, Table 14, Table 15, Table 16, and Table 17 show a comparison of the time spent on answering questions on the same algorithm, and from the same group (Viz/Non-Viz), but carried out in different sub-sessions.

Table 14. Time Spent on Answering Questions: Termination Detection Algorithm - Viz Group

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>11</td>
<td>15.99</td>
<td>10.54</td>
</tr>
<tr>
<td>Second</td>
<td>8</td>
<td>10.14</td>
<td>3.93</td>
</tr>
</tbody>
</table>

Table 15. Time Spent on Answering Questions: Termination Detection Algorithm - Non-Viz Group

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>9</td>
<td>22.19</td>
<td>10.10</td>
</tr>
<tr>
<td>Second</td>
<td>10</td>
<td>12.78</td>
<td>4.89</td>
</tr>
</tbody>
</table>
Table 16. Time Spent on Answering Questions: Distance Vector Routing Algorithm – Viz Group

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>10</td>
<td>23.66</td>
<td>8.78</td>
</tr>
<tr>
<td>Second</td>
<td>10</td>
<td>13.68</td>
<td>5.24</td>
</tr>
</tbody>
</table>

Table 17. Time Spent on Answering Questions: Distance Vector Routing Algorithm – Non-Viz Group

<table>
<thead>
<tr>
<th>Sub-group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>8</td>
<td>20.1</td>
<td>9.19</td>
</tr>
<tr>
<td>Second</td>
<td>11</td>
<td>12.16</td>
<td>4.18</td>
</tr>
</tbody>
</table>

It is obvious that the participants always took more time answering questions in the first sub-session than in the second sub-session. The t-test was also carried out to determine the significance between those means (t = 1.487, df = 17, p>0.05; t = 1.456, df = 17, p>0.05; t = 3.089, df = 18, p<0.01; t = 2.546, df = 17, p<0.05). For the “Termination Detection” algorithm, there is no statistically significant difference between those means, while for the “Distance Vector Routing” algorithm, there is a statistically significant difference between the means in sub-groups that learned the algorithm in the first and second sub-sessions. The significance level is 0.01 and 0.05 for the Viz and Non-Viz groups, respectively. Those significance levels mean that the possibility that the obtained difference would happen by sampling error is equal or smaller than 0.01 and 0.05 [Bartz 1999]. Therefore, the null hypothesis can be rejected confidently.

In order to eliminate the effect of the order in which the two algorithms were learned, the average time spent by the participants of the Viz and Non-Viz groups, who made the same algorithm evaluation in the same sub-session, was compared and are shown in Table 18, Table 19, Table 20, and Table 21.
Table 18. Time Spent on Answering Questions: Termination Detection Algorithm - Group That Learned the Algorithm in the First Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>11</td>
<td>15.99</td>
<td>10.54</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>9</td>
<td>22.19</td>
<td>10.10</td>
</tr>
</tbody>
</table>

Table 19. Time Spent on Answering Questions: Termination Detection Algorithm - Group That Learned the Algorithm in the Second Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>8</td>
<td>10.14</td>
<td>3.93</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>10</td>
<td>12.78</td>
<td>4.89</td>
</tr>
</tbody>
</table>

Table 20. Time Spent on Answering Questions: Distance Vector Routing Algorithm - Group That Learned the Algorithm in the First Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>10</td>
<td>23.66</td>
<td>8.78</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>8</td>
<td>20.1</td>
<td>9.19</td>
</tr>
</tbody>
</table>

Table 21. Time Spent on Answering Questions: Distance Vector Routing Algorithm - Group That Learned the Algorithm in the Second Half Session

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean (in minutes)</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>10</td>
<td>13.68</td>
<td>5.24</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>11</td>
<td>12.16</td>
<td>4.18</td>
</tr>
</tbody>
</table>

The trends were mixed and there is no statistically significant difference between the means \( t = 1.263, df = 18, p>0.05; t = 1.239, df = 16, p>0.05; t = 0.838, df = 16, p>0.05; t = 0.737, df = 19, p>0.05 \); therefore, no conclusion could be made here.

Per Question Correct Responses

The number and the rate of people who answered each question correctly are shown in Appendix J.

Experimental Observations and Discussions

In this study, the quantitative data did not show any conclusive evidence that visualization can aid learning of distributed algorithms, though a few participants gave
positive comments on the effectiveness of the visualization. After a thorough overview of the whole procedure after the study, several problems were found that might have affected the results of the study.

Lack of Incentives

The study was conducted in three days in the last week of the Spring Semester, 2001. The students earned class credit by participating in the study, and all the students did participate. However, their performances did not affect the credits that they earned, which likely led to the biggest problem encountered in this study: many students did not have enough incentive to explore all the materials provided and merely tried to get it done, although a prize was offered to the top three participants. A minimum of 50 minutes participation time was required to encourage and force the students to spend enough time on the evaluation. Still, some students may not have treated the study seriously and only tended to get the class credit by participating.

Some students did try their best to understand the algorithm during the test and provided valuable comments afterwards. The highest number of correct response was 13 out of the total 18 questions. Some students even felt that the 75-minute maximum participation time was not enough for them to explore all the materials. For those students, the shortage of time might have been a factor affecting their evaluation results, which is another potential problem. We note, however, that this problem applied equally to both the Viz and Non-Viz groups.

Insufficient Participation Time

A common pattern during the evaluation is that almost all of the participants spent more time in the first sub-session than they did in the second sub-session. For the
“Distance Vector Routing” Algorithm, the means of time spent on answering questions in
the first and second sub-sessions differed at the significance level of 0.01 for the Viz
group, and differed at the significance level of 0.05 for the Non-Viz groups. It is obvious
that most of them were rushed in the second sub-session no matter which group (Viz or
Non-Viz) they were in.

Several factors could have contributed to this pattern. For example, the
visualization was implemented in Java 3D. Due to the limited facility of the lab, only two
PCs can run the Java 3D programs. Therefore, when there were more than two
participants in the same session, they had to switch to different computers between the
two sub-sessions. As a result, it happened frequently that one participant was done with
the first algorithm and ready to begin the second one while the computer needed was still
being used by other participants. A typical session had more than two participants. This
probably affected some of the evaluation results because they had a maximum of 75
minutes to learn both algorithms. Some of them might have wasted some time waiting for
the machine and did not have enough time to learn the second algorithm. This effect on
the second sub-session was the same regardless of the Viz or Non-Viz groups.

In addition, the switching of machines between the two sub-sessions or only
proceeding to the second algorithm on the same machine might have biased evaluation
results in some degree since some participants might have realized that other participants
were proceeding to the second algorithm or someone was waiting for the computer they
were using, and thus finished evaluation earlier than they would have otherwise. The Viz
and Non-Viz groups were equally affected by this factor.
Most of the participants did not use all the 75 minutes allowed. However, it is likely that some of them have finished the evaluation in a rush and earlier than they would only because other participants in the same session had left, therefore affecting their performances, especially the performances of the second sub-session. This factor was also likely the cause of the differences found between the first and second sub-sessions.

**Distraction during Study**

Because of the large number of the participants and the limited lab facilities, the participants had to be grouped to attend the study. Although a maximum of only four students were allowed to participate in each session, some students still commented that having other people participating in the same session was distracting. In addition, the lab where the study was conducted is a research lab shared by three professors and there were people working in the lab all the time, which was also a distraction to the participants. However, these distractions affected both Viz and Non-Viz groups equally.

**Suspected Effect on Results**

*Inconsistent Explanation of the System*

At the beginning of each session, one experimenter briefly explained the consent form and the entire procedure to the participants. Although in most of the sessions the explanation was conducted by the same experimenter, there still was some inconsistency among different sessions, such as the details of the evaluation procedure being explained. Even though the experimenters did try to make the explanation consistent among different sessions, the participants’ attention was another affecting factor. In one case,
one participant even did not notice there was a hard copy of the algorithm description provided to him.

Problems of the Evaluation Method

Nine questions regarding each algorithm were given to the participants to test how well they understand the algorithms with the materials provided. These questions were brainstormed by the researchers and the number of questions was limited due to the time limit of the evaluation. Although the questions were intended to test how well the participants understood the algorithms, these questions may not be ideal for the evaluation, just as Stasko and Lawrence [1998] pointed out that it is very difficult to determine the effectiveness of an algorithm animation in an easily quantifiable way. Finding a suitable method to evaluate algorithm animation’s efficacy is still a challenging task and needs more study.

The participant could only exit the evaluation system after all the questions were answered. This design is to avoid accidental exit and to force the participants to put more effort into the study. However, this design has an inevitable flaw: some of the participants did not know the answers to some questions, and they had to guess the answers just to finish the evaluation. Thus, responses to some questions may represent guesses rather than well-considered responses.

