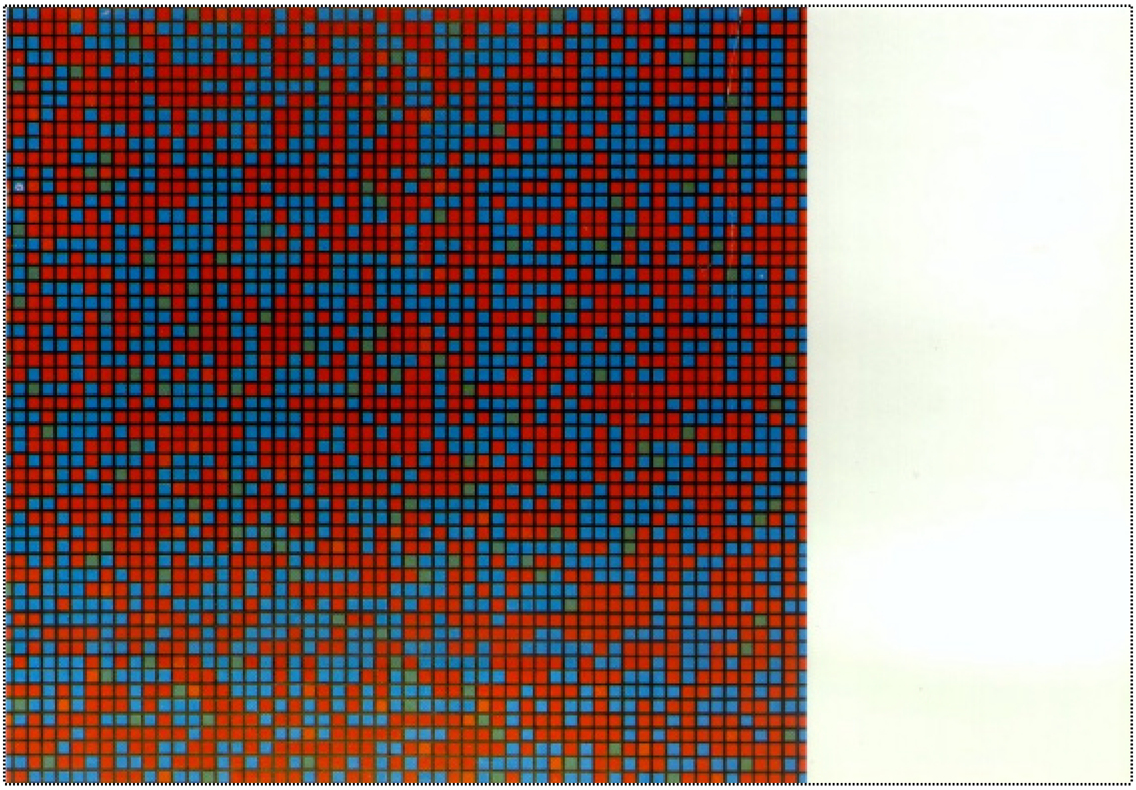
Visualising Multiple Overlapping Classification Hierarchies

Martin James Graham

A thesis submitted in partial fulfilment of the requirements of Napier University for the degree of Doctor of Philosophy

December 2001



François Morellet – répartition aléatoire (40% bleu, 40% rouge, 10% vert, 10% orange) - 1960

Abstract

The revision or reorganisation of hierarchical data sets can result in many possible hierarchical classifications composed of the same or overlapping data sets existing in parallel with each other. These data sets are difficult for people to handle and conceptualise, as they try to reconcile the different perspectives and structures that such data represents. One area where this situation occurs is the study of botanical taxonomy, essentially the classification and naming of plants. Revisions, new discoveries and new dimensions for classifying plants lead to a proliferation of classifications over the same set of plant data. Taxonomists would like a method of exploring these multiple overlapping hierarchies for interesting information, correlations, or anomalies.

The application and extension of Information Visualisation (IV) techniques, the graphical display of abstract information, is put forward as a solution to this problem. Displaying the multiple classification hierarchies in a visually appealing manner along with powerful interaction mechanisms for examination and exploration of the data allows taxonomists to unearth previously hidden information. This visualisation gives detail that previous visualisations and statistical overviews cannot offer.

This thesis work has extended previous IV work in several respects to achieve this goal. Compact, yet full and unambiguous, hierarchy visualisations have been developed. Linking and brushing techniques have been extended to work on a higher class of structure, namely overlapping trees and hierarchies. Focus and context techniques have been pushed to achieve new effects across the visually distinct representations of these multiple hierarchies.

Other data types, such as multidimensional data and large cluster hierarchies have also been displayed using the final version of the visualisation.

Contents

List of Figures

Acknowledgements

Thanks to my family and Eleanor. Thanks to Jessie Kennedy, David Benyon and Chris Hand for supervising and guiding my PhD at Napier, and also to the RBGE for acting as guinea pigs for my work and their input, especially Mark Watson, Mark Newman and Martin Pullan. My research was funded by an EPSRC studentship.

Contributing Papers

Graham, M., Kennedy, J. B. and Hand, C. (1999) The Challenge of Visualising Multiple Overlapping Classification Hierarchies. Proc. of User Interfaces to Data Intensive Systems (UIDIS '99) (Edinburgh, UK, September 5-6, 1999), IEEE Computer Society Press, p. 42-51.

Graham, M., Kennedy, J. B. and Hand, C. (2000) A Comparison of Set-Based and Graph-Based Visualisations of Overlapping Classification Hierarchies. Proc. of AVI 2000 (Palermo, Italy, May 23-26, 2000), ACM Press, p. 41-50.

Graham, M., Kennedy, J. B. and Benyon, D. (2000) Towards a methodology for developing visualisations. International Journal of Human-Computer Studies, 53 (5), p. 789-807.

*Raguenaud, C., Graham, M. and Kennedy, J. (2001) Two approaches to representing multiple overlapping classifications: a comparison. Proc. of SSDBM 2001 (Fairfax, Virginia, USA, July 18-20, 2001), IEEE Computer Society Press, p. 239-244.

*The author’s contribution consisted of a discussion of the rationale behind the underlying multiple hierarchy model used by the visualisation, and a description of the operations possible with such a model, as detailed in Chapter 3.4.

Graham, M. and Kennedy, J. (2001) Combining linking & focusing techniques for a multiple hierarchy visualisation. Proc. of IV 2001 (London, UK, 25-27 July, 2001), IEEE Computer Society Press, p. 425-432.

1 Introduction

A common complaint presently attributed to information working is that of 'information overload' – the inability of the human brain to process, recall, or even remember to recall, the ever-increasing amounts of information that are being produced. The overhead of dealing with the information bulk itself reduces the ability to analyse or understand it in any detail, and the outcome of this cumulative effect is that much information is ignored or given a scant once-over, and higher level connections between pieces of information are left undiscovered.

The practice of Information Visualisation (IV) is one approach to dealing with such problems. The argument is that if the information is displayed in such a manner as to take advantage of the strengths of the human perceptual system, a user's cognitive abilities are freed to ask more involved, complex questions. Simply put, if a user can perceive the 'who', 'what', 'where' and 'whens' without effort, they are free to think towards the 'hows' and 'whys'.

Such questions are being asked of new types and classes of information, among them structures known as *multiple overlapping hierarchies*, or as Robertson [149] terms them, *polyarchies* – structures formed from re-organisations or evolutions of hierarchical information sets and the inter-relationships the various hierarchies thus form with each other. These structures occur in taxonomies, ontologies, clustering algorithms, and hierarchical data with multiple indices such as digital libraries. Not surprisingly, understanding the full complexity of these data sets in a traditional, textual manner is difficult, requiring constant cross-referencing, checking and back-tracking across the individual hierarchies that form the overall structure. An example of such a task attempted on a group of five hierarchies is shown in Figure 1.1, in which a taxonomist was attempting to explain the nature of the problem with regard to their data sets.

In short, such a method of working spends a vast amount of time focusing on the 'wheres' and 'whens', leaving a bored, frustrated worker little time to attend to the connections between the information and the reasons for them.

An Information Visualisation of these structures should, if designed properly, enable a user to pay much more attention to these 'hows' and 'whys', indeed, even encourage them to undertake such a task at all when previously the effort appeared to outweigh the potential gain.

Currently though, Information Visualisation techniques have not been designed to cope fully with the demands of investigating and visualising multiple hierarchies, a fact that has also been pointed out by Robertson.

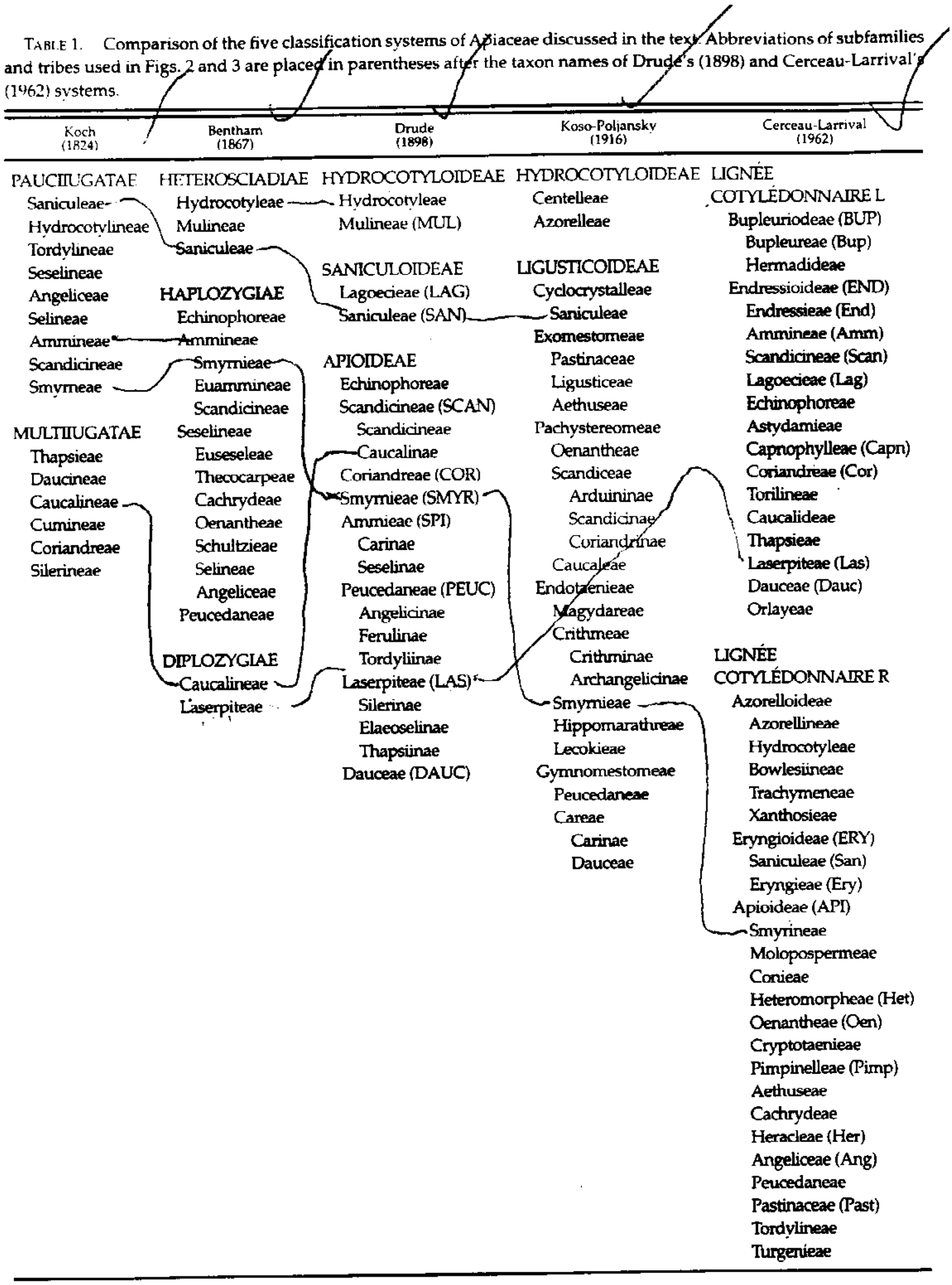


TABLE 1. Comparison of the five classification systems of Apiaceae discussed in the text. Abbreviations of subfamilies and tribes used in Figs. 2 and 3 are placed in parentheses after the taxon names of Drude's (1898) and Cerceau-Larrival's (1962) systems.

Koch (1824)	Bentham (1867)	Drude (1898)	Koso-Poljansky (1916)	Cerceau-Larrival (1962)
PAUCIIUGATAE	HETEROSCIADIAE	HYDROCOTYLOIDEAE	HYDROCOTYLOIDEAE	LIGNÉE
Saniculeae	Hydrocotyleae	Hydrocotyleae	Centelleae	COTYLÉDONNAIRE L
Hydrocotylineae	Mulineae	Mulineae (MUL)	Azorelleae	Bupleuriodeae (BUP)
Tordylineae	Saniculeae			Bupleureae (Bup)
Seselineae		SANICULOIDEAE	LIGUSTICOIDEAE	Hermadideae
Angeliceae	HAPLOZYGIAE	Lagoecieae (LAG)	Cyclocrystalleae	Endressioideae (END)
Selineae	Echinophoreae	Saniculeae (SAN)	Saniculeae	Endressieae (End)
Ammineae	Ammineae		Exomestomeae	Ammineae (Amm)
Scandicineae	Smyrnieae	APIOIDEAE	Pastinaceae	Scandicineae (Scan)
Smyrneae	Euammineae	Echinophoreae	Ligusticeae	Lagoecieae (Lag)
	Scandicineae	Scandicineae (SCAN)	Aethuseae	Echinophoreae
MULTIIUGATAE	Seselineae	Scandicineae	Pachystereomeae	Astydamieae
Thapsieae	Euseseleae	Caucalinae	Oenantheae	Capnophylleae (Capn)
Daucineae	Thecocarpeae	Coriandreae (COR)	Scandiceae	Coriandreae (Cor)
Caucalineae	Cachrydeae	Smyrnieae (SMYR)	Arduininae	Torilineae
Cumineae	Oenantheae	Ammieae (SPI)	Scandicinae	Caucalideae
Coriandreae	Schultzieae	Carinae	Coriandrinae	Thapsieae
Silerineae	Selineae	Seselinae	Caucaleae	Laserpiteae (Las)
	Angeliceae	Peucedaneae (PEUC)	Endotaenieae	Dauceae (Dauc)
	Peucedaneae	Angelicinae	Magydareae	Orlayeae
		Ferulinae	Crithmeae	
	DIPLOZYGIAE	Tordyliinae	Crithminae	LIGNÉE
	Caucalineae	Laserpiteae (LAS)	Archangelicinae	COTYLÉDONNAIRE R
	Laserpiteae	Silerinae	Smyrnieae	Azorelloideae
		Elaeoselinae	Hippomarathreae	Azorellineae
		Thapsiinae	Lecokieae	Hydrocotyleae
		Dauceae (DAUC)	Gymnomestomeae	Bowlesiineae
			Peucedaneae	Trachymeneae
			Careae	Xanthosieae
			Carinae	Eryngioideae (ERY)
			Dauceae	Saniculeae (San)
				Eryngieae (Ery)
				Apioideae (API)
				Smyrineae
				Molopospermeae
				Conieae
				Heteromorpheae (Het)
				Oenantheae (Oen)
				Cryptotaenieae
				Pimpinelleae (Pimp)
				Aethuseae
				Cachrydeae
				Heracleae (Her)
				Angeliceae (Ang)
				Peucedaneae
				Pastinaceae (Past)
				Tordylineae
				Turgenieae

Figure 1.1. Tracking across multiple hierarchies by hand. Original text from Plunkett *et al* [141].

1.1 Aim of PhD Research

Thus, the aim of this thesis work is to use Information Visualisation to address the problems of supporting users in working with multiple overlapping hierarchies, the information structures that represent different complete or partial classifications of a set of objects. In particular, these structures occur in the field of taxonomy, in which the repeated classification and re-classification of objects such as plants and animals results in large, complex datasets of historical information which need to be understood and manipulated by specialist biologists, known as taxonomists.

The main tasks in using multiple taxonomies, currently only possible using paper-based systems, are the discovery and understanding of the distribution of single objects and groups across differing hierarchies, as is demonstrated by the lines in Figure 1.1. Further, the comparison of the taxonomic hierarchies' overall structure and groupings are of interest to taxonomists.

This research tackles this problem through the use of established techniques, and where necessary, the development of novel IV techniques. This is achieved through the understanding of the current techniques used in IV for such problems, and their application to this particular problem. Using such background knowledge, prototypes were developed for user testing with taxonomists, upon which an iterative cycle of design, development and evaluation was conducted to arrive at a suitable final visualisation.

1.2 Thesis Organisation

The main contribution of this thesis is the ability to visualise and examine radical *structural change* across a set of multiple overlapping hierarchies. This itself involved the development of space-efficient tree visualisations and the extension of linking techniques to multiple tree structures and focus + context techniques. This thesis work is organised by chapters into the following segments:

Chapter 2 gives a review of existing IV techniques and principles, which are categorised using an existing framework of display and interaction techniques. Seminal and state-of-the-art developments in IV are noted.

Chapter 3 describes the practice of taxonomy, specifically how the field gives rise to the problematic data sets of multiple, overlapping hierarchies. Information visualisation is proposed as a possible solution to understanding such structures, and describes a set of requirements that such a visualisation should be able to fulfil.

Chapter 4 continues on from the preceding two chapters by examining existing IV applications that can handle and display multiple trees, and for each of these visualisations describes why they are not suitable for the taxonomy data and/or its requirements. Thus, grounds for developing an original IV for this problem area are established.

Chapter 5 outlines a methodology for developing a suitable IV, and proceeds to describe the ideas and development behind two differing prototypes after representative paper sketches were shown to and commented upon by the taxonomists.

Chapter 6 elaborates on this methodology with the detail of an iterative cycle of development, testing, and re-design. Various properties of the prototypes are tested and probed at each cycle up until a final test is conducted with nineteen taxonomists using the final prototype.

Chapter 7 describes the extension of the visualisation to encompass information from two other domains, multi-dimensional data and the outcomes of multiple clustering algorithms.

Chapter 8 discusses where this thesis work fits within the existing body of IV research, what specific contributions it makes, and what possible future work it could lead into.

2 Information Visualisation

The field of Information Visualisation (IV) is concerned with reducing the cognitive overheads of understanding complex information structures through the use of visual representations. IV achieves this by utilising appropriate visual metaphors composed of representative and interactive elements that take advantage of the strengths of human visual perception.

The strength of information visualisation is its applicability to numerous areas throughout the workplace and science, wherever information is being handled. However, for compiling a review of the subject, its strength becomes a weakness, as relevant literature is strewn throughout the journals and conferences of computer science. Lately, conferences such as the IEEE's Information Visualization symposia have acted as gathering points for new research, but still, at the time of writing, there is no specific information visualisation journal (though one is now in preparation.) Even the difference in spelling between the British and Americanised forms of 'visualis(z)ation' can cause difficulties in automated searches of the literature.

2.1 Origins in Related Fields

Information visualisation evolved from scientific visualisation (SV) in the early 1990s, with the first paper to explicitly address IV concerns being the Information Visualizer by Card *et al* [31]. The two main differences between the new field and its originator concern the qualities of the data to be visualised, and the type of user who is to view the resulting visualisation. Gershon and Eick [71] also make a third distinction on the different types of task that are likely to be performed with IV compared to SV systems.

The first difference concerns the data sets, which are more abstract than the orthogonal physical data sets analysed in SV applications such as Geographical Information Systems (GIS). IV data dimensions tend to be more artificial than the natural, physical dimensions of SV data sets. Links are the dominant measurement between IV data, so abstractions can be a feature of the relations between data as well as being a quality of the data itself. IV data also, due to its abstractness, tends to be more qualitative and discrete, compared to the quantitative, continuous nature of the data and events that are modelled and measured in SV fields. This last feature explains why IV refers to *information* rather than simply *data*. Kreuseler *et al* [107] present a more formal, set-based definition of some of these properties.

The second major difference concerns the type of users involved in IV. They tend to be less specialised than the scientists who utilise SV systems, and cover a broader band of work areas, such as business and finance, information handlers such as librarians and researchers, and computer scientists themselves.

The bulk of current users of IV systems are computer scientists, simply because they are the people who develop the systems and techniques. As such, they tend to develop applications to overcome problems they or their peers face. An example of this is Seesoft [55], developed for software maintenance. In fact, this type of system, generically called “software visualisation”, has its own field which pre-dates IV. It can be argued that logically it is a subset of IV but due to its greater maturity it is still generally regarded as a separate field.

Because of this greater diversity of potential users, IV appears to have “placed more emphasis on the human side of the equation” [49] when compared to SV. Gershon, Eick, and Card state that IV “combines aspects of SV, human-computer interfaces, data mining, images, and graphics.” [72]

An interesting point is made by comparing financial visualisations [114] and cosmological visualisations [128]. Money can be counted, operates according to mathematical principles on a continuous scale, and is represented physically, which appears to make it a non-abstract dimension. However its users, such as financial consultants, fit the category of typical IV users, and due to this fact financial visualisations tend towards being classified as IV systems. By the same token, time can be measured but never seen, and physicists have developed various space-time visualisations that question its very orthogonality to the spatial dimensions. However, physicists are scientists, so such displays are classed as applications of SV. (A physicist would also say their information is absolutely fundamental and definitely not abstract, but a stockbroker would probably say it was very abstract indeed. After all, what is abstract to one person may not be to another and vice versa. Maybe ‘non-physical’ would be a clearer term than ‘abstract’ as a defining quality for IV data sets.)

Perhaps the intended task defines what is IV more than the information, as a third distinction is also made on the basis of task focus [71]. This distinction argues that users of IV tend to want to find trends or specific pieces of information, rather than the deep understanding of the whole data set required by scientists. Therefore, the emphasis moves towards searching and discovery techniques, and this is another area of IV research. The distinctions as to what is IV rather than SV are shown to be subjective in any case.

Differing views on the exact definition of IV can be found, exemplified by Gershon and Eick [70], Chalmers [35], and Jern [95], and all to some extent reflect the authors' own views and backgrounds. Jern, for example, states that IV is database-oriented and multi-dimensional, reflecting his background in scientific visualisation, whilst Gershon and Eick stress the difference between *information* and *data*; information tending to be qualitative and data being quantitative. However, all are agreed on the main differences between IV and SV, and on the focii and divisions of IV research that are to be outlined in this chapter.

2.2 Information Visualisation Space

Most IV literature concentrates on specific applications that are designed to overcome specific problems by applying and extending existing IV techniques. In contrast, a number of papers attempt to give a general overview of the constituent parts of IV and a concise description of what it entails, though the scope and design space are still poorly defined. The consensus is that IV consists of display and interaction techniques that take advantage of the human perceptual systems, specifically vision, to increase the understanding of abstract information. Card *et al* state this concisely by proposing that IV is "the use of computer-supported, interactive, visual representations of abstract data to amplify cognition." [30, Ch. 1] A call for the other human senses, especially hearing, to be included in a larger domain of "information perceptualization" is made by van Dantzich [47], as at the moment research is almost exclusively focused upon vision.

Shneiderman [155] offers an IV taxonomy of seven data types by six tasks that users could wish to perform, the taxonomy being designed to mark out some of the data sets and situations that can benefit from IV techniques. The data types are temporal, 1D, 2D, 3D and Multi-Dimensional data, along with hierarchical and network data. The tasks include overview, zoom, and others that are combined into what is described as the Visual Information Seeking Mantra, "Overview first, zoom and filter, then details-on-demand", put forward by Shneiderman to be a useful generalisation for most IV-based tasks. However, Shneiderman himself states that the "seven data types are an abstraction of the reality. There are many variations on these themes." [155, p. 339] Also the multi-dimensional category has been criticised as being a catchall for data types that do not fit in any of the other categories.

Card and Mackinlay [29] developed a semi-formal notation that describes visualisations as mappings of data values (nominal, ordinal, quantitative) onto visual attributes such as colour, size and positioning. They described the common visualisations of the time with their notation and also used it to suggest new combinations. Chi's [39] model is more detailed and focuses on the steps needed to produce a given visualisation from its data. The model categorises and defines existing visualisations via a series of abstractions and transformation processes along a path from raw original data to final visualisation, and also incorporates the interactions that are possible with the data at each stage. These more formal approaches, as compared to Shneiderman's, offer the lure of identifying and eventually being able to re-use steps or processes that are found to be common across many visualisations. For example Chi's examination of common visualisations with his taxonomy enabled him to state that "many hierarchical techniques share similar operating steps that can be standardised in a system." Indeed, Kazman and Carrière [99] identified this aspect of hierarchy IV's, producing an IV software architecture that allowed hierarchical data to be rapidly and easily transformed into different visualisation styles, with only the final display parameters and metadata differing between the majority of such visualisations.

Gershon, Card and Eick's tutorial [69] divided information visualisation research into a simpler but still wide-ranging taxonomy composed of the sub-topics of human visual perception, display techniques, and interaction techniques. In turn, the display and interaction technique categories are broken down into appropriate sub-categories such as hierarchies, node-and-link, and 'other' for the display approaches, and focusing, filtering, and linking techniques for interaction. While this suffers from the same problem of using a catch-all display category ('other') as Shneiderman's data-only classification, this is not a severe problem as this thesis is mainly concerned with hierarchies and graphs (node-link data), two identified display sub-categories. Also, their classification was designed to deliver an overview of IV research, and as such, a review of information visualisation research is presented that uses their classification framework as a guide. It must be remembered that the boundaries between the sub-groupings they describe are fuzzy and, as IV systems are composed of perceptive, display *and* interactive components, some work will have relevance to more than one of these categories, though they are arranged according to their primary significance with respect to the work in this thesis.

2.3 Human Visual Perception

One important, fundamental area of research for IV is concerned with the properties of human perception of visual information. This field addresses questions regarding a spectrum of perceptual cues and their effects in conjunction with the human visual system, producing the foundational experimental evidence on which IV techniques are, or at least should be, based. In practice most of the literature regarding perception within the IV community tends to concentrate on the effects of various aspects of motion or colour.

The power of many basic visual attributes such as colour, size and brightness stems from the fact they are part of a set of attributes known as *pre-attentive cues*. These are visual attributes that immediately "jump out" from a scene without any conscious effort on the part of the viewer, the elements already tagged by the human visual processing system as being particularly worthy of attention. Common consensus states that visual properties are split into two groups, one consisting of these pre-attentive cues and the others of attributes that require cognitive effort to be processed and identified. Lately, research in psychophysics has indicated that there is a continuous range of cues instead of this dichotomy. As such, there can be given an ordering of perceptual power of visual attributes, a selection of which is given by Csinger [46] as, in descending order of perceptual potency: motion, colour intensity, colour hue, and lastly shape.

2.3.1 Motion

Motion related studies have dealt with various investigations that evaluate motion's abilities to increase the power and comprehension of IVs, and also into its basic properties in co-ordination with the human visual system. One example of research into using motion to aid comprehension involves its suitability as an aid for dis-entangling otherwise ambiguous 3D graph structures, as in Hubona *et al* [90] and Ware and Franck [170]. Their findings show that rotation of a 3D network made its structure much clearer than was the case with static representations, even if stereoscopic-depth views of the static structure were used.

Among fundamental research into the properties of motion, Bartram [9] explored motion as a technique for encoding further dimensions of an information set, stating that perceptual properties such as harmonic oscillation, blinking, and frequency of movement were suitable candidates for pre-attentively representing some types of information. Bartram expanded on this later, theorising that objects oscillating with a common phase had an almost Gestalt-like ability to stand out as a group from other objects [10], and demonstrated that animation was an especially strong attractor in peripheral vision, where other effects such as colour and shape are physiologically less effective [13]. McCrickard *et al* [120] backs this up by showing that *occasional* animation at the viewer's periphery of vision did attract a user's attention and, furthermore, was not significantly distracting in terms of inducing errors as had first been thought. Their work also demonstrated that monitoring constantly moving ‘ticker-tape’ displays as found on websites etc made users perform their primary tasks more slowly than was the case with a simple fading in and out of different pieces of information.

Finally, Bederson and Boltman's work [17] on animation properties shows that for some simple tasks, animating between viewpoints in an abstract information display improves the ability of users to reconstruct that space. In effect, animation allowed users to clarify their mental maps of the information space without any performance penalties, reflecting the specific findings of Ware and Franck for 3D structures. In summary, much work on animation is still ongoing, and like other areas of perception research, there still remains much to learn.

2.3.2 Colour

Work on colour properties has shown the importance of colour in differentiating groups of objects or information, and thus colour selection and perception for visualisation is a topic of importance in IV. For example, Drevermann and Travis's [52] work on colour encoding particle physics events demonstrates that background colour can significantly alter the perception of foreground colours, and hence a neutral background should be chosen where possible.

Colour itself can be regarded as 3-dimensional. These dimensions can be the common red, green, and blue (RGB) values, or hue (what people normally call colour), saturation, and value attributes (HSV), or obtained from other colour spaces calibrated for certain properties. One other such colour space is the LUV space, designed to be a *perceptually uniform* colour space, where the L value encodes perceived luminance and combinations of the U and V variables define chromacity. Within any of these colour spaces, one particular colour will map to one point within the 3D space.

Healey's [82] approach to selecting suitable colours for data visualisation was to separate candidate colours as distantly as possible within a colour space (HSV, LUV, RGB spaces etc), and their work also backed up the common finding that using more than seven colours makes distinguishing between similar hues difficult. Healey also showed that the common categorisations of hue (green, red, green-yellow etc) did not map to equal portions of the LUV colour space, despite that colour space being designed to be perceptually uniform, thus revealing inadequacies in the thinking behind the model specification. Further on this point, Levkowitz and Herman [113] describe perceptually linear colour scales that correct the non-linearity of the common RGB colour scale in human vision, so that the separation of two colours on the linearised scale is proportional to their perceived difference.

MacDonald [115] described other perceptual and physiological advantages and pitfalls of colour and its (mis)uses, and among these are heuristics such as not using blue hues to display fine detail, as the human eye is much less sensitive to blue than it is to green or red hues. Many more findings and discussions on colour properties can be found in the wider literature than can be detailed here, but the main point made by MacDonald is that colour, like other perceptual cues, can be as misleading as it can be revealing without proper consideration of its use and properties.

Healey *et al* [83; 84] also performed experimental studies demonstrating the conflicting effects of several perceptual visual cues operating simultaneously, such as colour, shape, and motion; their findings stating that the use of colour tends to dominate and effectively mask information coded by shape. Such examples of perceptual conflict should be borne in mind when mapping data attributes to visualisation attributes.

The experimental findings provide a proven basis for developing IV systems that co-operate with human visual capabilities, and at the very least they can be used to ensure the wrong visualisations aren’t communicated to the user. However, one of the current criticisms of IV is that its researchers and practitioners don’t take this perception research sufficiently into account. Mackinlay [116] among others claims that what there is of it remains insufficient, stating that understanding and taking advantage of the interaction between perception and cognition will be a major topic for IV in the future.

2.4 Display Techniques

Information may be abstract, but it often has a structure that categorises it, such as a network or a hierarchy. Research on these structures has concentrated on tackling the conflicting issues of size, layout, and legibility on limited screen area. Other information sets have less obvious structures, if any, and require display techniques that accent the important dimensions whilst still retaining an overview of the others. A more formal approach to describing the mappings between information and its on-screen representations is constructed by Card and Mackinlay in “The Structure of the Information Visualization Design Space” [29].

Rather than following Gershon *et al*’s categorisation exactly, this section focuses on the three most relevant information types for the multiple overlapping hierarchy structures: hierarchies, graphs and networks (node-link data), and multi-dimensional information. Therefore the following sub-sections describe a representative sample of seminal and state-of-the-art research into visual representations for these structures.

2.4.1 Hierarchies

Hierarchical information sets are an efficient concept for storing, classifying, and manipulating objects that can be grouped and sub-grouped according to common attributes. As a result, hierarchical information sets are a ubiquitous information structure that crop up in file systems, document classifications, taxonomies, organisational structures and sports league structures to name a few. Subsequently a great deal of effort has been channelled into the visualisation of these information sets in IV.

Hierarchies tend to be strictly *trees* (structures in which an object, or node, has exactly one parent object/node, except the root, which has no parent), though this is not always the case as some more complex structures can be hierarchical too. This is indeed the case with multiple taxonomies, and in Figure 1.1 it can be seen that many of the traced names have different parents in each column, though they are always organised hierarchically. In short, a hierarchy can also contain elements or nodes that have more than one parent. However, consideration shall be given first to representations suitable for hierarchical trees.

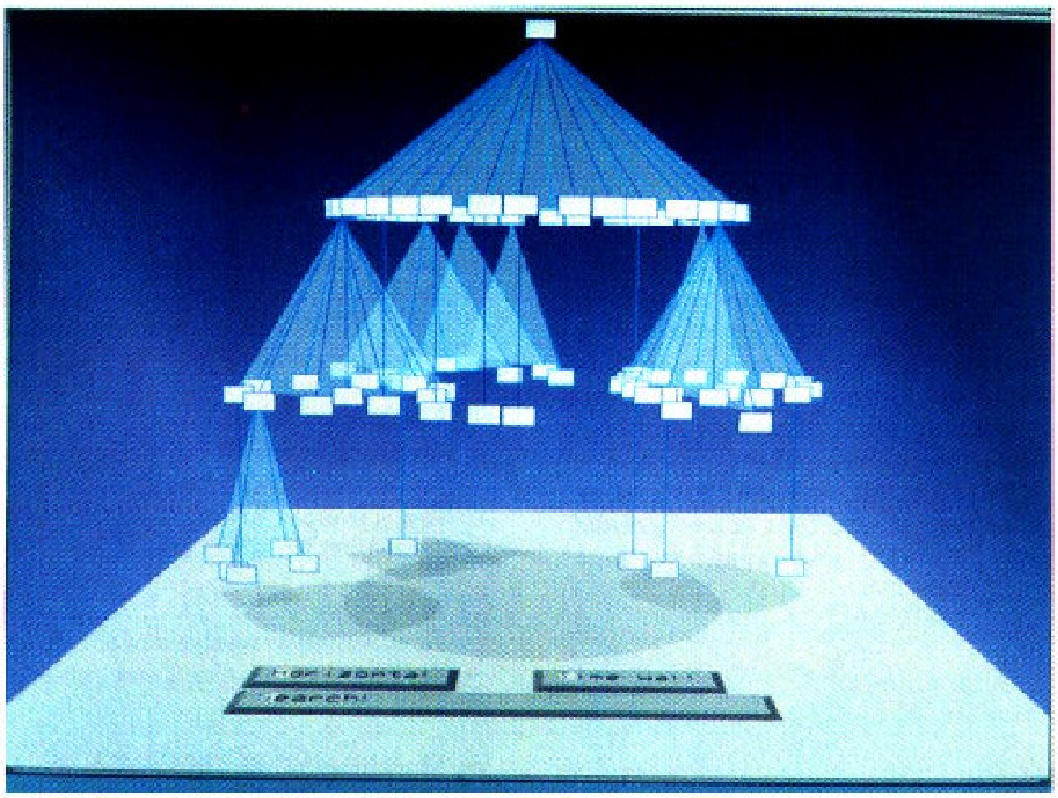
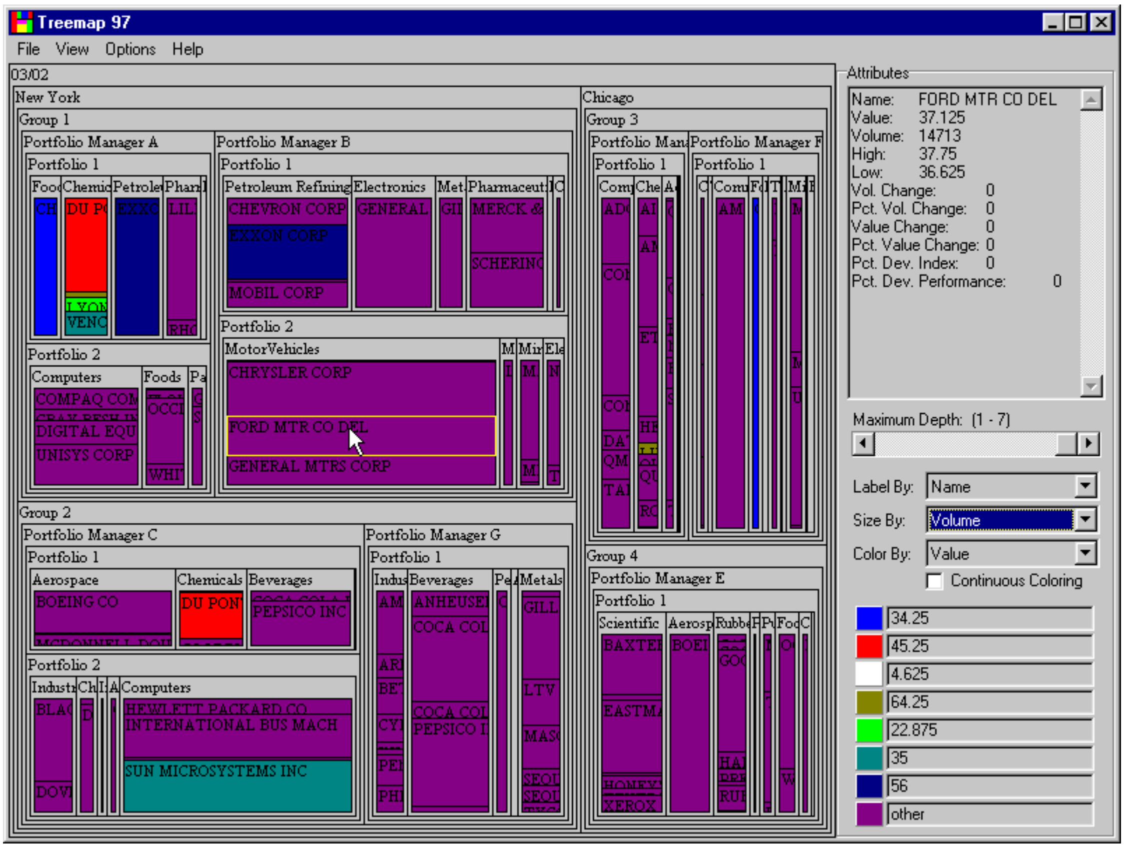


Figure 2.1. Robertson *et al's* Cone Tree

The two classic approaches to drawing tree structures are the node-link and nested box representations. The node-link metaphor generally visualises nodes within the tree structure as small boxes, and graphically connects these with representations reflecting the logical connections, or links, between the nodes. The nested-box metaphor represents trees by drawing 'child' or deeper nodes as smaller boxes within a larger box representing the 'parent' or higher node that contains it. This process is then carried out recursively, dividing the smaller nodes according to their child nodes and so on.

