NON-VISUAL INTERACTION WITH GRAPHS.

A thesis submitted to the University of Manchester for the degree of Doctor of Philosophy in the Faculty of Engineering and Physical Sciences

2008

By
Andrew Jeffrey Brown
School of Computer Science
Contents

Abstract 12

Declaration 13

Copyright 14

Acknowledgements 15

1 Introduction 16

1.1 Information Accessibility ................................. 16
1.2 Graph Accessibility ......................................... 19
1.3 Research Questions ......................................... 22
1.4 Thesis Outline ................................................ 24

2 Literature Survey 26

2.1 The Nature and Uses of Graphs ............................ 26
2.2 Advantages of Diagrammatic Presentation ................. 28
2.3 Presenting Information Aurally ............................. 32
2.4 Models for Graph Browsing ................................. 35
2.4.1 Mental Models ............................................. 36
2.4.2 Information Retrieval ..................................... 39
2.4.3 Spatial Exploration ....................................... 41
2.4.3.1 Mental representations of environments ........... 44
2.4.3.2 Application to non-visual graph browsing ........ 45
2.5 Tools for Audio Presentation of Graphs .................. 47
2.5.1 The Tools .................................................. 47
2.5.2 Comparison of Theory with Practice ................... 50
2.6 Summary ...................................................... 51
3 Formative Experiments

3.1 Kekulé: Browsing Molecules

3.1.1 Data Structure

3.1.2 Browsing

3.2 Evaluation

3.2.1 Method

3.2.2 Results

3.2.2.1 TLX Scores

3.2.2.2 Cooperative Evaluation

3.2.2.3 Logging

3.2.3 Discussion

3.3 Description Experiments

3.3.1 Method

3.3.2 Results

3.3.2.1 Vocabulary

3.3.2.2 Annotation

3.3.2.3 Spatial Descriptions

3.3.2.4 Description Sequence

3.3.2.5 Speed and Memory

3.3.2.6 Control of the conversation

3.3.2.7 Summarisation

3.3.3 Discussion

3.3.4 Conclusions

3.4 Summary

4 Annotation

4.1 Annotation in Interfaces

4.2 Annotating Graphs

4.2.1 Chunking

4.2.2 Home Node

4.2.3 Relationship

4.2.4 User Notes

4.2.5 Location

4.2.6 Direction

4.2.7 Deedle

4.2.8 Visit Histories
5 Graph Summarisation

5.1 Literature Review .......................... 102
5.2 Complexities ............................. 104
5.3 Descriptive Summaries ...................... 106
5.4 Comparative Summaries ...................... 108
  5.4.1 Theory .................................. 108
  5.4.2 Practicalities ......................... 110
  5.4.3 Experiment ............................ 111
5.5 Audio Summaries .......................... 112
  5.5.1 Requirements ......................... 115
  5.5.2 The Algorithms ....................... 116
  5.5.3 Evaluation Method ..................... 117
    5.5.3.1 Phases 1 and 2 ................... 118
    5.5.3.2 Phase 3 .......................... 120
    5.5.3.3 Phase 4 .......................... 121
  5.5.4 Results ............................... 122
    5.5.4.1 Phases 1 and 2 ................... 122
    5.5.4.2 Phase 3 .......................... 125
    5.5.4.3 Phase 4 .......................... 126
  5.5.5 Discussion ........................... 128
5.6 Conclusions ............................. 130

6 Node Identification ........................ 132

6.1 Literature Review ........................ 132
  6.1.1 Node Differentiation .................. 133
  6.1.2 Landmarks ............................ 134
6.2 Labelling ................................. 136
8.2.2 Tasks ........................................... 182
8.2.3 Analysis ................................. 184
8.2.4 Participants .............................. 185

8.3 Results ........................................... 186
  8.3.1 Success Rates ......................... 186
  8.3.2 TLX scores .............................. 187
  8.3.3 Questionnaires ......................... 190
  8.3.4 Cooperative Evaluation ............ 191
      8.3.4.1 Logic Circuits ................... 191
      8.3.4.2 Family Trees ..................... 194
  8.3.5 Software Logging ....................... 195
  8.3.6 Discussion .................. 196
      8.3.6.1 Summary .......................... 196
      8.3.6.2 Orientation ....................... 197
      8.3.6.3 Relating ......................... 200
      8.3.6.4 User Task ......................... 200
      8.3.6.5 Overall ....................... 201

8.4 Summary and Conclusions ............. 202

9 Discussion ..................................... 204
  9.1 Overview of this Thesis ............... 204
  9.2 Significance of Major Results ....... 205
  9.3 Outstanding Issues ...................... 209
  9.4 Future Work ............................. 210

Bibliography ...................................... 213

A Kekulé Evaluation Details ............. 224
  A.1 Introduction for Participants ....... 224
      A.1.1 Introduction ....................... 224
      A.1.2 The Software ....................... 224
          A.1.2.1 Basic Mode ................... 225
          A.1.2.2 Advanced mode ............... 226
  A.2 Software Instructions ................. 227
  A.3 Questions .................................. 228
      A.3.1 Simple ............................. 230
D.3 Instructions for Part 2 ........................................ 257
D.4 Graphs used in Part 2 ........................................ 258
D.5 Results .......................................................... 258

E RDF Schema ......................................................... 260

F Evaluation Details .................................................. 263
F.1 Introduction ..................................................... 263
F.2 Questionnaire ................................................... 264
F.3 Software Instructions .......................................... 266
F.4 Questions ......................................................... 267
   F.4.1 Logic Circuits ........................................... 267
   F.4.2 Family trees ............................................. 267
F.5 TLX scores ....................................................... 270
## List of Tables

3.1 TLX score differences. ........................................ 63  
3.2 Complexities for the abstract graphs. ......................... 71  
4.1 A taxonomy of annotations. ................................... 99  
6.1 Results of the evaluation. .................................... 150  
7.1 List summary example for family trees. ....................... 166  
7.2 List summary example for logic circuits. ..................... 166  
7.3 The user-interface lists for the running example. ............ 169  
7.4 The user-interface lists for the running example. ............ 169  
7.5 The user-interface lists for the running example. ............ 177  
7.6 The user-interface lists for the running example. ............ 178  
7.7 The menu structure of the software. ......................... 179  
8.1 Experimental design. .......................................... 182  
8.2 Question success rates. ....................................... 187  
8.3 TLX score differences. ....................................... 188  
A.1 Commands available in the basic mode ....................... 228  
A.2 Commands available in the advanced mode .................... 229  
A.3 Raw TLX scores. ............................................. 233  
B.1 Times taken for graph descriptions. ......................... 236  
D.1 First choices for phase 2. ................................... 259  
F.1 Raw TLX scores. ............................................. 272
List of Figures

1.1 A classic example of a graph: the London Underground map. . . . 20
2.1 Diagram showing the structure of ethanoic acid. . . . . . . . . . . 27
2.2 Recognition of features of a graph. . . . . . . . . . . . . . . . . 29
2.3 Diagram showing the structure of benzene. . . . . . . . . . . . . 30
2.4 Diagram representing the process of model building and manipu-

3.1 The structure of the phenylalanine molecule. . . . . . . . . . . 55
3.2 Hierarchical organisation of groups in Phenylalanine. . . . . . 57
3.3 Examples of question molecules. . . . . . . . . . . . . . . . . . 61
3.4 The seven abstract graphs used for the experiment. . . . . . . . 81
3.5 The semantically rich graph used for the experiment. . . . . . . 82
3.6 Scan of the sketch drawn by pair 4 of graph 3.4(g). . . . . . . . 82
3.7 The original graph (g) and two transformations of the same. . . 83
4.1 Examples of annotation in a text book. . . . . . . . . . . . . . . . 86
4.2 Examples of annotation in a family tree. . . . . . . . . . . . . . . 96
5.1 The benefits of a brief summary. . . . . . . . . . . . . . . . . . 102
5.2 Pairs of graphs selected as the ‘most similar’. . . . . . . . . . . 112
5.3 An example of application of the summary-generating algorithm. . 116
5.4 The graphs used in phase one. . . . . . . . . . . . . . . . . . . . 120
5.5 The graphs used in phase two. . . . . . . . . . . . . . . . . . . . 120
5.6 The graphs used in phase three. . . . . . . . . . . . . . . . . . . 121
5.7 The scores of individual participants for all phases. . . . . . . . . 123
5.8 Results for phases 1 and 2. . . . . . . . . . . . . . . . . . . . . . 124
5.9 Double-ring graphs were difficult to differentiate. . . . . . . . 124
5.10 Results for phase 3. . . . . . . . . . . . . . . . . . . . . . . . . . 125
5.11 Results for phase 4. ............................................ 126
5.12 The graphs presented in phase 4. ........................ 127
6.1 Graph showing nodes numbered using the alphanumeric system . 141
6.2 Graph showing nodes numbered using the radial system .... 142
6.3 The graphs used in phase 1. .............................. 145
6.4 The graphs used in phase 2. .............................. 146
6.4 The graphs used in phase 2. .............................. 147
7.1 The place of RDF in the Semantic Web. ................. 157
7.2 Part of an RDF graph for a molecular structure diagram. . 162
7.3 The architecture of this system. ........................ 175

A.1 Schematic representation of the structure of ethanol. ...... 225
A.2 The hierarchical nature of the structure of carboxylic acid. 227
A.3 The first simple molecule (Molecule 1). .................... 230
A.4 The second simple molecule (Molecule 2). ............... 230
A.5 The first complex molecule (Molecule 3). ............... 230
A.6 The second complex molecule (Molecule 4). ............ 231
A.7 The third complex molecule (Molecule 5). .............. 232
A.8 The fourth complex molecule (Molecule 6). ............ 232

C.1 The audio glance algorithm. .............................. 241
C.2 Diagram of a graph containing a ring ................... 242
C.3 Diagram of a graph with two rings. .................... 242
C.4 The graphs displayed in phase one of the experiment. .... 243
C.5 The graphs displayed in phase two of the experiment. .... 244
C.6 Diagram of a graph with two nodes highlighted. ....... 247
C.7 Distractor graphs for phase three. ...................... 248
C.8 Screen shot of phase four. ............................. 250

F.1 The first logic diagram. ................................. 268
F.2 The second logic diagram. ............................. 269
F.3 The first family tree. ................................. 270
F.4 The second family tree. ............................. 271
Abstract

This thesis investigates how the advantages of graph-based diagrams can be recreated such that visually impaired people may gain some of the benefits that this form of presentation offers sighted readers. The hypothesis is that annotation offers a means for achieving this, as well as for minimising some of the disadvantages of aural presentation and for reducing disorientation.

A literature survey is combined with experiments to understand the ways in which graphs benefit sighted people and the difficulties encountered when exploring graphs non-visualy. Different forms of annotation that may address these problems are explored, classified and evaluated, including notes designed to summarise and to aid node differentiation. A user-evaluation is designed and applied to demonstrate that a graph which is annotated requires less mental effort to explore than one which is not, and that tasks can be achieved more effectively and more efficiently by using annotation to replace the benefits of visual graphs.

While providing non-visual access is relatively simple for linear information, it becomes increasingly difficult as the structure of the information becomes more non-linear. For several reasons, however, non-linear forms of presentation, such as node-arc graphs, are a powerful and elegant means of visual information presentation. These graphs are commonplace, particularly in education, and it is important that everyone can access the information they convey.

This research demonstrates that by understanding how diagrams benefit sighted people it is possible to improve accessibility for non-sighted people. Annotation offers a powerful and flexible technique for transferring these benefits, as well as for reducing disorientation while moving around the graph and for tackling some of the inherent disadvantages of using sound. The findings of this thesis offer insight into how accessibility can progress beyond linearly organised information, and beyond even the non-linearity of equations and tables, to enable non-visual access to the rich world of graphs and diagrams.
Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institution of learning.
Copyright

Copyright in text of this thesis rests with the Author. Copies (by any process) either in full, or of extracts, may be made only in accordance with instructions given by the Author and lodged in the John Rylands University Library of Manchester. Details may be obtained from the Librarian. This page must form part of any such copies made. Further copies (by any process) of copies made in accordance with such instructions may not be made without the permission (in writing) of the Author.

The ownership of any intellectual property rights which may be described in this thesis is vested in the University of Manchester, subject to any prior agreement to the contrary, and may not be made available for use by third parties without the written permission of the University, which will prescribe the terms and conditions of any such agreement.

Further information on the conditions under which disclosures and exploitation may take place is available from the Head of the School of Computer Science.
Acknowledgements

I would like to thank my wife, Jean, for her support and assistance and my son, Christopher, for putting up with me spending too much time on my computer.

I would also like to thank my supervisors, Robert Stevens and Steve Pettifer, for their sage advice.
Chapter 1

Introduction

1.1 Information Accessibility

This thesis builds on the body of accessibility research by examining the problem of accessibility of graphs. It collates literature from cognitive science, psychology, information management and computer science to identify the benefits of diagrammatic presentation and the causes of difficulties encountered when browsing graphs non-Visually. It then proposes and evaluates techniques that should ease these difficulties.

The amount of information, and ease of access to it, has increased dramatically over the last century, particularly over the last ten or twenty years, at least in developed countries. Even if the claim that ‘a weekday edition of The New York Times contains more information than the average person was likely to come across in a lifetime in seventeenth-century England’ [82] is a little dubious, it is almost certainly true to say that the seventeenth century gentleman did not have access to the same amount of published information as he would today. Nowadays anyone with access to the World Wide Web can retrieve huge quantities of information, from weather forecasts, to bus timetables, to gossip, to academic papers, and so on. And if the World Wide Web is not sufficient, a great deal more information is published on paper each year, including books and newspapers.

If visually impaired people are to take a full role in society, they must have

---

1 57% of households in Great Britain could access the Internet between January and April 2006, up by 26% since 2002 [67]

2 A person is considered blind if he or she has a visual acuity (as measured for example using a Snellen letter chart) of less than 0.05; they are considered to have low vision (i.e., be partially sighted) if their visual acuity is greater than 0.05 but less than 0.3 [66]. Visually impaired is a
access to information. This is neither an insubstantial problem, nor applicable only to a tiny number of people. 370,000 people in the UK are registered as blind or partially sighted [65], and The Royal National Institute for the Blind estimates that there are up to a further 20% that are eligible to be so [79]. Although the majority of these are over 65 years old, a significant proportion are of working age [65]. The American Foundation for the Blind estimates that 10 million people in the United States are blind or visually impaired [32].

Although computer use amongst the blind is below the levels of the population as a whole, a ‘minimal estimate’ is that there are approximately 1.5 million visually impaired computer users in the US, including some 102,000 regular users that are severely visually impaired [32]. It is also reasonable to suggest that as current computer users age, and the numbers of older computer users increases, the number of blind or visually impaired computer users will also increase substantially. Access to computer-presented information for these people is an important challenge.

While access to information is important for the general population, it is probably more important for those in education. It is all very well enabling visually impaired adults to access vast quantities of information, but this is of little benefit if they have not received a suitable education. Education, by its very definition, requires information to be presented and understood; although much of this information may be presented verbally or physically, educating relies significantly on published material, both in electronic and in book form. Clearly accessing this information is critical to a good education. There are nearly 4,000 people in England of school age (under 18) on the register of blind people, plus approximately 4,800 registered as partially sighted [65].

An added impetus for enabling accessibility is legislation — governments are increasingly aware of the need to integrate visually impaired people into society, and of the importance of information access for this goal. As a result, many countries have legislated to ensure rights for visually impaired people (and persons with other disabilities) in education, the workplace and elsewhere. In 1998 the United States Congress amended the Rehabilitation Act to improve accessibility to Federal information technology. The Section 508 website\(^3\) states the following in its introduction to the law:

\(^3\)http://www.section508.gov/

term that covers people who are either blind or partially sighted.
1.1. INFORMATION ACCESSIBILITY

“The law applies to all Federal agencies when they develop, procure, maintain, or use electronic and information technology. Under Section 508 (29 U.S.C. § 794d), agencies must give disabled employees and members of the public access to information that is comparable to the access available to others.”

There are similar requirements in the UK, largely resulting from the Disability Discrimination Act 1995 [42] and the Special Educational Needs and Disability Act of 2001 [43].

So much for the demands for accessibility and the benefits it can bring, but information comes in a variety of forms, both in the way in which it is organised and the medium with which it is presented. The discussion so far has focused slightly on the demands for computerised information to be made accessible, as this is increasingly the most important medium. Even if only digital information is considered, however, there are still numerous ways in which the raw information (say at the level of letters, numbers, symbols, lines, etc.) can be organised. Electronic documents regularly feature text, both in simple linear paragraphs, or non-linearly structured using hypertext, tables, equations, photographs, or even sound and video. Making each of these individual forms of information accessible is a significant challenge in itself, not even considering the problems of enabling seamless access to multimedia.

The standard means of computer access for a blind user in the UK is the screen reader: software that reads text on the screen (including simple document text, button labels, menus and other software controls) and presents it as speech via a speech synthesiser (hardware or software — nowadays this is often incorporated into the screen-reading software). While giving an acceptable level of control over the computer, screen readers are far from perfect. One particular problem is that they rely on the operating system and software to pass it the necessary information. A further problem is that navigating non-linear information is difficult.

The latter problem has been tackled mainly in the field of hypertext accessibility [72, 113], with the huge prize of efficient Web access as the main driver. These systems are largely text-based, in that the raw information is simple text, but the chunks of text are organised in a more complex way than a simple linear narrative. Giving visually impaired users an experience as rich and efficient as sighted users is a difficult task that, arguably, has not yet been achieved. In other
domains, tools for accessing mathematic equations [96] have been researched and
developed; here the non-linearity is present at a much smaller granularity making
this probably the most closely related to graphs. Tables [21, 115] and numerical
graphs [116, 56] are also being investigated. In fact, most types of visual information
have been the subject of research into making them accessible to the blind,
including graphs [73, 41] and photographs [81].

1.2 Graph Accessibility

An important class of diagram, the graph is, mathematically speaking, a set of
nodes connected by arcs: in layman’s terms ‘things connected by lines’. This type
of diagram is particularly suitable for depicting relationships, and is very common.
For example, consider railway diagrams such as the London Underground map
(reproduced in figure 1.1), flowcharts, hierarchies, trees (e.g., family trees) —
all of these might be encountered in everyday life. In technical and educational
environments graphs are even more common, ubiquitous even — it would be
difficult to find a science text book not containing a graph. This research is
limited to exploring accessibility of information that is (or can be) presented as
a graph.

We are concerned only with diagrams that are graphs in the strict mathematical
sense, that is graphs where the positioning of the nodes is for presentation
and does not affect the semantics. It should be noted that this is not quite as
clear a distinction as it might at first appear. Take, for example, a diagram repre-
senting a rail network with nodes representing stations and arcs the railway lines:
this graph could be laid out as a map, in which case the location of the nodes is
meaningful (the distances between stations are represented), but if the intention
is simply to illustrate the routes between stations the layout is not strictly im-
portant. Here the intended use of the graph influences whether it is considered a
graph or a map. This research is not considering how to present diagrams where
spatial layout is important.

Presentation of diagrams to blind people obviously requires the information
to be presented to the other senses of the user; discounting the senses of
taste and smell leaves touch and sound. Devices are available to present in-
formation through haptic interfaces (including both tactile and force-feedback
1.2. GRAPH ACCESSIBILITY

Figure 1.1: A classic example of a graph. In this schematic representation of the London Underground, the nodes represent stations and the arcs represent the railway lines; the arrangement is such that it only shows how to get from one station to another — the distances between stations is meaningless.
1.2. GRAPH ACCESSIBILITY

interfaces), and these have been utilised by many research groups (for example [49, 12, 80, 108, 117]). The equipment required for these interfaces is, however, specialist, and often expensive. Even braille and raised-paper diagrams are not ideal, requiring specialist hardware to generate them, and being unable to respond quickly to changes in the data. Braille also needs to be learnt, and is by no means universally known amongst blind people (the RNIB state that only 4% of blind and partially sighted people in the UK use braille [78]). For these reasons, this thesis concentrates on keyboard input and audio output; it is unusual nowadays to use a computer without keyboard or speakers. Many of the techniques developed for exploring graphs through such an interface may well be applied to other, more specialist interfaces.

What, then, are the aims of this research, and what can it contribute? There have already been several research projects that have studied non-visual accessibility for graphs (for example [8, 12, 49]), and more that deal with a specific type of graph, such as the hierarchy [16, 15, 89]. While Chapter 2 discusses these in more detail, it is worth noting here that while these tools meet with some success there is undoubtedly room for improvement. It is also notable that the design of these tools often appears to be largely based on intuition, with little of the underlying cognitive science cited. A rigorous approach, combining theory with experiments should yield valuable insight into the problem, and suggest suitable approaches to solving it.

The general aims of this work are to:

- Understand the cognitive science behind graph reading and the benefits of diagrams in order to identify what is necessary in a non-visual graph browsing tool.

- Investigate techniques for replacing or replicating these using the keyboard / audio interface.

- Consider solutions applicable to graphs generally, rather than one particular domain or application.

- Evaluate the techniques, both independently (where possible) and as part of a whole system.

More specifically, this thesis is an investigation of the following hypothesis:
Annotations can be designed to replace certain of the benefits imparted by visual presentation of graph-based diagrams, including making implicit information explicit, grouping related items, interactivity and acting as an external memory, and to reduce disorientation while moving around the graph. A graph which is annotated in such a way requires less mental effort for a visually-impaired user to explore than one which is not. Tasks can be achieved more effectively, efficiently and with more satisfaction through use of annotation to replace features of a visual presentation.

This thesis investigates this hypothesis by examining the benefits of diagrammatic representations and the problems associated with non-visual browsing, proposing annotations for the graph that might replicate the benefits and alleviate the problems, then evaluating whether they do, in fact, benefit users.

1.3 Research Questions

We now pose the research questions that must be answered in order to support or refute the hypothesis — this section states the aims of the research as specific questions. Although the questions are presented here, at the start of the thesis, there is, in fact, a certain interdependence since some of the questions follow from answers, or at least hypothetical answers, to others. Implicit in these questions are the potential contributions of this thesis: each answer to a research question would represent a contribution to the current state of accessibility research.

Q1. What is difficult when browsing graphs non-visually?

It is reasonable to state that solving tasks using diagrams is a complex procedure, even for simple tasks on simple graphs. In order to propose techniques for easing a complex problem, it is not sufficient just to state that it is difficult: it is critical to identify the precise reasons why it is difficult. The first question is thus: what are the major difficulties posed to readers when reading graphs non-visually? As noted above, many attempts at improving diagram accessibility have failed to approach the subject rigorously and proposed solutions have been developed from what seemed to the researcher a ‘good idea’. In this thesis, two more scientific approaches are used: firstly a survey of cognitive science literature...
explores problems one would expect to encounter when attempting to explore graphs non-visually; secondly the combination of a review of the literature on previous similar tools and pilot studies identifies some of the problems that have been observed when people have tried these tasks.

Q2. What annotations can be added to a graph?

Having identified some of the specific problems associated with non-visual graph browsing, and considering our hypothesis that annotation can help alleviate these problems, the second research question arises naturally: With what information can graphs be annotated so that non-visual readers can benefit from some of the advantages of presenting information as a graph? This question needs to be answered with reference to the particular difficulties highlighted in the literature review and pilot studies.

Q3. Can Earcons\textsuperscript{4} be used to convey the gist of a graph?

Comprehension is thought to be improved if information has been previewed, so this thesis examines how graphs may be annotated with summary information. In particular it addresses the specific question: can non-speech sounds (in particular Earcons) be used to present the gist of a graph — can they convey an impression of its size and structure?

Q4. How can nodes be differentiated?

Another particular problem with non-visual browsing is that of node identification and differentiation. A system for distinguishing nodes with identical names is necessary when 2D indexing can no longer be used. This thesis explores the requirements and methods for differentiating nodes, particularly the question of how to number the nodes in a graph.

Q5. How can annotated graphs be represented?

Any tool that is to use annotation to help visually impaired users explore graphs must represent both the graph and the annotations. Although the question of

\textsuperscript{4}Earcons are abstract, structured, non-speech sounds [11] ; the audio equivalent of visual icons.
1.4 Thesis Outline

The remainder of this thesis is organised as follows:

**Chapter 2** Examines the benefits of presenting information diagrammatically and looks at the issues involved with presenting aurally. It also discusses how relevant and useful the concepts of real world travel can be as a metaphor for understanding graphs non-Visually, and concludes with an overview of previous research in this field.

**Chapter 3** Describes the evaluation of a previously developed tool for non-visual graph exploration, and experiments designed to identify vocabulary and strategies used when describing graphs.

**Chapter 4** Introduces the concept of annotation, looking at how it has been used in different interfaces, then how it may be applied to non-visual exploration of graphs. It concludes with a classification of annotations.

**Chapter 5** Is an in-depth examination of one class of annotation – summaries – concentrating on the development and evaluation of audio glances at graphs.

**Chapter 6** Discusses another class of annotation for aiding orientation by landmark creation and assisting users to differentiate between nodes.

**Chapter 7** Explores the requirements for representing annotated graphs, and describes the use of semantic web technologies to do this in the implementation of a tool for non-visual graph exploration. The chapter also details
the particular annotations used in the evaluation, and introduces the user-
interface with a running example of the software in use.

**Chapter 8** Explains how this tool was used to test our hypothesis, then presents
and discusses the results of this evaluation.

**Chapter 9** Reviews the work presented to ascertain the extent to which the
research questions have been answered. The significance of the work is
discussed, as are the outstanding issues and some avenues for further work.
Chapter 2

Literature Survey

This chapter starts to answer the first research question: what is difficult about browsing graphs non-visualy. It reviews several different fields of the literature that are pertinent to non-visual graph exploration. The aim is to develop an understanding of the theoretical issues that will need to resolved if an effective tool is to be produced. Firstly it is necessary to understand why diagrams are used when presenting information to sighted readers: what benefits, if any, do diagrams have over other representations? Consideration must also be given to the limitations imposed by using sound as the basis for presentation; how does it differ from vision; what are its disadvantages; and what are its advantages? It is also instructive to consider the way in which people extract information from graphs, i.e., how do they interact with the information? Two models for this interaction — information retrieval and spatial exploration — are compared, with the intention of identifying problem areas and potential solutions. For example, taking the spatial model, there is literature on development of cognitive maps; can understanding this process shed any light on problems that might occur when exploring a graph? Finally, a selection of previous work in this field is reviewed to see what difficulties they identified and attempted to minimise.

2.1 The Nature and Uses of Graphs

As an example of a graph, consider Figure 2.1, which shows the structure of the molecule ethanoic acid.

In this type of graph the nodes represent atoms and the edges the bonds between them. Edges have only one attribute — order, i.e., whether the bond is
2.1. THE NATURE AND USES OF GRAPHS

Figure 2.1: Diagram showing the structure of ethanoic acid.

a single bond, double bond, etc. The nodes have different attributes, including
the type of atom (unlabelled ones are carbon) and its electric charge. An inter-
esting feature of this graph is that hydrogen atoms connected to carbons are not
explicit; to benefit from a simpler diagram the user is expected to have sufficient
chemical knowledge to deduce their locations. This diagram also gives an exam-
ple of drawing conventions: the molecule is a three-dimensional object but, to
be represented on paper, needs to be mapped onto two dimensions. In the case
of molecules the atoms and bonds are normally laid out on a roughly hexagonal
grid, and further chemical knowledge is necessary if the three-dimensional struc-
ture is to be known. Indeed, representing the relative 3D orientation of bonds
often requires variation in the diagram notation. Note, however, that the tasks
these representations are used for often do not require this level of understanding.

The features of these diagrams illustrate all three of the factors highlighted
by Peebles [69] as involved when reasoning with diagrams. He described dia-
grammatic reasoning as behaviour involving interaction between the cognitive
and perceptual skills of the reasoner, the graphical properties of the external
representation being used, and the specific requirements of the task undertaken.
Here we see first that the user must be trained in the use of these diagrams to
understand that an unlabelled node is carbon, and the meaning of 3D bond no-
tations, and must have sufficient chemical knowledge to determine the locations
of hydrogens. Peebles’s second factor is that the reasoning he is able to perform
depends upon the diagram; in this example the reader is generally unable to make
deductions about the 3D structure unless 3D bonds information is given. The
third factor, the task, will be discussed later.

In order to understand how to present diagrams non-visually it is important
to know the range of reasons why they may have been created in the first place.
Finding the intention of a diagram is not always a simple task, but the following
are among the possibilities:

- Many diagrams are simply used to illustrate the relationships between the entities in a concise way; it was easier for the author to draw a diagram than to create a well-written piece of prose describing the same system.

- In other instances a diagram may be used simply to break up a section of text that might otherwise be intimidating to read or appear ‘dry’ and boring.

- When appearing amongst a long body of text, a diagram is easy to find and therefore may be used to present some information that the reader will need to refer back to, saving the need to search text.

- If a diagram is used to describe a system where there may be action between nodes (e.g., a chemical plant schematic), it can allow the reader to perform a pseudo-animation, following a text description with a finger tracing a route across the diagram.

In some cases the diagram is an essential part of the communication, while in others may be safely ignored. Clearly this decision must be left to the reader, although providing a means of assessing the diagram could save significant effort.

2.2 Advantages of Diagrammatic Presentation

Assuming diagrams can actually facilitate understanding and reasoning, knowing which aspects are important, and why, should allow designers of non-visual interfaces to develop analogues or replacements.

Larkin & Simon [51] asserted that 2D indexing of the information in diagrams can support extremely useful and efficient computational processes. By examining the computation required for problem solving using sentential (sequential representations, like propositions in a text) and equivalent diagrammatic representations, they concluded that diagrams facilitated problem solving by easing search and recognition.

The first of these conclusions is that localisation of related nodes in diagrammatic representations reduces the need for searching, and allows computation without generating and matching symbolic labels. That is, the two-dimensional
space can be used to group related nodes much more efficiently than a one-dimensional string of text (or speech). The second conclusion, and in their view the more important one, was that diagrams make recognition considerably easier. They illustrate this with a geometry problem that describes two parallel lines crossed by two transversal lines, which intersect between the parallel lines. With a diagram (figure 2.2) it is immediately apparent that there are two triangles formed, while with the sentential description above, some mental computation is required to make this inference.

![Figure 2.2: Recognition of implicit triangles. Diagram representing the following information: “Two transversals intersect two parallel lines and intersect with each other at a point x between the two parallel lines. One of the transversals bisects the segment of the other that is between the two parallel lines”. From [51].](image)

To give a chemical example of this, it is possible to compare the effort required to deduce the existence of a ring in the pseudo-XML for benzene, below, with the same information in Figure 2.3.

```xml
<molecule id="benzene">
  <atomArray>
    <atom id="a1" element="C" hCount=1/>
    <atom id="a2" element="C" hCount=1/>
    <atom id="a3" element="C" hCount=1/>
    <atom id="a4" element="C" hCount=1/>
    <atom id="a5" element="C" hCount=1/>
    <atom id="a6" element="C" hCount=1/>
  </atomArray>
  <bondArray>
    <bond id="b1" atoms="a1,a2" order=1/>
  </bondArray>
</molecule>
```
2.2. ADVANTAGES OF DIAGRAMMATIC PRESENTATION