Problems with the Learning Materials

The visualizations had not been reviewed by the whole research group prior to deployment of the study. Some participants commented that they expected the visualization would help them understand the algorithms. However, the visualization given did not achieve what they expected. Later review of the visualizations identified
several problems. The major problem was that the graphics lacked proper notations. Although a hard copy of the explanation of the visualization and how to control it were given to the subjects, the lengthy text and its separation from the visualization system hindered subjects’ understanding of the visualization. In addition, the ability to navigate the 3D visualization was provided by Java 3D and was intended to give the subjects the flexibility of choosing the views they preferred. However, it was found sometimes that the subjects could not relocate the graphics back to a good view after moving it because they were not familiar with manipulations of the objects in the visualization system; hence, the animation displayed in a less ideal view might not have provided the subjects as much information as it was supposed to.

Similarly, later review of the explanation materials uncovered some problems. Specifically, all the explanation materials of the evaluation system and the visualization were textual hard copies, and had not been reviewed by the whole research group. It turned out that some explanations were either too brief or too long, and without a direct relationship (graphical explanation, e.g., screen shots) to the system. The participants rarely referred to those materials and instead explored the system by themselves.

The problems identified in the study and the inconclusive nature of the results led the researchers to consider this as a pilot study, to correct the problems with the methodology, and to perform a second study. Chapter 4 described the revisions that were made and the results obtained from this second study.
CHAPTER 4
SECOND STUDY

Although the results from the first study did not give conclusive evidence, some problems that may have affected the results were discovered. In order to eliminate those confounding factors and to obtain more accurate results in the evaluation, the second study, with several modifications made according to the findings in the first study, was conducted. Details of this study are described in the following sections.

Materials and Methods

Modifications to the General Study Practice

The participants for the second study were volunteers from the Computer Science Department. To boost the incentives, a cash prize of $20 for the top performer was again offered.

To ensure that the subjects have enough time to explore all the materials given, there were no time restrictions. The subjects were allowed to participate in the study as long as they desired, which we hoped would provide them a more relaxing environment so that the results would be more accurate.

Only one algorithm, the “Dijkstra-Scholten Distributed Diffusing Termination Detection” algorithm, was used in the second study to avoid participant fatigue and the distraction of switching between the algorithms. It also provided sufficient time for the participants to explore all the learning materials of the algorithm more thoroughly.
In the first study, the researchers briefed the participants in each session about the study. In the second study, a PowerPoint slide show with audio was displayed at the beginning of each session to ensure consistency.

In addition, a demo with a simple algorithm (finding the greatest value in an unordered array) embedded in the same testing environment as the one used in the formal study was used as a training tool to allow the participants to explore all the features of the testing environment and to become familiar with it before the formal study. This also saved many explanations in the instructions, which otherwise might be ignored by the participants.

Implementation Modifications

Modifications to the Visualization

Prior to the second study, the visualization of the “Dijkstra-Scholten Distributed Diffusing Termination Detection” was improved to be more informative after a group review of the original version. In addition to several notation changes to the node and the network layout, a set of detailed legends, which makes the visualization more informative and more self-explainable, was added to the visualization as shown in Figure 5. This also reduced the lengthy textual explanations used in the first study, and gave the participants more vivid and related explanations of the visualization. The ability to navigate the 3D visualization was turned off and one fixed ideal view of the animation was displayed. The virtual tree view of the nodes was modified by adding arrows to indicate parent-children relationships more clearly.
Figure 5 **Screenshot taken from the visualization of the Termination Detection Algorithm** A set of legends is displayed at the left side of the window. The top layout indicates the virtual tree view (parent-child relationship) of the network. The bottom layout displays the network’s basic layout, each node’s status and attributes. The messages among nodes are represented by the small spheres on the connections between nodes, and different colors of the spheres indicate different types of messages.

**Modifications to the Testing Environment**

Some modifications were also made to the testing environment based on some problems found in the first study and to make it more integrated, interactive and easier to use. The major changes included:

- Changes to the page that links to all the embedded learning materials
- Changes to the control panel
- Changes to the pseudocode display

A screenshot taken from the page that links to all the embedded learning materials is shown in Figure 6.
Figure 6 Screenshot of a page from the testing environment. The group of buttons at left provides quick navigation to different learning materials embedded. The group of buttons at right provides controls of program execution.

The participants were no longer able to run the program with randomly chosen network configurations. This functionality was turned off because it was decided that the network configuration randomly chosen by the program might not have an interesting pattern that could help the participants learn. It happened often in the first study that the participants spent much time running the program several times and didn’t see any interesting patterns. Therefore, in the second study, a specific network configuration was chosen and two different complexity levels of the program execution exhibiting interesting patterns were prepared in advance by the researchers. The participants were allowed to choose either of these sample runs from the drop-down list, as shown in Figure 6, to display the corresponding output and/or to run the corresponding
visualization. The goal of this change is to save some time in the evaluation session and permit the participants to use the evaluation time more efficiently.

Four questions from the first study were modified to fit the specific network configuration displayed in the program output and/or visualization. A particular execution run was created for each of the three questions, and made available to the participants. The subjects in the Non-Viz group could view the program output corresponding to the specific question, while the subjects in the Viz group could view both the program output and the visualization corresponding to the specific question.

In the first study, a separate control panel was used to control only the algorithm animation itself. In the second study, to make all the learning materials more integrated, a more generic control panel was implemented to control both the animation and the output display pace. A group of buttons that link to different learning materials was also added to the control panel to facilitate navigation among different windows. In addition, the generic control panel was embedded into the code display window and each question page to facilitate control and navigation among different windows. A screenshot taken from the question page is shown in Figure 7.
Figure 7 Screenshot of a question page from the testing environment The control panel, the same as that displayed in Figure 6, is located at the top of the page. The question index is listed at left, and the question content is displayed at right. Clicking on the index item or the buttons at bottom will change the question page accordingly.

In addition, the Java-like pseudocode display, seen in Figure 8, also went though some improvements. First, the print statements in the code corresponding to the text displayed in the output display could be turned on or off by double clicking the statements in the code window. This interaction gave the participants the opportunity to filter the content in the output window to fit their own needs. This feature is similar to adding or deleting print statements in code to debug the program. A side panel was also added to the code display to indicate each process’s point in execution of the code.
Figure 8 **Screenshot of the Java-like pseudocode display** The letters on the left panel indicate different processes and the rectangles with different colors indicate each process’s point in execution of the code. The print statements in blue can be turned on and off by double clicking. The control panel at the bottom is the same as the one displayed in Figure 6.

**Modifications to the Questions**

The nine questions of the algorithm from the first study were reviewed by the research group and most of them were retained for the second study. In order to relate the questions more closely to the specific program output and visualization, four questions were modified to fit the specific network configuration depicted in the program output and/or visualization. A particular execution was created for each of the three questions in advance, and a partial output and a static image taken from the visualization were provided for the remaining question. Therefore, the participants could either view the program output and/or the visualization corresponding to the specific question.
One conceptual question was also added to the batch of questions for the second study. The complete list of questions are listed in Appendix K.

**Study Subjects**

Twenty volunteers recruited from the Computer Science Department, mainly graduate students and a few undergraduate students, participated in the study to compete for cash prizes.

**Procedure**

The twenty participants were randomly assigned to one of the two groups regardless of gender, age, or computer experience. One group learned about the algorithm with visualization (Viz group) in addition to the other learning materials and the other group (Non-Viz group) learned about the algorithm only with those learning materials mentioned in earlier sections. Among those twenty subjects, there were fifteen males and five females. Gender was not a factor considered in the experimental design. Thus, they were randomly assigned to the two groups. After the study, however, it turned out only one female from those five females was assigned to the Viz group and all the remaining four females were in the Non-Viz group.

**Data and Results**

**Correct Responses of the Questions**

Table 22 shows the numbers of participants in each group, and the means and the standard deviations of the number of questions answered correctly. Each participant answered ten questions for each algorithm. The Viz group has a dramatically larger mean than the mean from the Non-Viz group. The complete data are listed in Appendix M.

Table 22: Number of Correctly Answered Questions
T-tests were again used to determine whether there was a statistically significant difference between the means of the Viz and Non-Viz groups. The result showed that means of the number of correctly answered questions were significantly different with p<0.01 (t = 3.719, df = 18) between the two groups. This significance level means that the probability that the obtained difference would happen by sampling error is equal to or smaller than 0.01 [Bartz 1999]. Therefore, the null hypothesis can be rejected confidently.

In the second study, two categories of questions were identified in terms of the materials needed to produce a correct answer. The answers for one category of questions can be found in the general learning material, while the answers for the other category of rely on the specific textual output or visualization of the program. Table 23 and Table 24 show the data from those two types of questions with respect to the Viz and Non-Viz groups.

Table 23: Number of Correctly Answered General Questions (totally 6 questions)

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>10</td>
<td>4.2</td>
<td>1.32</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>10</td>
<td>2.2</td>
<td>1.75</td>
</tr>
</tbody>
</table>

Table 24: Number of Correctly Answered Questions with Specific Input (totally 4 questions)

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of Participants</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>10</td>
<td>3</td>
<td>1.25</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>10</td>
<td>1.5</td>
<td>0.71</td>
</tr>
</tbody>
</table>
In order to see if the visualization has a different impact on those two types of question, t-tests were carried out to compare the means of the Viz and Non-Viz groups regarding the different type of questions. It turned out that both of the means have significant difference with p<0.01 (t = 3.308, df = 18; t = 2.887, df = 18).