The original IV tree hierarchy visualisation was Cone Trees by Mackinlay *et al* in 1991 [150], developed as part of the Information Visualizer paradigm at Xerox PARC [31]. A tree structure is displayed in three dimensions in an attempt to increase the number of nodes that can be presented on-screen, as shown in Figure 2.1. Groups of links emanating from a particular node to its child node group form a translucent cone, with the child nodes arranged uniformly around the circumference of the cone's base. Selecting any node would bring that node to the front of the view of the Cone Tree in a smooth animated sequence, the use of animation preserving the users' mental model of the visualisation as change takes place. Animation's superiority to the alternative, an abrupt move to the final position without any intermediate views, has been shown [9; 17]. However, Cone Trees suffered from viewing problems due to occlusion caused by the 3D representation, as seen in the screen-shot. Still, its appearance and possibilities were enough to drive a chain of further research into its properties and how they could be improved or extended. Jeong's [94] adaptation of Cone Trees tackled the occlusion problem, and other work studied and began to resolve the problems of enhancing the perceptual cues used in Cone Trees, and advanced filtering and focusing mechanisms, namely Tversky *et al* [167], and later, Carrière and Kazman [33].

As Kaugars [98] notes, it is a hierarchy's very ability to store large amounts of information efficiently that consequently makes them difficult to visualise. Also aware that traditional node-link tree diagrams wasted approximately 50% of the available screen space, Johnson and Shneiderman developed Treemaps [96] as a space-efficient approach to increase the size of trees that could be displayed legibly on-screen.



Name: FORD MTR CO DEL
Value: 37.125
Volume: 14713
High: 37.75
Low: 36.625
Vol. Change: 0
Pct. Vol. Change: 0
Value Change: 0
Pct. Value Change: 0
Pct. Dev. Index: 0
Pct. Dev. Performance: 0

Maximum Depth: (1 - 7)

Label By: Name
Size By: Volume
Color By: Value
☐ Continuous Coloring

🟦 34.25
🟥 45.25
⬜ 4.625
🟫 64.25
🟩 22.875
🟦 35
🟦 56
🟪 other

Figure 2.2. Treemap layout, with colour indicating common attributes. © HCIL.

Treemaps, an example of which is shown in Figure 2.2, uses the nested box representation in which an area on-screen is divided according to the number of top-level nodes in a tree. These divided areas are then sub-divided according to the number of children of each node, and the process continues recursively down to the leaf nodes. The relative size of each sub-area is calculated according to some metric (file directory size in the original application). Individual leaves may be coloured according to another metric (file type in the original application). This approach is extremely space-efficient, utilising all of the space within the original area, but gives visual prominence to the leaf nodes at the expense of the internal structure. Internal nodes can be given borders that help visualise them as ‘peeking’ out from underneath their child nodes, but this obviously reduces the space available for the display of their child nodes.

As was the case with Cone Trees, Treemaps has been successful enough to provoke further investigations into its use despite its restrictions. The use of Treemaps in other data domains has been explored e.g. Asahi *et al’s* analytical decision charts [7]. Various extensions and refinements to the original design, such as incorporating 3D effects to improve perception of structural depth by van Wijk and van de Wetering [172], and improving the aspect ratios of displayed leaves [26], have also been developed. Lately, research has focused on layout algorithms combining stability and pleasing aspect ratios when visualising dynamic data [157] and layout of objects of bounded minimal size [16]. Treemaps has now even spawned its own workshop (http://www.cs.umd.edu/hcil/soh/2001/w2.shtml).

An interesting and closely related visualisation to Treemaps is *”Information Pyramids”* by Andrews [6]. Using a similar nested-box metaphor for displaying structure, it adds a 3D effect that gives a pyramid-like appearance to the visualisation, and combines this with a perspective view that brings closer objects to visual prominence. In the visualisation, the leaves of the tree (in this case, files) are given different representations to distinguish them from the internal nodes (directories) of which they are members. Its advantages are that the important parts, the files, are always on top and visible, but enough of the underlying directories are still visible to obtain a view of the whole structure.

Rekimoto and Green extended the nested box metaphor to three dimensions with the Information Cube [146]. This approach used translucent cubes, nested inside each other like Russian Matryoshka dolls to convey the effect of nodes being contained within other nodes. The cubes were translucent rather than transparent and the user could control the translucency setting to adjust how far into the structure they wished to see, enabling them to hide deeper nodes and also the far sides of the larger cubes to avoid visual clutter.

Lately, hybrid approaches between the node-and-link and nested approaches for displaying trees have evolved, termed “radial space-filling” visualisations. The approach is a space-efficient display where nodes are shaped to take up a high proportion of available screen space, but are still arranged in a familiar top-down (or more correctly, *centre-out*), root and branches layout pattern familiar from the node-link displays. To accomplish this, the trees are displayed in a radial pattern, with the root node at the centre of the circle. Each subsequent level of depth in the tree is assigned to a concentric circle, which is sub-divided into sectors according to the current metric (i.e. file size) and quantity of nodes at this depth. These node areas are then abutted to the appropriate nodes in the previous, inner circle, precluding the need for a link representation. Visualisations adopting this approach include those by Stasko and Zhang [162], shown in Figure 2.3, and by Andrews and Heidegger [5].

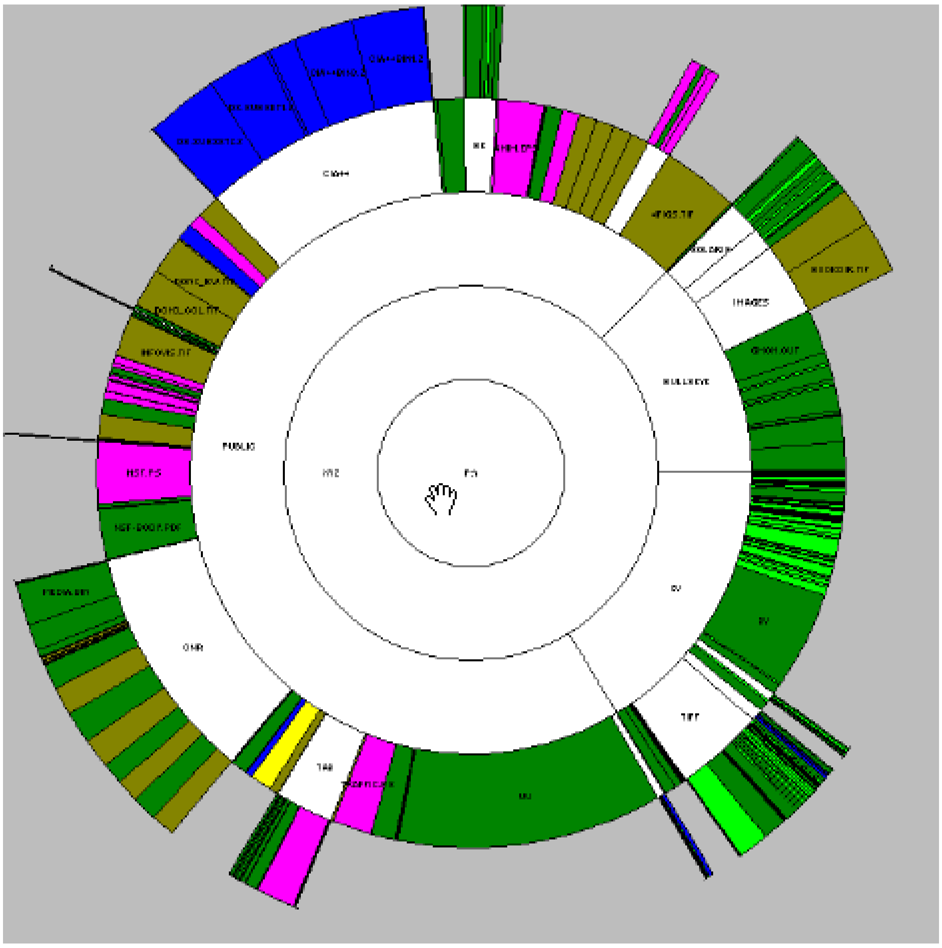


Figure 2.3. Stasko and Zhang's radial space-filling tree visualisation.

Another visualisation existing between the node-link and space-efficient layouts is Beaudoin *et al's* Cheops tree visualiser [14]. This combines a 2D layout with a system of overlapping sub-trees in such a manner that nodes and sub-trees of interest are given visual priority. A parent node and its children are represented as small, neighbouring triangles, which are always the same size regardless of their whereabouts in the hierarchy. Triangles that are representing the children of a particular node will therefore tend to overlap each other. A user-selected sub-tree will appear on top of and partially occlude neighbouring triangles and completely occlude their neighbour's children, though by selecting any neighbour the sub-tree emanating from that node can then be seen. This system allows very large hierarchies to be viewed in an information-dense manner.

All these tree visualisations are not the sole domain of tree data structures. Many visualisations have more complicated structures but reduce them through simplifying techniques to trees, in an effort to take advantage of the visually pleasing tree layout techniques described. Hao *et al* [78] and Munzner [127] both reduce graphs to structures known as 'spanning trees' – tree structures that hold the vertices of a full graph but just a partial edge set. Similarly, Mukherjea *et al* [125] reduce a network to a particular tree structure by omitting certain links. This is a recurring theme throughout the IV literature, the notion of taking a complicated structure and reducing it to one or more simpler sub-structures for visualisation.

2.4.2 Node and Link structures (Graphs & Networks)

The second class of structure that accommodates a large variety of abstract information sets is the graph, or network, commonly known as a node-link structure where, as with trees, nodes represent individual items of information, and links represent the relations between them. Graphs can have structures anywhere along an increasing scale of complexity from trees, which have their own specialised visualisation approaches as previously described, through restricted graphs such as DAGs (Directed Acyclic Graphs), up to multigraphs and full, general graphs. Herman, Melançon and Marshall [87] describe in detail the common graph structures and their associated visualisations in their recent survey paper.

It has been found that forming useful visualisations of the general types of graph structures is notoriously difficult, especially compared to the well-understood visualisations that exist for trees and simple hierarchies. These issues arise primarily from a graph not ordinarily being a simple hierarchy, and thus not lending itself in its natural state to a hierarchical layout. It has been shown that visual edge crossings, unavoidable in the visualisation of non-planar graphs, make the comprehension of a graph structure more difficult than is the case with non-crossing structures such as trees [143].

Because of the difficulties associated with visualising general graphs, a number of approaches begin by attempting to reduce the graph to a simpler structure. The aim is to transform the graph either logically and/or visually to a type of tree, as described in the Hao and Munzner examples at the end of the last section, or to a restricted graph that has properties that lend themselves to more understandable visual layouts.

Visualisations for restricted graphs above the complexity level of trees include Bartram *et al's* Ztree [12] and Melançon and Herman's DAG drawing system [121]. The Ztree system displays a strongly tree-like graph, with the tree portion drawn in a nested TreeMap style, and the other relationships displayed as cross-links between the appropriate nodes or sets of nodes in the hierarchical structure. As such, the *static* display policy can be described as an augmentation of a purely tree-like layout, with further properties revealed by the interaction techniques described later. Melançon and Herman's visualisation takes advantage of the fact that a DAG structure can be broken into layers, where a layer is a set of nodes in a graph that have no direct links between themselves. Each layer can thus be displayed in a visually distinct manner, with the directed links always being drawn outwards to other layers. For a more tightly restricted class of DAGs the layers can be displayed in a hierarchical top-down manner, such that the links from a higher layer to a lower layer always have the same directionality (i.e. directed *from* the higher layer *to* the lower layer).

The other main display technique in IV for drawing a full graph is to use a *self-organising system* that positions the nodes and links of the whole graph according to a set of rules. These are set up with the aim of producing aesthetically pleasing layouts, but obviously the final judgement of this is down to the user, not the algorithm. As such, pure research into full, un-simplified graph visualisations is concerned primarily with the speed and aesthetics of the graph layout: Harel and Koren's recent work [80] focuses on both these issues. The latter consideration, which they term 'beautification', is explored in an effort to make such complex visualisations more understandable and easier to assimilate, and focuses on issues such as reducing visual edge-crossing and promoting visual symmetry when the logical graph itself has symmetrical properties.

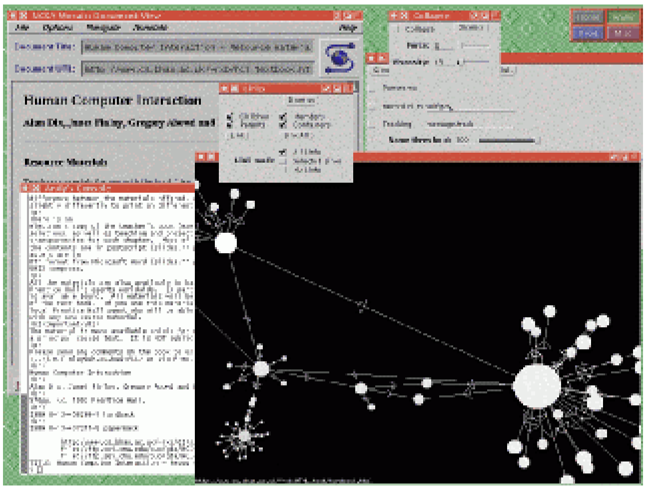


Figure 2.4. The Hyperspace web viewer.

One example of a graph visualisation applied to a specific domain is Hyperspace [176], displayed in Figure 2.4, which visualises the hypermedia structure of the World Wide Web (WWW) as a graph. Individual pages form spherical nodes and links between pages form lines in the visual representation. Hyperspace visualises the graph by letting the user select an area or keyword of interest, whereupon related pages concerning this topic move closer together, and dissimilar pages repel. In the end a graph with clusters of related pages is formed and displayed.

Hyperspace's particular type of self-organising structure, based on a metaphor known as the spring-mass model (described in Section 2.4.2.1), occurs in several systems such as Hyperspace's successor Narcissus [85], and others [51; 177]. The effect of this clustering is analogous to the concept of chunking in drop-down menus. Similar items are grouped together, and the user recognises them as sharing common attributes due to their visual proximity.

2.4.2.1 Force-Directed Layout

The spring-mass model is a metaphor for an algorithm that calculates the positions of a group of entities according to the links between them, such as that described by Fruchterman and Reingold [62]. The unrefined algorithm is as follows: nodes may be given individual 'weights' or 'masses' and the links between them given 'strength' or 'stiffness'. Over a number of iterations the forces modelled by the springs are calculated and the nodes are moved in a bid to minimise the forces felt, mimicking the action of a physical system of springs and masses. For example, two distant nodes connected by a strong link would be moved closer towards an equilibrium position; too close and the spring will 'push' instead of 'pull'. Also, in practice, unconnected nodes are subjected to small repulsive forces between each other. This spreads out the nodes for a more pleasing layout, and helps to avoid overlapping node representations. The model is used for graph data, or multi-dimensional data when suitable links are calculated between the data items.

Various refinements have been made to the basic technique, including those by Kumar and Fowler [108] and Gansner [68], who respectively extended the metaphor to 3D and tackled the problem of node occlusion due to unfavourable positioning. Kamada and Kawai's algorithm [97] combined the effects of attractive and repulsive forces in one formula; their idea was to have nodes that were separated by a shortest path of N links distinguished by a distance proportional to a function of N in the display. This is achieved by calculating forces between all possible pairs of nodes, each individual force proportional to their particular N value.

Another type of force-directed model, based on a process known as *simulated annealing* [48], is more costly in algorithmic terms, but allows moves that go against the local force gradient. In effect, this allows layouts to escape from positions in which the classic spring layout would be trapped, and head towards layouts that are closer to the possible minimum sum of forces felt throughout the network.

Some graph data sets are just too large to display as node and link visualisations, even with clustering methods that allow a level-of-detail (LOD) approach to their viewing, such as Eades and Huang's [53] clustered graph visualisation. Recent work by Abello, with Krishnan [2], and Korn [1] respectively, tackles the problem of visualising massive graph structures by breaking down graphs hierarchically through the aggregation of suitable edges. The derived graph is then viewed as a surface, an N by N matrix where the derived graph nodes are compared against each other, as a 2D scatterplot or a 3D terrain. Edge collections between a node on the X-axis and a node on the Y-axis are signified by a suitable variable mark at the point (X, Y) in the plot. This technique can be used on a graph without any pre-processing but is generally disregarded as, unless a graph approaches full edge density, the majority of spaces in the matrix will be empty, indicating that no edges exist between most of the possible pairings of nodes. However in this approach the points in the derived matrix will each represent a set of real or potential edges further down the hierarchy that models the original graph, and therefore are unlikely to be empty.

2.4.3 Multi-Dimensional Information

In IV the main thrust of research into non-tree or graph information is towards the visualisation of multi-dimensional information, i.e. an information set whose objects can have many shared attributes or dimensions, of which some may be orthogonal, and others strongly related. One of the main difficulties with regard to multi-dimensional information is the mapping of these many abstract dimensions to the two spatial dimensions displayable on a computer monitor.

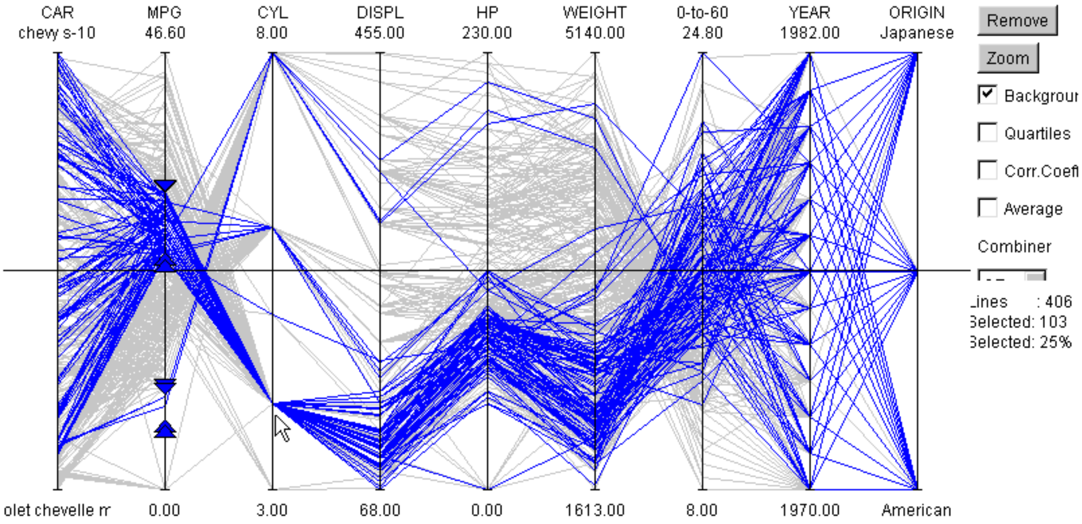
3D projections can help, and VR techniques can increase the depth perception of 3D visualisations, but even this only gives us one extra spatial dimension. It also introduces its own set of problems such as occlusion and effective depth cueing, though these issues can be tackled [178; 90]. Also recent discussions in the IV community have led some to consider that 3D visualisations are not that helpful for understanding non-spatial information, the drawbacks of the metaphor outweighing the benefits.

A simple, straightforward answer to the problem appeared to be the mapping of dimensions to non-spatial cues such as colour, brightness, transparency, and shape, an early example of which was proposed by Benedikt [19]. Subsequently, a number of techniques used Benedikt's idea of extrinsic (position and orientation of object) and intrinsic (non-spatial object properties) dimensions to display high-dimensional information sets, examples of which are the Q-PIT information terrain [44] and the music visualiser of Graves et al [76]. An initial problem with this approach lies in deciding which dimensions should be assigned to the more perceptually potent extrinsic dimensions. This can only be decided by analysing the information in conjunction with the user's tasks and deciding what aspects of the information they are most likely to be searching for. Even within the intrinsic qualities such as colour and shape there is a perceptual ordering, established by the experiments on human visual perception described earlier. One solution to this is to let the users themselves dynamically assign properties to dimensions according to the task at hand.

A further problem is that abstract dimensions are not easily made to fit on the numeric and ordinal scales that Benediktine dimensions use, extrinsic and intrinsic. The fallback position is to use one of the arbitrary scales that Benedikt proposes, which are alphabetical, geographical, and chronological. Dimensions should be able to map to at least one of these orderings, though doing so might appear to make no difference to comprehending the information.

Another approach pioneered by Feiner and Beshers [58] consists of nesting co-ordinate systems within the points of other co-ordinate systems, and thus viewing only a subset of the actual dimensions and/or objects present in the information set. A further approach is to divide the screen into sub-areas in which pairs of dimensions can be compared against each other. This is the multiple scatterplot approach pioneered by Becker and Cleveland [15], and brought about the notion of linking, one of the interaction techniques described in Chapter 2.5.3.

Inselberg and Dimsdale [91] introduced *Parallel Coordinates*, a system whereby N dimensions are allocated one-to-one to an equal number of parallel axes on-screen. An object in an information set would then be mapped as a series of points, one per axis, with the position of each point on the axis being dependent on their value in the associated dimension. The points would then be joined together by line segments from one axis to its immediate neighbour, forming a "poly-line" across the set of axes. This process would be repeated for each object in the information set.



ines : 406
3elected: 103
3elected: 25%

Figure 2.5. Parallel co-ordinates as visualised by Siirtola's interactive applet [158].

This method allows similar objects to be seen as having similar lines, and in later applications of the technique, axes could be moved about to enable the user to order the dimensions as they see fit. However, one problem is that if two objects share the same value in a particular dimension, they will share the same point on the corresponding axis, and as such their respective poly-lines will appear to merge and then separate again. Without additional cues such as colour, it is impossible to determine which line is which after the merge and separation effect. Such a situation can be seen clearly, or rather, unclearly, in the third axis from the left in Figure 2.5, labelled ‘CYL’.

With the previous techniques, there is still a limit on the number of dimensions that can be represented through dimensional nesting, parallel co-ordinates, or simple one-to-one mappings of colour, hue, shape, size, positioning etc, and some high-dimensional data sets have upwards of hundreds of discrete dimensions. Thus, many approaches for visualising high-dimensional information sets such as document corpora rely on comparing dimensional vectors between objects and using adaptations of the self-organising techniques such as the spring-mass algorithm to settle the objects into a comfortable final configuration. This configuration is then converted to 2D or 3D co-ordinates and can be displayed in a number of ways: as a basic-scatterplot; as a data landscape if 3D (an enveloping surface is drawn over the points), or as a node-link visualisation if there are significant correlations between documents. From such a visualisation the clusters of similar objects can be viewed, inspected and explored.

One significant early system that visualises the correlations of text documents as a data landscape using this methodology was Chalmers and Chitson’s Bead project [37], and similarly Sprenger *et al’s* [160] system visualised patterns in a set of multi-dimensional economic data. Incidentally, whilst systems such as Bead only displayed the final outcome of the reduction algorithm, Bentley and Ward’s work [20] focused on the animation of the Multidimensional Scaling (MDS) process. Chen’s [38] visualisation of the WWW also employs data concerning web page content similarity and user browsing patterns, along with structural data to generate a final visualisation. As such the network’s nodes and links are formed partly from explicit links between web-pages and implicit links that are generated due to analysis of these extra two factors. These are then combined into an overall network using specialised analysis techniques. This network, as with Hyperspace, is then visualised using a self-organising approach.

However, with the reduction in dimensionality, geometric closeness on screen will not always imply the same degree of closeness in the higher dimensionality. A useful analogy can be made by the observation of the constellations that make up the typical night sky. These are patterns or clusters formed from stars that appear to be grouped together when projected onto the naked eye’s essentially 2D view of the sky. In fact these can be, and usually are, composed of stars separated by much greater distances than is apparently the case, as they reside in a 3D not a 2D universe. Stars appearing to be concentrated in one area of the sky may be greatly dispersed along the line of sight from earth. (Technically speaking, the human eye can only resolve the angular components of a star's positioning from earth, and not the radial component.)

In summary, multi-dimensional information is currently the focus of a great deal of research in the Information Visualisation community, much of it driven by the recent proliferation of digital libraries and the problems of searching and browsing the documents they contain.

2.5 Interaction Techniques

Gershon, Card and Eick's third sub-topic was "interaction techniques", in which they classified the general interaction styles that have been used with visualisations. The aims of these techniques are to simplify and aid the user's tasks of finding hitherto unknown patterns or locating specific details in the displayed information. Even though these techniques may be executed differently across a range of data sets and display techniques, many share similar underlying principles. As such, they are grouped by Gershon, Card and Eick [69] into three general categories termed *focusing*, *filtering*, and *linking*.

2.5.1 Focusing Techniques

Focusing techniques are concerned with the variable magnification of graphical displays. Their utility is in giving greater prominence to a certain area(s) of the visualisation space, and hence greater prominence to whatever is displayed there. They allow navigation and overview of large information sets and simultaneously allow close detailed inspection of specific pieces of information. Focusing techniques use as a general metaphor the effects observed when looking through optics such as fisheye lenses or magnifying glasses. Hence the effects produced may take the form of a gradual magnification effect centred on a focal point with the numerous lens-style viewers, or a uniform increase in magnification termed zooming, or a combination of both techniques in which distinct areas have their own discrete level of magnification. The non-zooming metaphors are also termed *distortion techniques*, though Keahey and Robertson [100] observe that distortion itself is an artefact of the boundary between different levels of magnification, either a sharp delineation between two discrete areas of magnification or forming a gradual and global distortion in the lens viewers.

The fisheye or lens viewers (also known as focus+context viewers), as defined by Leung and Apperley [112], increase the size of the information at the focal point of a screen to the detriment of information which is visualised further away and reduced in scale but still visible. Sarkar and Brown's paper is a specific example of a general lens distortion technique [152], whilst Carpendale [32] pushed the general lens metaphor to three dimensions.

A class of lens viewers termed *hyperbolic lenses* takes advantage of a non-Euclidean geometry with unintuitive properties; one being that infinity in hyperbolic space can be projected back to a finite bounded point in ‘normal’ space. As such, all points in a structure mapped to a hyperbolic space can be projected back into a bounded area of normal space with a visually pleasing fisheye effect, with the object at the origin in the hyperbolic space being displayed at the origin in normal space. The equations involved are non-trivial, and in practice many objects are reduced to sub-pixel size and not displayed. Munzner [127], and Lamping, Rao and Pirolli [110] have applied hyperbolic lenses to network and tree layouts respectively, the latter’s work culminating in the production of a commercial visualisation product, shown in Figure 2.6. Munzner’s work also extended the hyperbolic distortion to three dimensions from the two dimensions that Lamping and Rao’s initial technique utilised.

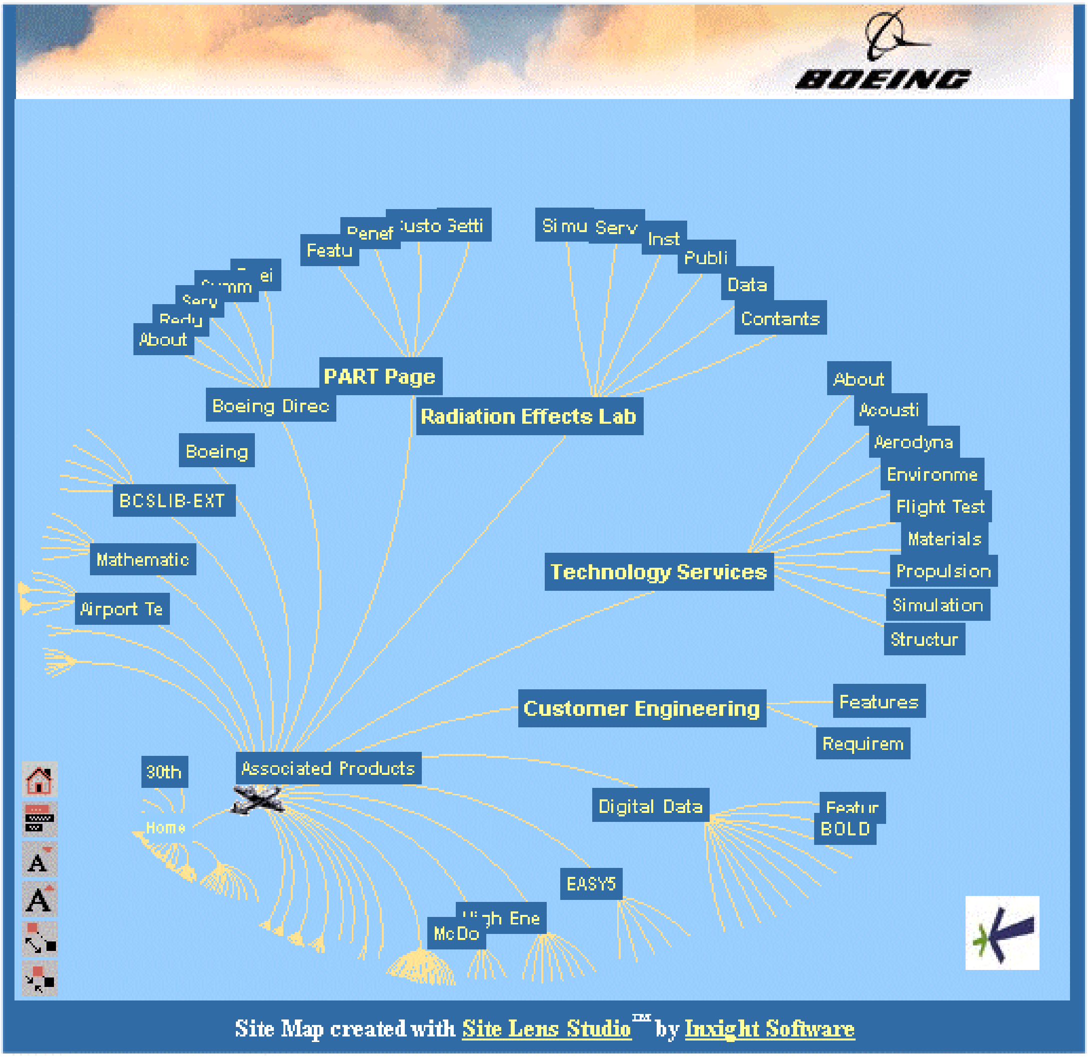


Figure 2.6. Inxight’s Hyperbolic Tree web browser

Leung and Apperley described how the lens technique could be pushed further to develop polyfocal (multiple focii) viewers, where users may wish to see detailed information associated with a number of spatially distinct objects rather than just a single object or focus. Robert and

Lecolinet [147], and Osawa [137], have developed two recent examples of visualisations that depend on and take advantage of multiple focal points.

The second major class of focusing techniques are the zoom methods; the difference being that in a zoom the entire window is always at the same level of magnification. The effect is still to focus on a particular piece or subset of information, but to lose a lot of the periphery information altogether. Separate overview and detail windows can be linked together which partially solves this problem, but it is less elegant than the lens techniques, and requires the user to cognitively integrate the information in the two windows. Bederson *et al* [18] took the zooming metaphor further by introducing a *semantic zoom* in their Pad++ interface. Using this approach, not only did objects shrink or expand visually, but the level of detail they showed changed correspondingly as well.

Lens techniques are considered superior to zooming techniques, as demonstrated in user experiments by Schaffer *et al* [153] that compared both techniques on a large graph visualisation. The drawback is that the variable magnification levels of the lens techniques can produce visual distortions that may be disorientating for some users compared to the more natural feel of the zooming metaphors. Incidentally, it can also be claimed that one advantage of 3D visualisations is that they provide a method of focusing that is natural to the human perceptual system i.e. closer objects appear larger, and distant objects appear smaller.

These lens-based focus + context techniques consist primarily of either spatial transformations on continuous co-ordinate spaces, such as those described by Keahey and Robertson [100], or relative layouts such as the Continuous Zoom by Bartram *et al* [11], where an object's absolute co-ordinates depend on their neighbours positioning, rather than any global transformation function. Specific examples involving hierarchical structures are, for the former, Hyperbolic Trees [110], and for the latter, TreeMaps [96]. These relative layout techniques depend on functions called Degrees Of Interest (DOI) associated with objects in the visualised structure e.g. for TreeMaps the DOI is traditionally the relative size of files within a visualised directory structure.

Furnas [65] described a DOI as a function which "assigns to each point in the structure a number telling how interested the user is in seeing that point, given the current task". Furnas' approach was then to compare these values against a threshold value, which would decide whether an object was shown or not, resulting in an all-or-none decision. However, as DOI functions can produce a range of values, they can be used to produce a range of outcomes. In the case of a DOI-based focus + context technique, the outcomes will be a range of sizes for the visualised objects. In effect there are two steps; the first is deciding how to calculate DOI *values*, and the second being how to use those DOIs to calculate object *sizes* within the display.

To unify the various continuous focal techniques, Keahey and Robinson [100] presented the idea of non-linear magnification fields as an effective and efficient method for describing and computing a multiple-focus, and indeed any single-focus or zoom, visualisation effect. Their method is based on deriving magnification information from a transformation description of a space or surface and vice versa, and stated that distortion is in turn the gradient or rate of change of a magnification field. They then make the observation that magnification and not distortion is the useful property of a focus and context view. In a further attempt to formalise and clarify the thinking behind focus+context visualisations, Björk *et al* [21] contend that such visualisations can be thought of as second-order visualisations, formed from the original first-order undistorted display of the visualised structure.

The focus metaphor can also be used at discrete levels as well as generating the continuous range effects of the lens and zoom-based techniques. One such effect is *elision*, where groups or clusters of information can be either hidden or represented as one visual representation, to give room and prominence to other information. Parker *et al* [138] used this technique amongst others on a large graph visualisation, to keep the number of displayed nodes at a comprehensible level. However, the elision technique can also be used with some filtering techniques, and as such the boundaries between focusing and *filtering* become blurred with such methods. Therefore, it is filtering techniques that are described next.

2.5.2 Filtering Techniques

Filtering techniques are used when the user wishes to home in on information that has common attributes or values. The query mechanisms underlying the filtering are outside the scope of this thesis; the filters referred to are visual filters on the displayed information, where a set of conditions controlled by the user affects the visualisation in some way. The usual effect is to highlight information that matches the desired conditions, or removal from the visualisation of information that doesn't correspond.

Fishkin [60], Eick [54], and Ahlberg [3] describe the user setting filter conditions on unstructured information, essentially multi-dimensional information. These filters are usually defined as ranges upon a particular set of dimensions, such that the filter accepts objects with values within the given ranges and others are rejected. The use of the filters results in the removal of the unwanted information from the visualisation, and hence the visual promotion of the information the user is interested in. Colby and Scholl [43] showed that a similar effect of visual promotion could be achieved by using transparency and blur effects on information that did not match user requirements.

Filtering can also be used on structured information sets, such as in the filtering of a tree visualisation by Kumar et al [109]. Here, the user sets filter conditions and the effect is to remove certain sub-trees instead of unstructured groups of information. What remain are the parts of the tree, and thus the leaves and paths to the leaves, which the user has declared an interest in.