Figure 2.3: Diagram showing the structure of benzene, as given in the pseudo-XML.

```xml
<bond id="b2" atoms="a3,a2" order=2/>
<bond id="b3" atoms="a3,a4" order=1/>
<bond id="b4" atoms="a4,a5" order=2/>
<bond id="b5" atoms="a5,a6" order=1/>
<bond id="b6" atoms="a1,a6" order=2/>
</bondArray>
</molecule>

It is clear that these two representations, although informationally equivalent, are not computationally equivalent. One might, of course, argue that XML is a poor choice for sentential presentation, but the example is nevertheless instructive — understanding and applying the benefits that Larkin and Simon demonstrated were afforded by diagrammatic representation would be likely to result in a more accessible graph.

Further research on diagrammatic representations has developed the work of Larkin and Simon to consider the interaction between internal and external representations — distributed (or external) cognition. These are discussed by Scaife and Rogers [83]. For example, Bauer and Johnson-Laird [7] challenged Larkin and Simon’s conclusion that diagrams were not helpful in inference making. They demonstrated that users solving certain types of problem (involving double-disjunctive reasoning, where one must envisage and remember certain alternative states) were significantly quicker when using diagrams than sentential representations. The suggestion was that the diagram acts as an external memory; the problem states and solution are more explicitly represented in the diagram, so reasoners are much less likely to overlook possible configurations. Scaife and Rogers, however, disagree; they argue that the benefits arise because the problem has been re-represented into simpler and different tasks, i.e., the diagram constrains the variety of errors possible when interpreting the problem. Whichever, both agree that the diagram helps.
A further theme highlighted by Scaife and Rogers is that of interactivity: suggesting attempts to maximise computational offloading from the internal to the external representation.

Work on visual perception has indicated that (given certain constraints) it is inevitable that people perceive the whole before the parts. The classic experiments (as described in, for example, A Handbook of Cognitive Psychology [29]) demonstrating this used large letters that were formed from many smaller letters — people perceived the larger letter first. Later experiments refined this by identifying an upper limit for the proportion of the visual field taken by the large letter ($8^\circ$), above which the smaller letters became easier to identify.

Palmer [68] proposed a theoretical model to account for this phenomenon. His model lies between the Gestaltist view that the whole is all, and the opposing view that only the primitive components of a visual scene are perceived. He proposed that the visual form is analysed hierarchically starting with the overall configuration and moving down towards the basic features or elements. The clustering of components to form structural units (which he compared to Miller’s chunks [58], introduced in the following section) occurs selectively, in a way that maximises connections between units which have ‘important’ relationships. In the abstract geometric patterns Palmer was using, importance was determined by spatial proximity; this is clearly a domain-dependent criterion. Palmer performed various experiments that confirmed some predictions of his model. Two of these involved tasks that required manipulation or synthesis of patterns and indicated that low level cognitive processes deal with the information in a manner consistent with his model. If this model accurately reflects the mind, it suggests that process of visual perception might organise diagrams into a hierarchical form. We might speculate that the cognitive processes that deal with diagrams are therefore optimised for this type of data structure.

It is interesting to note that it is also the established view that spatial environments are represented in the mind hierarchically. For example, Stevens and Coupe [94] accounted for distortions in spatial judgements by proposing a hierarchical coding of the information. Other studies have provided evidence supporting this model and this is now the dominant view [18].

These theoretical studies highlight some of the attributes of diagrams that make them useful. The main feature of a diagram is that it facilitates recognition of information; that which would be implicit in some representations often
becomes explicit when presented as a diagram. Diagrams also facilitate searching by using 2D indexing, allowing related nodes to be easily identified. These features should be replicated, if possible, when a diagram is presented non-Visually. Palmer’s model of perception suggests that building the data into a hierarchical structure might allow processes to perform in as similar a way as possible to visual perception.

Before considering how the medium affects how information can be presented, it is worth summarising the main benefits of diagrammatic over sentential representation:

- Recognition makes implicit features explicit.
- Two-dimensional grouping of related items helps search.
- The external diagrammatic representation facilitates external cognition — in particular memory demands are lower and error-making constrained.
- There is evidence of hierarchical representation of both visual perception and spatial environments

It is the aim of this research to investigate how these features can be replicated, or replaced, when the information cannot be presented as a visual diagram. What effect then, does the medium have on this process?

2.3 Presenting Information Aurally

Visually presented information typically reaches the reader as light reflected from the surface containing the diagram (in the case of paper) or transmitted from a screen; either way there is an essentially continuous stream of light from the diagram to the user. The light changes depending on the direction in which the user is looking — variation in the properties of the light over two-dimensions can be used to convey information. Although the focus of that user can only be on one portion of the scene at any one time, the continuous presence of the light, combined with an ability to scan with the eyes (hence the attention), allows any portion of the diagram to be visited at will. Crucially, any part of the diagram may also be revisited in an instant (although note that finding the part of interest may not be trivial — e.g., when scanning text for a word). Aurally presented
information, on the other hand, reaches the ears as sound waves. These can convey meaning by variation over time, and (in a limited way, e.g., stereo) over space. Once the sound has been heard, however, it has gone and cannot be revisited.

It might also be informative to consider the ways in which information is encoded in these modes. In visually presented information, various characteristics of the light may be changed to convey meaning; these include colour (hue), intensity, saturation, position (i.e., location within a 2D or 3D environment), plus variations over time of any of these attributes. With information presented in audio, it is possible to vary pitch, volume and location, although the last of these has less resolution than in vision (and specialist equipment is required to generate more than a simple stereo effect). Again, variation of these attributes over time can also be used, and may be particularly powerful (consider music, for example).

How do these differences affect the way in which information is presented using these media?

The form of the representation will have been influenced by the constraints imposed by its medium. In visual diagrams these are mainly the availability of only two dimensions, but also restrictions on the available area. Presenting the same information through other channels leads to a dilemma; do we present as close a translation as possible of the ‘paper copy’, including any artefacts created by its constraints, or do we try to present the raw information in as simple a manner as possible, given the constraints of the new medium? The choice made has implications for the ability of the reader to communicate with others who have read the same diagram on paper. There is a similar problem with the notations and conventions used when drawing; it is presumably desirable that the reader does not have to learn how to read conventional diagrams in order to read them non-visually, yet a common language is necessary for discussion.

There are other features of a traditional diagram that influence how it is read. The most striking, perhaps, is that all parts of the diagram are seemingly instantly accessible. This allows the user to gain an overview of the diagram at a glance (cf. hierarchical perception, above), gaining an appreciation of the level of complexity and the rough structure without needing to examine detail. If the same information is presented as a straight-forward speech description the reader will only have access to the fine detail, unless an overview is explicitly given.

The ability to move rapidly around the diagram also facilitates distributed
cognition, where understanding the diagram (or problem solving using it) occurs as a cognitive process continuously spread over the user’s partial mental model of the diagram and the user’s perception of the diagram itself; the diagram may be considered as an external memory. Particularly when using paper, this may be developed further with the possibility for annotation [83].

It might be that lack of a permanent external representation fundamentally changes the nature of problem solving. The benefits of having information instantly accessible are clear, particularly if that external memory is writable. As a simple example (from [63]) consider that the time taken to multiply two four-digit numbers with pencil and paper is of the order of 100 times quicker than when done mentally. This arises from the fundamental characteristics of memory, particularly the limited capacity of short-term memory (STM) and the long write time for long-term memory (LTM).

For certain tasks, it might be easier for the reader to build a complete mental representation, before performing the task without significant interaction with the diagram. In their book on human problem solving, Newell and Simon [63] stated explicitly how the problem solving mechanism was dependent on the memory available:

“A problem solving program cannot be specified independently of its external memory any more than it can be specified independently of its internal STM and LTM.”

Being unable to offload cognitive processing to the external representation, as one can do with a visual diagram, puts great strain on the short-term memory. It has been known for years that we are limited to holding only a handful of items in short-term memory, but that if items may be ‘chunked’ together, we can still recall a similar number of chunks [58]. If a diagram contains more than $7 \pm 2$ items of information, it is unlikely that a user will be able to build a complete mental representation of the diagram unless some chunking takes place or LTM utilised.

Are there any advantages that using sound might offer? The ability of humans to recognise temporal variations in sound allows information to be conveyed very quickly — sound is therefore well suited to regular presentation of a particular message. It has also been found that, despite the fact that people attending a message to one ear are unaware of semantic information presented to the other [26],
they are aware of their own name (the ‘cocktail party effect’ [59]), and possibly other words with low thresholds (such as ‘Fire!’) [102]; presenting parallel streams of information might therefore be possible in an audio environment.

In summary, the traditional media for presenting diagrams constrain some aspects of the presentation which may, or may not, need to be replicated for non-visual presentations. All parts of a visual presentation are instantly accessible, providing the reader with overviews, and reducing the load on his or her short-term memory.

2.4 Models for Graph Browsing

In addition to considering the benefits of visual diagrams, and the cognitive processes involved in perceiving and using them, it is necessary to appreciate the processes involved in any mechanisms that might substitute for them for non-visual readers.

This section considers the nature of reasoning and introduces mental models, before comparing two overlapping, but different, ways of describing the process of reading a diagram. First it may be considered as purely a problem of information retrieval - the graph contains information which the reader can discover by combinations of searches. The second approach is to treat the graph as a spatial environment which the reader needs to explore, building a cognitive map of the information, moving to regions of greatest interest and extracting the relevant information. We can distinguish these approaches with the definitions of retrieval and browsing given in Modern Information Retrieval [3]:

**retrieval task** “the task executed by the information system in response to a user request”

**browsing** “interactive task in which the user is more interested in exploring the document collection than in retrieving documents which satisfy a specific information need.”

Although Baeza-Yates and Ribeiro-Neto identify these as distinct tasks, they are not independent, simply different ways of finding information in data space. Indeed many IR systems are described using the spatial analogy.
2.4.1 Mental Models

There is still debate in the cognitive science / psychology research communities about how information is represented and manipulated in the mind during reasoning. While some maintain that humans reason with verbal or propositional representations, the majority view seems to be in favour of an analogue representation, or a mental model. In this theory people represent information by building working models in their mind that may be manipulated and inspected in order to draw inferences (note that the concept of a model is distinct from the idea of representation through imagery; see [45]). The idea of mental models has been around in one form or another for many years [47], but really developed after Ehrlich and Johnson-Laird provided some experimental evidence [28]. Johnson-Laird’s book [46] describes this and other experiments in some detail.

In one such experiment participants were asked to reason about the spatial arrangements of five objects [53]. For example they were given a set of statements such as:

- The spoon is to the left of the knife.
- The plate is to the right of the knife.
- The fork is to the right of the spoon.
- The cup is in front of the knife.

Half of the descriptions were determinate, that is consistent with only one layout, while half were indeterminate, that is consistent with more than one layout. After hearing the description, the participants were shown a diagram and asked if it was consistent with the description. A further, unexpected (to the participants), part of the experiment was a test of their memory: they were given four descriptions (including the original one, an inferable one, e.g., “The fork is to the left of the cup”, and two ‘foils’) and asked to rank them in order of resemblance to the original description. The experiments found that the confusion (‘foil’) descriptions were less likely to be chosen when the original statements were determinate than when they were indeterminate, suggesting that the gist is remembered better. It was also found that the original description was more likely to be ranked ahead of the inferable one for the indeterminate descriptions, suggesting that verbatim detail was remembered better. These findings were accounted for by suggesting that participants formed a model of the determinate descriptions but not the indeterminate ones.
More recent evidence supports the model representation. For example, Morra found that the effect on memory of performing tasks (such as described above) followed the predictions made by the mental model theory [60]. Carreiras and Santamaría [23] examined how fast participants were able to make reasonings, and found that time correlated with the number of models required for solution, not the number of inferential steps required. They also found that this followed for non-spatial descriptions (e.g., “A studied more than B”, “C copied from A”).

Recently, Goodwin and Johnson-Laird have developed the theory of mental models, describing five principles upon which it depends [34]:

1. The structure of the model is iconic as far as possible.\(^1\)

2. The logical consequences of relations emerge from models constructed from the meanings of the relations and from knowledge.

3. Individuals tend to construct only a single, typical model.

4. They spontaneously develop their own strategies for relational reasoning.

5. Regardless of strategy, the difficulty of an inference depends on the process of integration of the information from separate premises, the number of entities that have to be integrated to form a model, and the depth of the relation.

The latter principle is perhaps the most interesting here, as it suggests that what is difficult in inference is model formation, and highlights the key attributes that make model building difficult: number of entities and the depth of relation. Depth of relation is a measure of how much information is required to make an inference; essentially the number of statements about relationships that must be considered simultaneously to make an inference.

How can this understanding of reasoning be applied to help non-visual graph browsers? The process can be represented diagrammatically, as shown in figure 2.4. There are essentially two phases to the process: building the model, and manipulating it, although these may occur somewhat in parallel. For the first phase, knowledge of the task and its context (including the domain) will give the reader a very approximate notion of the nature of the graph, even if this is more in the sense of excluding possibilities than imagining the graph itself. This vague

\(^1\)The term iconic here is used in the sense of symbolic, in specific contrast to direct imagery.
2.4. MODELS FOR GRAPH BROWSING

Figure 2.4: Diagram representing the process of model building and manipulation. With some initial knowledge of the task and the graph (e.g., domain, etc.) the user builds a vague model of the graph. This is formed into a more concrete representation of the data as the user reads the diagram. This model may be manipulated so that the solution becomes apparent.

The model is refined into a more detailed and accurate representation of the data as the diagram is read. For simple graphs this may be brief and simple; for complex ones this is probably a more lengthy process involving considerable interaction with the graph. While this is occurring the person may manipulate the model so that the solution becomes apparent, again a process that may be utterly simple or extremely complex, and may or may not involve manipulation of the diagram itself. Note that both the refinement and manipulation parts of the process occur with reference to the task, e.g., the task may dictate that only part of the graph may need to be considered.

What difficulties are exacerbated in this process, or introduced to it, by presenting the data non-visually? The biggest problem for non-visual interaction is the increased difficulty in reading the graph — at all points in the process, the model is being developed or refreshed by reference to the data: the diagram is an external memory with fast read-times that is also sometimes writable. Changing to another, non-visual, representation of the data is likely to make this more difficult. Notwithstanding the extra read times of the data, the loss of other benefits of diagrams highlighted by Larkin and Simon, etc., can also hamper the
different stages of this process. For example, recognition of implicit features not only simplifies refinement of the model (the chunking effect essentially reduces the complexity of the graph and means that fewer refreshes are required), but may also help (or eliminate the need for) manipulation. How can the information be presented non-visually in a way that minimises these extra difficulties?

2.4.2 Information Retrieval

An information retrieval (IR) task is “the task executed by the information system in response to a user request” [3]: in essence a search. While it appears unlikely that searches might be used exclusively to understand a graph, one of the main benefits of diagrams highlighted by Larkin and Simon [51] was the way they facilitate searching by using two-dimensional space to group related items, so it is worth briefly exploring the literature in this field.

One of the notable features of the literature on information retrieval, with its bias towards the web, is the range of strategies users employ when searching. There is consensus that more experienced users (for example professional librarians) use a wider range of techniques than do novices. An example is the use of the ‘similar pages’ tool available on Google\(^2\), or ‘related records’ on the ISI Web of Science\(^3\). Are novices simply unaware of the existence of these tools, or are they unaware of their usefulness? People in unfamiliar environments are often unwilling to explore the options and stick to the first technique they find that works (like, for example, novice Internet users’ over-use of the ‘back’ button on their browser [88]).

Chen and Dhar [25] classified search strategies into 5 classes. These were:

1. Known-item-instantiation: use of known search terms to retrieve documents from which other search terms may be obtained.

2. Search-option-heuristics: Search for words expected in title or subject

3. Thesaurus-browsing: Librarian translation of users search terms to index words.

4. Screen-browsing: Browsing the list of catalogue index terms.

\(^2\)http://www.google.com

\(^3\)http://wok.mimas.ac.uk/
5. Trial-and-Error: Searching with words that just come to mind.

These strategies are generally aimed at retrieving a small subset of interesting items from a large set of documents, typically with boolean queries. Although it will be important to allow a search using a variety of strategies, this type of search is not difficult in graph browsing, due to the relatively small size of the graph and domain restricted content. The strategies we are interested in are those which allow users to conceptualise the relationships between items, a task very closely related to mapping. This difference may be characterised as relating rather than collating information. The general idea, however, is the same — strategies are skills which are developed with experience, and new or occasional users either don’t know all available strategies, or find it difficult to select the most efficient one. In the IR field, Brajnik [14] developed a collaborative help system to assist users in making strategic and tactical decisions.

It could be argued that it is not possible to understand the structure of a graph by searches alone: as mentioned it is difficult to relate the information retrieved. Extending the concept of searching to more general queries opens the possibility of allowing the graph to be understood not by moving around its constituents, but by posing a series of questions that focus in on the particular information required. Retrieval of the necessary information could then be achieved in a manner akin to the ‘20 Questions’ game. In such a situation, however, the focus of the questioner’s attention is likely to shift around the graph as an understanding develops; this was the case in the vocabulary experiments (described in Chapter 3). It could be argued that this manner of directing questions brings some feel of browsing or exploration to the process. Browsing will be discussed in the next section.

Despite these differences, there are lessons that can be taken from visual information retrieval research and potentially applied in the non-visual domain. If nothing else, it is probably worth considering Shneiderman’s ‘Visual Information Seeking Mantra’ [85]. This principle emphasises the need to start with an overview of the information, then zoom in on details of interest and filter out uninteresting ones, finally getting the fine details on demand (i.e., not immediately presenting the user with lots of fine detail). This principle can easily be translated to non-visual interfaces — as done by Zhao et al. [118] with their Auditory Information Seeking Principle: Gist, Navigate, Filter, then Details-on-demand. It should also be noted that this principle is equally applicable under a spatial
exploration model as it is under an IR model.

In a sense, the IR approach is aligned with the propositional model of reasoning, in which case it is necessary to support the reader in making inferences. This is perhaps best achieved by making as much information as possible explicit. This is clearly difficult, except perhaps in some domains where the range of tasks is limited and predictable. Even if the developments in IR are only considered for providing basic search mechanisms and not the more difficult problems involved in non-visual interaction with graphs, they do highlight the need for such facilities, and stress the usefulness of having multiple strategies available and providing support for their use.

### 2.4.3 Spatial Exploration

Reading the diagram can also be considered a journey through information space, this being an environment where information is distributed; travellers may move between items of information, querying the content at certain point (this is similar to the old idea of navigating hypertext [10]). The spatial configuration of information space may be said to reflect the relationships between items of information. Movement through this space has direction, and this is determined by the traveller — the activity of deciding route choice is reliant on the reader querying the local information, reviewing the options and making a decision: this is navigation. Under this metaphor, the goal of the reader is to locate the information required, or determine its absence, and perhaps deduce its configuration (relationships). This metaphor sits neatly with the mental model explanation of reasoning, and suggests that the key requirement is to facilitate model building (cf. point 5 of Goodwin and Johnson-Laird’s requirements above).

The spatial metaphor for hypertext browsing (e.g., the World Wide Web, where it is common; used for example by Sorrows and Hirtle [90]) has been questioned by Boechler [13], who argues that a metaphor must not only explain the parallels between source and target domain but also the cognitive processes involved. She suggested the spatial metaphor of hypertext browsing might break down, particularly for more experienced users. One particular problem with this metaphor that she highlights is the concept of distance, and the three geometric axioms (symmetry, minimality and triangle inequality) implied by metric space. The difficult question that arises from this examination is ‘what is distance in hyperspace?’.
Despite the difficulties of this metaphor for hypertext browsing, a first glance indicates that the spatial metaphor may be applicable to graph browsing. Although the absolute positioning of nodes in a graph can change only the readability, not the meaning, of the graph, it is possible to consider distance as a real phenomenon. For example, the distance between two nodes might be given in terms of the number of arcs needed to traverse between them. It may be useful to note that this is domain dependent — for very large hypertext graphs, such as the World Wide Web, arc-distance measures a type of relationship that is not necessarily useful or interesting to the reader. For many graphs, however, this distance is very important. These difficulties will be discussed in more detail below.

This difference must not be dispatched with lightly, however, since applying such an analogy is only useful if its limitations are understood. In a graph the space between nodes is not accessible (or indeed, meaningful); not only does this lead to the difficulty in understanding distance, it also has implications for movement around the space, and these must be considered carefully, since it is exactly this that the analogy is supposed to illuminate. In traditional space, movement is much less constrained and people are able to move relatively freely between locations. They do this by navigating. Navigation ("to keep on one's course" [103]) is a question of moving from one location to another. Its precursor is orientation ("The action or process of ascertaining one's bearings or relative position, or of taking up a known bearing or position." [103]). In other words, to move somewhere, one must identify where one is and the direction to one's destination (orientate), then follow that direction (navigate), not necessarily along a simple path, and likely requiring reorientation.

How does moving around a graph differ from moving around real space? One essential difference is perhaps that navigation is no longer a significant problem: there is no real space so there is no need for a compass. Since one is always somewhere meaningful (a node or an arc) and the space between nodes is meaningless, moving between locations is a matter of following a path. Although there are aspects of navigation in this task (similar in some ways to moving around a city, where one is restricted to the streets), it is perhaps more useful to consider it as a relating task, in that the problems are: firstly, to know where one is (i.e., keep orientation) and secondly, to know how the current location relates to the
destination. Thus, although in many fields movement around this kind of information space is considered to pose problems for orientation and navigation, for this particular application, the latter is perhaps more usefully characterised as relating.

This idea of moving through information space has been equated with browsing. A useful definition of browsing is difficult to find, but its etymology suggests a process involving scanning and selection — the original use of the verb browse was:

“to feed on the leaves and shoots of trees and bushes; to crop the shoots or tender parts of rough plants for food... (... properly implying the cropping of scanty vegetation.)” [103]

Different types of browsing have been identified dependent upon how definite the reader’s target is, i.e., how well defined the search goal is. The spectrum of types ranges from highly structured, specific searching (‘closed’) to unsystematic exploration with little focus (‘open’); Cove and Walsh [27] select three points in this range:

1. Search browsing: a closely directed and structured activity where the desired product or goal is known.

2. General purpose browsing: an activity that consults specified sources on a regular basis because it is highly probable the sources contain items of interest.


It is worth noting that this range considers browsing in its widest sense, but some define browsing to be the only subset of these containing the least structured forms of search. It also should be noted that the nature of the reader’s target will be dependent upon not only his or her knowledge of the domain, but also the specific information they are exploring. This target will therefore be evolving in both the long term (graph to graph, as the domain knowledge increases) and the short term, as exploration increases their knowledge of the particular graph.

Having established that the concept of browsing within the metaphor of spatial exploration (of information space) is valid for this application, it is possible to
examine what is known about the cognitive processes involved in real spatial exploration, to understand the nature and development of cognitive maps with a view to applying this knowledge to the creation of tools for assisting non-visual readers in the understanding of graphs.

2.4.3.1 Mental representations of environments

It is the established view, although not conclusively proven, that spatial environments are represented in the mind hierarchically. For example, Stevens and Coupe [94] accounted for distortions in spatial judgements by proposing a hierarchical coding of the information. These errors are interesting by-products of a hierarchical style of data storage. In their example, participants judged Reno, Nevada to be northeast of San Diego, California, despite the fact that it is actually to the northwest. The knowledge that the superordinate unit Nevada is to the east of California influenced judgements about the locations of subordinate units. Other studies have provided evidence supporting this model and this is now the dominant view [18].

As well as understanding the nature of the cognitive representations of environments, it is also necessary to understand the process by which these representations are formed. Siegel and White [87] proposed a series of stages for the acquisition of spatial representations:

1. Landmark recognition: Individuals learn to discern and remember separate landmarks.

2. Landmark coordination: Landmarks are coordinated in a sequence, resulting in route representations, comprised of both spatial and temporal relations between landmarks.

3. Survey knowledge formation: Route representations become more refined and a representation of the entire space forms, including relations between landmarks and routes.

More recently, however, it is believed that these stages are not acquired strictly in order but can occur in parallel [39]. Boechler [13] stated that survey knowledge is qualitatively different from route knowledge since ‘it incorporates the understanding of metric distance relations between landmarks and routes’; this is a useful distinction which will be returned to later.
The research described above concerns how models of environments are built by people who are within those environments. But the way in which people construct models of environments from descriptions has also been studied. For example, Taylor and Tversky [98] asked participants to read descriptions of an environment then answer questions about that environment. The descriptions were either route-based or survey-based. Another set of participants answered questions after examining a map. They found that although questions were answered faster and more accurately when taken verbatim from the description (cf. questions requiring inference), there was no difference between route descriptions and survey descriptions. They did note, however, that participants spent more time reading the route descriptions, suggesting that these are integrated into a mental model more slowly. Other findings from Taylor and Tversky’s experiments were that model formation was automatic, that is models encoding the spatial relations among landmarks were constructed even when participants were not expecting to need that type of information.

Overall, these experiments suggest that not only are models formed by interacting with an environment, but also by reading descriptions of an environment. The model structure does not appear to be influenced by the perspective of the description. They did, however, note that there are several important requirements for the descriptions:

“The narratives must be well constructed: coherent, organized and unambiguous. People fail to construct adequate representations when the spatial information comes from indeterminate, poorly organized or overly difficult descriptions”

2.4.3.2 Application to non-visual graph browsing

If reading graphs non-visually is considered to be a browsing activity, and that this has an exploratory or spatial feel to it, then any tool facilitating it must enable users to build a cognitive model of the graph via steps analogous to those involved in acquiring spatial mental representations of environments. It is, however, important to note that compared to real environments, travel in a node-arc environment is constrained by the requirement to move only along arcs. Boechler’s distinction between survey and route knowledge, above, would suggest that for graphs survey knowledge is largely irrelevant, since only arc distance is meaningful; this is the basis for her dismissal of the spatial metaphor for this domain.
2.4. MODELS FOR GRAPH BROWSING

It is still possible to argue that there are similarities in the first two stages of exploration that may be exploited, and we could propose an analogous final stage, where the explorer perceives the relationships between different nodes and routes to understand the topology of sections of the graph, or of the graph as a whole. In this case, we can identify the stages involved in understanding a graph, and suggest some difficulties that are likely to be encountered.

1. Landmark recognition can be considered a phase of exploration where the user gains an understanding of the types of nodes present. The essence is *discovery*, and it is typified by wandering around the graph with no particular direction in mind, rather an intention to see what there is. In this phase difficulties could include:

   (a) Understanding the scope of the graph, e.g., knowing when all nodes have been visited.

   (b) Distinguishing nodes from one another.

   (c) Visiting all nodes, i.e., finding unvisited nodes.

2. Landmark co-ordination is a phase where movement generally has an intended destination, even if the exact location of the destination is unknown. For example, the user wishes to know the arcs and nodes between two particular nodes.

3. The graph exploration analogy of survey knowledge formation is probably linking (relating) sections of graphs rather than combining routes into a survey, although it is essentially the phase where an overall understanding of the graph is developed.

The phases given above are for general exploration; browsing for a particular task might not require development of a full mental model of the graph. For example the user might only need to know from a graph whether a certain node type exists. Similarly, many tasks might only demand a good model of a subgraph; in this instance the rest of the graph need only be explored to the level of landmark identification (or sufficient to recognise the area of interest).

If we are to apply this model to non-visual graph browsing, can we use it to suggest a presentation model — one that makes the development of route and survey knowledge easy? The next section shows that hierarchical data structures
have been used previously [96, 101, 8] and that they facilitate overviews. Learning
that spatial environments are also represented hierarchically in the mind, suggests
that, at worst, there is unlikely to be significant disadvantages to using a hier-
archical presentation model. We must bear in mind, however, that Bennett [8]
found that some problems were better suited by a connection-based presentation.
A model that allows both seems the best option; perhaps some ideas from infor-
mation visualisation, where coordinated multiple views are considered an effective
way of exploring data [86], may be adapted for non-visual exploration.

2.5 Tools for Audio Presentation of Graphs

This section examines some previous research into presenting graphs to visually
impaired users, with emphasis on evaluated tools. It does not cover all proposed
solutions, instead it describes a selection chosen to exemplify the different prob-
lems and approaches. As discussed in section 1.2, this thesis concentrates on audio
solutions, mainly due to their relative cheapness and widespread availability; this
review is similarly biased.

After introducing the tools and the key findings of the research (Section 2.5.1),
Section 2.5.2 examines how the systems described above tackle the problems of
presenting through the audio channel, and how, if at all, they attempt to replicate
the benefits of diagrammatic (visual) presentation.

2.5.1 The Tools

Several groups have considered the provision of overview facilities to be useful
or essential. One example is the development of tools to make algebra nota-
tion accessible to visually disabled students, a task that has many parallels with
graphs in that the different sub-expressions of an equation can be considered
nodes connected by operators (such as +, -, =), and the equation cannot be sim-
ply read from left to right. Stevens et al. [96] considered a complex equation to
have a hierarchical structure where a sub-expression is itself composed of sub-
expressions. They highlighted the need for the reader to be active in his reading,
not just passively have the equation read to him. They also stressed the need for
summarisation as a method for the reader to keep in mind his ‘location’ in the
equation and estimate the complexity of an expression. This was facilitated by
the hierarchical view. Most of the techniques were designed to overcome lack of an external memory and to increase the control over the flow of information.

The TeDUB project \(^4\) ("Technical Drawings Understanding for the Blind") is looking at making technical diagrams accessible, with much effort concentrating on extracting semantics from diagrams in their visual form. Its partners completed an evaluation of how experts in different fields described diagrams to each other \([100, 41]\). This, as well as reiterating the usefulness of overviews, also highlighted the task dependence of the descriptions and the usefulness of relative locators (such as North, South, East and West). It appeared that floor plans were described as if walking through them \([71]\), and were considered very different from Unified Modelling Language (UML) or circuit diagrams. Further work extended to systems that used hierarchical data structures to allow overviews, in a similar fashion to Stevens, above. Evaluation of their ‘EuroNavigator’ system \([99, 41]\) (which contained data about European countries) indicated that this reduced cognitive load, but they commented that strict hierarchies were not effective for navigation between related nodes, and that ‘Navigating hierarchies at the lower levels no longer remains intuitive and it can be difficult to know where something of equal rank down another branch is’. Their later system for understanding digital circuit diagrams \([41]\) presented information in a hierarchical structure, for example a group of logic gates might be grouped to form a half-adder. This was intended to chunk the information to reduce the demands on short-term memory and facilitate overviews. It also addressed the issues around pure hierarchical navigation by allowing both hierarchical and connection-based movement.

A different approach to those structuring the data into hierarchies was that taken by Blenkhorn and Evans \([12]\), who concentrated on the connections between nodes. They created a system, known as ‘Kevin’, which used a tactile pad in combination with audio output to allow visually disabled users to read and edit a form of data-flow diagram used by software developers. The tactile pad was composed of two regions, with the output area split into a \(N \times N\) grid, where \(N\) was the number of nodes in the graph. The leading diagonal of the grid gave access to the nodes and their attributes, while the remainder was used to give access to information about the connections. The user could find out what nodes there were, and to which other nodes they were connected, by following his finger along either the row (connections leaving the node) or column (connections entering the

\(^4\)http://www.tedub.org/ (last checked 9/3/07)
node) containing that node. It is clear that this system would be inappropriate for graphs containing large numbers of nodes, but the diagrams considered form part of a hierarchy themselves, so one diagram should never be too complex.

Bennett looked at the limitations of the Kevin system described above [8]. He felt that the method by which the Kevin user moved around the diagram was different from the original and the two representations therefore lacked computational equivalence. He proposed that presenting information as a hierarchy would afford the benefits Simon and Larkin associated with grouping. He investigated how the nature of the task influenced whether diagrams were better presented with this hierarchical structure, or with a connection based structure, as with Kevin. He conducted some experiments using central heating schematics as test diagrams, and demonstrated that hierarchically presented information facilitated hierarchical tasks, but that if the tasks were navigational the information was best presented with an emphasis on connections. Although he mentions a system allowing both types of browsing, this was neither described nor tested.

Presentation of location information was more commonly found (implicitly) in systems that used tactile methods for presentation, such as Audiograf [49]. Bennett, however, also investigated its use in audio presentation, since he felt that previous work ‘suggests that position information is part of the reason why diagrams are successful representations’; he argued that not knowing the location of the components in the original diagram creates an informational inequivalence. He therefore also investigated if musical ‘earcons’ (the audio equivalent of graphical icons) presenting coordinate information would ease the tasks, but found no evidence to support this hypothesis. It is debatable if lacking coordinate information destroys informational equivalence; it is arguable that in graph-based diagrams the inequivalence is purely computational.

The use of non-speech sounds was a feature of some work, notably Stevens et al, who investigated their use, along with prosody, to improve the quantity of information output without overwhelming the listener with descriptive speech. Considering the use of hierarchies in many of the systems described above, another relevant piece of research concerning non-speech sounds is that by Brewster et al. into the use of 3D earcons to aid navigation through a hierarchical menu in telephone-based interfaces [15]. They found the earcons helpful in giving a quick reminder of current position, but problems arose with earcon length when deep in a hierarchy.
Although the designers of many of the systems described above are not explicit about their reasons for implementing in a particular manner, some themes emerge. It is clearly considered useful to get an overview of the diagram; this would appear to be the reasoning behind the hierarchical methods of navigation favoured by some. The building of a hierarchy was also considered beneficial by reducing memory load. There is also agreement over the need to give the user control over the information flow — all systems allow the user to move around the diagram to control what information they receive. The methods for moving around diagrams appear to be the most variable feature of the systems examined, although there was recognition that the best method of interaction is task-dependent.

2.5.2 Comparison of Theory with Practice

The model proposed by Palmer for visual perception fits with the hierarchical data organisation used by some of the systems above (e.g., Bennett, TeDUB), although none of these cite it as justification. Building a hierarchical data structure for the information in the diagram (or providing another means of viewing the information hierarchically) could allow the audio representation to be overviewed in a manner analogous to (although presumably much slower than) a visual representation. Whether this then provides real benefits to the non-visual reader is perhaps open to question, although it seems intuitive that overviews are useful, for example by allowing an uninteresting or irrelevant diagram or part of diagram to be safely ignored.

It is striking that only one of the systems described above provided any facility for making implicit features of the diagram explicit. This is the feature of diagrams identified by Larkin and Simon as the most beneficial, yet it seems to have been ignored. Even the TeDUB DiagramNavigator, which identified larger features (such as half-adders), did not justify this feature in terms of easing recognition, but claimed reduction of the number of items in the diagram to remember. This is a valid claim, and is as important as easing recognition when users are denied the easily accessible external memory afforded by the visual representation.

The idea of giving users positional information was proposed, and found to be ineffective, by Bennett. Analysis of the reasons for the benefits of visual diagrams explains this point. The knowledge of the location of a node is not in itself particularly interesting (although it might help readers build a mental
representation that is closer to one built by a sighted colleague), rather it is grouping by location that provides benefits. These are achieved by making it easier for the reader to search, e.g., to identify related nodes; it is therefore not as useful to know exactly where a node is, as to know what other nodes are nearby. The closest relationship between nodes on a graph is probably that of connectedness, so it should be made possible to identify to which other nodes a node is connected. This gives some explanation for why the TeDUB studies found pure hierarchical systems unsatisfactory.

2.6 Summary

This chapter has examined some of the cognitive science behind the use of graphs. The main advantages of using diagrams to present information to sighted readers are that searching is simplified because related information is grouped, and some features of the graph that would otherwise need to be deduced are made explicit. If the visual channel is not available, and sound is used instead, several new challenges are faced by the reader, largely as a result of the essentially linear nature of sound. When explored visually, all parts of a diagram are instantly accessible, allowing rapid movement over the information and enabling it to be viewed at different resolutions, from gross overview to the finest detail. This instant accessibility, and the persistent nature of the diagram, allows the diagram to act as an external memory that can be interacted with. This contrasts to the transient nature of sound, which can strain the limited short-term memory.

The concept of representing information as mental models (as opposed to propositionally) was introduced. Model formation is seen as the critical step in problem solving; this is difficult if many statements must be combined to infer a model, but simplified if the information is explicit and presented in a suitable order.

Two potential analogues for non-visual graph reading have been explored - information retrieval, and spatial exploration (browsing). Although the field of information retrieval offers useful lessons, the task is considered to be more like spatial exploration. This analogy is not without difficulties, however, since the nature of graphs means that the space between nodes, and therefore distance, is meaningless. Although the number of arcs between nodes can be envisaged as an equivalent of distance, problems remain with the concept of navigation.
2.6. SUMMARY

It is proposed that the problem of navigation is replaced by that of identifying relationships between nodes.

Hierarchies have been a recurring theme: Palmer proposed that visual perception is hierarchical, while there is also evidence that spatial environments are represented hierarchically in the mind. The development of a cognitive representation of an environment is proposed to follow three stages: landmark recognition, landmark coordination and survey knowledge formation. The difficulties noted in relation to distance suggest that survey knowledge is perhaps better characterised for this problem as the stage where relationships between nodes or sections of the graph become apparent.

Basing non-visual graph presentation on a hierarchical structure fits neatly with perception and the mental representation of environments, offering a neat mechanism for overviews and presenting groups of nodes as a single explicit feature. However, if the spatial analogy is to be useful, connection-based exploration must also be available.

Several tools for non-visual access to graph-like information exist, although they have generally been presented with little or no explanation as to the design decisions. No tool exists which addresses all, or even the majority, of problems associated with non-visual reading, and few of the benefits of diagrammatic presentation are replicated. There is a clear need for a more rigorous approach to the design of this type of tool, with design based on an understanding and application of relevant cognitive science.
Chapter 3
Formative Experiments

This chapter describes two experiments designed to provide a more concrete understanding of the types of difficulties encountered when exploring graphs. They were intended to support the literature survey in the development of an understanding of the process of reading graphs, i.e., answering the first research question. Two approaches were taken: the first was an evaluation of a tool for non-visual browsing of molecular structure graphs, Kekulé\(^1\); the second was an experiment where people were asked to describe graphs to each other.

Kekulé was largely developed during previous research\([19]\) with some modification for the evaluation. This tool was designed to address some of the issues raised in the previous chapter, particularly making some graph features explicit to users, and the evaluation was performed to identify if this made tasks easier and to highlight where further development was required. Section 3.1 describes Kekulé, concentrating on which aspects of Chapter 2 were implemented and how users use it to interact with the graphs. A running example is used to demonstrate. Section 3.2 describes the evaluation method and results; these are discussed, with reference to the last chapter, in Section 3.2.3.

Observation of some difficulties with the vocabulary used by Kekulé prompted a further experiment aimed at identifying how people would describe graphs given the freedom to choose their own methods and vocabulary. The description experiment involved pairs of participants describing graphs to each other: the describer could see the graph and tried to describe its structure so that the listener would be able to visualise the topology then draw it. It had two main aims — to gain

\(^1\)Friedrich Kekulé was a 19\textsuperscript{th} century German chemist who discovered that carbon always had four bonds and could form chains.
3.1. KEKULÉ: BROWSING MOLECULES

A common and important type of graph, chemical molecules were chosen as an exemplar class for a variety of reasons. Organic chemistry is an important subject, so although visually impaired professional chemists may be rare, it is important to enable pupils at school to obtain an appreciation of this subject. This is impossible without understanding molecular structures. Figure 3.1 shows a typical example of such a diagram; this is for the amino acid phenylalanine. These graphs are a useful test case from the researcher’s point of view: as a class they form a good example of the general problem of non-visual graph browsing, while having enough constraints to keep the problem manageable. These constraints include limits on the number of node and edge types, and particularly limits on node connectivity (e.g., a carbon atom may only ever have a total of four bonds connecting to it). A well defined nomenclature should also aid description.

Molecular structure diagrams are used to deduce or understand how the structure of a molecule affects its properties. For example, early lessons in organic chemistry describe how the length and linearity of hydrocarbon chains affect melting and boiling points. For chemists with a higher level of skill, knowledge of the atoms and their layout enables prediction and explanation of how different chemicals will react. Of special interest are those clusters of atoms which are neither carbon or hydrogen — functional groups. Examples of functional groups are the carboxylic acid and amine groups marked on phenylalanine in Figure 3.1. These groups are given names and react in known ways. If a non-visual representation of a molecule is to be useful, the reader must be able to identify what functional groups are present and how their locations relate to each other and the structural features (hydrocarbon chains and rings) of the molecule.

Kekulé is a tool for reading molecular structures from Chemical Markup Language [61] files — CML² is an international extensible markup language standard where molecules are essentially described using cross-referenced lists of atoms and bonds. The software allows users to move around the different parts of the molecule in order to gain an understanding of its structure. The user may browse at different levels of detail, either moving between adjacent atoms or between

²http://www.xml-cml.org/
3.1. KEKULÉ: BROWSING MOLECULES

Figure 3.1: Phenylalanine. This molecule is an amino acid which has a ring of 6 carbon atoms (known as a phenyl group, highlighted) connected to the base structure of amino acids. This is composed of groups of atoms — a carboxylic acid and an amine, also highlighted.

higher level features of the molecule.

3.1.1 Data Structure

In Kekulé, the data are structured to allow the browser to recreate those features of visual diagrams identified by Larkin and Simon as useful. Specifically Kekulé identifies features of molecules that can be explicit in the diagrams, but are only implicit in the initial (CML) representation. These features are used to build a hierarchical data structure that parallels the hierarchical nature of visual perception proposed by Palmer, with atoms at the bottom and the entire molecule at the top. Figure 3.2 shows such a hierarchy for phenylalanine. The features include both fundamental geometric patterns that may be identified by any user, such as rings and chains (e.g., the ring in phenylalanine encountered above), and features that would be obvious only to an experienced chemist, i.e., functional groups (e.g., the carboxylic acid highlighted in Figure 3.1).

In the example of phenylalanine, this hierarchical structure allows the user to see that the molecule is composed of a phenyl ring, an amine, a carboxylic acid and a couple of other carbon atoms. At a more detailed level, it is possible to see that the carboxylic acid contains four atoms: a carbon, two oxygen and a hydrogen.

In order to enable searching, Kekulé is designed to allow users to find out what is connected to the current object of focus. Thus those nodes that are most closely related to the current node — the atoms which are bonded to it — are easily found. For example, having established the components of the molecule,
3.1. KEKULÉ: BROWSING MOLECULES

connection-based browsing allows one to determine how they are connected to
one another. The key features of Kekulé are the explicit presentation of implicit
features through hierarchical browsing, and the combination of this hierarchical
browsing with connection-based browsing so the benefits of clustering are not
lost.

There are two phases to the use of Kekulé: building the data structure, and
browsing. The latter, a process which will continue as long as the user desires, is
described below, while the former is composed of three distinct sub-phases:

1. The CML file is read using the publicly available CML Document Object
   Model (CMLDOM), and a simple data structure created. For phenylalanine
   this will have a single node representing the molecule, while below this in
   the hierarchy are nodes for the 12 non-hydrogen atoms.

2. The molecule graph is searched for chemically interesting features. Ull-
   mann’s algorithm for subgraph isomorphism [104] is used to identify func-
   tional groups; Balducci and Pearlmann’s algorithm [5] is adapted to identify
cyclic features and chains of carbon atoms.

3. An optimisation algorithm simplifies the data structure. The result is tree-
   like with the molecule as root and atoms as leaves; the intermediate nodes
   represent collections of atoms such as chains or functional groups. Figure
   3.2 shows the optimised data structure for phenylalanine.

3.1.2 Browsing

Kekulé is designed to help readers understand molecular structures for the sorts
of tasks described in the introduction to Section 3.1: detection of functional
groups and their relationships with each other and the structural features of the
molecule. With Kekulé, a user issues commands from the keyboard and text
strings representing output are sent to a speech synthesiser. The commands
are designed to allow the user to move around the data structure in either a
connection-based or hierarchical manner, and are illustrated in an example below.
The essential commands are those that allow the user to zoom in and out, that
is examine the molecule at greater or lesser detail, and move between connected
atoms or groups. Other commands help with summarisation (e.g., the formula
3.1. KEKULÉ: BROWSING MOLECULES

Figure 3.2: Hierarchical organisation of groups in Phenylalanine. This is the final form of the data structure in Kekulé.

tells the user how many of each type of atom there are within the current group), or give supplementary information that might help them to build a mental image (e.g., the spatial position). Spatial information may be available if the CML file contains coordinates for the atoms — this is often the case as these files are mostly generated from molecule drawing software. Kekulé operates on a ‘move as you hear’ mode: whenever an atom or group is spoken it becomes the focus.

When the initialisation is complete the user is located on the whole molecule, i.e., the focus is on the top of the tree. Kekulé is designed to allow users to browse at as high a level of the hierarchy as their chemical knowledge allows. It was envisaged that they initially use the ‘zoom in’ command to determine what the components of the molecule are, then move between them to find out how they are connected. If the user is not familiar with a component, or there is not sufficient information to know its structure in full detail, they may zoom in further and explore it in a similar fashion.

For example, if the reader is trying to look at the structure of phenylalanine (Figure 3.1) they can zoom in from the top level (see Figure 3.2) and simply hear:

‘Zoom In. Amino acid, CH$_2$ number 7, phenyl’

Thus the user knows that phenylalanine has three components: an amino acid group, a phenyl group and a lone carbon atom (numbered 7). If they understand the structures of these components, the overall structure may be fully understood by moving along the connections (bonds) between these three nodes. If the
structure of an amino acid is unfamiliar, or uncertain, the user may zoom in to explore it at a greater level of detail.

To illustrate this in more detail we may follow a worked example, exploring again the structure of phenylalanine. Imagine here that the user wishes to understand the structure of phenylalanine, with emphasis on its functional groups. The example starts from the top of the tree, and is given in the format:

**keyboard command** A description of the command.

‘Speech output’ Comments.

Starting on the molecule:

n The name of the current node. ‘Phenylalanine.’ The name of the molecule is phenylalanine.

+ Zoom in. ‘Zoom in. Amino acid,’ Zooming in to see what the molecule is composed of. We have a list of three components at this level, only the first of which (amino acid) is spoken automatically in order not to overwhelm the listener. This is now the focus.

↓ The next item. ‘CH₂ number 7,’ This accesses the second component in the list — a carbon with two hydrogens; it has the (arbitrary) identifier 7. This atom becomes the focus now.

↓ The next item. ‘Phenyl.’ Followed by a non-speech sound, this indicates that the final component is a phenyl group. We can move between these three nodes with the ↑ and ↓ keys.

← Back in history ‘Phenylalanine.’ Move the focus back to where we were before issuing the zoom in command — on the molecule.

f Get the formula. ‘C₉,’ There are 9 carbon atoms in the molecule. Using the ↓ key as before gives the other elements: ‘H₁₁,’ ‘O₂,’ ‘N₁’. This gives an idea of the size of a node, and whether it contains anything interesting.

+ Zoom in. ‘Zoom in. Amino acid,’ Zoom in to the amino acid to see what this is.

f Get the formula. Using ↓ to scroll down the list of elements gives ‘R₁,’ ‘C₂,’ ‘H₁,’ ‘O₂,’ ‘N₁’. The R₁ indicates that the amino acid is bonded to one other thing; the rest give the atomic composition.
+ Zoom in. ‘Zoom in. Carboxylic acid,’ Zoom in to find the components of the amino acid.

↓ The next item. ‘CH number 8,’

↓ The next item. ‘Primary amine.’ This is the last component of the amino acid — assuming the user has enough chemical knowledge to understand what these components are, it is now necessary to discover how they are connected. This item is currently the focus.

c Get the connections ‘1 connection. To CH.’ The primary amine is connected to a carbon (which becomes the focus).

return More information about the connection ‘North bond to CH number 8.’ The bond is single, and the carbon is C number 8, and lies to the North of the amine.

c Get the connections ‘3 connections. To CH2,’ CH number 8 is connected to a carbon. The focus changes from the primary amine to the carbon.

↓ The next connection. ‘Carboxylic acid,’...and a carboxylic acid (now the focus).

↓ The next connection. ‘back to primary amine.’ The 3rd bond is the one we have just travelled along.

↑ Back up the list ‘Carboxylic acid,’ Get the focus on the carboxylic acid.

return More information about the connection ‘North East bond to carboxylic acid.’ We can now deduce that the amino acid is a carbon with one hydrogen (CH number 8) which has single bonds to the carboxylic acid and the primary amine, and also has a bond to a CH$_2$ (which is not part of the amino acid).

- Zoom out. ‘Zoom out. Amino acid,’ Zoom back out to the amino acid.

- Zoom out. ‘Zoom out. Phenylalanine,’ Return to where we started — the molecule as a whole.
3.2 EVALUATION

Using this combination of zooming and exploring allows users to understand the components of the molecule. Note that in this case some basic chemical knowledge (what primary amine and carboxylic acid functional groups are) allows the user to browse at a high level: it is not necessary to explore individual atoms. The example above does not illustrate all of the commands; the others are:

p  Gives the current spatial position on the diagram, for example, ‘Top left.’. This is only given if available in the CML file.

s  Summarises the components of the current node as the numbers of rings, chains, functional groups, polycyclics and other atoms.

w  Gives the location in the hierarchy, e.g., ‘On hydroxyl,’ ‘on carboxylic acid,’ ‘on amino acid,’ ‘on phenylalanine.’.

3.2 Evaluation

The evaluation was intended to establish whether the features provided by Kekulé for recognition of implicit features and chunking of information made it easier for users to understand the structure of a molecule. This was an exploratory investigation intended to elucidate generic problems, and no attempt was made to determine the effect of individual aspects of the software, although participants comments were noted.

3.2.1 Method

To assess the ease with which users were able to develop an understanding of the structure of molecules, two versions of the program were compared — the full version (as described above), and a more basic version. This version started with the user located on the first atom (as defined in the CML file) and able only to move between atoms along the bonds — a connection based system, in contrast to the combined hierarchical and connection based approach of the full version. The users were given three molecules in each mode and asked two questions about each molecule. The first molecule in each mode was considered simple while the second two were more complex (see figure 3.3 for examples). Molecule complexity is subjective, but was judged according to the size of the molecule, as well as the
3.2. EVALUATION

presence of complex features such as rings or large functional groups. The order in which the participants used each mode varied, as did the molecules used for each mode.

![Figure 3.3: Examples of question molecules. The questions were: Molecule (a) 1. How many atoms does the longest chain in this molecule contain? 2. How are the non-carbon atoms connected? Molecule (b) 1. How are the acid chloride and amide connected? (These terms were explained.) 2. Where is the double bond in the ring (relative to where the ring is connected to the rest of the molecule)?](image)

In order to assess how difficult the participants found the tasks three methods were used:

**NASA Task Load Index** After completing the tasks for each mode the participants were asked to assess the different aspects of difficulty using the NASA task load index (TLX) [37]. The TLX scores tasks on 6 aspects: Mental Demand; Performance; Effort; Frustration Level; Physical Demand; Temporal Demand. In each case a higher score (on a scale of 0 to 20) indicates a more difficult task.

**Cooperative Evaluation** [111] The evaluator talked with the participants and encouraged them to voice their thoughts about the tasks; these were recorded.

**Software logging** The software logged all keystrokes and output for later examination.

Participants were volunteers from the Computer Science department at the University of Manchester. They had varying levels of chemical knowledge, from