In addition, a “between-within” analysis was conducted to see if the different group (Viz/Non-Viz), the question type (general questions/questions with specific input), and the interaction between those two independent variables have an effect on the number of questions answered correctly. It turned out that neither the question type nor the interaction between those two variables had a statistically significant effect on the number of questions answered correctly, while the group assignment was the variable that had a significant effect on the number of questions answered correctly with p<0.0013.

**Participation Time**

The time recorded in this study was approximate, as in the first study. Table 25 shows the data of the time the participants spent answering all the questions, and Table 26 shows the data of the total time the participants spent in the evaluation system.

<table>
<thead>
<tr>
<th>Table 25. Time Spent on Answering Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Viz</td>
</tr>
<tr>
<td>Non-Viz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 26. Total Time Spent on Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Viz</td>
</tr>
<tr>
<td>Non-Viz</td>
</tr>
</tbody>
</table>
T-tests were also carried out and there was no statistically significant difference between the means of the Viz and Non-Viz groups in either of those collected times (t = 0.425, df = 18, p>0.05; t = 0.105, df = 18, p>0.05).

**Per Question Correct Response Rates**

With respect to the questions, Table 27 shows the rate at which the subjects answered each question correctly.

Table 27. Rate of people who answered the questions

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of participants</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>10</td>
<td>0.6</td>
<td>0.7</td>
<td>0.6</td>
<td>0.7</td>
<td>0.4</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>1.0</td>
<td>0.8</td>
</tr>
<tr>
<td>Non-Viz</td>
<td>10</td>
<td>0.1</td>
<td>0.7</td>
<td>0.4</td>
<td>0.3</td>
<td>0.1</td>
<td>0.1</td>
<td>0.4</td>
<td>0.4</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>Total</td>
<td>20</td>
<td>.35</td>
<td>0.7</td>
<td>0.5</td>
<td>0.5</td>
<td>.25</td>
<td>0.4</td>
<td>0.6</td>
<td>.65</td>
<td>0.8</td>
<td>.65</td>
</tr>
</tbody>
</table>

**Experimental Observations and Discussions**

In the second study, the Viz group had a dramatically greater mean than the Non-Viz group in terms of the number of questions answered correctly. The t-test indicated that those two means differed significantly with p<0.01. This supported our assumption that visualizations aid understanding of distributed algorithms.

In terms of the questions themselves, the Viz group outperformed the Non-Viz group for all questions. One participant in the Non-Viz group commented that for those questions that have specific program input, visualization would be more helpful and would make it easier to search for the information.

One participant in the Viz group, who correctly answered all the questions, commented that the visualization did help him better understand the algorithm. In
addition, when he couldn’t find the answer for the question in the textual algorithm description, he referred to the visualization.

Several participants in the Viz group also commented that the visualizations boosted their interest to learn the algorithm.

Those results and comments also justified to some degree that the modifications made to the second study were proper and effective.

In spite of the promising results obtained in the second study, however, there still were some problems observed. The biggest among them was the lack of incentives. As with the first study, the second study was conducted in a week that was only three weeks away from the end of semester. Most students were busy working on course projects. Although all of the subjects were volunteers and there were not any restrictions on the participation time, some of them still tended to finish the evaluation in a short time and did not explore all the materials thoroughly. Therefore, the results might have been affected. We note, however, that this factor is expected to affect the Viz and Non-Viz groups similarly.

The study was conducted in a separate lab where no other people were working. Thus, there was no distraction from other people. In most of the sessions, there was only one participant. However, there were sessions in which two participants attended. Again, the distraction from the other participant might have affected the participation time. We note that in the case of two subjects working simultaneously, one subject was assigned to the Viz group and the other subject was assigned to the Non-Viz group. Thus, this factor should affect both groups in the same way.
One of the participants in the Viz group, who answered seven questions correctly, commented that he did not refer to the visualization much because the graphical representation of the data in the visualization was different than the one in his mind. He used mostly the algorithm description, pseudocode and the output to learn about the algorithm and answer the questions. Ideally, an effective visualization should map well to the users’ mental model of the computation. However, because each individual may have a different mental model of the computation, it is a challenging task to develop a visualization fitting everyone’s mental model.

Although all the participants in the second study were Computer Science Department students, and mainly graduate students, the individual learning ability and cognitive processes are different. The participants were assigned to different groups randomly. However, this may still be a factor that affected the data due to the relatively small sample size. Gender differences were not considered when allocating participants to different groups. Males and females were randomly assigned to each group. After the study, however, it was found out that among all the five females participated in the second study, only one of them was assigned to the Viz group. Therefore, this might have an effect on the results obtained.
CHAPTER 5
CONCLUSIONS AND FUTURE WORK

Two studies were conducted to evaluate whether algorithm animations can be helpful in the teaching/learning of distributed algorithms.

Although the quantitative results from the first study did not give conclusive results, some problems in the materials and the practice of the study were identified. After improving the materials according to the findings from the first study, a second study was conducted. The quantitative results from the second study strongly support the notion that visualization can be useful in learning about distributed algorithms. Some comments given by the participants also support the benefits of the visualization.

However, some problems, which might have affected the results, were still revealed in the second study. According to the findings in the study, the following improvements or methods may be implemented in future work:

- Develop visualizations more consistent with user’s mental model of the computation. We note that this is an extremely challenging task due to differences between users.
- Make more detailed observations of the participants’ use of the learning materials during the study. Video may be taken to record all the details of the whole procedure. The “Think aloud” method might also be used to trace the participants’ mental activities.
• Have only one participant instead of multiple participants in one evaluation session to avoid distraction from other people.

• Conduct a mandatory interview after each session to ask more subjective questions about the visualization and the study, and ask for comments and suggestions. The combination of quantitative and qualitative data may provide more complete and reliable information as mentioned by Gurka and Citrin [1996].

• Use other measures in addition to answering questions regarding the algorithm to evaluate the participants’ understanding of the algorithm. For example, asking the subjects to make some modifications to the algorithm to include new functionalities after learning about it, or asking the subjects to apply the algorithm learned to build a more complex algorithm.

• Provide more incentives for the participants to put more effort into learning the algorithm.

• Recruit volunteers and conduct the study in an early period of the semester when students have more free time, and are thus more likely to spend sufficient time in the study.
REFERENCES


APPENDIX A

Dijkstra-Scholten Distributed Diffusing Termination Detection Algorithm

Description (by Mihail E. Tudoreanu)

The algorithm presented here assumes a distributed environment, which consist of a set of nodes that can communicate with each other only through message passing. The nodes are independent machines, like the ones that form the Internet, and they are linked to some of the other nodes (not necessarily all) by links or channels. A node knows only the number and the name of its neighbors, i.e., nodes to which it has a link. The only interaction between the nodes is via messages sent or received on the links. For example, it is impossible for a node to find the value of a variable A at one of the neighbors directly; it must send a message to ask for that value, and if the neighbor understands the request, it sends the value back.

A termination detection algorithm is designed to allow a node to determine if a task that is computed by multiple nodes has finished or not. Such a task (also referred to as an application) can be a project on which multiple people, each at home and using only email, work. The project might be a software product, which is divided among multiple people. Each person knows how to do certain things and cannot perform another. After they program some parts of a module (i.e. the networking functions), they might need to send that module to some other people so they can work out other aspects of the module (such as the dialog interface). At any time they might receive a request to do something from the people they know of (neighbors in the distributed environment). They can decide to work on that request, or to divide it among other people, or both.
The problem is how the initiator of the project detects the project’s completion. It is possible that people not known to the initiator are still working on the project. Furthermore, even if everybody responds that they are done, the assumption that the task is finished might not be correct. A request might be in transit between two people that are not working, and neither the sender or the receiver might be aware of the status of the request: the sender might assume that the request was already processed, and the receiver might not know that a work request is going to arrive. Thus, the task is not finished until the job in the request is performed.

The termination detection algorithm is design to provide the initiator of the project with an answer of whether the application (project) is finished. Note that the project itself can be any distributed application (any algorithm) not only the example presented above. Termination detection works on top of this underlying application, i.e., people are still performing their duties, but they also do the termination detection.

The termination detection needs to know three things about each node (person), whether they are idle or working, the number of other nodes to which this node has sent a request, and the person that gave them a request. Three variables are kept at each node (and because this is a distributed environment are known only to that node):

- a boolean named `idle`, which is false when the node is doing something.
- an integer named `count` which contains the number of application messages (requests) that have not yet been acknowledged. An application message is any message that would be sent by the application even when the termination detection is not used. An acknowledgement is a new type of message different from any application message.
-a string named parent that contains the name of the node that gave some work to this node.

The parent is initially null, to show the fact that the node is not working on a task. The initiator (called root) also does not have a parent. When a node receives a request (application message) and it has not already had a parent, then its parent becomes the node that sent the request. A node, however, does not change its parent if it is not null.

When a node is done, it sends an acknowledgement to its parent. The node replies immediately with an acknowledgement to every application message that was received after its parent was set.