Filtering techniques are similar to focus-based methods in that they aim to convey more relevant information to the user; i.e. they can be said to ‘focus’ a user’s attention. But they affect the display attributes of the visualised objects directly rather than the space in which the objects are to be visualised, as is the case with a focal lens or zoom technique. Also, filters can be applied to other perceptual cues, such as colour, as well as size, and can be more binary in their outlook i.e. items can be either displayed or not, sometimes with no smooth transition between the states.

The confusion in interpreting the terminology can be clearly seen with Furnas’s seminal paper on fisheye views [65], which though often cited as a fisheye viewer, is better described as a system that applies a filter based on a fisheye metaphor to information around a particular point. A recent hierarchy visualisation technique by Kaugars [98], the OTree, that combines tree pruning with multiple focal points further blurs the distinction. If a node is elided has it been filtered or focused down to a zero size? Though if the technique is effective, does it matter whether its approach is classified distinctly as focus or filter? Noik [134] for instance, is quite happy to describe such techniques as ‘filtering-distorting hybrids’. The fact that no general framework or methodology (to my knowledge) for IV filtering methods has been attempted indicates, to the writer at least, its standing as the ‘Cinderella’ of the three interaction techniques listed in this chapter.

2.5.3 Linking Techniques

The third main interaction technique is linking, whereby an action carried out on one particular view of an information set will result in that action’s consequences being mirrored in other views that are present. Early work by Becker and Cleveland [15] on brushing (temporary highlighting via a pointer) and linking scatterplots helped form a basis for the technique’s extension into IV displays, their work itself being one solution to the problem of displaying multi-dimensional information. An example of linked scatterplots is given in Figure 2.7 from Tweedie’s Interactive Visualisation Artifacts (IVA’s) [168]. The selection of the bounding yellow box (acting as a ‘brush’) in one of the small squares, each of which represents a different view of the same information, will update the similar yellow boxes in the other squares. As with the other interaction techniques, the update must be rapid for it to be useful.

The linking technique can also be used to mirror user choices between different types of visualisation. An adaptation of the Bead system by Brodbeck *et al* [24] updates selections made with a set of parallel co-ordinates on a scatter-plot of the same information that is shown alongside. The reciprocal action of selecting items on the scatter-plot, and having the choices shown in the parallel co-ordinates, is also available. Koschat and Swayne [104] describe a system where scatterplots and a displayed spanning tree of the same information can be brushed bi-directionally, and work in Fua *et al* [64] also shows the linking paradigm applied to two different visualisations of the same hierarchical information set.

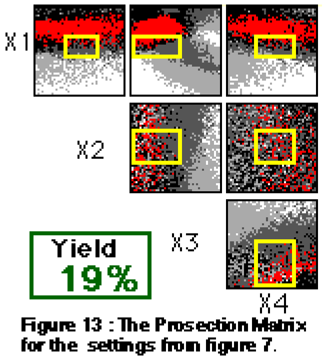


Figure 2.7. Example of Linking in Tweedie's IVA's.

Deciding whether a visualisation will benefit from having a set of linked views rather than one overall integrated visualisation is the subject of various research efforts. Roberts [148] describes a system that generates simultaneous multiple views of *scientific* data sets, and notes the problem that while an information-dense integrated view can help for some data sets, for others it may simply obscure vital information which would be visible in a split, multiple view.

Baldonado *et al* [8] describe guidelines for the development of multiple view systems, revolving around issues including *complementarity* (when multiple views combine to bring out correlations), *decomposition* (partitioning complex data into views so each has a subset of data or data attributes), and *attention management* (using perceptual cues to ensure the user is focused on the right view at the right time). They also discuss what type of data sets and tasks are likely to benefit from multiple views, and the tradeoffs involved in using multiple views instead of a single view.

North and Shneiderman [135] offer a taxonomy of possible linking techniques between two views: selection of items in one view resulting in the same items being selected in another view; navigation of a view being mirrored in the navigation of another view; and selection of an item linked to and affecting the navigation of another view. All these tasks can be on views sharing the same information set, or featuring different information sets. This taxonomy, rather than assisting designers in assessing whether multiple views would be a useful approach for a given information set and task, is focused towards categorising the potential possibilities in preparation for the semi-automatic generation of linked views, an application of which North describes in later research [136].

In short, linking techniques provide good reasons for splitting complex information sets and their resulting visualisations into more individually comprehensible, multiple displays. Co-ordinating data and perceptual cues between these many views will then still allow a sophisticated picture of the information at hand to be constructed. Furthermore, the user need only correlate as many views as is necessary to discover the information they require.

2.5.4 Combining Interaction Styles

As described in Chapter 2.5.1, a considerable body of research in Information Visualisation has addressed various single and multiple focus + context techniques. These techniques resulted from a need to show in detail specific information within a larger structure, but at the same time show this information’s relationships with the whole of the structure. Integrated focus + context methods combined these two needs seamlessly in one visualisation.

Likewise, the described linking techniques were developed to enable users to easily ascertain correlation’s in a multiple view visualisation. Objects highlighted or manipulated in one view are similarly affected where they occur in other views.

Although visualisations have utilised various combinations of linking, brushing and filtering such as is found in the Bead system [24] amongst others, previous visualisations have not utilised combinations of linking and focusing, specifically in the case of focus + context techniques. The focusing and linking technique of Buja *et al* [27] conveys the now standard linking effect on selected data between overview and detail windows, but doesn’t convey between windows the variable focus and context effects that can be attached to objects.

Sometimes the idea of linking areas of focus + context does not seem useful. For example, take Becker and Cleveland’s or Tweedie’s linked scatterplots [15; 24] for multidimensional information, where points or nodes are defined by sets of values which are then mapped to a series of absolute 2D co-ordinate spaces. Focusing effects in one scatterplot could be achieved by expanding areas around the points concerned, though a scaling function would be required along the co-ordinates. However, then linking such areas of focus to other scatterplots using data points as references could create conflicts. If for example two points in one scatterplot, one a focal point and the other not, shared the same position in another scatterplot, should the space contract or expand at such a co-ordinate? Brushing or filtering would seem to be a more practical method for these structures. Trees and graph structures, however, can also be visualised in wholly relative layouts as the information they represent is primarily relative. Nodes are defined by their relative logical and thus relative spatial relationships to other nodes, rather than to an absolute external co-ordinate system. Different nodes will not share the same co-ordinates in such visualisations, so the conflict of how to link focal values from one visualisation to another is removed. As long as a focal value is attached to a given node, it should be viable to propagate this value across all the views it is present in.

This is essentially the continuous (space-warping) versus relative (DOI-based) focus + context dichotomy described in Chapter 2.5.1, and as long as a focus + context method respects and preserves these relative relationships, linking focal effects should not cause a problem in interpretation.

2.6 Visualisation Architectures

In the 1970s and 80s the advent of Graphical User Interfaces (GUIs) prompted research into the development of UI architectures, both conceptual and implementational. The pressing problem was the great amount of extra resources needed to produce GUIs compared to the traditional Command Line Interfaces (CLIs) as detailed by Kratzner [106]. Consequently, UI architectures were constructed in an effort to separate functional from presentational concerns to facilitate modularity and portability, and hence promote re-use, change, distribution and development of interface components, and ultimately to save time and costs.

Various models and architectures for the conversion of raw data and logical models to on-screen representations have been proposed and developed for graphically-intensive interfaces. For example, the AVS rendering system developed by Upson *et al* [169] is a pipeline model specifically designed for large-scale scientific visualisation purposes, in which a linear series of manipulations and transformations transforms raw data into a final representation.. Similarly, Kazman and Carrière’s more recent generic hierarchy visualiser [99] uses an adaptation of the common Arch-Seeheim user interface architectural model to compartmentalise and isolate different stages in the rendering of a tree from the initial data. However, such pipeline-style models have been shown to develop bottlenecks, and consequently a new generation of object-oriented UI architectures came into being, such as MVC (Model-View-Control) [105] and PAC (Presentation-Abstraction-Control) [45], which separated functional, output, and input concerns in a modular, object-oriented fashion within an application.

An advantage of the MVC approach, though common to many other UI architectures, is its ability to separate Model data (logical model and raw data) from interface concerns. Subsequently the View (output) and Control (input) components can be changed without having to alter the Model itself. In IV, this can be used to generate many different visualisations from the one model, saving development time. MVC can do even better however as, if multiple visualisation components are attached to the one Model, operations affecting the model caused by one visualisation will be reflected in the other views, effectively constructing an automatic linking effect that has proved very powerful in previous interactive visualisations. The Graph

Visualization Framework (GVF) of Marshall *et al* [119] acknowledges this point, allowing parallel displays of the same graph with different layout algorithms. North's thesis [136] focused on the power of an architecture that enabled different visualisation outputs to co-operate over and reflect facets of the same data set.

A more common version of MVC [61] is perhaps more accurately described using the acronyms MV or M(VC), where the input and output concerns (Control and View) are quite tightly coupled, and in effect merge into one component. Pure MVC modelling demands that associated View and Control components are also interchangeable with no effect on each other. However, this rarely occurs because separating input and output concerns for some UI objects is either implausible or impossible. IV environments can only exacerbate this situation, where in many display and interaction techniques the view or output *is* the visual input object. In Treemaps, for example, the rendered tree and its nodes are the inputs for a user wishing to manipulate the visualisation. When a node is selected the knowledge that the model is a tree, and of its exact structure, is not enough. The Control needs to know the exact co-ordinates of the representations in the View to know which node was interacted with. This could be done by querying the View by proxy through the shared Model, and then searching through the entire set of node co-ordinates for the closest match or intersection. In the general case, this would work for most visualisations. However, if the Control has the knowledge that the View is laid out in a recursive, nested manner, as is the case with Treemaps, the Control can quickly search through top-level node co-ordinates for a match. Once found, the Control can search that node's children for a tighter co-ordinate match, and so on recursively until the leaf nodes are reached. This depth-first targeting is much faster and more efficient than the naïve, general search method, but requires the Control and View to be tightly entwined. However, the Control component could now not be used with another hierarchy visualisation output, such as a Cone Tree, as the layout strategy for that visualisation is different.

2.7 Conclusion

To summarise, all these issues should be considered when devising useful visualisations. One of the challenges of IV is to find the best way of representing the data, as each type of data has its own particular characteristics, so that although one technique may work well for one type of data it may not be applicable to another. A good visualisation will address and utilise the perceptual issues, display techniques, and interaction techniques of IV to achieve its goal of increasing user understanding of a given information set.

The particular problem faced in this thesis will involve the visualisation of hierarchical structures, which has been a focus of research within information visualisation since Xerox PARC's work on Cone Trees. As discussed, research has since continued along the lines of increasing the information density of visualised hierarchies, focusing and filtering techniques, and improved 2D and 3D layout algorithms. However, to current knowledge, there are currently no visualisation methods that allow for the tracking of *structural change* within hierarchies, a phenomenon that frequently occurs with the reclassification of an existing hierarchy. Alternatively, the situation could be described as tracking structural differences across a set of hierarchies holding the same node information.

One discipline where such information sets occur and the ability to track information from one classification to another is required is taxonomy, the study of scientific classification. A more detailed explanation of the field and its particular problems follow.

3 Multiple Classification Hierarchies in Taxonomy

To proceed with the visualisation of multiple taxonomies, an understanding of the taxonomy domain must be gained, the practitioners and the tasks they perform. Therefore, this chapter describes the methodologies behind the creation of taxonomies and how they proliferate over the same, single data set. This is followed by a listing of the basic tasks that taxonomists would wish to perform to compare and contrast multiple classifications. Finally, all this information is combined to produce a logical structure that can support the information finding tasks described.

3.1 Taxonomy

Taxonomy, in general, is the science of classification.

Botanical, or plant, taxonomy is specifically concerned with the study and subsequent naming and categorisation of plants to generate classifications depicting their presumed relationships.

Individually, these classifications are hierarchical structures in which *specimens* (actual physical plant samples) are grouped into categories termed *taxa* (singular: *taxon*) according to a given criterion e.g. morphological similarities. These *taxa* are then recursively merged through a bottom-up methodology into higher level taxa using the same criteria. Eventually a high enough level of aggregation is achieved such that all plants concerned are grouped into one *taxon.*

In the common *Linnean* methodology, *taxa* are assigned to pertinent ranks such as species, genus, or family, which specify the level of a taxon in the hierarchical classification system shown in Table 3.1. Taxonomists may use different combinations of these existing ranks, or in certain cases, develop new ranks, in their taxonomies. Typical examples of taxonomic work cover only a few of the many ranks described in taxonomic research. The reason for this is not a matter of lacking an appropriate visualisation, but because an individual taxonomist can only physically handle and describe perhaps a few thousand specimens or genera. More than this overwhelms traditional working practices, so cut-offs in the level of detail and scope are made, generally using one of the compulsory ranks such as genus or family as an upper or lower bound.

The classification is then published for other taxonomists to use and is now considered a valid classification.

- **Regnum**
- Subregnum
- **Divisio or Phylum**
- Subdivisio Or Subphylum
- **Classis**
- Subclassis
- **Ordo**
- Subordo
- **Familia**
- Subfamilia
- Tribus
- Subtribus
- **Genus**
- Subgenus
- Sectio
- Subsectio
- Series
- Subseries
- **Species**
- Subspecies
- Varietas
- Subvarietas
- Forma
- Subforma

Specimens (actual plants)

Table 3.1. Linnean taxonomic ranks. Items in bold type indicate compulsory ranks, to be used whenever the scope of work encompasses such a rank.

Jeffrey [93, p.35] states that the research work of a plant taxonomist can be outlined as so:

> *"Reduced to its basic terms, it may be defined as the organization of information about plants and the making of this information readily available to others. It is not directly concerned with the production of information about plants, which is the task of all other branches of botany, though in practice taxonomists are usually obliged to produce many of the data they need themselves."*

Whilst botanical taxonomy may not produce direct information about plants, it certainly produces meta-information. Every taxonomy produced is an organisational structure, defining for a given criterion the similarities and relations existing between a set of plant species or higher taxa such as genera.

Having outlined how a single taxonomic hierarchy may be produced, the reasons why multiple overlapping hierarchies occur must be described. Firstly, a single taxonomy is produced according to a single criterion, such as leaf shape. Obviously there will be more than one way to classify the same set of plants, especially with the rise in botany of modern techniques such as DNA sequencing and other biochemical systematics. Classifying by a different criterion will inevitably organise the same set of objects differently.

Secondly, new specimens of plants are discovered continually, and for these to be classified a new taxonomy or a revision of a historical taxonomy must be undertaken to include these new specimens. Hence, whilst it might be the first time the new species are classified, it will be yet another classification for the genera or family as a whole, and thus for many of the existing taxa and specimens too. This new classification does not supersede the historical classification, and as such more taxonomies for the same set of taxa accumulate.

Finally, a taxonomist may undertake a revision of an existing taxonomy simply because they do not agree with it. Their conflicting classification will not replace the older classification, for who is to say their revision is more correct than the original? Thus, the new classification co-exists alongside the old, adding to the stockpile of multiple overlapping classifications.

Thus a challenge generated by the way taxonomists work is the management of the accumulation of historical classifications, even if some classifications are not considered valid by the majority of taxonomists.

Prometheus (EPSRC/BBSRC ref. BIO10516), a collaborative project between Napier University and the Royal Botanic Garden Edinburgh (RBGE), is developing a database to support taxonomic working practice. Full details of taxonomic working practice can be found in Pullan *et al* [142], and a description of the Prometheus database to support multiple classifications in Raguenaud *et al* [145]. The strictly visualisation issues described here have arisen out of the work with the taxonomists at RBGE.

3.2 An Example of How Multiple Classifications Evolve

Before examining the taxonomists' requirements, it is worthwhile to consider a small example of how different classifications arise, in order to clarify the process. Figure 3.1 depicts a simplification of the kind of scenario found in taxonomy. The information available grows over time, the criteria used for classification vary and the number of levels (ranks) used in the classification process varies. The coloured shapes at the leaf nodes represent individual specimens to be classified.

The top-left figure (Classification 1) is the earliest classification based on a smallish set of specimens. This classification was based on the shape and resulted in a two-level hierarchy. Square specimens are *typified* by the red square, triangular specimens by the blue equilateral triangle and circular specimens by the yellow circle i.e. these specimens are chosen as representative of the taxons in general. Shapes in general are typified by squares and are therefore represented by the red square.

Types in botanical taxonomy are products of the naming processes that follow after the classification process. Their purpose is to ease naming difficulties in higher taxa caused by reclassifications of lower taxa. It should be noted that a 'type' is not an *average* member of a taxon, rather it is a *representative*.

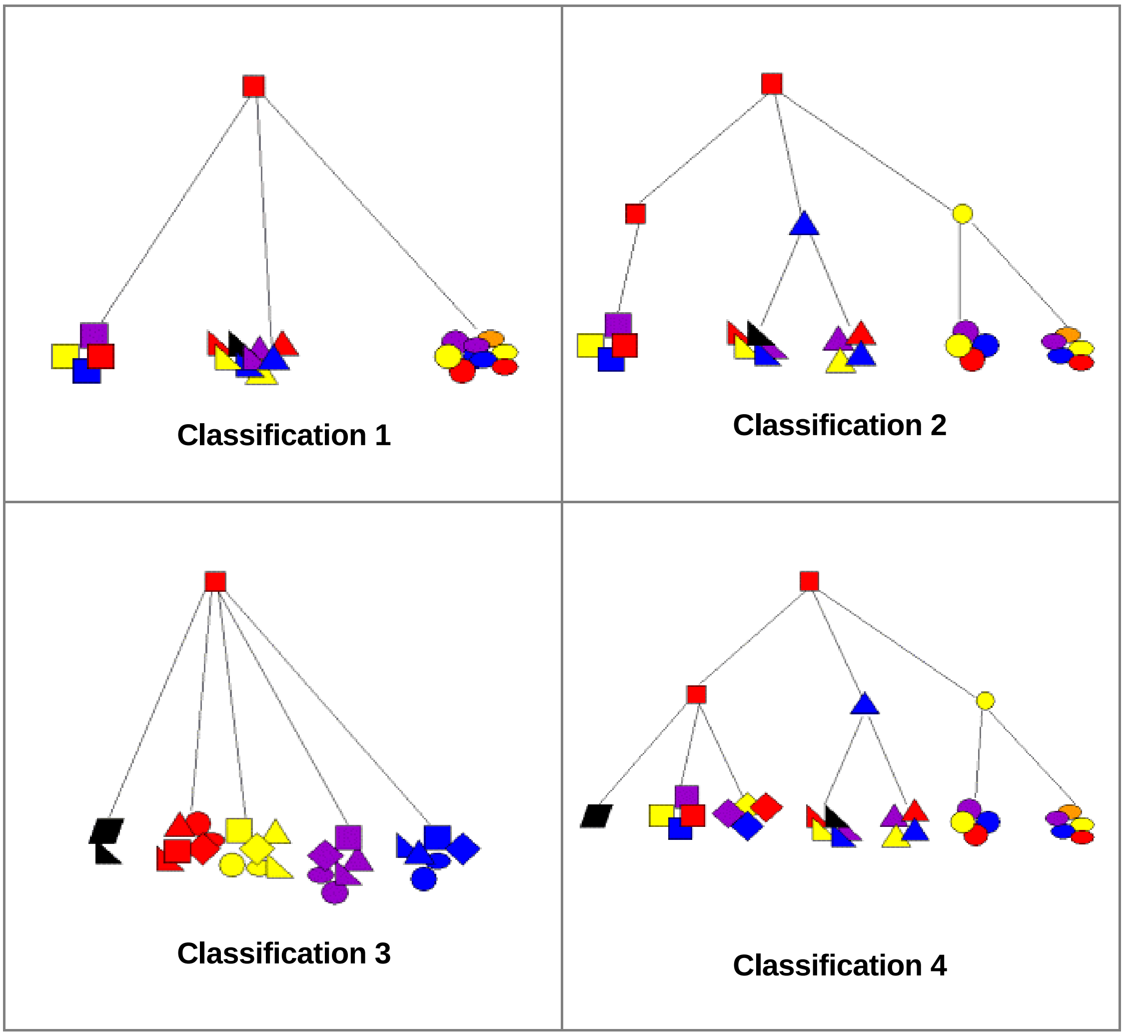


Figure 3.1. Four classifications with overlapping specimens and concepts

Subsequently a second taxonomist (Classification 2) decides that an intermediate level in the classification would make things clearer and introduces the general types: square, triangle and circle. Along with these they introduce two sub-types of triangle, equilateral and right angle, and two sub-types of round shape, circles and ovals. Due to the naming conventions, squares are still typified by the same red square, triangles by the blue equilateral triangle, and circular shapes by the yellow circle. However new types are required for the right-angled triangles and ovals.

A third taxonomist (Classification 3) then decides that shape is not an important characteristic and reclassifies the previous specimens along with some newly found ones, according to their colour. This creates a two level classification with five groups (he ignores one particular colour, orange, as there is only one instance of it). Co-incidentally each group contains an existing type specimen and therefore no new types need to be defined in the classification. In practice often several types will end up in one group, requiring the oldest type specimen to be chosen.

Finally a fourth taxonomist (Classification 4) comes along, and reclassifies the existing specimens and some new examples by shape again.

The reality in taxonomy is much more complicated and involves many more specimens. However, the general principle and reason for the existence of multiple classifications should be clear. It should be noted that this type of organisational classifying behaviour is different from that practised in the modern, emerging field of *cladistics*, which produces sets of *phylogenetic* trees. Phylogenetic trees describe possible evolutionary histories of organisms, and are often produced without having the Linnean system of ranks imposed upon them.

The lack of tools that handle multiple contradictory classifications limits the ability of taxonomists to compare such classifications. In essence taxonomists have a need to represent overlapping multiple classifications to allow them to compare and contrast the classifications produced by different taxonomists or to try what-if scenarios on a classification. A visualisation that supports the work of taxonomists must allow them to explore the similarities and differences between the classifications.

3.3 Taxonomist Requirements

From the preliminary interviews with the taxonomists a number of tasks that a proposed visualisation should be able to carry out or support were discerned. The tasks are quite basic as the paper-based nature of their current system prevents them from carrying out operations on multiple taxonomies in detail. The tasks’ structures are dependent to an extent on the physical basis of the system they use at the moment. However, these are the tasks that the taxonomists hoped to be able to perform using a visualisation:

1. To track a particular taxon, its siblings and parents across re-organised taxonomic structures, if present.
2. To track a particular higher-level taxon's children across re-organised taxonomic structures, if present.
3. To compare the number of distinct levels within and across a set of taxonomic hierarchies.
4. To compare the structure of whole classifications against each other, though this was stated to be an infrequent and secondary task.

Surprisingly, comparison of different sub-trees across hierarchies was stated to be a meaningless concept by the taxonomists, due to the nature of the way in which different taxonomies are arrived at. Therefore, support of this task would not be necessary.

As stated by Espinosa *et al* [57], capturing such task and domain information is a vital step in designing useful visualisations. Most visualisations are influenced, rightly, by the type and scale of information to be displayed. However, it is wrong to use this as the only guide to visualisation construction. As with end-user systems in general, an IV system is composed not only of information to be displayed, but also of the user population and the work they wish to perform. These too should then be considerations in the visualisation design process.

3.4 Multiple Classification Model

After gathering task and domain information, the next step was the full description of the type of information structure that forms multiple classifications. The data used by the visualisation is a simplified subset of the taxonomic data handled by the Prometheus taxonomic database and model [142; 145], which uses and generates a more complicated web of information composed of herbarium types, collectors, and other taxonomy specific fields and information. By contrast, the structures of all classifications used in the visualisation are known in advance and matching elements between them is done on the basis of names only.

The nomenclatural code used in botanical taxonomy has evolved over time and hence will affect the naming of taxa across classifications spanning large time periods. This reduces accuracy when dealing with very old taxonomies such as those produced in the 1800s when naming conventions were either non-existent or localised. Latterly, a set of nomenclature rules, the International Code of Botanical Nomenclature (ICBN) has been enforced, and provides a degree of stability for names across new and modern taxonomies. Some taxonomists argue that the notion of name stability is actually at odds with a field that deals with rapidly changing data sets [50], however, for this thesis work the concern is purely with matching across multiple classifications according to name. Defining a model that will allow matching of non-identical names is not within the scope of the visualisation (this is handled by the Prometheus database) and also allows a resulting model to be more applicable to other domains. In fact, the wish to unite the different visualisation and database models [144] is now the focus of a new research project.

As *taxon* names are re-used (albeit in different contexts) across classifications, the visualisation data model concentrates name and classification information within a set of ‘name’ nodes. One node is assigned per unique name, representing all the non-classification specific information that can be associated with that one name. These 'name' nodes are represented as the labelled elliptical entities in Figure 3.2, one of which is shown in close-up on the left-hand side (representing the name ‘Coriandreae’). Data concerning these names but unique to specific classifications, such as parent and child relationships, are allocated to multiple sub-objects within these nodes, with one sub-object describing the state of one *taxon* name within one classification. These sub-objects and relationships are represented in the figure by the coloured circles that inhabit the name nodes and by the corresponding coloured lines. In essence, the classifications are described separately and bundled together at given points by the 'name' nodes. The relationships that form the links between the taxa (edges) are defined as either child (one or more per sub-object) or parent (only one per sub-object) pointers internal to a particular classification, and have no information attached to them; they are merely pointers, not objects or 'decision-makers' in their own right. On the right-hand side of Figure 3.2 these relationships are shown for a small set of overlapping hierarchies. For example, Genus ‘F’ is a child of Tribe ‘C’ in the red classification, but a child of Tribe ‘D’ in the blue classification.

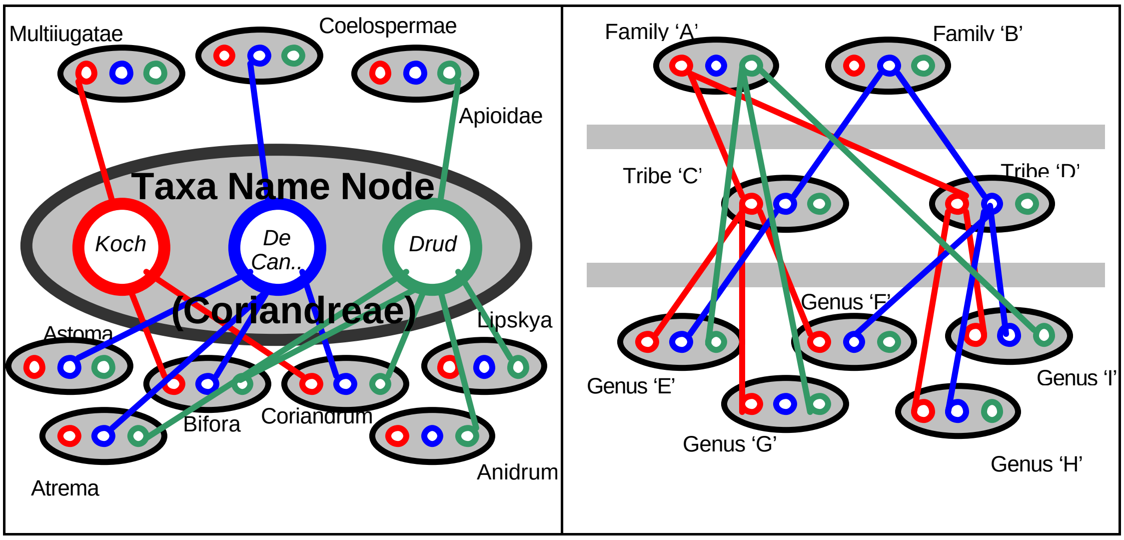


Figure 3.2. Unique names are linked together through multiple classification structures.

Together the classifications and the linking between them afforded by the name nodes build an overall structure we call a DAMG (Directed Acyclic MultiGraph), a restricted class of general graphs related to DAGs (Directed Acyclic Graphs), with the following properties:

· **Directed** when following links exclusively from parent to child links. i.e. direction has meaning.

· **Acyclic** if and only if directed. Following parent-child links recursively through the structure will never bring a path back to where it started. Accordingly, *self-loops* (edges with the same end and start node) are not allowed on a node.

· **Multigraph** – Multiple edges can exist between a pair of name nodes. This would indicate the same immediate relationship existing between two nodes in different classifications. Self-loops are not allowed (a restriction on general multigraph theory) to help preserve the acyclic property of the structure. An example can be seen in the right-hand side of Figure 3.2 where there are two distinct and direct links, red and blue, between the nodes Tribe ‘C’ and Genus ‘E’.

· **Layered** – The structure can be broken into distinct layers, where no direct links exist between nodes in the same layer. This is implicit in the taxonomy example, where *taxa* at a certain rank are composed only of, and therefore linked to, *taxa* from lower ranks. This can be seen on the right-hand side of Figure 3.2, where the thick grey lines separate out taxa belonging to the same rank, internal to which there are no direct links.

Searching and linking within this structure can now take place using the following traversal techniques. Traversal from a particular node for a given classification is simply a question of performing standard depth-first and breadth-first searching mechanisms for trees. Only the tree in which the traversal takes place, indicated by a simple integer index, is tracked to keep traversal operations within the correct node sub-objects. Traversal between classifications is simply a matter of switching between the sub-object within the nodes that hold relationship information, and in this way relationships between the classifications can be explored. Perhaps a useful metaphor can be given by examining Figure 3.2 and thinking of the coloured classifications as different underground or metro lines displayed on a map. The name nodes act as “transfer stations” that allow a traversing algorithm to hop off one “line” (classification) and onto another.

This approach to modelling the taxa data gives access to ready-made hierarchies within the overall graph structure, as in effect a restricted graph is constructed from hierarchies rather than vice versa, eliminating the problem of extracting individual taxonomic classifications from the overall graph. Furthermore, speed is an important factor for an interactive visualisation, and having all the classifications connected together but easily distinguished makes the operations performed on the visualisation's data model extremely efficient compared to the case for a general graph.

It must now be asked how the model would facilitate the tasks deemed by the taxonomists to be basic requirements for a suitable visualisation.

- To track a particular *taxon*, its siblings and parents across re-organised taxonomic structures, if present.

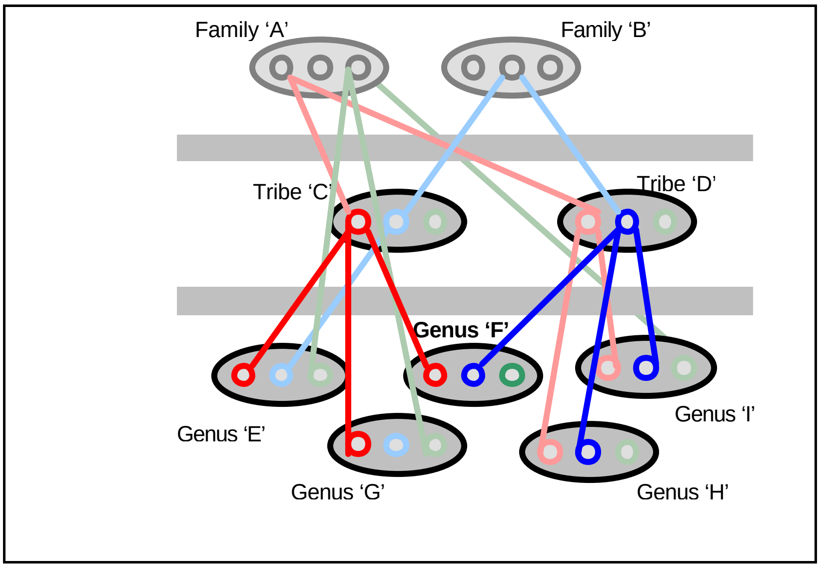


Figure 3.3. Tracking a taxon's siblings and parents in the model.

Selecting a name node effectively marks it as selected across all classifications that it occurs in, as selection is a global attribute within each unique name. Therefore this operation automatically marks out a particular node regardless of classification considerations. However, it is up to an associated visualisation to display this information in an understandable manner.

Siblings and parents of this chosen node may be marked out on a per classification basis by accessing the appropriate classification sub-object within this node. The parent link in the sub-object will then point to its parent node for that classification, and this node can be marked accordingly. Similarly, accessing the corresponding classification sub-object in the parent node, and then following the child links will give access to all sibling nodes of the original node (a sibling being a node with a shared parent), including the original node itself. These can be marked as selected too. The whole operation can then be repeated if needed on other classification sub-objects in the original node, to obtain parents and siblings in other classifications. An example can be seen in Figure 3.3 where the parents and siblings of Genus 'F' have been located by following the links for each classification that it is present in.

- To track a particular higher-level taxon's children across re-organised taxonomic structures, if present.

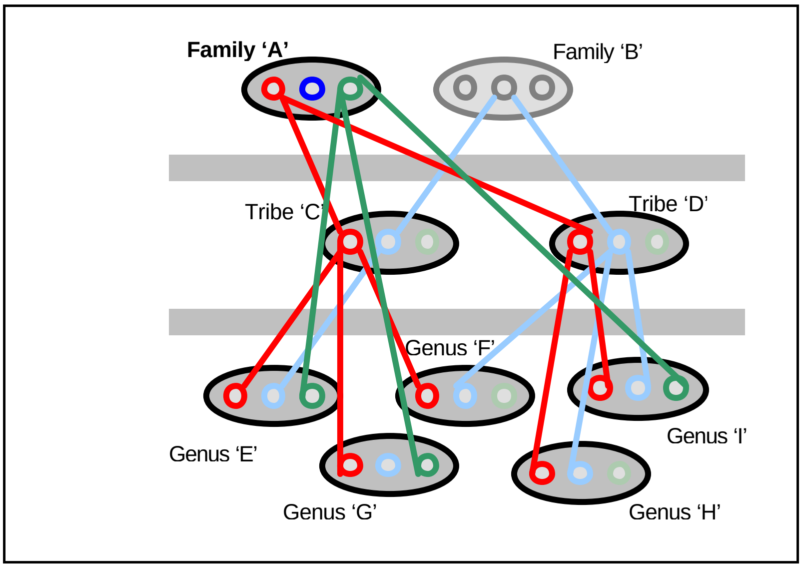


Figure 3.4. Tracking a non-leaf node's children in the model.

This operation is similar to that described above, in that child links for a particular node and classification are followed and the nodes they point to are marked. For all descendent nodes to be marked, the process is repeated recursively, making sure to keep within the correct classification. In Figure 3.4 the children of Family 'A' within the red and green classifications have been marked out recursively. Family 'A' does not occur within the blue classification, but to view where any of Family A's descendant nodes occur within the blue classification, if they are present, is simply a matter of following the blue links upwards.

- To compare the number of distinct levels within and across a set of taxonomic hierarchies.

Comparing ranks for taxonomists is a matter of comparing where taxa are placed in the taxonomic hierarchy of family, genus etc rather than comparing the depth from the root of nodes within an individual classification. As such each 'name node' has an attribute indicating this rank, as it will be invariant across classifications it therefore does not need to be stored on a per classification basis. Comparing the classifications is a matter of traversing each classification structure and identifying the ranks of the nodes along the way. In Figure 3.4 it can be seen that the red classification encompasses nodes on 3 different levels, whereas the green classification includes only 2 levels, skipping any nodes in the central level. Performed as pre-processing prior to a visualisation, this can allow the display of labels or signifiers of ranks associated with each classification. Again, it is up to the visualisation to render the provided information in an understandable form.

- To compare the structure of whole classifications against each other, though this was stated to be an infrequent and secondary task.

This operation is even more dependent on direct visual comparison of taxonomic classifications, and therefore on the qualities of a visualisation. The model plays its part by ensuring the classifications are separated logically, via the classification sub-objects, and as such are available to the visualisation for whatever display technique is deemed appropriate.

The structure also allows other operations to be performed on the multiple classifications. For instance, nodes unique to a particular classification will have only one, or one non-empty, classification sub-object. This again is a single query on the node and will not involve any searching or traversal operations.

3.5 Data Sets

The taxonomists at the RBGE supplied two data sets of multiple, overlapping classifications. The first contained historical classifications of genus-level taxa, and the other containing ongoing revisions of one specimen-level data set taken from the Prometheus database.

The first data set, the genus-level data, consisted of seven historical classifications of the *Apiaceae* family, which is roughly speaking the carrot family. These classifications were constructed during the period 1820-1962 and represent different taxonomists' views on the family, at different levels of completeness, according to varying criteria, and established over an evolving set of working practices in the field during the 143-year period it covers. Unsurprisingly, the result is a set of very different classifications. During the course of this thesis work an eighth classification, constructed during 2000, was added. This classification was based on morphological and molecular characteristics. In total, the final data set consisted of 776 genera, each of which occurred in some or all of the classifications, forming a final count of 1,500 placements of genera and other taxa up to the family rank in the eight classifications.