3.2. EVALUATION

almost none (beyond understanding that atoms are connected by bonds) to degree level chemistry. To avoid over-use of the relatively small pool of non-sighted people available for evaluations such as this, all but one of the participants (a graduate biochemist) were sighted. We believe this approach is justified for exploratory evaluations of this nature, on the grounds that non-seeing sighted users have all the problems associated with (at least adventitiously) blind users, but have not had the practice required to develop work-around solutions — they therefore behave like naïve visually disabled participants.

No time limit was given for answering the questions, although users were encouraged to consider giving up if they were spending a long time without feeling they were making progress. Each participant completed the tasks at a different time and had a short period of training where the command language was demonstrated and they could try browsing a simple molecule. During the training the participants could see a visual representation of the molecule on the screen, with their current location highlighted. This was intended to accelerate learning, and was not available during the exercises. Participants were not allowed use of a pen or paper. Each evaluation took approximately 90 minutes. Further details, including instructions for participants are in Appendix A.

The hypothesis was that the functionality offered by Kekulé would aid diagram exploration and understanding, so there would be a significant difference in TLX scores between the full and basic modes of software.

3.2.2 Results

3.2.2.1 TLX Scores

As each participant had different ideas about absolute levels of demand, frustration, etc., we have calculated just the differences between the scores for each mode. The score differences for four of these aspects are given in Table 3.1; physical demand and temporal demand were recorded, but are not presented here as they are not considered relevant to the hypothesis. In this table a positive score means that the participant found the full version of Kekulé easier to use than the basic version. Examination of Table 3.1 shows a general theme — on the whole the participants found exploration of molecules less demanding using Kekulé in its full version than the basic one. The scores are typically 2 or 3 points lower (on a 20 point scale) for the full version. The biggest difference was noted for
3.2. EVALUATION

Mental Demand, while the smallest was for Performance.

<table>
<thead>
<tr>
<th>Participant</th>
<th>MD</th>
<th>P</th>
<th>F</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-8</td>
<td>-4</td>
<td>-4</td>
<td>-10</td>
</tr>
<tr>
<td>2</td>
<td>-2</td>
<td>-2</td>
<td>-3</td>
<td>-2</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>-1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>-3</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>6</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>-3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>-1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Mean</td>
<td>2.58</td>
<td>0.75</td>
<td>2.00</td>
<td>1.50</td>
</tr>
<tr>
<td>SD</td>
<td>4.23</td>
<td>3.70</td>
<td>2.95</td>
<td>4.44</td>
</tr>
</tbody>
</table>

Table 3.1: Selected TLX score differences. The table value gives the difference between scores for the full and basic versions of Kekulé. MD is mental demand, P performance, F frustration, E effort. A positive value indicates the full version was scored lower (i.e., as less demanding).

Participant 1 showed a significant preference for the basic mode, for example scoring Mental Demand as 12/20 compared with 20/20 for the full mode. This is a significant outlier (Grubbs’ Test \( Z = 2.5 \), critical \( Z = 2.41 \) for 95% confidence). Removal of this participant would increase the mean difference in Mental Demand to 3.55, with a standard deviation of 2.73. Participant 2 also found the full version more demanding to use, although his preference was less strong than that of p−1. The visually impaired participant was not significantly different from the other participants.

If we discount the outlying participant 1, we may determine if the mean difference values are significantly different from 0. Using a one-sample t-test we get the following two-tailed P values: mental demand \( p = 0.0016 \); performance \( p = 0.2957 \); frustration \( p = 0.0053 \); effort \( p = 0.0107 \). These demonstrate that the TLX scores were significantly different for the full version of Kekulé for three of the four criteria. This confirms our hypothesis that understanding molecular structures was facilitated by the features offered by Kekulé.

On the whole the results show that participants found visualisation of these graphs a demanding task requiring good concentration. The median values for
mental demand were 14 and 17 for full and restrictive versions respectively. The maximum of 20 was given twice in each mode.

### 3.2.2.2 Cooperative Evaluation

Participants managed to answer many of the questions, although this often took considerable time. They were more likely to give up completely when using the basic version of the software. Although limited by the concentration needed to solve the problems, discussions with the participants during and after the evaluations lead to some interesting findings.

Domain knowledge has a big effect. The very different scores of participant 1 are explained by his complete lack of chemical knowledge. Although this participant understood that the atoms were connected by bonds and was able to explore molecules at this level with some success (better in fact than some with reasonable chemistry experience), he appeared to be very intimidated by the chemical nomenclature used to name the groups of atoms in the full version. Thus, despite being able to zoom in to atomic level and browse in the same fashion as the basic version (which he had done first) he felt unable to explore effectively. He commented:

> It’s grouping things together, and it’s putting these complex names to a group of things: I’m having real difficulty associating these names to the bits I zoom into.’

In contrast another user (p−9), a graduate biologist with a knowledge of genetics, was able to answer one question very quickly by using his domain knowledge. The question was on phenylalanine (participants were not given the names of the molecules) and asked ‘There is a carbon connected to an NH$_2$ in this molecule; to what other functional group is this carbon also connected?’ After getting the name (‘molecule’ — to check he was at the top level), his first command was to zoom in, upon which he was told that the molecule contained an amino acid group. In about 20 seconds these three keystrokes were sufficient for him to answer the question correctly. Compare this to a typical performance (p−10) on this question in basic mode, where approximately 90 keystrokes over 6 minutes were required. Worse, some participants got lost in the ring and never found the NH$_2$! A non-chemist (p−6) solved this problem using the full version with 35 keystrokes in about 6 minutes.
Atom numbering was confusing and needs to be improved. There is a standard method for numbering atoms in molecules that is meaningful and expected by chemists. Kekulé currently uses the arbitrary order of atoms in the CML file to number the atoms in turn. For the basic mode it was essential to label atoms, but the arbitrary nature of the numbering caused problems. For example, when moving around a ring the numbering might go 3, 4, 6, 8, 7, 5, rather than the more logical 3, 4, 5, 6, 7, 8. The additional position numbers used in rings and chains in the full version was useful, although at least one user commented that:

‘There’s too much to think about anyway, without having to cope with two numbering systems.’

This user (p−12, a graduate chemist) also found that if position numbering departed from the strict chemical rules it was even more confusing. Another (p−4) noted that since spatial information was available ring positions should be numbered in a consistent direction.

The method for following connections (the ‘move as you hear’ idea, described above) was confusing to some users. They wanted to be able to look around their current location without actually moving, but often changed atoms without moving themselves in their mental model. This discrepancy made the task exceedingly difficult. Inability to keep track of position was common, particularly in the basic mode. Many users said things such as ‘Where am I now?’, or ‘Right, what am I on now?’, indicating that it was not easy to remember.

Rings posed a particular problem, especially in the basic mode. Several participants made comments like ‘Oh no, I’m not in a ring now am I?’ It was said to be very difficult to know when one was in a ring or when one had moved out. One participant (p−12) in this situation said several times that he would really like to be able to zoom out now’ (he had used the full version of Kekulé first).

Strategy development appeared to be more simple with the basic software. The small number of commands limited the possibilities, so participants quickly developed strategies for exploration. These allowed small molecules to be explored fairly quickly with little learning time, but the strategies broke down when the molecules became too complex. For example, one of the more complex molecules had two joined rings which meant there were three adjacent carbon atoms, each with three bonds; it was not possible for users to remember which of these paths had been explored. Comments suggested that this was due to overloading of
short-term memory. In contrast, the extra commands available when using the fully functional version of Kekulé both took longer to learn and made it more difficult to develop strategies. The visually impaired participant commented on the difficulty of learning and remembering the commands, but afterwards thought that with practice it would be possible to ‘explore some pretty difficult molecules’.

An interesting feature of one participant’s (p−12) exploration in basic mode was his chunking. He identified a pair of atoms as forming a cyanide group, and thereafter, whenever he came across the carbon he said: ‘OK, so that’s the cyanide carbon’. This labelling presumably helped him identify and memorise this atom.

On a similar theme, another participant commented on the difficulty of trying to return to a previously visited atom, and suggested afterwards that a bookmarking facility would be useful:

‘Would it be a nice feature to be able to allow the user to set a “bookmark” on a particular node and then allow them to either jump back to there, or just tell the user when they are back there? One of the problems I frequently had was losing myself on the way back somewhere. If you could have multiple ones, you could use them to mark points where you had bits left to explore.’

This was a common problem: users often asked themselves if they had visited a certain atom or group previously.

3.2.2.3 Logging

Observation and analysis of logs indicate that a common strategy for coping with the difficulties of remembering position was to nominate an atom as a ‘home’. It was noticed that participants often latched on to a distinctive atom (such as a non carbon, or atom number 1) and used it as a base to explore from, returning (using the history ‘back’ command) when they got confused. This appeared effective, although it was not possible to go very far from the base before they could no longer remember the path. Such a strategy was not used in the full version of the software but participants used a similar technique of just zooming out to the whole molecule when confused.

There was reluctance to mix hierarchical browsing with connection-based browsing. This was possibly due to difficulty in visualising connections between
groups of atoms, when the bonds are actually between individual atoms. One participant (p−9) commented that he wanted to visualise himself on a central or important atom within the group, but did not know which that should be. He therefore did not have a concrete understanding of where he was when on a higher level node and, as a result, he was not sure how to deal with descriptions of bonds to and from it. For example, if he was located on the carboxylic acid of phenylalanine, he wanted to think of himself on a particular atom, probably the carbon, but it was not always clear which it should be.

3.2.3 Discussion

More interesting than the success or failure of this prototype is an analysis of which aspects of it succeeded and which failed, and the reasons why. The findings of this evaluation suggest that representing explicitly features that are implicit in the data reduces the effort required to understand the structures of these graphs. Not being restricted to either hierarchy-based or connection-based browsing also appeared beneficial, although the latter could perhaps have been made easier.

The results show that on the whole participants found it easier to explore molecules using the fully functional version of Kekulé than the basic version. Development of suitable strategies was, however, harder in the full mode, suggesting that more training or practice would result in significant improvements in performance. Similar improvements might also be achievable by improving the interface. This is unlikely to be the case for the basic mode and is perhaps the reason for four participants rating their performance as better in the basic mode, despite rating the actual demands as higher. The problems experienced also suggest deficiencies in the user interface, in particular the command structure.

Taken as a whole the findings of this evaluation suggest that exploring a graph non-visually shares some characteristics with spatial exploration and it is useful to allow browsing in both a hierarchical and connection-based manner, but that there are difficulties when doing this. A particular difficulty is gaining a sense of place when located on a group of atoms; without this it is difficult to make sense of its connections.

Domain knowledge was found to have a significant impact, to the extent that the naming of functional groups added to the confusion for non-experts, although expert users were more confused by vague or slightly inaccurate use of chemical
nomenclature. If formal names are to be used they must be correct. The numbering issues affected all, and finding an algorithm to re-number the atoms sensibly could help significantly. As with the naming, it is important that numbering follows the standards of the domain.

One of the greatest challenges for participants was simply keeping their current location in mind. The command system and navigation methods need to be designed to minimise confusion; reducing the cognitive load for this will free up resources for other aspects of the problem. In many respects this challenge parallels that being tackled by people investigating web accessibility. For example techniques such as ‘travel objects’ \cite{114} are likely to prove beneficial in this application; the web is, after all, a large graph.

Dealing with cyclic structures posed particular problems, and these were compounded if the cycles overlapped. When navigating from atom to atom, it was very difficult to determine if one was in a ring. Similarly it was difficult to navigate around the atoms of a ring without unknowingly leaving it. Enabling some form of bookmarking or annotation could help overcome this issue, while also facilitating the kind of labelling used by p-12 and allowing creation of ‘home bases’. Non-speech sounds also offer possibilities for providing this type of information.

Although this evaluation has used molecular structures as a test case, one may expect many of the findings to be transferable to graphs from other domains. A review of the results highlights some issues which need to be investigated if general graphs are to be explored non-visually:

- Effectively enabling both connection- and hierarchy-oriented browsing will allow a range of tasks to be accomplished.

- Techniques for making the implicit accessible have proven important in Kekulé and need to be generalised.

- Annotation by the reader offers possibilities for enabling computational offloading

- Orientation is difficult but might be tackled with techniques used for non-visual web browsing.
3.3 Description Experiments

During the evaluation described above, one or two participants found it slightly unnatural having what they envisaged as a paper-based diagram described using points of the compass. Some participants needed to translate the compass direction into a direction on the paper; a minor task, but nevertheless distracting when added to the already demanding requirements of building a mental model of the molecule. For this reason, it was decided to perform a series of experiments to capture the actual vocabulary people choose to use to describe diagrams. It was also hoped that the experiments would give some insight into the type of strategies people used to describe, explore and understand graphs.

Previously, similar work has been done as part of the TeDUB project\(^3\), where they observed how experts in different fields described diagrams (circuit diagrams, UML diagrams and architectural floor plans) to each other [100, 71]. The TeDUB experiments were designed to identify any protocols used, and differed from our experiments by allowing the listener to draw the diagram during the description. In our experiments this was not allowed, in the expectation that the requirement for holding the graph in memory might necessitate different strategies and possibly vocabulary.

Abstract graphs were chosen as they should give generic results, which although probably not ideal, should be applicable to any solution. The vocabulary captured from descriptions of abstract graphs should be usable as a basis for any specific application. It should be noted that the TeDUB experiments concluded that architectural floor plans were described using different protocols than the UML or circuit diagrams: the descriptions were given as if the reader was guiding the listener through the building. This difference is not of great concern, however, as architectural diagrams of the sort used are, strictly speaking, not graphs in the mathematical sense: the information contained in the diagram is held both in the relationships between its entities and their relative spatial layout. It is hoped that the vocabulary and strategies used in abstract graphs will be applicable to specific instances of more meaningful graphs.

\(^3\)http://www.tedub.org/
3.3. DESCRIPTION EXPERIMENTS

3.3.1 Method

Pairs of volunteers were asked to take turns describing graphs to each other, with one person taking the role of describer and the other of questioner (participants instructions are given in Appendix B). There were eight graphs, seven of which were abstract (i.e, unlabelled nodes and edges) — these are shown in Figure 3.4, while the final graph was semantically rich (Figure 3.5). The abstract graphs were taken from a variety of sources, including molecular structures, process models, and different UML diagrams, and stripped of any meaning. The final graph was a drastically reduced and slightly modified version of the London Underground map. The version used showed a selection of the stations from the central portion of the map with the different lines mostly, but not completely, corresponding to real tube lines. These differences were expected to hinder any participants with a good knowledge of the real system.

The participants were instructed to have a casual conversation so that the person without the graph could build a complete mental image of it, but with the conversation led by the questioner. In practice this did not last long and nearly all conversations turned to the describer leading the dialogue. They were also instructed that the spatial layout of the nodes and edges in the graphs were unimportant: communicating the connectivity was the aim of the exercise, but they were free to use the layout if it helped the description. They alternated so they described four graphs each. After the questioner was satisfied with their understanding, they were asked to draw the graph (they could not write, draw or make notes during their discussion). The conversations were recorded for later analysis.

Volunteers were recruited from the Computer Science Department at Manchester University. Although not all were native English speakers, they were all fluent in the language. Pairs two and three comprised one male and one female, pairs one and four both had two male participants.

3.3.2 Results

The simpler diagrams (Figure 3.4(a) – (d)) were all completed fairly easily, although there were occasionally minor errors. The more complex graphs (Figure 3.4(e) and (f)) were also generally completed correctly, although they took more time, and minor errors were more common. The participants appeared to find
3.3. DESCRIPTION EXPERIMENTS

graph 3.4(g) significantly more difficult than the previous ones (this is supported by timings and error rates for the graphs).

There are many methods for determining the complexity of graphs, for example the Augmented Valence Complexity index [76] was designed for determining molecular complexity. This calculates the complexity by summing for each independent node (i.e., ignoring those that are identical by symmetry) the number of nodes it is connected to, the total number of nodes these are connected to, etc., with each level of separation, $n$, weighted by $\frac{1}{2^n}$. The complexities for the abstract graphs calculated using this metric are given in Table 3.2 below. This table also gives the complexity per node, i.e., the total complexity divided by the number of nodes; this is an indicator of the connectivity of the graph.

<table>
<thead>
<tr>
<th>Graph</th>
<th>Nodes</th>
<th>Complexity</th>
<th>Complexity / node</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>6</td>
<td>21.94</td>
<td>3.35</td>
</tr>
<tr>
<td>b</td>
<td>6</td>
<td>28.75</td>
<td>4.79</td>
</tr>
<tr>
<td>c</td>
<td>8</td>
<td>74.94</td>
<td>9.37</td>
</tr>
<tr>
<td>d</td>
<td>9</td>
<td>55.22</td>
<td>6.14</td>
</tr>
<tr>
<td>e</td>
<td>22</td>
<td>215.73</td>
<td>9.81</td>
</tr>
<tr>
<td>f</td>
<td>20</td>
<td>206.87</td>
<td>10.34</td>
</tr>
<tr>
<td>g</td>
<td>13</td>
<td>162.84</td>
<td>12.53</td>
</tr>
</tbody>
</table>

Table 3.2: Complexities for the abstract graphs.

It is interesting to compare these complexities to the difficulty perceived by the participants. As was noted, graph 3.4(g) was considered the most difficult and resulted in the most mistakes; although the complexity score for this graph is considerably lower than graphs 3.4(e) and (f), it has a higher connectivity, hence higher complexity per node. Participants commented that it was difficult to spot obvious features on which to base a description. It is possible that the complexity seen by the participants was at least partly derived from the layout of the diagram as presented to them; transformation could lead to a graph that is much simpler to describe. It should also be noted that near symmetry, as seen for example in graph 3.4(f) can be used to ease description, but does not have an impact on this complexity metric.

The meaningful graph, Figure 3.5, was not successfully communicated as a whole, although some pairs could recreate the structure without the names. One pair was able to recreate the entire graph with only two station names omitted and one misplaced — this participant was familiar with most of the station names.
on the map, and also claimed to have a much better verbal than spatial memory.

The vocabulary used was on the whole rather simple, and relatively consistent across the pairs. The discussions also proved interesting from a strategic point of view. The vocabulary is described next, and is followed by observations of the types of strategy used.

3.3.2.1 Vocabulary

Despite instructions that only connectivity was important, the vocabulary used to describe the graphs nearly always contained spatial terms; indeed it was rare that the layout of the graph was not described. Absolute locations were given using ‘top’, ‘bottom’, ‘left’ and ‘right’. There was more variety in the vocabulary used when describing direction, although it was most common to use ‘up’, ‘down’, ‘left’ and ‘right’, or combinations thereof. Take as an example the hexagon of graph 3.4(d), where it was necessary to understand the relative locations of the attachments. Between the four pairs, three different methods of description were used:

Pair 1. Described the hexagon as 2,2,2 rather than 1,2,2,1, (i.e., with an edge along the top, rather than a node) then said:

   “the right middle node — there’s one coming off of that; and there’s also the bottom left node — there’s a chain coming off of that, of two other nodes”.

Pair 2. The second pair used a simple numbering system. Having first established the orientation of the hexagon, they assigned the top right node as number 1, then agreed on clockwise numbering (this was strictly unnecessary to describe the connectivity). Having done this the remaining nodes were described as connected to numbers 2 and 4.

Pair 3. Used the same method as the first pair, although the orientation differed.

Pair 4. These described the connections in relation to points on the compass; the single node coming off at North, then the chain of 2 nodes comes off at South East.

It would also be possible to use the clock (pair 3 used this system describing the connections to hexagons in graph 3.4(f)), e.g., a single node connected at 12 o’clock and a chain connected at 4 o’clock.
3.3. DESCRIPTION EXPERIMENTS

When moving from the initial simple graphs to the more complex ones encountered later, it was evident that the vocabulary used by each pair of participants evolved. The development of the vocabulary occurred in the sense that the describers were initially unsure how to describe these abstract graphs (this is, after all, not an everyday task), but the tactics and vocabulary developed quite rapidly, and after just a couple of graphs, the pairs seemed to settle in to a ‘system’ which they continued to use for all the more complex abstract graphs. In terms of vocabulary, it was observed that a certain phrase would sometimes be unclear when first used, so the listener would need to ask for clarification, but this same phrase could then be used for the rest of the session. An example of this was the description of the left part of graph 3.4(g) as a rhombus; this shape was explained as a skewed square.

Nodes were often labelled, normally implicitly, but sometimes artificial names or numbers were agreed. This form of annotation will be discussed more in the section about strategy.

Similes were occasionally used, mostly referring to graph 3.4(f), which was described by two pairs as resembling honeycomb, and another as being similar to how a football is stitched together. In these cases it was typical for the describer to use the simile as an overview, e.g., they would discover if their partner was familiar with honeycomb (or a football), then ask that they bear that concept in mind during the more detailed description. Graph 3.4(e) also provided scope for similes with one pair referring to ‘house shapes’ and another to wine glasses. The third pair described the uppermost node in 3.4(f) as being like a tent pole; this description was then used as a label when referring back to that node.

By far the most common method of describing the graphs, particularly for those of medium complexity, was to use the shapes formed by closed loops of nodes and arcs, such as squares, triangles and hexagons. Pair 2 even identified a pattern in graph 3.4(f) as being an rhombus (albeit with an incomplete perimeter); this was a successful tactic.

It was observed that participants generally used a vocabulary that was familiar to them. For example, those familiar with graphs used terminology like directed and cyclic to describe the graphs. In some instances the exact meaning of the phrases used was not initially clear to both people; in these cases the term was clarified or abandoned. They also used phrases like hub (to describe a central feature of the graph — this is discussed further in the strategy section). Those
3.3. DESCRIPTION EXPERIMENTS

familiar with hierarchies used the parent/child terminology of this field when possible. For example graph 3.4(a) was described as a hierarchy — the first node has 2 children, the first of which (on the left) also has 2 children and the second of which (on the right) has one.

The vocabulary used on the tube map (Figure 3.5) was similar to that used on the abstract graphs. Some pairs called the nodes and arcs stations and lines respectively, while others continued using the abstract notation of nodes and arcs/edges.

3.3.2.2 Annotation

Most pairs used some form of labelling when describing their graphs, although often this was implicit. In these cases the describer would typically describe a feature, then use a shortened version of that description when referring back to the item. For example, in graph 3.4(c) the two diamond shapes were called ‘the diamond’ and ‘the other diamond’. Similarly the hexagons in graph 3.4(f) became the middle hexagon, top hexagon and bottom hexagon (or top, left and right if the graph was oriented differently). These semi-implicit labels were often, as in these examples, descriptive of the feature or its position in the graph. As described previously, the uppermost node in 3.4(g) became ‘the tent pole’ for one pair.

Labelling with numbers was less common, although numbering was used for cycles, and pair 4 labelled the four nodes in the horizontal chain of graph 3.4(g) as 0, 1, 2, and 3 (these are computer scientists!). A scan of their sketch is shown in Figure 3.6. Three of the graphs drawn by participants have some nodes annotated by their numbers. When used, these proved effective tactics — when the participant in this pair drew his model of the graph he labelled these four nodes in his sketch. Numbering was also introduced to clarify the connections in graph 3.4(b) after pair 1 had become confused by the triangles.

The semantically rich graph is also interesting from the annotation point of view, as it may be seen as a ready-labelled graph. In general the participants felt that the labels added extra information that would be impossible to remember, and therefore attempted merely to understand the structure of the diagram in the same way as the abstract graphs. Despite this, nearly all participants ended up using the names of the most important stations, i.e., the hubs of Piccadilly Circus and Oxford Circus. The names of the stations on the circle were typically
replaced with numbers 1 – 11 (or 0 – 10 in the case of pair 4). Pair 2, however, seemed to remember the sequence of stations on the circle by a combination of repeated listing and numbering (from Kings Cross).

### 3.3.2.3 Spatial Descriptions

It was noted above that the vocabulary used to describe these abstract graphs contained many spatial terms, and that the descriptions tended to reflect quite strongly the actual layout of the graph. Most of the graphs drawn closely resembled the layout of the original, with major differences more commonly found on the simpler graphs (it is hypothesised that this was because very simple graphs, such as Figure 3.4(a) could be transformed by the describer to make it easier to describe, but this was too difficult for more complex graphs). The initial instruction that spatial layout was unimportant meant that some pairs gave initial descriptions in very abstract terms, but soon discovered that describing the layout of the nodes was often simpler. Directions were nearly always abstract, not personalised, for example ‘hexagon A is connected to hexagon B’, or ‘that arc goes to the tent pole’, rather than ‘if you go from the top of the V’.

Most graphs were described with reference to the layout in which they were presented to the describer. Occasionally the describer would transform parts of the graph to ease description. Perhaps the most common sub-graph transformation was that of the loop of three nodes sprouting from the middle hexagon of graph 3.4(f). In the graph presented to the participants, this was shown with one node inside the middle hexagon, and two outside. The participants sometimes described this layout, but more commonly described the loop of three nodes abstractly (i.e., not describing their position, or saying ‘it doesn’t matter’) or as being, for example, a line of three to the left of the hexagons. Transformations of the entire graph only occurred for the simplest graph; in this case the fourth pair described graph 3.4(a) as a chain of 4 nodes, with a fork off the fourth one. Other pairs described this starting with the top node.

### 3.3.2.4 Description Sequence

In most cases the first item described was a central feature — the graph was not described linearly (it was rare that there was any sense of travel through the graph); instead a core was described, then further sets of nodes and arcs added to complete the description. In some instances this was prompted by the questioner
asking for central or important features, in others the describer identified one. The participants in the pilot study got into the habit of asking if there were any hubs. The describers took some time considering the graph before starting the description or giving the central features, indicating that these are difficult to identify for some of the more complex graphs.

For example, graph 3.4(f) was described starting with the central hexagon, then the two flanking hexagons, followed by the extra node forming the pentagon. The two single nodes were described next, finishing with the loop of three nodes on the central hexagon. This pattern of description was identical for all four pairs.

Graph 3.4(g) was typically described around the central chain of four nodes, although the pilot pair used the diamond at its end as a starting point, and pair 1 identified a rhombus formed by the nodes on the left. Graph 3.4(e) was subtly different from the rest of the graphs due to the repetition of a pattern. Most pairs ended up describing this pattern then explaining how it repeated, finishing with the extra nodes on the right hand side. Pair 2, however, described it in terms of horizontal layers, again finishing with the right hand nodes.

In the tube map (Figure 3.5) the descriptions all started by introducing the circular line. Kings Cross, at the top, was considered the first station for all pairs.

### 3.3.2.5 Speed and Memory

The time taken for the entire exercise varied from 45 minutes to 90 minutes. This seemed to be largely based on how careful the listener was to check assumptions and describe his model for checking. Some participants felt it was necessary to do the description rapidly to avoid losing information from memory, and it was noticeable that once they felt they understood the graph they tried to draw it quickly.

Some of the errors were clearly due to memory failures — an accurate description of some part of the graph had been given, and repeated back to the describer, but then drawn incorrectly. In these cases the participant trying to understand the graph often realised his mistake on seeing the original. There were also one or two instances of people drawing an incomplete graph, but when prompted (‘is there anything else?’) remembering the remainder. One example of this was the two single nodes of graph 3.4(f).

Interestingly, one participant thought he had an understanding of graph 3.4(b)
until he started to draw it. At this point he realised that he did not have a clear picture, and needed to continue the questioning. This is suggestive of a model-based mental representation, rather than building up a mental image.

### 3.3.2.6 Control of the conversation

As was mentioned in the description of the method for these experiments, the intention was for the person without the graph to ask questions, using the answers to build up a model or image of the graph. In actual fact, the conversations rapidly changed so that the describer was leading, with the other participant listening and interjecting when something was not clear. This extended to the describer ignoring the instructions of their partner to give details about a certain feature, instead telling them not to worry about that part until later. This shift in control of the conversation was quite rapid, even after reminders were given that the describer should mainly be responding.

### 3.3.2.7 Summarisation

The initial questions asked often related to some form of summarisation, either in terms of giving an idea of the size of the graph (asking for the number of nodes and edges) or its important features. As described above, similes were sometimes used to give a rough impression of the type of structure to expect.

### 3.3.3 Discussion

The use of geometric shapes when describing the graphs can probably be attributed to attempts at chunking. It is well known that short-term memory can only hold approximately $7 \pm 2$ items [58], and participants stated several times that it was difficult to remember everything. Since even the simplest graphs in this experiment had six nodes, remembering the nodes and their connections must require either chunking or use of long-term memory, or both. Considering graph 3.4(d) we can imagine this could be chunked into a hexagon, a chain of two nodes, and a single node; remembering also the positions of both attachments to the hexagon (although strictly only a single relative position is required) gives a total of 5 items to remember — considerably less than 9 nodes and 9 arcs individually.
3.3. DESCRIPTION EXPERIMENTS

The descriptive nature of the labels was noted — it is possible that repetition of a label that also describes a feature or its location might enable use of long-term memory, thereby reducing demands on short term memory. We can also note that, particularly in graph 3.4(g), where there were less obvious features, that labelling was only applied to important nodes, i.e., ones that were used as a reference point; these appear to take the role of landmarks. This is repeated in the labelled graph, where the labels were generally only given for these landmark nodes.

The lack of use of the first person in these descriptions (e.g., “if I move to . . .”) contrasts to the Kekulé evaluation. In the latter, people appeared to envisage themselves moving around the diagram, as evidenced by questions such as “Where am I?” when they got lost, as well as describing the graphs in the abstract (e.g., “so there’s a methyl, a carboxylic, . . .”). It is not clear whether this is due to the nature of the task or the interface, or indeed the nature of the participants.

It was mentioned in the introduction to the results that graph 3.4(g) was considered hardest despite having a lower complexity than the two preceding graphs, and that this might result from the layout. Figure 3.7 shows the original graph (as presented to the participants) together with two possible transformations. The nodes are given arbitrary labels to make these transformations clearer.

The difficulties in the original graph appeared to lie in the difficulty of identifying significant features to describe. We may therefore suppose that the problems arise from difficulty in chunking, increasing the number of items to remember beyond the memory capacity. Palmer [68] suggested that abstract patterns are chunked into a hierarchy as part of the process of visual perception. Although it should be possible for those viewing the graphs to chunk in less obvious ways, this is effectively a form of transformation.

It is possible that it might be easier for a describer to identify large-scale features, and therefore chunk the information, in one of these transformed graphs than the original, the first transformation in particular appears simpler, grouping eight of the nodes into a single line. The high connectivity of this graph, however, means there are still large numbers of arcs to remember. It should also be noted that graph transformation has the following problems associated with it:

1. In the circumstances of the experiment, it would not be possible for the
3.4 SUMMARY

describer to make these transformations mentally, and then keep the transformed graph accessible to describe it correctly or answer questions about it.

2. The layout of a semantically rich graph is often designed to ease comprehension; this would be impaired by transformation, even if the meaning remains unchanged. For example, graph 3.4(g) was originally a process model diagram and oriented vertically; in this state, progressing down the vertical axis represented progress through a series of states.

3. Readers understanding transformed graphs may have difficulty discussing them with other readers who have only seen the original.

3.3.4 Conclusions

The vocabulary used was generally very simple, usually relying on up, down, left, right, top and bottom to describe the location of nodes or groups of nodes. Annotation was used by all participants in the form of labelling nodes or groups of nodes, although it was often implicit. The spatial layout of the nodes and arcs was used in all but the simplest of graphs to describe the connectivity. Nodes and arcs were chunked into larger units, generally the geometric shapes they formed, presumably in an effort to reduce the number of items to remember. Performance appeared limited by memory: those graphs where there were large numbers of nodes or arcs, and chunking was not possible proved the most difficult to communicate. The ability to chunk allowed graphs with higher complexity scores to be communicated more easily. Graphs were often described starting with a central feature before progressively describing other sets of nodes and arcs attached to it.

3.4 Summary

Formative experiments have highlighted some of the issues involved with exploring graphs non-visually. The evaluation of Kekulé demonstrated that enabling hierarchical browsing as well as connection-based browsing eased the tasks. This is thought to be mainly due to functional groups being made explicit and chunking the molecules to reduce memory demands. Difficulties were observed with the interface as well as some more interesting areas. In particular, the numbering
scheme used to identify nodes was confusing and cycles in the graphs were problematic. Possibly the greatest problem faced by participants was understanding their current position within the graph, in other words orientation. This suggests that the spatial metaphor is accurate, or at least useful: many of the problems that occur when travelling around a new environment were observed when participants explored graphs.

Although disorientation was possibly exacerbated by the interface design, there is no doubt that it was still an issue. It was observed that participants developed strategies such as the ‘home node’ to help overcome disorientation: this was a distinctive node (analogous to a ‘landmark’) that was used as a base for exploration that was returned to when lost. Allowing annotation such as bookmarking could simplify this process. Improving node differentiation, e.g., with an effective numbering system, or by allowing users to label nodes, could help the general problems of disorientation as well as making some nodes more suitable as landmarks — thereby increasing the number of candidates for home node in homogeneous graphs. Enabling users to add other forms of note to the graph might also allow some computational offloading.

The description experiments indicated firstly that, for abstract graphs at least, people find it easiest to describe graphs using the layout to chunk nodes into easily described (shape-wise) groups. These shapes were explicit to the describer, and clearly required less effort to describe as a whole than as their constituents. These groups, and some other nodes, were given labels, usually implicitly, which often described the chunk. Descriptions were abstract (‘there is . . . ’ rather than ‘you go from . . . ’) and started from a central node or chunk and worked outwards.

Finally, these experiments demonstrate the necessity for grouping nodes, both as a means of reducing the memory load and of simplifying description by making features explicit: in both cases chunks of the graph were identified and labelled by users. This activity can be characterised as annotation — although unable to create traditional written notes on the graph, the information conveyed could be clearly and concisely expressed in that form. It is also possible to identify some of the other activities as essentially being mental annotations: numbering is annotating; nominating a home node is annotating; naming shapes is annotating; making features of the graph explicit is annotating. If annotation offers a method for helping users explore these graphs, the questions become what forms of annotation are required and how can they be interacted with?
Figure 3.4: The seven abstract graphs used for the experiment. The diagrams were presented in the order shown, although not necessarily with the same orientation.
3.4. SUMMARY

Figure 3.5: The semantically rich graph used for the experiment. This was presented last.

Figure 3.6: Scan of the sketch drawn by pair 4 of graph 3.4(g).
3.4. SUMMARY

Figure 3.7: The original graph (g) and two transformations of the same.
Chapter 4

Annotation

Analysis of the literature has identified some of the difficulties that might be expected by someone exploring a graph non-visually. This has been supplemented by observation of people using the Kekulé system and describing graphs to each other. These initial investigations have shown that non-visual graph browsing shares many characteristics with travel, and have indicated that annotation may be a technique that could ease some of the associated problems. Determining if this is, in fact, the case is the key research question of this thesis. The first step in answering this question is to map potential forms of annotation to the various problems we know or suspect users will encounter.

The relevant definition of annotation [103] is simple:

**annotation noun.**

1 A note by way of explanation or comment

Supplementing information with annotations is a common behaviour. For example, comments in computer code are a form of annotation that (if done well) can significantly affect the ease with which the code may be understood. More traditionally, Marshall [54, 55] examined how university students annotated text books, noting how some annotations had clear benefits, both to the annotator and to future readers. Annotation is not restricted to text: diagrams may also be annotated. If annotated text books can be useful to future readers, why not diagrams also? And if annotation augments diagrams, can suitable notes be added to graphs that will ease some of the problems associated with exploring them non-visually?
This chapter surveys some of the different types and uses of annotation, before examining some possibilities for annotating graphs. Each possibility is described within the context of the problems of browsing identified over the previous two chapters. Finally, two orthogonal methods for classifying annotations are introduced and a taxonomy of notes is presented.

4.1 Annotation in Interfaces

Annotation is not new — it has certainly been used since ancient Egyptian times [31] and probably since the development of writing. Traditionally, it has been thought of as handwritten notes in the margin of a book; figure 4.1 gives an example. Types of annotation one might find there include:

- Correcting errors.
- Highlighting important passages.
- Re-phrasing confusing sentences.
- Referencing other explanations.
- Suggesting changes.

Marshall considers annotation in a hypertext environment [55], suggesting the following set of dimensions to reflect the forms these annotations may take:

**Formal vs. informal** For example, marginalia are informal, compared to metadata that follows a defined structure.

**Explicit vs. tacit** Annotations for a reader’s own benefit are often brief and difficult for others to interpret, while those intended for others tend to be more explicit in their meaning.

**As reading vs. writing** Annotations form a continuum, between simply reading aids, to substantial contributions.

**Hyperextensive vs. extensive vs. intensive** For example, link-oriented annotations (hyperextensive) connect documents while intensive annotations relate to a deep reading of a single source.
4.1. ANNOTATION IN INTERFACES

Figure 4.1: Examples of annotation in a text book. The author of the notes has highlighted key phrases (underlined) and bookmarked portions of text with a word (‘innateness’ or ‘probs’) that summarises what the portion is about, and allows it to be found again easily.

**Permanent vs. transient** Some annotations only hold benefit for a single reading of a text, while others may also benefit future readers.

**Published vs. private** Some annotations are private (or, at least, intended to be so), while some are for the public, e.g., annotated editions of important scholarly works.

**Global vs. institutional vs. workgroup vs. personal** Some annotations are intended only for the reader, while others may be targeted towards larger groups.

Wolfe [110] explored the power of annotations, looking at how readers were influenced by them. This was achieved by studying persuasive essays written by undergraduates receiving primary source materials annotated in various ways.
This study indicated that annotations “improve recall of emphasized items, influence how specific arguments in the source materials are perceived, decrease students’ tendencies to unnecessarily summarize”. It was also found that “students’ perceptions of the annotator appeared to greatly influence how they responded to the annotated material”. It was thought that students considered the annotator as a potential reader of their essay, and thus, if they perceived the annotator as agreeing with their point of view, less argument was used in the essay.

But ‘a note by way of explanation or comment’ is a wide definition, encompassing much more than just marginalia considered by Wolfe, or even the hypertext annotation explored by Marshall. For example, annotation is also used in many computer applications and on the web. The popular photo sharing site Flickr\(^1\) and bookmark sharing site del.icio.us\(^2\) both use tagging. A tag is simply an annotation given to a bookmark or photograph that describes or classifies it in some way. For example, a skiing photograph might be tagged with the name of the resort, the name of the skier, and other words such as ‘skiing’, ‘snow’, and ‘mountain’. Multiple tags allow an item to belong to multiple classifications and searches can be performed with multiple tags to allow efficient selection of interesting items. Since the tags are given by users, efficiency is limited by the quality of tagging and by ambiguities in the meanings of many words.

Another approach to annotation is to use notes taken from a well-defined set, such as an ontology. An example of this is genome annotation, where segments of a protein sequence (e.g., DNA, RNA) are annotated with their function. Stein’s review paper on this subject from 2001 [93] gives an introduction to this field.

According to Marshall [55] annotation is a ‘fundamental part of hypertext’; nowadays it is seen as critical to the semantic web — the next stage of the biggest hypertext resource, the world wide web. The ‘semantic web’ is a term that captures the goal of many researchers [9] — a world wide web where the meaning of web content is not only human-readable, but is also computer-processable. The idea is that intelligent services will be able to process web content not only in terms of content and layout, but also in terms of meaning, and will therefore provide greater functionality than at present. Critical to the whole semantic web vision is metadata, that is information about the content. This is generally accepted as being supplied by annotation, in this case with the notes being taken

\(^1\)http://www.flickr.com/
\(^2\)http://del.icio.us/
from an ontology so that their meaning is clear [30].

In addition to the uses outlined above (and many more), annotation has been used for applications more closely related to non-visual graph browsing. Several groups have looked at how website accessibility can be improved by identifying and annotating certain features [2, 97, 112]. Yesilada [114] applied the travel metaphor to web browsing, and identified and classified travel objects on a page; these are ‘environmental elements that are used during a journey’. For example, a menu was classified as a decision point, as it is a point where alternative paths are possible. The Danté project [112] continued this work, examining how travel objects on a page could be identified and annotated (so that they become explicit), then using these annotations to transform the page in order to improve its accessibility. Evaluation demonstrated that this approach could improve the experience of visually impaired web travellers.

A difficult part of the application of a Danté approach is the automatic identification of the travel objects. One solution is to generate the annotations during the design phase of the website, ideally as an automatic part of the process. This is the technique described by Plessers et al. [75], who integrated the WAfA (Web Authoring for Accessibility) ontology used by Danté into the WSDM (Web Site Design Method) web design method. This automatic approach could currently generate approximately 70% of annotations; this could potentially be extended to 85% without manual input, while full annotation could be achieved with some effort from the designer. A significant problem with annotation during design is that the accessibility of existing websites is not improved unless they are retrospectively annotated. Nevertheless, this research demonstrates the power of simply annotating information, and how this can improve accessibility.

Another use of annotation in accessibility is Wall and Brewster’s use of ‘beacons’ for people exploring bar charts through a haptic force feedback device [109]. They argued that, when blind people interact with visualisations, ‘there is no easy way to mark points of interest or to access external memory’ (see section 2.2 for a discussion of external memory). They therefore implemented and evaluated a system that allowed people to add multimodal beacons at points of interest on a bar chart. These beacons were represented by non-speech audio MIDI percussion sounds, differentiated by timbre, and, upon issuing a command, the user could be dragged from their current position to a beacon. It was noticed that participants found these ‘bookmarks’ (this term was not used by Wall and Brewster, but it
is clear that this is the essential function of these beacons) were most useful for comparing two bars on a chart that were distant from one another. Although they were unable to do a quantitative analysis, and some problems with the implementation became apparent during the evaluation, this is an interesting application of annotation, and participants’ comments suggested it could be helpful.

Can a similar approach to that used in Danté be used for non-visual graph browsing? From the schematic diagram in figure 2.4, it can be seen that in general terms the role of annotation in such a system would include:

- Simplifying the process of building a mental model of the graph. This requires annotations that make moving around the graph easier, and will include notes making implicit features explicit.

- Enabling manipulation of the model. Annotations that make re-visiting and re-reading sections of the graph quick and those that allow it to act as an external memory will be important for this.

Working in the audio domain means that annotations cannot take their traditional form of marginal text: there is no margin (the reader is always focusing on a node, arc, or chunk of graph; the space between does not really exist), and the information must be presented as sound. The latter difference is not necessarily a difficulty; indeed it might be seen as a benefit — freed from the constraints of being represented on paper, non-visual annotations may take the form of speech or non-speech sounds, or some combination. The converse, however, is that without a margin, annotations (particularly speech-based ones) must take up a primary position in the information stream, perhaps temporarily preventing presentation of the main information and potentially acquiring a significance on a par with it. Neither is locating an annotation a significant problem, since traditional notes are implicitly (or sometimes explicitly) linked to some of the original information. In the case of non-visual graphs, it is necessary to allow annotations to be associated with nodes, arcs, the whole graph, and chunks of graph of intermediate size.

4.2 Annotating Graphs

If annotation has proven useful in the variety of applications overviewed above, what types of note can augment a graph so that the benefits of visual presentation
are recreated and the problems of aural presentation are minimised? This section introduces some forms of annotation, giving theoretical reasons why they should be useful. Some are merely introduced, being discussed in considerable detail in the following two chapters. The section concludes with an example graph annotated with some of these types of note.

4.2.1 Chunking

Chapter 2 described Larkin and Simon’s assessment of the benefits of representing information diagrammatically [51]. They concluded that one of the main reasons diagrams were effective was that information that might otherwise be implicit becomes explicit when presented in a diagram. One of their examples was a description of a geometry problem involving four lines, from which it could be inferred that these formed two triangles; this could be recognised without any inference in a diagram. If we are to allow visually impaired users to use diagrams effectively, then implicit information must be made explicit to them.

Recognition of implicit features necessarily involves groups of nodes; a note can be attached to each of these to highlight its membership of the group. As an example, consider the phenylalanine molecule. The structure of this molecule is shown in Figure 3.1, with some groups of atoms highlighted — these groups would be recognised by an experienced chemist. If the nodes comprising the groups are annotated as belonging to a group (e.g., part of carboxylic acid functional group), the user can be made aware when moving onto a node belonging to such a group, or be told what groups are present in the graph. The former should help as it will let the reader know what connections there are (assuming they are familiar with the group). The latter will allow the mental model to be constructed more easily, with larger chunks (hence fewer components). It would also enable the user interface to provide hierarchical and connection-based browsing.

An addition, or alternative, to making these features explicit with hierarchical browsing is to use ambient sounds to notify changes. Sounds could be played when exploring in a cyclic area, or in a functional group. The former would almost certainly have aided the participants in the Kekulé evaluation (chapter 3), who spent time moving round in circles unaware of their location. Presentation of this type of annotation is a user-interface issue, but ambient sounds can offer a less obtrusive form of presentation than speech.
4.2. ANNOTATING GRAPHS

4.2.2 Home Node

The formative evaluation of Kekulé, described in chapter 3, showed how important it was to allow users to jump back to the start if they became disoriented. In the hierarchical exploration, people tended to zoom back out so they were viewing the entire graph, while in the basic exploration, several participants used a distinctive node as a base for exploration, returning (if possible) to this node when confused. Although many of the different types of annotation described here are designed to minimise disorientation, given the difficulty of possible tasks it is still likely to occur.

Enabling any node to be nominated as a ‘home’ node, and annotated with this information, allows a user interface to provide this functionality more easily. Defaulting the home node to the starting node makes the ‘go home’ natural in that it returns to the start point; allowing it to be moved gives the user flexibility. For example if the start point is at the end of a chain, distant from the most important part of the graph, it would be tedious to traverse this chain after each reorientation.

4.2.3 Relationship

The essence of the information presented in a graph is not visual: a graph is simply a depiction of the relationships between entities, as represented by arcs and nodes respectively. Determining the relationship between two nodes, adjacent or distant, is a common task. For example one might need to find the bonds between two atoms in a molecule, how two people in a family tree are related, or sub-class and interface relationships in UML class diagrams.

While the direct relationships between nodes (where they are directly connected with a single arc) are explicit when presented both visually and non-visually, this is not the case with relationships between more distant nodes. Often these relationships are explicit when the graph is presented visually, or take little effort to deduce. For example, in a family tree it should be relatively simple to spot the uncle/nephew relationship by relative positions. This may be considered as a special case of how diagrammatic presentation enables recognition, as discussed by Larkin and Simon.

When exploring a graph non-visually, determining relationships becomes more and more difficult as the distance increases. Discovering them requires detailed
exploration of the graph to find both nodes and the path between them, and to remember the details of the path so that the relationship may be determined. It probably would be beneficial if these could be made explicit for non-visual readers.

Having a node annotated as a home node allows these distant relationships to be made explicit. It is possible to annotate all nodes with their relationship to the home node. Users would thereby be able to discover how their current node relates to their home node, wherever they are in the graph. The ability to change the home node allows the relationships between any two nodes to be made explicit relatively simply.

Landmarks will be discussed in more detail in chapter 6, but it is clear that annotations of relationship can allow the home node to act as a global landmark, with the relationship note allowing the user to orient themselves with reference to home.

4.2.4 User Notes

The benefits of external memory have been discussed previously: the existence of an external representation of the graph eases the memory demands, but also allows interaction. Manipulation of the model can be performed externally to help in various tasks, for example in a large graph, even a simple task such as counting the number of nodes can be eased by marking off each node as it is counted. There are countless other situations where marking, or annotating, the diagram can reduce the effort required to solve a problem; providing such a facility for non-visual browsers is likely to be essential.

The nature of user notes can vary widely, so it is important to allow flexibility. Two types of annotation are immediately obvious — labels and notes. Labels allow the user to give a node a new name, a type of note that should be readily accessible, e.g., presented automatically each time the node is encountered. These are therefore most likely to be short, used either to give a node a more distinct identity (it therefore becomes more salient as a landmark), or as part of a calculation. Notes could be longer annotations describing a node or arc (or the graph as a whole) in some way. If the user interface allows searching of annotations, user notes could perform the function of bookmarks, allowing users to move rapidly back to previously visited nodes.
4.2.5 Location

The evidence as to whether knowing the location of a node in the original representation aids understanding when browsing non-visually is unclear. Bennett [8] found that presenting positional information (as pairs of notes giving coordinates) was not useful to his participants. It is possible, however, that other methods for presenting positional information may prove useful. Some participants in the Kekulé evaluation appeared to find the position (presented as ‘top left’, ‘middle’, etc.) useful. Annotating each node with its location would allow a user interface to take this information and present it in any form (e.g., Cartesian or polar coordinates, presented as spoken numbers or with tones, or descriptions such as ‘top left’). Clearly this form of annotation is only possible if the original representation contains the information, a common situation where graphs have been generated using (visual) drawing software.

4.2.6 Direction

If node locations are known, arcs may be annotated with direction. For similar reasons to presenting locations, it is possible that this information will help the reader build a spatial mental model that resembles the original graph. A potential additional benefit is with navigation. It has been shown that landmarks lie at decision points [57], that is they indicate to (or remind) an explorer where to change direction. For example, ‘turn left after the town hall’. If the spatial exploration analogy is to be followed, we must therefore not only ensure the graph contains landmarks but that, at these landmarks, the options for travel are distinct; ‘turn after the town hall’ is useless. Differentiation of the nodes provides this to an extent (‘go to node X, then on to node A’ (not B or F)), but direction of travel can supplement this in an intuitive manner.

4.2.7 Deedle

An important concept in user interfaces is that irreversible actions are undesirable; this is discussed by Donald Norman in ‘The Psychology of Everyday Things’ [64]. Application of this concept to non-visual graph browsing suggests that it would be desirable to allow users to make explorations of the graph whilst easily being able to return to their earlier location. For example if a decision
point is reached, each option might be explored to a certain extent to find the one of most interest.

While annotating a node as home allows this to a certain extent, it may prove useful to allow a separate annotation specifically for this purpose. One option is to allow the user to enter a special mode\(^3\), at which point their current node is annotated as a start point; they would then be free to explore. We may call exploration of this type ‘deedling’\(^4\), and a different voice may be used to indicate that the user is in this mode. If they get lost, or simply decide that their exploration is not proving useful, they can exit deedling, at which point they will return to the start point. Equally however, if the exploration leads the user to an interesting part of the graph, they might wish to continue normal exploration, in which case deedling is left without returning to the start (and the start label is removed).

### 4.2.8 Visit Histories

As an addition to jumping back to a previously visited node such as the home node or the deedle start node, it is important that the path of exploration may be retraced in smaller steps. While there are several methods by which this may be achieved, some form of annotation could certainly enable this functionality. For example, the graph as a whole could be annotated with the path travelled, just as one may trace a path over a diagram using a pencil. Alternatively each node could be annotated with the time of each visit. This type of path travelled annotation could also be used to warn users if they are exploring a new part of the graph, or conversely if they have visited a node previously — potentially useful information when trying to ascertain the extent of a graph. Annotating graphs with this type of information is discussed in more detail in section 6.3.

### 4.2.9 Summaries

In addition to annotating individual nodes or arcs, as for example with relationships or locations, the entire graph, or groups of nodes, may also be annotated.\(^3\)\(^4\)

---

\(^3\)Note that this is somewhat different to a ‘normal’ node change, as all exploration may continue as normal and all commands continue to be available and work in the same manner. The modal nature simply allows the user to return rapidly to a previous state.

\(^4\)The term ‘deedling’ is a colloquialism coined by Dr. Pettifer to convey the notion of fidgeting around a space and looking at things; lots of small movements used to explore something.
For example there are several features of a graph that may benefit readers if made explicit. These include the number or nodes and arcs, the number of different types of node, and the complexity of the graph. Information about the graph as a whole is essentially summary information. Annotation with summaries is discussed in detail in chapter 5.

4.2.10 Node Identification

In visually presented graphs, nodes are differentiated by their location; on the whole no two nodes share the same location. It is therefore relatively easy to distinguish nodes. Further differentiation may be provided by other information on the graph, for example names or labels next to the nodes, the way in which a node is drawn (e.g., in logic circuits). Similarly any other information about a node will be instantly visible and may thereby help differentiation, for example if a node has a particularly long or short list of properties.