The parent variable creates a virtual tree of all the nodes that are involved in computing the task (work for the underlying application). This tree can expand as more nodes are involved in the application and can shrink as they finish their parts, due to the distributed nature, it is possible for a part of the tree to shrink while another part expands. A node can enter and exit the tree multiple times for the same project.

A node exits the tree only when it knows it is done (although it may be activated again later). For a node to be done, it is not enough to be idle, it also needs to have no other children. Otherwise, those children might be still working on the project, and the detection might give false answers.

The project is done when the root is done. This property can be detected by the root and ensures that no node is working on the project.

Question instructions:

- assume that at most one application task is performed in the entire network at one time
APPENDIX B

Distance Vector (Bellman-Ford) Routing Algorithm Description

(by Mihail E. Tudoreanu)

The algorithm presented here assumes a distributed environment, which consist of a set of nodes that can communicate with each other only through message passing. The nodes are independent machines, like the ones that form the Internet, and they are linked to some of the other nodes (not necessarily all) by links or channels. A node knows only the number and the name of its neighbors, i.e., nodes to which it has a link. The only interaction between the nodes is via messages sent or received on the links. For example, it is impossible for a node to find the value of a variable A at one of the neighbors directly; it must send a message to ask for that value, and if the neighbor understands the request, it sends the value back.

A routing algorithm is designed to allow a node to address messages to any reachable node in the network. Because initially only the neighbors are known, it is impossible for a node to send a message to any other nodes except its neighbors. The routing algorithm assumes that nodes cooperate and forward each other’s messages. In this way, a node can communicate to any other reachable node.

A routing algorithm must handle crashes and additions of both links and nodes. If a link or node fails, it may be necessary to use a different route between the same two nodes. Similarly, additional links or nodes might offer shorter routes or might make it possible to reach new nodes.

In distance vector routing a node knows, based on the destination of the message, to which neighbor to forward that message (next hop). Therefore, a node keeps a table
that has three fields: destination, next hop, and distance. The first two fields are used for forwarding messages, while the third is employed to choose the shortest path. In most cases the shortest path excludes loops.

To build the routing table, neighbors tell each other what are the destinations they have heard of and what is the distance to that destination. When an update message is received (from one of the neighbors), a node checks to see if new destinations are included or if a shorter route exists to an already known destination.

Initially, a node knows only of its neighbors and starts with a table that contains one entry for each neighbor, with that neighbor as destination, as next hop, and with the distance equals to one. That implies that to send a message to a neighbor, the next hop is the neighbor itself at distance of one.

The table is modified only when an update message with new destinations or shorter distances is received, when a failure is detected, or when a new link appears (for simplicity, we don’t discuss how a link failure or addition is sensed). In case of a link failure, all destinations that use that link are removed from the table. Note that a node crash appears to all its neighbors as a link failure; it is impossible for the neighbor to determine whether the link or the node has failed.

An update message is sent to all the neighbors in three cases:

-when the table is changed.

-when a new link is detected.

-when an update message from a neighbor does not contain a destination known to this node (The can happen after one of the neighbor’s links has failed), or it contains a longer route to a destination.
APPENDIX C

Dijkstra-Scholten Distributed Diffusing Termination Detection Algorithm

Pseudocode (by Mihail E. Tudoreanu and Ritu Dhawan)

//Node Class: each node contains pipes to read incoming messages & send
//outgoing messages. Non-root nodes have a parent.
//Root nodes start the thread.

public class Node {
    int parent = -1; //parent ID or -1 for no parent
    int n = 0; //number of nodes
    public Pipe[] outgoing; //outgoing link to node i; null for no link
    public Pipe[] incoming; //incoming link from node i, null for no link
    public boolean root = false; //true if this is a root
    public int myid = 0; //the ID of this node
    boolean idle = true; //true if this node is idle (not working)
    int count = 0; //counter to keep track of messages not acknowledged

    public Node(int num, int id) {
        n = num; myid = id; //initialize the node
        outgoing = new Pipe[n]; incoming = new Pipe[n]; //n is max num in/out links
        for (int i = 0; i < n; i++) {
            outgoing[i] = null; incoming[i] = null; //initially unconnected
        }
    }

    public void addOutLink(Pipe p, int neigh) { //add outgoing link to node neigh
        System.out.println("NODE " + myid + ": outgoing link to " + neigh);
        outgoing[neigh] = p;
    }

    public void addInLink(Pipe p, int neigh) { //add incoming link from node neigh
        System.out.println("NODE " + myid + ": incoming link from " + neigh);
        incoming[neigh] = p;
    }

    public void run() { //what this node does
        int aux;
        //write the current values of the variables
        System.out.println("NODE " + myid + ": +" + root + "= root");
        System.out.println("NODE " + myid + ": +" + counter + "= counter");
        System.out.println("NODE " + myid + ": +" + idle + "= idle");
        System.out.println("NODE " + myid + ": +" + parent + "= parent");
        //the underlying application may send messages to other nodes
        if (root == true)
            Application.sendMsg();

        while (true) { //this is a daemon, so it runs continuously
            for (int i = 0; i < n; i++) { //check each possible link of the node
                if (incoming[i] != null) { //there is a link from i
                    aux = incoming[i].get(); //get the message if any
                }
            }
        }
    }
}
if(aux != NO_MSG){
    //if there is a message
    System.out.println("NODE "+myid+": message type "+aux+" from node "+i);
    if(aux == APP_MSG){
        //application message
        idle = false; System.out.println("NODE "+myid+": idle is "+idle); //start working
        if(parent == -1 && root == false){
            parent = i; System.out.println("NODE "+myid+": parent is "+parent);
        } else {
            outgoing[i].put(ACK_MSG);
            System.out.println("NODE "+(char)('A'+myid)+": Sending Ack to the parent "+(char)('A'+Parent));
        }
        //the underlying application may send messages to other nodes
        Application.sendMsg();
        System.out.println("NODE "+myid+": App message sent to "+otherNodes);
        System.out.println("NODE "+myid+": counter is "+counter);
    } //if (aux == APP_MSG)
    if(aux == Pipe.Ack){ //message is an acknowledgement
        count--;
        System.out.println("NODE "+myid+": counter is "+count);
    } //if (aux != NO_MSG)
} //if (incoming[i] != null)
}//end of for loop
Application.checkIdle(); //check if the underlying application finished; this might modify idle
System.out.println("NODE "+myid+": idle is "+idle);
if(root && idle == true && count == 0)
    System.out.println("Node "+myid+": task finished");
if(!root && parent != -1 && idle == true && count == 0){ //acknowledge to parent
    outgoing[parent].put(ACK_MSG);
    System.out.println("NODE "+myid+": ack sent to parent "+parent);
    parent = -1;
    System.out.println("NODE "+myid+": parent is "+parent);
} //end of while(true)
}//end of run method
}//end of class Node
APPENDIX D

Distance Vector (Bellman-Ford) Routing Algorithm Pseudocode
(by Mihail E. Tudoreanu and Ritu Dhawan)

//Node Class: each node contains pipes for incoming and outgoing links.
//each node is a separate process

public class Node extends Thread{
    boolean changed = false;   //true if the table has changed since the last update
    int n= 0;                              //the max number of nodes in the network
    public int[][] table;               //routing table
    public Pipe[] outgoing;      //outgoing channels
    public Pipe[] incoming;      //incoming channels
    public int myid;                  //this node's id

    public Node(int num, int id){
        n = num;
        myid = id;
        //reset the links
        outgoing = new Pipe[n];
        incoming = new Pipe[n];
        for(int i=0; i<n; i++){
            outgoing[i] = null;
            incoming[i] = null;
        }
        //reset the table
        Table = new int[n][3];
        for(int x=0; x<n; x++)
            for(int y=0; y<3; y++)
                Table[x][y]=0;
    }

    public void addOutLink(Pipe p, int neigh){  //method to add outgoing link
        System.out.println("Node "+myid+" outgoing link to "+neigh);
    }

    public void addInLink(Pipe p, int neigh){ //method to add incoming link
        System.out.println("Node "+myid+" incoming link from "+neigh);
    }

    public void run(){
        //updates to neighbors are always going to contain the self destination at distance 0
        //the destination table contains the self destination, with
        //the next hop the node itself, at distance 0
        table.addEntry(myid, myid, 0);
        table[0]=myid;
    }
}
table[0][1]=myid;
table[0][2]=0;

int l=0;
for(int i=0; i<n; i++){
    //check each channel of that node
    if(incoming[i] != null){
        table.addEntry(i, i, 1);
    }
}

System.out.println("Node "+myid+: "+printTable(table));

//this broadcasts the destinations and their corresponding
//distances to all neighbors
broadcastTable();

while(true){
    for(int x=0; x<n; x++){
        if(incoming[x]!=null){  //connected to x
            if(incoming[x].failed){  //if link failed
                //invalidate all table entries that have x as the next hop
                invalidateEntriesOf(x);
                System.out.println("Node "+myid+: "+invalidating next hop "+x);
                System.out.println("Node "+myid+: table "+printTable(table));