The second data set consisted of four snapshots of an ongoing revision of the specimens within the *Globba* genus, a member of the ginger family found in South-East Asia. Each snapshot encapsulated a different stage of development as the classification progressed from an initially amorphous and unclassified pool of specimens. Correspondingly, each subsequent revision in the data set was producing a classification with a more definite structure, and a smaller group of unassigned specimens. This behaviour is similar in some aspects to the evolving web ecologies visualised by Chi *et al* [40], as described in Section 4.2. However, the focus of the problem here is to visualise change in the structure, rather than change in the information attached to nodes, such as traffic load, as was the case with Chi *et al.*

There was also a fifth, smaller, classification consisting of a basic historical categorisation of some specimens dating from 1904. The revising taxonomist considers the modern Globba genus to contain approximately 1,400 specimens. Over the four revision snapshots this gives 5,600 specimen placements within this data set, a much bigger proposition than the Apiaceae data set. One feature of this data set was that the names were much more stable than in the genus-level classifications. Two factors were at work here; one was that specimen names cannot change once allocated (specimens are not taxa, they are the fundamental physical plant objects that are then organised into taxa, thus there are no sub-components of a specimen to be re-arranged that could affect their naming). The second factor being that as just a single taxonomist is producing the taxonomy in a relatively short period of time, it was likely that they would keep the same higher taxa names across all revisions.

3.6 Data Format

One issue to come up as a result of the development of the data model and obtaining the initial data sets was: what format should the data be stored in? There are numerous data formats for describing tree structures, and focus was given to those designed for use in bioinformatics as the initial data sets were botanical classifications. The format settled on was an extension of the NEXUS format [118] used for systematic data, which stored all the relevant data in a single text file. It was hoped to use an unaltered NEXUS syntax to describe the trees to enable compatibility with existing data sets, as one of the biggest logistical problems that occurred was the initial hurdle of converting paper-based data to electronic form. Having access to readily available sources of data would have been extremely useful, but unfortunately proved not to be possible due to two problems. Firstly, the basic NEXUS syntax would not allow the description of pre-defined taxonomic ranks, as the tree syntax concentrated on allowing the definition of phylogenetic data. In such data, a node’s depth is described as the continuous sum of similarity distances on the path between itself and the root i.e. the sum of intervening branch lengths. This is in distinction to the Linnean taxonomies such as the Apiaceae and Globba classifications, which are organised hierarchically on fixed, discrete levels such as genus and family. Also, phylogenetic trees only have data attached to terminal nodes. Taxonomic hierarchies have taxa placed at internal nodes and again most NEXUS data sets did not incorporate such information. Thus, the syntax had to be augmented as shown in Table 3.2 to enable the taxa information to be described sufficiently, but this at least allowed a whole multiple tree data set to be defined in one text file.

```
BEGIN TAXA;
DIMENSIONS NTAX=776;
TAXLABELS
'Apiaceae 0'[1]
'Pauciiugatae 1'[2]
'Multiiugatae 1'[3]
'Orthospermae 1'[4]
'Camplyospermae 1'[5]
'Coelospermae 1'[6]
'Heterosciadiae 1'[7]
'Haplozygiae 1'[8]
'Diplozygiae 1'[9]
'Formae_Desciscentes 3'[10]
'Hydrocotyleae 3'[11]
'Bupleureae 3'[12]
```

```
TREE Berchtold_&_Presl_1820=
(0, (1, (17, (96, 97, 98, 99, 100,
519), 12, (58, 59, 60, 61), 15,
(83, 84, 85, 86, 88, 89, 90), 10,
(44, 45, 775, 47, 776, 49, 50, 51,
52), 11, (53, 54, 55, 56, 113, 57),
13, (62, 63, 64, 65, 66, 67, 68,
69, 70), 16, (91, 92, 93, 94, 95),
18, (102, 103, 104, 105, 106, 107,
108, 109, 110, 111, 112), 14, (71,
73, 74, 75, 76, 72, 77, 78, 79, 80,
81, 82))));
```

```
TAXADEPTHS=
0 Family,
1 Sub-Family,
2 Legion,
3 Tribe,
4 Sub-Tribe,
5 Grex,
6 Genus;
```

Top Left: Fragment of taxa name index, each name incorporating a reference to their respective taxonomic rank.

Bottom Left: Index of taxonomic ranks, as referenced by the taxa name index.

Top Right: One classification, brackets indicating parent-child relationships, numbers referencing to taxa name index. In this manner, multiple trees can use the same taxa name index.

Table 3.2. Modified NEXUS data format for multiple taxonomic tree description.

Ruths, Chen and Ellis [151] encountered a similar problem of trying to convert a phylogenetic data syntax for use with taxonomic rank structures in their Arbor3D visualisation. Using the same Newick data format [59] that NEXUS adopted for tree structures, their solution of attaching a rank value to each taxa name is similar to that used here. They did not split the name and rank value types into separate tables, or de-reference the taxa names from the tree description, as their visualisation was only of one taxonomic tree and hence all the necessary information could be concentrated together in the tree description.

Lately, syntaxes for describing graph structures have began to appear in the Information Visualisation and Graph Drawing fields, motivated by the need for a common, extensible language to enable data exchange between different graph visualisation systems. The most noteworthy is perhaps Herman and Marshall's GraphXML format [86], based on the XML (eXtensible Markup Language) standard. GraphXML allows the description of multiple graphs along with whatever associated properties anyone could wish to tag on collections of nodes and edges. However, inspection showed that nodes and edges were local to each graph description; that is, a set of external nodes cannot be defined to be used on several graph structure descriptions. Each graph must have node information redundantly repeated and also a mechanism for inter-connecting the graphs must then be constructed. Undoubtedly this could be done in XML but the effort required would appear to be almost that of constructing a new XML format from scratch.

XML formats for taxonomic data and classifications are now also under ongoing development, an example being Gilmour's Taxonomic Markup Language [74]. However, at the time of writing, this format only supported one classification per XML file, meaning that several would be necessary for use in our visualisation. Also, a mechanism for cross-referencing taxa between the classifications would need to be constructed, the same problem as with the GraphXML format.

In conclusion, despite the apparent simplicity of the data, it was found that there were no existing data formats that could fully describe overlapping taxonomic hierarchies. In that case, adoption and adaptation of a format is almost an arbitrary choice, depending on the amount of extra work needed to describe the data. Therefore, as the original data set was taxonomic, use of the augmented NEXUS format to store it was continued.

3.7 Conclusion

In this chapter, the mechanisms by which taxonomic classifications can occur have been described, along with the problems and challenges they pose to working taxonomists. Consequently, this led to the design of a data structure that could hold these multiple classifications at an elementary level, ignoring additional data such as herbarium and collector that a fuller descriptor such as the Prometheus database must encapsulate. The model was demonstrated to be able to readily provide answers to the initial requirements put forward by the taxonomists, and should thus provide a suitable basis for a visualisation to display and allow interaction with multiple, overlapping classifications, accommodating the functions necessary for the performance of the requested tasks. It should be borne in mind that the model only provides the opportunity for a visualisation to do this, it doesn't guarantee that an appropriate visualisation will be produced. Just as an ill-thought out data model could cripple a potentially useful visualisation, care must be taken so that a suitable model is not rendered useless by an inappropriate visualisation, as both components are needed to operate successfully for the system to perform adequately. It was also found that the data format for storing the multiple taxonomies was problematic, given that no existing format was designed to cater for such data as the multiple Apiaceae and Globba classifications. The solution was to adapt an existing syntax, the NEXUS format, which consequently allowed the description of multiple, overlapping hierarchies.

4 Limitations of Existing Multiple Tree Visualisation Techniques

From the taxonomic problems described it is important to provide a visualisation that allows taxonomists to compare and contrast classification hierarchies. Previous visualisations that have aimed at showing change in structures have settled primarily on one of two approaches. Firstly, visualisations have animated the information regarding the structure, relying on the animation's inherent perceptual qualities to enable users to comprehend the change, as is pointed out by Bartram [9]. Secondly, some visualisations have laid out a series of snapshots of the structure at critical phases, allowing the user to visually compare all or most of the relevant information at the same time. This type of visualisation is best known as Tufte's 'small multiples' [165, Ch. 4]. Lastly, a further technique is to combine the multiple structures into one integral visualisation, either by visually overlaying structures so they match up at the corresponding places, or by logically fusing the structures before the visualisation stage.

These three techniques and their resultant visualisations are evaluated below as possible solutions to the problem of visualising multiple overlapping classification hierarchies.

4.1 Animation

A number of visualisation techniques, including Huang and Eades' visualisation of huge graphs [89], the Latour tree visualisation system described in Herman *et al* [88], and Wittenburg and Sigman's Treeviewer [175], use animation as the cue to show change in the structure of a hierarchy. Animation's perceptual qualities for attracting the user's attention are well known, and also allow a user to easily track incremental changes between stages of development.

In Huang's technique, a large graph is visualised as a tree by strategically omitting certain links in the display. The hierarchy itself is then displayed using a spring-mass metaphor as shown in Figure 4.1. The visualisation can be regarded as showing a roving area of focus within a larger graph structure, and shrinking and expanding of sub-trees within this structure cause the animated change in the visualisation, rather than the display of a succession of differing structures as would necessarily be the case with the taxonomy hierarchies. Therefore, this particular visualisation technique is unsuitable for tackling the problem central to this thesis.

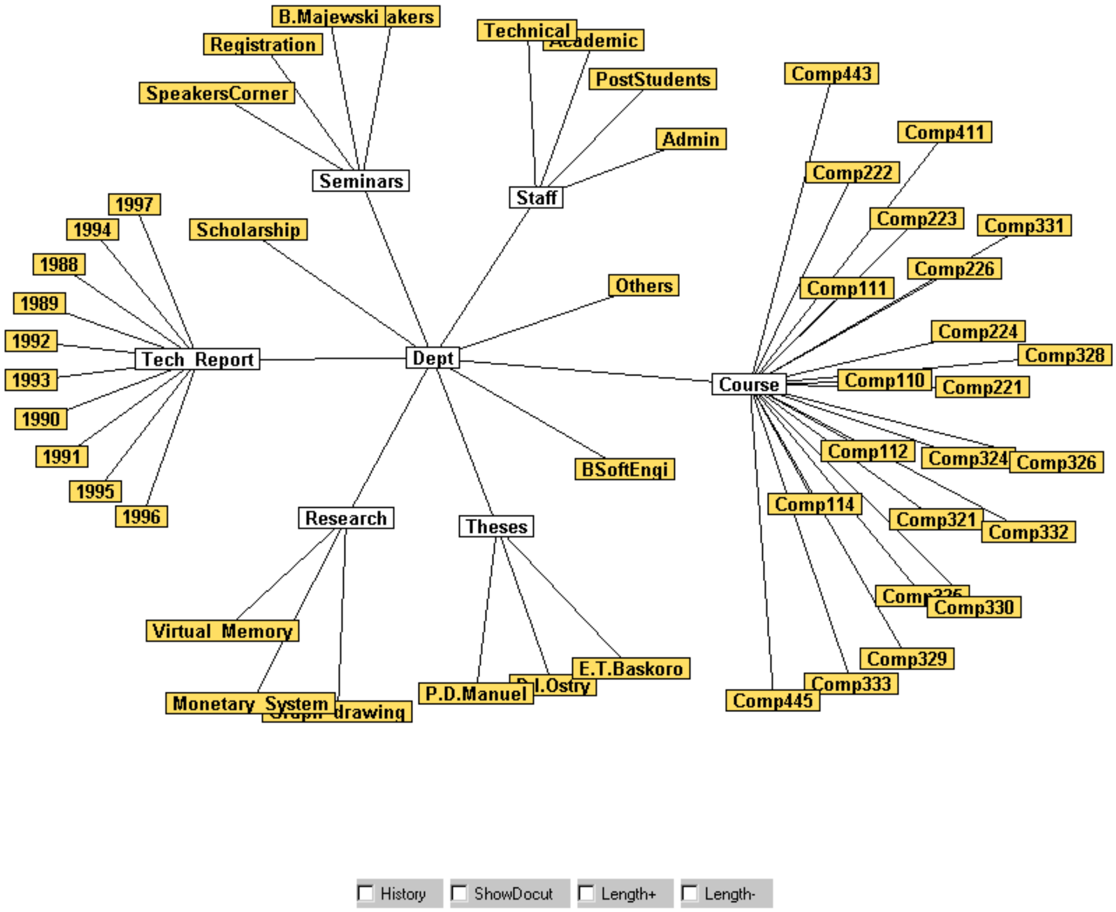


Figure 4.1. Huang and Eades' Huge Graph Viewer.

The Latour system is a visualisation framework for trees and directed acyclic graphs. It includes an animation option that can be utilised to explore a sequence of trees. Herman *et al* state that the animation is used to show an "evolution of data over time", taking as input data a base tree plus the differences associated with each successively displayed tree compared to this base tree. As with Huang's visualisation, this system does not show the reclassification of a hierarchy, but shows the gradual development of a hierarchical structure.

Wittenburg and Sigman's Treeviewer is a mainly textual visualisation of web-search queries, and again animates addition/deletion of nodes, rather than the reclassification of the hierarchy's existing nodes. In Ghoniem and Fekete's [73] recent work on animating Treemaps the focus is on visualising changes in layout caused by adjusting the metric associated with the Treemap layout algorithm. The visual structure may alter in some cases, but the underlying logical structure does not. They do however note that the comprehension of addition and deletion of nodes within a structure could be aided using their animation process.

Animation itself has two intrinsic drawbacks for the type of information to be visualised here. Firstly, animation allows only direct visual comparison between two states, the last and the next stage of the animation. Other comparisons between states resulting from the animation must be recalled from memory, pushing the work onto the cognitive rather than perceptive systems. Secondly, whilst animation works for visualising gradual changes, such as Huang's traversal of a graph structure, it would become overly complex for major structural changes caused by reclassification of existing nodes. Wittenburg and Sigman acknowledges this point by stating that their system could employ a fade-in/fade-out approach between two states or structures where there is a poor degree of correlation, rather than utilising the 'in-betweening' style of animation. The response of Herman *et al* is to specifically indicate that the animation is to display an evolution of a base tree, rather than show drastic reorganisations of a set of nodes.

4.2 Small Multiples

Some visualisations allow comparison using Tufte's [165] idea of small multiples, where miniatures of changing data sets and structures are laid out beside each other for visual comparison. Such multiple visualisations have to be 'miniaturised' to fit in the available display area, but are extremely well suited for interactive *linking* techniques as described in Chapter 2.5.3.

Chi *et al* [40] discuss the problem of displaying the evolution of a website over a number of months, a situation and data set they describe as a *web ecology*. In their visualisation, displayed in Figure 4.2, certain points during the period are visualised by displaying the sites' hierarchy in the form of a compact 'disk tree'. The disk tree is similar in construction to the 'space-filling radial' visualisations described in Chapter 2, with successive disk trees displayed next to each other, enabling visual comparison. The set of multiple disk trees is termed a 'time tube'.

Leaves and paths in the disk trees are coloured green, with the saturation of the colour directly related to the amount of traffic in that part of the web site. Yellow and red representations denote nodes that have been added or removed from the web site since the last snap-shot.

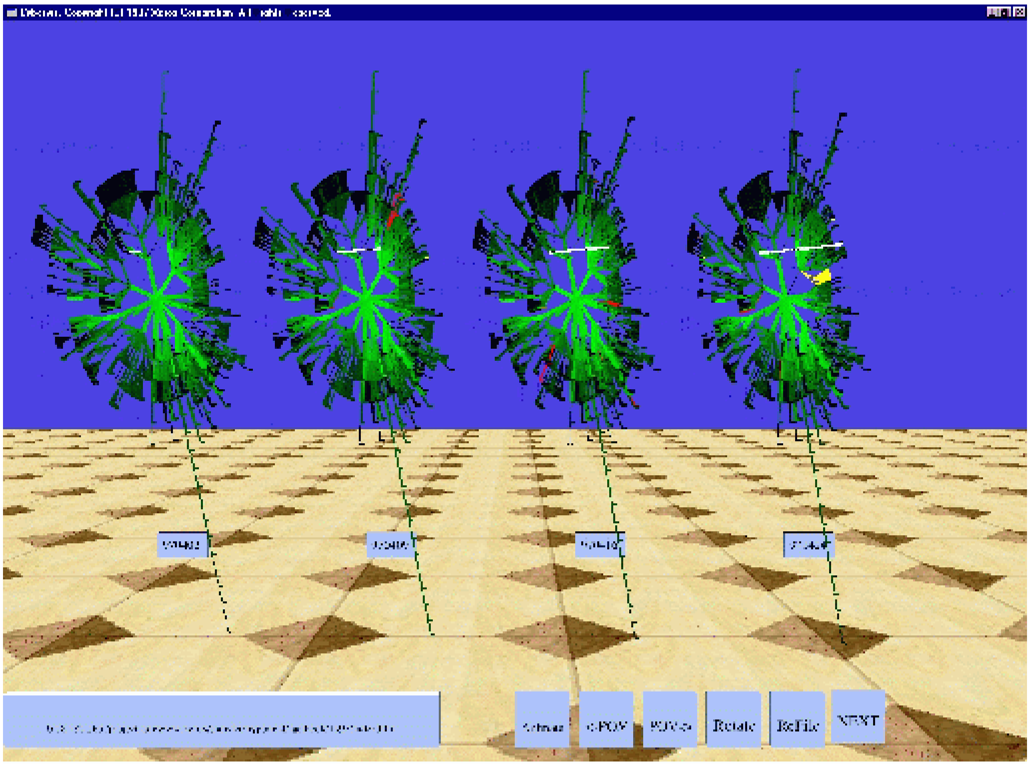
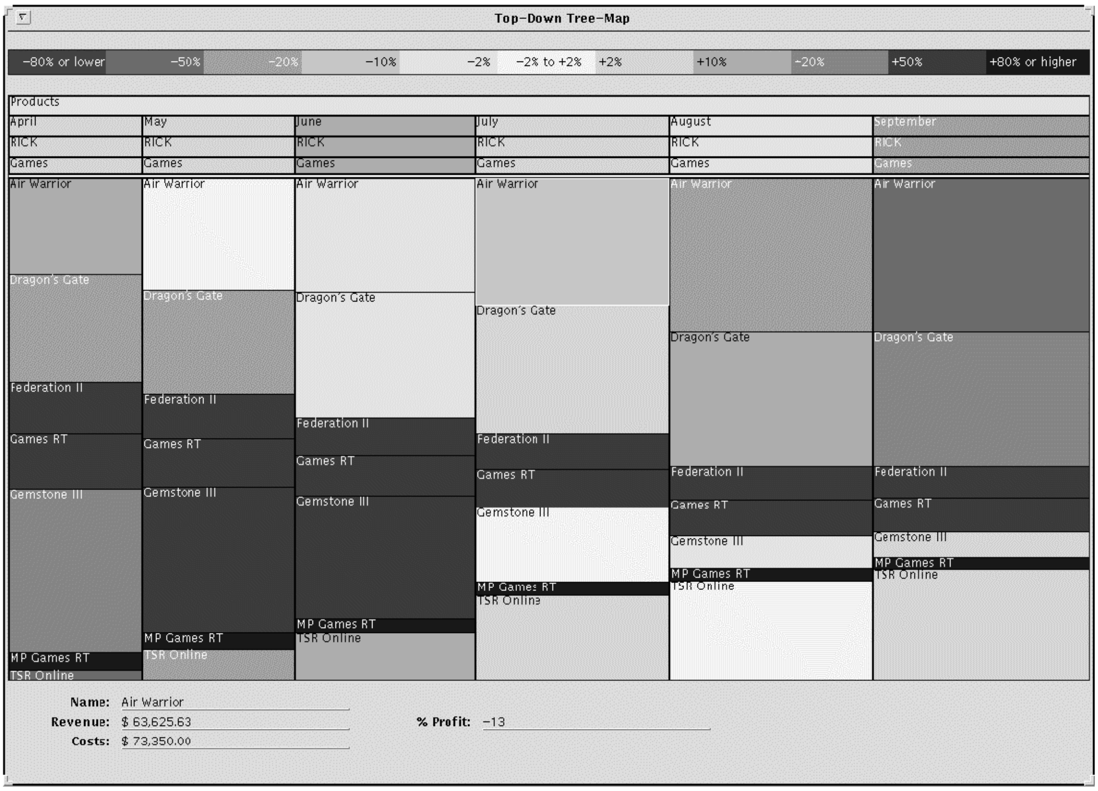


Figure 4.2. Web Ecology Visualisation by Chi *et al*.

The drawbacks of this technique, when considering the multiple taxonomy requirements, are that the visualisation is designed to highlight evolution of a hierarchy resulting from addition or deletion of nodes. Consequently, visual prominence within the evolving structure is given to these processes. A visualisation that shows a *restructuring of the hierarchy* resulting from a reclassification of the nodes from which it is formed is required. In the web ecology, nodes that already exist but have changed their links, and hence altered their position within the hierarchy are not differentiated from nodes that have remained static within the hierarchies' organisation. There is a provision for highlighting a particular node's progress through a time tube, but due to the manner in which the disk trees are drawn, all nodes are displayed at the same position within each tree. Therefore seeing 'movement' due to restructuring is not aided.

Another visualisation prompt that is not present here, nor in the other techniques, is the ability to see the *context* in which a particular node has been restructured. The context for a particular node is the other nodes with which it has relations in the hierarchies, namely its parent, sibling, and child nodes as appropriate. This is necessary for the multiple taxonomy visualisation, as nodes do not actually move when conceptual structures are re-organised (reclassified). Rather, they are *grouped* differently, possibly with different nodes. This principle also applies to sub-trees of various sizes, as well as individual nodes. Knowledge of the other nodes or sub-trees with which it shares relations in each version of a hierarchy will hint at the methodology behind that particular classification.

Turo and Johnson’s visualisation technique [166] pictured in Figure 4.3, and based on Johnson and Shneiderman’s Treemaps [96], also includes an option to visualise change in trees or sub-trees over time, again using the small multiple approach. However, as with Ghoniem and Fekete’s Treemap animation technique, the changes they are concerned with are related to information attached to individual nodes, rather than changes in the structure of the hierarchy.



Name: Air Warrior
Revenue: $ 63,625.63
Costs: $ 73,350.00
% Profit: -13

Figure 4.3. Turo and Johnsons’ comparative sub-trees using Treemaps.

Two software visualisations tackle the problem of displaying multiple trees by using small multiples, though both are solutions to different problems. Brazell and Jeffrey’s [23] visualisation is of tree data structures that are activated during the runtime of a C program. Their visualisation displays a vertical arrangement of multiple trees, each drawn in a node-link style, but their example is limited to quite simple trees. Changes that occur to any particular tree are shown, but again, these are limited to showing addition and deletion of nodes rather than re-arrangement of existing nodes.

The second system that uses the small multiple approach is the software visualisation system of Gall *et al* [67], which displays the growth in the structure of a software package during iterative development. It differs from the other small multiple visualisations in that it utilises a 3D approach to visualise a set of hierarchies, with the third dimension being used to represent time. Hence the individual hierarchies, corresponding to the version release dates for a software package, are positioned at various points along this axis instead of at different positions in a 2D space. This has the advantage of enabling a comparison of levels across the hierarchies by the judicious positioning of the 3D viewpoint. However, it resurrects the problem of occlusion that occurs in 3D visualisations. For example, in the screen-shot of Figure 4.4, it is easy to compare the top nodes of each hierarchy, relatively easy to judge the middle level, but the bottom levels are obscured by the higher levels of the hierarchy displayed in front. Also, once again it is change in node information and not in the containing structure that is being displayed here. Parts of the software system that are not present in certain releases are represented as black-coloured nodes, and hence each hierarchy has the same spatial structure.

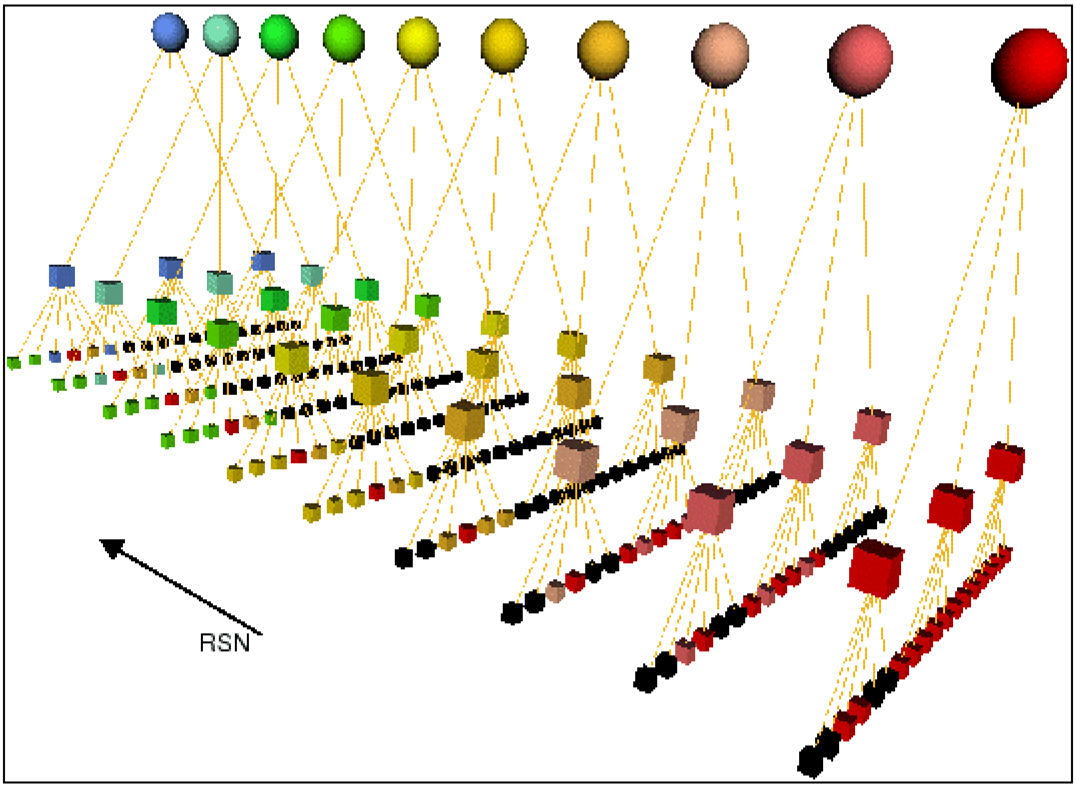


Figure 4.4. Visualizing Software Release Histories by Gall *et al*.

A further IV application, the NicheWorks environment by Wills [173], which is a large-scale graph viewer, has as part of its' future work plan a 'time-series' option to 'measure evolution and structure changes in graphs'. However, at the time of writing, this has not been carried out, and furthermore it is unknown whether this is to be a statistical or visual approach.

In conclusion, the main disadvantage of the small multiple approaches is a simple lack of space on-screen due to the resolution and size of the average monitor. To visualise a larger set of hierarchies will require smaller, more compact visualisations that are still intelligible to the viewer. Also, they lack animation's more powerful pre-attentive visual cues, hence placing more cognitive load on the user when using the visualisation.

4.3 Other Approaches

Another approach of interest to this field is Furnas and Zacks' Multitrees [66], describing both a graph structure that enables the re-use of hierarchical structure, along with a visualisation of such a structure. Furnas and Zacks' work diverges from the specific problem this thesis tackles as their system re-organises entire sub-sections of an existing hierarchy to give different viewpoints on the hierarchy. The example they use is professors giving different reading lists composed from fragments of a hierarchy of volumes, books, chapters and sections. Conversely, taxonomy re-organises and re-uses just the nodes. Pieces of previous taxonomies may be incorporated into another taxonomy, but this is not always the case.

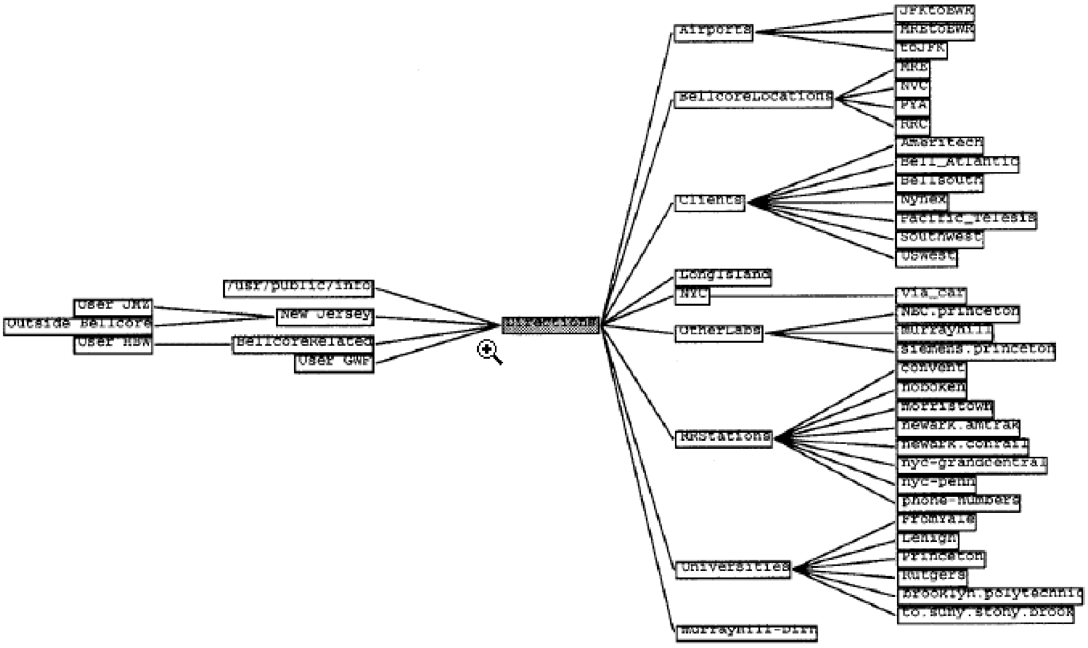


Figure 4.5. Furnas and Zacks' Multitrees visualisation

Multitrees' specification also forbids two nodes from having more than one distinct path between them. Multiple hierarchy taxonomies often have this characteristic, whereby one particular species would have two distinct parents in different taxonomies, but then both of these taxa would have a common parent in their respective hierarchies. Therefore between the family and the species would lie more than one distinct path, so Multitrees could not fully describe the inter-relations between the multiple hierarchies this thesis is concerned with. Furnas and Zacks acknowledge this, and describe it as a systematic problem for the Multitrees' structure and visualisation. They suggest that such structures could be broken down into separate Multitrees, and the classifications could then be visualised separately. This means that correlations between different classifications cannot be shown directly, and it is also indicated that the approach was not carried out but remains a putative solution to the problem.

Their visualisation focuses upon one particular node, from which descendent and predecessor nodes are drawn in columns, with links drawn between related nodes, as in Figure 4.5.

Research by Wittenburg *et al* [174], into visualising overlap between different browser bookmark files, is probably the closest work in concept to the problem tackled in this thesis. It combines the Multitrees' graph structure, which is used to amalgamate a number of bookmark structures, with a small-multiple Treemap-based visualisation of the separate bookmark hierarchies. The whole visualisation runs under the Pad++ zooming user interface environment [18].

It can be argued that as the individual bookmark hierarchies re-use only the individual bookmarked pages, and not any higher structures, it is not a true example of Multitrees as Furnas and Zacks envisaged the concept, being closer in methodology to the node-only re-classifications of taxonomy. However, this is a moot point as it is the visualisation that interests us most.

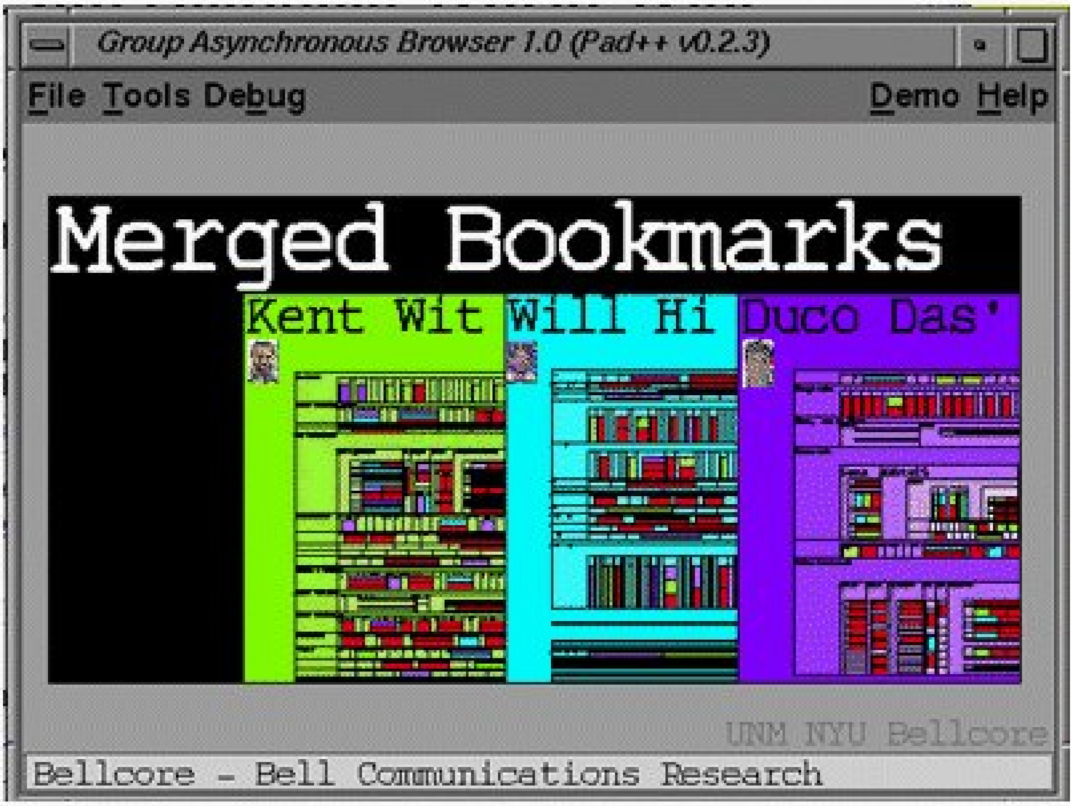


Figure 4.6. Multiple Treemaps with colour-coded cross-references.

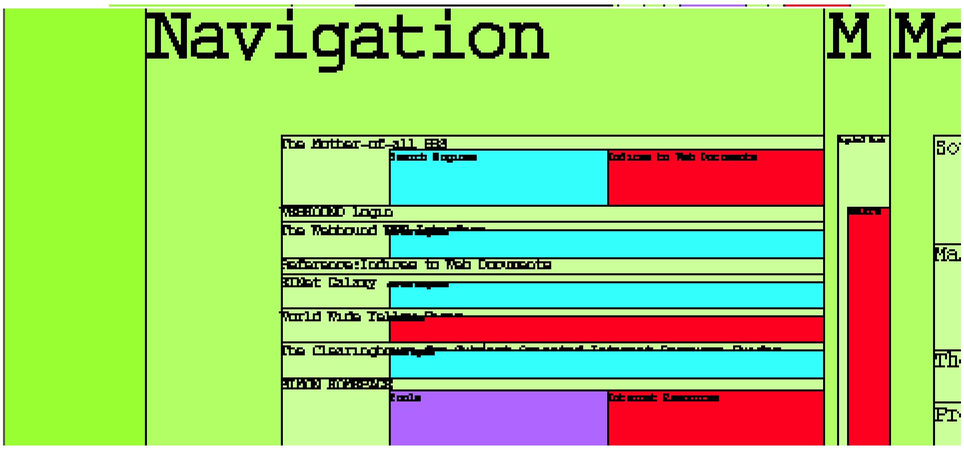


Figure 4.7. Close-up of one Treemap, showing cross-reference nodes more clearly.

Each bookmark hierarchy, displayed as a Treemap, is given an associated colour that is used to mark its particular nodes. Additional cross-reference nodes are placed in a Treemap if a particular bookmark is shared between bookmark hierarchies, and these cross-reference nodes are coloured according to the hierarchy they refer to. Screenshots of the visualisation are shown in Figure 4.6 and Figure 4.7.

This system would be enough, when applied to multiple taxonomies, to show the shared nodes between each classification. However, it can give no indication of how one particular sub-tree in a particular hierarchy, equivalent to a family or genus, is distributed by itself or in respect to its' peers. This information is unavailable as the colour-coding of the nodes is static and set for each hierarchy. Information regarding the distribution of parts of a particular tree is therefore indistinguishable among the other parts of the hierarchy.

This is an important distinction, as unlike the bookmark hierarchies there is a high degree of overlap between taxonomic hierarchies. Applying Wittenburg *et al's* visualisation would simply show a large number of cross-reference nodes with no detailed indication of how they group with each other across the other hierarchies.