Differentiating nodes in a non-visual environment is more challenging. Location is not immediately apparent, and the other information can only be given with a significant time (and attention) cost; it is not desirable to present all the information about a node merely in the hope that this helps the reader distinguish it from other nodes — there would be information overload. The constraints of audio presentation mean that each node must be distinguishable with a short sound (speech or otherwise). Chapter 6 discusses the importance of differentiating nodes and explores the possibilities for using annotation to facilitate this.

4.2.11 Example

To exemplify some of these types of annotation, Figure 4.2 shows an annotated family tree. In this example, a home node is selected, and nodes are annotated with their number and relationship (to home). Nodes are also annotated with group membership — they are chunked into families. The graph as a whole is annotated with some summary information. Note that if this were drawn strictly as a graph the relationships would be between individuals, e.g., a child would have a relationship to each parent, rather than to the marriage. The graph is shown in the standard form for simplicity and clarity.

Some of the benefits of these annotations are clear, even when reading the
Figure 4.2: Examples of annotation in a graph. This family tree is annotated with a home node, relationships, chunks (family groupings), numbering, and a summary.
4.3 A TAXONOMY OF ANNOTATION

graph visually. Although the layout makes relationships reasonably simple to deduce, it is much easier when the annotation makes them explicit. The numbering can distinguish between people of the same name — not uncommon — and give a clue as to whereabouts on the tree the person is (very roughly, a lower number indicates they are nearer the top). A further, domain specific, annotation that could help in this regard (not shown) is to annotate each person with the number of their generation. Similarly the position note (also not shown) can give clues about position in the tree.

4.3 A Taxonomy of Annotation

The previous section presented a collection of different types of annotation which may help non-visual exploration of graphs. The forms, function and origins of these were wide and varied, from annotations with sounds that summarise the entire graph, to labels generated by the user, to temporary annotations enabling cost-free exploration. If the effectiveness, or otherwise, of these annotations is to be evaluated, tasks must be designed that will present a sufficiently wide range of challenges. Classifying the annotations should help this task.

This section introduces two ways in which annotations may be classified. The first classification considers the quality of the information, the second the difficulties the annotation are intended to ease.

4.3.1 Provenance

One attribute upon which annotations can be classified is the provenance, or quality, of the information. If information is being presented to the user, it is important that they know whether that information is true, i.e., explicit in the original representation, inferred, or estimated. This type of categorisation is useful, since it strongly reflects the method by which the annotation was generated.

We may fit annotations into four classes of provenance.

Original information That which is explicit in the base representation (e.g., CML).

Inferred information That which is inferred from the graph, for example detected features. This can be further broken down into subjective or objective. This category also includes some information which may be considered
fairly artificial or arbitrary, such as node numbers, or the initial home node.

**User-given** Information that is provided directly by the user.

**Recorded** Information that is collected about the user’s behaviour, e.g., breadcrumb trails.

Although these classes can be placed on a scale running from information that can be called true with some certainty (where ‘true’ is used in the sense of accurately reflecting the base information) to information with which the original author may not agree, this should not imply that this correlates with usefulness. Some of the less ‘truthful’ annotations are simply one of a set of choices (e.g., numbering systems) that, even if not optimum, should help exploration. Thus the distinction between inferred and artificial is rather blurred — e.g., the complexity of a graph can be calculated, but the choice of method is artificial, so is the value inferred or artificial?

### 4.3.2 Application

An alternative, or complementary, method of classifying annotations is to examine how they are supposed to benefit readers. Categorisation in this dimension should give a clearer picture of whether annotations are capable of helping with the range of difficulties users may encounter. Of course, as we have seen above, some forms of annotation can be beneficial for several different reasons; the classes used are therefore not exclusive.

Broadly speaking, the reader may encounter three main areas of difficulty, which become apparent when the problem is viewed using the spatial analogy. These are: summary, orientation, and relating. A further class is for annotations that help user tasks.

**Summary** Notes that give the reader an idea of the scope of the graph, as discussed in chapter 5.

**Orientation** “The action or process of ascertaining one’s bearings or relative position, or of taking up a known bearing or position” [103]. At any point in the exploration it is important that users have a clear understanding of their location (i.e., they are where they think they are). This class of annotation includes notes that help prevent readers from becoming disorientated.
4.4. SUMMARY

Annotation is a long-standing method for augmenting information, traditionally as hand-written notes in the margin of a book, but more recently in a variety of applications. Annotation is used to organise bookmarks or photographs, to understand the human genome, and to help the visually impaired surf the web.

<table>
<thead>
<tr>
<th>Provenance</th>
<th>Annotation</th>
<th>Summary</th>
<th>Orientation</th>
<th>Relating</th>
<th>User tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>Node name</td>
<td>y</td>
<td>y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Node counts</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inferred</td>
<td>Chunking</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>Relationship</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Graph complexity</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Summary</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Node Numbers</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User-given</td>
<td>Note</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>Label</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>Home</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td>Recorded</td>
<td>Deedle start</td>
<td></td>
<td></td>
<td>y</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>Visit history</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1: A taxonomy of annotations.

Relating As discussed in section 2.4.3, the major problem in moving around this type of space is more one of building up a mental model of how the nodes are related than one of traditional navigation. These notes, therefore, are ones which are intended to assist explorers determine how nodes are related.

User task Notes which may directly help a user perform a task with the graph, rather than helping them read the graph.

4.3.3 Classification

The different types of annotation discussed may now be classified using the two schemes described above. Table 4.1 presents this classification. This classification makes more explicit how each type of annotation is expected to help, and should therefore inform both the design of any evaluation, and assessment of whether the annotations were beneficial.

4.4 Summary
Chapter 2 introduced some of the cognitive science behind the problems of non-visual graph exploration, while chapter 3 described experiments that showed how parts of graphs were implicitly labelled when being described. It also discussed the problems faced by users of Kekulé, and how annotation might help.

This chapter has introduced annotation and combined the findings of the previous two chapters to suggest a series of annotations that could make non-visual exploration of graphs less demanding. Each is founded with theory, either from the literature or from the results of the Kekulé evaluation. Two types of annotation (summaries and node identification) are discussed in more detail in the next two chapters.

Finally this chapter has introduced two methods for classifying annotations: provenance and application. The former considers the source of information for the note, while the latter considers how it is expected to benefit users. The annotations described are placed in this taxonomy — this classification should help in the design and analysis of an evaluation.

Having introduced the concept of annotating graphs in chapter 3, the idea has been widened and developed, with a range of types of notes described. Each of these notes is intended to help the task of non-visual graph browsing in some form or another, and the classification of these notes suggests that annotation is a tool that could help with all of the major difficulties that may be encountered.

Before evaluating these annotations, it is necessary to explore in more detail the use of annotation for two particular problems. First, can the whole graph be annotated with information that summarises it, thereby simplifying the task of building a mental model of the graph: this is covered in the following chapter. Second, how can annotation help the problem of node identification — crucial for orientation, and therefore both building and manipulating the mental model: this is discussed in Chapter 6.
Chapter 5

Graph Summarisation

The first stage in many everyday graph reading tasks, from planning a journey to reading, is a glance. A quick glance at a graph to gain an impression of its size, content, complexity, etc. takes very little time but gives the reader a broad understanding of the nature of the graph and consequently the nature of his task. In the simplest of problems this glance may give sufficient information for an understanding to be reached with little further effort; in more difficult cases it serves merely to provide context — the reader has a rough idea of the problem complexity and can start to develop strategies for reaching an understanding, including strategies for reading the graph in more detail. Furthermore, glances are often used for orientation once more detailed exploration has begun.

What if the user is unable to see the graph? It is technically possible to enable a visually disabled user to move between nodes in some application (for example [20, 21]), reading the details as they go and possibly building an understanding of the topology. This is a bottom up strategy; how does such a reader get the head start sighted readers gain from their glance? One might argue that the information given by a glance is more important for the non-seeing user; if he or she has an approximate understanding of a graph’s complexity and topology before commencing node to node exploration, he or she should be less likely to become disoriented and hopefully better able seek out the relevant parts of the graph. In essence, annotating a graph with summary information should help the user to build a mental model of the graph more easily, by constraining the possible structure (size, node types, etc., depending on the nature of the summary) to a small part of ‘graph-space’. This is illustrated in figure 5.1.

This chapter considers summaries as notes: these are one of the major classes
5.1 Literature Review

There is remarkably little in the literature about overviews, the bulk of relevant research, however, appears to be in the field of reading comprehension. For example, it has been shown that newspaper readers scan before selecting which parts to read in more detail, both when reading traditional newspapers and online publications [40]. We may assume that this scanning is a filtering process where the reader is selecting the items of most interest. Most relevant research interest in this area is related to education, where the benefits of previews have been demonstrated.

A preview is, to all intents and purposes, identical to an overview or summary, in that it is a condensed version of the information that gives its gist. For example, Graves et al. [35] looked at the effect of previewing short stories to seventh and eighth grade children who were reading about three to four grades
5.1. LITERATURE REVIEW

below average. They found that previewing the stories significantly improved the results of comprehension test:

“it appears that previews of short stories can produce large gains in comprehension and retention of factual information from stories and in students’ ability to make inferences related to the stories.”

Although these studies used low-ability children, Graves cites two other papers in support of his assertion that the findings are “robust with respect to readers’ ages and ability levels”. Graves also cites Stanovich’s interactive model of reading [91] in explanation of how previews should help. Stanovich’s model has high-level processes (e.g., domain knowledge) and low-level processes (e.g., letter recognition) working interactively such that “a deficit in any particular process will result in a greater reliance on other knowledge sources, regardless of their level in the processing hierarchy”. Under this model, semantic knowledge gained from a preview allows more processing capacity to be applied to obtaining information from other knowledge sources. Thus, to account for the particular improvement of inference after previews (38% improvement in inferential comprehension compared to 13% for factual, although this difference was not statistically significant) this model suggests that less effort is required to follow the central gist of the story, so more processing resources are available for generating inferences.

Neuman et al. [62] found that the benefits of preview were not limited to short stories. They tested the benefits of video previews on children’s (6 to 9 years old) comprehension of television stories. They found that children who had watched the preview had significantly increased comprehension of plot-essential information, but their recall of incidental information and their ability to draw inferences were not improved. A notable difference between these two studies is that the short stories used by Graves et al. were considered difficult, whereas the televised stories used in Neuman’s experiments were easily comprehended.

The benefits of overview in Human Computer Interaction (HCI) are intuitive and are extolled by many, but appear not to have been evaluated. One of the major proponents of overviews in the visual domain is Ben Shneiderman, whose visual information seeking mantra has already been introduced [85]:

“Overview first, zoom and filter, then details-on-demand”

His paper on Dynamic Queries for Visual Information Seeking [84] emphasises the benefit of seeing dynamically how changing search parameters affect
search results and gives many examples of visual overviews, although the word ‘overview’ is not actually used. Overviews are also considered important by the assistive technologies community — see section 2.5.1 for some examples. Further examples include web-page summarisation, where summaries of web pages are automatically generated to help visually impaired browsers determine their usefulness (i.e., ‘should I read this?’); see, for example, Harper’s ‘Summate’ [36]. Further examples of audio summaries are discussed in more detail in section 5.5.

In all, the literature on summaries suggests that there are benefits in giving readers (including non-visual readers of graphs) a brief overview that gives the gist of the information they are about to explore more deeply. If effective, this information could free some processing resources for other activities, such as the task for which the reading is being done.

Before proposing some annotations that can give summary information about graphs, it is worthwhile considering what we require from these notes. What attributes are necessary for an effective summary?

- Summary notes must be short, both relative to the size of the graph and in absolute terms – the information needs to summarise, not just rephrase.
- Summary notes must be accessible, that is read at any point. While the previews described above were given in advance, having the summary accessible from anywhere on the graph, at any point in the exploration allows further benefits. First, it allows the memory to be refreshed in the absence of any external representation; second, some types of summary may be used to aid orientation.
- Summaries must be predictable; i.e., for two similar graphs, the summaries must be similar.

### 5.2 Complexities

One piece of information that the graph as a whole may be annotated with, is a measure of its complexity. Although this may be discerned to a certain extent from the audio glance (see section 5.5), it may be useful to the reader to quantify this somehow. Knowledge that the graph is of complexity 20 rather than, say, 28, may influence whether the user decides to explore further. If further exploration is undertaken, this knowledge may determine the nature of
the reading strategies used. The usefulness of a complexity measure is extended in a hierarchical browsing environment, where the same measure may be used to describe the complexity of subgraphs.

The difficulty is, of course, finding an appropriate method for calculating and describing the complexity of a graph. What makes a graph complex? Do we want to measure the intrinsic complexity of the graph (if there is such a thing) or predict how difficult it will be to explore? Do the two correlate? Will a measure of complexity be generic, or will domain-specific metrics be required? These questions are, of course, interdependent.

The simplest solution is a generic measure of the graphs intrinsic complexity. As was noted in the experiments of Chapter 3, connectivity had a significant impact on how participants in the description experiments performed. Connectivity was also the basis for a complexity measure introduced in that chapter (section 3.3.2) — Randić and Plavšić’s Augmented Valence Complexity index [76]. Although the key question of whether this measure predicts exploration difficulty has not been explored, a quick evaluation was performed to test if some different variations on their measure correlated with peoples perception of complexity when presented with a visual diagram.

As part of the evaluation of audio summaries, described above, participants were given ten multiple choice questions. For each one they were asked to select which of four graphs (presented as diagrams) they thought the sound represented. Additionally, they were asked to identify which graph of the four they thought was the most complex and which the least complex. These selections were recorded for comparison with different complexity metrics.

The results showed that subjective judgements about complexity approximately agreed with Randić and Plavšić’s metric (note that the connectivity and layout of the graphs were similar to what might be found in molecular structure diagrams, for which Randić and Plavšić designed their metric. Using the metric as described agrees with the participants choice of most complex graph in nine of the ten graphs. Interestingly, the choice of least complex graph was best predicted by a variation on their metric that essentially calculates the mean connectivity per node (i.e., follow the first stages of their algorithm, but instead of summing the values for all distinct nodes, calculate a mean for all nodes): this gives agreement with participants choices in eight of ten graphs, compared to six from ten for the complete metric.
5.3 Descriptive Summaries

The concept of the overview need not be restricted to a brief, high-level glance (as described in section 5.5); in fact, overviews may be useful at different levels of detail. Although the overviews described in section 5.1 were of stories, articles or more numeric data, it is still possible to identify several benefits of overviews for information structured as a graph; indeed, many of the benefits translate directly.

Perhaps the first use is to give the reader a better idea whether it is worth exploring further. In section 2.1, several possible reasons for the creation of a graph were given, and it was noted that sometimes the graph would be of no interest to the reader. A suitable overview is one means for readers to assess the value of further exploration. Of course, since the requirements of the reader are unknown, and the overview needs to be brief, this will not always be achievable. Integrating whole-graph overviews into an interface which allows exploration at different levels of detail (such as the hierarchical data structure described in Chapter 2) should, however, meet most needs.

What other information about the graph as a whole could be useful? In addition to informing the reader about the value of exploration, overviews can set the scene for it, giving them information that will allow them to construct the mental model more efficiently. For example, it is proposed that overviews could:

- Give a more concrete idea of size (e.g., number of nodes and arcs)
- Give an idea of the complexity of the graph.
- List the types of components, e.g., 4 nodes of type X, 3 of type Y, etc. This type of summary could also allow direct access to these nodes, fulfilling a kind of ‘overview and zoom’ function.
- Hint at the type of structural features that may be encountered (e.g., if there are rings or other features that may cause confusion).

Part of the function of overviews is to inform about the value of the graph, and part is to give the reader information as a basis for building a mental model. Overviews can also start to give some of the implicit information about the diagram explicitly. For example, the components list may be a list of groups of nodes; this essentially groups the information into recognisable chunks, allowing much more to be conveyed quickly. Understanding the presence of these implicit
features at the overview stage could facilitate the value judgements discussed above and help simplify the problem space (see below).

How does this overview actually benefit the reader? Imagine a chemistry problem about the reactivity of a molecule — if the reader knows that there are only two oxygen atoms in the molecule, only the areas including these may need to be explored. The task would be simplified further if the overview allowed direct access to these two nodes. Annotating the graph with an overview has reduced the extent of the graph that needs exploration — browsing can be more selective, hence more efficient and less demanding.

It is important to compare whole-graph overview annotations with the hierarchical ‘part-of’ node annotations. The latter is a method of making features of the graph explicit, but when used to allow a hierarchical zoomable interface, it essentially becomes a form of local overview, serving many of the functions described above, and reducing mental load by allowing parts of the graph to be safely ignored.

Another function that an overview might fulfil is as a warning. An overview may alert the reader to potential difficulties, allowing strategies to be considered before they are encountered. For example, many nodes might share the same name, the graph may be particularly complex, or it may contain features such as rings (or even fused rings) which can cause disorientation and confusion. Prior knowledge of such difficulties should at least help users understand why they have become confused, even if that confusion is not avoided.

Clearly some aspects of the overview (e.g., size, complexity) are generic, while others will require a domain-specific implementation. This is the case for component lists, where the domain will define how nodes are grouped. To give some examples: in molecules nodes could be grouped by atom type, for family trees by surname, for UML class diagrams by type (class, interface, etc.).

The precise structure of descriptive summaries may also need to be informed by an understanding of the differences between route-based and survey-based descriptions, and by how ordering items makes it easier or more difficult to make inferences when building a mental model.
5.4 COMPARATIVE SUMMARIES

5.4 Comparative Summaries

Imagine one had used a graph to solve a problem: this could have taken considerable effort — some to understand the structure of the graph and more to solve the problem. If one came across a similar graph at a later date this process would have to be repeated. If, however, the similarities were known about before exploring the second graph, it is possible that considerable exploring effort could be saved. For example, if the two graphs were phenylalanine (see Figure 3.1) and another molecule the same, but for an additional chain of two carbon atoms attached to the ring, the user could be told upon loading the latter ‘this graph is the same as phenylalanine but with two additional nodes’ (or something more precise or more domain-specific). This summary comparing one graph to another would clearly give the user a significant head-start, assuming he or she is able to recall some of the structure of phenylalanine. A second obvious application for comparative summaries is comparing two versions of the same graph, e.g., one that has been edited by a colleague with the original. This section discusses in more detail why these types of summary should be useful, and the barriers to implementation.

5.4.1 Theory

Chapter 2 discussed how recognition was a key to the benefits of presenting information diagrammatically. Larkin and Simon demonstrated that some implicit features became explicit when presented as a diagram, thus reducing the mental effort required to solve problems [51]. But rapid identification of features within a diagram is not the only type of recognition that benefits users: experience also enables recognition. When solving a problem, users refer to their experience to recognise problem classes and find solutions more rapidly. For example, it has been shown that grandmaster chess players are excellent at recognising board configurations [24] and that this plays a major role in performance [33]. In the latter paper, Gobet and Simon demonstrated that grandmasters’ performance in multiple simultaneous games (where they have insufficient time to use look-ahead strategies) is only slightly lower than in tournament conditions, where time is less limited. This confirms the importance of recognition in problem solving for experienced chess players; it is not unreasonable to expect this is also the case in other domains.
Before the application of a solution to a recognised problem, as discussed above, it is necessary for the problem to be conceived in a manner that enables recognition. According to Jonassen [48], there are two critical attributes of problem solving: “First, problem solving requires the mental representation of the situation in the world. That is, human problem solvers construct a mental representation (or mental model) of the problem, known as the problem space [63]” and “Second, problem solving requires some activity-based manipulation of the problem space”. In this model, we can say that when grandmasters recognise board configurations, they are recognising that the problem space is common with that of some previously encountered configurations. Considering the concept of mental models introduced in section 2.4.1, this implies that recognition is a matching of the mental model of the current problem with one in long-term memory. Thus the mental representation must be constructed before recognition of the problem can take place and, since diagrammatic presentation facilitates model building (through recognition of features), it is reasonable to assert that diagrams should thereby facilitate problem recognition.

For non-visual explorers, it is not just recognition of a familiar problem space that could be useful, however, since the difficulties for them are extended to include reading the diagram (building the model). Consider two very different tasks on the same diagram: For sighted users, there is unlikely to be a benefit from knowing that the two graphs are the same, although they will probably recognise the fact immediately. For someone solving the problems non-visually, unless they were told beforehand, some time and effort, perhaps considerable, would be used exploring the two graphs. If they were aware in advance that the two graphs were identical, the exploration experience gained while solving the first problem (at least some form of mental model of the graph will have been formed) could be applied to the second, thereby reducing the effort required to solve it. Using the spatial metaphor for exploration, we could compare this to the experience of shopping in chain stores: even if a particular store has not been visited previously, knowledge of other stores in the chain makes it easier to find particular sections or items in the new store.

This argument could be extended further by asserting that knowing that a new graph is similar to one that the user has previously explored would be useful. To extend the spatial metaphor used above, we could say that, even if stores are not laid out identically, knowledge about their organisation is still likely to be
beneficial. This is, in essence, a type of overview or summary: it is information about the graph as a whole that can help the reader form a mental model of the graph before starting detailed exploration.

5.4.2 Practicalities

The previous section has demonstrated how recognition of a graph as belonging to a particular type may ease problem-solving for experienced users. It has also discussed how, for non-visual readers, enabling recognition of a familiar graph structure could reduce effort in understanding the structure, even for different tasks. This section discusses the practicalities of implementing such recognition. There are two main areas for difficulty: how to detect similarities and differences, and how to present the information.

One could imagine that, in practice, a graph-reading system would record all the graphs explored by a user, and each time a new graph is opened, it would be compared to all previously read graphs. If any of these are sufficiently similar to the new graph it would be compared to the most similar. It might also be possible to describe the similarity in relation to more than one graph, for example if the new graph is simply two previously explored graphs connected to each other.

Perhaps the fundamental problem is that inexact graph matching is an NP-complete problem. This does not in itself, however, mean that generating these comparisons is impractical, at least for small graphs; the subgraph isomorphism problem is also NP-complete but can be solved without significant problems for reasonable sized graphs. For example, Ullmann’s algorithm [104], which combines a backtracking technique with forward checking was implemented in Kekulé for recognition of features (subgraphs) within the graph (section 3.1.1) and used for graphs with up to 28 nodes, searching for 23 different subgraphs with between 1 and 7 nodes. It is also advantageous that, for this type of application, the optimal solution is not required, merely a good one. For example, if we can map the differences between two graphs in different ways, any good mapping could provide the user with useful information — it need not be the very best mapping.

Determining graph similarity is a matter of calculating graph edit distance, that is calculating the cost of transforming one graph to the other (see Bunke’s review of graph matching [22] for a discussion of how different graph matching problems are essentially special cases of graph edit distance computation, and for more details on graph matching generally). This distance is calculated by
assigning a cost to each type of edit (e.g., delete node, change node name, add arc) and calculating the set of edit operations that transform one graph to the other with minimum cost. Although the costs can be equal for all operations, measuring similarity between real graphs is likely to give best results if a domain-specific cost function is determined. This is difficult: it is subjective and fairly arbitrary and would need evaluation (probably an iterative process of evaluation and tweaking) over a range of graphs.

One further significant difficulty in helping readers in this way is describing the similarity to the reader. The most simple situations would not be too difficult: for example ‘A is the same as B’; ‘A is a subgraph of B’. Essentially it would become more difficult to describe the more different the graphs. At worst case, however, knowing that there is some similarity could prove useful.

5.4.3 Experiment
To explore what makes graphs similar, and to allow future testing of any algorithm, the 23 participants in the audio summaries evaluation were asked to rate which pairs of graphs, from the four in each multiple choice, were the most and least similar. These questions were presented on the same page (of the web-based evaluation), when the participants had the opportunity to listen to the audio glance multiple times and simultaneously view the graphs (i.e., stage 3 in section 5.5.3.1). Participants were asked to make this choice on ten questions, covering a set of twenty-one abstract graphs.

The selections made suggest that there was more consensus on which graphs were most similar than on which were least. For example, comparing the percentage of votes given to the most popular pairs, for similar pairs the most popular pair gets a mean of 73%, while for the least similar pairs, the most popular choice gets a mean of 52%. This difference is statistically significant to greater than 95%. This result is perhaps because of the use of a close distractor for audio glance in the multiple choice, although it should be noted that the distractor was chosen to test the ability to distinguish the glances, not on appearance.

The most obvious correlation within similar pairs was that in all but one of the ten questions, the two graphs paired as ‘most similar’ were either both cyclic or both acyclic, i.e., either both, or neither, contained a ring. In the exception, a simple long chain was matched with a simple chain with a small ring on the end (Figures 5.2(g) and 5.2(h)). In this case the 13 participants who selected
this pairing considered the chain to be the most significant part of the graph. The next most popular pairing (selected by 9 participants) matched two acyclic graphs, the simple chain with a chain with some branches on the end (Figures 5.2(g) and 5.2(j)). In contrast, eight out of the ten questions had one cyclic and one acyclic graphed paired as the least similar. Figure 5.2 gives examples of some of the most popular pairings.

![Figure 5.2: Pairs of graphs selected as the ‘most similar’ from among four graphs. Participants were not given guidelines for determining similarity.](image)

**5.5 Audio Summaries**

Turning from the theoretical benefits of overview to some concrete ways in which graphs may be annotated with these sorts of summary, this first section considers, appropriately, the reader’s very first view of a graph. This initial impression, formed in a brief period of time, forms the basis for all further reading. The
impression the reader forms of the graph is used to inform strategies for more detailed investigation. Physically, the glance is formed from a series of saccadic eye movements across the diagram, with the attention falling on the most visually salient features of the graph [44]. The later sections in this chapter consider more detailed types of summarisation; this section explores the feasibility of giving similar glances through the audio channel.

The audio summaries described in this chapter are descended most directly from the work of Stevens et al. [95]. Theirs was probably the first sonification of complex structured data; they demonstrated that it was possible to generate audio glances of mathematical equations. Despite the obvious differences in the structure of the information being presented, this current work is in many respects an extension of theirs: the intention is identical — to provide a glance that facilitates planning prior to reading. Stevens used different timbres to represent different types of element within the expression (operands, fractions, subexpressions, etc) and manipulated the timing, pitch and amplitude according to a set of rules depending on the structure of the equation. Complex items were presented using a continuous tone. They found that listeners were able to derive sufficient syntactic information from the earcons to enable them to select the correct equation from among four alternatives.

Probably the only previous research explicitly aimed at sonifying graphs is Brewster’s work with hierarchies. They examined if earcons could be used to present to users their position in a hierarchy [15, 16]. They tested both compound earcons and hierarchical earcons. In the more successful, compound earcons, different timbres were used to represent different numbers, and the hierarchy position was generated by concatenating. For example, the concatenated number ‘3.2’ would represent the second child of the third child of the root node. Using a tree (hierarchy) of 27 nodes in 4 levels, they found that participants were able to identify their location by listening to a compound earcon with an accuracy of 97%. This work was aimed at giving specific information about position in a graph rather than an impression of its overall structure and as such, it is comparable more with the highlighting in phase four (see Section 5.5.3.3) than the overview glances described in the bulk of this chapter. Although successful, it is a much more constrained approach that could not easily be generalised beyond trees.

Hermann and Ritter [38] describe an approach for ‘interactive exploratory sonification of high-dimensional data’ which is quite closely related to creating
audio glances of graphs. First, their technique is designed to give an *impression* of the data as a first step in data analysis, and potentially allow one to spot features that would otherwise be hidden within the high dimensionality of the data. Second, the sonification is actually performed on a graph — the network graph formed by employing a growing neural gas network on the data; here the neurons form the nodes while the edges are the connections between them. They sonify this graph by treating it as masses connected by springs; when a mass is excited, either directly by the listener or by another node via its connecting springs, it emits a noise. The entire network can be excited by ‘shaking’ it, or the data can be explored by exciting a particular node and listening to the sound spread. This approach is very general and could well be used for summarising graphs rather than data. The interactive aspect is particularly appealing.

Another method for presenting spatial information is to scan over the data in a known pattern, sonifying each data point as it is passed over. This can be combined with two-dimensional or pseudo two-dimensional sound — the approach taken by Zhao *et al.* for presenting geo-spatial data [118]. Their ‘spatial chloropleth map’ presented an audio glance of data from US states by scanning the map, sonifying the data while speaking the state names. The stereo position gave the longitude, while the latitude was given by the pitch of a piano note. Volunteers were able to identify the class of data pattern (e.g., vertical strips) with 56% accuracy after a single listen (cf. 20% for random guess). This type of sonification appears to require a spatial arrangement of the data that will fit a simple scanning pattern; it would be interesting to repeat the experiment replacing the states of the US with, for example, the counties of the UK. More generally, research into sonification of numeric data has much to offer in terms of use of different aspects of sound to present different aspects of data (parameter mapping; Barrass and Kramer discuss this briefly in their overview of sonification [6]).

Another effective use of audio summaries is Kildal’s TableVis — a tool allowing non-visual access to numerical tables [50]. Kildal and Brewster used a graphics tablet to give visually impaired users a sense of space; all tables were scaled to fill the tablet. Users were able to get summaries of rows (or columns) by selecting the appropriate mode and moving a stylus to a point on the tablet: touching a cell would give a sonification of that row (or column). Sonification was achieved by mapping the values to MIDI notes and playing the cells in the row from left to right in rapid succession (or top to bottom in the case of columns).
The user had easy control of the speed of sonification by turning a control knob — this changed the character of the summary from a series of notes (which may indicate a trend) towards a ‘dissonant chord’ (which was thought to give a good indication of the average value). Due to the short time to hear the row summary, it is possible to gain an overview of the whole table by listening to the sonifications for each row in rapid succession. TableVis also allowed access to the detail through synthetic speech. The experiment described in [50] suggested these overviews were effective.

5.5.1 Requirements

What is required from an audio glance if it is to achieve the same objectives as its visual counterpart? The following characteristics are proposed, in addition to the general requirements of a summary given above:

1. It must be short. A glance is by definition a quick activity; more sophisticated overviews can be provided in the next stage of exploration. It is, however, unrealistic to expect them to be as short as visual glances; a few seconds is probably sufficient, and considerably shorter than ‘bottom-up’ exploration.

2. It must give an impression of the size and complexity. Even in the largest and most complex of graphs, just knowing that it is large and complex is useful. For the simplest of graphs, the listener may decide it is so trivial that reading each node and arc in turn is an appropriate strategy.

3. It could give an idea of topology, but this is not essential. The visual glance merely gives an impression; the audio glance should do likewise. Although it is impossible for a glance to convey the detailed topology of large and complex graphs, useful information may still be gleaned. For example, it would be beneficial to identify that a graph has a simple linear section connected to a group of highly connected nodes.

4. It is better to be vague than misleading: a misleading glance could cause great confusion, while a vague one merely demands further effort to develop the necessary overview.
5.5.2 The Algorithms

Earcons are abstract, structured, non-speech sounds [11] and as such are suitable for presenting structured information (e.g., graph topology) in a rapid manner. In this prototype system, earcons are used to present the audio glance (the terms ‘earcon’, ‘audio glance’ and ‘audio summary’ are used interchangeably in this thesis). These are generated by an algorithm that uses both the topology and spatial layout of the graph.

The algorithm developed for presenting glances of graphs generates earcons from graph representations that describe both the topology (i.e., to which other nodes each node is connected) and layout (i.e., co-ordinates for each node). It is essentially a breadth-first algorithm, and works by following the connections from the left-most node. This node, identified by its co-ordinates, is played first — a tone with a duration of 100ms. After a pause of 300 ms, all nodes connected to it are played, with 50 ms between each. After another pause, all nodes connected to these nodes are played. This continues until all nodes have been played. In addition to representing the distance of a node from the left most node (as measured by number of arcs, rather than space) by time, the stereo axis is used to represent the vertical spatial dimension. This is done by detecting the centre of gravity along the y-axis, then the further above this line a node is, the further left its sound is played, and vice versa.

![Figure 5.3](image)

Figure 5.3: The algorithm applied to a simple graph. The horizontal line in (a) represents the y-value of the centre of gravity of the graph; nodes close to this line are played in the centre while nodes above and below are played towards the left or right respectively. The left-most node is played first (a). After a 300 ms pause the two nodes connected to it are played (b), one beep just left of centre, and another beep on the far right. Next all nodes connected to these two are played (c), and so on until all nodes have been played (d).
This design of algorithm ensures that some of the requirements outlined above are met. The earcons it generates are short for typical graphs (the glance generated for the graph shown in 5.3 is approximately two seconds long), and have the potential for conveying size, complexity and topology. The evaluation described in Section 5.5.3 is designed to determine if this potential is realised and the other requirements met. The one significant problem with this algorithm is meeting the predictability requirement: similar (or even identical) graphs could give quite different sounds if originally drawn with different orientations. Although these differences do not affect how effectively the glance gives an impression, they do make recognition more difficult.

It might be argued that mapping connective distance to time and spatial y distance to left-right is confusing, but this algorithm was preferred to a purely spatial one (where the two spatial axes are mapped to time and stereo) for its better representation of the graph topology. In order to keep the size of the experiment reasonable, it was decided to test one algorithm; if this showed promise for the principle of summarising graphs with audio glances, further experiments could be performed to identify effective algorithms.

The second algorithm used in this experiment was identical but for the addition of some ‘ambient’ sound. In this case, the beeps representing the nodes were annotated with a sound to signify the presence of a ring. This sound was continuous, playing from just before the first node in the ring until just after the last. If there was more than one ring in a graph, each was played at a slightly different pitch.

5.5.3 Evaluation Method

The aim of the evaluation was to determine whether the earcons generated by our algorithm meet the requirements outlined in Section 5.5.1. An experiment was devised to test this in two ways. First, it was tested indirectly, by seeing if a graph could be recognised after using an audio glance; second, it was tested more directly, by asking participants to listen to a glance then write a description of their impression. A range of graphs with different sizes, complexities and topologies was used.

The experiment itself was divided into four phases, which are described in more detail below. In summary, the first two phases presented a glance to the participants, then tested if they could recognise it from among three other graphs.
These two phases differed only in the graphs used and a small change to the algorithm used to generate the earcons. The third phase required participants to listen to the glance, then write down their impression of the graph, the quality of which was determined by asking them to match graphs to their descriptions. A fourth phase examined a different question, testing if summaries could be used as an orientation aid, highlighting particular nodes in a graph. Although not a specific requirement for a glance, the ability to orientate oneself with a glance is one of their possible uses. Each phase consisted of five questions. All graphs were abstract undirected graphs, with between 3 and 15 nodes, which resulted in audio summaries between 1.3 and 4.5 seconds long.

Achieving a success rate for matching graphs that is significantly greater than that expected by chance would support our hypothesis that the earcons successfully give an impression of the graph. To test the final requirement, that it is better to be vague than misleading, participants were asked to give the confidence they had in the correctness of each answer. A significantly lower confidence for wrong answers would indicate that, in general, participants were aware of when they were guessing, and would therefore be less likely to be misled in practice.

Evaluations were performed over the web with pages generated using PHP and Java applets\footnote{Although results are no longer being collected, the evaluation may still be completed at \url{http://aig.cs.man.ac.uk/people/andybrown/audioeval/intro.php}.}. These were designed to make cheating difficult, and each user action (listening to a glance or loading a page) was recorded. Participants were free to do the evaluation where and when they wished and were not supervised. Volunteers were sighted computer users recruited from the School of Computer Science at Manchester University. Volunteers were encouraged to use headphones for the evaluation, although stereo speakers were acceptable. They were asked which they were using and the answer recorded. Each participant did phases one to four, in order, typically taking a total of between 30 and 45 minutes to complete the evaluation. Further details, including participants instructions, distractor graphs, and all the descriptions given by participants in the third phase are given in Appendix C.

5.5.3.1 Phases 1 and 2

These phases had identical structures, but a different algorithm was used to generate the audio summaries for a different set of graphs. The aim of these phases
was to see if the algorithms were successful in conveying an impression of the graph. This was tested by playing an audio glance, then asking the participants to match it to one of four graphs. Pilot studies indicated that presenting the graphs and sound together resulted in participants playing the sound multiple times, each time attempting to match the beeps directly to the nodes on one of the graphs. This was clearly not testing whether they had formed a valid impression, so their actions were restricted and the procedure altered to the following:

1. Present an audio summary of a graph. This was played when the participant pressed a button and could only be heard once.

2. Ask the participant to select the graph the summary was thought to represent, from a choice of four. The options were not visible when the sound was initially heard. Moving from the listening phase to the selection phase required action by the user — it was hoped that this would limit their ability to remember the precise pattern of the glance and concentrate instead on the impression it gave.

3. Allow the participant to listen again to the summary, as many times as desired, while viewing the four choices, and ask the user to select again. This would allow us to see if multiple glances were of any benefit. Each listen to the sound was recorded.

This was repeated on five graphs for each of the algorithms (see Figures 5.4 and 5.5). Distractors were designed to test participant’s ability to use the glances to distinguish certain characteristics of the graphs. Questions 3 and 5 included graphs of different sizes. For the remainder, all choices were eight node graphs — this was to test properly the ability to distinguish complexity and topology rather than just size. For both selections, the user was asked to rate the confidence they had in their decision. This used a slider bar that was initially positioned at 50%; higher values (to the right) indicated greater confidence in the decision. As part of some related work, participants were also asked which of the four graphs they thought were most and least complex, and which pairs most and least similar. These latter questions were asked after stage 3 above so as not to interfere with the main task.
5.5. AUDIO SUMMARIES

5.5.3.2 Phase 3

The authors were concerned that phases one and two were testing the ability of participants to recognise graphs from their sounds, which was not necessarily the same as testing whether they gained a useful impression of the graph topology. It should be noted, however, that successful recognition would still suggest that the glances are giving useful information about the graph. Phase three aimed to test more directly whether people were able to identify anything about the graph (size, complexity, topology, etc.) from the summaries.

Participants were presented with five buttons for playing audio glances of five graphs of different size and complexity (see Figure 5.6). Each glance could be played up to two times. Next to the buttons were text fields for them to enter a description of the impression given by each summary. After submitting their descriptions (all at once), the volunteers were presented with 13 different abstract graphs. They were also presented with their five descriptions, although in a different order, and with each description a selection of check boxes. They were asked to select which graph they thought their description matched. Although
5.5. AUDIO SUMMARIES

121

(a)  (b)  (c)  (d)  

Figure 5.6: The graphs used in phase three.

encouraged to select only one graph, they could choose more if they were unde-
cided. This was intended to allow us to identify if, when a particular graph could
not be identified, the type of graph could be recognised. As in the earlier phases,
they were given a slider to rate the confidence they had in their choice.

5.5.3.3 Phase 4

The final phase of the experiment was to test whether audio glances could be
used as an orientation aid during exploration, as well as providing context at the
start of reading. To give orientation information, one or more nodes of the graph
were highlighted in the summary. This was achieved by playing them at a higher
pitch, and was intended to give the listener an impression of their current position
in the context of the whole graph. The ability of people to identify which graphs
were highlighted was tested as follows.

1. An audio glance of a graph was presented without highlighting. Volunteers
could listen up to five times.

2. On pressing the continue button, they were shown the graph visually. This
was intended to replicate detailed non-visual exploration, since orientation
aids are only useful once one has started exploring.

3. On pressing the continue button again, the graph was hidden and they were
shown a button that enabled them to listen to a summary where one or more nodes were highlighted. Hiding the graph while playing the glance avoided problems of direct matching of beeps to nodes identified in the pilots of phase one.

4. After the second listen, the graph was shown again and they could listen no more. They were asked to select which nodes were highlighted; this was done by clicking on the appropriate nodes in the diagram.

5. On clicking the submit button, participants were taken to stage one of the next graph. Five graphs were completed in this manner.

Completing the fifth graph of this phase represented the end of the experiment, and participants were asked for their comments. Figure 5.12 shows the graphs used and which nodes were highlighted.

5.5.4 Results

Twenty-four participants completed the evaluation; the results are described in detail below. It is possible to make some general comments relating to all phases. Results showed that overall there was a high success rate, with the average participant identifying the correct graph in 13 of the 20 questions (65%). There was a wide variation in individual performance (standard deviation of 14%) with scores varying between 45% and 90%; these are shown in Figure 5.7.

Several participants commented that they felt that they were improving as they went through the study — in the words of one:

‘Interesting learning as the test went along. I could definitely spot the structures (or something pretty close indeed) from the sounds by the end, although maybe only because a key dozen or so structures were used so I knew what I was listening for.’

Despite claims by those piloting the study that it was easier using headphones, the difference between those listening through headphones (67% correct) and those using speakers (59% correct) was not statistically significant.

5.5.4.1 Phases 1 and 2

Figure 5.8 shows the percentage of participants that selected the correct graph for each question, both after the first listen and after multiple listens. Questions
5.5. AUDIO SUMMARIES

Figure 5.7: The scores of individual participants for all phases. Each bar gives the score obtained by participant for each phase, with phase 1 at the bottom and 4 at the top. There was a maximum score of 5 for each phase.

1 to 5 use the plain algorithm, for questions 6 to 10 the audio summary also highlighted rings. This shows that recognition was on the whole good, with an average of 70% of participants choosing the correct graph after just one listen. Interestingly there was no significant difference between this and the mean percentage of people correct after multiple listens (77%). Participants achieved an average of 7 questions correct out of 10; a binomial test for this, with a probability of success being 0.25, gives a probability of this result happening once by chance of 0.003.

Although there was no improvement in scores from annotating the sounds to indicate the presence of rings, analysis of the results shows that there was a clear advantage in having it — only once was a graph not containing a ring selected when the audio glance contained this ring sound. This was on the first question for the algorithm and the mistake was rectified after listening again. There was more difficulty in identifying the topology of fused rings, as evidenced by questions 7 and 9. The correct graph and the close distractor for question 9 are shown in Figure 5.9; all participants recognised the presence of rings but 50% were unable to distinguish between the correct graph (5.9(b)) and the close distractor (5.9(a)) after the first listen. The pattern of beeps is identical for these graphs, so distinguishing them requires careful identification of where the two ambient
5.5. AUDIO SUMMARIES

Figure 5.8: Results for phases 1 and 2. The left hand bars show the percentage of participants that selected the correct graph after one listen, the right hand bars after multiple listens.

sounds denoting the rings start and finish. Similarly, twelve people incorrectly thought the glance for question 7 (which represented the graph shown in Figure 5.9(c)) was actually for 5.9(a). The recognition rate improved (with six people changing their selection) after further listens for this question, presumably because the different beep pattern helped identification. The level of detail required to answer these two questions correctly, particularly question 9, is probably close to the limit of that which a glance could be expected to provide.

Figure 5.9: Double-ring graphs were difficult to differentiate.

The star topology of question 4 also presented some difficulty, although after further listens a recognition rate of 61% was achieved. Analysing the results with those from the same graph presented in phase 4 (Figure 5.12(c)) suggest that people might imagine the central node to be the one played first.
It is also possible to examine how accurate people’s perceptions of correctness were, i.e., did they know when they might be wrong? If the mean confidence rating for correct answers (80%) is compared with the mean rating for incorrect answers (54%) it can be seen that participants usually had a reasonable idea when they were correct and when they were guessing. A two-tailed t-test ($T=7.33$) gave the probability of this difference being due to chance as $p < 0.001$. There were still, however, cases where the participant was 100% confident, but wrong.

### 5.5.4.2 Phase 3

Figure 5.10 shows the percentage of participants that selected the correct graph for each question. The average score was 3.4 correct from the 5. With 13 graphs to choose from the probability of selecting 3 correct graphs by chance is 0.004.

![Figure 5.10: Results for phase 3.](image)

Everyone got the first question right and all but one the second, and the descriptions matched the graphs well, indicating that the correct impression had been given. Nearly all descriptions for the first graph (Figure 5.6(a)) were variations on ‘Two rings then a chain’, and the second were ‘Short chain’ or ‘Three nodes in a chain’. The single error on graph 2 was where the description simply ‘A Chain’ and an 8-node linear chain was selected instead of the 3-node one.

There was more difficulty with graph 3 (Figure 5.6(c)); although some described it quite accurately (e.g., ‘binary tree on its side’), most of the descriptions were more vague. Eight participants mentioned branching or forking, five used
the term ‘tree’, and five explicitly noted that there were no rings (e.g., ‘no rings just a lot of branches, medium complexity.’); this was clearly noticed by most as only five participants selected graphs with rings (7 of the 13 choices contained one or more rings).

Graph four (Figure 5.6(d)) was identified by a little over half the participants. All but one participant identified that there was a chain followed by a ring, but seven erroneously thought the ring was followed by another node or two. Five of these selected a graph with two nodes then five nodes forming two joined rings, with one node hanging off the end.

The final graph (Figure 5.6(e)) caused the most difficulty. Again most people recognised that there were no rings in the structure — only three matches were made with ring structures. Fifteen (63%) recognised (and specifically mentioned) branching, tree-like structures or intersecting chains, but only four of these could match the graph with their description. One example is ‘a short chain then a branch to make a tree like structure’; this is an accurate description of the graph, but its author was unable to make the correct choice.

5.5.4.3 Phase 4

Overall the results from the highlighting phase of the evaluation indicate that people were able to identify approximately which nodes were highlighted but not always the exact combination. Figures 5.11 and 5.12 summarise the results.

Figure 5.11: Results for phase 4.
5.5. AUDIO SUMMARIES

Figure 5.12: The graphs presented in phase 4. The height of the bars represent the number of people who thought a node was highlighted; a red bar indicates it was actually highlighted, a blue bar that it was not. Some of the graphs have been rotated slightly for clarity (although this would have no effect on the sounds).

These results show that there was little difficulty identifying the highlighted nodes in simple graphs where the beeps representing these nodes were not mixed amongst other nodes. For example, 88% of participants were correct in graph 4, and 71% in graphs 1 and 3, while the nodes in the lower branch of the tree of graph 2 were mixed amongst the nodes in the higher branch and it was difficult to identify exactly which were highlighted. The same was seen, to a greater extent, in graph 5. Figure 5.12 shows, however, that it was possible to identify the approximate area of the highlighting. It was also easier to identify a single node than a cluster. In these cases, all participants recognised that only a single node was highlighted, even if they did not identify exactly which node(s) were selected. Where multiple nodes were highlighted, and particularly where they were mixed with other nodes, it was clearly more difficult — on graph 5, where
6 nodes were highlighted, participants selected between 2 and 9.

### 5.5.5 Discussion

The results presented above confirm our hypotheses and indicate that the audio glances created fit the requirements outlined in Section 5.5.1. The algorithm ensures that glances are kept short, at least over the range of graphs used in this evaluation. Evidence that the size and complexity of a graph is conveyed by a glance is provided by the successful recognition over a range of graphs, supported by the descriptions given in phase three.

The third phase also demonstrates that successful recognition is through the formation of a mental impression of the graph rather than an ability to match glance to diagram. It is quite clear that some topology is conveyed by the glances, at least in simple cases. Evidence has also been presented that shows that wrong answers were associated with lower confidence, suggesting that the glances also fulfil the fourth requirement. Stevens use of a single tone to represent complex regions [95] might be worth investigating should misidentification prove a problem.

It is not too speculative to suppose that regular use of such glances would bring improved results. As described above, some participants described a learning effect through the evaluation. We can also examine the cause of some of the misidentifications and attribute them to lack of familiarity. For example, although the extra sounds warning of the presence of rings were successful in preventing participants incorrectly identifying the sound as representing an acyclic graph, the converse was not true. It is reasonable to suggest that more practice would help users remember that the lack of a ring noise means that there are no rings present. Brewster’s work [16] was, essentially, testing whether people could learn the earcons representing hierarchy position — their high recall rates suggest that patterns of sound can be learnt.

The use of these glances for highlighting has only been explored in a very limited field by Brewster ([15], described above), but this evaluation shows that particular nodes can be highlighted and their approximate location identified (or even the exact location in simple graphs). There are many possible applications for this type of highlighting in non-visual graph exploration. As mentioned in the introduction, it could be used as an orientation aid, reminding the user where in the graph they are, or as a marker to indicate where a graph has been edited.
Search results could also be summarised in this way, giving readers a rapid idea of where in the graph the nodes of interest are located.

The evaluation has, however, highlighted some problems with the algorithm. There is a tension between topology, which is the real meaning of the graph, and layout. The knowledge of the layout of a graph is probably important to visually impaired users, particularly if they are to discuss it with a sighted colleague, and may be a useful orientation aid when exploring (e.g., to know that they are near the top right of the diagram). The algorithm, however, does not present both topology and layout. This tension is demonstrated in the final graph of phase three (Figure 5.6(e)); although descriptions were quite accurate, matching was not. We suggest that this was because the mental image of a branching tree involved branches directed to the right with the root on the left. The diagram, however, had two branches of nodes aligned vertically which might have been expected (incorrectly) to be played as one group. We would however argue that, in this case at least, the impression given was a good basis for further investigation. Approaches similar to that of Zhao et al. [118] may offer a solution — using 2D or pseudo-2D sound to provide further clues to the topology.

Another related criticism that might be levelled at this algorithm is its left-to-right reading (with up and down mapped to left-right stereo). The choice of starting node will have a big impact on the sound of the glance and how much information can be gleaned from it. One approach could be to sonify along the longest dimension of the graph, but if we are to use positional information in the later exploration, it is of no use having an apparently random orientation for the initial glance. It is not clear how to resolve the issue of topology versus layout, but the type of sonification used by Hermann and Ritter [38] might offer something, and its interactivity could be useful during later exploration — giving readers the ability to listen to the graph from different viewpoints. A further area for investigation is to examine how effective this algorithm is at representing more highly connected graphs; those used in this evaluation are on the whole quite sparse.

Less fundamental problems also arose. It was clear that, despite their usefulness overall, the noises presenting rings could be improved. The difficulties observed in distinguishing the graphs of Figure 5.9(b) and erroneous identification of an extra node or two to the right of the ring in the graph of Figure 5.6(d) indicate that participants found it difficult to identify the exact start and finish