                //make sure the new table gets broadcast
                changed=true;
            }
            else if(incoming[x].hasMsgs()){//a table was received
                comTable = incoming[x].get();
                System.out.println("Node "+myid+: received from "+x+" update "+printTable(comTable));

                for(dest in comTable){
                    //get the entry for "dest" in the table
                    TableEntry e=table.getEntryFor(dest);
                    if(e==null){ //don't know about dest
                        table.addEntry(dest, x, 1 + comTable.distanceTo(dest));
                        System.out.println("Node "+myid+: dest "+ dest+", next hop "+x+", distance " +table.distanceTo(dest));
                        changed = true;
                    }
                }
            }
        }
    }
}
else if(comTable.distanceTo(dest)+1 < table.distanceTo(dest)){
    //record a shorter distance to an existing destination
    table.modifyEntry(dest, x, comTable.distanceTo(dest));
    System.out.println("Node "+myid+: dest "+ dest+", next hop "+x+", distance " +table.distanceTo(dest));
    changed = true;
}
}

*/
for(int l=0; l<n; l++){
    if(table[l][1]==x){ //if the next hop of this route is
        //the neighbor sending the update message
        int p = 0;
        for(; p<n; p++){
            if(table[l][0]==comTable.aux[p][0]){ //if next hop is the neighborhood
                send update
                if(table.distanceTo(dest)!=comTable.distanceTo(dest)+1){
                    table.addEntry(dest, x,
                    comTable.distanceTo(dest)+1);
                    //update the distance to the destination
                    System.out.println("Node "+myid+": dest "+
dest+, next hop "+x+, distance"
                    +table.distanceTo(dest));
                    changed = true;
                }
                break;
            }
        } //end of if outer
    } //end of for
    if(table.getDestination(desc)=null){
        //if destination not found in update message invalidate the entry
        System.out.println("NODE: "+myid+ " invalidating next hop "+x);
        invalidateEntriesAt(l);
        changed = true;
    }
} //end of if
if(table[l][1]!=x){ //if the next hop of this route is not the neighbor
    //sending this update message
    if(!getDestination(table[l][0], comTable.aux) ||
    getDestination(table[l][0], comTable.aux) &&
    comTable.distanceTo(desc)>table.distanceTo(desc)){
        //if destination found and the distance of received
        //table is greater than the distance to destination
        //of the current node table make changed true
        changed = true;
    }
} //end of for
System.out.println("Node "+myid+": table "+printTable(table));
} //end of else
} //if check for no table
} //if check for no pipe

if(changed){
    broadcastTable();
    changed=false;
} //end of if to check changed
} //end of outer for
} //end of run method

returns the row number which has destination equals to x
public int getEntryFor(int x){
    for(int i=0; i<n; i++)
    

if(table[i][0]==x){
    return i;
}
return -1;
}

//returns the next available row
public int nextAvailableRow( ){
    for(int i=0; i<n; i++)
        if(table[i][0]==-1)
            return i;
    return -1;
}

//invalidate the entry in case of the failure
public void invalidateEntriesOf(int x){
    for(int i=0; i<n; i++)
        if(table[i][1] == x){
            table[i][0]=-1;
            table[i][1]=-1;
            table[i][2]=-1;
        }
}

//returns the row if destination is found
public int getDestination(int d, int[][] compare){
    for(int j=0; j<n; j++)
        if(d == compare[j][0])
            return j;
    return -1;
}

//prints the table
public String printTables(int[][] a){
    for(int i=0; i<a.length; i++)
        for(int j = 0; j<a[i].length; j++)
            s =s+ a[i][j]+" ";
    s= s+"\n";
    return s;
}

//method to Brodcast the tables
public void broadcastTable(){
    for(int j = 0; j<n; j++)  {
        if(outgoing[j] != null) {
            System.out.println("Updated Table sent from Node: " + myid+" to Node: "+j+"\n");
            outgoing[j].put(Table);
        }
    }
}
} //end of start method
} //end of node
APPENDIX E

Screenshots of All the Screens form the Testing Environment for the First Study

Login Page

Java-like Pseudocode Display
Output of Termination Detection Algorithm

NODE 1: counter is 2
NODE 1: message type 1 from node 2
NODE 1: idle is false
NODE 3: message type 1 from node 2
NODE 3: idle is false
NODE 3: message type -1 from node 4
NODE 3: counter is 2
NODE 2: message type -1 from node 0
NODE 2: counter is 6
NODE 2: message type -1 from node 1
NODE 2: counter is 5
NODE 2: message type -1 from node 3
NODE 2: counter is 4
NODE 0: message type -1 from node 2
NODE 0: counter is 3
NODE 2: message type 1 from node 0

Textual Output Display

Usability Study

Question 1:
Which of the following choices best describes the condition under which a node will send an acknowledgment to its parent node?

Answer:
- (a) The node's count goes to zero;
- (b) Timeout;
- (c) The node's status is idle;
- (d) The node's count goes to zero and timeout;
- (e) The node's count goes to zero and status is idle;
- (f) None of the above.

First Question Page
Thank you for taking time participating in our usability study. You answered 2 questions correctly out of 9 questions. The correct rate is 0.22222222.

Before exit the system, please answer a couple of simple questions below, which will be helpful to the data analysis.

Did you know this algorithm before?
- Yes, I knew this algorithm.
- No, I didn’t know this algorithm.

Did you have enough time to answer all the questions?
- Yes, I had enough time to finish all the questions.
- No, I ran out of time.

We would like you to give us any comments or suggestions about our visualization or the usability study. Please enter in the area below and click the “Submit” button when you are done.

Result page where comment can be entered

Thank you for taking time participating in our usability study, your evaluation will be very helpful to us.

Final logout page
APPENDIX F

Questions of the Dijkstra-Scholten Distributed Diffusing Termination Detection

Algorithm for the First Study

1. Which of the following choices best describes the condition under which a node will send an acknowledgement to its parent node?

A. The node's count goes to zero;
B. Timeout;
C. The node's status is idle;
D. The node's count goes to zero and timeout;
E. The node's count goes to zero and status is idle;
F. None of the above.

2. Given below is a list of all the nodes in the network and their parents (only null or non-null values, not specific parents) and corresponding count values. Please tell which nodes are participants in the task for which the termination detection is being performed.

Node A: count = 0, parent = non-null;
Node B: count = 3, parent = non-null;
Node C: count = 0, parent = non-null;
Node D: count = 1, parent = non-null;
Node E: count = 0, parent = null;
Node F: count = 2, parent = null;
Node G: count = 0, parent = non-null;
Node H: count = 0, parent = non-null.

A. BDF;
B. ABCDGH;
C. ABCDFGH;
D. None of the above;
E. The information is insufficient.

3. Given below is a list of all the nodes in the network and their parents and corresponding count values (same as in the previous question). Please tell which node is the root.

Node A: count = 0, parent = non-null;
Node B: count = 3, parent = non-null;
Node C: count = 0, parent = non-null;
Node D: count = 1, parent = non-null;
Node E: count = 0, parent = null;
Node F: count = 2, parent = null;
Node G: count = 0, parent = non-null;
Node H: count = 0, parent = non-null.
A. E;
B. F;
C. E and F;
D. None of the above;
E. The information is insufficient.

4. Is it possible to have two nodes listed below as participants in the task for which the termination detection is being performed?

Node A: count = 2, parent = B;
Node B: count = 3, parent = A.

A. Yes;
B. No;
C. The information is insufficient.

5. A network has five nodes. What would be the maximum count value of a node if all nodes could participate in the task for which the termination detection is being performed?

A. 5;
B. 4;
C. 1;
D. None of the above.

6. Given below is a list of all the nodes in the network and their parents. Please tell which of the following values could not be Node C's count value.

Node A: parent = C;
Node B: parent = C;
Node C: parent = null;
Node D: parent = A;
Node E: parent = null.

A. 1;
B. 3;
C. 4;
D. None of the above;
E. The information is insufficient.

7. Given below is a list of all the nodes in the network and their parents and corresponding count values. Assuming there are no message in transit and no further application messages will be sent, how many acknowledgments need to be sent until the root detects the termination of the task?

Node A: count = 0, parent = F;
Node B: count = 3, parent = F;
Node C: count = 0, parent = D;
Node D: count = 1, parent = B;  
Node E: count = 0, parent = null;  
Node F: count = 2, parent = null;  
Node G: count = 0, parent = B;  
Node H: count = 0, parent = B.

A. 3;  
B. 4;  
C. 6;  
D. None of the above;  
E. The information is insufficient.

8. Given below is a list of all the nodes in the network and their parents and corresponding count values (same as the network in the previous question). B's status is idle, which nodes are the nodes Node B is waiting on?
   Node A: count = 0, parent = F;  
   Node B: count = 3, parent = F;  
   Node C: count = 0, parent = D;  
   Node D: count = 1, parent = B;  
   Node E: count = 0, parent = null;  
   Node F: count = 2, parent = null;  
   Node G: count = 0, parent = B;  
   Node H: count = 0, parent = B.