A further system that could represent overlapping classification information is Harel's *Higraphs* [79]. Rather than being a computer-based visualisation, Higraphs are manual diagram constructions, much like Venn Diagrams, for visualising structures with properties associated with both sets and graphs.

The basic approach is to draw nested and intersecting sets as in a conventional Venn Diagram and then to indicate further relationships through the addition of directed *hyperedges* between sets (hyperedges are graph edges that may connect more than two objects). The relationships indicated may apply to the sets linked directly to the edge, or to various subsets of the connected sets on the hyperedge. This latter situation will abstract the precise relationships, but reduces the visual clutter that would occur if the exact sets and elements involved were to be joined individually. Formalisms for representing operations such as Cartesian products on sets are also described, and Harel constructs Higraph examples of Entity-Relationship diagrams and state charts.

The Ztree [12] visualisation by Bartram *et al* uses a restricted Higraph style representation (nested but not intersecting sets) to display a hierarchy with additional non-hierarchical links. As multiple hierarchy information can be modelled as consisting of sets of hierarchical data interconnected by a graph structure across the taxonomies, an initial examination of the Higraph approach with regard to this information was made.

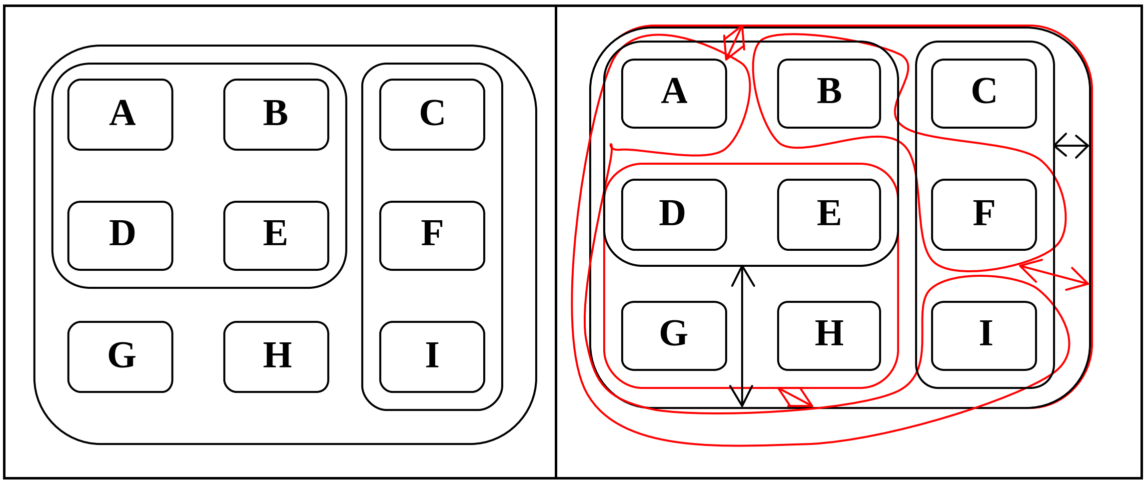


Figure 4.8. One and two overlapping classifications shown using Higraphs.

It quickly became apparent that even a simple example of multiple classifications produced problems. The left-hand side of Figure 4.8 shows a classification based on 9 sets or elements, labelled A to I, and for one classification this is simple enough. However, when a second classification is described, as on the right-hand side of the Figure, difficulties arise in interpretation. The sets intersect each other and to distinguish the classifications, edges are introduced that link sets to parent sets for specific taxonomies (the second classification is merely drawn in a different colour to help further distinguish the two classifications). Although the figure does display both classifications, the convoluted set boundaries combined with their many crossings introduce visual interpretation problems associated with general graph node-and-link visualisations. Harel stated that these *overlapping* ‘blobs’ may well affect the clarity of the diagram, but at the time no relevant experiments such as those of Purchase *et al* [143] had been performed. Considering that this example consists of just 2 overlapping classifications based on 9 nodes, it is not hard to extrapolate that this will not be a suitable method for displaying N overlapping classifications over hundreds or possibly thousands of nodes.

4.4 Botanical Classification Visualisations

During the course of this thesis work, visualisations that displayed multiple classifications of botanical data were also being developed at other institutions, though each displayed the data using distinctly different methods. Investigating both visualisations, it was found that neither supported the tasks that the taxonomists felt they needed to perform.

An example of a botanical taxonomy browser, developed at the same time as this thesis work, was implemented by Amavizca *et al* [4]. Termed the ‘3D Tree’, it is based on a Cone Tree visualisation and purports to show multiple taxonomies through this 3D-tree structure. The system does this by displaying alphabetical ranges of genera underneath the appropriate family root node, selection of which expands the range into a more detailed selection of nodes. The tree is not structured after any particular taxonomy, and as such no intermediate levels are displayed (genus and family are compulsory taxonomic ranks, intermediate ranks are taxonomy dependent). Resultantly, all the taxa from the multiple taxonomies are merged, but at the expense of losing visual indicators of the structures that defined the taxonomies; therefore direct comparison of taxonomies, or subsets thereof, is not possible. Such comparisons, which are part of the taxonomists’ requirements, are noted by Amavizca *et al* under the heading of future and ongoing work.

Klingner’s work [103] focuses on visualising statistical similarities between a set of *phylogenetic* trees, which are structures that describe possible evolutionary histories for a group of organisms. Rather than direct visual comparison of the different trees, Klingner used a multi-dimensional scaling algorithm that reduced the trees to points in a 2- or 3-dimensional space, similar in approach to the application of force-directed methods to multi-dimensional data mentioned in Chapter 2.4.3. Thus, the distance between any two points in the visualisation is proportional, as far as can be, to the structural difference between the two trees represented by those points.

The visualisation allows selection of a particular point or group of points, upon which the associated tree or *consensus tree* (a kind of ‘average’ tree of the set of selected trees) is drawn in full. The consensus tree strategy means that only one tree is displayed in full at any one time. As such, while this visualisation gives a good indication of overall statistical correlations of a set of trees, it does not allow for direct visual comparison of the internal structures of different trees. This visualisation can tell us by how much two trees or sets of trees differ, but not where in the internal structure of the trees the differences occur. Also, the data used is strictly concerned with the same taxa for each tree, whereas historical organisational classifications may have varying amounts of data, caused by addition of new data or by different levels of completeness when the classification was constructed.

4.5 Conclusion

To summarise, in this chapter, current formalisms and visualisations that represent, in some form, multiple trees have been detailed and explored. For each of these visualisations, a description of how they fall short of the problem requirements for a visualisation that specifically shows changes in structure between different hierarchies of the same node objects has been outlined. In general, of the techniques, the animation approach seems distinctly inappropriate, given that it is a medium best used to reflect gradual, evolutionary change, rather then complete structural re-organisations. The small multiples fare better for this type of information, but have the drawback of space constraints in the display, and the combining-overlaying approaches manage to integrate the information spatially and temporally, but the resulting mass of information tends to be difficult to visualise in an understandable or complete manner. The small multiple and combination approaches do however show promise for representing multiple, overlapping hierarchies and allowing appropriate task interaction. Therefore, with the need for an appropriate visualisation in botanical taxonomy having been previously established, and with no such visualisation currently existing, the need for the development of a novel visualisation is thus established.

5 Visualisation Development & Methodology up to Initial Prototypes

Having determined the taxonomists’ high-level requirements, along with the understanding of the underlying model that described these linked hierarchies, there was now the task of developing a visualisation through to fruition. This in turn meant deciding on a methodology for development of the visualisation. Documented life-cycle developments of IV’s that could guide a new IV development are rare, and specific design methodologies seem to be even rarer, as IV itself is still a relatively new area of research. Smith and Duke [159] have noted a similar dearth in the area of Virtual Environments (VE), a similarly fresh field of research; the experience being that methodologies tend to arrive in a field after it has reached a certain level of maturity. However, there is one case study described in Ellis, Rose and Plaisant [56] that stands out. Detailing the development of their visualisation, a proposed design evolved through a number of versions in co-operation with *representatives of the target user population*, with early testing relying on screen mock-ups for feedback, and formal user testing occurring later on with a fully interactive prototype. Their development of an IV-based interface had much in common with the iterative, explorative and co-operative development techniques employed for developing standard CLI and GUI interfaces [75] and produced an acceptable final result. Consequently, it was decided that the development of the multiple taxonomy visualisation could follow an iterative process through co-operation with the taxonomists without any foreseeable problems.

As such, the design and development of visualisation prototypes progressed through a number of iterative cycles, each iteration having a different emphasis. The first stage consisted of paper-based ideas that could be quickly constructed and revised, and once these were recognised as appropriate, the necessary approaches were developed into interactive prototypes. As such, the taxonomists were initially shown sketches of how their required tasks could be performed, or at least visualised, and later, in the second phase, interactive prototypes were developed. This led into cycles of testing, analysis and development of the prototypes, as shown in Figure 5.1, and these further iterations are discussed in detail in the next chapter.

Initial development of interactive prototypes would begin earlier in the process than it did with Ellis *et al* as, unlike the situation they had found, there was no need for consultation with and between many separate groups of people affected by the system development. Also, the system itself was not there to replace some existing system that users had become attached to. As stated previously, the multiple taxonomy visualisation was to tackle a problem for which there was no existing tool, and whose undertaking until now relied on cross-referencing paper documents manually, a simple example of which is shown in Figure 1.1. Usually however, taxonomists would have to collate separate documents and trace taxa across them, rather than having them conveniently laid out in columns on the same page. As such there was anticipation of such a tool, rather than any resistance to it. Furthermore, it became obvious that most of the tasks the taxonomists had envisaged would require interactive prototypes for demonstration and evaluation. Static diagrams could show some properties of the information and tasks, but many ideas, especially with regard to interaction, were simply better served by quickly setting up narrow, vertical prototypes that could demonstrate their impact.

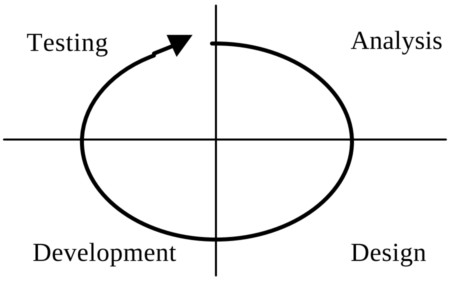


Figure 5.1. Iterative prototyping cycle for the multiple hierarchy IV.

At each stage of prototyping, different functionality and usability aspects of the system would be developed and tested, making the approach similar to the spiral family of design models. As Muñoz [126] states, both functionality and usability should be involved and refined during prototyping. It should not be something that is performed solely on the user interface aspects of a system. This becomes increasingly important with IV systems, where the visual display is more complex and dependent on the underlying data and functionality than a general GUI.

For the first test, the aim would be to see if the functionality requested by the users in their requirements was present and working. Also some usability testing was included to study whether the prototype visualisations could easily communicate the outcomes of these functions on an example data set. At this stage, a decision on which prototype to proceed with from a number of possibilities based on differing metaphors was required. Therefore the users were also asked questions on how they regarded each prototype.

The second test looked at the further and revised functionality that had been incorporated into the system, and also to consider wider usability issues relating to the interface as a whole, not just with regard to the visualisation output. By this stage it was expected that the functionality would be complete apart from minor tweaking. This testing should also catch major usability flaws in the visualisation and interface.

The third set of tests was therefore dedicated almost solely to usability issues. The aims were to validate the removal of the major usability flaws found in the second test, capture most of the remaining minor flaws, and also to capture any new flaws introduced by re-working of the interface or visualisation.

The final set of tests consisted of a more statistically rigorous approach, to obtain error rate values for the visualisation, and a statistically valid subjective score for the visualisation. Past this stage the system would be taken into different information domains to explore its generality past taxonomic classifications.

As can be seen, the testing methodology moves through the iterations from validating functionality concerns towards testing usability concerns. The usability testing methodology for the first three stages followed the approach of discount usability engineering proposed by Nielsen [132]. This technique was selected, as discount usability engineering is easy to use, rapid to obtain results from, and is less intimidating than full and formal testing procedures for the users who are being monitored and for the evaluators who are using it. Qualitative issues were the main concern at the start of the testing cycle, so only a few representative users were needed as per Nielsen’s suggestions, as no statistically valid quantitative data needed to be gathered. In short, at the initial prototype stage the concern was to recognise whether the taxonomists could perform tasks with the visualisation, rather than how quickly or how accurately they performed them.

This chapter discusses the development of the prototypes up to, but not including, the first user test. The subsequent testing, revision and development of the prototypes up to their final form is detailed in Chapter 6.

5.1 Initial Sketches for Visualisation of Multiple Overlapping Hierarchies

Initial talks with the taxonomists had yielded an understanding of their field of study and their particular problem of comparing multiple overlapping classifications. Furthermore, they had also elicited the tasks a suitable tool should be able to perform to examine such data sets. From reflections on these requirements and the previous work on multiple tree visualisations (Chapter 4), a number of initial sketches were produced in preparation for constructing a suitable visualisation. As well as giving an initial indication of what a possible visualisation could resemble, these sketches crystallised and confirmed the validity of the tasks that they had requested to carry out.

As mentioned in Chapter 2, display techniques for hierarchies concentrate on issues such as increasing the number of nodes on screen and clarity of layout. These needed to be considered during the development of a suitable visualisation and, as discussed in Chapter 4, multiple trees also require an extra notional dimension to be displayed. The initial sketches attempted this by using space in a small multiple-style layout. The resulting sketches were, in effect, a stylised representation of the underlying restricted graph structure. The sketches could possibly have imitated animation by drawing each hierarchy on a separate piece of paper and viewing them flipbook-style, but animation's unsuitability for allowing users to reconcile sudden changes meant this option was disregarded.

As noted previously, interaction techniques at early stages of development are difficult to visualise, as they require interactive prototypes for a full demonstration of their effects. However, an attempt to show interaction through static sketches is shown in Figure 5.2 - Figure 5.5, where various filtering effects on the trees are shown. They at least give an indication of the visual effects of such mechanisms at work.

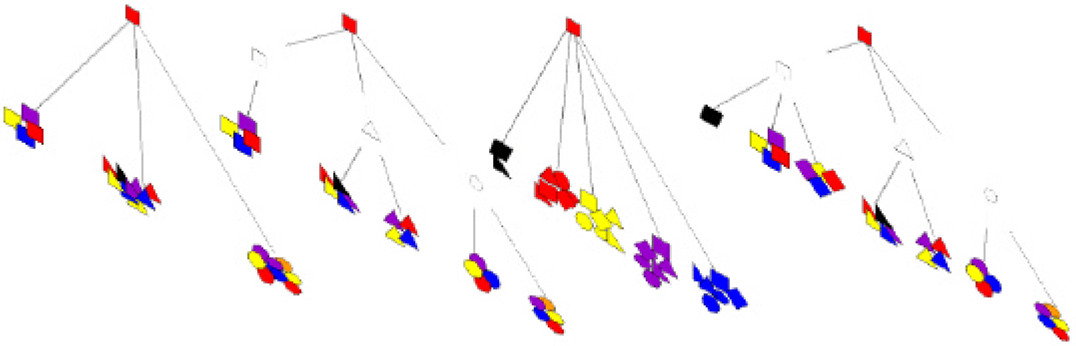


Figure 5.2. Filtering of intermediate levels in hierarchy

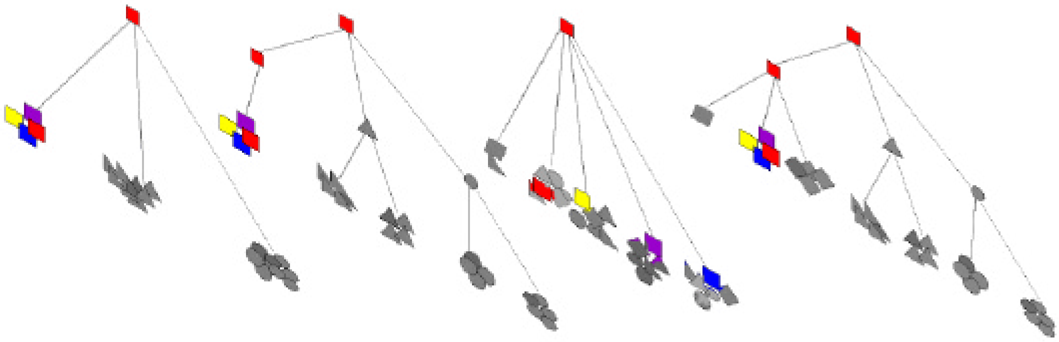


Figure 5.3. Tracking of a sub-tree through hierarchies

For example, in Figure 5.2, the intermediate level used in two of the classifications is faded out to allow easier visual comparison of the levels common to all hierarchies. This is a representation of the requirement to compare hierarchies according only to shared levels or ranks. In Figure 5.3, all nodes apart from the squares have their colour bleached, an example of visual filtering, which allows the user to attend more easily to a sub-group (the squares) of particular interest.

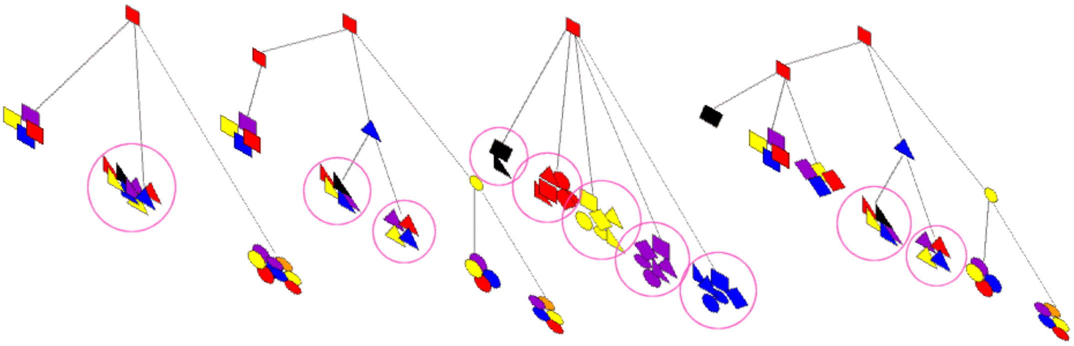


Figure 5.4. Highlighting of all sub trees that contain triangles

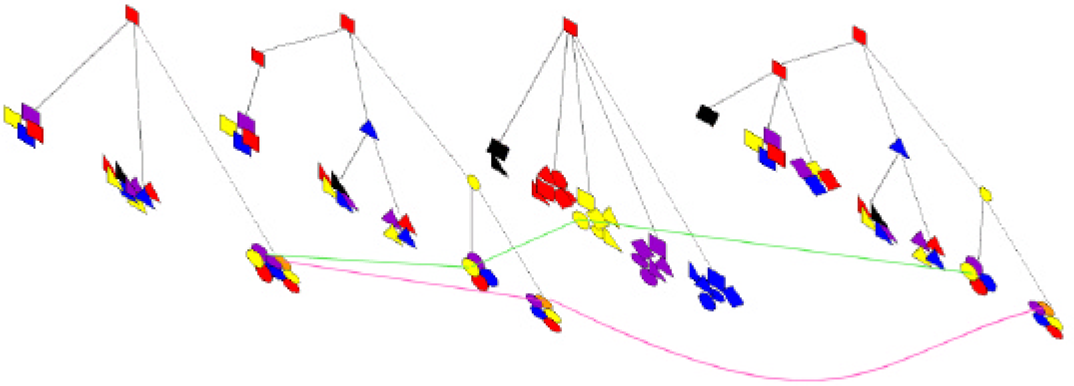


Figure 5.5. Tracking of individual shapes across hierarchies

Further, in Figure 5.4, a simple highlighting technique shows all the sub-trees that triangles appear in. This has a similar effect to Figure 5.3 as it draws the user's attention to the location of the specimens of interest. For taxonomists these visualisations would help determine the method used to formulate the hierarchies. Different methodologies would result in differing patterns of distribution, as can be seen from Figure 5.4's indication that triangles appear in all of the third hierarchy's sub-trees. A focusing or filtering technique could act on the encircled groups, emphasising them even more.

Another method to visualise a node's progress through a number of hierarchies would be to draw a path that linked its separate representations. Other visual cues of lesser prominence for the sibling, child and parent nodes would also show their journeys through the multiple classifications. This would enable a viewer to see the context change of the various hierarchies, one of the requirements outlined in Chapter 3. An example of this is shown in Figure 5.5, tracking the orange node and a sibling, the yellow circle. It also highlights missing information as one of the nodes is not represented in the third classification, and so that nodes' path skips this hierarchy.

After examining these representations, the taxonomists agreed that they reflected the type of situations and information they were hoping a visualisation could bring out, and while the sketches may not be identical to subsequent prototypes they gave a clear, visual representation of the requirements.

5.2 Interactive Prototypes

After the consultations with the taxonomists to realise and confirm their requirements, work began on prototyping visualisations that could achieve these requirements and overcome the drawbacks that occurred with the previous visualisations of multiple trees described in Chapter 4. What follows is a description of the prototypes and the ideas behind their application along with the problems encountered in their execution.

Two initial prototypes were programmed in Java 1.2 (now known as Java 2) using the Kawa Java environment. Java was chosen as the language to build the prototypes for two main reasons. Firstly, any prototypes built can be shown on the World Wide Web as an applet to a wide, critical population to gain general feedback. Huang's visualisations of very large graphs [89] and Inxight's hyperbolic tree visualisation [92] have on-line demonstrations of this nature. Secondly, novel visualisations' graphical interfaces tend not to be composed of standard UI objects, for example, instead of scroll-bars and text windows IVs have objects such as spring-mass models and fisheye lenses. Resultantly, such objects are resistant to rapid prototyping environments, and trying to implement or mimic such features is often harder than coding them in a complete programming language such as Java in the first instance. Muñoz [126] reinforces this point, stating that 'high-fidelity' prototyping tools and environments rarely allow for data visualisation or interaction concerns. As the 'lo-fidelity' prototypes such as sketches and storyboards cannot show interactive elements easily, this leaves the development of custom prototypes as the way forward.

The first, graph-based, prototype was a direct visualisation of the underlying DAMG super-structure of the multiple taxonomies, and was constructed as it could display all the information within one distinct visual structure. The second, set-based, prototype utilised a small multiple approach in which each hierarchy was displayed separately. This set visualisation evolved as it was surmised that keeping the familiar tree layout of each classification could match a users' mental model of the information more closely.

The UI architectural model used as a framework for the prototypes was the M(VC) model described in Chapter 2.6. The M(VC) model has succeeded in separating presentational and functional concerns when applied to traditional GUI's, and as the chosen application language, Java 2, has based its graphics and UI libraries on the MVC model, design and refinement of the visualisations are also based on the same architecture.

In practice, it was convenient to separate the Model into two parts; one consisting of the static underlying tree models, and the other part consisting of model meta-data such as selection and activity states that the views, and indeed other models, could access, affect and share. In this way, transient information relating to the base model but used by the views was separated out from both considerations. Therefore, mechanisms such as a new selection model could be slotted in without affecting the views or the underlying, static model.

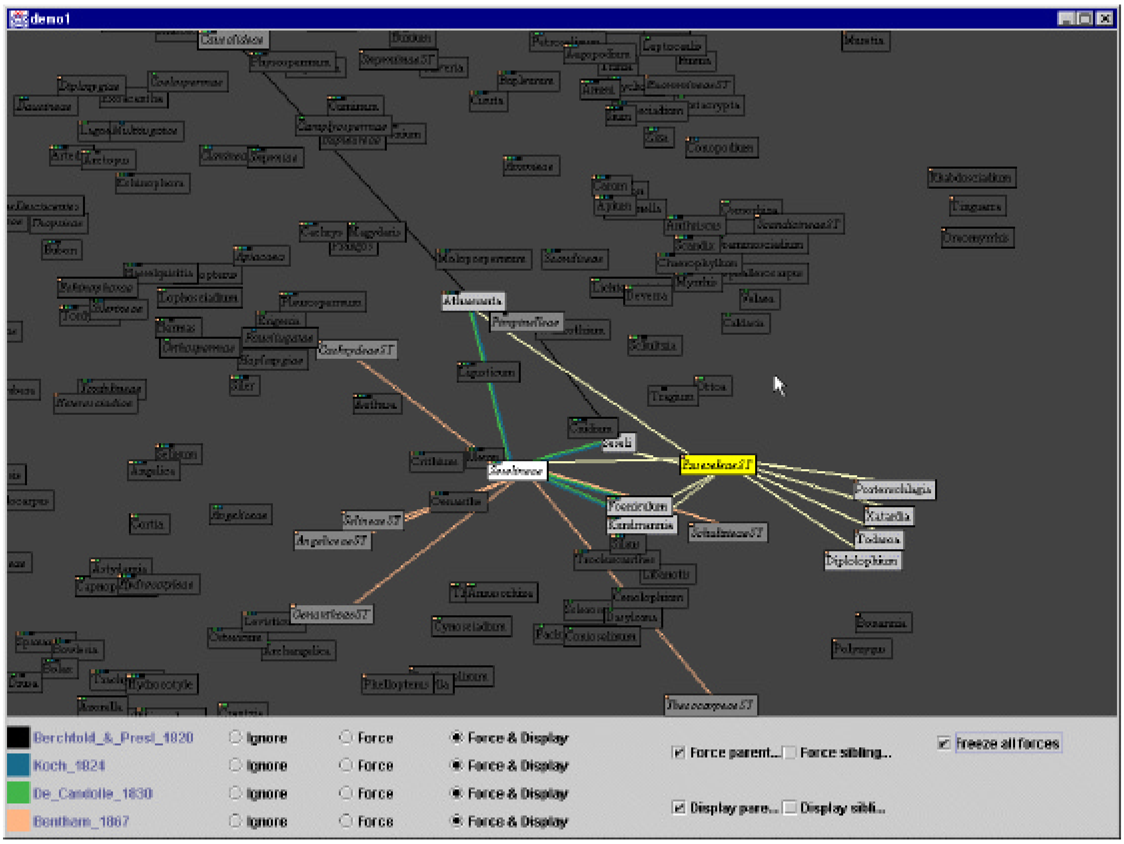
The MVC nature of the visualisations progressed steadily with time rather then being structured as so entirely from the start. This was because the aim of prototyping is to make a representative system available quickly, and implementing according to a UI architecture takes time.

5.3 Design of a Graph-based Prototype

The first prototype developed was a direct visualisation of the underlying logical model, the Directed Acyclic MultiGraph (DAMG), and is shown in Figure 5.6. Rather than visualising the hierarchies separately, this prototype kept them together in the overall graph structure, and this structure would then be visualised using Fruchterman and Reingold's spring-mass model [62] as described in Chapter 2.4.2.1.

The approach appears to go against common wisdom in that devising visualisations of network and graph structures is much harder than producing good visualisations of individual hierarchies. Indeed, Mukherjea *et al* [125] take the approach of deconstructing a graph structure, in their case the hypermedia structures of the WWW, into a number of different hierarchical structures, each of which can be viewed separately. In defence of this prototypes' methodology, Mukherjea *et al* were focusing on a task that involved seeing hierarchical organisation within a graph. Conversely, the taxonomists wish to see how multiple hierarchies correlate with each other. In effect they want to know how simpler structures interact to produce a more abstract structure, the reciprocal of the task Mukherjea *et al* were concerned with. It is the users' tasks after all that should be the overriding concern in forming a visualisation.

The visualisation of the DAG is displayed via a spring-mass model that moves nodes towards appropriate positions. Unlike most spring-mass model based systems, these positions are not final and are constantly re-calculated by the prototype, as in Donath's [51] system. This is because the prototype allows the user to toggle on and off the display of individual hierarchies, and the spring-mass model will re-adjust itself to include only the hierarchies that are to be displayed in its calculations. Showing the changing node positions as the spring model updates itself introduces an intrinsic animation effect into the visualisation, which should allow the user to more easily reconcile changes in layout.



■ Berchtold_&_Presl_1820 ○ Ignore ○ Force ● Force & Display
🟦 Koch_1824 ○ Ignore ○ Force ● Force & Display
🟩 De_Candolle_1830 ○ Ignore ○ Force ● Force & Display
🟧 Bentham_1867 ○ Ignore ○ Force ● Force & Display

☑ Force parent... ☐ Force sibling...
☑ Display pare... ☐ Display sibli...
☑ Freeze all forces

Figure 5.6. Graph-based visualisation prototype

The visualisation displays the links between the nodes using a colour coding designed to differentiate the links belonging to each separate hierarchy that forms the graph. The colours are from a linear colour scale, see Levkowitz [113], with the colours chosen forming an even distribution along the linearised scale. Each name node is displayed as a labelled rectangle, coloured along a grey-scale according to its depth from the taxonomic root. This can be done, despite the notion that nodes might be at differing depths in different hierarchies, due to the fact that botanical taxonomy enforces a strict organisation on the nodes it organises. Any particular nodes' depth is defined by its position in the taxonomic structure of families, species, genera etc, rather than its depth in any selected hierarchy.

These particular mappings of visual attributes to information were implemented because colour scales and grey scales differ in human perception. Grey scales are perceived as being ordinal, having some quantitative meaning, and colour scales are generally seen as being nominal, only indicating membership or some other qualitative meaning [46]. Therefore, colour was chosen to indicate membership of a particular tree, a qualitative attribute, and the grey scale to show distance from the root, a metric that can be quantified.

At the time of the first user tests, the users' interaction entailed adding or removing particular hierarchies from the graph, along with panning and zooming controls to gain an overview or a close-up of a particular area of the graph. The user also has the ability to view only one node and its relations (siblings, children etc), thus filtering out the other nodes and links. Nodes can also be picked up and moved manually to partially overcome any occlusion problems, and the same mechanism allows nodes to be 'plucked' or shaken by the mouse pointer, upon which related nodes will also move, attached as they are by links to the selected node.

The taxonomist's requirements were fulfilled by the graph visualisation with varying degrees of success. Comparing overall trees was easily achieved by enabling only the particular trees that users wished to compare. After waiting for the visualisation to settle, similarities and differences could be seen by observing grouping effects and the different link structures. Close, parallel links would indicate the same relationship being present across different classifications, and close grouping of nodes and the convergent ray of links that emanated from them towards a parent node signifies groups that share common parents in one classification. Also, if a group then had a common parent in another classification, that group of nodes would tend to cluster together more tightly and separate out from surrounding nodes, as in the example in Figure 5.7. Another set of links would converge towards this parent.

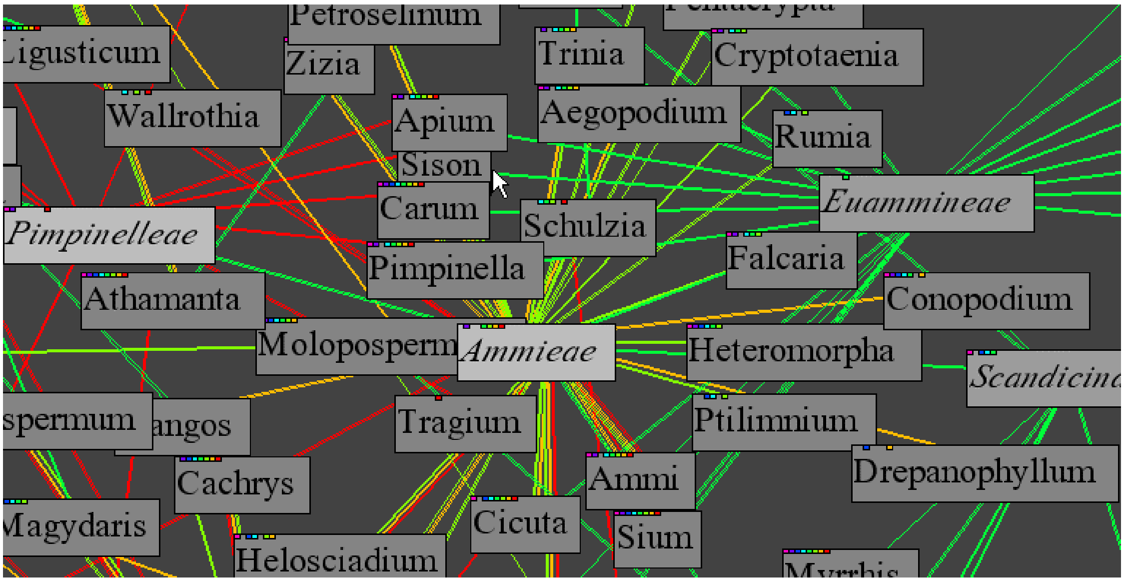


Figure 5.7. *Apium*, *Sison*, *Carum*, and *Pimpinella* share the common parents *Pimipinelleae*, *Ammieae* and *Euammineae*

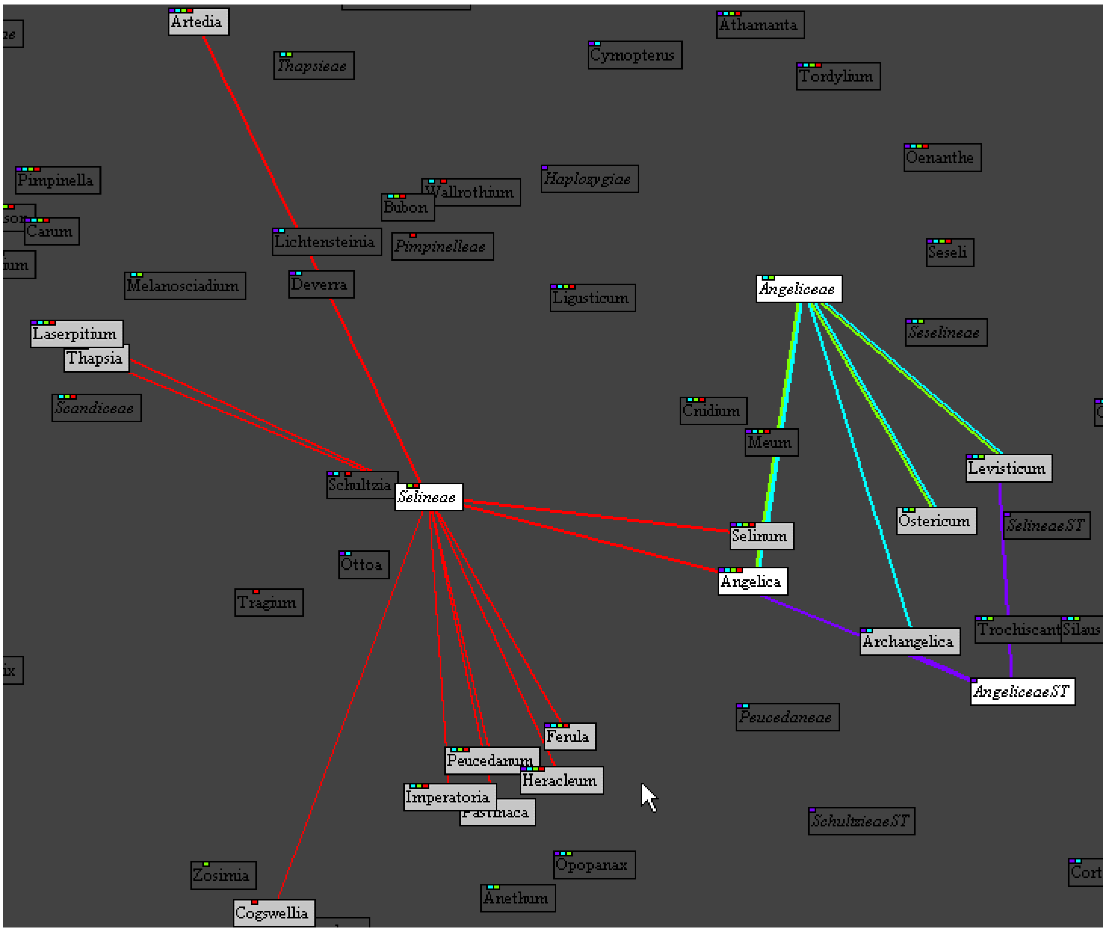


Figure 5.8. The relationships of *Angelica* across multiple classifications.

Selecting a node, which filters out other unassociated nodes and links, reveals its particular close relationships with other nodes. Figure 5.6 shows the selection of the *EuseseleaeST* sub-tribe, with its parent highlighted in white, children in grey, and siblings in darker grey. Figure

5.8 shows the same situation for the genus *Angelica*, which has several parents and sets of siblings.

A distinct drawback with the graph visualisation is the absence of any spatial cue that informs the user of a node's depth within the hierarchy. As stated, the visualisation does use a grey scale to shade nodes according to their depth within the overall hierarchy, but the work on visual perception mentioned in Chapter 2.3 shows this is a much less powerful perceptual cue than grouping by positioning. This is unfortunate as level, or rank, identification was one of the taxonomists' requirements. Consequently, a version of the graph visualisation was built that restricted node positioning to concentric bands around the root node, each band corresponding to a unique level of depth within the taxonomic hierarchy.