of these sounds. Further investigation could bring improvements. Equally, it is necessary to consider how useful ring emphasis is for implementation in a concrete domain. It is believed that the presence of rings in a graph is likely to increase disorientation during exploration, and that perceiving their presence in an initial glance is going to be of benefit for any type of graph. Nevertheless, some domains may have other features, the presence of which may be of more interest to the user, and the techniques applied here to emphasise rings should be transferable.

Some participants also complained that the speed was too great. Although we would argue that it is important that the glance is rapid, some facility for adjusting the speed of glances could also assist users, particularly in the early phases of learning. Another issue to consider is scalability of these algorithms. Although the evaluation included graphs of between 3 and 15 nodes, real applications are likely to include larger graphs still, and possibly graphs which are more highly connected. The algorithm used to generate the audio summaries is such that even graphs of twice this size should not take too long to listen to (and it is believed that the glances could be played faster to experienced listeners), but further work is required to determine if it would still be possible to glean useful summary information from these glances.

Beyond the difficulties discussed above, there is also much scope for investigating other parameters of the algorithm such as the relative size of the pauses, the nature of the beep representing the nodes, and how best to distinguish multiple rings. Having demonstrated that such glances can be successful, there is a need to apply it to real graphs. This would require investigation into how to present different node and arc types, directed arcs and various other features. A further avenue for investigation is to examine how audio glances like these may be used in an audio analogue of the coordinated multiple views of visualisation [4].

Nevertheless, these points reveal opportunities for investigating improvements to a successful audio glance at abstract graphs. These structures form an important part of much information and a glance at their form is an important aspect of improving access to many information sources.

5.6 Conclusions

Evidence from studies of reading and videos indicates that previews increase comprehension. There is also a strong pedigree of using summaries in information
visualisation, providing more anecdotal evidence that they are useful. This chapter has examined how graphs may be annotated with summary information, with the intention that these notes help users by simplifying the process of mental model formation. Requirements for annotations presenting summaries have been suggested — the principle ones are brevity and accessibility. This chapter has examined three different types of summary note, concentrating on audio glances.

An algorithm has been designed to generate short non-speech sounds that are intended to convey an impression of a graph. This algorithm was evaluated by asking participants to listen to the sound, then match it to a graph. They were also asked to listen to audio summaries of some graphs and describe their impressions. Both the matching and the descriptions indicated that this algorithm was successful in conveying a rough impression of an abstract graph. As a final part of the same evaluation, the possibility for using these summaries to highlight nodes was tested — this was less successful, but still effective to a certain extent, especially on simpler graphs.

This chapter has also discussed other forms of summarisation, namely giving the complexity of a graph, presenting a list of its components, and comparing it to previously explored graphs. The first of these is relatively simple, although the notion of complexity can be tricky as many attributes of a graph influence how complex it is to understand, including size, connectivity, symmetry and its semantics. Comparative summaries are potentially extremely useful, but very difficult to implement. Inexact graph matching is a difficult problem for abstract graphs as it is an NP-complete problem and, for meaningful graphs, domain knowledge is almost essential for identifying differences. Further difficulties are likely to lie in generating good descriptions of how two graphs differ.

Annotating a graph with summary information is likely to help users by simplifying the process of mental model formation, and could also be useful for orientation during exploration. Thus we see that summary annotations may sit in both the summary and orientation categories of the taxonomy. It is possible to generate notes that fulfil the requirements for such annotations; it is now necessary to evaluate if these are effective.
Chapter 6

Node Identification

Having considered forms of annotation that can be applied to graphs as a whole, we now turn to the other end of the scale — annotations that can be applied to individual nodes. In particular, this chapter considers notes that enable node differentiation, such as numbering, and landmark formation. These types of annotation should help navigation and reduce disorientation; i.e., help the user build an accurate mental model of the graph.

The first section considers the requirements: why do we need to differentiate nodes, and how are landmarks important? Section 6.4 describes the development and evaluation of a numbering algorithm that may be applied to graphs. Sections 6.2 and 6.3 consider how the use of ‘breadcrumb trails’ and user-labelling of nodes might help reduce some of the problems of non-visual exploration.

6.1 Literature Review

Before considering the specific requirements for node identification, we must consider the tasks that are being performed in the environment. In this discussion, we are limiting this to the low-level task of movement around the graph, rather than considering the overall problem that the user wishes to use the graph to solve. Allen [1] has proposed a functional framework for wayfinding, which links the tasks with cognitive requirements, so is worth describing in some detail.

Firstly, Allen classifies wayfinding tasks into three types: commute, or travel along a known route; explore, or travel for the purpose of learning about the environment; quest, or travel to a known, but unvisited destination. He goes on to describe the means used for achieving these tasks and, from there, gives the
cognitive abilities required by these means. In the context of non-visual graph exploration, the key tasks are more related to exploration and to quests (in order to determine the relationship between nodes), rather than commuting, so it is on the means used in these tasks that we shall concentrate. The tasks given by Allen as relevant to exploration and quests are:

**Piloting** involves landmark-based navigation, where the traveller moves from one landmark to another towards a goal. The key cognitive abilities required for piloting are object recognition, paired-associate learning, and sequential learning.

**Path Integration** involves ‘updating one’s current location with reference to a point of origin’, and therefore requires movement monitoring and position updating abilities. Allen notes that this is demanding on memory, particularly if there are many changes in direction, or if the path crosses itself. He also comments on how landmarks greatly increase the accuracy of short-cut attempts.

**Navigation by Cognitive Map** is navigation using a mental model of the environment to determine direction. “The cognitive map serves as a knowledge base, but the actual ability to navigate using this information entails the computation of inter-landmark vectors and metric distance”.

We can identify from Allen’s framework that object recognition and the use of landmarks are important in all tasks. These are discussed in more detail below.

### 6.1.1 Node Differentiation

If objects in an environment are to be recognised, they must appear different. This is more of a problem when exploring in the non-visual domain since environmental features (nodes in this case) have fewer distinguishing characteristics; they cannot be distinguished by size or colour for example. For graphs in particular, differentiation is simpler in some domains than others: those where all nodes are required to have unique names, for example. But consider domains where this is not the case: in molecular structure diagrams it is not unusual for many, or even all, nodes to represent carbon (hydrogen atoms are usually implicit). This homogeneity could cause great confusion when exploring, being
similar to visiting a maze, which is designed to be difficult to explore. Without distinguishing features, recognising routes (or even current location) becomes very difficult. As Sorrows and Hirtle [90] said of the problems associated with maze navigation: “This shows the importance of differentiation in composing a cognitive understanding of an environment”. Clearly mechanisms must be available for differentiating nodes where the name does not.

6.1.2 Landmarks

Sorrows and Hirtle [90] examined the literature on landmarks in real and virtual environments. They gave a rough definition as:

“Landmarks are prominent, identifying features in an environment, which provide an observer or user of a space with a means for locating oneself and establishing goals.”

They proposed a three-way classification of landmarks. In summary these are:

**Visual Landmark** An object that is a landmark primarily because of its visual characteristics. These include its contrast with surroundings and prominence of spatial location. As an example in a hypertext environment, Sorrows uses the homepage of a typical university, which often has different images and layout than its subordinate pages.

**Cognitive Landmark** A landmark where the meaning stands out, e.g., it has typical meaning, or is unusual within the environment.

**Structural Landmark** A landmark whose importance comes from its location in space, e.g., Trafalgar Square in London, or the index page of a website.

They also emphasise that landmarks might belong to more than one of these categories, and that the strongest landmarks in an environment will be landmarks in all three categories. In the non-visual environment the first of these categories needs some reconsideration: here it is the name (or sound, or feel in a tactile or haptic world) that must stand out.

It can be seen that in the situation described above, where all nodes are of the same type, there will be no visual landmarks and it will be very difficult to discern cognitive landmarks. Although there may be structural landmarks,
these too will not be particularly useful if the user cannot differentiate the routes from them. Thus we see that the difficulties in a homogeneous environment are essentially due to lack of landmarks.

Since landmarks are so useful for navigation and orientation, which are themselves key parts of problem solving with non-visual graphs, it is necessary to consider how the graph environment may be modified to generate useful landmarks. The key factor determining if an object is a landmark is its relationship to its neighbours: ‘Thus the landmark saliency of a feature does not depend on its individual attributes but on the distinction to attributes of close features. Being a landmark is a relative property’ [77]. Node differentiation, although essential, does not mean that there will automatically be useful landmarks.

A further differentiation in landmark type that is worthy of consideration is that between local and global landmarks. In real environments there are often landmarks that are visible from most locations (global landmarks) as well as those which are only available more locally. It has been noted that both types of landmark are used for orientation, although experiments performed in a virtual environment by Steck and Mallot [92] demonstrated that some people used exclusively one or the other type of landmark, while others used both. They further noted that if one type of landmark were removed from the environment, the other type could be used by nearly all participants.

Having defined what landmarks are, how they can be classified and what makes objects salient as landmarks, it is perhaps worthwhile considering their function. From Allen’s framework above [1], it can be seen that landmarks act as familiar points that mark a route, either as confirmation (‘I am on the correct route’) or indicating a change of direction (e.g., at a junction). Distant (perhaps global) landmarks can also provide orientation and directional clues [77]. A study of landmarks by Michon and Denis [57] showed that in route directions landmarks were generally found at decision points, i.e., points in a route where changes of direction were likely to occur.

What, therefore, can be said about the requirements for landmarks in graphs? In some domains, certain nodes may have some of the characteristics that make them salient as landmarks (cf. [77]), but there can be no certainty about these nodes lying on decision points (branches in the graph). Equally, without knowledge of the user’s task, the semantic aspects of landmark saliency cannot be known. Hierarchical browsing automatically creates at least one landmark: the
‘whole graph’ level of browsing is a structural landmark equivalent to the home page example from Sorrows [90], as it has quick access to or from all other locations in the graph and may be easily returned to after exploratory travel.

If the user is to do any node-to-node browsing however, this is unlikely to be sufficient. It is therefore necessary to enable users to create their own landmarks, and to provide annotations and a user-interface that allows them to be used effectively. For a landmark to be effective, not only must it be salient, but once recognised, the user must remember which path to take (or not to take). It must be possible, therefore, for the different routes to be distinguishable from one another.

6.2 Labelling

The requirement for landmarks has been demonstrated, both above, and by the explicit and implicit labelling used by participants in the description experiments (section 3.3.2.2). It was also noted that, since their presence cannot be guaranteed, the user must be given the ability to create them. This can be considered as enabling a kind of audio graffiti — the user leaves tags at key locations that enables them to be recognised immediately. The discussion on landmarks, above, highlighted the need for them to be salient, that is stand out from their surroundings and be easily recognised. This must be taken into account when implementing user-generated landmarks in any system; a landmark is ineffective if the user needs to perform any additional actions to become aware of its status. Indeed, when a user is approaching a landmarked node for a second (or subsequent) time, it is desirable that it is recognisable as a landmark before the user actually arrives.

One mechanism which, when coupled with a suitable user interface, could enable landmark creation is node labelling. In this context a label is a relatively brief alternative name for the node. For example, if someone is exploring a family tree where several people share the similar names, giving one the label ‘mad uncle Bob’ would help that node stand out.
6.3 Visit History

A significant problem during the early exploration phase, when one is identifying landmark objects, is remembering where one is, and recognising whether one has returned to a previously visited location or not. As discussed above, some graphs can be quite homogeneous, in which case this becomes a significant difficulty. Landmark creation can help, but a certain degree of exploration is required before it becomes apparent which nodes would make useful landmarks. ‘Breadcrumb trails’ (from the Brothers Grimm fairy tale ‘Hansel and Gretel’) is a term covering a set of techniques for displaying location. This section looks at the use of ‘breadcrumb trails’ for non-visual graph exploration.

In HCI, the term Breadcrumb Trail is normally used in reference to a horizontal strip of text that gives the web browser’s current location. For example, if browsing a shop for computer components, it might be:

Store > Hardware > Storage > Hard disc drives > SCSI

This indicates where, in a hierarchically organised site, the user is located. According to Wikipedia¹, there are can be three types of breadcrumb trail: *path* breadcrumbs show the path a user has taken to reach a page (within a certain site); *location* breadcrumbs show the location within a hierarchy, as above; *attribute* breadcrumbs give information categorising the current page. Curiously, the term does not appear to be used for the closest parallel with the original story — the back button facility on web browsers.

There is no clear parallel for attribute trails in graph exploration, but path trails could potentially be very useful. There are many ways to represent node visit histories, but the information can be stored by simply annotating each node with the fact that it has been visited, when it was last visited, or the time of each visit. This simple annotation could enable powerful interaction. For example the user could be told each time a node is visited whether it is a new node or not, or a kind of back button could allow his or her steps to be retraced. In addition to the above benefits, whether a node has been visited or not is an additional differentiator that could prove useful in early exploration, particularly of homogeneous graphs.

Can breadcrumb trails be of any use for non-visual interaction with graphs? Petrie et al. [70] suggest that they are highly visual:

¹http://en.wikipedia.org/wiki/Breadcrumb_(navigation)
“Even a simple text tool such as a breadcrumb trail, actually works, if it works effectively, because the user can glance at the trail and immediately click on the place to which they wish to return.”

While the ability to use the trail with just a glance is undoubtedly important, that does not preclude its use here. In web pages, the difficulty is more that the user needs to move from his or her current position on the page, then locate the breadcrumb trail, before it can be used — a considerable investment, particularly considering that it is not trivial to regain position should the trail not be useful. In a specific graph browsing tool, however, this sort of trail may be made available through commands that can be performed without the need to change focus first.

If the nodes of the graph are annotated with ‘part of’ comments that form a hierarchical data structure, there is a clear parallel with the location breadcrumb trails available. Indeed, there was a command in Kekulé that gave such a trail (section 3.1.2), although this was not widely used.

6.4 An Evaluation of Numbering Systems

As shown above, any system for non-visual browsing of graphs will need to enable its users to distinguish between the different nodes. While some systems simply use spatial position for this (e.g., tactile displays) others, particularly those where presentation is solely through the audio channel, need a different system for unique identification. This section examines the use of numbers for this purpose, looking at how sighted users number different graphs, and evaluates a variety of node numbering algorithms.

Previous evaluations [20] have shown that a poor numbering system can be confusing (see section, users were asked to answer questions about molecules, using a non-visual browser to examine the molecular structures. This often involved moving around the graph from node to node. Because atom names were not unique (there were typically many carbon atoms in each molecule), the identifiers (integers) defined in the input CML\(^2\) file (an XML-based representation of molecular structures) were used to distinguish the atoms. These identifiers were, in turn, generated by the output routines of the molecular drawing package, and were related to the order in which the atoms were drawn. This often resulted in

\(^2\)http://www.xml-cml.org/
neighbouring atoms having very different numbers; a feature that proved confusing when browsing, for example, along chains of atoms.

As a result of the problems the evaluation highlighted with the numbering system (participants explicitly commented on the problems it caused), improved methods for identifying the nodes have been investigated. For speech output, simplicity and brevity are important [74], so only simple integer or alphanumeric systems were considered. The numbering schemes derived and evaluated are described below, followed by a description of the evaluation method and the results.

6.4.1 The Systems

This section describes the different node numbering algorithms compared in the evaluation.

6.4.1.1 FEM-based system

It appeared that the main reason for the confusion caused by the numbering system in the original evaluation was the lack of contiguity when moving from node to node. For example, a user navigating along a chain of four atoms might encounter atom numbers 2, 5, 3, then 9, in that order — a discontinuity that conflicted with their developing mental model of a continuous series of nodes. A node-numbering algorithm was therefore developed which was designed to number with maximum (or at least good) contiguity.

Numbering nodes in a graph or network is also a problem for engineers using the finite element method (FEM) for modelling. Manevitz, Gigoli and Margi [52] present a method for numbering nodes in FEM meshes which is ‘heuristic and is designed to mimic the methods of a human expert numberer’. Since they state that ‘To optimise the numbering a user tries to keep the node numbers as continuous as possible’, their algorithm was adapted and implemented for numbering nodes on a graph.

They identify essentially three stages of numbering by an expert:

1. Subdivide the mesh into disjoint blocks.

2. Choose the order in which the blocks are to be ordered.

3. Choose a strategy for each block
6.4. AN EVALUATION OF NUMBERING SYSTEMS

This strategy also reflects another issue in non-visual graph browsing, namely that of chunking. It has been shown that visual perception is hierarchical [68], and therefore that parts of a scene are ‘chunked’ (i.e. grouped together as a unit) when viewed. The effects of chunking on short-term memory (an overstretched resource when browsing through the audio channel) are also well known [58]. A numbering scheme where contiguity is maximised overall, but also maximised within the perceived chunks (sub-units) of the graph, is likely to be a significant improvement over the old CML system.

In this system the graph is searched for structural components: rings and chains of nodes. These searches use modifications of Balducci and Pearlmann’s algorithm [5] for identifying the smallest set of smallest rings. This is equivalent to stage 1 of the FEM method above, and could be replaced by other chunking techniques as appropriate for any particular graph domain. The next phase, as in stage 2 above, is to sort the chunks into the order in which they will be numbered. For this we start with the largest component, before finding the biggest chunk that is connected to it. This loop continues, each time ordering the largest chunk that is connected to one of the chunks we have already done, until all chunks are in order. Note that this is in contrast to the Manevitz method, where the largest chunk is ordered first, but the order of the chunks connected to it is that of increasing size. This was changed for this application because the FEM system seemed to give numbers that would naturally lead to small dead-ends when traversing the graph node to node, whereas it was thought more useful for users of a non-visual graph browser if following the numbers with smallest gaps would lead along the main components of the graph. The difference perhaps reflects the difference between the highly connected meshes used in FEM with the less connected graphs (such as molecules) that are being considered here.

Stage 3 is the numbering of nodes within chunks. Rings are numbered clockwise (if node position information is available), starting with the node that is connected to the previous chunk (or vice versa if the ring is the first chunk to be numbered). Rings that share nodes are numbered in the same manner, although it is obviously not possible to generate continuous number sequences round each one. Chains are numbered in order starting with the node connected to the previously numbered chunk. Individual nodes are numbered last.
6.4.1.2 Alphanumeric

One of the main sources of confusion for users performing the tasks on the molecules was disorientation, particularly when investigating structures where there were multiple rings sharing nodes; in these cases it was easy to move in circles without even realising the existence of the cycles. There are several ways of tackling this problem, including summarisation of the graph and using ambient sound to indicate what sort of environment (e.g. a ring) the current node is in, but the numbering system could also be used to give orientation clues. This algorithm moved from a simple numeric system to an alphanumeric one. Nodes that were members of multiple ring systems were numbered sequentially from 1 around the ring (with a random start point), but this was prefixed with a letter identifying the ring. Non-ring nodes were numbered with the above algorithm (i.e., numbers only). If a node were a member of two or more rings, the identifiers would be concatenated. Figure 6.1 gives an example graph numbered with this system.

![Figure 6.1: Graph showing nodes numbered using the alphanumeric system](image)

6.4.1.3 Radial

For the radial system, the node closest to the centre of the graph (geometrically speaking, as presented in 2D) was numbered 1. Then starting from 0° (towards the top of the page) and moving clockwise, the nodes that were connected to it were numbered 2, 3, . . . etc. The numbering continued in the same way from nodes
6.4. AN EVALUATION OF NUMBERING SYSTEMS

connected to node 2, then 3, and so on, until all nodes had been numbered. This resulted in a system where the size of the number loosely reflected the distance (in terms of number of arcs) of the node from the geometrically central node. Figure 6.2 gives an example of a graph numbered using this system.

![Graph showing nodes numbered using the radial system](image)

Figure 6.2: Graph showing nodes numbered using the radial system

6.4.1.4 CML

In this system, the graphs were generated using a drawing package that exports to CML, and the identifiers from this CML file were used. This system was evaluated to provide a minimum standard: since users in the Kekulé evaluation (section 3.1) found this system confusing, any replacement system must perform better than this.

6.4.1.5 Random

In this system, the nodes were assigned random numbers. This was done by generating a randomly ordered list of numbers\(^3\) that were assigned to the nodes using the numbers generated by the FEM-based system to decide the order.

6.4. AN EVALUATION OF NUMBERING SYSTEMS

6.4.1.6 Geometric

The geometric numbering system takes no consideration of the topology of the graph, instead deriving the numbers from the layout of the nodes when drawn. In this case, numbering was performed from top left to bottom right in rows. This was done quite strictly, in that two nodes were only considered as being in the same row if their y-coordinates were equal.

6.4.1.7 Trees: breadth-first

While devising the numbering systems described above, it was noticed that none of them particularly suited a large class of graphs — trees. As a result, it was decided to investigate whether people would naturally number graphs from this domain differently from other graphs. One tree was included in the evaluation and, instead of the CML system and the random system, participants were given two numbering schemes devised purely for trees. The first of these numbered breadth-first, i.e., the root node was first, then all its children, then all the grandchildren, etc., with numbering within a generation determined by position when drawn (left to right or top to bottom, depending upon the orientation of the tree).

6.4.1.8 Trees: depth-first

The second system for numbering trees was depth-first, first it numbered the longest chain from the root, then the next longest remaining chain, and so on.

6.4.2 Method

The primary aim of the evaluation was to determine which of the numbering systems outlined above was preferred by the participants. However, it was felt that it would be beneficial to get participants thinking about the problem before asking them to choose a solution, and that it would be useful to be able to identify any other strategies that might be particularly popular. The evaluation was therefore divided into two sections; in the first participants were presented with a series of six graphs (Figure 6.3) and asked to number the nodes. The only constraint was that the nodes were to be numbered with sequential integers, starting from 1. In the second section participants were presented with a graph numbered with five different systems, and asked to rank them in order of preference. Nine graphs were presented in section two (Figure 6.4). Finally, the participants were given
the opportunity to comment. The evaluation used abstract non-directed graphs with between eight and seventeen nodes. These were based on graphs from a variety of domains, including molecular structure diagrams, flowcharts and UML diagrams. They were selected to cover a range of structures, but sizes were constrained so that the evaluation was not too long or boring for participants.

The evaluation was web-based, using php and Java applets; participants completed the questions unsupervised from their own computers. More details, including the instructions for participants are given in Appendix D.

6.4.3 Results

Thirty-six participants started the evaluation, thirty-one completed it. Most participants were from the Department of Computer Science at Manchester University, including undergraduates, postgraduates, and postdoctoral staff, although there were a number of other volunteers, all with a similar level of education. None of the participants were visually impaired.

6.4.3.1 Section 1: Free Numbering

This section discusses the first phase of the evaluation, where participants were given the freedom to number the nodes in any order they wished. In a broad sense, two types of strategy appeared to have been used; those where numbering was purely based on the geometric positioning of the nodes (typically, but not always, rastering top left to bottom right), and those where features inferred from the topology (such as rings) were used to break the graph into smaller chunks.

There was a notable tendency for the numbering to start at a node located near the top left of the diagram. One may presume this is related to the culture of reading from top left to bottom right that the majority of participants shared. It was observed that this was less common when the graph was a tree, in which case the majority of participants started at the root node. There was, however, still a preference for numbering from the top; in the tree with the root at the top 25 of 35 participants started at the root, while when a slightly different tree was presented with the root at the bottom, only 20 participants started at the root node. With the phenylalanine structure, 29 of 36 people started within the ring, the majority starting at either of the two top left nodes (17) or the bottom node (7), the latter allowing them to continue numbering from the ring along
6.4. AN EVALUATION OF NUMBERING SYSTEMS

Figure 6.3: The graphs used in phase 1. The numbers shown are for reference, and were not visible to participants. Graphs are taken from real examples in a variety of domains; the range of sizes was limited by a desire to keep the tasks relatively simple for the participants.
6.4. AN EVALUATION OF NUMBERING SYSTEMS

Figure 6.4: The graphs used in phase 2, annotated with the most popular numbering system. Graphs are taken from real examples in a variety of domains.
Figure 6.4: The graphs used in phase 2, annotated with the most popular numbering system.
the connected chain. The choice of starting node for the polycycle varied widely, with the most popular choice, the upper node of the short chain, being chosen by only 7 people. Even in this case, there was still an observable top-left bias, in that where people numbered one of the three smaller rings continuously, it was nearly always the top left one.

Although the detail of the systems people used varied, there was more consistency with how they appeared to break the graphs into sub-units, or chunks. Rings were an obvious feature that the majority of participants numbered as a unit, although the ring present in one of the trees was typically ignored and a ‘standard’ tree system used (only four people broke this graph into the ring plus the two side chunks, one of whom was a professional organic chemist). Where it was not possible to use individual rings as chunks (in the polycycle), three main strategies emerged. One was to number the outer (large) ring continuously (5 people), another was to number one of the rings continuously, then fill in the remaining nodes (15), while the third was to number round the nodes in a snake-like path that allowed all polycycle nodes to be numbered in a continuous sequence (8 people). The latter inevitably ended up with at least one ring having its nodes numbered in sequence. We can infer from this sort of system that continuity is considered desirable.

It is also possible to examine how the chains were chunked in the phenylalanine graph. Eleven participants divided the chains into one of the four possible 4-node chains plus two single nodes, all but two of those using a7, a8, a9, a10/a11, but much more popular (20 participants) was to divide the nodes into two 3-node chunks, i.e., a7, a8, a12 and a9, a10, a11.

After seeing how the graphs were divided, we can look at how common chunks were numbered internally. There was a definite tendency for rings to be numbered in a continuous sequence. Of the 36 participants for the phenylalanine graph, only five did not number the ring nodes as a group, and only five of the remainder did not number the nodes in sequence. Of the twenty-six that did, the majority (21) numbered clockwise. A common strategy was to start at a3 and number clockwise, thus allowing numbering to continue uninterrupted along the chain. Clockwise numbering was also predominant in the different strategies used for numbering the polycyclic.

The systems people used for chains were interesting, particularly where chains of three nodes were connected to the rest of the graph from the central node. In
the two trees, the numbering used for these three nodes was merely whatever fitted with the general strategy (depth-first or breadth-first in most cases), but in the polycycle we see different tactics. Where the three nodes were numbered as a chunk (32/35 participants) it was most common to number the central node first, then the two connected nodes (or occasionally vice versa), while few people (7/32) chose to number the three nodes as a continuous chain. Note that this was often the case even when the number given to the central node did not directly follow the node it was connected to.

6.4.3.2 Section 2: Multiple Choice

Each participant ranked 5 possible numbering systems for each graph. The numbering systems used for each varied slightly, but may be inferred from table 6.1 (each graph has 5 scores — these are the systems used for that graph), while the descriptions of the algorithms in section 6.4.1 allows one to recreate the graph used. In order fully to compare preferences, each system was given a score for each graph, calculated using the following formula, where i is the rank (1 representing first choice) and \( n_i \) the number of participants who gave that system rank i.

\[
score = \sum_{i=1}^{5} \frac{100(5 - i)n_i}{4N}
\]

This scoring system was intended to measure the overall popularity of a system and indicate how much consensus there was; the scores represented popularity as a percentage of the maximum achievable (i.e., a system that everyone ranks as best would score 100) and always added up to 250. A range of scores indicates a consensus that the top scoring system is best, while five similar scores indicate that there was no overall preference for any one system. Table 6.1 gives the results. No score is given if that system was not presented to the participant. The alphanumeric system was only given on those graphs with multiple fused rings (it would otherwise be identical to the FEM-based system); in other cases, the radial system was presented.

Comments from some of the participants suggested that they found ranking hard after the first two or three choices, so their lower ranks were more or less random.

If the tree (figure 6.4(a)) is considered first, since this had different options available, it can be seen that there was a strong preference for numbering in
6.4. AN EVALUATION OF NUMBERING SYSTEMS

Table 6.1: Numbering system scores for the nine graphs presented in the evaluation. A blank cell indicates that that system was not a choice.

<table>
<thead>
<tr>
<th>Graph</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>FEM-based</td>
<td>55</td>
<td>89</td>
<td>76</td>
<td>98</td>
<td>89</td>
<td>88</td>
<td>67</td>
<td>86</td>
<td>70</td>
</tr>
<tr>
<td>Alphanumeric</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Radial</td>
<td>13</td>
<td>48</td>
<td>46</td>
<td>61</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>CML</td>
<td>18</td>
<td>31</td>
<td>44</td>
<td>13</td>
<td>26</td>
<td>31</td>
<td>19</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>19</td>
<td>12</td>
<td>9</td>
<td>22</td>
<td>13</td>
<td>35</td>
<td>12</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>Geometric</td>
<td>25</td>
<td>48</td>
<td>83</td>
<td>53</td>
<td>65</td>
<td>46</td>
<td>69</td>
<td>78</td>
<td>76</td>
</tr>
<tr>
<td>Trees: breadth-first</td>
<td>88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trees: depth-first</td>
<td>69</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

generations from the root (breadth first), indeed 22 people from 33 selected this as their first choice. Second most popular was the system that numbered in chains from the root; this was still the first choice of 7 participants.

Looking now at the overall picture given by the rest of the results, it can be seen that the FEM-based system is consistently popular, while the random numbers and CML system are consistently unpopular. Interestingly, the only time where either of these generally unpopular systems was chosen as best was in one of the more complex graphs — graph 7 — a complex graph for which there is little consensus. It is suspected that the participants did not like any of the systems presented and so ranked randomly.

The geometric numbering was clearly a popular choice, particularly on more complex graphs, although it did not suit all graphs. It appeared to be more popular when the connectivity of the graph suited it, i.e., if the connectivity meant that numbers given would be highly discontinuous when moving between nodes, then it would be less popular. For example, geometric numbering was more popular on graphs 3 and 5 (where there is strong horizontal connectivity; it got 12 and 8 first choices respectively) than on graphs 2 and 4 (where there are more diagonals; it was only first choice for one person in each).

The use of alphanumerics for numbering multiple fused rings had mixed popularity. For simple graphs, such as the double ring system of graph 2, most people selected the alphanumerics system as first or second choice (12 and 15 people respectively from 33). The most popular choice was the FEM-based system, probably because this gave a numbering which was continuous around the graph. The situation was very similar for the slightly more complex graph 6 with three rings, but with the more highly connected graph presented in two different forms.
as graphs 7 and 9 opinion was divided. For both of these graphs, 7 people selected this system as their favourite, while 12 selected it as their least favourite. Some of the participant’s comments reflected this; at least two people remarked that they liked the alphanumeric system for relatively simple graphs, but not for more complex ones. One possible reason for the dislike of some people for this system, despite the extra information, is the complexity of the numbers, and hence the ease of reading or understanding them.

The radial system, where numbers represent the distance from the most central node was typically the third most popular choice, i.e., more popular than the essentially random systems, but generally not as popular as the others.

### 6.4.4 Discussion

When considering the results of this evaluation, it is important to bear in mind that both the numbering applied by the participants, and the systems preferred by them, were selected visually. Although this is a limitation for applying the findings in a non-visual browsing system, it has given a valuable insight into how important it is for a system to be understandable to users.

The results suggest that the algorithm developed from the Finite Element Modelling system generates node numbers that are considered sensible by the participants. There is some requirement perhaps for different systems to be used for different domains. In the case of trees for example, a choice between breadth-first and depth-first numbering schemes would keep most users happy, while directed graphs would also demand a system that observes arc direction. Application of different schemes for different domains should be followed with caution, however, particularly if the same user is to be browsing different domains; unique numbering schemes should perhaps only be used where the graph structures are definably distinct.

Evidence from the first phase of the evaluation suggests that the chunking algorithm within the FEM-based system needs to be examined. For example, there was a strong preference for breaking the branched chain section of the phenylalanine graph into two chains of three nodes rather than a chain of four and two single nodes. Analysis of these systems shows that the chunking preferred by participants does, in fact, have more continuous numbering, as measured with the following formula, which gives the mean difference in node numbers at the end of each arc (a higher number indicating lower contiguity). In the formula N
is the number of arcs and $n_{i1}$ and $n_{i2}$ are the numbers of the nodes at either end of arc $i$.

$$discontiguity = \frac{1}{N} \sum_{i=0}^{N} |n_{i1} - n_{i2}|$$

Using this, the mean discontiguity of the four node chain plus two individuals is 1.67, while that for the two three-node chunks is 1.50.

Chunking is an area where domain specificity can play an important role, as there may be natural sub-units into which the graph can be divided, based upon the properties of the nodes and arcs; division of molecules into functional groups is an example. This would be particularly important if the browsing system enabled hierarchical browsing — less disorientation would be likely if sub-units were numbered contiguously.

People’s comments suggested that they preferred easy to understand systems, particularly when the graphs become more complex. It would not be wise for a graph browsing system to use different systems for graphs of different complexities, however, so one system must be identified for all levels of complexity. Unfortunately the more popular system for the more complex graphs, the geometry-based numbering, is not really suitable for non-visual browsing. Although generating a number from the position is easy, and understanding the system is easy when viewing the whole graph, the operation is not reversible, i.e., the position of a node cannot readily be implied from its number. Of the systems presented in this evaluation, it is the FEM-based one that is perhaps most widely applicable, particularly if the use of alphanumerics could be determined by the user. It would be interesting to see how people’s preferences developed over time spent using a browser — would the complexity of the alphanumeric become more useful as the difficulties of keeping orientated while navigating a complex ring system outweigh the complexity of the label (exacerbated by it being presented as speech)?

It is possible that the tendency to number the 3-node chains from the central node was influenced by the spatial layout of the nodes; the three nodes were arranged in a ‘V’. It would be interesting to investigate people’s methods if the nodes were arranged in a straight line, or if the chain was longer or asymmetric.
6.5 Summary

Distinguishing nodes from one another is essential if one is to explore a graph; without this ability even the most basic level of orientation becomes very difficult. Furthermore, landmarks are known to be important for moving around an environment and communicating directions. We have seen that, for a place to be a landmark, it must contrast with its surroundings, either visually, in terms of its meaning, or its location. Similarly, for nodes in a non-visual graph space, a node needs to be distinct if it is to become a landmark.

The most basic level of distinctness is whether a node can be distinguished at all from its neighbours. In some domains, this is necessarily the case but, in others, nodes may have identical names. While this is not problematic when presented visually — they are distinct also in their location — some additional means of identification is required when presented aurally. The simplest is numbering, but the Kekulé evaluations demonstrated that problems arose when poor numbering systems were used. An evaluation of several numbering systems on abstract graphs showed that when graphs were numbered (visually) different systems were preferred for different graph types. For example, tree-like graphs were numbered from the root node, while very complex graphs were numbered geometrically. A system based on node numbering schemes for finite element models was found to be a good general-purpose solution, although domain-specific numbering may be preferable.

Although numbering nodes means they that may be distinguished unambiguously, it does not necessarily mean that they form good landmarks. After all, the node Carbon 15 is not so different from nodes Carbon 14 and Carbon 16 that it has high landmark saliency. Landmarks are also most useful if they lie at decision points in an environment. Allowing users to annotate nodes with their own labels allows nodes to have a much higher landmark saliency, and labelling can be done at any node in the graph, decision points included.

An additional problem, encountered in the Kekulé evaluation, is that can be difficult to identify whether one has visited a node before, or is exploring a new region. Annotating nodes with information about when they have been visited could enable a user interface to communicate this to the reader. Breadcrumb trails, more in the original sense (like the back-button on web browsers) than the current HCI usage, could also be used to allow a confused reader to retrace his or her steps to a familiar location.
Chapter 7

Implementation

This chapter describes how a tool has been implemented that allows non-visual exploration of graphs. This application uses Semantic Web technology to implement many of the types of annotation discussed in the previous three chapters, so that they may be evaluated. The primary purpose of the implementation was to enable an evaluation that would answer the main research question, i.e., in brief, to test the hypothesis that annotation makes browsing easier. The first section examines what is required to represent a graph and annotations. It then gives a short background to the Resource Description Framework (RDF), demonstrates how it fulfils those requirements, and details how it was used to describe and annotate graphs. The second section gives the particular set of annotations that were implemented for the evaluation. This evaluation, described in detail in the next chapter, used graphs from two domains — logic circuits and family trees. While the principles being tested, and indeed the software used for their testing, are expected to be generic, some types of annotation were fine-tuned to each domain; these differences are discussed here. The final section of this chapter explains the user-interface. A running example is used to give an impression of the system in use.

7.1 Representing Annotated Graphs

Any tool that is to support annotation of graphs so that visually impaired users may explore them more easily, must have internal representations of the graph and the notes. How to do this gives our fifth research question: What kind of representations will support the variety of annotations discussed in the preceding
chapters so that they may be available for effective presentation to the user?

In order to answer this question, it is necessary first to identify the requirements for the representation, as demanded by the types of task, and annotation, any system will need to support. The tasks, and classes of annotation that support them, are outlined in the taxonomy as presented in table 4.1. The following requirements are proposed:

- The semantics of a graph must be fully described. In other words, the representation must describe all the nodes and their properties (name, etc.), all the arcs and their properties, and which nodes are connected by which arcs.

- The representation must be able to describe node positions. Some forms of annotation use this information (if this is available), so it must be possible to hold it.

- The representation of the graph must allow referencing of its constituents so that they may be annotated, i.e., the representations for the graph and the annotations must be compatible.

- Annotations may refer not only to individual nodes, but also to groups of nodes, or even the entire graph — the representation must support this.

- Annotations must be able to take different forms, including numbers, words and sounds. Of course, there are many ways to represent sound.

- The representation of the annotations must be dynamic, i.e., it must be able to have information updated during exploration. This is to support both user annotation, and automatic annotations such as visit histories.

In addition to the above, three further characteristics of the representation may be identified that are desirable, if not strictly required.

- Representation of any type of graph.

- Extensibility of the types of annotation beyond those identified here.

- Persistence of the representation, such that users may store annotated graphs.
It may also be noted that, unless they are to be stored, some annotations need not be held ready for use, but may be calculated when required. Since the medium of presentation, sound, is transient, some annotations (namely those that do not change over the course of exploration, e.g., the audio glance) may also be transient, calculated on the fly. Similarly, all annotations need not be created at the start of exploration; instead the graph may be annotated lazily, generating the notes when first requested, then storing for future use.

Considering all the requirements above, plus the usual desirables such as simplicity, it is hypothesised that a graph-based framework that can represent both the graph and its annotations might be an appropriate basis for a graph exploration tool. In particular, the Resource Description Framework (RDF) appears to fulfil the requirements: it is designed to do precisely what is required — describe resources. Furthermore, as RDF is itself graph-based, it naturally describes graphs.

7.1.1 RDF and the Semantic Web

The ‘semantic web’ is a term that captures the goal of many researchers — a world wide web where the meaning of web content is not only human-readable, but is also computer-processable. The idea is that intelligent services will be able to process web content not only in terms of content and layout, but also in terms of meaning, and will therefore provide greater functionality than at present. Critical to the whole semantic web vision is metadata: information about the content.

Metadata is data that describes other data, in this context data that describes web content. For example, the name of the author and date of creation are simple metadata; more sophisticated examples are pointers to related documents, and definitions of what inputs are required for a form. If the vision of a semantic web is to be realised, metadata must be provided in a standard form that can be read and processed automatically, and ideally support inference-making. RDF has been developed by the World Wide Web Consortium (W3C) to fulfil this role. The critical role envisaged by Tim Berners-Lee for RDF in the semantic web can be seen from his diagram, reproduced in figure 7.1.

RDF is simple in concept: resources are described using statements (known

\footnote{http://www.w3.org/RDF/}

\footnote{From a presentation entitled “XML and the Web”, found at http://www.w3.org/2000/Talks/0906-xmlweb-tbl/}
7.1. REPRESENTING ANNOTATED GRAPHS

Figure 7.1: The place of RDF in the Semantic Web, from http://www.w3.org/2000/Talks/0906-xmlweb-tbl/

as triples, or tuples) formed from three parts:

subject; predicate; object

In other words, a triple defines what is being described (the subject), which property is being described (the predicate), and the value of that property (the object). For example the triple ‘cat; colour; black’ is equivalent to the statement ‘the cat is black’. There is a specification for representing these triples in XML [107], but the essence is simple and powerful. A key requirement of this method of description is that all subjects need to have unique identifiers.

An important feature of RDF is that it is a graph (the W3C primer on RDF states “RDF is intended to be used to express statements about resources in the form of a graph, using specific vocabularies” [105]) and may be presented as such. RDF graphs show triples as arrows between boxes: each arrow flows from the subject to the object, with the arrow label describing the predicate. Traversing such a graph enables statements to be related but, if this is to be done automatically, the meaning of the predicates must be defined. This may be done by a schema [106]: a document that defines how an RDF document can be structured, essentially describing the metadata (RDF) that describes the data. A schema defines classes of resource and the predicates that can be used
to describe them; it can also restrict how these are used, e.g., which resources a predicate may describe. If the vocabulary is to be tightly controlled and defined, it is possible to use terms from an ontology as predicates; this would then allow inferences to be made automatically.

### 7.1.2 Annotating Graphs with RDF

Annotation, as we have seen in chapter 4, is a method for supplementing information. In the case of graphs, the original information is the graph itself, and the annotations supplement this with information about individual nodes or arcs, or about groups of nodes and arcs, or about the entire graph. RDF is a framework for describing things. Although a new form for representing annotations for graphs could be invented, RDF offers an existing option, one that has the potential for being integrated with other semantic web technologies, should that prove useful. In essence, all annotations are a form of description and can be phrased in the form of a triple. The subject is the node (or arc, graph, etc.) and the object is the information given by the annotation, while the predicate is the relationship between the two — what the annotation is meant to convey.

As examples, consider some of the annotations in figure 4.2 as triples (subject; predicate; object):

- Henry Brown; number; 1
- Henry Brown; relationship to home; grandfather
- Graph; summary; 13 people in 4 generations, etc.
- Andrew Brown; is home node; true
- Jean Pearson; in family; Pearson family
- Jean Pearson; in family; Brown family

Note that sometimes the object may be a collection, so to keep objects singular a natural language statement must be split into several triples. For example the statement ‘this graph contains a carboxylic acid group, which is made of nodes 1, 2 and 3’ (annotating the graph with the explicit presence of a chemical functional group) could be expressed in triples by using an entity (collection 1) that may be kept hidden from the user:

- Graph chunk A; has name; carboxylic acid
- Graph chunk A; contains nodes; collection 1
7.1. REPRESENTING ANNOTATED GRAPHS

For such a system to be effective, the subjects, predicates and objects of each statement need to be well defined. Subjects are typically entities within the graph (including the graph itself); predicates need to come from a defined list (e.g., a schema) so that it is understood what they are and how the user-interface may deal with them; objects may also be graph entities, or more simply numbers or text. Since the set of predicates needs to be known in advance, they must be sufficiently general to cover all circumstances on a range of graphs, but specific enough to be meaningful. Appendix E contains those predicates used in this application (further standard predicates were also used, including those such as collection, and some ‘Dublin Core’ elements 3).

Does RDF fulfil the requirements for annotation given in section 7.1? This question can be divided into two sub-questions: first (and most important), does the framework of subject-predicate-object triples allow graph annotation as required?; second, does the W3C specification for XML representation of RDF pose any barrier to its implementation? The answers to these questions are that RDF is suitable. Taking the requirements for annotation in turn: the use of collections allows nodes or arcs (or both) to be grouped and the group to be described; objects in triples can take the forms of numbers, or text, hence XML, or a musical score, etc.; there is no reason why the RDF graph cannot be modified or updated while the graph is traversed. The further desirable — persistence — can also be achieved using the XML representation of RDF, given a suitable form for representing non-speech sounds (e.g., MIDI).

Since RDF is an existing technology that is essentially designed for annotation, and one that clearly fulfils the requirements, it appears an obvious choice. Before considering the important question of whether it is also capable of representing the base graph, thereby allowing a simple, single, coherent framework for representing annotated graphs, it is worth exploring how effective, or otherwise, RDF could be as the basis for a user interface. An application that uses RDF to record annotations needs to present them to the user in a clear manner. The method of presentation is, of course, independent from the internal representation, but it is worth commenting on how the UI-RDF interface allows a variety

3http://www.dublincore.org/
of approaches. Using RDF, the graph objects must be well defined, and predicates are also defined, either with a schema or using an ontology. It is therefore possible to search the set of RDF statements for a graph, either for a particular predicate (form of annotation), or a particular part of the graph. For example, any annotations about a particular node can be found by searching for statements where that node is the subject; this would return notes such as number or label. Statements where that node is the object would also need to be found for a full picture; these would return information such as the chunks of which it is part. More efficient searching can be performed if a known form of annotation is required, since the search can have defined subject (or object) and predicate.

As an example consider the results of a search of all RDF statements for Node A in a graph representing a molecular structure (to simplify the triples, search results showing Node A as part of a collection have been expanded to show what the collection node represents):

Node A; name; Carbon
Node A; number; 4
Node A; label; alpha-carbon
Chunk 4; nodes; collection 2; has item; Node A
Node A; property; property 3;

Thus we discover that Node A is a carbon and has been given the label ‘alpha-carbon’ and been numbered 4. Node A is part of Chunk 4. It also has a property, the meaning of which may be found by searching for triples with ‘property 3’ as the subject:

property 3; name; hydrogen count
property 3; value; 2

We have now discovered that this carbon atom has a hydrogen count of 2. This ‘has property’ predicate, combined with statements of the type seen above, allows general properties to be expressed about items in the graph that could not be anticipated. In this manner, a list of RDF statements can represent annotations on a graph and, although this might be a verbose representation, annotations can be found through simple searches.
7.1.3 Describing Graphs with RDF

If RDF is suitable for describing the annotations that can be applied to a graph, can it also be used for describing the graph itself? A single format for describing the structure of a graph and its annotations has the great advantage of simplicity over mixing two representation formalisms. Is RDF capable, however, of representing graphs in a way that fulfils the requirements given in section 7.1?

It is not difficult to imagine a set of triples that, although long-winded, precisely describe a graph. This could be done with three classes of resource: node, arc, and graph, and properties to describe the set of nodes and arcs in a graph, the names of nodes and arcs, the set (pair) of nodes each arc connects, etc. Returning to the example used above, what might the results of a search of all RDF statements for Node A look like if the graph structure is also expressed in RDF? The following triples might be found (collections etc. are shown as before, to make this more clear):

Graph; nodes; collection 1; has item; Node A
Chunk 4; nodes; collection 2; has item; Node A
Node A; name; Carbon
Node A; number; 4
Node A; label; alpha-carbon
Arc 2; ends; collection 5; has item; Node A
Arc 4; ends; collection 3; has item; Node A
Node A; property; property 3;

Thus we discover that Node A is a node in the graph, and has two arcs connecting to it. The other ends of the arcs may be discovered by searching for triples with Arc 2 (or Arc 4) as subject and ‘ends’ as predicate, then finding the members of these collections. The same search also gives us the annotations, as before.

The RDF description of a graph and annotations can be presented as an RDF graph. This shows triples as arrows between boxes: each arrow flows from the subject to the object, with the arrow label describing the predicate. The portion of the RDF graph describing the molecular structure graph relevant to the excerpts above is given in figure 7.2. This graph demonstrates not only that this is apparently a complex and verbose way of describing a graph and its annotations, but also the benefits of representing both together — any information that needs
to be presented to the user may be gleaned by traversing this single graph. Note that this traversal is by the system in order to find information to present to the user, who is traversing the real graph. Despite its apparent complexity, however, since the types of entity and the predicates are known by the system from the schema, this traversal is simple and can be automatic. The application developed for the evaluation used RDF to represent the graph and the annotations. The RDF schema defining the vocabulary actually used is given in Appendix E.

Can RDF meet not only the requirements for representing annotations, but also those for representing the underlying graph? The fact that it is itself a graph suggests that it is suitable, and the examples above suggest that RDF is flexible enough to allow description of nodes and arcs, their properties, and their connectivity. It is obviously simple to have predicates for describing certain

Figure 7.2: Part of an RDF graph for a molecular structure diagram. Triples are represented as arrows from subject to object, labelled with the predicate. RDF describing the base graph is drawn in black while that describing the annotations is blue. This selection shows the statements about Node A; it also shows that Node A is connected to Node B and one other node.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{rdfs.png}
\caption{Part of an RDF graph for a molecular structure diagram. Triples are represented as arrows from subject to object, labelled with the predicate. RDF describing the base graph is drawn in black while that describing the annotations is blue. This selection shows the statements about Node A; it also shows that Node A is connected to Node B and one other node.}
\end{figure}
layout attributes, such as node coordinates; the second requirement is met. RDF clearly meets the third requirement — since the same framework can be used for describing both the graph and its annotations, there will be no compatibility problems. Furthermore, as far as we know, RDF can be used to represent any type of node-arc graph.

Thus we see that the Resource Description Framework has the characteristics required to make it an appropriate means for representing annotated graphs as a basis for non-visual exploration. While other representations may also be suitable, it was felt that an exhaustive search and comparison of different techniques was unnecessary: RDF fulfils all requirements and enables an implementation to be built for evaluation.

7.2 Annotations Used

Not all forms of annotation described in the previous three chapters have been implemented for the evaluation. This section lists those used and gives some detail about how they were implemented. First, however, it is worth briefly mentioning those that were not implemented, explaining the reasons.

The form of annotation that is probably the most important — making explicit the presence of implicit features (that have been automatically recognised) — has not been tested. Kekulé used a combination of hierarchical and connection-based browsing to present features of molecules (functional groups, rings and chains). Although there were some issues arising from this evaluation — in particular how to successfully combine the two viewpoints — the benefits of presenting these features explicitly were clear (chapter 3). The most striking example was the near-immediate identification of a complex molecule from three commands. Since both theory and practice suggest this to be a powerful form of annotation, it was decided not to evaluate it again: its large effect was likely to make it difficult to evaluate other forms of annotation without a balanced experiment using many participants — difficult to achieve when recruiting visually impaired participants. Another form of annotation not implemented was graph similarity, due to the difficulties described in section 5.4.2. Deedling was implemented, but was not included in the evaluation in order to keep the interface as simple as possible.

The following types of annotation were both implemented and available for
7.2. ANNOTATIONS USED

the evaluation.

**number** In order to assist node differentiation (as discussed in section 6.4), each node was given a number. Numbers were simple integers allocated using a domain-dependent algorithm. For family trees, the people were numbered breadth-first (the most popular choice for trees in the evaluation: see section 6.4.3.2. For logic circuits, the inputs were first, then all nodes connected to the inputs, then nodes connected to those, etc.

**home** Any node could be allocated as home. Initially this was the node numbered 1, which was the node from where users started exploration.

**label** Any node could be given a label (see section 6.2), which was automatically spoken after the name, each time the node was encountered (e.g., in a list of connections).

**note** Any node could have a note attached, accessible by request.

**visit histories** Each node was annotated with the time of each visit (see section 6.3). This was presented to the user by playing a non-speech sound each time a previously-visited node is revisited.

**location** Each node was annotated with its location, based on a simple $3 \times 3$ grid covering the whole graph, and expressed as top-left, middle, bottom, etc.

**audio summary** The graph was annotated with a sonic-summary, based upon the audio summaries described in section 5.5. The summary is generated by a domain-dependent algorithm and can be listened to from any point in the graph. Family trees were sonified by grouping into generations — a beep represented each person, with those in the first generation played first, those in the second generation after a short pause, etc. Logic circuits were sonified from input to output, essentially the same order as numbering, with nodes grouped by distance (in arcs) from input.

**complexity** The graph was annotated with a measure of its complexity (see section 5.2). This is reported as “X nodes, mean complexity Y”, where X is the number of nodes in the graph, and Y is the mean complexity. This is
calculated using a method based on Randić and Plavšić’s Augmented Valence Complexity index [76]. The value for each node is calculated without taking symmetry into account; $Y$ is the mean of all nodes.