A. E, F;  
B. D, F, H;  
C. D, G, H;  
E. None of the above.

9. A modification to the algorithm is to allow multiple tasks being performed in the network at the same time, which means having multiple virtual trees in the network and each tree will handle its task simultaneously with other trees. Each node would be possible to participate in more than one task. Termination detection is considered to be correct if a root detects the termination after the task initiated by that root is terminated. In other words, there is no node working on that task any more, but it is possible that there are nodes still working on other tasks. What is the minimum modification to the termination detection algorithm to make this multiple-task termination detection work?

A. Having multiple parents in each node;  
B. Having multiple counts in each node;  
C. Having multiple parents and count in each node.
APPENDIX G

Questions of the Distance Vector Routing Algorithm for the First Study

1. In the routing table of a node, how many entries (routes) exist for one destination?
   
   A. One;
   B. As many as the number of the available different routes from this node to that destination;
   C. Depends on the information exchanged, between one and the number of all the available different routes;
   D. None of the above.

2. In what situation would the routing table entries be changed in response to an update message?

   A. The distance to a destination is less, either through the same or different next hop;
   B. If a route's next hop is the neighbor that sent this update message and the distance is changed, either more or less;
   C. The update message includes new destinations;
   D. a and b;
   E. a and c;
   F. a, b and c.

3. Given below is a routing table of a node, what will its update message be?

   A. 
   
<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>6</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>B</td>
</tr>
</tbody>
</table>

   B. 
   
<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>6</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
</tr>
</tbody>
</table>

   C. 
   
<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>6</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>B</td>
</tr>
</tbody>
</table>
D. None of the above.

4. Given below is a list of all the nodes and links in a network. What will be the routing table of A, assuming the distance of each link is one and all the route information has been exchanged (no routing table changes for a long time)?
Nodes: A, B, C, D, E, F;
Links: (A, C), (A, D), (B, D), (B, E), (D, E).

A. Destination | Distance | Next Hop
-----------------|-----------|--------
B                | 2         | D      
C                | 1         | C      
D                | 1         | D      
E                | 2         | D      
F                | infinity  | null   

B. Destination | Distance | Next Hop
-----------------|-----------|--------
B                | 2         | D      
C                | 1         | C      
D                | 1         | D      
E                | 2         | D      

C. Destination | Distance | Next Hop
-----------------|-----------|--------
B                | 2         | D      
B                | 3         | D      
C                | 1         | C      
D                | 1         | D      
E                | 2         | D      
E                | 3         | D      

D. Destination | Distance | Next Hop
-----------------|-----------|--------
B                | 2         | D      
B                | 3         | D      
C                | 1         | C      
D                | 1         | D      
E                | 2         | D      
E. None of the above.

5. Given below are a routing table of a node and an update message from one of its neighbors. What will the updated routing table be?

Original Routing Table:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>B</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>B</td>
</tr>
<tr>
<td>E</td>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>F</td>
<td>2</td>
<td>B</td>
</tr>
<tr>
<td>G</td>
<td>2</td>
<td>A</td>
</tr>
</tbody>
</table>

Update Message from A:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>6</td>
</tr>
<tr>
<td>E</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>10</td>
</tr>
<tr>
<td>G</td>
<td>3</td>
</tr>
<tr>
<td>H</td>
<td>4</td>
</tr>
</tbody>
</table>

A. Destination     Distance     Next Hop
-------------------------------------------
| A          | 1          | A          |
| B          | 1          | B          |
| C          | 4          | A          |
| D          | 7          | A          |
| E          | 6          | A          |
| F          | 11         | A          |
| G          | 4          | A          |

B. Destination     Distance     Next Hop
-------------------------------------------
| A          | 1          | A          |
| B          | 1          | B          |
| C          | 4          | A          |
| D          | 7          | A          |
| E          | 6          | A          |
F  |  11  |  A  
G  |   4  |  A  
H  |   5  |  A  

C. Destination  |  Distance  |  Next Hop  
-----------------|-------------|------------
A  |  1  |  A  
B  |  1  |  B  
C  |  4  |  A  
D  |  5  |  B  
E  |  6  |  A  
F  |  2  |  B  
G  |  4  |  A  

D. Destination  |  Distance  |  Next Hop  
-----------------|-------------|------------
A  |  1  |  A  
B  |  1  |  B  
C  |  4  |  A  
D  |  5  |  B  
E  |  6  |  A  
F  |  2  |  B  
G  |  4  |  A  
H  |  5  |  A  

E. Destination  |  Distance  |  Next Hop  
-----------------|-------------|------------
A  |  1  |  A  
B  |  1  |  B  
C  |  8  |  B  
D  |  5  |  B  
E  |  6  |  A  
F  |  2  |  B  
G  |  2  |  A  
H  |  5  |  A  

F. None of the above.

6. The last four questions are related to the self-stabilization of the algorithm. The same initial scenario as in this question applies to all of the next three questions as well.
Suppose there are three nodes A, B, and C, in the network and their routing tables (after exchange update information) are listed below:

Nodes: A, B, C;
Links: (A, B), (A, C), (B, C).

Routing Table of Node A:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>C</td>
</tr>
</tbody>
</table>

Routing Table of Node B:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>C</td>
</tr>
</tbody>
</table>

Routing Table of Node C:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Distance</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>B</td>
</tr>
</tbody>
</table>

Assume the link between B and C fails first, followed by the link between A and C. Does B know immediately that the link between A and C has failed, and does A know immediately that the link between B and C failed?

A. Yes, both A and B send an update message to the other;
B. No, A and B might never know that the other route to C has failed.

7. For the scenario described in the previous question, is it possible that A receives an update message from B with distance to C equals one after the link between B and C failed?

A. Yes, but A will ignore that update message from B and won't change the routing table;
B. Yes, and A will change its routing table accordingly; however, it will be corrected later;
C. Yes, but this will only happen if B malfunctions;
D. No, this sequence of events is impossible.

8. After the situation described in question 6, assume that A receives an update message that appears to be sent from B, with distance to C equals one (after the links between B and C, A and C have already been down). Later on, A sends out an update message. What information will be included in A's update message?

A. Yes, but A will ignore that update message from B and won't change the routing table;
A. Information only about the route to B;
B. Information about route to B and C, but B will ignore the information about route to C, and will inform A back that A's information about route to C is incorrect;
C. Information about route to B and C, and B will update its routing table accordingly (i.e., has a route to C via A and with distance = 2).

9. After the situation described in question 6, is it possible that at a later time, more update messages containing a (non-existent) route to C are going to be sent between A and B? Notice that this can make A and B every so often assume that they have a route to C.

A. Yes, this scenario is possible;
B. No, this scenario is impossible.
APPENDIX H

Dijkstra-Scholten Distributed Diffusing Termination Detection Algorithm Data from the First Study

The number of question answered correctly (9 questions in total) and time spent on answering all the questions