Unfortunately, the resulting visualisation was extremely disappointing for a number of reasons. Firstly, the nodes bunched and pulled together into a rather narrow sector of the circle if all the forces of the displayed trees were applied. Thus it was decided that only one chosen tree could enforce its links at any one time, and in the screenshot of Figure 5.9 this is the purple hierarchy (Force 3 radio button). Essentially this meant that one tree was chosen as a base tree in the spring-mass calculations, upon which other trees had their links superimposed at the correct positions but had no participation in the force calculations.

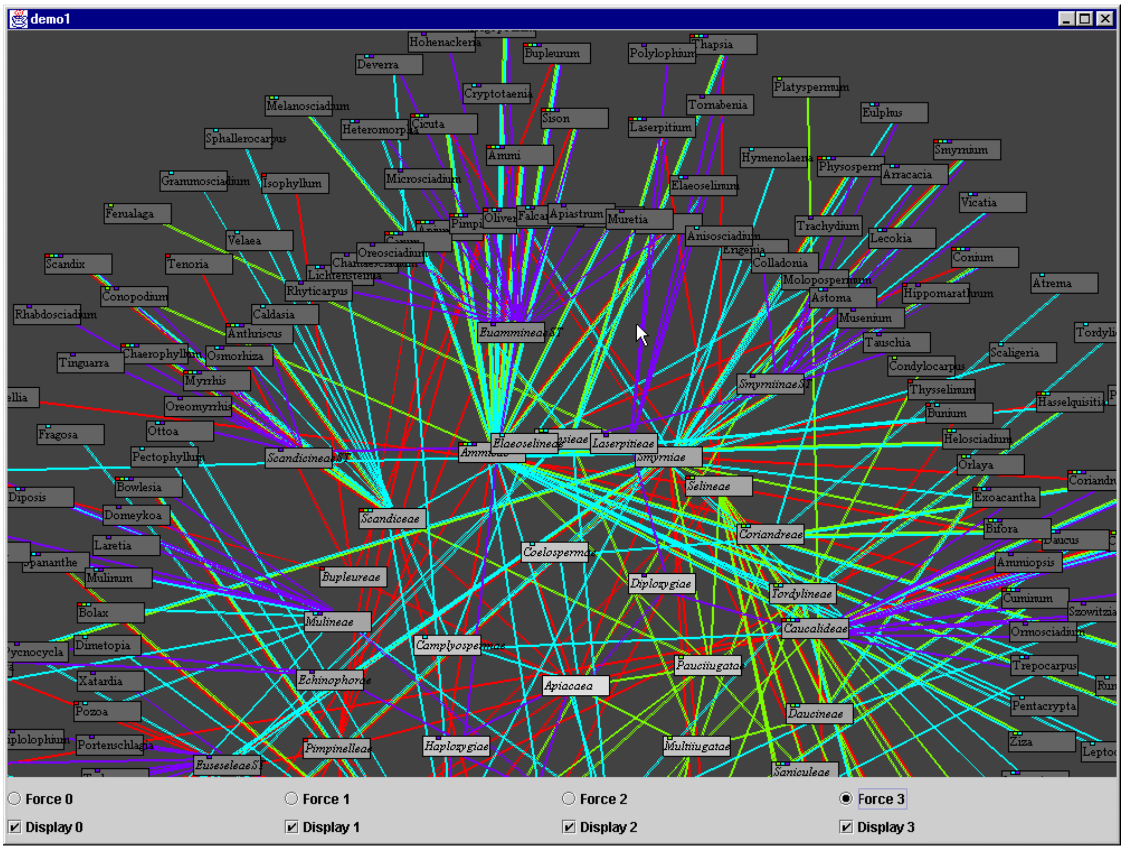


Figure 5.9. Enforcing a rank structure on the graph caused more problems than it solved.

Secondly, restricting the nodes to the concentric bands increased the displayed node density in these areas, introducing occlusion problems even at high zoom factors. Even though each node band had an area that was proportionally larger than its neighbouring, inner band, the number of nodes at each depth increased, as tree structures tend to do, at an almost geometric rate.

Thirdly, restricting the movement of nodes to an essentially one-dimensional path along a particular circumference (albeit with a small amount of leeway either side), dramatically increased the likelihood of nodes being trapped in local minima caused by the inter-node repulsion. A solution to this was to annul these inter-node forces when switching to a new base hierarchy for spring-mass calculations within the graph, and then slowly build the repulsion factor back up; thus giving nodes a chance to get to their desired positions according to the new effects of the attractive link forces. This worked to a degree, but did not completely solve the situation. Widening the concentric bands would give the nodes more freedom to organise themselves, but at the same time it would reduce the gaps between bands that delineate the hierarchical structure in the first place.

Finally, and partly as a result of the first and third points just mentioned, edge crossings in the visualisation increased, making links hard to follow. Combine this situation with the density and occlusion problems outlined in the second point and the result is a visualisation that is hierarchically laid out, but in which it is almost impossible to comprehend any overall structure, as can be seen in Figure 5.9. It is interesting to note that Melançon and Herman's barycentric DAG visualisations [121] suffer from these same problems.

The graph approach also has disadvantages in terms of human factors and algorithmic complexity. The layout of the spring-mass model will tend not to always generate the same layout for the same data. As with other iterative methods, any tiny disturbances in the initial or previous states, either by the user or introduced from other sources such as rounding errors, will snowball in effect as the layout algorithm progresses (a chaotic effect). Also Misue *et al* [123] claim that methods designed primarily for layout *creation*, such as the spring-mass model, are not always the most suitable methodologies for layout *update*. The need, and thus ability, for users to absorb and process changes in layout rather than just comprehending an initial layout may ultimately require a different style of layout mechanism.

Another problem with this approach is the amount of time needed to recalculate and redisplay the nodes, especially in Java, designed as it is for portability and reusability, not for speed. As such, in its original state the prototype approached a limit of showing and updating roughly 250 node positions at a rate of 4/5 refreshes per second, though this of course would vary with processor speed, Java version etc. This latter problem of calculation speed was caused primarily by the calculation of inter-node *repulsion* forces in the spring-mass model. This algorithm, as with other unrefined N-body algorithms, is of $O(N^2)$ complexity, so to reduce this it was decided to attempt to integrate Chalmers' linear time layout algorithm [36], which is of order O(N), for this part of the computation. It is important to note that the *attractive* forces resulting from links are calculated as before. This is because in a tree of *X* nodes, each node except the root has one link to its parent node, meaning there are *X-1* links overall in the tree. Therefore this part of the force calculation is already of O(N) complexity, and even with multiple trees it only scales to O(*k*N), where *k* is the number of trees.

Chalmers' algorithm works by defining two sets of nodes, a neighbour set and a random set, for every node. The random set is composed of a new random selection of nodes on each iteration, and the neighbour set consists of the closest logical neighbours of a node, but is initially empty. All random and neighbour set nodes then induce their combined effects on the target node. If a random node is a closer logical neighbour than any neighbour set node, it is pushed into the neighbour set to replace that farther node. This action gradually builds up and refines a neighbour set of nodes close to the target node as measured by logical graph distance.

The initial approach was not as successful as hoped for two reasons, the first being that the graphical update on-screen took a considerable amount of time, and the algorithm could not reduce this. Secondly, the algorithm was designed to move objects rapidly to their final positions, upon which the objects would then be visualised. The graph prototype here continuously displays the positions of nodes during updates, and as Chalmers' algorithm uses a degree of stochastic sampling, this introduced an unacceptable amount of visual jittering as changes were made in the spring-mass model. This situation is in fact an example of the different requirements of layout update against layout creation discussed by Misue *et al.* The nodes are also slow to move apart, as the neighbour sets, which due to their closeness should invoke stronger repulsive forces than the random sets, require some time to be constructed.

To overcome some of these difficulties, Chalmers' algorithm was adapted so that it was aware of the link-based nature of the structure; the algorithm originally being aimed at multi-dimensional information sets with no pre-defined structure, and as such Chalmers' algorithm starts off with empty neighbour and random sets. The multiple-tree graph has an intrinsic structure given by the links, and in other force-directed drawings of trees it is mostly siblings that occlude any given node. Therefore, the starting conditions were altered so that the neighbour sets didn't start off empty, but contained a random sampling of sibling nodes for each node to start with. Secondly, to prevent jitter, random nodes can replace a member of the neighbour set if closer, but do not have a direct influence. This however, increases the chances of a node getting stuck in a local minimum. Also, if a randomly selected node has the same logical distance but is closer geometrically it could now replace a node in the neighbour set.

This was allowed, as logical distances in graphs are much more discrete than in high-dimensional data sets i.e. siblings of one node will all have the exact same logical distance, and there may be a hundred siblings for a particular node.

The end result of these alterations was that nodes moved apart faster initially, but there was now ceaseless, though smooth, movement rather than the original jitter. This movement was due to a situation in which, say, Node A was a member of Node B's neighbour set. Node A would then be repelled, perhaps further than another node, Node C, currently not a member of Node B's neighbour set. Eventually, due to random selection, Node C would replace Node A in B's neighbour set. Node A would now not be repelled by Node B, but Node C would, eventually passing by Node A again, enabling A and C to substitute one another again in B's neighbour set and the cycle would repeat. The situation becomes chaotically complex with hundreds of nodes.

This phenomenon was the result of a change to the original algorithm; allowing nodes at the same logical distance to replace each other in neighbour sets when the non-member was at more risk of causing an occlusion. Removing this condition would lead to nodes being occluded by non-member siblings, as many nodes had far more siblings than available slots in their neighbour sets. In short, some problematical emergent layout features had to be accepted along with the efficiency of the sampling-based algorithm.

In summary, the main advantage of the original graph prototype is the integration of the many hierarchies spatially and temporally, avoiding the drawbacks and problems described in Chapter 4 with the animated and small multiple approaches. However, the approach also has inherent problems due to the nature of the spring-mass algorithm and the resulting mass of displayed information. In short, the notion of seeing all the hierarchies at once in a single visualisation can be a double-edged sword.

5.4 Design of a Set-based Prototype

The second prototype was influenced by a number of factors that emerged from background reading and the initial implementation of the graph prototype (though it was mostly developed concurrently with that prototype). These included such issues as the speed problems of Java and the visual clutter caused by merging all the hierarchies into one visual structure.

When visualising an organisational hierarchy, a designer is able to draw on a strong delineation between leaf nodes and non-leaf (internal) nodes. Essentially in such a *classification* hierarchy, leaf nodes are objects of some type, and internal nodes are categories of varying abstraction used to impose an organisation on the leaf nodes. For example, a file directory consists of files as leaf nodes and directories as internal nodes, which corral the files into sets and subsets. Hence, this visualisation prototype moves away from the node-and-link metaphor and towards a set-based visualisation. This could be more productive as Parunak [140] states that taxonomic reasoning, the categorisation of objects, is essentially set-based, and thus the users' mental model of such a process is inclined towards a set-based model, rather then a node-link system. A visualisation for aiding such a task should therefore benefit from mirroring a set-based metaphor on-screen.

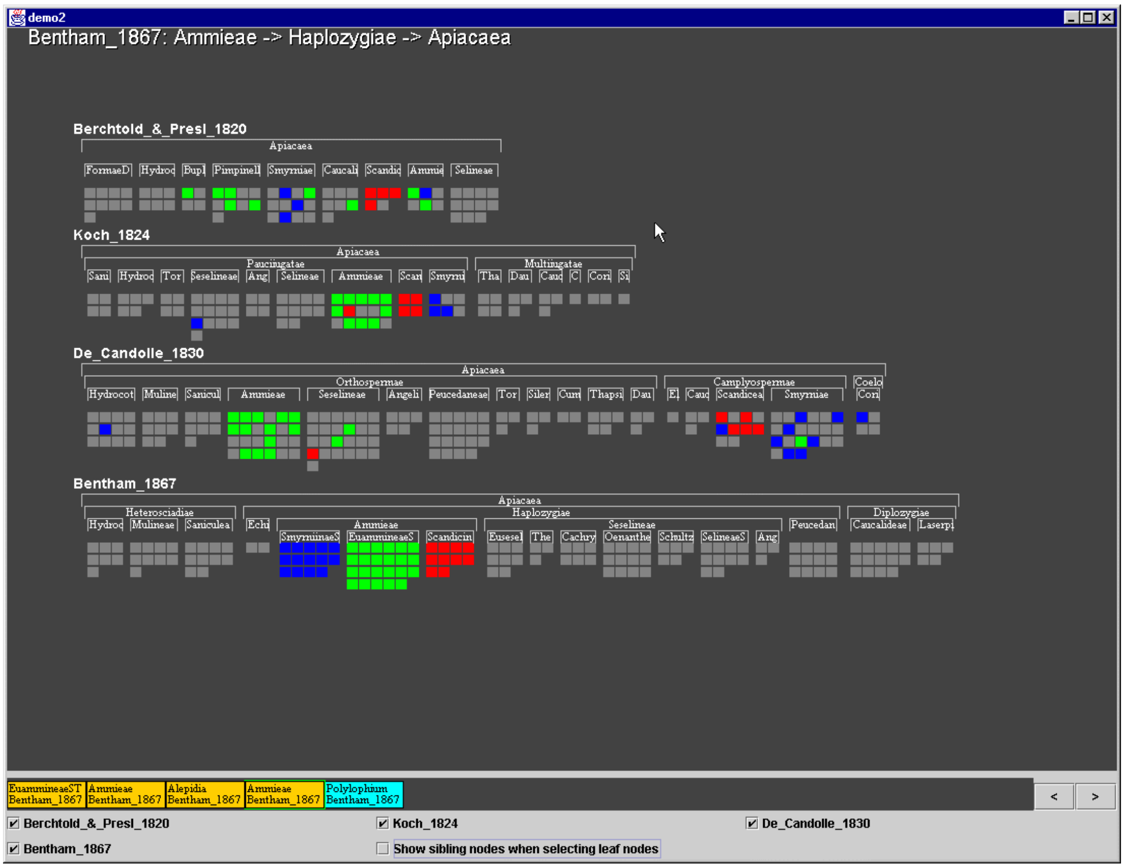
Andrews' Information Pyramids [6] is an example of a visualisation that distinguishes leaf and internal nodes visually in a 2.5D Treemap-style visualisation. Other hierarchy visualisations, such as Cone Trees [150], do not distinguish between the two types of nodes even when dealing with structures such as files and directories. Alternatively, trees that show navigation routes such as Huangs' web-browsing visualisation [89] have no such organisational distinction to draw on. Leaf nodes indicate the same type of object as internal nodes, the difference being that no further navigation can be or was made from the leaf nodes.

Parunak also contended that people, in general, preferred to combine many objects of simpler topologies than work with one object of complex topology. His words were targeted at hyperbases, essentially databases with hypertext properties, of which he wrote:

"The insight for hypermedia is that a hyperbase structured as a set of distinguishable hierarchies will offer navigational and other cognitive benefits that an equally complex system of undifferentiated links does not, even if the union of all the hierarchies is not itself hierarchical." [139]

The taxonomic hierarchies form this type of hyper-structure, and have the benefit of being hierarchical even when unified in the overall structure, not in the sense of being tree-like, but in that the taxonomic rank mechanism enforces an ordering down the unified structure. It can be argued from a strictly taxonomic standpoint that the objects being re-categorised in taxonomic hierarchies are families or genera, and hence abstract categorisations themselves, the physical objects being the instances of plant specimens. However, if it is these categories that are being re-organised at a higher level, remaining constant across the different hierarchies, they can be regarded as indivisible sets of objects to be categorised, and not as categories themselves.

As a result the second prototype was developed as a set of linked small multiples with one distinct representation per tree, accepting the restrictions of this approach that have been previously stated in Chapter 4. To reduce screen space problems, the leaf nodes of each bottom level category are arranged in a grid formation, as opposed to the normal style of a linear layout for each level of a hierarchy. This reduces the chances of individual hierarchies' displays spreading horizontally across the screen width and exceeding the overall display dimensions. The leaf nodes, drawn as small squares, were given a different representation to the internal nodes, which were each drawn as elongated brackets of such a length as to encompass all their sub-groups, as can be seen in Figure 5.10.



Bentham_1867: Ammieae -> Haplozygiae -> Apiacaea

☑ Berchtold_&_Presl_1820
☑ Koch_1824
☑ De_Candolle_1830
☑ Bentham_1867
☐ Show sibling nodes when selecting leaf nodes

Figure 5.10. The set-based visualisation prototype.

This prototype also avoids the visual clutter of the first prototype, though at the cost of losing a level of detail. The leaf nodes are not individually labelled, and only one set of distinct relationships between the hierarchies can be seen at any one time. However, it also circumvents the speed restrictions of the graph-based prototype, as the visualisation and internal model are not updated continuously, only when the user initiates interaction. Keeping the representations as distinct trees avoids the perceptual problems of following link-crossings as in a graph-based visualisation. Indeed, this visualisation has no displayed links of any type, membership of a group being represented by having a node positioned underneath one of the bracketing group nodes. Discerning the parent or even the path to the root from any particular node in a classification is then merely a matter of travelling straight upwards from that node and observing which group nodes such a route intersects. In this, the set-based visualisation has much in common with the hybrid tree visualisations mentioned in Chapter 2.4.1, which combine space-efficiency with the ability to display structure in a simple and understandable manner.

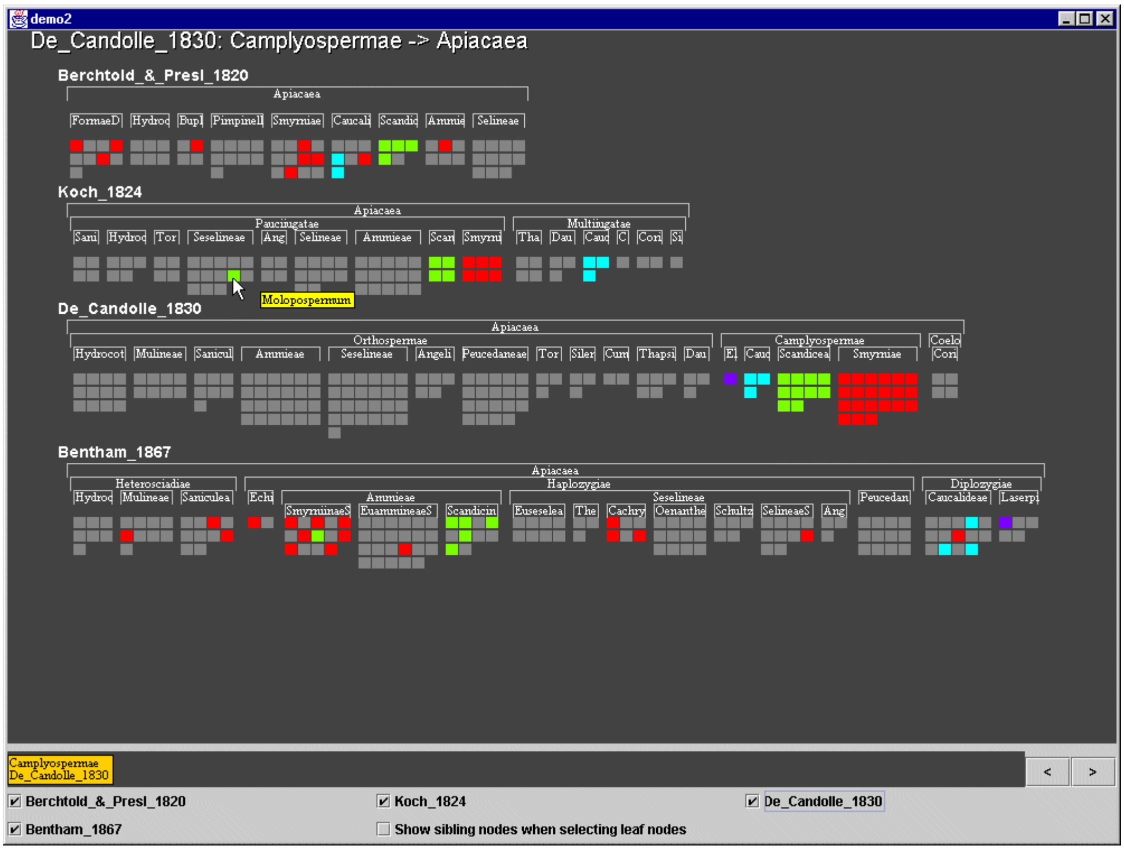
Interaction is performed using the linking metaphor. A selection of a particular node, particular category or set of categories in one hierarchy is shown in the context of the others, by the simple use of colouring all the selected nodes' representations where they occur throughout the classifications. Each particular category or leaf node highlighted is given a separate colour and its distribution throughout the other hierarchies can then be seen, enabling comparisons of correlations to be made across the set of hierarchies. In Figure 5.10 it can be seen that when 'Ammieae' is selected in the bottom hierarchy (Bentham 1867) its three sub-groups are coloured differently, and these nodes are giving matching colours where they occur in the other classifications. This is believed to be a novel application of linking techniques, as it is applied across a set of visualised hierarchies, not scatterplots as in Becker and Cleveland's work [15], or two differing stylistic representations of the same hierarchy as performed by Fua *et al* [64].

With regards to the taxonomists' requirements, the correlations, or similarities of the overall tree structures, can be seen immediately by visual inspection. Cohesion of smaller groups across the multiple classifications can be discerned by selecting the parent node of that group and viewing the subsequent distribution of coloured representations across the classifications, as in Figure 5.10. Similarly, individual nodes can be selected and their distributions observed. A sibling function allows the siblings of a particular node to be viewed and coloured on a per classification basis. In this manner, a taxon, which keeps mostly the same siblings throughout the classifications, will produce a visualisation where there is one predominant colour, and any fresh siblings that occur in further hierarchies are given different colours. In contrast, a taxon that moves around and has many groups of almost mutually exclusive siblings in each classification will produce a visualisation with large amounts of different colours. The number of levels, or ranks, in individual classifications can be seen by a glance at the visualisation, a distinct advantage over the graph visualisation.

Previous selections are shown in a history bar along the bottom of the screen. It was thought at first to represent past screen shots as miniature versions of the visualisation, as described by Nielsen [130], but the space needed to show such representations at a level of detail where clear differences could be seen would have been too great. Nielsen's idea was originally applied to web pages or documents in which the differing size and shape of text areas on screen provided the miniatures. In the set-based visualisation, the basic layout of the data was always consistent, with selections resulting in changes of colour rather than layout. Instead, historical selections were represented by small boxes containing the names of both the node and classification involved in the selection. Selecting one of these labelled boxes would take the visualisation to the state that clicking on the node in that classification would provoke.

5.5 Set-based Visualisation Example

An example of how the set-based prototype operates is now described. In the screenshot of Figure 5.11, the family known as Camplyospermae has been selected in the third hierarchy down (De Candolle 1830), to view the distribution of its' component genera throughout the other hierarchies. This has shown a pattern of distribution that indicates the correlation between the second and third taxonomies (Koch and De Candolle) is quite strong, with only one node from De Candolle's taxonomy being split. This is the green-coloured node Molopospermum that is being investigated by the mouse pointer, a member of Scandiceae in De Candolle's classification. In reality as Koch's is the earlier classification, the visualisation shows that this particular node has been taken from one part of the hierarchy and grouped with the other green nodes in De Candolle's classification. Looking at the distribution of the green Scandiceae nodes in the other hierarchies, it shows again a strong grouping across the hierarchies, with only one green node in Bentham's classification being grouped dissimilarly.



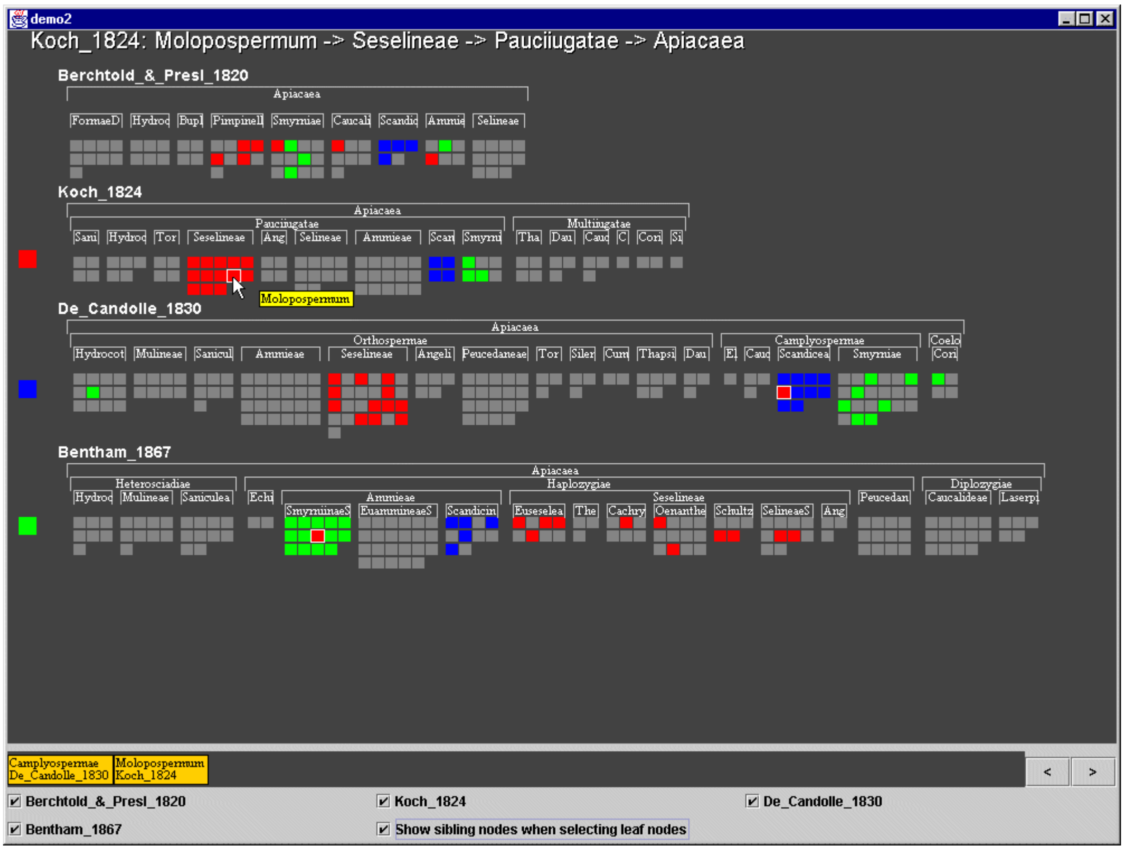
De_Candolle_1830: Camplyospermae -> Apiacaea

☑ Berchtold_&_Presl_1820 ☑ Koch_1824 ☑ De_Candolle_1830

☑ Bentham_1867 ☐ Show sibling nodes when selecting leaf nodes

Figure 5.11. The genera of De Candolle's *Camplyospermae* sub-family shown across other classifications

Further investigation of the rogue node Molopospermum is warranted. The option to show all sibling nodes of this node is activated and the node selected within the Koch classification. The resulting visualisation is shown in the next screenshot, Figure 5.12. The screenshot shows that the red nodes that are present in Koch's classifications, and that form Molopospermum's siblings in this classification, are also classified together in De Candolle's classification, all with the exception of Molopospermum which is ringed by a white border. De Candolle has chosen to group it with a new family, whose distribution across the hierarchy is indicated by the blue nodes. In Bentham's classification Molopospermum has been taken out from the blue nodes and grouped with a new family, indicated by the set of green nodes, and this is the third completely different group it has been associated with in as many classifications; no red or green nodes from the previous hierarchies have been grouped with it in Bentham's taxonomy, it is completely isolated from them. The screenshot also shows that Molopospermum was the detached green node in Bentham's classification in Figure 5.11. Together with the fact that with the exception of Molopospermum the sibling groups tend to stick together well over the classifications, this gives an indication that Molopospermum could be a difficult node to classify, perhaps exhibiting many common features or perhaps having none at all.


Koch_1824: Molopospermum -> Seselineae -> Pauciiugatae -> Apiacaea

☑ Berchtold_&_Presl_1820 ☑ Koch_1824 ☑ De_Candolle_1830

☑ Bentham_1867 ☑ Show sibling nodes when selecting leaf nodes

Figure 5.12. Display of siblings of *Molopospermum* across all classifications

5.6 Conclusion

In this chapter an approach for deriving and testing an original IV-oriented interface based on the common iterative design and development cycle was presented. Each stage was designed to use appropriate techniques to test differing aspects of the interface, such as initial functionality and usability, the bulk of which will be discussed in the next chapter.

From the initial requirements that were gathered, visual representations were produced for the purpose of confirming the requirements and as a starting point for visualisation prototyping. The development of the two initial prototypes in turn followed on from the user requirements and underlying data structure described in the previous chapter, and from the initial sketches of simple task scenarios. One visualisation prototype, the graph-style interface, reflected this underlying structure directly, whilst the set-based prototype displayed individual hierarchies separately, using the data structure to propagate selections between the distinct but associated node representations.

Initially the prototypes were developed rapidly without much consideration of software engineering issues, but later development benefited from an architectural structuring that disassociated the strictly model and visualisation components. Changes to either component could then be contained or transmitted as appropriate and the range of their unwelcome knock-on effects limited.

Including the initial prototype development, four prototyping stages of design, development and testing were deemed appropriate. The first test would tackle major functionality issues, mainly regarding the fulfilment of the taxonomists' initial requirements, while the second stage would explore further functionality and some usability issues i.e. could the user access the functionality. The third stage would focus on ironing out major usability problems and the remaining functionality concerns. Finally, the fourth stage would build up to the construction of a final prototype and its validation via a formal test procedure.

6 Development & User Testing from Prototypes to Final Visualisation

6.1 Choosing Evaluation Methods

Once the initial prototypes were developed it was necessary to determine their respective capabilities and potentials, and whether they fulfilled the taxonomists' requirements. This judgement needed to be performed in a manner that would not only reveal the good points of each visualisation, but also the drawbacks, and importantly, how such drawbacks could be rectified and thus the visualisation improved.

Surely though, rather than a resource-heavy empirical testing methodology, there are other methods of measuring or verifying a visualisation's qualities? In fact, there are *metrics* that visualisations can be subjected to such as those proposed by Brath [22]. These attempt to give a quantitative measure of characteristics such as on-screen data density, percentage of occlusion in 3D systems, and percentage of identifiable data points. However Miller *et al* [122] point out that these metrics are suitable only for static visualisations, and usable metrics for interactive visualisations have yet to be established, so the utility of applying Brath's work to the prototypes developed so far is questionable. Metrics relating to the dimensionality of the observed information were difficult to use on the hierarchical information sets, such as multiple taxonomies, as the dimensionality of such information is not obvious. Some metrics, such as data density and amount of occlusion, could be useful for determining initial or pre-set conditions in an interactive visualisation, but an example of the problems interaction could cause metrics such as Brath's can be seen in the graph prototype screenshot shown in Figure 5.6. Here a user has both zoomed in and filtered out irrelevant information in the graph structure. These actions would have significantly reduced the on-screen data density by zooming, and the number of identifiable data points by the filtering action, thereby reducing the metric scores as defined by Brath, though it is obvious that the information is now of more relevance to the user.

Therefore, it was felt that initial testing of the prototypes would be best served by having representative users try the prototypes using representative tasks, rather than to rely on potentially unsuitable metrics.

6.2 Testing Methodology

As stated in Chapter 5, testing of the interactive prototypes consisted of four iterative cycles. The first three cycles concentrated on issues such as implementing and validating functionality, graduating towards usability concerns as the stages progressed. During each of these cycles the taxonomists participated in co-operative evaluations of the prototype visualisations during the testing stages.

In these evaluations, the taxonomists were asked to describe out loud their actions, observations, and difficulties as they progressed through a series of pre-defined tasks, which could also disclose task misunderstandings and generate further ideas for future iterations of the prototype. This style of co-operative evaluation, known as a think-aloud protocol and described by Tognazzini [164], is suitable for bringing out qualitative issues in early prototype testing, and is one of a suite of approaches advocated by Nielsen's "discount usability engineering" [132]. In line with this methodology, the testing was performed on only 3-6 users, but as gathering statistical data was not a concern at these stages, this was acceptable. More important was the fact that being working taxonomists, these users represented the real end-users of the visualisation. Other data capture methods such as data logging and video recording were attempted at various stages with differing degrees of success, and are detailed at the appropriate points in the text.

The speed of the overall interaction was not a concern for any of the test cycles. Whilst using their time efficiently is important for the taxonomists, getting the correct information and finding new patterns or correlations is a more important factor for this visualisation. Secondly, speed tests would require statistically valid quantitative data, and another system to compare results against fairly, such as is found in Stasko, Guzdial and McDonald [161], or Cockburn and Mackenzie [42]. Quantitative data would not be gathered until the final test, and in any case it was felt that the taxonomist's current paper-based system was not a viable comparative system with regard to task completion speed, especially as some tasks were simply not viable with the paper-based system. However, performance evaluation using quantitative data would have made more sense if the comparison was against another computer-based system utilising a command-line or graphical user interface.

The taxonomists who participated during various stages of this testing were of a range from PhD students to experienced taxonomists with many decades of research behind them, and also displayed a wide range of self-confessed confidence on using windows-based computer applications. The first three stages of testing used relatively small samples of participants: three in the first and second tests, and six in the third testing cycle. In the final test, which involved

19 taxonomists, there was a 2:1 split of males to females. All together, apart from the common involvement with taxonomy, they formed quite a heterogeneous group of participants.

6.3 First User Test

The first round of testing would validate the functionality of both prototypes against the taxonomists requirements and also gather feedback on their respective appeal to the taxonomists. To gain feedback on both of the initial prototypes shown in Figure 5.6 and Figure 5.10, the same three taxonomists from the RBGE who had supplied the initial requirements were invited to participate in an informal, co-operative user test. The taxonomists were supplied with a list of 12 example tasks to perform with both prototypes, a full listing of which is supplied in Appendix ‘A’. These tasks, a representative example being *“discover all siblings of the genus Kundmannia”*, were designed to cover the initial requirements the taxonomists had supplied us with, to discover if the interactions and resulting visualisations were understandable to them.

Following testing, a number of observations related to the performance of certain tasks were made during the evaluation, some of which are described below:

6.3.1 Example Graph-based Visualisation Observations

- Overlapping nodes in the graph-based prototype caused confusion. Though techniques exist for alleviating this, it is a common problem in graph/network visualisations. The zooming technique also worsened this effect when the entire graph was shrunk. In some tasks, it led to the belief that all the relevant nodes were visible, whereas some had been completely obscured. Therefore the overlapping nodes could lead to incorrect perceptions of the correct outcome for a task. Brath describes a metric for measuring the amount of occlusion that could have been applied here, though users being able to interactively alter the visualisation through filtering and zooming again would complicate it.

- The lack of spatial ordering of the different levels within the graph-based prototype caused problems, a difficulty that again occurs due to the layout of force directed graphs. Previous attempts to restrict the placement of nodes to certain areas of the layout, dependent on rank, had failed as it gave rise to areas of dense visual clutter. Therefore tasks that involved finding a name at a particular rank were time-consuming, as it was difficult to gain visual cues to tell if the name was indeed at the required rank. In short, users found it impossible to derive rank information from spatial layout alone.

6.3.2 Example Multiple set-based Visualisation Observations

- The set-based prototype abbreviated the names of the higher rank nodes. This was noted to be a problem as rank within the taxonomic structure can also be distinguished by name endings, and these were cut off by the abbreviation. Tasks that required names at a specific rank to be discovered were therefore hindered. This indicated that purely spatial cues were not enough to indicate rank in this prototype.

- When the task involves finding the existence of a specific non-genus taxa in a hierarchy, the visualisation shows, as in all other tasks, the distribution of the genera classified under that non-genus node. This has two effects. Firstly, if these genera were distributed across more hierarchies than the non-genus node itself was present in, it can appear to indicate to some users that the non-genus node was also present in more classifications than was actually the case. Secondly, as it was the genera that were highlighted, it sometimes caused the perception that one of the highlighted genera was actually the node the user was supposed to find.

From these and other observations, it was necessary to see how the prototypes could be improved to overcome any problems that were captured. One framework that allows us to pursue this goal is Monk and Wright's observation-invention pairs [124]. Using Monk and Wright's idea, each observation that is noted with the taxonomists will result in a corresponding *invention* that deals with the problem the observation is based upon, either by direct application or by using the invention to generate further possible solutions/inventions. For example, the observations above lead to the following possible solutions:

6.3.3 Graph-based Visualisation Solutions

- All directly or indirectly selected nodes, should be displayed on top of all other nodes, ensuring they are not occluded by information that is currently not of interest.

- As enforcing spatial ordering had already failed (the concentric circle spatial restriction on the graph nodes), a non-spatial cue for indicating rank appears to be a solution. In addition to the glyphs that indicate individual classification membership, each node could also carry a prominent glyph that indicates rank.