**list summary** An overview of the size of the graph and the types of nodes it contains was given by categorising the components of the graph in a domain-dependent manner; this summary also allows access to all nodes (section 5.3). The summary was presented as a hierarchical list, the first item was a general summary, the rest gave the number of nodes in each category. These list items are also lists themselves, giving finer-grained detail, until individual nodes are reached. Tables 7.1 and 7.2 give examples for the two domains. In these tables, each mention of a specific node allows the user to query that node, including its properties and connections. This arrangement of lists matches the user-interface model, which will be described in the next section.

**generation** Each node in a family tree was annotated with its generation, generation 1 being the oldest. This was to make explicit information that is implicit in the graph, but can be explicit in well laid out family trees. Since this is a domain-specific annotation, the annotation is added as a property of the node, rather than as a note with a class of its own.

**relationship** Nodes were annotated with a description of their relationship to the current home node. In family trees, this was expressed as a natural relationship (uncle, grandfather, etc.) or expressions such as ‘related by marriage’ for more distant relationships. In logic circuits, the relationship was described in terms of number of arcs distance, and whether the home node was up the circuit, down the circuit, or on a parallel path.

**wire direction** Logic circuits are not drawn as directed graphs, but have an implicit direction. As part of the phase where the RDF description of the graph was generated, the directions of the wires were calculated. This annotation was used when describing connections (e.g., ‘from Input 2’).
### Table 7.1: List summary example for family trees. This is part of the summary for the graph shown in figure 4.2

<table>
<thead>
<tr>
<th>Summary list</th>
<th>sub-list</th>
<th>sub-sub-list</th>
</tr>
</thead>
<tbody>
<tr>
<td>13 people in 4 generations</td>
<td>2 in generation 1</td>
<td>Henry Brown</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thelma Stevens</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 in generation 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 Brown</td>
<td>Henry Brown</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Susan Brown</td>
<td></td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>3 Pearson</td>
<td>Raymond Pearson</td>
<td></td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 7.2: List summary example for logic circuits. This summarises an imaginary circuit with a total of 8 nodes, including 1 input, 2 Or gates, and 5 others (not shown here).

<table>
<thead>
<tr>
<th>Summary list</th>
<th>sub-list</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 nodes in total</td>
<td>Input</td>
</tr>
<tr>
<td></td>
<td>Or 1</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td>1 Input</td>
<td>Input</td>
</tr>
<tr>
<td>2 Or</td>
<td>Or 1</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>


7.3 The User Interface

The user interface is entirely distinct from the use of annotation, but needs to be carefully designed in order to make best use of that annotation. This section describes the user interface designed for evaluating the benefits of annotation. The evaluation of Kekulé illustrated some difficulties that can be caused if the user interface is confusing. In that evaluation, the biggest interface difficulty was the ease of movement — whenever the name of an atom was spoken, the focus moved to that node. Some, but not all, of the participants found this confusing; for example when browsing the list of connections to a node, the focus moved between each of the connections but users often imagined focus remaining on the original node.

This implementation shares many of the design principles of Kekulé. It is essentially based on the spatial model, i.e., users envisage themselves moving around the graph from node to node. Since annotation of groups is not being evaluated, the data structure is flat: there is no hierarchy. Thus the user starts on a node, and moves from node to node, with the focus always on one specific node. Although some domains might require arcs to have the same status, this was not the case in the logic circuits or family trees used for this evaluation. Questions could be asked of the focus node or the whole graph at any time. As with Kekulé, user input is via the keyboard and output is audio: a combination of synthetic speech and non-speech sounds. There is a very basic visual interface designed to help development of the software, but this was not visible to participants.

It was decided to keep the move-as-you-hear model employed by Kekulé, despite the difficulties encountered in its evaluation. The first reason for this was that some users had no problems using it, notably the visually impaired user and another blind person who did some piloting; it was therefore considered useful to use this evaluation, with a greater number of visually impaired users, to compare these with sighted users. It was hypothesised that the apparent difference between sighted and visually impaired participants might be significant, and explicable by how they can be forced to work differently. For example, when exploring a physical space, it is generally trivial for a sighted person to scan the space from their start position before selecting where to go; a blind person, on the other hand, is forced to actually move around the space to explore it. Similarly, when using a screen-reader, text can only be spoken if the cursor is moved to it. Thus the idea of constant motion for exploration could be more natural to them.
than to the sighted participants. Secondly, it was hoped that other refinements to the interface could reduce the difficulties.

The interface is based on a model of linked lists. At any point in the exploration the user is viewing a list of items and can move up and down this list, or back to the previous list. If the last list item spoken was a node, the user is now ‘on’ that node, i.e., the focus has moved, and any queries will relate to that node. Any commands (other than those moving up and down the list, or back to the previous list) will generate a new list and move to its first item. It is possible to move back to previous lists, but not forward again.

Some lists will be simple, containing one item (e.g., the name or number of the node), others will contain multiple chunks of information (e.g., properties of a node), while others contain a set of nodes. In all of these, the effects of moving up and down the list are the same — if the name of a node is spoken, the focus moves to that node; any command issued will be processed with reference to that focus node. There is, however, one subtle difference between these types of list: if a list contains no nodes the focus node cannot change. Issuing a new command will therefore replace the list with a new list containing the results of that new command. This means that information-only lists are not retained in the set of linked lists; this in turn simplifies the process of returning to a previous state by moving back through the lists.

The principle behind this model is simplicity: an intuitive and straightforward interface is necessary to free mental resources for difficult tasks, such as non-visual graph exploration. This model offers the following benefits:

- The user is always browsing a list; there are no confusing mode changes.
- The user effectively generates a breadcrumb trail as he or she browses — it is possible to return to a previous state simply by (repeatedly) moving back to the last list.
- Movement and exploration can be rapid — few commands are required to browse a list. For example, consider a list of nodes returned from a search; to scan these, the user simply needs to move down the list. A property of each node can be queried (if the node is of interest, it can be user-annotated or explored further), then one command issued to return to the search results list, another to move to the next node, and so on.
7.3.1 A Running Example

Consider again the family tree shown in figure 4.2. On starting the software, the focus is on the first node on the list of all nodes, i.e., Henry Brown (this node is also the home node). Moving up and down the list allows all nodes to be accessed. The list is ordered by number. One can stop on Thelma Stevens and ask for the properties — this generates a new list as shown in table 7.3. This is presented to the user with an introduction (“3 properties”) and the first list item (“born 17/8/1921”). Moving up and down this list allows the three properties to be browsed in simple chunks.

<table>
<thead>
<tr>
<th>All Nodes</th>
<th>Properties of Thelma Stevens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Henry Brown</td>
<td></td>
</tr>
<tr>
<td><strong>Thelma Stevens</strong></td>
<td>——</td>
</tr>
<tr>
<td>Roger Horlock</td>
<td>born: 17/8/1921</td>
</tr>
<tr>
<td>Susan Brown</td>
<td>generation: 1</td>
</tr>
<tr>
<td>Jeffrey Brown</td>
<td>sex: female</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.3: The user-interface lists. The user has been browsing the first list (of all nodes) and has asked for the properties of the second person on the list. This person (Thelma Brown, highlighted in bold) is the focus node.

While browsing this list the focus remains on Thelma Stevens, so the command for connections generates a list with her connections. Since the properties list is simple (it is a list of statements, not nodes) the connections list replaces it. The result is shown in table 7.4.

<table>
<thead>
<tr>
<th>All Nodes</th>
<th>Connections of Thelma Stevens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Henry Brown</td>
<td></td>
</tr>
<tr>
<td><strong>Thelma Stevens</strong></td>
<td>——</td>
</tr>
<tr>
<td>Roger Horlock</td>
<td>son: Jeffrey Brown</td>
</tr>
<tr>
<td>Susan Brown</td>
<td>daughter: Susan Brown</td>
</tr>
<tr>
<td>Jeffrey Brown</td>
<td>husband: Henry Brown</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.4: The user has asked for the connections of the second person on the list. This has replaced the list of properties. The first item on this list is read out, and becomes the focus (bold).

The user is now browsing the connections of Thelma Stevens. The graph can
be traversed in this manner; for example to find if Thelma has any grandchildren through her daughter, the user can browse her connections by issuing the connections command while on Susan Brown. The resulting lists are shown in table 7.5. Moving up and down this list demonstrates to the user that Susan Brown has no children, so the user can investigate if Thelma has any grandchildren through her son. To do this the user must return to the last list (connections of Thelma Stevens), move up to the first item (son: Jeffrey Brown) and ask for his connections. The results of this command are shown in table 7.6.

The input and output, as seen by the user for this set of commands is given below. The format is as follows:

**command** *A description of the command.*

'Speech output' *Comments.*

Starting on the first list, the initial output is an introduction: the name of the file and the name of the start node.

'New family tree: Browns'

'Start node: Henry Brown (home)' The first node is also the home node.

**next in list** *Move down the list.* 'Thelma Stevens.' The second name on the list - the focus is now on this node.

**properties** *List the properties of the current node.* '3 properties: born: 17/8/1921' There are three properties, the first is her date of birth. This corresponds to table 7.3.

**next in list** *Move down the list.* 'generation: 1' The second property is the generation — first generation means the oldest generation in the family tree.

**connections** *Get the connections of the current node.* '4 Connections: son: Jeffrey Brown' The connections of Thelma Brown. There are 4 in total, Jeffrey Brown is the first - the focus is now on this node. This corresponds to table 7.4.

**next in list** *Move down the list.* 'daughter: Susan Brown' The second connection is Susan Brown.
connections Get the connections of the current node. ‘3 Connections: husband: Roger Horlock’ The connections of Susan Brown — there are three in total, the first is her husband. This corresponds to table 7.5.

next in list Move down the list. ‘father: Henry Brown’ Her father.

next in list Move down the list. ‘mother: Thelma Stevens’ Her mother. A non-speech sound indicates that this is the last in the list: the user concludes that Susan has no children.

back to last list Move back to the last list. ‘back to Connections of Thelma Brown: Susan Brown’ The focus is back on Susan Brown, and the user is browsing the list of Thelma Brown’s connections.

previous in list Move up the list. ‘son: Jeffrey Brown’ The previous one of Thelma’s connections is her son.

connections Get the connections of the current node. ‘5 Connections: son: Andrew Brown’ There are 5 connections of Jeffrey Brown, the first is his son — Andrew Brown. This corresponds to table 7.6.

relationship Find how the current node is related to the home node. ‘grandson’. Andrew is Henry’s grandson.

Thus the user can traverse the tree, finding how nodes are connected (who is related to whom) and finding their properties (date of birth, etc.).

### 7.3.2 Searching

The use of annotation for making features of the graph explicit has already been discussed. Larkin and Simon [51] also identified a second key benefit of diagrammatic presentation: search. They noted that the ability to use the spatial layout to group related items had benefits for problem solving with diagrams. Although in graphs the spatial layout is not meaningful, it is still likely that related nodes will be close, since more closely related nodes are likely to have fewer arcs between them (i.e., the shortest path between them will traverse few arcs). It is therefore important to enable searches in the non-visual interface.

Several types of search have been implemented for this evaluation. Users may search the graph for graph entities, such as nodes (by name) or for the following
7.3. THE USER INTERFACE

annotations: numbers, labels, notes, relationships, properties, or all of these. The user inputs a search string (which may include wildcards) and matches are presented in a new list. The user may then explore the matches, returning easily to the list should a match prove uninteresting.

As an example of how this is important, consider the family tree — these are typically arranged with each generation horizontally aligned, and direct ancestors approximately vertically aligned. It is therefore often possible to see at a glance how closely related one person is to another, and to move to their relations, for example, aunts and uncles. When browsing non-visually, direct relationships are simple to discover as the user can just ask for the connections, but more distant ones can be difficult. As noted previously, however, nodes have been annotated with their relationship to the home node; if these annotations can be searched it becomes simple to move to any related node.

Searching also allows nodes with particular properties to be found, and moved to, very easily. While a visual graph may identify different classes of node (e.g., different types of gate in a logic circuit) with different symbols such that the diagram may be scanned and these nodes identified, non-visually scanning is a slow and complex process. To a certain extent, the node access through summaries can perform this role, but the needs of users will vary and not always coincide with the categories used in the summary. In this case, scanning can only be done by performing an explicit search.

The ability to search annotations increases their functionality. It allows user-generated annotations, such as notes and labels, to be found easily and therefore act as bookmarks as well as landmarks.

7.3.3 Issuing Commands

So far, this chapter has described the annotations that have been implemented for the evaluation, and the principles behind how the graph and its annotations are explored — the user-interface. The means by which the commands are issued has not been introduced. Although the ideal would be an application that behaved like any other software, and could be accessed with the users choice of screen-reader, ease of implementation meant that, for evaluation purposes, a self-voiced application was developed. The behaviour of the interface was, however, designed to mimic as closely as possible the behaviour of a Windows application being
accessed through Jaws\(^4\).

Overarching all of this, however, was the intention to follow best-practice for speech-based software, as described by Pitt and Edwards [74].

All commands can be accessed through a set of menus. These are outlined in table 7.7. They are accessed in the same way as menus in Windows applications are accessed by the popular screen-reader Jaws. The commands are grouped into five categories with a menu for each, plus a menu for opening new graphs and closing the application. The categories are:

**Annotation** Commands for users to create or read their own annotations.

**Speak** Commands relating to speech control. As noted in [74], it is critical to “ensure that information flow is, as far as possible, controlled by the user”. These commands include muting, repeating the last phrase, and increasing or decreasing the speed at which the synthetic speech is presented. This menu also contains commands for moving up and down lists, and for a reminder of the current list position. Finally one command gives access to more information, if available. When there is information supplementing what has been requested, or information about an arc (e.g., the date of a marriage) its presence is indicated by a non-speech sound, and it can be listening to by issuing the ‘more info’ command.

**Navigation** Commands for moving around the graph, such as creating a list of nodes connected to the current one. This menu also contains commands for moving back to the previous list or to the home node.

**Information** This category is for commands that give information about the node, including annotations such as number and relationship to home, and the summaries of the entire graph.

**Search** Commands for searching the graph.

Note that the Navigation → List sub-menu has its single item supplemented when reading family trees, to allow parents and children to be listed separately — the apparent anomaly of a sub-menu with a single item was retained for consistency.

\(^4\)Jaws is a popular screen reader for the Windows operating system. Version 7.00. www.freedomscientific.com
Menu items have shortcuts, like Jaws (e.g., Alt-I gives the identity (number) of the current node), and the most common commands can be issued with a single keystroke.

### 7.3.4 Non-speech sounds

In addition to their use in audio glances of the graph, non-speech sounds have been used to present information that may be encountered regularly. This is typically a simple notification rather than more sophisticated information; presenting with a non-speech sound is quicker, allowing a certain degree of differentiation between requested information and information that is ‘incidental’.

**Start or end of list** A buzz is used to notify the user when the end of a list is reached (or the start if moving up a list). The same buzz is used to tell the user that he or she has reached the end of a menu list (although in this case it is possible to cycle through to the start).

**Visited Nodes** A beep indicates if a node has been visited previously. It is presented with the node name each time it is encountered.

**More Information** If the information is too long to be presented in a single chunk it is broken into smaller chunks. The first of these is presented to the user and is followed by a beep indicating that more information is available.

### 7.4 Architecture

The implementation of this system was designed with an architecture as depicted in Figure 7.3. The graph, and its annotations, is represented using RDF. The RDF representation is generated with components that read the graph (in its original format) and add some forms of automatic annotations, such as feature recognition. The RDF is read, and modified, by the core of the user-interface — the list controller. This stores the list of lists, thereby keeping track of where the user is currently exploring, and has components that translate input into output. Input and output are effectively filtered through domain-specific components to allow extra commands and different vocabularies for different domains. Output is through speech, non-speech sound, and visual components that allow a layer of abstraction from the particular hardware.
7.5 Summary

Testing the hypothesis that annotating graphs helps visually impaired users requires the implementation of a tool; this chapter has discussed some of the issues around this topic. How the graph and its annotations are represented is crucial, and the requirements for doing this have been laid out. In summary these are that the semantics of the graph must be fully described, and any system must support a range of types of dynamic annotation, with notes that can relate to chunks of the graph, not just individual nodes.

The Resource Description Framework is an established technology, designed for annotation within the semantic web. This, and the fact that it is itself graph-based, suggests that it may fulfil the requirements. This hypothesis is proven: upon examination the framework is seen to be flexible and powerful, and RDF can be used to represent both the graph and its annotations in a seamless description.

Many of the different forms of annotation discussed in the preceding three
chapters have been implemented for evaluation. These include the home node, audio glances, descriptive summaries and numbering. An interface has been developed that allows the user to move around the graph from node to node. This interface is based on a simple list-based model. Some annotations are presented automatically, while others are presented only upon request. A running example of the application in use has been presented. This tool is suitable to allow us test the hypothesis of this thesis: that annotation helps browsing.
Table 7.5: The graph is being traversed: the user has asked for the connections of Thelma Brown’s daughter.

<table>
<thead>
<tr>
<th>All Nodes</th>
<th>Connections of Thelma Stevens</th>
<th>Connections of Susan Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td>Henry Brown</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thelma Stevens</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roger Horlock</td>
<td>→ son: Jeffrey Brown</td>
<td>→ husband: Roger Horlock</td>
</tr>
<tr>
<td>Susan Brown</td>
<td>daughter: Susan Brown</td>
<td>father: Henry Brown</td>
</tr>
<tr>
<td>Jeffrey Brown</td>
<td>husband: Henry Brown</td>
<td>mother: Thelma Stevens</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
7.5. SUMMARY

Table 7.6: The user has retraced his steps to Thelma Brown, and asked for the connections of her son.

<table>
<thead>
<tr>
<th>All Nodes</th>
<th>Connections of Thelma Stevens</th>
<th>Connections of Jeffrey Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td>Henry Brown</td>
<td>→ son: Jeffrey Brown</td>
<td>→ son: <strong>Andrew Brown</strong></td>
</tr>
<tr>
<td>Thelma Stevens</td>
<td>daughter: Susan Brown</td>
<td>daughter: Catherine Brown</td>
</tr>
<tr>
<td>Roger Horlock</td>
<td>husband: Henry Brown</td>
<td>wife: Rosemary Irwin</td>
</tr>
<tr>
<td>Susan Brown</td>
<td></td>
<td>father: Henry Brown</td>
</tr>
<tr>
<td>Jeffrey Brown</td>
<td></td>
<td>mother: Thelma Stevens</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 7.7: The menu structure of the software. Some menus contain sub-menus, indicated by the → symbol.
Chapter 8

Evaluation

This chapter describes how the tool introduced in the previous chapter was used to evaluate the benefits of annotation. For this, cooperative evaluation, software logging, and questionnaires were used, while participants performed tasks in two domains. This evaluation was designed to test those types of annotation not previously tested, and to examine how the different forms could be used as a coherent whole.

8.1 Evaluation questions

The purpose of the evaluation was to test the hypothesis of this thesis:

Annotations can be designed to replace certain of the benefits imparted by visual presentation of graph-based diagrams, including making implicit information explicit, grouping related items, interactivity and acting as an external memory, and to reduce disorientation while moving around the graph. A graph which is annotated in such a way requires less mental effort for a visually-impaired user to explore than one which is not. Tasks can be achieved more effectively, efficiently and with more satisfaction through use of annotation to replace features of a visual presentation.

The preceding chapters have explained how various forms of annotation can be added to the graph in a way that might replace those benefits; this chapter describes how it was determined whether these annotations actually help users. To do this, it is necessary to compare how people explore annotated graphs with
how they explore those same graphs without annotation. To allow quantitative, as well as qualitative comparison of the two modes of exploration, we must be able to measure its effectiveness — this can be done by giving participants tasks that require exploration of the graphs to complete.

These tasks must, however, be designed so that the participants are likely to encounter the types of difficulty that are typically associated with non-visual exploration. This can be achieved if tasks are designed with reference to the taxonomy of annotation (section 4.3). Although difficult to identify the benefits of individual types of annotation, it should at least be possible to note whether different classes of note are useful.

The four categories of difficulty used in the taxonomy of annotation are: summary, orientation, relating, and user-tasks. Based on our main hypothesis, what are the expectations relating to each of these annotation types?

**Summary** Participants are better able to understand the gist of an annotated graph than a graph that is not annotated.

**Orientation** Participants are less likely to become disoriented moving around annotated graphs than around non-annotated graphs.

**Relating** Participants will be able to understand better how distant nodes are related when a graph is annotated.

**User task** Tasks typical of the domain will be easier to perform if the user can make their own annotations on the graph.

Although the particular aim of the evaluation was to test the main hypothesis, if the expectations above can be tested individually, the result should be a greater understanding of the benefits of annotation. In summary, questions need to be devised so that the difficulties represented by each class are likely to be encountered during the evaluation.

### 8.2 Method

This section describes how an experiment was designed so that the main hypothesis could be tested, while also giving as much information as possible individually on the four expectations. Appendix F gives further details about the experiment, including participants instructions.
8.2. METHOD

8.2.1 Experimental Design

Due to the relatively small number of participants available, and the limited time for each evaluation, the following design was used. In this, each participant answered questions on two graphs of one domain with annotation and two graphs of the other domain without annotation. In order to remove learning effects, etc., the order in which the domains were presented varied, as did whether annotation was used first or second. This was intended to minimise any effects of participants trying to please the experimenter. The design is best explained in table 8.1. In each domain the two graphs were always presented in the same order: a relatively simple graph, followed by a more complex example. There was no intention to compare the two graphs within a domain: all four questions were considered as a single task.

<table>
<thead>
<tr>
<th>Participant</th>
<th>First domain</th>
<th>Second domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Family trees, no annotation</td>
<td>Logic circuits, annotated</td>
</tr>
<tr>
<td>2</td>
<td>Logic circuits, no annotation</td>
<td>Family trees, annotated</td>
</tr>
<tr>
<td>3</td>
<td>Family trees, annotated</td>
<td>Logic circuits, no annotation</td>
</tr>
<tr>
<td>4</td>
<td>Logic circuits, annotated</td>
<td>Family trees, no annotation</td>
</tr>
<tr>
<td>5</td>
<td>As participant 1</td>
<td></td>
</tr>
</tbody>
</table>

Table 8.1: Experimental design.

8.2.2 Tasks

Having laid out the main aim of the evaluation, the expectations may be used to inform the particular types of question that need to be asked. These are:

**Summary** Questions testing this need to determine if users can pick up the gist of a graph, or some information about the graph as a whole, rather than just about individual nodes. Examples could include questions about the size of the graph, or the types of node it contains.

**Orientation** This class of annotation includes notes that help prevent readers from becoming disorientated. Testing the benefits of these notes can be done by requiring participants to move around the graph — mistakes would indicate disorientation.

**Relating** Questions can test this difficulty explicitly, asking how nodes are related. This is perhaps not as meaningful in logic circuits, but questions that
8.2. METHOD

require users to understand which gates affect which other gates should test this expectation.

**User task** This cannot really be tested by designing particular questions, but by ensuring that the questions are typical of the domain, and observing whether participants use the annotation not just to help them read the graph, but also to perform the task.

Clearly real-world tasks will typically involve more than one of the difficulties outlined above, and it was not practical to design realistic questions to test each of the four expectations independently. These expectations informed the selection of tasks such that all types of difficulty should be encountered, and that at least qualitative evidence for or against each expectation could be gathered.

For each domain, two questions were asked on each of two graphs. The same graphs were used for all participants, although the order of domains, and the use of annotation varied according to the experimental design (above). For each graph, the questions followed a similar pattern: the first asked a general question about the properties of the graph, e.g., about the number of nodes, while the second asked a more specific question appropriate for the domain. This structure was designed to balance sufficient time to get used to the system, while keeping the overall evaluation to a reasonable time (between sixty and ninety minutes, typically).

For family trees, the basic graph was a simplified version of that shown in figure 4.2 (the nodes representing Tessa Williamson and Ann Pearson were omitted). The questions for this graph were:

1. How many people are there in this tree who do not have the surname ‘Brown’?

2. Who is the youngest person in this family tree?\(^1\)

The other graph represented the relationships between some of the Roman Emperors in the Julio-Claudian dynasty. This graph was both large (29 people) and complex: there were eight generations, several cases of multiple marriages and adoptions, and marriage within the family. It was further complicated by

\(^1\)This question was originally intended to be about the heir of the family (thus requiring movement around the graph and understanding of relationships), but the first two participants did not understand this term, and the question had to be changed.
repeated use of certain names over the generations. The questions for this graph were:

1. How many generations are represented in this graph?

2. How are the Emperors Claudius and Tiberius related?

For logic circuits, the two graphs were closely related; in fact the simpler was a sub-graph of the more complex, with two nodes removed. This was to test if participants detected any similarity. For these graphs, the questions were also similar: the first asked for some information about the graph as a whole (a kind of summary), for the first, ‘how many And gates are there?’; and the second, ‘How many gates are there in total?’. The second question for each was identical — it asked them to solve the circuit (i.e., find the output(s)) given inputs of 1 and 0. The graphs and questions are given in Appendix F, which also includes the questionnaire, information, and instructions given to participants.

The first question for all of these graphs is asking for information that describes the graph as a whole, that is summary information. If our first expectation is correct, it will be easier for people to gather this summary information from annotated graphs. The second expectation (orientation) should be tested by some of the other questions: it should be possible to observe any difficulties encountered when moving around the non-annotated Julio-Claudian family tree (for either question), and solving logic circuits should also require users to retain their orientation. The third expectation is tested explicitly with the question about Emperors Tiberius and Claudius; it is also possible that participants need this type of information while solving the logic circuits. The final expectation, that annotations can help user tasks, will be tested by several of the questions — the second question on each graph can be considered a typical task for the domain.

8.2.3 Analysis

The following techniques were used to assess the outcomes:

Task Success Did the participant complete the tasks successfully?

NASA TLX Participants were asked to rate the difficulties of the tasks using the NASA Task Load Index [37]. This system was used in the evaluation of
8.2. METHOD

Kekulé, and is described in more detail in section 3.2.1. Rating was done twice — once for each domain, with participants asked to rate all four tasks together.

**Cooperative Evaluation** Participants were encouraged to think aloud while performing the tasks. The experimenter also occasionally asked them to explain their actions. The evaluation and discussion was recorded.

**Software Logging** All inputs and outputs of the software were logged with times. This gave another record of users actions to help in strategy analysis, while timings could indicate how difficult tasks were found (although it is important to recognise that the use of cooperative evaluation diminishes the reliability of timings).

**Questionnaire** A brief questionnaire asked participants what they found difficult, what features of the software they liked and disliked, and what additional features they would have liked available.

It was intended that the above combination would allow quantitative analysis, using task success and TLX scores, while also capturing more qualitative evidence about motives and strategies.

8.2.4 Participants

Due to the relative scarcity of visually impaired people, not to mention the danger of over-using those who are available for evaluations, this experiment was performed using a mixture of sighted and visually impaired participants. All participants, regardless of the state of their vision, performed the evaluation without any visual display. The main intention behind this was to maximise the number of participants, thereby improving the likelihood of capturing any problems or any interesting behaviour. It has been noted already, however (in relation to the ‘move as you hear’ principle behind the user interface), that sighted and visually impaired users might not experience quite the same difficulties. For this reason, the hypothesis could not be tested using sighted participants only; having two groups (if sufficiently large) could allow any apparent differences to be tested for statistical significance. If no differences between the groups were detected, the participants could be considered as a single group which, with its greater size, might make it easier to detect other effects.
A total of 20 participants completed the evaluation; 13 were male and 7 female. Sighted participants comprised 14 post-graduate students and post-doctoral researchers from the School of Computer Science at Manchester University. All were experienced with computers and used them daily; they were also all at least passingly familiar with both domains.

The visually impaired participants included two with no useful vision (p13 and p16) and four with less severe impairment. These participants covered a wider range of ages and occupations, although all were either university graduates or currently studying for a degree. Due to a hardware problem, the evaluation with p13 was conducted with the experimenter performing the role of speech synthesiser — this was not thought to have had any impact.

All evaluations were performed in a laboratory in the University of Manchester, with the exception of participants p13 and p16. In these cases, the experimenter travelled to the participant and undertook the evaluation in a suitable location. Experiments typically lasted between 60 and 90 minutes.

8.3 Results

The following sections present the TLX scores given by the participants, together with selected observations from the cooperative evaluation. Since no significant differences were observed between the visually impaired and the sighted participants, their results are grouped as one. Where any individual differences or comments are presented, however, it is noted if they were visually impaired.

8.3.1 Success Rates

The abilities of participants to correctly answer the questions may indicate if annotation was providing any benefit. For logic circuits, 4 of 40 questions were answered incorrectly (or the participant was unable to give any answer) on annotated graphs, while for plain graphs the figure was 12 out of 40. For family trees, all questions were answered correctly when annotated circuits were used, while for non-annotated circuits 15 of 40 questions were answered either incorrectly or not at all. Full results are given in table 8.2. These indicate that while the simple family tree was simple under both conditions, the complex example was more readily solved in the annotated condition. Similarly, with the logic circuits,
overview questions could be answered easily in either condition, while solving the
circuit was more likely to be achieved if it was annotated.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Annotated</th>
<th>Plain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>Logic circuits</td>
<td>Family trees</td>
</tr>
<tr>
<td>Question</td>
<td>1 2 3 4</td>
<td>1 2 3 4</td>
</tr>
<tr>
<td>1</td>
<td>y y y y</td>
<td>y n n n</td>
</tr>
<tr>
<td>2</td>
<td>y y y y</td>
<td>y n n n</td>
</tr>
<tr>
<td>3</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>4</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>5</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>6</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>7</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>8</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>9</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>10</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>11</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>12†</td>
<td>y n y n</td>
<td>y y n n</td>
</tr>
<tr>
<td>13††</td>
<td>y y y y</td>
<td>y y n n</td>
</tr>
<tr>
<td>14†</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>15†</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>16††</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>17†</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>18</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>19</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>20</td>
<td>y y y y</td>
<td>y y y y</td>
</tr>
<tr>
<td>total</td>
<td>10 9 10 9</td>
<td>10 10 10 10</td>
</tr>
</tbody>
</table>

Table 8.2: Success rates for the different questions under the two conditions. The
questions are given by their numbers from section F.4. Table values give ‘y’ for
a correct answer, ‘n’ for an incorrect answer, or if no answer was given (i.e., the
participant gave up) ; 10 participants attempted each question in each condition.
Participants 12–17 were visually impaired, as indicated by a † symbol, 13 and 16
were totally blind (††).

8.3.2 TLX scores

The task load indices can be used to compare the relative difficulty of exploring
annotated graphs with non-annotated graphs. To do this the absolute scores are
ignored; instead the difference is calculated. This method should average out
any differences between the domains. Table 8.3 gives the differences for the each
index plus the total demand (the sum of all indices); the raw values are given
in Appendix F (table F.1). The mean and standard deviations are given, as are significance probabilities. These probabilities give the chance of these values being selected from a population where the mean difference is zero; they are estimated using the one-sample Student’s t-test.

<table>
<thead>
<tr>
<th>Participant</th>
<th>MD</th>
<th>E</th>
<th>TD</th>
<th>P</th>
<th>F</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>17</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>8</td>
<td>2</td>
<td>11</td>
<td>-12</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>5</td>
<td>-10</td>
<td>10</td>
<td>-5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>6</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>-1</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>-2</td>
<td>10</td>
<td>23</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>10</td>
<td>-1</td>
<td>10</td>
<td>1</td>
<td>31</td>
</tr>
<tr>
<td>11</td>
<td>8</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>12 †</td>
<td>-5</td>
<td>-5</td>
<td>-2</td>
<td>-8</td>
<td>-8</td>
<td>-28</td>
</tr>
<tr>
<td>13 ††</td>
<td>7</td>
<td>7</td>
<td>4</td>
<td>8</td>
<td>4</td>
<td>30</td>
</tr>
<tr>
<td>14 †</td>
<td>10</td>
<td>15</td>
<td>1</td>
<td>16</td>
<td>19</td>
<td>61</td>
</tr>
<tr>
<td>15 †</td>
<td>11</td>
<td>10</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>40</td>
</tr>
<tr>
<td>16 ††</td>
<td>-4</td>
<td>-2</td>
<td>-2</td>
<td>-1</td>
<td>-3</td>
<td>-12</td>
</tr>
<tr>
<td>17 †</td>
<td>7</td>
<td>7</td>
<td>2</td>
<td>11</td>
<td>10</td>
<td>37</td>
</tr>
<tr>
<td>18</td>
<td>10</td>
<td>7</td>
<td>5</td>
<td>10</td>
<td>5</td>
<td>37</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

Mean       | 4.65 | 5.05 | 1.50 | 4.60 | 2.45 | 18.25 |
SD         | 4.58 | 4.64 | 3.86 | 5.92 | 6.80 | 19.26 |

p (T-test) | 0.00031 | 0.00016 | 0.099 | 0.0028 | 0.12 | 0.00056 |

Table 8.3: TLX score differences. Visually impaired participants are marked, as before with a † symbol. The table value gives the difference between scores for the annotated and plain graphs. MD is mental demand, E effort, TD temporal demand, P performance, F frustration. A positive value indicates the annotated tasks were scored lower (i.e., as less demanding). Probability values are given for the likelihood of these differences being due to chance; these are calculated using the one-sample t-test.

The statistics indicate that, on the whole, participants found the tasks on annotated graphs easier than those on plain graphs. From a maximum total score of 100, the mean difference was 18 points, with a mean sum score for annotated graphs of 42 compared to 60 for plain graphs. Note that, as different people interpreted the limits of the scales differently, there was a wide range of
scores given (standard deviations of 18 and 14 respectively); these means do, however, put the 18 point difference into perspective: the difference was not just statistically significant (to greater than 99%), it was also substantial.

Table 8.3 also shows that participants found that the annotated tasks were significantly less mentally demanding and required less effort, and they felt they performed better. The difference in frustration was not significant; neither was the difference in perceived temporal demand, although this is not surprising as no pressure was put on participants to perform quickly (except, perhaps, their own wish to do the evaluation quickly).

Two participants, p12 and p16, stand out as having consistently rated exploration of annotated graphs as being harder than the non-annotated graphs. In the case of p16, this was despite performing considerably better with the annotated graphs: all questions on the annotated logic circuits were answered correctly, compared to only half the questions on family trees. When questioned about this, this participant explained that he ‘just felt more comfortable with family trees’. Participant 12, however, clearly had more difficulty with the (annotated) logic circuits. Note again that this participant still managed to answer two questions correctly on each graph. Although this participant claimed to understand the functions of logic gates, and the example circuit, it was clear during the experiment that the level of understanding was very low. Thus, both of these outliers can be, at least partly, attributable to their levels of domain experience. It is not believed that it is meaningful that these outliers were both visually impaired.

It is also possible to make comparisons purely within a domain, although in this case it is only possible to compare absolute TLX scores, making it more likely that variations in individual interpretations of the scale mask differences between annotated and non-annotated. For logic circuits, the differences for all measures are small (approximately 3 or less) and for all but temporal demand are lower for the annotated graphs, but none of these differences are statistically significant. For family trees the differences are much starker — annotated graphs were rated as 8.9 points easier for mental demand, and 9.4 points easier for effort. The small sample size (10 in each group) makes statistical comparisons difficult, but these differences are significant to 95% (Wilcoxon Two Sample Test and Student’s t-test) for all but frustration.

As with the evaluation of Kekulé, it was notable that these were difficult tasks: median values for mental demand were 15 and 10.5 for the plain and annotated
graphs respectively.

The results also suggest a mild learning effect, which should be cancelled out by the design of the experiment. If the TLX scores for the first domain encountered by the participants are compared to those for the second (irrespective of which is annotated), the scores are consistently slightly lower for the second domain. This effect is only significant for performance, where the mean difference is 3.5 (11.8 vs. 8.3; t-test $p \leq 0.034$). Similarly, the mean score for performance in the 10 cases where the annotated graph was explored second was 5.1, compared to 11.4 for those where the annotated graph was explored first (t-test $p \leq 0.020$). Although this does not affect the overall conclusion that TLX scores are lower for annotated graphs, it does suggest that practice improves performance, or, at least, perceived performance.

### 8.3.3 Questionnaires

Probably the feature most commented upon as useful was the search facility. This allowed people to gather summary information, such as the number of nodes of a certain type, as well as giving a route for jumping to a specific node. For example, one participant used the search to return to the input nodes when exploring non-annotated logic circuits. It was also popular in family trees, where people used search to find the two emperors so that they could determine their relationship — having to traverse the graph manually to find these two would have made the task substantially harder.

Labels were also popular, cited by half of those who explored annotated logic circuits as a useful feature. P9 said that labels made it ‘easier on the memory’, reflecting the way that the strategy of labelling gate values eliminated the need to visualise the whole circuit. The same participant suggested that such a feature would also be useful for family trees. Several people also either liked the home node facility or requested it (discussed below). In a related area, two people suggested (after exploring non-annotated graphs) that some means of determining their location in terms of paths (e.g., distance from an input or output) would be useful.

Distinguishing nodes was the most common difficulty reported by those exploring non-annotated logic circuits, being mentioned by five people. It was also suggested by another (p11) that numbering the gates would be helpful.

Finally, the components summary was commented upon by four participants,
although it is not clear whether they liked it for the overview it provided or as a means of accessing certain types of node.

8.3.4 Cooperative Evaluation

This section discusses some of the more qualitative information gathered during the evaluations. It contains observations on the difficulties and strategies used by participants; each domain is treated separately.

8.3.4.1 Logic Circuits

The logic circuit questions were answered well. The summary questions were all answered correctly, in both annotated and non-annotated conditions. These questions were asking about the size of the graph — how many gates (overall, or of a certain type) did the graph contain. Although everyone was able to answer these correctly, the methods differed according to condition. Those exploring the plain graph typically traversed the list of all nodes, counting as they went (13 of 20 questions), or used the search function (7 of 20 questions), while those exploring the annotated graph only used search twice, more often using the graph summary (8 of 20 questions) or counting from the list (10 of 16 questions). One or two of the latter group realised after that they could have used the summary: “Oh, I should have used the summary for that”. Another (p15) started to read down the list before stopping: “I want an overview rather than going up and down the list”.

The questions requiring the participants to calculate the outputs of the circuits appeared to take much more effort to solve. For these questions, there was also a clear difference between those using annotated graphs and those who were not: all but one of the former group (p12, who had minimal experience with logic circuits) calculated the outputs correctly, while only 4 out of 10 of the latter managed; most gave up trying. Most of the group using annotated graphs interacted with the graph by labelling nodes (gates) according to their value. One (p16) used some labels and notes to indicate either the function of a gate (e.g., for connector (4) “brings B”, meaning that that connector simply passed on the value of input B), its output, or the value of one of its inputs. Another two participants (p13, who had no vision and was a graduate electronic engineer, and p18) solved the circuit without using any labels or notes. Those exploring the non-annotated
graphs tried to work from the inputs, or back from the outputs, to calculate the values.

A couple said they were trying to deduce the structure of the circuit before calculating the output. One insight that appeared particularly useful for these participants was that knowing that one input to an And gate was 0 meant that the other input did not need to be known; similarly knowing that one input to an Or is 1 is also sufficient to know the output. For example, p7 concluded:

“So, the And has two inputs — 1 and 1, so it’s 1. So the other Or has at least one 1 as an input, so the output will be 1.”

It appeared to the experimenter that only p13, the graduate electronic engineer, had formed a clear picture of the circuit in his mind; indeed this person recognised the circuit and understood it in terms of its logic, not just the output. This was apparent in the way he solved the circuit — as he moved through he would comment, e.g.,

“Right, so And 5 is (A and B), And 6 is (A and not B)”

After a little more exploration, he recognised part of the circuit:

“Ah, this is the carry function isn’t it”

A little more exploration and double-checking led to the (correct) conclusion:

“So you’ve actually got your half-adder as before: ((A and not B) or (B and not A)) goes to output C, which gives you an exclusive Or, a 1, and output D is a straight (A and B).”

It was perhaps notable that very few participants described the nodes in terms of their function (e.g., A and not B).

Several of the other participants (less experienced in this domain) trying to calculate the outputs of annotated logic circuits initially tried to solve it in their mind, but realised fairly soon that this was difficult and started a labelling strategy. This sometimes needed some prompting. For example, p5 started moving from an input trying to work out the structure:

p5: “Okay, it’s getting a bit bigger now”
8.3. RESULTS

p5: (using his fingers on the table) “And, inverter, inverter, or”

experimenter: “So, what are you trying to do? Are you trying to just picture the whole graph then solve it?”

p5: “Yeah, and if it’s too big I’ll have to think of something else to do”

experimenter: “It’s getting quite big already”

p5: “It’s getting big, yeah”

experimenter: “What else do you think you could do?”

p5: “I could work out the result at each point then forget about further above.” P5 continues imagining the circuit. . .

p5: “Argh, it’s too big for me to handle”

experimenter: “I think trying to remember it all might be quite difficult”

p5: “It’s finding new techniques, I suppose”

experimenter: “Yes, how would you do it the other way?”

p5: “I work out everything that goes into that And gate and what its result is, and remember that.”

experimenter: “Do you need to remember it?”

p5: “I need to remember it — I need to use it. Or I can store it as a note!”

experimenter: “Or as a label, yes.”

This participant used labels to store the values at each node and successfully solved the circuit. There was some confusion, however, as the labelling technique needed careful application: it appeared easy to get confused whether the labels referred to input or output values. Some participants commented that it would have been easier to label the wires.
8.3. RESULTS

8.3.4.2 Family Trees

The questions on the simpler family tree were generally answered correctly using both annotated and plain graphs. The more complex graph (representing the Julio-Claudian Roman emperors) was a different matter: all participants exploring the annotated graph were able to answer both questions correctly but of those using the plain graph one of the questions was only answered correctly by one person.

The summary questions were answered in markedly different ways. With the annotated graphs, all but one question was answered using the components summary (the other used search). With the non-annotated graphs, the number of non-Browns was generally determined again by browsing the list of all nodes and counting (the remaining two got the total number of people from the position in list and subtracted the number of Browns found with a search); all participants tried to determine the number of generations represented in the Julio-Claudian tree by traversing the graph. All were unsuccessful — most counted four or five generations, but were not confident that they had seen all the nodes (there were eight generations), while one managed to traverse from generation one to eight but via an adoption (between Julius Caesar and his great-nephew) so was unsure of the number of generations.

To solve the question about the youngest Brown a range of strategies was used. Most (all but one participant, who was exploring the non-annotated version) managed to find the correct answer, although many of these were aware that they could not be sure that there was not another, younger person. One participant (with a plain graph) used a technique not envisaged by the designer of the experiment — he simply moved down the list of all nodes asking for the properties (hence date of birth) for each participant. He actually stopped when he found the most likely answer, but could have simply continued to be certain. Most other participants who were exploring the plain graph moved around the tree in a roughly depth-first pattern trying to find likely candidates.

Some different techniques were used by those exploring the annotated graph. The graph was annotated with a summary giving the number of generations (4) and each node was annotated with its generation; the people in the youngest generation can be found by search or through the components summary. Again this only gave a likely answer — it was always possible that a member of generation 3 was younger. Not all people actually used this: many traversed the graph
8.3. RESULTS

anyway; one did this (getting the generation number as he traversed) then used the audio glance to confirm that there was only one person in generation 4. Another effective strategy employed by one participant (p14) used the components summary to access the people in generations 3 and 4, checking their birth dates to ensure that the one member of generation 4 was indeed the youngest.

All participants attempted to determine the relationship between the emperors Claudius and Tiberius by searching for one of them (either by name, or searching properties for ‘Emperor’), although p19 initially started browsing the tree. Those exploring the annotated graph all set one emperor as their home node, then moved (by a new search or by returning to their list of search matches) to the other emperor. They then read the relationship annotation to give them the answer explicitly; all were successful. Those exploring the plain graph tried to traverse the graph from one node to find the other. Only two people were successful (p8, p20), and these both read the properties of each node to discover the dates on which they became emperors and used this information to direct the search. Another person (p16) used dates to inform his exploration strategy in exactly the same way, but lost orientation while moving from one emperor to the other, thus deducing the wrong relationship (grandfather). One other participant (p1) also managed to move from one emperor to the other but failed to remember the route correctly and therefore deduced the wrong relationship (great-grandson rather than nephew).

8.3.5 Software Logging

The software logs did not reveal a great deal that was not either already observed by the experimenter, commented upon by the participants, or revealed by the TLX scores. One interesting observation, however, was the times taken to solve the logic circuits\(^2\). The three participants who solved these without annotation took a mean time of slightly under 10 minutes to solve the first graph, and slightly under 8 minutes to solve the second. Those solving with annotation took much longer, typically around 15 minutes for the first graph, but slightly quicker on the second (around 12 minutes). Thus, although the success rate achieved using annotated graphs was higher, those who were able to answer without annotation

\(^2\)Although the think-aloud protocol used in this experiment makes the use of timings difficult, the difficulty of the tasks meant that conversation was very limited. It is therefore believed that these comparisons are meaningful.
were quicker.

It is also possible to use the logs to contrast the time taken to solve the Tiberius-Claudius relationship with and without annotation. The two people to deduce this correctly without annotation both took just over 8 minutes while the mean time for the 10 people using annotation was 3:21 (ranging from 1:33 to 6:08)

8.3.6 Discussion

What do these results tell us? Were the problems expected without a visual representation observed, did the annotations perform as expected, and do the results support the hypothesis? These questions can be answered firstly with respect to the four expectations outlined above before finally drawing some conclusions about the main hypothesis.

Overall, as tables 8.2 and 8.3 show, the tasks were more likely to be answered correctly and were considered easier when performed on annotated graphs; they were also completed with a higher success rate. The two outliers are believed to have occurred as a result of domain experience; this was not adequately balanced in the experimental design. Most of the problems participants were expected to encounter with plain graphs were observed. In particular, problems with node differentiation led to disorientation, and memory demands were clearly high. Both disorientation and memory demands, together with the potentially large number of possible paths, made relationship deduction hard. When graphs were annotated these problems were generally eased: numbering nodes made them easier to distinguish, and reduced disorientation; allowing users to add labels provided an external memory for certain tasks; and allowing relationships to be made explicit made those tasks simple. These findings are discussed in more detail below.

8.3.6.1 Summary

On smaller graphs, a simple list of all nodes acted as quite an efficient basic summary and participants were able to make inferences about the size and scope of the graphs. What was less easy was making inferences about the shape of the graph, for example the depth (number of generations) of a family tree. This was particularly difficult for larger graphs as it required substantial traversal of the graph. Annotation eased some of these aspects considerably — those exploring
the annotated Julio-Claudian family tree were quickly able to understand the approximate size and shape of the tree, while those exploring the plain version traversed the graph a little before realising it was very complex, but with little real understanding of its depth.

Summary notes were used when participants wanted particular summary information, but were not obviously used as a first stage in a reading strategy. This was the kind of use that was anticipated, so why was it not observed? With visual graphs it is almost impossible not to get some kind of summary before reading, but with non-visual graphs a summary must be explicitly requested. When people had listened to the summary, they did occasionally repeat it to themselves later, suggesting that it was useful information. For example one participant, on reaching an And gate in a logic circuit, said to himself:

“And gate. How many were there? Three, okay.”

There was also a need for participants browsing plain graphs to get summary information. For example, after a few minutes traversing the graph, p10 repeated the summary information discovered answering the first question (the number of And gates) then returned to the list of all nodes to find out how many nodes there were in total, and how many there were of each type. Similarly, p11 wanted the components summary when he started to browse the logic circuits, but instead scrolled the list, and the first action of p20 when browsing the second (plain) family tree was to find the total number of people (from the position in list command). It would be interesting to see if an early view of a summary became a common strategy for experienced users. The components summary was also commonly used as a quick and easy route to certain nodes.

The audio glance was rarely used, although when it was it gave the desired information quickly. The complexity note was similarly rarely used, and was not really understood when it was. Although familiarity would probably make the numbers more meaningful, whether this annotation would then be helpful cannot be determined.

8.3.6.2 Orientation

Orientation was difficult when graphs were not annotated. For example, although three people were able to move between the emperors Claudius and Tiberius (three arcs) only one of those was able to deduce their relationship. That the
others remembered their paths wrongly, e.g., as moving up the tree three times (i.e., to a great grandfather) rather than up twice and down once (i.e., to an uncle), can be attributed to disorientation. Similarly, when moving around plain logic circuits, people found it difficult to remember where they were. For example, having explored into the circuit from one of the inputs, p6 moved back in the lists to start again, only to discover she had been exploring from Input B, not, as she had thought, Input A:

**System:** “Back to connections of Connector: Input B”

**P6:** (surprised) “Input B?”

**Experimenter:** “Where did you think you were? The other one?”

**P6:** (confused) “I was on Input A!”

Similarly, p7, was exploring back from an output, which was the result of an Or gate. Having explored one of the inputs to this Or (an And gate which he determined output a 0), he then wanted to explore the other input (also an And). He therefore returned to the Or, then realised he could not distinguish the two And gates:

“Oh, no! Now we’ve got another problem: I’ve got to remember which of the two Ands it was.”

This was a common problem: p18 also asked: “How do I know which And it is?”, and p19: “How do I distinguish between the Ands?”. Both commented afterwards that a way of distinguishing nodes with the same name would be helpful.

Giving numbers to the nodes, and making them explicit on nodes with identical names, clearly helped. On the annotated graphs, the need for keeping orientation was reduced by making explicit information that would otherwise have needed to be deduced through exploration. When users did perform tasks that required orientation on annotated graphs, however, fewer difficulties were noted. For example, when solving logic circuits, those using plain graphs often struggled to differentiate gates, while those exploring the annotated graphs appeared to make use of the numbers, e.g., they repeated the numbers when thinking aloud. One participant commented afterwards that the numbers were a useful feature.
8.3. RESULTS

Numbering on family trees was not obviously beneficial. This is perhaps a result of the other annotations almost eliminating the need for general exploration; with the non-annotated graphs, some participants were clearly confused by the multiple instances of Tiberius Claudius Nero.

Returning to the start position was a strategy used by probably all participants, either when confused or when they wanted to explore a new avenue (e.g., having moved some way into the graph from one input of a logic circuit, they wanted to repeat the process from the other input). The home node (through its associated ‘go home’ command) was used for this by some participants, particularly on the logic circuits. For example, p12 became confused while exploring the first logic circuit; she exclaimed: “I’m confused. Just start again…” and returned to the home node. It was also cited as one of the commands found particularly useful by participants in the questionnaire (p15, p6, p7), e.g., p15 said that the home node was useful “for orienting yourself”. Many of those who did not use it achieved a similar result by different means, for example using search or the components summary to find and return to an input of a logic circuit.

People exploring graphs without a node annotated as home often wanted this facility. Answers to the question ‘Is there anything that would have made the tasks easier?’ include:

- p14 — “returning to start”;
- p10 — “moving back to the first or overview easily”;
- p5 — “key back to overall list”.

Those who did not have the facility to ‘go home’ often worked around its absence by multiple left-cursor presses to return to the first list. When exploring the first non-annotated logic circuit, p6 wanted to return to an input and did so by moving back through the lists; while she did so she said “just go home”. This was particularly interesting as this participant was exploring the non-annotated graphs first, and had not been introduced to the home node; other participants looking at these graphs second (e.g., p15) also often wanted to ‘go home’ (trying the command and being disappointed that it was not implemented for that domain). The home node was also used as a crucial component of the relationship annotations (see below).

Visit history information was similarly not used regularly, although it was used as intended. The experimenter actually found the beep sound irritating
once more than a few nodes had been visited and concluded that it might be as useful, but less disturbing if used in its negative, i.e., to highlight nodes that have not been visited. Only one participant commented on this noise, however, saying he hadn’t distinguished it from the end of list noise. Participant 20, when exploring plain logic circuits, commented that he wanted to “mark where I’ve been”.

Labels were rarely used as orientation aids — their main application was to record values in the logic circuit. P16, however, labelled the connectors in the logic circuits ‘brings A’ and ‘brings B’, which saved him from remembering which input was connected to which connector — this could be seen as aiding orientation.

8.3.6.3 Relating

Relating nodes was a more direct problem with the family trees than the logic circuits; indeed presenting this kind of information is the central role of a family tree. Annotating the graph to make relationships explicit had a dramatic, albeit unsurprising, effect on people’s ability to deduce relationships from the non-visual graph. Even for a relatively close relationship (uncle/nephew, it was found to be difficult to move from one node to the other while remaining sufficiently oriented to remember the path.

The relationship annotations were only viewed once in logic circuits: p20 traversed from Input A (home) to the output, then asked for the relationship, and was surprised to hear that the distance was only 4 arcs (the traversal had not taken the most direct route):