<table>
<thead>
<tr>
<th>Subject</th>
<th>sub-session</th>
<th># Correct Answers</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz_A</td>
<td>seond</td>
<td>5</td>
<td>13.43</td>
</tr>
<tr>
<td>Viz_B</td>
<td>seond</td>
<td>4</td>
<td>10.65</td>
</tr>
<tr>
<td>Viz_C</td>
<td>seond</td>
<td>1</td>
<td>10.23</td>
</tr>
<tr>
<td>Viz_D</td>
<td>seond</td>
<td>2</td>
<td>7.83</td>
</tr>
<tr>
<td>Viz_E</td>
<td>seond</td>
<td>2</td>
<td>6.28</td>
</tr>
<tr>
<td>Viz_F</td>
<td>seond</td>
<td>1</td>
<td>6.2</td>
</tr>
<tr>
<td>Viz_G</td>
<td>seond</td>
<td>5</td>
<td>17.83</td>
</tr>
<tr>
<td>Viz_H</td>
<td>seond</td>
<td>3</td>
<td>8.62</td>
</tr>
<tr>
<td>Viz_I</td>
<td>first</td>
<td>8</td>
<td>28.15</td>
</tr>
<tr>
<td>Viz_J</td>
<td>first</td>
<td>4</td>
<td>9.4</td>
</tr>
<tr>
<td>Viz_K</td>
<td>first</td>
<td>4</td>
<td>6.85</td>
</tr>
<tr>
<td>Viz_L</td>
<td>first</td>
<td>6</td>
<td>18.33</td>
</tr>
<tr>
<td>Viz_M</td>
<td>first</td>
<td>3</td>
<td>34.9</td>
</tr>
<tr>
<td>Viz_N</td>
<td>first</td>
<td>1</td>
<td>10.77</td>
</tr>
<tr>
<td>Viz_O</td>
<td>first</td>
<td>2</td>
<td>24.43</td>
</tr>
<tr>
<td>Viz_P</td>
<td>first</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>Viz_Q</td>
<td>first</td>
<td>5</td>
<td>5.18</td>
</tr>
<tr>
<td>Viz_R</td>
<td>first</td>
<td>4</td>
<td>18.77</td>
</tr>
<tr>
<td>Viz_S</td>
<td>first</td>
<td>2</td>
<td>18.67</td>
</tr>
<tr>
<td>Non-Viz_A</td>
<td>first</td>
<td>3</td>
<td>8.92</td>
</tr>
<tr>
<td>Non-Viz_C</td>
<td>first</td>
<td>6</td>
<td>25.43</td>
</tr>
<tr>
<td>Non-Viz_D</td>
<td>first</td>
<td>3</td>
<td>22.63</td>
</tr>
<tr>
<td>Non-Viz_E</td>
<td>first</td>
<td>3</td>
<td>14.35</td>
</tr>
<tr>
<td>Non-Viz_F</td>
<td>first</td>
<td>5</td>
<td>13.88</td>
</tr>
<tr>
<td>Non-Viz_G</td>
<td>first</td>
<td>4</td>
<td>13.43</td>
</tr>
<tr>
<td>Non-Viz_H</td>
<td>first</td>
<td>2</td>
<td>29.58</td>
</tr>
<tr>
<td>Non-Viz_I</td>
<td>first</td>
<td>0</td>
<td>36.23</td>
</tr>
<tr>
<td>Non-Viz_J</td>
<td>first</td>
<td>5</td>
<td>35.22</td>
</tr>
<tr>
<td>Non-Viz_K</td>
<td>seond</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>Non-Viz_L</td>
<td>seond</td>
<td>2</td>
<td>8.4</td>
</tr>
<tr>
<td>Non-Viz_M</td>
<td>seond</td>
<td>4</td>
<td>12.83</td>
</tr>
<tr>
<td>Non-Viz_N</td>
<td>seond</td>
<td>4</td>
<td>5.55</td>
</tr>
<tr>
<td>Non-Viz_O</td>
<td>seond</td>
<td>4</td>
<td>19.88</td>
</tr>
<tr>
<td>Non-Viz_P</td>
<td>seond</td>
<td>7</td>
<td>21.07</td>
</tr>
<tr>
<td>Non-Viz_Q</td>
<td>seond</td>
<td>6</td>
<td>10.32</td>
</tr>
<tr>
<td>Non-Viz_R</td>
<td>seond</td>
<td>2</td>
<td>11.15</td>
</tr>
<tr>
<td>Non-Viz_S</td>
<td>seond</td>
<td>6</td>
<td>12.97</td>
</tr>
<tr>
<td>Non-Viz_T</td>
<td>seond</td>
<td>5</td>
<td>15.62</td>
</tr>
</tbody>
</table>
APPENDIX I

Distance Vector (Bellman-Ford) Routing Algorithm Data

The number of question answered correctly (9 questions in total) and time spent on answering all the questions

<table>
<thead>
<tr>
<th>Subject</th>
<th>sub-session</th>
<th># Correct Answers</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Viz_A</td>
<td>first</td>
<td>4</td>
<td>26.28</td>
</tr>
<tr>
<td>Non-Viz_B</td>
<td>first</td>
<td>6</td>
<td>10.33</td>
</tr>
<tr>
<td>Non-Viz_C</td>
<td>first</td>
<td>4</td>
<td>32.6</td>
</tr>
<tr>
<td>Non-Viz_D</td>
<td>first</td>
<td>6</td>
<td>8.48</td>
</tr>
<tr>
<td>Non-Viz_E</td>
<td>first</td>
<td>3</td>
<td>12.87</td>
</tr>
<tr>
<td>Non-Viz_F</td>
<td>first</td>
<td>4</td>
<td>27.28</td>
</tr>
<tr>
<td>Non-Viz_G</td>
<td>first</td>
<td>2</td>
<td>26.9</td>
</tr>
<tr>
<td>Non-Viz_H</td>
<td>first</td>
<td>3</td>
<td>16.05</td>
</tr>
<tr>
<td>Non-Viz_I</td>
<td>second</td>
<td>4</td>
<td>8.73</td>
</tr>
<tr>
<td>Non-Viz_J</td>
<td>second</td>
<td>3</td>
<td>16.3</td>
</tr>
<tr>
<td>Non-Viz_K</td>
<td>second</td>
<td>5</td>
<td>10.02</td>
</tr>
<tr>
<td>Non-Viz_L</td>
<td>second</td>
<td>5</td>
<td>16.63</td>
</tr>
<tr>
<td>Non-Viz_M</td>
<td>second</td>
<td>2</td>
<td>18.85</td>
</tr>
<tr>
<td>Non-Viz_N</td>
<td>second</td>
<td>3</td>
<td>13.9</td>
</tr>
<tr>
<td>Non-Viz_O</td>
<td>second</td>
<td>5</td>
<td>12.38</td>
</tr>
<tr>
<td>Non-Viz_P</td>
<td>second</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Non-Viz_Q</td>
<td>second</td>
<td>4</td>
<td>7.37</td>
</tr>
<tr>
<td>Non-Viz_R</td>
<td>second</td>
<td>2</td>
<td>14.6</td>
</tr>
<tr>
<td>Non-Viz_S</td>
<td>second</td>
<td>4</td>
<td>6.95</td>
</tr>
<tr>
<td>Viz_A</td>
<td>second</td>
<td>3</td>
<td>13.7</td>
</tr>
<tr>
<td>Viz_B</td>
<td>second</td>
<td>3(3)</td>
<td>15.68</td>
</tr>
<tr>
<td>Viz_C</td>
<td>second</td>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>Viz_D</td>
<td>second</td>
<td>5</td>
<td>21.92</td>
</tr>
<tr>
<td>Viz_E</td>
<td>second</td>
<td>5</td>
<td>13.8</td>
</tr>
<tr>
<td>Viz_F</td>
<td>second</td>
<td>2</td>
<td>12.07</td>
</tr>
<tr>
<td>Viz_G</td>
<td>second</td>
<td>6</td>
<td>9.73</td>
</tr>
<tr>
<td>Viz_H</td>
<td>second</td>
<td>1</td>
<td>3.1</td>
</tr>
<tr>
<td>Viz_I</td>
<td>second</td>
<td>3</td>
<td>11.77</td>
</tr>
<tr>
<td>Viz_J</td>
<td>second</td>
<td>3</td>
<td>19.98</td>
</tr>
<tr>
<td>Viz_K</td>
<td>first</td>
<td>3</td>
<td>15.73</td>
</tr>
<tr>
<td>Viz_L</td>
<td>first</td>
<td>7</td>
<td>10.35</td>
</tr>
<tr>
<td>Viz_M</td>
<td>first</td>
<td>3</td>
<td>23.32</td>
</tr>
<tr>
<td>Viz_N</td>
<td>first</td>
<td>5</td>
<td>13.82</td>
</tr>
<tr>
<td>Viz_O</td>
<td>first</td>
<td>6</td>
<td>19.58</td>
</tr>
<tr>
<td>Viz_P</td>
<td>first</td>
<td>6</td>
<td>29.53</td>
</tr>
<tr>
<td>Viz_Q</td>
<td>first</td>
<td>1</td>
<td>37.55</td>
</tr>
<tr>
<td>Viz_R</td>
<td>first</td>
<td>3</td>
<td>33.28</td>
</tr>
<tr>
<td>Viz_S</td>
<td>first</td>
<td>5</td>
<td>27.63</td>
</tr>
<tr>
<td>Viz_T</td>
<td>first</td>
<td>6</td>
<td>25.83</td>
</tr>
</tbody>
</table>
## APPENDIX J

### Per Questions Data from the First Study

**Number of people who answered the questions correctly**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Num. of participants</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detection</td>
<td>37</td>
<td>20</td>
<td>10</td>
<td>21</td>
<td>23</td>
<td>5</td>
<td>7</td>
<td>21</td>
<td>22</td>
<td>7</td>
</tr>
<tr>
<td>Routing</td>
<td>38</td>
<td>14</td>
<td>16</td>
<td>15</td>
<td>17</td>
<td>18</td>
<td>18</td>
<td>14</td>
<td>14</td>
<td>21</td>
</tr>
</tbody>
</table>

**Rate of people who answered the questions correctly: Termination Detection Algorithm - Viz group**

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of participants</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>10</td>
<td>0.6</td>
<td>0.3</td>
<td>0.6</td>
<td>0.6</td>
<td>0.3</td>
<td>0.2</td>
<td>0.5</td>
<td>0.5</td>
<td>0.2</td>
</tr>
<tr>
<td>Second</td>
<td>8</td>
<td>0.375</td>
<td>0</td>
<td>0.375</td>
<td>0.875</td>
<td>0.125</td>
<td>0</td>
<td>0.5</td>
<td>0.375</td>
<td>0.25</td>
</tr>
<tr>
<td>Total</td>
<td>18</td>
<td>0.5</td>
<td>0.167</td>
<td>0.5</td>
<td>0.722</td>
<td>0.222</td>
<td>0.111</td>
<td>0.5</td>
<td>0.444</td>
<td>0.222</td>
</tr>
</tbody>
</table>