6.3.4 Multiple Set-based Visualisation Solutions

- Additional rank information could be indicated by textual labels aligned to the respective ranks at the side of each classification.

- Non-genus as well as genera nodes could be highlighted when selected. This in conjunction with the previous solution should give more clarity to the non-genus node when it is selected.

These solutions, and others, could be implemented on the next iteration of the prototypes, tackling the problems encountered by the taxonomists who used the prototypes.

Suggestions made directly by the taxonomists included allowing more manual control over the assignment of colours to groups and sub-groups of nodes. At the time of the test colours were automatically assigned, one to each group of leaf nodes selected, such that selecting a group that contained five sub-groups would result in five different coloured sub-groups being produced. The taxonomists asked if only one colour could be assigned per selection, and then any further investigation could be undertaken by selecting a particular sub-group, thus differentiating that sub-group with a different colour to the previously selection.

One striking feature was that the users envisaged further tasks after seeing the prototypes, and these included the ability to see if certain nodes only appeared in one classification, a task neither visualisation prototype could adequately perform at the time. Another wish was to see the extension of the visualisation of a single genus’s sibling distribution to a larger set of genera. Working with the IV prototypes and seeing the information graphically appeared to encourage them to imagine new ways of manipulating the information, which were not possible with their paper-based system.

While both prototypes could show the information necessary for answering the questions posed by basic tasks, there was a definite preference for the set-based prototype. The taxonomists stated that the multiple tree effect and the grouping of sets was closer to how they viewed classifications when working with taxonomic data, rather than a node-link diagram. This supported Parunak’s view [139; 140] that a set-based presentation style would match a taxonomist’s mental model more closely than the graph-based visualisation, which instead closely matched the underlying data abstraction. It was felt that even with filtering mechanisms the graph-based prototype showed too much visually, leading to clutter and confusion. An initialisation period necessary for the graph visualisation to settle into a preliminary layout was also detrimental.

Therefore the main outcome of these tests was that continued development of the set-based visualisation was given precedence over the graph visualisation, due to the overwhelming preference for the set-based visualisation. Many functionality and usability issues were discovered through user observation, and these were tackled in development before the second stage of testing.

6.4 Second User Test

After the functionality issues brought up in the first test were addressed (though in the set-based visualisation only), the second stage of testing would aim to confirm that the proper functionality was now in place, and also find usability issues with the interface. The main changes made during development included the implementation of the new colour selection mechanism as suggested by the taxonomists. As each new selection added a new colour distribution to the display, additional controls for clearing, inverting, and merging the displayed colours were also incorporated into the general interface. The second major change was the extension of the inter-tree linking technique to a brushing technique, in which the mouse pointer acts as a temporary selector, brightening selected nodes as the mouse pointer passes over them. The brightening effect is then linked to the nodes where they occur in the other displayed classifications. Brightness was used as a cue as with intelligent implementation it would not interfere with the hue dimension of the displayed colours e.g. bright green is still identifiable as green, bright red is still red etc.

Up to this point, the bulk of the development effort had been concentrated on the input and output of the actual visualisation display, and not to the larger interface in which it was now embedded. It was also necessary to discover whether components of the interface such as the control panel, list and history bar in conjunction with the visualisation were useful and functional.

The twelve tasks used in the first test were combined into three small scenarios (Appendix B) for the taxonomist users to run through, and adapted to ensure that the users attempted to utilise all the aspects of the interface that were to be tested. The move from tasks to scenarios is important. Whilst tasks are useful for examining the functionality of a system, they are too prescriptive to catch the contextual nature of situated work. Scenarios aim to address this point [34], and capture usability problems that might otherwise escape unnoticed.

They were also adapted to take into account the fact that the test data set of classifications had expanded from four to seven, as shown in Figure 6.1. The extra classifications contained more information in total (1,500 nodes compared to 500 previously) and had more complex hierarchical structures. This began to test the display limits and scalability of the visualisation.

Therefore, the important points concerned the accuracy of the information conveyed by the visualisation as perceived by the users, whether usability or visualisation issues tricked the users into observing wrong outcomes, and how the extra information affected the scalability of the visualisation.

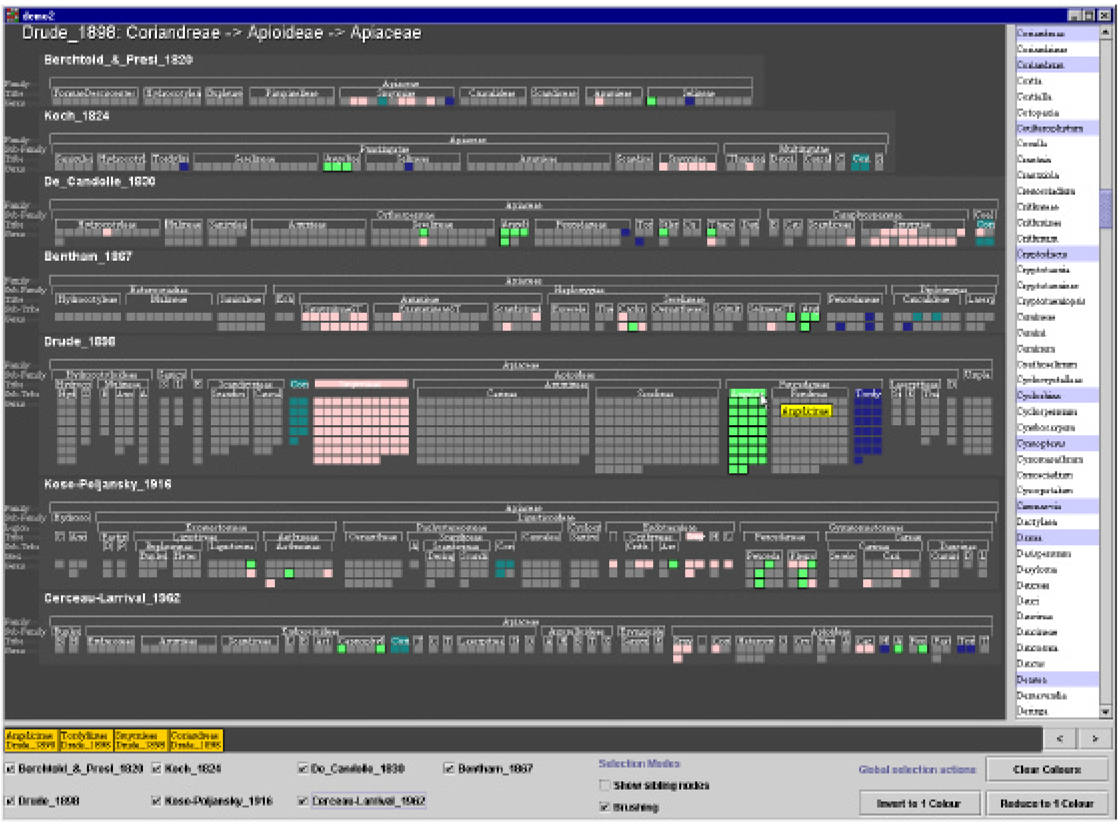


Figure 6.1. Set-based visualisation with extra classifications

The test used the same direct observation procedure described for the first test, but the screen was also recorded by a video camera. A log was automatically recorded by the software to track mouse actions, interface actions and nodes being investigated, along with the times of these events, as each individual test progressed. Afterwards, a small questionnaire was given to the users, asking them to judge what they perceived to be the best and worst aspects of the visualisation system so far.

During analysis of the collected data the software log proved invaluable in clearing up ambiguities in the note-taking records, showing the order of events such as swiftly executed multiple mouse clicks. However, the video footage was disappointing. It provided an adequate audio record, but the actual video output was affected by a combination of three factors:

- The continuous flicker of the monitor as picked up by the camera.

- The fact that the monitor was at a higher resolution (1280 by 960) than most standard video cameras can record without loss. This meant fine detail was lost.

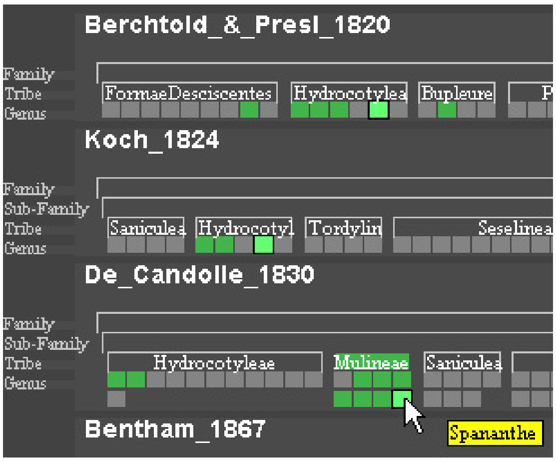
- The interactive elements in the set-based visualisation are substantially smaller than standard UI elements such as buttons, scrollbars etc. Monitoring interactions with these elements proved almost impossible, especially when compounded by the previous two points. The visualisation is not alone in this characteristic, as other space-efficient visualisations such as Johnson and Shneiderman's Treemaps [96] and Beaudoin, Parent and Vroomen's Cheops tree browser [14] deal with small, tightly packed elements. Keim's work, with Kriegel and Anderst [102], and with Herrmann [101], deals with individual IV elements as small as the size of pixels. Whether the average IV element is smaller than the average standard UI element is not something that has been calculated, but these characteristics should be borne in mind when using video in IV user testing.

Therefore for IV, it can be argued that a direct screen capture method to remove flicker and resolution problems, using a tool such as Camtasia [28], could be a more fruitful approach than using standard video recording.

The users confirmed that the underlying functionality in place was valid and working as expected, even if usability issues made the results of some operations on this functionality ambiguous in interpretation. A number of usability errors were found, using Monk and Wright's methodology [124] as in the first test. The most crucial discoveries were that the right-hand list in the set visualisation was the source of many problems, and the use of checkboxes on a separate control panel to hide or display individual hierarchies led to problems. The list was a basic Java UI object, and was found by the users to lack the functionality of a comparable Windows UI list object, which has features such as keyboard-driven search as well as mouse-driven functionality.

The control panel for manipulating individual classifications was situated below the main visualisation display. It was found that users would have preferred indicators and controls for individual classifications to be integrated into the visualisation, rather than having to divert their attention elsewhere. Specifically, a problem arose because as users became focused on the visualisation panel, they forgot about the existence of hidden classifications. This led to erroneous answers in some scenarios that involved finding unique nodes or the first historical appearance of nodes.

The newly introduced brushing technique was found to be extremely useful in exploring the data, as not only did the user not have to press the mouse button to see a node or group's distribution, moving the mouse off that node automatically cleared the temporary selection as well. Compared to the manual selection and 'clear' button combinations that would have otherwise been necessary to see such data, the brushing metaphor allows questions to be asked of the data quickly and easily. An example of the brushing behaviour found in the second round of tests can be seen in Figure 6.2, the software log fragment showing the behaviour of the user on the selected nodes in the associated screenshot. The timings show that each inspection took about one second, quicker than could be achieved by selecting/de-selecting each node, but slow enough to show that the user was observing the outcome of the brushing, rather than just moving the mouse pointer about. Users also felt that the ability to toggle the brushing mode on and off was useful, as in some circumstances, especially when investigating sibling distributions, the constant flashing proved distracting when all they wanted to discover was the name of a node using the mouse tooltip.



```
405 :Mouse hover over node
Trachymene at classification
De_Candolle_1830
406 :Mouse hover over node
Huanaca at classification
De_Candolle_1830
407 :Mouse hover over node
Mulinum at classification
De_Candolle_1830
408 :Mouse hover over node Pozoa
at classification
De_Candolle_1830
408 :Mouse hover over node
Spananthe at classification
De_Candolle_1830
410 :Mouse hover over node Drusa
at classification
De_Candolle_1830
411 :Mouse hover over node
Diposis at classification
De_Candolle_1830
411 :Mouse hover over node Bolax
at classification
De_Candolle_1830
```

Figure 6.2. Brushing behaviour on a group of previously selected taxa.

The outcome of this round of testing was that usability problems entailing essential changes to the visualisation's interface were identified. These are to be rectified in time for the next round of user testing. It was also discovered that standard video recording methods were not as useful for the IV application as they are claimed to be for standard UI usability analysis. Almost paradoxically, it was found that software logging data revealed more information about the user's interactions with the visualisation, although this data was gathered in a non-visual form.

6.5 Third User Test

The third stage of the development progressed to tackle the problems found during the analysis of the second stage of testing. The list had been augmented with more sophisticated keyboard-driven navigation mechanism, and due to the underlying MVC architecture, the selection colours were easily linked from the main visualisation to the taxa names in the list. The brushing metaphor had also been extended to include the list, so that hovering the mouse above a name in the list would brighten its representations in the main portion of the visualisation. Further selection mechanisms had been integrated into the visualisation, so that selecting a classification name in the visualisation would toggle the display or concealment of that particular classification. Concealed classifications had their names displayed in italics at the appropriate point in the visualisation, giving a visual indicator to the user of their presence even when hidden.

So at this stage it was necessary to ascertain whether these *usability* issues brought up in the previous test had been adequately dealt with. Testing would consist of the methodology used in the previous cycles, but adjusting for the assessment of the techniques used so far. The limits of the visualisation with regard to scalability were also tested further, and it was decided to view the effects of a system with less screen space or processor power on the visualisation.

Figure 6.3. Visualisation at the time of the 3rd User Test

Essentially, the third set of user tests was a re-working of the second test, although focused almost exclusively on usability rather than underlying functionality concerns. Of course, if any suggestions for additions to the function suite were made they would be given consideration, as however fastidious the testing and resulting analysis is, the design and development of such a novel application cannot be partitioned neatly into distinct stages. This is one of the reasons why the iterative model was first mooted for application design and development in general. Issues such as functionality and usability are fundamentally entangled; usability being a function of how well a user can manipulate a system’s functionality [131]. Muñoz [126] backs this view up, stating that *“usability is determined by the whole functional process, not just the* *'look and feel' of the UI."* Simply put, if the functionality isn't there that the user expects, then neither is the usability. New users, of which there were three in this test, could pick up on what they consider to be missing functionality. For instance, an example concern from this test was that the ordered list did not support the common Windows functionality of navigating via the up and down arrow keys, the Java list widget instead using these cursor keys to select rather than navigate. To the developer, this would be a question of changing the lists' functionality. To the user who discovered this problem, it was a usability issue; they couldn't use the list as they expected to. In this test though, the focus is on *primarily* monitoring how the users can interact with the functionality that is already in place, rather than focusing on whether more utility needs to be added. This testing should also discover whether any system modifications have brought up new, unexpected problems.

The test methodology was the same as for the second user test but without video recording. The three users had grown to six through the addition of three new taxonomists to avoid the problem of the original users' familiarity with the system becoming an overly significant factor. The tests were performed at the RBGE, after conversion of the Java application version into an applet available over the World Wide Web, which also gave the opportunity for further potential users to provide feedback. A set of three scenarios was given to each participant and they were asked to describe their thoughts as they progressed through them. Each participant had a maximum of 45 minutes to perform the scenarios, simply as a result of the taxonomists only having limited availability due to their work schedules, and all of them completed within that time limit. Logically enough, the rationale behind the use of scenarios became partially redundant here with respect to mimicking interruptions to the users' concentration. As the tests were conducted in the RBGE offices, the users' real work environment, they received real interruptions such as telephone calls and colleagues asking them a quick question during the testing.

The data set had now acquired an extra taxonomy, bringing the total up to eight and the number of displayed node representations to 1,700. As with the second test, note-taking plus verbal protocols along with event logging was used to capture user actions and reasoning. Monk and Wright's observation and solution framework was then applied to this data, and in this cycle revealed 32 unique problems from a total set of 46 found during the test; different users experiencing the same problems accounting for the differential. The full list of observations and solutions can be found in Appendix C.

One trial took place using a monitor with a screen resolution of 1024x768 pixels, instead of the usual 1280x960 resolution used until then (an issue that had been expected in moving the test to the users workplace – they don't have the same equipment!). Whilst not appearing to be widely dissimilar, the difference between the two is such that the lower resolution has less than two-thirds of the total number of pixels of the higher resolution. Compounding this situation is the fact that the list, control and history panels all have a fixed allocation of space, so the pixel loss was concentrated solely in the visualisation panel, which was calculated to have less than half as many pixels as it would at the higher resolution. Unsurprisingly, the user who performed the trials on this display encountered problems that the other users did not.

Nodes were drawn at a fixed size at this stage in the visualisation development, and any over-run on the screen was accommodated by allowing the user to scroll the visualisation vertically. This proved to be necessary with the 1024x768 resolution but not for the higher 1280x960 resolution. Observing the participant who used the lower resolution, it was noticed that they conspicuously gave up on some tasks that involved comparing information between hierarchies when scrolling back and forth was required. Not only did they have to manipulate the visualisation during comparisons, but they also had to remember the display of one classification, scroll, and then compare it from memory against another classification. This kind of behaviour nullifies the fundamental perception over cognition benefit central to visualisation, and obviously this situation needed to be rectified. The proposed solution, to be implemented in time for the fourth and final testing stage, was to use relative layout sizes for nodes rather than absolute fixed sizes so all classifications needed for a task could be viewed simultaneously.

This participant also had to deal with the consequences of a slower processor, which manifested itself as a slowdown in the response of the system to user actions. This caused the user to repeat commands before a response to the original action had occurred, and resultantly the visualisation eventually recognised and performed the action multiple times, leading to obvious frustration. The machine was stated to be of a poor specification (133 Mhz Pentium), and was soon to be replaced. Given that CPU speeds are still obeying Moore’s law, doubling every 18 months, but average monitor sizes are growing slowly and linearly [77], it must be re-iterated that the restriction of screen real estate is, and will continue to be, the major limiting factor for the set-based visualisation.

One particular usability problem that was a result of changes made after the previous round of testing was discovered in this round of testing. The New Drude classification had been introduced with two nodes at its uppermost rank, which formed two separate family trees within the classification. To relate these nodes for manipulation and display purposes a ‘virtual’ top node was introduced, to which all nodes without a parent would attach themselves. This virtual node carried over unintentionally and unnoticed into the visualisation layout, introducing a gap between the classification name label and the family rank taxa of each classification. With a gap now above *and* below a classification name, some users were confused as to which classification the name labels referred to, and this ambiguity led to wrong selections. The next stage of development would need to include the removal of this misleading layout quirk.

One further functionality request was made for the ability to compare classifications only by higher ranks within the classifications. This would necessitate the hiding of all nodes beneath this target rank, with the nodes at the target rank forming a new layer of leaf nodes. This would require an extension of the intermediate rank-hiding requirement shown below in Figure 6.4.

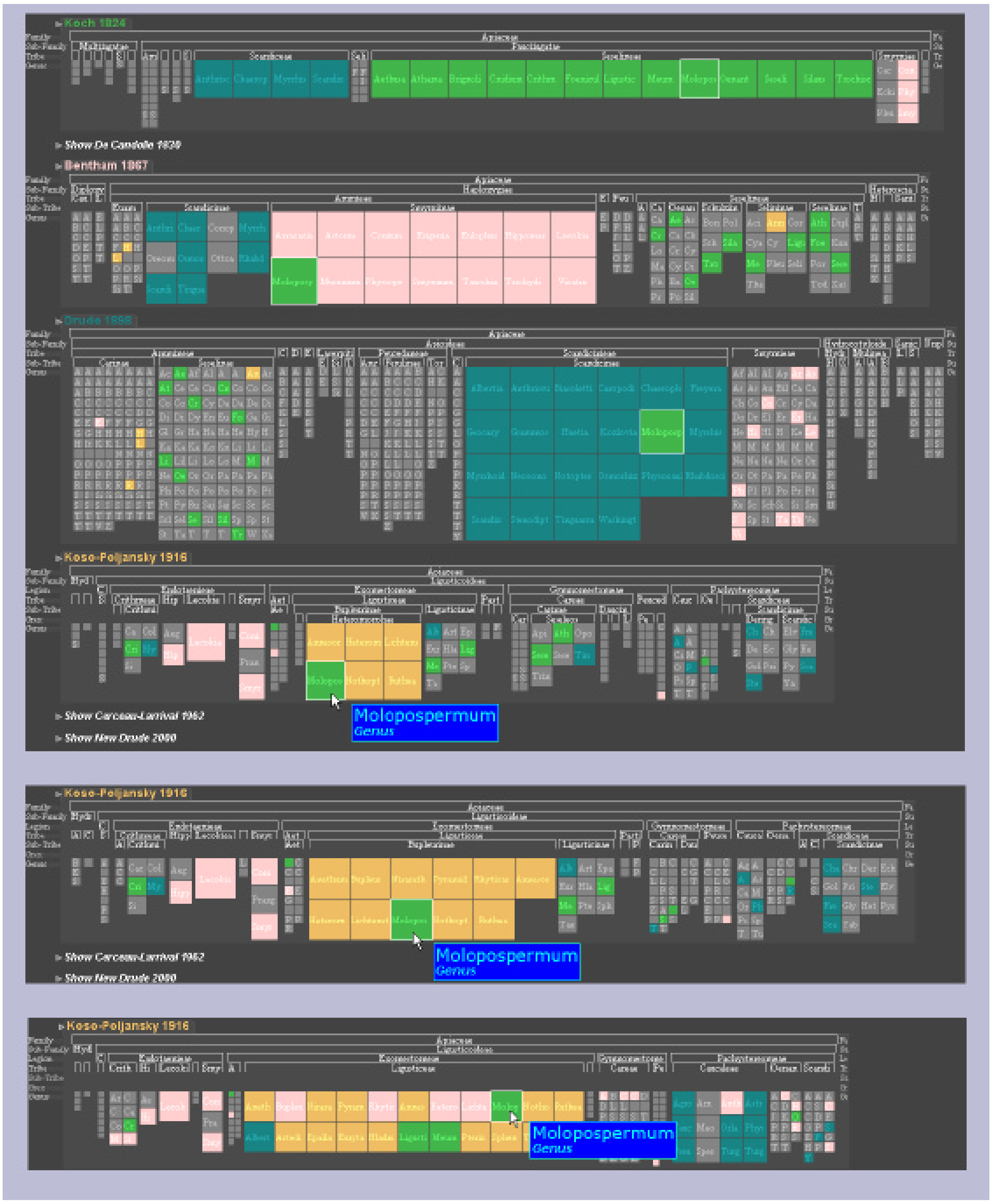


Figure 6.4. Rank hiding. Successive internal levels of the hierarchies are removed so comparisons can be made by common ranks only.

Here, in Figure 6.4, the genus *Molopospermum* has been selected as the focus of a sibling query. The screenshot shows it has been grouped with a completely different set of taxa each time, as witnessed by the previously coloured nodes being scattered well away from the genus in each progressive hierarchy. However, not all the classifications have been defined with the same group of ranks. For example, in Koso-Poljansky, *Molopospermum* is a member of *Heteromorphae*, a taxon at a rank unique to Koso-Poljansky's classification – *Grex*. Perhaps it would be fairer to make a comparison at the higher and more common *Sub-Tribe* rank, without the effect of the interceding *Grex* rank? Thus, the second screenshot in the figure shows the same selection in Koso-Poljansky after the removal of the unique *Grex* rank, eliminating its effect. However, this screenshot shows that *Molopospermum* is still grouped, under the *Sub-Tribe Bupleurinae*, with a set of taxa it has had no previous relationships with at the sub-tribe level; all the nodes in the sub-tribe are still coloured orange. So in the third portion of the figure the *Sub-Tribe* rank has also been removed, to discover if there are intersections between taxa in *Tribes* that contain *Molopospermum* in the four classifications. Now, there is at least evidence of previous associations, as taxa, represented by the pink, blue, and green nodes, appear with *Molopospermum* in the *Tribe Ligusticeae*.

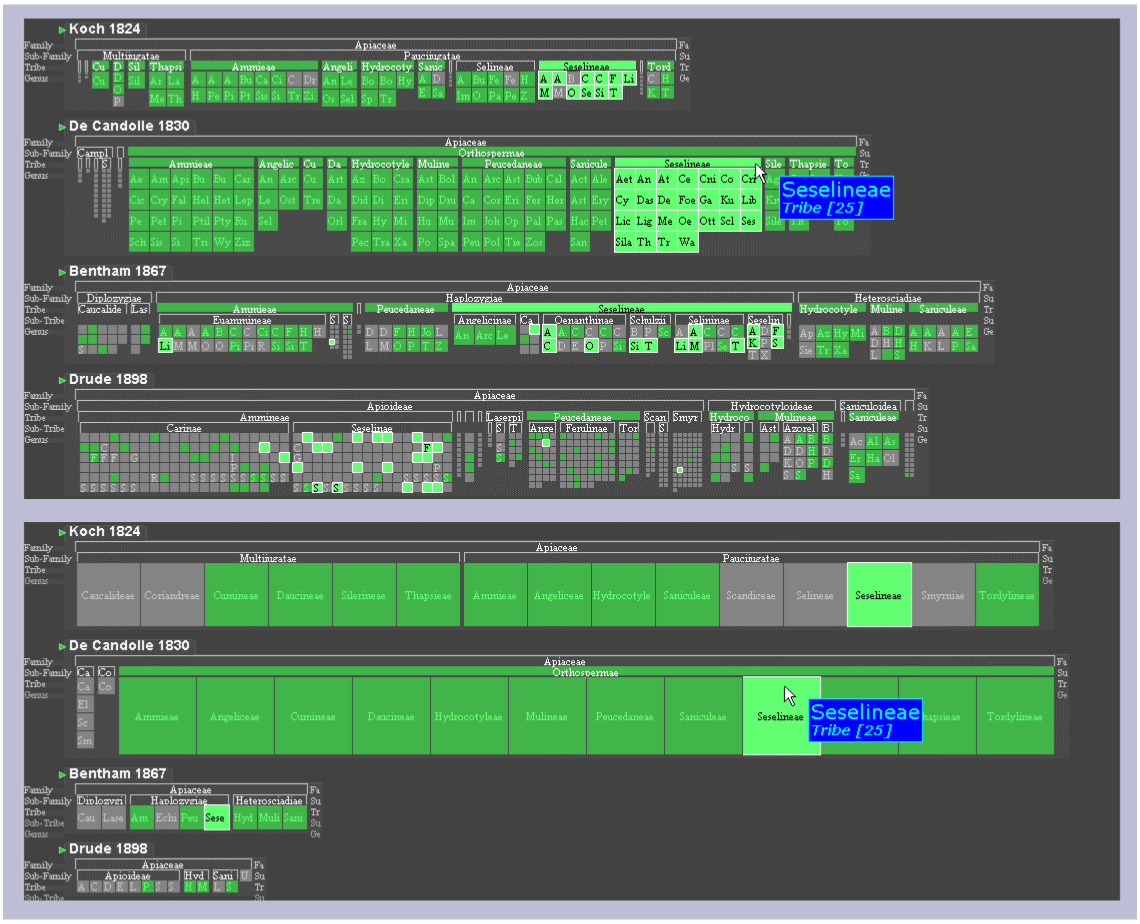


Figure 6.5. Removal of lesser ranks in the lower screenshot allows a clearer picture of shared *Tribe* names to appear.

This mechanism was extended so it could produce situations such as that shown in Figure 6.5. Here, in the screenshot in the upper half of the figure, the selection of a sub-family in the *De Candolle* classification has lit up a large portion of the visualisation in green. However, if the user wishes to see only those *Tribe* names which are common from *De Candolle* to other classifications, then all the extraneous information highlighted will tend to mask the information they are searching for. Allowing the hiding of the ranks below *Tribe* level will lead to only selected *Tribe* nodes being highlighted in green, which is the information the user is concerned with. This situation can be seen in the lower half of Figure 6.5.

6.6 Combining Linking & Focusing Techniques

By this stage in development the set-based visualisation had reached a limit of about 1,500 nodes (the original Apiaceae data set displayed in Figure 6.3), beyond which individual node representations became too small to manipulate without difficulty. As described in the previous section, lack of space was a problem on smaller monitors, and scrolling had been an unsuitable solution to this situation. As focus + context techniques allow visualisations to give space to more interesting information, it was decided to investigate their possible use in the set-based visualisation, aiming to increase the size of the data sets it could comfortably handle. However, as the nodes were linked across several hierarchies, such a technique would need to accommodate multiple focal points linked across these hierarchies.

It can be argued that a general graph visualisation technique, such as Osawa's [137] which allows multiple focii, would combine the multiple hierarchies into a single visualisation artefact, so any linking between spatially distinct visualisations for a focus + context technique is unnecessary. However, as discussed in Chapter 2.4.2, graph visualisations tend to be harder to comprehend than tree visualisations, and this was confirmed by initial user testing on the graph and set-influenced prototypes as well as previously by Purchase *et al* [143].

Chapter 2.5.4 showed that while linking and filtering has been used as a combination of interaction techniques, focus+context methods have not been used in conjunction with linking techniques. The main difficulty is in linking the effects of continuous space-warping focus+context techniques between multiple representations of objects. However, it was also stated that if the focus-defining attribute was discrete and associated directly with the object, such as is the case with a 'Degree Of Interest' value (DOI), that value could be linked between multiple representations of that object. Given that the set-based visualisation consists of linked, multiple hierarchies, an attempt can be made to extend the focus + context metaphor through linking areas of interest between the hierarchies using DOIs. This would enable the visualisation to focus on more relevant information within larger data sets than it has been able to accommodate beforehand.

6.6.1 DOI and Display Calculation for a Single Hierarchy

The DOI function is calculated as follows when applied to just a single hierarchy. All leaf nodes start with an average value as their DOI, indicating that the layout is initialised with no focal areas, as shown in Figure 6.6. Whenever a node representation is selected within a tree representation (or from an ordered list positioned to the side of the trees), the node and all its descendent nodes, if any, are given a maximum DOI value as seen in Figure 6.7. All other nodes not affected by this selection have their DOI values reduced slightly to or above a non-negative minimum DOI value.

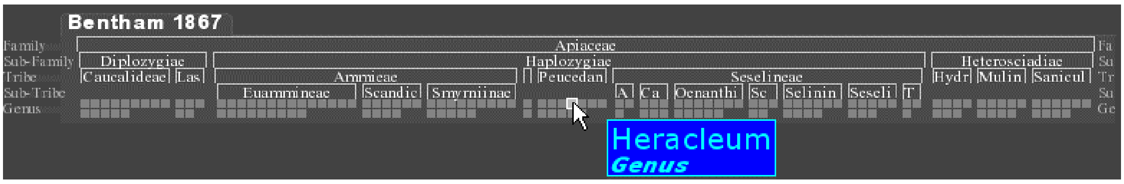


Figure 6.6. Initial display of a hierarchy. All nodes are of a uniform size.

The minimum, average and maximum DOI values are assigned as non-negative integers e.g. 1 for minimum, 6 for average and 12 for maximum. Hence in this case after five selections without participation, a node's DOI would drop from the average to the minimum value, and a node chosen in the first selection, would now have a DOI dropping back from the maximum value and approaching the average value. Alteration of these values can have various effects e.g. setting the average and maximum values to be equal would result in a situation where unselected nodes had their DOI values reduced but selected nodes' DOIs remained at the average value.

The policy of slightly reducing unselected nodes' DOI values reflects the notion that a new selection would slightly reduce the interest of the user in all other unaffected nodes. Over a number of selections this gives nodes and sub-trees of little activity smaller and smaller DOI values in absolute terms as well as relative terms, and provides a mechanism for bringing DOI values back down from the maximum threshold. It also differentiates the DOI function presented here from Furnas' original description [65], as it does not have his API (a priori importance) measure but rather incorporates an on-the-fly historical importance factor.

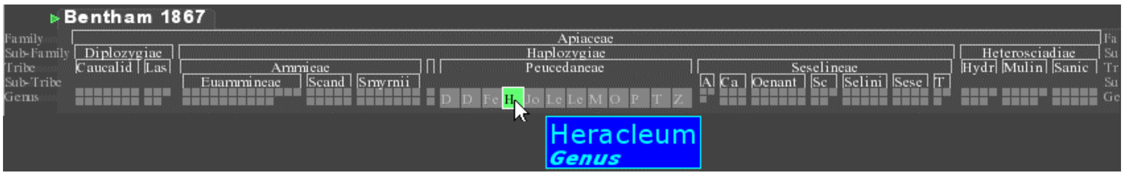


Figure 6.7. Selection of a single node increases the DOI and size of all nodes in that group.

At this point, visualising nodes directly by a linear extrapolation of their individual DOI values would give a display of larger, selected nodes distributed throughout the tree. However, one of the properties of the set-based visualisation is that it permits browsing of the structure through the selection of previously unselected but related nodes. This behaviour was noted in previous user experience with the visualisation, which had shown that users tended to make an initial selection from a list of ordered nodes, and would thereafter browse selected or related nodes using the tree representation. Therefore all nodes in each distinct group of leaves within a tree (*leaf group*) are drawn as the same size, indicating that the unselected but related nodes are also of interest, and also to make them physically large enough to select. To accomplish this, it was decided that the size of node representations should be dependent on the average DOI value for the nodes in a leaf group.

One problem with this approach is that selection of one node, and therefore increasing the DOI of one node, in a large leaf group will not lead to a discernible overall increase in the average DOI value of that group. The average DOI value of a leaf group is controlled by the ratio of selected to unselected nodes in that group, regardless of whether 1 or 100 nodes were selected in total. To give a fairer average value, each unselected node in a leaf group has their DOI value altered by the following function:

NewDOI += (MaxDOI – OldDOI) * (|{nodes in last selection} n {leaf group nodes}| / |{nodes in last selection}|)

Simply put, a node has its DOI value increased to a value somewhere between its current DOI and the maximum DOI. The proportional increase in value is directly related to the percentage of nodes picked by the last selection action that are siblings of the node (in the same leaf group). So, if only one node was selected in the last action; and was not a sibling of another node in this tree; that other node would not have its DOI value for that tree increased at all, as the proportion of selected nodes in its leaf group was 0%. However, if the two nodes were siblings, then the DOI value for the unselected sibling would increase to maximum, as all (one) of the selected nodes were siblings in that tree. All other nodes in the same leaf group would similarly have their DOI values for that tree increased to maximum.

Figure 6.8. Multiple selections. The first of these selections, towards the left of the figure, are beginning to shrink in size.

This method ensures that DOI values are a function of proportional measure, rather than absolutes. Having 100 selected nodes rather than just 1 selected node will not necessarily result in a higher average DOI value for a leaf group. Rather, it is the proportion of selected nodes from the whole selected set that affect the group's average DOI. Figure 6.7 shows the result of a single selection whilst Figure 6.8 shows the results after five selections. Figure 6.8 also demonstrates the reduction in DOI values of previous selections. The figure shows a number of selections that have been made going from left to right in order of selection. The selected groups towards the left-hand side of the hierarchy have slightly smaller node representations than the selected groups towards the right-hand side, emphasising the fact that they are not the most recent selections. The unselected groups have shrunk almost to vanishing point.

6.6.2 Display Calculation

After each selection, DOI values are recalculated for all leaf nodes. Following this, the distribution of free screen space is similar to the Continuous Zoom [11] technique developed by Bartram *et al.* Free screen space in this context is understood to be the remaining space after tree details and internal nodes have been displayed. The horizontal space for the tree is allocated proportionally to leaf groups, the metric being the group's share of the tree's total DOI value.

This gives each leaf group an area decided by the amount of free vertical screen space with horizontal space decided by the groups' own relative DOI total within the tree. The individual leaves in a group are arranged in a grid pattern throughout the group's allocated space. As previously stated, the leaves in a group are drawn as the same size to aid browsing behaviour, irrespective of whether they have been directly selected or not. Individual DOI ratings are calculated, so leaf nodes could be drawn in proportion to these if needed. However, this would reduce the size of unselected sibling nodes, making mouse selection of such nodes, and hence browsing of the structure, more difficult; effectively reducing the main part of the visualisation interface to an output channel only.

This approach also gives the leaf groups a crisper, more cohesive appearance than would be the case by assigning individual sizes proportional to the varying DOI values of the leaves in a tree. Furthermore, it simplifies calculation of a leaf group's internal layout and helps emphasise the set-based nature of the visualisation.

One problem encountered with the layout method was that leaf groups of only one or two nodes in a large hierarchy would sometimes be squeezed to less than one pixel in width after the space allocation stage. This occurs because the algorithm distributes area to the groups without consideration of proportion. It is hard to draw meaningful representations in a strip 1 pixel wide by 60 pixels deep. This was remedied by calculating a vertical 'efficiency' for each group, namely the amount of vertical space they would use from their allocation if square node representations were drawn in their allotted area. Groups with a low ‘vertical efficiency’ are given ‘extra’ horizontal pixels to work with, in which larger width, and therefore heights, of square leaf node representations can be drawn.