“Four’! It sounded like there were about ten!”

Participant 6 (exploring a plain graph) commented that she wanted to know how far it was from a node to an input or output. The labelling approach that nearly all participants ended up using to solve the annotated logic circuits perhaps meant that this type of information was not needed.

8.3.6.4 User Task

Annotation was used extensively, when available, to solve logic circuits. Labelling nodes with their output values allowed the output of the circuit to be calculated
8.3. RESULTS

without an understanding of the structure of the graph. As some participants showed, these tasks were still possible without annotation; indeed when this was done successfully it was generally done more quickly than when done with annotation. It was also noted that the within-domain TLX comparison suggested little difference from annotation. Despite this, one participant (p10, who was not successful) commented during exploration “I'd like to be able to assign results to nodes”. It is also likely that the technique of annotating nodes will scale to circuits of any size and complexity, while such circuits are not likely to be solved mentally. In this case, we can clearly identify that the labels are giving the users an interactive external memory.

The most common task for family trees — identifying relationships — was explicitly annotated once the user nominated their home node. As noted above, this type of task proved difficult on plain graphs.

8.3.6.5 Overall

Having commented on how the problems and annotations fitted with the expectations described above, the main hypothesis can be tested more rigorously. Did the annotations enable the tasks to be performed more effectively, efficiently, and with more satisfaction?

Effectiveness is perhaps the most important desirable: are users more likely to complete tasks with annotated graphs? The evidence of this evaluation is that this is the case. For some types of task, annotation turned the task from very difficult to quite straightforward (e.g., determining relationships in the family tree). For others, such as solving logic circuits, it enabled strategies that allowed problems to be solved using a simple method, that would otherwise demand considerable mental effort. Furthermore, this strategy should scale to problems that would be considered near-impossible otherwise.

It was noted above that those who solved logic circuits without annotation did so more quickly, i.e., more efficiently, than those using annotation. Is this a problem with annotation in general, with the particular annotations used, with the user-interface, or was there another cause? One aspect is likely to be strategy development - all participants exploring annotated graphs started trying to solve the circuit without using the labels, only deciding upon the label strategy after discovering the difficulty. There was also confusion about what was being labelled (input or output values) and although this could perhaps be improved by allowing
different label types, it is indicative of a strategy that is not fully developed. It is speculated that, with a little practice, users would develop this strategy such that it is more effective; this is supported to an extent by the fact all participants (with annotation) were quicker on the second circuit, even though it was more complex. Furthermore, it should also be noted that annotation should never be a handicap as it can be ignored!

On the whole, however, the evidence from the TLX scores is that participants found the tasks less mentally demanding with annotation. This, surely, is evidence that efficiency (performing a task with minimum effort) is improved.

As was noted above, some participants seemed to feel more comfortable with one domain over the other, almost irrespective of whether it was annotated, or even how successful they were answering the questions. The best measures of satisfaction are performance and frustration, since people will be more satisfied if they feel they have done well, and less satisfied if they were continuously feeling frustrated. On these measures, the TLX scores showed no significant difference in frustration, but a mean improvement in performance of over 4 points (on the 20 point scale) — a difference significant to over 98%. It can therefore be concluded that, at the very least, satisfaction is not worsened by the annotation and that it is probably improved.

8.4 Summary and Conclusions

The evaluation has shown that annotating a diagram can allow it to be explored with less mental effort, allowing tasks to be achieved more effectively and more efficiently. There is also some evidence that users find these graphs more satisfying to use than plain, non-annotated, ones. Taken in combination with the evaluation of Kekulé, which demonstrated the effectiveness of chunking to make implicit information explicit, it has shown that many of the other advantages of diagrammatic presentation, can be addressed using annotation. Similarly, consideration of the problems as comparable with spatial exploration has suggested annotations that address the issue of disorientation; these too have been shown to help, with node differentiation an important issue.

An extension of the theme of recognition that has proved very useful, is to take information that is not quite explicit, but the inference of which is facilitated by a diagram, and to make that explicit too. The most successful example is that
8.4. SUMMARY AND CONCLUSIONS

of relationships in family trees — the typical layout of a tree (in generations) makes inferring these relatively simple, and it is an important task, so making them explicit proves beneficial.

Some forms of annotation were not popular, including visit histories, audio glances, and complexities. In addition, summaries were not used as previews except when forced (e.g., by a previous question), in which case they appeared to be of benefit later. This type of summary could perhaps be given to the user while loading the graph.

Unfortunately, the relatively low number of visually impaired participants has made quantitative comparison with sighted participants difficult. Qualitatively, there did not seem to be any differences between the groups, with the two outliers in the visually impaired group attributed to domain experience. It has not been possible to address the issue of the method of moving between nodes. Qualitatively it can be noted that one of the five visually impaired participants did not like the system, wanting instead to be able to survey before moving. It was when considering those lacking any vision, however, that a difference was hypothesised, and although blind participants performed well with this design, no conclusion can be drawn.
Chapter 9

Discussion

This chapter concludes the thesis with a summary of its findings and their significance. Areas of outstanding work are also discussed and potential avenues for future research are introduced.

9.1 Overview of this Thesis

Using and understanding graphs is difficult for visually impaired persons, particularly as most of the benefits of this type of presentation are visual. The overall objective of this thesis was to investigate techniques that would make this important class of diagram more accessible non-visually. This was to be achieved by examining the literature to determine the benefits of presenting information diagrammatically and considering how these may be replicated or replaced in an audio environment. The hypothesis was that graphs could be augmented with annotations that replace certain of these benefits, such that non-visual understanding becomes less difficult.

This hypothesis has been supported by the work presented in the preceding chapters. The main benefits of diagrammatic presentation, and the main problems of non-visual interaction were elucidated so that annotations could be designed that should support users. These were classified over two axes and, where possible, evaluated independently. Finally, an evaluation of the hypothesis as a whole was designed and conducted, using a mixture of sighted and visually impaired participants, that demonstrated that graphs augmented with suitable annotations could reduce the demands of graph exploration.
9.2 Significance of Major Results

By answering the research questions posed in Chapter 1, this thesis has made the following contributions:

Q1. What is difficult when browsing graphs non-visually?

One of the more significant contributions of this thesis is the insight that an effective tool, such as a non-visual graph reader, can only be developed from an understanding of both the processes it is replacing (i.e., how visual graphs are used and how they benefit readers) and the differences in presenting information between the available media.

An examination of the literature, coupled with experiments, has identified the major benefits of diagrammatic presentation, and the major problems of audio presentation when using graph-based diagrams. Much of the previous research in this area has focused on technological solutions rather than the problems [12, 49], with papers describing tools, sometimes not even evaluated. It could be argued that this has resulted in some of the particular problems encountered by visually impaired computer users not being adequately addressed, even by those addressing the problem in a more rigorous manner [8]. In particular, this thesis tackles the problem of making explicit features of graphs that would otherwise be implicit (unless presented visually) — the single-most important benefit of diagrams, according to Larkin and Simon [51].

In addition to these theoretical studies, experiments evaluating a non-visual molecular structure browser, Kekulé, highlighted some other difficulties that may be encountered, for example, confusion caused by poor node differentiation. Taken with the literature review, this has given a reasonably comprehensive understanding of the main types of problems likely to be encountered when trying to understand graphs non-visually.

Q2. What annotations can be added to a graph?

A collection of annotations have been developed, presented and classified; these are intended to augment graphs such that some of the difficulties identified above
are reduced. The design of these annotations has focused on recreating the benefits of diagrams in a non-visual environment and addressing the problems associated with exploration. In particular these included automatic identification and annotation of features of the graph that are implicit but would have become explicit if presented as a diagram, and summaries and overviews of the graph. Graph browsing was likened to spatial exploration (albeit with notable differences) and further annotations were designed to facilitate travel.

A taxonomy of annotations has been developed that classifies them over two categories: provenance and application. Provenance reflects the relative ‘truth’ of an annotation: is it original information, i.e., explicit in the base representation; inferred; provided by the user; or recorded (e.g., the last visit of the user to a node). Classification by application describes the intended benefit of an annotation, be it for summarising the graph, aiding orientation, relating nodes, or helping with user tasks.

**Q3. Can Earcons be used to convey the gist of a graph?**

A key benefit of the diagram is the ability of the reader to scan it all quickly, thereby gaining an impression of the size and complexity of the information and enabling reading strategies to be developed. This thesis has explored and evaluated a novel use for Earcons: presenting an audio summary of a graph so that listeners can gain insight into its size, complexity and structure. Earcons are structured sounds used to present information about “some computer object, operation or interaction” [11], typically representing something like a menu item (e.g., [17]) where the whole sound must be understood to know its meaning. This thesis has investigated the use of structured sounds, like Earcons, to present more of an impression, so that the detail of the sound is not crucial. An evaluation with simple abstract graphs showed that people who had listened to such an audio glance could not only recognise graphs, but also describe the graph structure in a more or less detailed way. Further tests showed that there was potential for using audio glances to highlight portions of a graph.

**Q4. How can nodes be differentiated?**

Another important feature of diagrams that is lost if presented using audio is the ability to differentiate nodes by their spatial position. Attaching numbers
to nodes can act as a surrogate for this 2D-indexing, but experiments showed that poorly considered numbering schemes could cause confusion. This thesis explored different methods for distinguishing nodes, and presented the results of an evaluation of a range of numbering systems. Applying the spatial analogy to graph exploration suggested the importance of landmarks; the use of annotation to facilitate landmark creation has been explored.

**Q5. How can annotated graphs be represented?**

The requirements for representing graphs and annotations have been extracted from an analysis of the form and function of the different types of note. The most important of these are: the semantics of the graph must be fully described; annotations can take many forms; parts of the graph of different size may be annotated (e.g., individual nodes or groups of nodes); and annotations must be dynamic. The requirement for a certain amount of domain-specificity also demands flexibility from the representation. The Resource Description Framework allows for coherent representation of both the graph and its annotations in a manner that supports these requirements, as well as the requirements for user-interaction.

**Q6. Can annotation facilitate non-visual graph browsing?**

The various forms of annotation explored here have been combined in a tool (that represented the graphs and their annotations using RDF) to allow non-visual exploration of logic circuits and family trees. An evaluation of this has shown that augmenting graphs with annotations that replicate or replace the benefits of diagrams can reduce the difficulties encountered by people trying to understand them non-visually.

The main benefits of diagrams as a means of presenting information, and the ways in which annotations can replicate them are as follows:

Diagrams facilitate recognition — features that would otherwise be implicit become explicit when presented diagrammatically. Automatically identifying these features and annotating the graph to highlight their presence can simplify exploration, especially if used to build a hierarchical view of the graph.

Diagrams provide an external memory that readers may access very quickly. The transient nature of sound, however, radically changes this, making graph reading more like spatial exploration; annotations can be designed to ease the
problems associated with exploration (see below). Diagrams may also be interactive, with readers able to modify and add notes of their own. Allowing similar user interaction in the form of notes and labels can enable non-visual graphs to be used in a similar way. For example participants in the evaluation annotated a gate in a logic circuit with its value or function so they could follow the logic from input to output.

Diagrams automatically summarise — due to the way they act as an external memory, and the rapidity of eye-movements, much information about the overall nature of a graph may be discerned from a glance, potentially informing readers of strategies for reading and using the information. Annotating the graph with summaries can allow quick access to similar information. Although the audio glances were shown to be able to convey the gist of simple graphs, summaries were only used in the main evaluation when summary type information was explicitly required, not as a first stage in general exploration.

Diagrams also benefit readers by grouping related items. This benefit can be replicated through both annotation, indicating when nodes are part of a group, and user-interface design, allowing groups to be explored hierarchically and enabling connection-based browsing. In addition, one of the key roles of graphs is to represent relationships. Augmenting a graph with annotations that make near and distant relationships between nodes explicit proved beneficial.

Considering next the ways in which understanding a graph non-visually compares with exploration, the following difficulties may be expected, and countered with annotation:

One of the most significant causes of disorientation is not recognising where one is due to an inability to distinguish locations. Differentiating nodes using numbers was shown to be a valuable means of helping users retain orientation.

Another means of differentiating nodes, particularly during the early phases of exploration, is by identifying if one is visiting a new location or returning to one that has been previously visited. Annotating the graph with non-speech sounds to present this difference, however, was not found useful by participants.

Even with various annotations described above, disorientation is still a distinct possibility. One particular problem for disoriented travellers is returning to a known location in order to re-orient. Having a node designated a ‘home’ node (and allowing this designation to be applied to any one node in the graph) can allow people exploring non-visual graphs a means for quickly jumping back to a
known location.

The concept of a node annotated as a ‘home’ node is also crucial to the idea of making relationships explicit. When a node is designated home, the relationship between it and all other nodes may be calculated, and these nodes annotated with the relationship. A user anywhere in the graph may the be able to rapidly relate her current position to the known location of home. The fundamental importance of relationships in graphs was demonstrated by the value of this type of annotation in the evaluation.

While there were issues with some forms of annotation, on the whole participants in the evaluation were able to perform tasks on annotated graphs more effectively, more efficiently, and probably with more satisfaction, than graphs that weren’t augmented in this way.

9.3 Outstanding Issues

Despite the above contributions, there remain some outstanding issues — areas which have been explored but which have not been fully resolved. These are discussed below.

Due to the relative lack of visually impaired participants, and particularly those who are totally blind, it was not possible to evaluate the hypothesis that these users would be more comfortable with the movement model of this user-interface. It did appear, however, that the model caused fewer problems than in the evaluation of Kekulé, possibly due to general improvements in the user interface, rendering this a less pressing question than before.

The expected benefits of summaries were not observed. Since one or two users made use of information recalled from summaries, this can be at least partly attributed to the lack of this information. Although summaries were available, users generally chose not to incorporate it into a general reading strategy, only asking for them if they had a specific need for summary knowledge. It would be interesting to see if use of summaries increased as users developed more sophisticated strategies, or if the information were forced upon them. The latter approach may sound unappealing, but playing the audio glance while the program was loading (and calculating all the other annotations) could be a beneficial option.

A fundamental concept in using annotation to recreate some of the benefits of visual diagrams, and probably bringing the biggest benefits, is recognition
of implicit features, marking them up and chunking nodes of the graph into a hierarchical structure. In Kekulé, this was observed to be of huge benefit for experienced chemists. Although subgraph isomorphism algorithms and ring and chain identification procedures make identification simple, and annotation makes presentation possible, there remain two difficulties. The first is the domain-dependent nature of the features — for molecules these are functional groups, rings and chains, for family trees family groups may be used, while for logic circuits it is possible to identify groups such as half-adders; for each domain the groupings must be individually determined. The second difficulty is that of where users conceptualise themselves when on a higher-order chunk — some participants in the evaluation of Kekulé commented that although the chunking helped greatly, they still wanted to imagine themselves on a particular node within the chunk, rather than on all at once.

Some of the other forms of annotation were not well used. It would be interesting to determine if the causes were:

1. The annotation was addressing a problem that was not significant.
2. The annotation was not an effective way to tackle the problem.
3. The user interface did not allow the benefits to be realised.
4. The user had not a sufficiently developed strategy for using the annotation.

Of course, all four factors may be influencing the usefulness of the annotation to a greater or lesser extent.

9.4 Future Work

The research leading to this thesis, and the contributions it has made, offer several avenues for future work.

As mentioned in the introduction to this thesis, although this research has focused on purely audio presentation, it is expected that the techniques may benefit other, more sophisticated interfaces. Understanding the benefits of visual graphs when designing such devices will certainly help, but can the concept of augmenting graphs with annotations help people exploring graphs through haptic interfaces? Although haptic interfaces allow spatial arrangement of information,
there are still limits to how densely information can be packed while remaining understandable. Can some of the annotation techniques used here, such as chunking, help? Similarly, the problems of overview remain when dealing with haptic devices, as does the need to make features explicit: can annotation with these types of information help users explore information haptically?

One area that has potential to significantly benefit users, which has been discussed here but not implemented, is graph similarity. As stated above, there are two difficult aspects to this: recognising similarities (particularly as this may involve judgements about how ‘important’ different parts of the graph are), and describing them. ‘Wizard of Oz’ type experiments\(^1\) could be used to test the benefits of this type of information, and explore what is most useful, before embarking on any difficult implementation.

Closely related to graph similarity is the idea of comparison. It is common that tasks involve comparing two graphs, or indeed other types of information. How can comparisons be made in the non-visual environment, without the external memory that makes visual comparison so simple? This is a crucial component in many types of task, and would form a fascinating avenue for further research. This could involve exploring how sighted users compare information of different types, including text, charts and diagrams, perhaps with the aid of eye-tracking equipment. Understanding this process could allow mapping to a non-visual equivalent, including development of a suitable mechanism for allowing appropriate flow of attention between objects, and means for detecting and summarising similarities and differences.

Having demonstrated that an audio glance of a graph can give an impression of its size and structure, there is plenty of scope for extending this concept. There are many potential methods for generating the Earcon, from simple algorithmic ones like those tested here, to techniques such as the mass-spring system, or even using a top-down approach, basing the sound on the presence of higher-order structures rather than the individual nodes. Again, experiments with sighted users might give further insight into the types of information that is actually obtained from the first glance at a graph; this could then inform what it is desirable to convey in the glance. Further exploration of the use of audio glances for orientation and highlighting might also be valuable.

\(^1\)That is, experiments where comparative information is presented to the user as if it had been calculated by the system, whereas it had actually been generated manually.
9.4. FUTURE WORK

Graphs are the source of much important information; allowing non-visual access to them is crucial to allow people with disabilities to participate fully in society. This thesis presents a technique that can help visually impaired people explore and understand graphs. Understanding why diagrams are good in print, and what difficulties are introduced by presenting through a different medium, gives insight into how these difficulties may be reduced. This work has shown that annotation is a powerful, and very flexible, technique for overcoming many of these difficulties, and should therefore provide a good foundation for continued improvement in this important field.
Bibliography


[99] TeDUB project. D5.2 Evaluation of the pre-prototype. Personal communication.

[100] TeDUB project. A study with domain experts. Personal communication.


Appendix A

Kekulé Evaluation Details

This appendix gives further details about the evaluation of Kekulé (Section 3.1).

A.1 Introduction for Participants

The following information about the evaluation was given to participants.

A.1.1 Introduction

Thank you for agreeing to take part in this experiment. The software you are to use has been developed with the aim of enabling blind students to understand the structure of organic molecules. The software is not intended to be a fully designed tool, rather it is a written to evaluate some techniques.

This experiment aims to determine if some of the techniques developed have an impact on the effort required to develop an understanding of molecular structures. Questions will be asked about two sets of three molecules, which are to be answered, if possible, by using the software to understand the molecule. You will be able to use the software while answering the questions. For one set of molecules you will have available only a restricted set of functions in the software, for the other you will be able to use all features. The two sets of molecules may come in either order.

A.1.2 The Software

This section gives an outline of the methods the software provides the user, and how to use it. Commands are given via the keyboard, each one by a single key,
and the output is synthesized speech.

A.1.2.1 Basic Mode

The software provides a basic method of browsing a molecule by moving from atom to atom along bonds. This is the method of browsing that will be used in the restricted software. In this mode you are effectively located on an atom and may move to any other atoms that are connected to it. The fundamental principle to bear in mind is that when an atom is spoken you may consider yourself ‘located’ on that atom. Note that hydrogen atoms are not treated individually, instead they are grouped with the atom to which they are connected. The number of hydrogens associated with any atom may be discovered with the ‘f’ command. The command set for this mode is given in Table A.1.

A beep can indicate two things. If there is no information, e.g., when the return key is pressed for more details, this will be indicated by a beep. Similarly a beep is used to indicate that there are no more choices. For example, if one is on an atom that is connected to three other atoms and press ‘c’ to move to one, the down arrow moves one between the choices. After the last option has been spoken a beep will indicate that there are no more connections. If the down arrow is pressed again the output will just be a beep. The mechanism is the same when moving back up using the up arrow.

Note that when connections are described, it is as a list and the connection described is that from the original atom (i.e., where one was located when the ‘c’ command was given).

As an example, consider the ethanol molecule, CH$_3$–CH$_2$–OH, shown in Figure A.1.

\[
\begin{align*}
\text{H} & \quad \text{O} \quad \text{H} \\
\text{H} & \quad \text{C} \quad \text{C} \quad \text{H} \\
\text{H} & \quad \text{H} \quad \text{H}
\end{align*}
\]

Figure A.1: Schematic representation of the structure of ethanol.
If we are located on the right-hand carbon, typing ‘n’ will give the following output:

‘Name: C’

This tells us we are on a carbon atom; typing return for more details gives:

‘Name: CH2 hash 2’

Now we know that this is atom 2, a carbon with two hydrogens. We can discover its connections by giving the command ‘c’:

‘2 connections. to CH3 hash 1’

Typing return tells us more about this connection, i.e., that it is a single bond:

‘bond to CH3 hash 1’

then typing ‘↓’ gives the next atom this is connected to:

‘to OH hash 3’

As this is the last of the connected atoms it is followed by a beep. Note that the meaning of these two connection statements is that our original atom (the right hand carbon) is connected via single bonds to both the CH$_3$ and the OH, not that it is connected to the CH$_3$, which is in turn connected to the OH. Since we are now located on the oxygen, typing ‘n’ results in:

‘Name: O’

And asking for the connections will tell us that there is a single connection to the CH$_2$.

Alternatively we could return to this atom by typing ‘←’:

‘Back to CH2 hash 2’ note that the connections described are those between the last atom and the connected groups

A.1.2.2 Advanced mode

When full functionality is available, the software detects structural features (such as rings and chains of carbon atoms) and (some) functional groups in the molecules. The browsing mode then becomes hierarchical. This means that the first view you get of the molecule will be at a higher level than atoms (the molecule as a whole), and it is possible to zoom in and out to explore the molecule at different levels of detail. Since the atoms get grouped, instead of moving between atoms you will be able to move between groups. The commands for this are, however, identical. Groups are given approximately correct chemical names, although since chemical nomenclature was not developed to describe parts of molecules they are not always entirely accurate. For example, numbers defining positions may not
be relative to the correct group, particularly on rings; they should, however, allow relative locations to be determined. If a name is not understood or recognised, the group may be explored in further detail by zooming in.

Detection of these structural and functional features in the molecule leads to a hierarchical data structure. For example a ‘carboxylic acid’ group is composed of a hydroxyl group and a carbonyl group, which are each composed of their individual atoms. Figure A.2 illustrates this concept.

![Carboxylic Acid](image)

**Figure A.2**: The hierarchical nature of the structure of carboxylic acid. The carboxylic acid group is composed of a hydroxyl group and a carbonyl group, both of which may also be decomposed into their constituent atoms.

There are some additional commands to make use of the extra functionality; these are outlined in Table A.2.

### A.2 Software Instructions

Participants were given a copy of either table A.1 or table A.2 (as appropriate — they would see both over the course of the evaluation) to help them remember
A.3 Questions

Participants were given one easy molecule and two complex molecules under each condition. Since participants were asked to rate the tasks as a whole (for each condition, i.e., 2 questions on each of 3 graphs), the simple graph was intended as practice before tackling the more complex ones. The graphs were selected randomly from the set (i.e., each participant encountered all the molecules) below. They were introduced to the first question with the following:

The following questions are intended to assess the techniques used by the software to present the structure of the test molecules; they are not designed to test you! The molecules in the test set range from extremely simple to very complex, and it is unlikely that you will be able to answer all of the questions. Please do not be afraid to just give up! Since we are assessing the system, it would be useful if you do not guess the answers; instead please explain any understanding you have of the problem. Similarly, if you don’t understand the question, please ask.

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Gives the name of the current atom.</td>
</tr>
<tr>
<td>c</td>
<td>Lists the atoms connected to the current one. The space bar or up and down cursor keys are used to move between them. Extra information from the ‘return’ command describes the nature of the connection (e.g., double or triple bonds).</td>
</tr>
<tr>
<td>↑, ↓</td>
<td>Move up and down through a list, such as the components of the formula or connected atoms.</td>
</tr>
<tr>
<td>r</td>
<td>Repeats the last item.</td>
</tr>
<tr>
<td>←, →</td>
<td>Move back and forward through your history (similar to web browsing).</td>
</tr>
<tr>
<td>return</td>
<td>Gives more information, e.g., about the nature of a connection.</td>
</tr>
<tr>
<td>p</td>
<td>Gives an approximate spatial location of the current group (e.g., ‘Top left’).</td>
</tr>
</tbody>
</table>

Table A.1: Commands available in the basic mode
### Table A.2: Commands available in the advanced mode

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Gives the name of the current group.</td>
</tr>
<tr>
<td>c</td>
<td>Lists the groups connected to the current one. The space bar or up and down cursor keys are used to move between them. Extra information from the ‘return’ command describes the nature of the connection.</td>
</tr>
<tr>
<td>↑, ↓</td>
<td>Move up and down through a list, such as the components of the formula or connected atoms.</td>
</tr>
<tr>
<td>f</td>
<td>Gives the formula of the current group.</td>
</tr>
<tr>
<td>r</td>
<td>Repeats the last item.</td>
</tr>
<tr>
<td>←, →</td>
<td>Move back and forward through your history (similar to web browsing), including zoom actions.</td>
</tr>
<tr>
<td>return</td>
<td>Gives more information, e.g., about the nature of a connection.</td>
</tr>
<tr>
<td>p</td>
<td>Gives an approximate spatial location of the current group (e.g., ‘Top left’).</td>
</tr>
<tr>
<td>+</td>
<td>Zoom in. Moves down into the hierarchy (i.e., towards atoms) Use the ↑ and ↓ keys to select the sub-groups.</td>
</tr>
<tr>
<td>−</td>
<td>Zoom out. Moves up out of the hierarchy (i.e., towards the molecule as a whole). In most cases a group is part of only one group, but the ↑ and ↓ keys allow selection in other cases.</td>
</tr>
<tr>
<td>w</td>
<td>Gives your current location in the hierarchy, e.g., ‘On hydroxyl, on carboxylic acid, on ethanoic acid.’ Use the up and down arrows to move through the hierarchy.</td>
</tr>
<tr>
<td>s</td>
<td>Gives a summary of the composition of the current group in terms of the number of rings, chains, functional groups and other atoms it is composed of.</td>
</tr>
</tbody>
</table>
A.3. QUESTIONS

Figure A.3: The first simple molecule (Molecule 1).

Figure A.4: The second simple molecule (Molecule 2).

A.3.1 Simple

A.3.1.1 Molecule 1

1. How many methyl groups are there in this molecule (a methyl group is any carbon atom with three hydrogen atoms attached)?

2. Describe how the carbon atoms are connected.

A.3.1.2 Molecule 2

1. How many atoms does the longest chain in this molecule contain (excluding hydrogens)?

2. Describe where the heteroatoms (atoms which are neither carbon nor hydrogen) are in relation to each other.

Figure A.5: The first complex molecule (Molecule 3).
A.3. QUESTIONS

A.3.2 Complex

A.3.2.1 Molecule 3

1. This molecule contains a primary amine (-NH\textsubscript{2}) that is connected to a carbon atom; to what other functional group (i.e., collection of atoms that containing atoms other than carbon or hydrogen) is that carbon also connected?

2. Describe the chain of atoms that connect the same carbon to the ring?

A.3.2.2 Molecule 4

1. How many carbonyl groups (any carbon double-bonded to an oxygen) are there in this molecule?

2. Describe what atoms are connected to the ring and their relative positions (e.g., 1-C, 2-P, 5-C to indicate that there is a carbon, a phosphorous on the adjacent position, then two positions with just a hydrogen, then a carbon)?

A.3.2.3 Molecule 5

1. Describe how the rings are connected.

2. Describe the relative positions of the chlorine atom and nitrile group (-C≡N) with respect to each other and the rings.

A.3.2.4 Molecule 6

1. Describe how the acid chloride (carbon double bonded to oxygen and single bonded to Cl) and amide (carbon double bonded to oxygen and single bonded to NH\textsubscript{2}) are connected.

Figure A.6: The second complex molecule (Molecule 4).
Figure A.7: The third complex molecule (Molecule 5).

Figure A.8: The fourth complex molecule (Molecule 6).
2. If the rest of the molecule joins the ring at position 1, what positions in the ring are connected by double bonds?

A.4 Results

Table A.3 gives the raw TLX scores given to the tasks by participants. The three graphs in each condition were considered a single task.

<table>
<thead>
<tr>
<th>Participant</th>
<th>MD</th>
<th>E</th>
<th>P</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12</td>
<td>10</td>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>20</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>12</td>
<td>16</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>14</td>
<td>18</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>13</td>
<td>11</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>18</td>
<td>18</td>
<td>10</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>15</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>20</td>
<td>14</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>20</td>
<td>15</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>10</td>
<td>13</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>17</td>
<td>16</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>12</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>18</td>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>16</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>9</td>
<td>18</td>
<td>17</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>16</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>18</td>
<td>15</td>
<td>4</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>9</td>
<td>7</td>
<td>13</td>
</tr>
<tr>
<td>11†</td>
<td>15</td>
<td>10</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>10</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>12</td>
<td>17</td>
<td>17</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>14</td>
<td>10</td>
<td>13</td>
</tr>
</tbody>
</table>

Table A.3: Raw TLX scores. The first row for each participant represents their scores for the basic version of the software, the second row the scores for the full version. The visually impaired participant is marked with a † next to the participant number. Columns give scores for each TLX index and their sum: MD is mental demand, E effort, P performance, F frustration.
Appendix B

Description Experiments

This appendix gives further details about the description experiments described in section 3.3.

B.1 Procedure

Participants were given a brief introduction to the experiments. They were told the purpose (that we were hoping to understand the vocabulary used and technique used to convey graph structures) and understood the wider context of the research (graph accessibility). They were given the instructions below, then were asked to decide who was to see the first graph. The graphs used in the experiment are given in the body of this thesis (Figures 3.4 and 3.5). These were always presented in the same order, as they were expected to roughly increase in difficulty through the experiment and it was anticipated that participants might improve with practice.

B.2 Participants Instructions

The following instructions were given to participants:

I am going to ask you to describe some graphs to each other. They are abstract graphs, that is, they are just nodes connected by arcs. One of you will be given a diagram showing the graph; I would like the other to ask questions in order to understand the structure of the graph. When you believe you understand the graph I will ask you
to draw it — you will not be able to draw or write anything down until this point. The spatial layout of the graphs is not important, but you may use it if you feel it is helpful. There will be eight graphs, and you will take turns as questioner/answerer. The eighth graph is slightly different as there are different types of edges and the nodes are labelled (it is based on a section of the London Underground map). On this last graph, I would like you to concentrate on the structure, but if you can also get the names, that would be ideal.

### B.3 Results

Of the seven abstract graphs across the four pairs, six were drawn incorrectly: Graph four (Figure 3.4(d)) once, by pair three, graph five (Figure 3.4(e)) twice, by pairs two and four, graph six (Figure 3.4(f)) once, by pair three, and graph seven (Figure 3.4(g)) twice, by pairs two and four. Graph eight (Figure 3.5) was not recreated completely by any of the pairs, although pair four recreated the structure correctly without node labels, and pair two recreated the structure correctly with many correct labels (but some labels misplaced).

Analysis of the dialogue was done by the experimenter: the exploratory nature of the experiment meant that bias was not thought to be a problem.

Table B.1 gives the times taken by the different pairs to convey the structure of the eight graphs. Note that one reason for the wide variation between pairs was the amount of confirmation they desired before being willing to draw the graph.
Table B.1: Approximate times taken by participants to convey the structure of the graphs. The values give the time elapsed between one participant seeing the graph and the other starting to draw, and are accurate to within approximately 5 seconds.
Appendix C

Audio Glance Experiments

This appendix gives further details about the audio summary experiments described in section 5.5.

C.1 Introduction for participants

The following introduction was given on the first page of the experiment web pages.

Evaluation of audio summaries for graphs

Introduction

I am currently researching how diagrammatic information (in particular node-edge graphs) can be most effectively conveyed, using a computer, to users who are unable to see the diagram, particularly blind or visually impaired users. An important facility that is lost to blind users is the ability to appreciate the gross structure and features of a diagram with a quick glance. As a part of my research I am examining if we can recreate the glance using audio information. I have developed a couple of algorithms for generating sequences of non-speech sounds that are dependent upon the structure of a graph. This evaluation is designed to determine if these algorithms can effectively give an impression of the structure of a graph. It is important to understand that although the structure of the sounds are directly related to the exact structure of the graphs, the sounds are intended only to give a vague impression of its size and shape. Because of this, for each graph you will be firstly asked to make a choice after only one listen. You will
then get the chance to listen again, as many times as you wish, and make another selection.

**Requirements:**

You will need to be able to play sound from Java applets, so you will need Java enabled in your browser, and your sound needs to be stereo, preferably headphones (otherwise well separated speakers). Your browser will also require cookies to be enabled. All the sound files are downloaded in one file of approximately 1.3MB when you view the first question, although you will need to remain connected to the internet so that results can be recorded. There is an example sound file on the first page of the questions which can be used to get the volume at a suitable level.

If you have any questions before or after completing the evaluation you may contact me at andrew.brown@cs.man.ac.uk.

**Format**

The evaluation consists of 4 phases. The first two are very similar, and test if the correct graph can be recognised by listening to the summary; the second two are more different, looking at if you can describe your impression of the graph, and if you can identify nodes highlighted in the sound. For all questions in this evaluation you will also be asked to rate how confident you are in your answer; please move the slider to the position that best reflects the extent to which you are guessing! Please also remember that this is not testing you, merely if the sounds can represent the graphs.

Below is an outline/instructions of the evaluation; full instructions are given later, but it is all quite simple. The whole evaluation should take somewhere between 30 and 45 minutes to complete - please remember that it is the impression that counts, so don’t spend too long thinking!

**Phase 1**

This consists of 5 multiple choice questions. For each you simply need to listen once to a sound (by clicking on the speaker icon button), then select which of the four graphs (presented as images) you think the sound represents. Selection is made by clicking on one of the circular radio buttons below and left of each graph.
When you submit your answer you will be shown the same question again, but you will be able to listen to the sound as many times as you wish. You again need to select which graph you think it represents; you may choose the same graph or a different graph to your choice first time round. When selecting the graph for the second time, you will also be asked to select which graphs you think are the most and least complex, and which pairs of graphs are the most and least similar (this is rating the graph structure only and is not related to the sounds). You might find this quite difficult; if so don’t worry too much about the answers, just go with your instinct.

Phase 2

This is identical to phase 1, but a different algorithm is used to generate the sounds. This new algorithm will be explained at the start of the phase (question 6).

Phase 3

The third phase of the evaluation asks you to describe your impression of 5 graphs, then sees if you are able to recognise the graph from your description. More detailed instructions will follow at the start of the phase.

Phase 4

The final phase looks at whether the audio graphs can be used to highlight certain features in the graph. As previously this will be done for 5 graphs. Full instructions are given at the start of this phase, although it is quite simple.

After the final question you will be given the opportunity to add any comments.

It would be useful (although optional) if you could give a name, or (if you are happy for me to contact you afterwards to ask about the evaluation) an email address: (text entry box)

Could you also let me know if you are using headphones or speakers (please use headphones if possible):
many thanks for your time.

Andy Brown

To start the evaluation press "Start"

(Start button)

C.2 Phase 1

The first questions in phase 1 all had the following layout; subsequent questions did not have the algorithm explanation; instead there was a link to an explanation of the algorithms. The graphs (Figure C.3.1) were presented in the same order for all participants.

Algorithm 1

The algorithm used in the first 5 questions is based largely on how the nodes are connected, with some influence from their positions. Each node is represented by a beep, starting with the left-most node; after a short delay there are beeps from all connected nodes. This continues to all nodes connected to these nodes, completing when all nodes have been played. In addition to resolving distance from the leftmost node (i.e., number of edges between) using time, the stereo effect is used to represent location, with those nodes nearer the top of the diagram being played towards the left and those nearer the bottom towards the right. Nodes that should be played simultaneously instead have a very small gap between them. The four images below give an example of this. Reading from left to right we see the sequence in which the nodes are played, with the nodes coloured red being played at that stage in the sequence. The dashed line on the first image approximately gives the centre of gravity - nodes above this will be heard on the left, nodes below on the right. You can listen to the sound for this example by clicking on this icon (you may need to wait until the sounds are downloaded before the speaker icon appears - this may take some time, depending on your connection speed):
C.3 Phase 2

The format of phase 2 was the same as phase one, although the first question had the following explanation about the different algorithm. The graphs (Figure C.5) were presented in the same order for all participants. The following information was presented to participants at the start of the this phase of the experiment:

Figure C.1: Figure explaining the algorithm used in the experiments. This was presented to users with the first question, and was accessible throughout the experiment.

In that sound sequence you should hear the sequence as groups of one, then two, then three, then finally one beep.

If you wish to be reminded of the algorithm, the above explanation can be opened in a new tab/window from the top of any of the question pages, or from here.

Question 1 of 10

Audio glance

Please listen to this audio summary once, and select which of the following graphs you think it represents.

(4 images in a 2 by 2 grid, with a radio button under each, labelled a – d)

How confident are you in your choice?

Just guessing Certain

(Slider control, set at mid-point)
C.3.1 Information

Click here to view the algorithm explanations (in a new tab/window).

Algorithm 2

The next five questions use a slightly different algorithm to generate the sounds. The sound for each beep is generated exactly as before; the only difference is that when nodes that form part of a ring are played, there is a background noise to emphasise this. For example, in the graph below, an ambient sound is played while the central 6 nodes are played. Listen here: *(Button to play sound)* *(Figure C.2 displayed here)*

![Figure C.2: Diagram of a graph with a ring. This was presented to the user with an audio glance for this graph, to explain how the presence of rings was made explicit.](image)

Different rings are played with slightly different pitches. For example, in the graph below, there are two ambient noises playing when the nodes that form the two rings are played; both are played while the two shared nodes are played. Listen here: *(Button to play sound)* *(Figure C.3 displayed here)*

![Figure C.3: Diagram of a graph with two rings. This was presented to the user with an audio glance for this graph, to explain how multiple rings were presented.](image)
Figure C.4: The graphs displayed in phase one of the experiment. Each row represents one question. The correct answers are highlighted.
Figure C.5: The graphs displayed in phase two of the experiment. Each row represents one question. The correct answers are highlighted.
C.4 Phase 3

The graphs for phase 3 are shown in Figure 5.6. They were presented in the same order for all participants.

C.4.1 Introduction

The following information was presented to participants at the start of the third phase of the experiment:
Click here to view the algorithm explanations (in a new tab/window).

Descriptions

That completes the second section of the evaluation. The next part looks at the same problem in a different way - you are asked to note your impressions of a graph after listening to its audio summary. You are later asked to see if you can match this description with a graph. As before, there are 5 graphs. The algorithm used is the same as that you have just been testing, i.e. rings are emphasised with ambient sound.
Please listen to the audio summaries below. You will be able to listen to each summary twice, after which you should write your impression of the graph structure in the text box to the right. This may be as detailed or as vague as your impression, e.g., ”very complex”, or ”a ring followed by a long chain”. In the next stage of the evaluation you will be asked to use these descriptions to select from a set of graphs.

(5 rows, each with a button to play the sound, and to its right a text field for entering a description.)
When you have completed all the graphs, press the submit button to continue.

C.4.2 Matching descriptions to graphs

The page displayed after pressing the submit button showed the following instructions.
Descriptions

In this section you will be given the descriptions you made (although not in the same order) and asked to match them up with the graphs shown. You will be able to select more than one graph as a match, but please only choose more than one if you are really unsure.

(Thirteen graphs were displayed, labelled ‘Diagram a’ to ‘Diagram m’. These are shown in Figure C.7)

Below are the descriptions you gave the 5 graphs after listening to audio glances of them. For each description please choose a graph from the selection above that fits best your impression. Please also use the sliders to indicate how confident you are that you have the correct answer.

(A continue button moved the participant to the next stage)

C.5 Phase 4

The graphs for phase 4 are shown in Figure 5.12. They were presented in the same order for all participants.

C.5.1 Introduction

The following instructions, for phase 4, were given on the next page:

Thank you. That completes the third phase of the evaluation. The next (and final) phase of this evaluation looks at whether audio glances can be used for orientation. We present the glances with one or more of the nodes highlighted (by use of a slightly higher pitch note); we want to test if this accurately conveys which nodes are highlighted. As an example, this glance: represents the graph below, with the nodes shown in red highlighted (click here to view the algorithm explanations)

(Figure C.6 displayed here)

We will again be looking at 5 different graphs, and for each of these the test has four parts. Firstly you will be given the audio glance for the diagram without highlighting. You will then be shown the graph. After looking at the graph you will be given the glance again, this time with one or more nodes highlighted; you will be able to listen to this twice only. After the second listen you will be
Figure C.6: Diagram of a graph with two nodes highlighted. This was presented to the user with an audio glance for this graph, to explain how highlighted nodes were presented.

presented with the graph again and asked to select which nodes you think were highlighted. Once done, you repeat this process for the next graph. To start the evaluation press "Start":

C.5.2 Instructions

The following instructions were visible throughout the final phase:
Click here to view the algorithm explanations (in a new tab/window)

Audio orientation evaluation:

(An initially blank area, with a button for playing the glance, and a continue button)

Instructions

There are 5 graphs, and 4 stages to each. These are:

1. Listen to an audio glance of the graph (without highlighting). You may listen up to six times.

2. View the graph. This is to help you understand the structure of the graph when listening to the highlights (as if you had been exploring the structure of the graph for a while).

3. Listen to an audio glance of the graph with one or more nodes highlighted. You may only listen twice. When listening please try to visualise which nodes are highlighted.
Figure C.7: The graphs from which the participant needed to choose graphs that matched his or her descriptions. The graphs were laid out a grid as here, in the same order.
4. Select which nodes you thought were highlighted. To do this you are presented with a picture of the graph again; clicking on a node highlights it (it should turn red), while clicking again clears that highlighting. When you are happy with your selection use the slider to show how confident you are in your choice then press the submit button. You are then taken to the first stage of the next graph.

C.5.3 Procedure

The participant could listen to the graph, then view it (as per items 1 and 2 in the instructions above. After clicking ‘continue’ to be able to listen to the highlighted graph (item 3 above), the following instructions were displayed above a button that, when pressed played an audio summary of the graph with some nodes highlighted:

Please listen to the audio summary by clicking on the button below. One or more nodes in this graph are highlighted. You will be able to listen twice, then shown the graph again, at which point you will be asked to identify which nodes were highlighted.

After the second listen, the following text appeared above a picture of the graph (with nodes represented as grey-filled circles:

Please select which node or nodes you think were highlighted in the audio summary you just heard. To select a node just click on it. To unselect it, click again. To unselect all, click the clear button.

Below the graph were a confidence slider and two buttons: a ‘clear’ button and a ‘submit’ button. The submit button was disabled unless at least one node was selected. Clicking on a node turned it red to indicate that it was selected. Figure C.8 shows a partial screenshot of the system with a single node highlighted. Once the submit button had been pressed, the participant returned to just seeing the play button, this time for the next graph.

The evaluation was completed with a screen thanking the participants and allowing them to leave comments.
C.6 Results

The following are the descriptions given by participants for the graphs heard in phase 3.

C.6.1 Graph 1

‘two rings followed by a long chain, complex.’
‘2 rings, long trailing chain.’
‘two rings, followed by a not necessarily straight chain, followed by a straight chain of 3 nodes.’
‘double ring followed by short chain.’
‘Two rings with long tail.’
‘2 rings, then a chain. Possibly 5-membered ring then 6-membered ring?’
C.6. RESULTS

‘Two rings connected directly together followed by a chain of 3 connected nodes.’
‘Two rings, second one bigger and then a linear tail.’
‘2 rings connected with a 4-5 chain off the second ring.’
‘two rings then short chain.’
‘Two rings, followed by chain..’
‘Two rings followed by a short chain.’
‘Double ring - regular followed by chain.’
‘R1-2-2-2-1-1-1-1, two rings.’
‘two rings then a string of nodes.’
‘two rings then a chain.’
‘2 rings followed by a chain.’
‘Two rings followed by a chain..’
‘a ring -> a ring -> a short chain.’
‘2 rings followed by a short chain.’
‘two rings followed by a short chain.’
‘two rings connected with 3 dots afterwards.’
‘two connected rings followed by a medium chain.’
‘Two rings followed by a chain.’
‘Two rings followed by a chain of 3.’

C.6.2 Graph 2

‘short chain of 3 nodes, very simple.’
‘3 dot line.’
‘a straight chain of three nodes.’
‘straight chain 3 long.’
‘Short Line.’
‘chain of 3 nodes.’
‘Three connected nodes.’
‘three linear nodes.’
‘3 long chain straight.’
‘chain with three nodes.’
‘simple chain.’
‘A short chain with three nodes.’
‘Chain - three nodes.’
‘1-1-1, no rings.’
C.6. RESULTS

’string of three nodes.’
’short chain.’
’3 dots in line.’
’A chain, I mean a number of nodes next to each other..’
’3 nodes in a chain.’
’short chain - 3 beeps.’
’3 dots in succession.’
’small chain (three nodes).’
’A chain with three nodes.’
’a chain of 3 nodes.’

C.6.3 Graph 3

‘no rings just a lot of branches, medium complexity.’
’dot braches to 2 then one braches to many and one of those linked to one more dot.’
’No rings. No straight chains - star topology?.’
’single point branching to two points then branching again.’
’Fork with uneven tails.’
’1 node, then 2 nodes, then 4 nodes (3?), with one further node. ”Tree structure”.’
’A single node connected two others, one connected to 3 the other to 2..’
’one node and then a bunch of nodes - but not in a ring.’
’1-2-4-1 chain, probably looks like a tree.’
’one node then ring then one node.’
’complex, no rings.’
’A chain with a branch.’
’node going into a branch.’
’1-2-4(3?)-1, no rings.’
’tree structure expanding from left to right’
’binary tree on its side.’
’2 rings.’
’I am not sure about this. It’s complex. It doesn’t have a ring, but I can tell it’s not a simple chain, may be moves ups and downs....’
’a node connected by 3 nodes. These nodes are connected to another node.’
'1 beep followed by either a branch or a star of beeps, more likely a branch with
2 beeps then a longer row of 3 or 4 then 1 beep.'
'1 dot followed by a fork then a second fork forming a tree structure as in the
previous examples.'
'a complex graph which is “ringless”, just nodes and chains, probably a star
chain.'
'Complex - a chain then a ring then a chain.'
'a tree, splitting into 2 then 3 on one side, and 1 on the other.'

C.6.4 Graph 4

'short chain followed by a ring, medium complexity.'
'difficult to follow. line then loop then some branching out of loop.'
'straight chain followed by ring followed by another branched chain..'
'short chain then ring then short chain.'
'Line followed by ring.'
'chain of 3 then 5-ring, then 4-ring? chain + 2 rings, at least..'
'A chain of nodes ending in a y-shape..'
'a line then a ring.'
'3 chain followed by single ring.'
'short chain then ring.'
'chain followed by one ring.'
'A chain followed by a ring structure.'
'short chain, small ring, node.'
'1-1-1-1-2-1, ring on final three or four.'
'string of nodes then a ring then a single node.'
'chain then ring.'
'chain then ring.'
'A chain followed by a ring then a smaller chain than the one at the beginning....'
'a chain of 3 nodes then a ring.'
'medium row (of 3) followed by a ring.'
'chain of dots then a ring then a smaller chain.'
'chain followed by ring.'
'A chain followed by a ring.'
'a chain of 3 followed by a ring of 6?.'
C.6.5 Graph 5

‘no rings just a few branches, simple.’
‘one dot branches to 3 lines.’
‘straight chain from which two branches eminate..’
‘1-2 3 3-1 Not sure - no ring?.’
‘Double fork.’
‘chain of 2 then 3 vertically, then 2 or 3 vertically.’
‘A single node branching to two equal arms, like a y-shape..’
‘two nodes then a bunch of nodes - but not in a ring .’
‘no ring, 2 chain with some branches.’
‘ring?.’
‘intersecting chains.’
‘A short chain with a tree structure.’
‘Node, going into branch.’
‘1-1-3-3, no rings.’
‘star shaped graph.’
‘short chain then like a binary tree.’
‘two rings with dot in the middle.’
‘A chain, complex bit then a smaller chain..’
‘a short chain then a branch to make a tree like structure.’
‘2 beeps (a short chain) followed by small branches or star, maybe two triplets .’
‘chain of dots followed by a branch of dots.’
‘chain followed by star chain.’
‘A tree-like structure with four leaves.’
‘a cain of three followed by several branches, longest on the right.’
Appendix D

Numbering Experiments

This appendix gives further details about the node numbering experiments described in section 6.4.

D.1 Introduction for participants

The following introduction was given on the first page of the experiment web pages.

Evaluation of numbering strategies for graphs

Introduction

I am currently researching how diagrammatic information (in particular node-edge graphs) can be most effectively conveyed, using a computer, to users who are unable to see the diagram, particularly blind or visually impaired users. Exploration of graphs from node to node requires a method for identifying and distinguishing nodes, and previous evaluation of an exploration system shows that a poor node numbering system can be confusing. The aim of this evaluation is to identify suitable strategies for numbering nodes in a graph.

Requirements:

The evaluation uses two applets, so your browser will need to accept Java applets. It will work with Java version 1.4 or later, and should work with 1.3, although this has not been tested. The applets need to connect to my server to record the results, so have been signed. If your browser won’t accept signed applets, or you
don’t accept the certificate the applets will not work. If in doubt, give it a try! The total download size of the pages and applets should be a little under 600kB.

**Format**

The evaluation consists of two sections. In the first part you are asked to number the nodes in 5 graphs. Numbering is with a simple series of integers. When numbering the graphs, you simply need to think about what system you think would be most logical when moving around the graph from node to node. In the second section you will be presented with a series of 9 graphs, each numbered using 5 different systems; you will be asked to rank them.

I would expect the entire evaluation to take between 10 and 20 minutes - you do not need to think too deeply about it! If you are on dial-up you may need to add another 5 minutes or so to download the applet for the part two.

After the final question you will be given the opportunity to add any comments. If you have any other questions or comments you can email me at browna@cs.man.ac.uk.

It would be useful (although optional) if you could give a name, or (if you are happy for me to contact you afterwards to ask about the evaluation) an email address:

*(Text entry field)*

Many thanks for your time.

Andy Brown

To start the evaluation press "Start"

*(Start button)*
D.2 Instructions for Part 1

After pressing the start button, the browser loaded a page displaying a graph with two buttons beneath (within a Java applet). Below these were the following instructions.

Please number each node in the graph. To number a node, click on it — the node will then be assigned the next number (whole numbers starting from 1). Clicking on the number again will remove its number and renumber all numbered nodes (so that numbering is continuous and they retain the original order). To remove all numbers and start again, click the clear button. When all nodes have been numbered click the submit button - your selection will be recorded and the next graph displayed.

The submit button was disabled until all nodes had been numbered.

Five graphs were numbered — these are shown in Figure 6.3. After the submit button was pressed on the last of these, the page for part two of the experiment was shown.

D.3 Instructions for Part 2

The page displayed the following text. Above the instructions section, a scroll pane showed five graphs with nodes numbered. Below this were ‘clear’ and ‘submit’ buttons.

Node numbering evaluation: part 2

That completes the first part of the evaluation. In the second part, below, you will be presented with a series of 9 graphs, each of which will be shown with 5 different numbering systems applied. You will need to rank these in order of preference (see instructions below).

Instructions

Please rank the graphs. When choosing, think of 'best' as meaning a system which you think makes most sense when moving around a graph (for example
you might like to keep as continuous a number sequence as possible). If you do not like any of the numbering systems, please just consider which you find least confusing. Some may seem equally good or bad, if so do not worry which rank you assign them.

The rank assignment works in the same way as node numbering in the previous section; you simply click on the graphs in order of preference, starting with whichever you consider best. When a graph has been clicked on, it will be assigned a rank, which is shown in red to its right. You can renumber or clear the numbering in the same manner as previously (click on a graph again to remove its ranking, and reorder the remainder, or click the clear button to remove all rankings). When all 5 numbering systems have been assigned a position, click the submit button - your selection will be recorded and the next graph displayed.

D.4 Graphs used in Part 2

The graphs used in part two are shown in Figure 6.4. For each graph five versions were presented in the scroll pane (these were presented in a random order for each participant); the numbering systems used can be inferred from Table 6.1.

D.5 Results

Table D.1 gives the number of first choice votes for each system in phase 2.
### Table D.1: Popularity of different numbering systems for phase 2.

Values give the number of participants who rated a particular numbering system as their favourite. Graphs 1 – 9 correspond to the graphs in Figure 6.4. Geom is the geometric system, Alpha is the alphanumeric system; Breadth and Depth are the breadth-first and depth-first tree numbering systems.

<table>
<thead>
<tr>
<th>graph</th>
<th>FEM</th>
<th>Geom</th>
<th>Alpha</th>
<th>Radial</th>
<th>Random</th>
<th>CML</th>
<th>Breadth</th>
<th>Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td>22</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>1</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>19</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>30</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>22</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>18</td>
<td>1</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>14</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>14</td>
<td>13</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>14</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix E

RDF Schema

The schema for the RDF used to describe and annotate graphs.