**Rate of people who answered the questions correctly: Termination Detection Algorithm - Non-Viz group**

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of participants</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>9</td>
<td>0.667</td>
<td>0.222</td>
<td>0.667</td>
<td>0.667</td>
<td>0.111</td>
<td>0.333</td>
<td>0.555</td>
<td>0.222</td>
<td></td>
</tr>
<tr>
<td>Second</td>
<td>10</td>
<td>0.5</td>
<td>0.5</td>
<td>0.6</td>
<td>0.4</td>
<td>0.1</td>
<td>0.4</td>
<td>0.9</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>0.579</td>
<td>0.368</td>
<td>0.632</td>
<td>0.526</td>
<td>0.053</td>
<td>0.263</td>
<td>0.632</td>
<td>0.737</td>
<td>0.158</td>
</tr>
</tbody>
</table>

**Rate of people who answered the questions correctly: Distance Vector Routing Algorithm - Viz group**

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of participants</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>10</td>
<td>0.4</td>
<td>0.6</td>
<td>0.5</td>
<td>0.4</td>
<td>0.5</td>
<td>0.5</td>
<td>0.4</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td>Second</td>
<td>10</td>
<td>0.5</td>
<td>0.2</td>
<td>0.3</td>
<td>0.5</td>
<td>0.2</td>
<td>0.4</td>
<td>0.4</td>
<td>0.4</td>
<td>0.5</td>
</tr>
<tr>
<td>Total</td>
<td>20</td>
<td>0.45</td>
<td>0.4</td>
<td>0.4</td>
<td>0.45</td>
<td>0.35</td>
<td>0.45</td>
<td>0.4</td>
<td>0.4</td>
<td>0.6</td>
</tr>
</tbody>
</table>

**Rate of people who answered the questions correctly: Distance Vector Routing Algorithm - Non-Viz group**

<table>
<thead>
<tr>
<th>Group</th>
<th>Num. of participants</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>8</td>
<td>0.375</td>
<td>0.5</td>
<td>0.5</td>
<td>0.625</td>
<td>0.875</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.333</td>
</tr>
<tr>
<td>Second</td>
<td>10</td>
<td>0.2</td>
<td>0.4</td>
<td>0.3</td>
<td>0.3</td>
<td>0.4</td>
<td>0.7</td>
<td>0.4</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>Total</td>
<td>18</td>
<td>0.278</td>
<td>0.444</td>
<td>0.389</td>
<td>0.444</td>
<td>0.611</td>
<td>0.5</td>
<td>0.333</td>
<td>0.333</td>
<td>0.5</td>
</tr>
</tbody>
</table>
APPENDIX K

Questions of the Dijkstra-Scholten Distributed Diffusing Termination Detection Algorithm for the Second Study

1. Which of the following choices best describes the condition under which a node will send an acknowledgement to its parent node?
   A. The node's count goes to zero;
   B. Timeout;
   C. The node's status is idle;
   D. The node's count goes to zero and timeout;
   E. The node's count goes to zero and status is idle;
   F. None of the above.

2. Observe the program execution of "Question 2" (choose from the drop down list). At the end of it, please tell which nodes are the participants in the task for which the termination detection is being performed.
   A. A, B, C;
   B. D, E;
   C. A, B, C, D, F;
   D. B, C, D, F;
   E. A, B, C, F.

3. Given below is a list of all the nodes in the network and their parents and corresponding count values. Please tell which node is the root.
   Node A: count = 0, parent = non-null;
   Node B: count = 3, parent = non-null;
   Node C: count = 0, parent = non-null;
   Node D: count = 1, parent = non-null;
   Node E: count = 0, parent = null;
   Node F: count = 2, parent = null;
   Node G: count = 0, parent = non-null;
   Node H: count = 0, parent = non-null.

   A. E;
   B. F;
   C. E and F;
   D. None of the above;
   E. The information is insufficient.

4. Given below is a snapshot taken from the visualization and a part of the execution output. Is it possible to have such a scenario according to the Termination Detection Algorithm?
A. Yes;
B. No;
C. The information is insufficient.

5. A network has five nodes. What would be the maximum count value of a node if all nodes could participate in the task for which the termination detection is being performed?

A. 5;
B. 4;
C. 1;
D. None of the above.

6. Observe the program execution of "Question 6 Input" (choose from the drop down list). At the end of the run displayed, please tell which of the following values could NOT be Node D's count value.

A. 1;
B. 2;
C. None of the above;
D. The information is insufficient.

7. Given below is a list of all the nodes in the network and their parents and corresponding count values. Assuming there are no messages in transit and no further application messages will be sent, how many acknowledgments need to be sent until the root detects the termination of the task?

Node A: count = 0, parent = F;
Node B: count = 3, parent = F;
Node C: count = 0, parent = D;
Node D: count = 1, parent = B;
Node E: count = 0, parent = null;
Node F: count = 2, parent = null;
Node G: count = 0, parent = B;
Node H: count = 0, parent = B.

A. 3;
B. 4;
C. 6;
D. None of the above;
E. The information is insufficient.

8. Observe the program execution of "Question 8 Input" (choose from the drop down list). At the end of it, Node B is waiting on which nodes?
9. The "parent" relationship forms a virtual tree among the nodes involved in the task for which the Termination Detection Algorithm is being performed. Can the virtual tree grow and shrink repeatedly?

A. Yes;
B. No;
C. The information is insufficient.

10. One interesting modification to the algorithm would allow multiple tasks to be performed in the network at the same time, with the result that multiple virtual trees exist simultaneously in the network. Each node would be able to participate in more than one task. Termination detection is considered to be correct if a root detects the termination after the task initiated by that root is terminated, when no node is working on that task any more. Note that nodes may still be working on other tasks. What modification(s) to the termination detection algorithm is (are) required to make this multiple-task termination detection work?

A. Having multiple parents in each node;
B. Having multiple counts in each node;
C. Having multiple parents and count in each node;
D. None of the above.
APPENDIX L

Some Screenshots from the Testing Environment for the Second Study

Login Page

Fourth Question Page
Thank you for taking time to participate in our usability study. You answered 3 questions correctly out of 10 questions. The correct rate is 30%.

Before exiting the system, please answer a few simple questions below, which will be helpful to the data analysis.

Did you know this algorithm before?
- Yes, I knew this algorithm.
- No, I didn’t know this algorithm.

Did you have enough time to answer all the questions?
- Yes, I had enough time to finish all the questions.
- No, I ran out of time.

We would like you to give us any comments or suggestions about our visualization or the usability study. Please enter in the area below and click the “Submit” button when you are done.

Result page where comment can be entered
Thank you for taking time to participate in our usability study. Your evaluation will be very helpful to us.
APPENDIX M

Dijkstra-Scholten Distributed Diffusing Termination Detection Algorithm Data

from the Second Study

The number of question answered correctly (totally 10 questions, 6 general questions, and 4 questions with specific input), time spent on answering all the questions, and the total evaluation time

<table>
<thead>
<tr>
<th>Subject</th>
<th># Correct Answers</th>
<th>G_Q</th>
<th>SP_Q</th>
<th>Q_Time (min)</th>
<th>TotalTime (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz_A</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>25.12</td>
<td>25.85</td>
</tr>
<tr>
<td>Viz_B</td>
<td>9</td>
<td>4</td>
<td>5</td>
<td>17.28</td>
<td>34.26</td>
</tr>
<tr>
<td>Viz_C</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>26.22</td>
<td>62.01</td>
</tr>
<tr>
<td>Viz_D</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>18.73</td>
<td>31.57</td>
</tr>
<tr>
<td>Viz_E</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>22.91</td>
<td>58.61</td>
</tr>
<tr>
<td>Viz_F</td>
<td>10</td>
<td>4</td>
<td>6</td>
<td>41.29</td>
<td>58.07</td>
</tr>
<tr>
<td>Viz_G</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>14.85</td>
<td>31.17</td>
</tr>
<tr>
<td>Viz_H</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>45.26</td>
<td>72.48</td>
</tr>
<tr>
<td>Viz_I</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>19.98</td>
<td>43.17</td>
</tr>
<tr>
<td>Viz_J</td>
<td>8</td>
<td>3</td>
<td>5</td>
<td>18.4</td>
<td>31.53</td>
</tr>
<tr>
<td>Non-Viz_A</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>12.83</td>
<td>14.42</td>
</tr>
<tr>
<td>Non-Viz_B</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>48.2</td>
<td>68.99</td>
</tr>
<tr>
<td>Non-Viz_C</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>40.02</td>
<td>110.58</td>
</tr>
<tr>
<td>Non-Viz_D</td>
<td>7</td>
<td>2</td>
<td>5</td>
<td>8.12</td>
<td>25.23</td>
</tr>
<tr>
<td>Non-Viz_E</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>74.39</td>
<td>94.02</td>
</tr>
<tr>
<td>Non-Viz_F</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>7.56</td>
<td>32.58</td>
</tr>
<tr>
<td>Non-Viz_G</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>35.63</td>
<td>50.03</td>
</tr>
<tr>
<td>Non-Viz_H</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>37.71</td>
<td>39.44</td>
</tr>
<tr>
<td>Non-Viz_I</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>7.46</td>
<td>14.18</td>
</tr>
<tr>
<td>Non-Viz_J</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>11.46</td>
<td>12.05</td>
</tr>
</tbody>
</table>