```xml
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

    <rdfs:Class rdf:ID="GraphObject">
        <rdfs:comment>Any object in a graph, including the graph itself</rdfs:comment>
    </rdfs:Class>

    <rdfs:Class rdf:ID="Node">
        <rdfs:comment>A node in a graph</rdfs:comment>
        <rdfs:subClassOf rdf:resource="#GraphObject"/>
    </rdfs:Class>

    <rdfs:Class rdf:ID="Arc">
        <rdfs:comment>An arc connecting more than 1 nodes</rdfs:comment>
        <rdfs:subClassOf rdf:resource="#GraphObject"/>
    </rdfs:Class>

    <rdfs:Class rdf:ID="Group">
        <rdfs:comment>A collection of nodes</rdfs:comment>
        <rdfs:subClassOf rdf:resource="#GraphObject"/>
    </rdfs:Class>

</rdf:RDF>
```
<rdfs:Class rdf:ID="Graph">
   <rdfs:comment>A collection of nodes that form a complete graph</rdfs:comment>
   <rdfs:subClassOf rdf:resource="#Group"/>
</rdfs:Class>

<rdfs:Property rdf:ID="hasNodes">
   <rdfs:comment>The nodes in a collection</rdfs:comment>
   <rdfs:domain rdf:resource="#Group"/>
   <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Seq"/>
</rdfs:Property>

<rdfs:Property rdf:ID="hasArcs">
   <rdfs:comment>The arcs in a graph</rdfs:comment>
   <rdfs:domain rdf:resource="#Graph"/>
   <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Seq"/>
</rdfs:Property>

<rdfs:Property rdf:ID="connects">
   <rdfs:comment>The nodes that an arc connects.</rdfs:comment>
   <rdfs:domain rdf:resource="#Arc"/>
   <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Seq"/>
</rdfs:Property>

<rdfs:Property rdf:ID="pointsTo">
   <rdfs:comment>The node towards which a directed arc is pointing</rdfs:comment>
   <rdfs:domain rdf:resource="#Arc"/>
   <rdfs:range rdf:resource="#Node"/>
</rdfs:Property>

<rdfs:Property rdf:ID="label">
   <rdfs:comment>An alternative name</rdfs:comment>
</rdfs:Property>
<rdfs:domain rdf:resource="#GraphObject"/>
</rdf:Property>

<rdf:Property rdf:ID="note">
  <rdfs:comment>Any comment</rdfs:comment>
  <rdfs:domain rdf:resource="#GraphObject"/>
</rdf:Property>

<rdf:Property rdf:ID="history">
  <rdfs:comment>When the object has been visited</rdfs:comment>
  <rdfs:domain rdf:resource="#GraphObject"/>
</rdf:Property>

<rdf:Property rdf:ID="property">
  <rdfs:comment>Any property, which should have a name and a value</rdfs:comment>
  <rdfs:domain rdf:resource="#GraphObject"/>
</rdf:Property>

<rdf:Property rdf:ID="home">
  <rdfs:comment>Is this node the home node</rdfs:comment>
  <rdfs:domain rdf:resource="#Node"/>
</rdf:Property>

<rdf:Property rdf:ID="relationship">
  <rdfs:comment>The relationship between this node and the home node</rdfs:comment>
  <rdfs:domain rdf:resource="#Node"/>
</rdf:Property>

<rdf:Property rdf:ID="number">
  <rdfs:comment>A unique number for the node</rdfs:comment>
  <rdfs:domain rdf:resource="#Node"/>
</rdf:Property>

</rdf:RDF>
Appendix F

Evaluation Details

This appendix gives further details about the evaluation, including the participants instructions, the graphs used and the questions asked. The raw TLX scores are also given in section F.5.

F.1 Introduction

The following information was given to participants as an introduction to the aims and structure of the experiment:

Aims of this evaluation.

Our research is looking at how to improve access to diagrams by visually impaired people; we are examining a class of diagrams that can be classed mathematically as graphs - these are where we have nodes connected by arcs (or things connected by lines, if you like), like family trees or the London Underground map. We’ve developed a tool to test some of our ideas, and that’s what this evaluation is all about. What we’re going to do today is use the tool to solve some problems with graphs to see what difficulties occur and what features are helpful or otherwise. We’re going to use two types of graph - family trees and logic circuits, and you don’t need to have any prior knowledge of the structures of these graphs. Possibly the most important thing to remember is that we are testing the tool, not you: don’t worry if you have difficulties, or can’t do a task - it is still interesting for me.
Structure of the evaluation.

To start I’ll tell you a little about how the system works, then I’ll let you have a play on a simple test graph - this should let you get used to the interface and the commands available. There will be time to become familiar with each type of graph if you have not used them before. Once you’re happy with it all, you can answer the questions - we’ll start with (family trees / logic circuits). There will be two questions on each of two graphs. When those have been completed we’ll move on to (logic circuits / family trees). Before you answer the second set of questions on you will get the chance to have another practice with a simple example graph. The features and commands will not be the same for the two domains, but the general interface of the system will be.

Overall, I’d expect this to take between 60 and 90 minutes.

During the evaluation, I will be recording our conversation, and the software will be logging the commands and its output. All this information will be stored anonymously. The recordings will only be used by me as a reminder of tactics you use and any things you might find difficult, although short anonymous quotes may be used when writing up this research.

If you happy with this, I have a consent form for you to sign, and a brief questionnaire about you experience with computers, and with the domains we’re using today.

F.2 Questionnaire

The following questionnaire was given to participants before they started the training. The forms were given to sighted participants to complete; they were read out to those who were unable to read them.
Participants questionnaire:

Participant number: .

Please answer the following questions about your experience with computers, and with the domains we’re using today.

Please indicate how long you have been using computers:
[ ] Less than a year [ ] 1-5 years [ ] 5-10 years [ ] More than 10 years

Please indicate how often you use a computer (for any purpose):
[ ] Rarely / never [ ] Monthly [ ] Weekly [ ] Daily

What assistive technologies do you use (on the computer):

Please indicate your experience with family trees:
[ ] None [ ] Occasional view [ ] Regular viewing [ ] Occasional editing [ ] Regular editing

Please indicate your experience with logic circuits:
[ ] None [ ] Occasional view [ ] Regular viewing [ ] Occasional editing [ ] Regular editing
F.3 Software Instructions

The following information formed the basis for instructing the participants on how to use the software:

Okay, so let’s move on to how the system works. Now it is a little complex, so I don’t expect you to remember everything - you will be allowed to ask how to do things. In fact this is useful to me, so if there is anything you would like to be able to do, just say! I’d actually like to encourage you to think aloud as much as possible while you are doing the tasks; anything you say will be useful - this is why I’m recording the evaluation.

The tool uses the keyboard for input - you have access to menus (like JAWS) and shortcuts for many commands.

It allows you to explore the graph by moving around from node to node, although there are also summaries to help give you an overview.

The thing to remember is that you are always on a node, and whatever command you give relates to that node. Whenever the name of a node is spoken you move to that node.

The interface is based on lists, for example you start on the list of all nodes, and can move up and down this list between the nodes. If you issue a simple command (e.g., what is the name of this node) the answer will be spoken; you will still be able to move up and down the list of all nodes. If you issue a command that returns a list of items (e.g., connections) a new list is created and you move to the first item on the list. Moving up and down the list gives you all the connected nodes. You can also return to the original list.

Summary commands relate to the whole graph.

Specific commands. Demonstrate, with the participant in control (starred items to be introduced before annotated version only):

Lists - up, down and how a command relates to the last item spoken (Name command).

Speech commands: repeat, more info, faster/slower.

Simple commands - node name, id, location*
F.4. QUESTIONS

List commands: properties, then demonstrate moving back to previous list. Position in list command.

Connections. Point out ‘seen before’ noise* and visit history*.

Labels*, home node*, notes*.

Summaries*.

Searching.

Participants were allowed to use the system to explore a graph. When they felt happy with how it worked, they moved on to the first of the questions.

F.4 Questions

F.4.1 Logic Circuits

The graphs used for the logic circuits domain are given in Figures F.1 and F.2. These were designed to be similar, to determine whether participants were able to identify this similarity. For Figure F.1, the questions were:

1. How many And gates are there?
2. If the inputs are A=1 and B=0, what is the value of output C?

For Figure F.2, the questions were:

1. How many gates are there (And, Or, Not)?
2. If the inputs are A=1 and B=0, what are the values of outputs C and D?

F.4.2 Family trees

The graphs used for the logic circuits domain are given in Figures F.3 and F.4. The questions for Figure F.3 were:

1. How many people are there in this tree who do not have the surname ‘Brown’?
Figure F.1: The first logic diagram. Inputs are $A$ and $B$, top; the output is $C$. 
Figure F.2: The second logic diagram. Inputs are A and B, top; the outputs are C and D.
2. Who is the youngest person in this family tree?

The questions for Figure F.4 were:

1. How many generations are represented in this family tree?

2. How is the emperor Tiberius related to the emperor Claudius? For this question, participants were warned that the names of the emperors were not necessarily the same as their given names, but that the former could be found in the properties.

F.5 TLX scores

The raw TLX scores are given in Table F.1.
Figure F.4: The second family tree. The dictator Julius Caesar, and those persons who were Emperors have their official names given in red; these were given as properties of the person node (participants were told that this was the case).
Table F.1: Raw TLX scores. Visually impaired participants are marked with a † next to their participant number. Columns give scores for each TLX index and their sum: MD is mental demand, E effort, TD temporal demand, P performance, F frustration.