These extra pixels are available through a pixel ‘salami fraud’ in the space allocation algorithm. Groups are given a real number of pixels in width, which in practicality is rounded down to an integer for display purposes. The missing fractions add up and form a cache of breathing space that can then be used to give small-sized groups a better chance of displaying themselves.

6.6.3 DOI Calculation Across Multiple Hierarchies

Given that in one tree the selection choice is simply between selecting one node in a group, or the entire group, the DOI calculation algorithm must be extended to cover multiple trees for its full use to be understood.

As in the underlying data structure, each node can appear within multiple hierarchies. Therefore each node is assigned multiple semi-independent DOI values, one per representation as shown in Figure 6.9, which quantify the current level of interest they hold for the user given the history of previous selections and their context within each hierarchy. The problem now is how to link these separate DOI values for the same node.

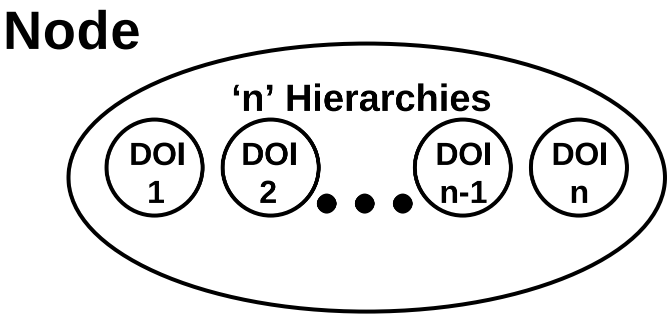


Figure 6.9. A node has multiple DOI values, one per hierarchy.

If a node is selected, the answer would seemingly be to simply give each of its DOI values the same rating, namely the maximum DOI value given to nodes when selecting within a single hierarchy. As previously stated, visualising nodes directly by their individual DOI values would give a display of larger, selected nodes distributed throughout the multiple trees. It would also seem to indicate that one global DOI for each node would suffice, calling into question the need for multiple DOIs for one node.

This cannot occur though, because the previous DOI function alters an unselected node’s DOI according to the proportion of selected nodes that are siblings. As nodes have different siblings in different trees, the proportions, and hence derived DOI values, will be different for the same unselected node in different trees. For example, consider unselected nodes in a leaf group in a tree that had 5 out of 25 selected nodes as siblings. They would see their DOI values for that tree increase by (5/25 =) 20% of the difference between their current and maximum values. In another tree, some of these same unselected nodes may have 10 out of the 25 selected nodes as siblings, leading to a (10/25 =) 40% increase for the nodes DOI value in that tree and so on. This leads to differing DOI values for the same node across different trees, hence the need for one node to accommodate multiple DOI values, and their description as ‘semi-independent’.

Selections can now be made in sequence to compare and contrast different leaf groups distributions. Each selection will produce it’s own focal effects across the hierarchies, and these will have a cumulative effect on the nodes’ DOI values.

6.6.4 Display Calculation for Multiple Trees

Displaying the multiple hierarchies is essentially the same process as for single trees, except of course the screen space needs to be vertically sub-divided first amongst the trees. Screen space is allocated in proportion vertically according to the relative total sums of leaf node DOIs for each tree. Trees can be marked as hidden from view, and subsequently such trees are given no space allocation. If the multiple trees’ other display elements overwhelm the display area by themselves, then leaf node sizes are set to a minimum, and the display can be scrolled vertically to view all trees.

Multiple trees may suffer from the space allocation problem described earlier that occurred between groups in the same tree. Smaller trees may have their space allocation squeezed by a number of larger trees to a point where meaningful node representations cannot be drawn, even if the tree has a high average DOI. Therefore each tree is given a minimum allocation of vertical display space to work in, and accounts for the perception that very small trees have an unfair allocation of space, as seen in Figure 6.10. Such a situation is preferable to one where the tree is hardly visible at all.

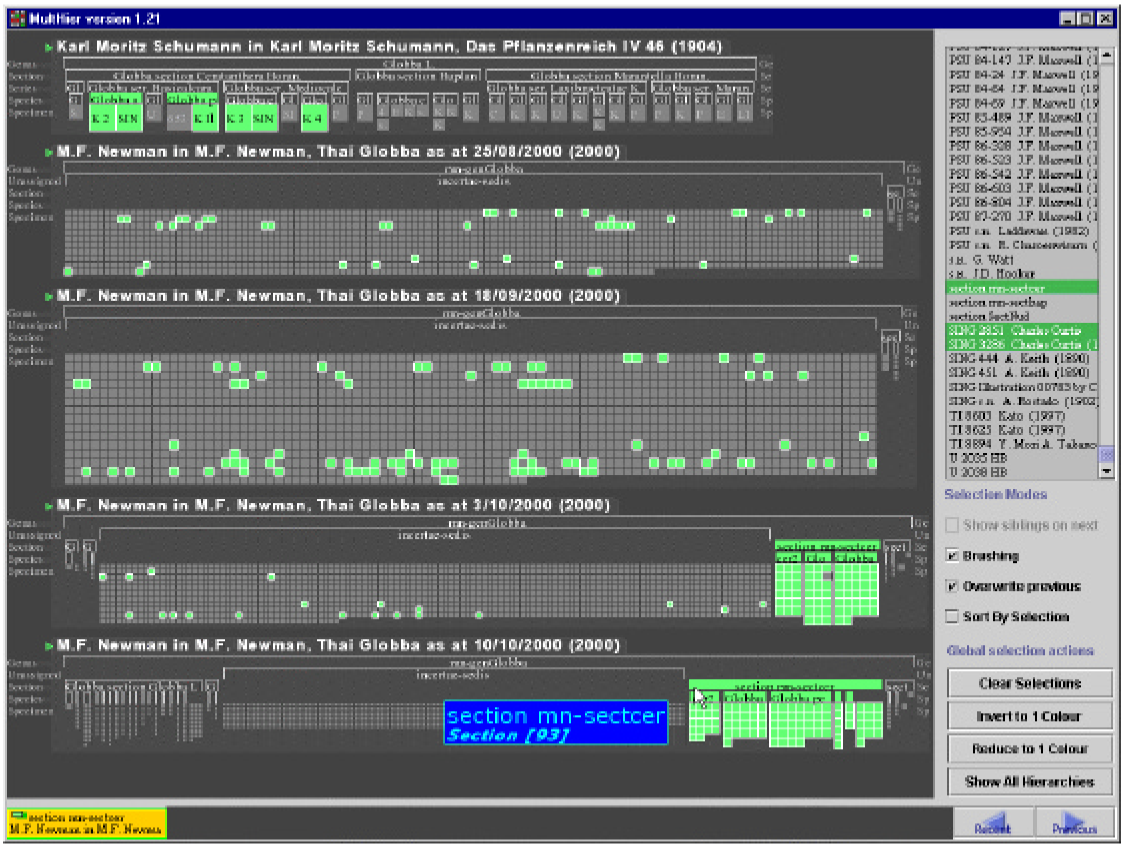


Figure 6.10. Selection of a group of nodes. The proportion of these nodes contained in other groups is the main factor in deciding their DOIs and sizes.

6.6.5 Example

An example interaction with the visualisation is described using the much larger multiple hierarchy data set shown in Figure 6.10. Each hierarchy displays a stage in the construction and refinement of a classification of plant specimens by a single taxonomist.

To begin with, the display of approximately 6,000 node representations (leaf and category) draws all leaf nodes at the same scale. In such an interface, picking out with the mouse, never mind locating, one particular node becomes difficult. However, the prior testing showed that users usually have particular target nodes in mind when browsing these taxonomic structures, analogous to the manner in which users tend to have a page they want to head to when they start up a web browser. The ordered list on the right-hand side enables the user to quickly locate their node of interest and select it there. (It could of course be argued that a lot of browser users head for a search engine site, but the ordered list serves the same function for the hierarchies.)

Selection of a single leaf node from the list increases the DOI of the groups in which the leaf node is present. As some groups will now have relatively larger DOI totals, their contained leaf nodes are drawn proportionally larger and as such are able to be readily picked out by the mouse pointer, thus enabling a browsing behaviour if needed in the visualisation from here on.

Figure 6.10 displays the selection of a sub-tree in the bottom hierarchy. Here, the proportion of selected nodes inside a leaf group affects the size of the leaf nodes in that group. The large node group in the third hierarchy down has large leaf node representations as it contains almost all of the nodes originally selected. The large groups in the second and fourth hierarchies have smaller leaf node representations, as they only contain a proportion of the selected node set, and the large group in the fifth hierarchy has no selected nodes and so is drawn with extremely small leaf nodes.

6.6.6 Discussion

Before the implementation of the multiple linked focus + context method, the set-based visualisation had reached a practical limit of about 1,500 nodes on-screen. Selected nodes were marked in colour, as they are now, but all nodes were drawn at a uniform scale. However, introduction of the linked focus + context method has so far enabled data sets with a total of up to 8,000 nodes to be displayed in full.

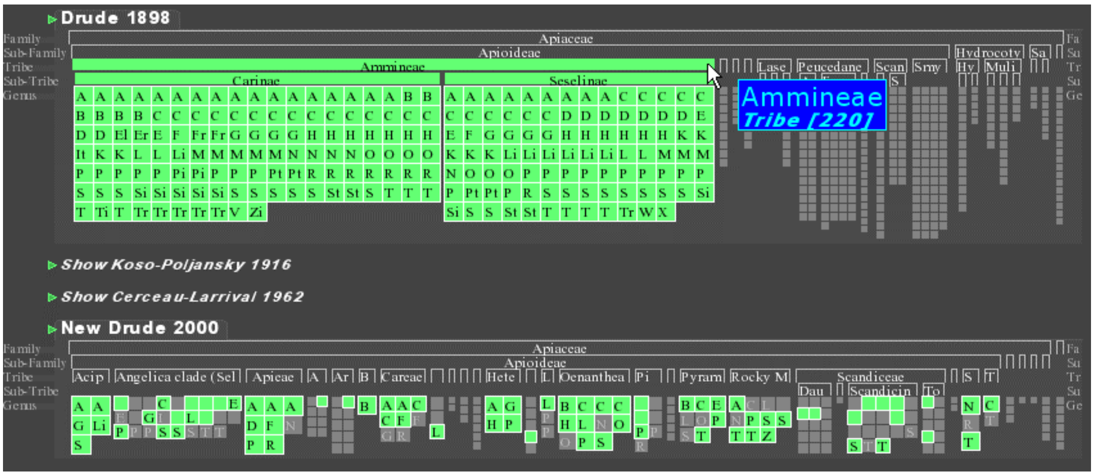


Figure 6.11. Focal spread in the bottom hierarchy. Many groups' DOIs are affected almost equally by the selection, and consequently no distinguishable focal areas appear in that hierarchy.

One limitation inherent in the approach occurs when a selected leaf group in one tree is almost equally distributed throughout the leaf groups of another tree. The current method will produce almost identical average DOI values for each leaf group in this tree, so no group will get a proportionally larger share of space at the expense of another. The display will then show the tree peppered with coloured node representations, but all the leaf groups will be drawn with representations of roughly the same size, each acting as context for the others as much as it is a focal point itself, cancelling the effect of both. This effect is termed 'focal spread'. If nothing else, such an occurrence could emphasise an almost deliberate orthogonality between the manner in which two trees classify or store the same nodes. Such an occurrence is shown in Figure 6.11, using another example data set. Here, the selected groups are almost uniformly distributed in the groups of the bottom hierarchy, under ‘New Drude 2000’. This leads to no groups standing out in particular. In Figure 6.11 there can be seen the use of the larger nodes to display information, in this case the first couple of letters of the appropriate node names. This results in a pleasing ‘periodic table’-style look to the selected groups.

Another problem occurs with unbalanced tree structures such as those in Figure 6.10. Large groups of leaves prove difficult to display in relation to small groups and the opportunity to increase the size of such groups is limited. However, such structures present difficulties for all tree visualisations, and it is believed the method developed here handles them as well as any, especially as multiple examples are being dealt with simultaneously.

An interesting point occurs when the visualisation’s display layout recalculates itself after a selection. Comparing Figure 6.6 and Figure 6.7 as examples, it can be seen that there is a noticeable change in layout when nodes are selected. Misue *et al* [123] state that when layouts are altered it helps if the relative positioning (“above”, “to the right of” etc) of objects remains the same. This eases the process of reconciling the new layout with the old layout. The layout algorithm does that between hierarchies and between leaf groups in individual hierarchies, but not internally within individual leaf groups. Keeping the same shape for each leaf group would be inefficient in terms of screen space, and rather than use animation the colour of the selected nodes act as perceptual anchors within the groups. For example, the highlighted node in Figure 6.6 can be easily correlated with the highlighted node in Figure 6.7, helping a user map between the changes in layout for groups of interest.

A further point is that if space is at a real premium, there is potential for the DOI mechanism to perform elision on individual leaf groups. If the minimum DOI value is allowed to be zero, after a number of selections a leaf group’s total DOI would drop to zero if none of its constituent nodes had been selected, as each DOI is reduced slightly if its associated node is not involved in a selection action. The space allocation algorithm would therefore reserve no space for this group. The essential difference is that the zero minimum DOI choice removes context at a threshold, suppressing its output altogether, similar to Furnas’s [65] original display policy. A minimum DOI value above zero would mean leaf groups were always visible as context, and this is the current policy.

6.6.7 Alternative Layout Method

As stated, the original layout algorithm was derived from the Continuous Zoom method of space division: divide the available space according to the relative sums of DOI’s of the involved groups. While this made for an extremely space-efficient visualisation, it introduced, as noted, effects such as groups within the same tree having different sized nodes – even if they had the same average DOI. This was a consequence of one group fitting more neatly into its space than another. The other group would have wasted, unused space, the existence of which would not have been accounted for during the allocation process. The extreme example of this was the sets containing only a single node, resulting in the allocation of a vertical strip of space, of which only the top fraction was used to display the node. Some compensation was attempted for this, but it was not guaranteed to be even.

Bederson’s Quantum Treemaps (QT) [16] provoked the search for a layout that was more exact in reflecting the DOI values of the node groups, such that nodes with the same DOI value in different groups were displayed as the same size. Bederson’s QT approach is based on the notion that some space efficiency can be sacrificed in return for a more evenly aligned layout. Bedersons’ work also focuses on generating layouts with pleasing aspect ratios within a 2D Treemap, but as the set-based visualisation’s layout is essentially a strictly vertical layout of horizontally displayed trees, there is neither the necessity nor the opportunity to explore this facet of their work. Also, QT layout focuses on objects of fixed (quantum) size, equivalent in the set-based visualisation to the layout of tree nodes with no DOI values. However, the focus+context effect is based on node-associated DOI values, and thus an algorithm must be constructed that will accommodate groups that have differing node sizes to each other.

One method is to calculate an average node size for given quantities of space and nodes. This is a simple matter of division, and of course takes no notice of wasted space considerations. The layout is then calculated with this node size, and if at any point it exceeds the screen dimensions, the current calculation is scrapped and re-started with a slightly smaller node size. Not an elegant solution, but given its simplicity and the fact that it calculates at least an upper bound for the node size, it is fast and efficient. The procedure can be done for all trees, so nodes appear equal across trees, or per tree, so the nodes only appear equal within trees.

The difficulty comes with integrating DOI’s into the method. To do this a measure known as ‘equivalent average nodes’ is calculated for each group and group DOI within a tree as shown in Figure 6.12, building up to a total for that tree, and ultimately for the whole set of trees. The area available can then be divided by this total and thus the size of an average node is discovered for the layout of a tree with such varying DOI values. Then space equal to the ‘equivalent average node value’ for each group can be allocated.

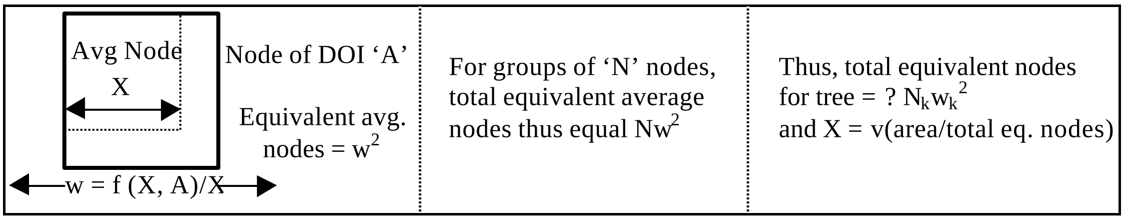


Figure 6.12. Calculating average node size for layout purposes with variable DOIs.

The problem lies in calculating these 'equivalent average node' values when using DOIs as *additive* factors to average node sizes, as occurs with the previous layout methods. With this approach, a node with DOI factor A (where A = `node DOI – average DOI`) is equivalent to $w^2 = ((X + A) / X)^2$ average nodes, where X is the side length of the average node size. However, it is this factor X that must be isolated, and whichever way you try the X cannot be removed or simplified out from this basic equation. Simply put, the number of average nodes a node or group of DOI A is equivalent to will depend on the size of the average node. But it is this measure of how big an average node will be which it is necessary to calculate.

An iterative Newton-Raphson method would, when given a likely starting value for X, undoubtedly home in towards an acceptable value. However, this issue is avoided by making DOIs a *multiplicative* factor, and not an additive one. So now, a node with DOI factor A (where A now equals node DOI / average DOI) is worth $w^2 = ((X * A) / X)^2$ average nodes. This simplifies to: $w^2 = X^2A^2 / X^2 = A^2$.

Therefore, for any value of X, a node with DOI A is worth A^2 equivalent average nodes of side X. The total of equivalent average nodes for the whole tree can then be worked out, and thus the 'average' node size for the layout computed. So the method remains workable even for groups of varying DOI values.

6.6.8 Conclusion

This work combines a focus + context technique with a linking metaphor across an example data set of multiple hierarchies. An initial node or sub-tree is selected in one hierarchy and the resulting focal effects for these nodes are also calculated for the other hierarchies in which they occur. Individual node DOI values are used to produce a uniform size for each leaf within a group of nodes to encourage a browsing behaviour with the visualisation.

The technique enabled effective interaction with larger data sets than was previously possible in the same display area. There are limitations to the technique, such as focal spread, as multiple focus + context or multiple view visualisations are both inherently more difficult to design than their singular counterparts.

The general idea of linking focus + context techniques can be applied to data which is represented through multiple views, which can be either multiple representations of the same structure, or multiple visualisations of data represented through different structures, though the latter requires relative layouts rather than absolute co-ordinate layouts

6.7 Fourth User Test

By the time of the final user test, the issues brought up by the third round of testing had been addressed. The visualisation had incorporated the linked focus+context mechanism previously described, and other solutions resulting from the observations made during the third round of testing.

For final testing, most IV evaluations have relied on a quantitative test to statistically validate a visualisation's claims e.g. Schaffer *et al* [153]. However, two points for the visualisation arise here. Firstly, relative measures such as speed require the test to include a comparable system against which to evaluate the visualisation. As stated, a comparison against the paper-based method to be fair is not considered to be a fair comparison. Secondly, it was also stated that speed of task completion is not of a particularly high relevance for these visualisations, as none of the users' tasks are primarily concerned with speed of operation. That leaves attributes such as error rates, that could be judged in isolation against an arbitrary marker (i.e. X errors per Y tasks.)

Certainly, performance measurements would be useful in the context of a visualisation having tasks that aimed for faster or more accurate performance, and which also had a comparable earlier computer-based system to judge relative success or failure against. However, perhaps a more relevant method for a novel visualisation would be to collect statistically valid amounts of questionnaire data from users on issues such as ease-of-use, enjoyability, and the frequency of use.

Subjective, as well as objective, data can be analysed using quantitative methods. Such data is obtained by having users complete a questionnaire that requires them to register opinions by selecting points on a numerical rating scale. Shackel [154] states that a combination of utility, usability and also *likeability* define the probability of a user being willing to accept a system, balanced against the social and financial costs of applying the system. Previous tests and requirements gathering had discovered most of the utility necessary for the visualisations, and many usability problems were also discovered and corrected. The final test will give a quantitative measurement of the objective usability performance of the system, and just as importantly, of the subjective likeability of the visualisation too.

6.7.1 Methodology

The methodology for the fourth test was as follows. A group of 19 taxonomists agreed to participate in the testing, which consisted of a pre-questionnaire which was distributed and returned beforehand, then tasks performed with the visualisation, and finished with two post-questionnaires, one being open-ended.

Analysis of the pre-questionnaires backed up previous thinking on the visualisation, also cementing the validity of the basic tasks the visualisation could perform. Asked whether they thought of multiple taxonomies as a single entity or as separate but related structures, 13 out of 17 respondents stated the latter. Three respondents hedged their bets by claiming aspects of both, and only one chose the former option. This vindicated the outcome of the first test, in which the multiple set-based visualisation gained favour over the merged, graph visualisation. It was also confirmed that accuracy, rather than speed, was the over-riding concern of the taxonomists' work, and that only three of the participants had previously encountered any other tools that displayed taxonomies graphically.

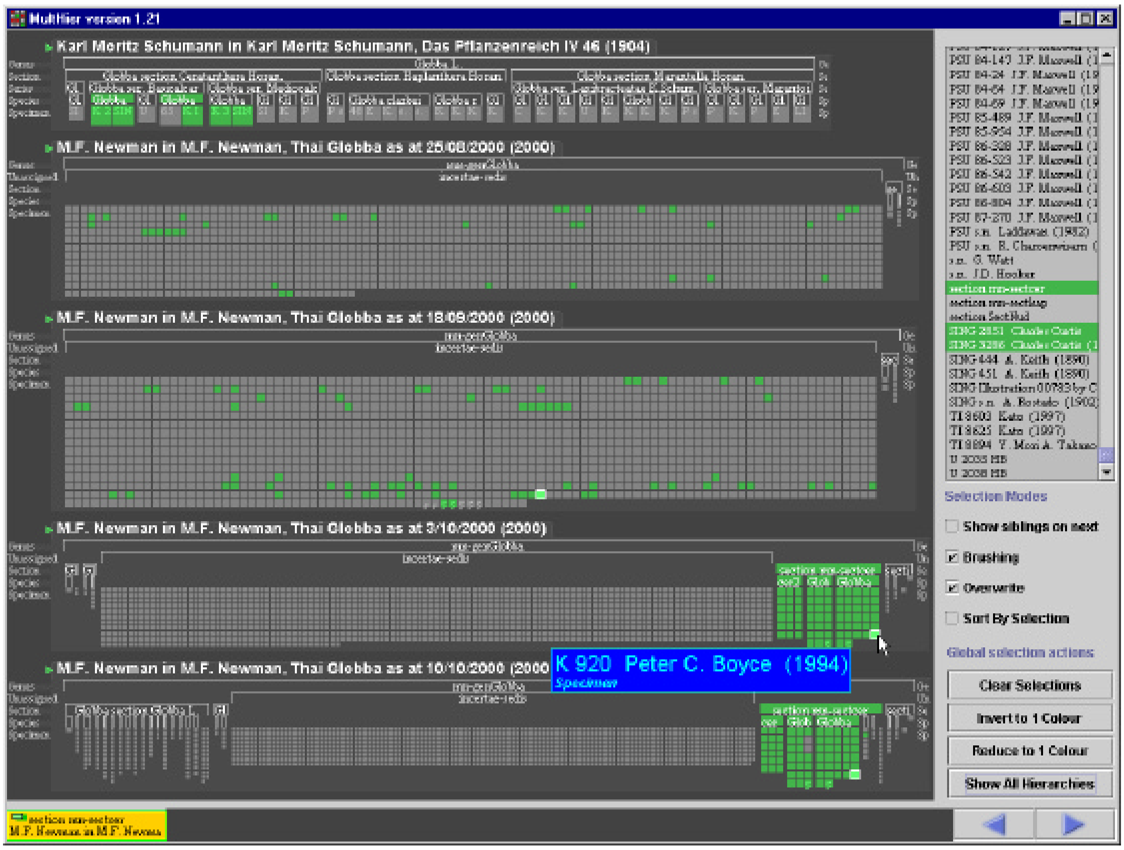


Figure 6.13. Visualisation as of the 4th User Test, using the *Globba* specimen data set.

The testing of the visualisation was again performed at the RBGE. The 19 taxonomists were divided into two groups. One group would attempt tasks using the original Apiaceae data set, and the second would use the Globba revision data set described in Chapter 3.5, which the visualisation had been unable to display adequately until the development of the linked focus+context technique. Inspection of the screenshot in Figure 6.13 shows an example of the patterns that appear when selections were made on the specimen taxonomies. Selecting a group that has been classified and taken from the large unassigned block shows the history of that group's construction. In this screenshot for example, the selected group has been formed over two distinct stages or revisions, with some specimens freshly classified in the last revision, but the bulk of the group was formed at the previous stage.

This between-subject approach was carried out so performances on the two types of multiple classifications identifiable in botanical taxonomy could be compared, namely the sets of historical classifications, and the stages of refinement during development of a new classification or revision. Participants were assigned such that previous users of the system (from the 3rd test) were distributed equally between the two groups. Therefore comparisons could also be made of previous against new users without the data set being an influencing factor.

It would have been preferable to have all participants try out both data sets as a within-subject experiment, so learning effects from one data set to the other could have been measured. However, this would have increased the time of each trial to almost two hours, and while the computing undergraduates typical to empirical testing can be arm-twisted or bribed into giving up such a substantial length of time, these participants were professional scientists of whom donating just an hour of time each was a considerable gesture. Also due to the tight schedule, preliminary training with the visualisation was not possible. However, the previous users, having used the system before, would give a measure of the likely effects of pre-training. Similar problems are commonly found in performing empirical testing in professional and corporate environments, an example of which is reported by Weiss-Lijn *et al* [171]. The decision was made that having less control over the length of the trials was an acceptable compromise in return for having *representative users* rather than just *users*. Neilsen states this to be the overriding factor when selecting users for trials.

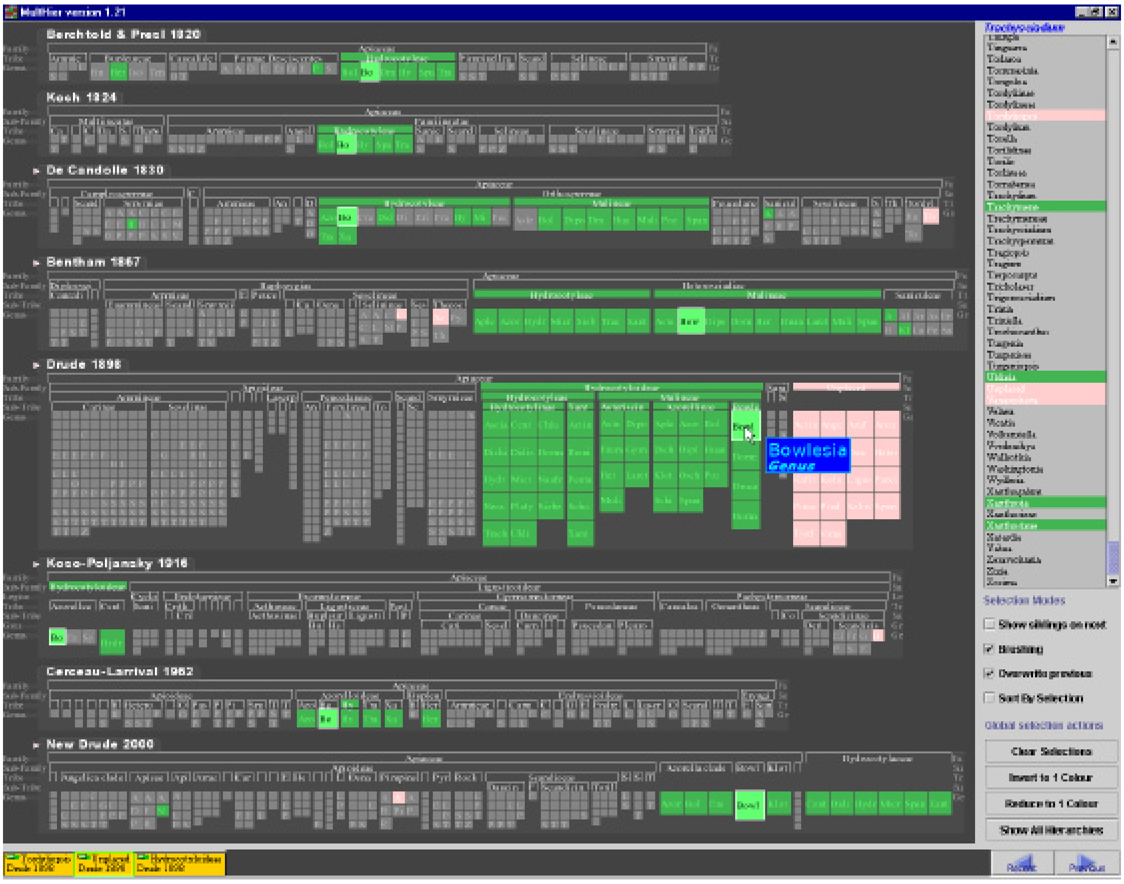
Prior to the testing, each participant was reassured that it was the visualisation and its interface that were being tested, and they as taxonomists were simply helping to test, not acting as the subjects of testing. The task sets, reproduced in Appendix D, were derived from the scenarios used in the previous user test, and as far as possible were the same for both data sets given the difference in construction and methodology. The tasks also asked for more concrete responses to questions in order to supply the necessary task completion data. As such, during the test, the participants were asked to give a verbal answer to each task point, as though the software log again picked up definite selections and brushing activities, it could not capture responses to purely visual inspections. The test monitor did not give help to the participants, as this would amount to training, but general comments were noted. Error rates were then calculated from the number of tasks correctly completed by each user.

Finally, subjective data in the form of an open-ended questionnaire and Brooke's Standard Usability Scale (SUS) [25] for each participant were gathered. In all, each trial took approximately just less than one hour, and no-one completed in less that half an hour.

6.7.2 Results

Analysis of the results revealed that users who had been presented with the Apiaceae data set achieved a 70% task completion rate, compared with 55% for those who worked with the Globba data set. The measured performance differences in terms of task completion between the Apiaceae and Globba set users came as no real surprise, as the suspicion had been that the larger Globba data set would be harder to work with than the smaller Apiaceae data set. The Apiaceae data set had 1,674 node representations compared to the Globba data set's 5,439 representations. As such it was expected that tasks with the Globba data would be harder to complete successfully just from a perceptual viewpoint, with each node having less than a third of the space available to its Apiaceae counterparts to make its presence known. The focus+context mechanism designed to give more prominence to nodes of interest would not have influenced the ordering of this result, as of course the technique was applied to the visualisation of both the data sets. The technique's intended effect was to make tasks on data sets of the scale of the Globba data set possible in the first place. The Globba data set, however, was composed of less fragmented structures than the Apiaceae classifications, being formed from a procession of refinements to an original data set. To say whether this feature played any part in the task completion rates would need further experimentation. The difference was extremely marked when comparing only new users on the Apiaceae and Globba data sets, with the Globba group of new users (45.3% task completion) performing considerably worse on task completion than the Apiaceae new users group (71.7%). The group of previous users was too small to state any internal differences regarding the effect of the type of data set with any degree of confidence.

Differences were measured between previous and new users of the visualisation, with, as expected, previous users being able to complete a greater percentage of tasks on average across both data sets. On average, previous users completed 72.5% of tasks successfully, compared to 57.5% for fresh test subjects.



☐ Show siblings on root
☑ Brushing
☑ Overwrite previous
☐ Sort By Selection

Figure 6.14. The *Apiaceae* data set as displayed in the 4th User Test.

The questions with the highest failure rate occurred in the scenarios dealing with sibling identification (Q15), counting of nodes within a group (Q3), and finding groups that were strict subsets of groups in other classifications using the brushing technique (Q18). The highest task completion rates occurred for Q’s 4 and 5, with only one failure each amongst the whole test group. These questions/tasks dealt with identification and tracking of single nodes through multiple classifications.

The SUS (Standard Usability Scale) results are presented in Appendix D, and they disclose two observations. Firstly, it establishes that the type of data set used in the trial made no significant difference to the user’s opinions of the system’s general usability. Secondly, there was a marked difference between the opinions of participants who had taken part in previous testing, and taxonomists for whom this was their first experience of the visualisation. Only one user whom hadn’t been previously employed in the tests gave a rating higher than any of the previous users. Questioning this taxonomist after the test, it was discovered that they considered the visualisation could have particular relevance to a set of problems they were currently working on. Encouragingly, even only one previous experience of the system greatly increased the perception of the system’s usability (35/40 median). New user’s reactions were ambivalent, scoring on average halfway along the SUS (21/40 median). Interestingly, there was no meaningful difference found between new users who had used the Globba data set and new users who had performed the tasks on the Apiaceae data set. This comparison had yielded the strongest differences in the analysis of the objective task completion results. The mean average for all users was 26.69 (2 d.p.) over 10 questions (25.5 median), giving an average of 2.67 (2 d.p.) per question on a 0-4 scale.

The questions with the highest cumulative score were Q's 6 and 8, asking whether users found the system to be inconsistent or cumbersome to use. The strong scores obtained after negating these questions indicated that the users felt strongly that the system was neither inconsistent nor cumbersome. The lowest cumulative score came from the responses to Q9: "I felt very confident using the system"; indicating that as far the subjective ratings went, confidence was the weakest factor.

A meta-analysis of subjective scale results by Nielsen and Levy [133] found a mean average rating of 3.55±0.12 on a 1-5 scale. Adding one to this result to move from a 0-4 scale to a 1-5 scale gives us a mean of 3.67, just within Nielsen and Levy's 95% confidence range. Therefore these results mirror the pattern for subjective scale results in general. The slightly higher than average (above 3) rating is accounted for by the user's tendency to be forgiving to an interface in subjective measurements, though a Likert scale's policy of asking alternate positively-worded and then negatively-worded questions, such as SUS, is meant to at least partially suppress such behaviour.

The subjective and objective differences between previous and new users point to a quality of the system that allowed the previous users to skip, or more accurately, not to have to re-take, the initial learning curve for the system. This quality, rather than being the more common learnability attribute, is the quality of *memorability* [131], where a casual user can return to a system after a period of time with little or no detriment to their interaction. Learning effects have much in common with memorability, but a way to distinguish the two is by thinking of a typical learning curve. Learnability will be a function of the gradient and shape of the curve. Memorability is a function of the ability to *maintain* a position on that curve after a long time period, somewhat akin to a base camp on a mountainside. Given that any previous users had experienced the system for three user tests at most, spread over the preceding 18 months (though one had used it frequently out of choice), it can be claimed that the visualisation has a good degree of memorability. Why this is the case is open to question, though perhaps the uniqueness of the visualisation compared to the normal Windows style interfaces had some bearing on this.

Also, asked if they found it harder to query the visualisation or interpret the outcome of a query in the visualisation, there was an almost even split in the user's reactions. Eight users said they found it harder to understand the input controls e.g. the syntax of the system, while five said understanding the resultant visualisation was tougher e.g. the semantics of the visualisation.

6.8 Conclusion

In conclusion to the development of this visualisation, a number of relevant points can be made. Firstly, metrics for visualisation-based interfaces [22] appear to be best suited for visualisations that are generally static in nature, without interactive filtering or focusing mechanisms, and composed of information of distinct dimensionality.

From the case study and development methodology, it can be concluded that the use of common techniques within an iterative design and evaluation of IVs produce roughly the same difficulties and results as they do for standard GUIs, apart from the following observations.

The users' preference for the visual representation of the information on-screen strongly reflected their own mental model of the information rather than the actual underlying structure of the information. This is of course true for traditional GUIs, but in IV-based interfaces this point increases in importance as the graphical representation of the information also forms the bulk of the interface mechanisms for interaction and display.

The initial user requirements were apt to undergo a great deal of revision as the initial visualisation prototypes encouraged the users to entertain new tasks or methods for working with the information at hand. This was particularly so in the taxonomists' case, as the information moved from a paper-based system, and was freed from the drawbacks associated with such a physically rooted system.

Standard video recording techniques for space-efficient IVs would be better replaced by either direct video screen-capture methods or by detailed software logging. Software logging was also found to be useful in conjunction with the note taking and the verbal protocol techniques, as it cleared up any ambiguities in the order of detailed interactions. These difficulties with the video recording and the ambiguities arose principally due to the fact that the visualisation contained densely packed interactive elements smaller than those found in a standard GUI.

Interestingly, with regard to speed of task completions, Chalmers, in Hascoët-Zizi *et al* [81], has previously predicted that a good information visualisation could persuade people to spend longer on tasks due to the IV making the task more enjoyable. Indeed, though the sample of 3 users is not sufficient to make any conclusive statement, the second test revealed that the most enthusiastic user spent twice as long performing the test as the other taxonomists. This was because the visualisation allowed a browsing-like behaviour, encouraging the user to explore parts of the data set at length that the scenario did not explicitly ask them to.

In the first user test, qualitative testing of the initial functionality of the prototypes and basic design choices was carried out, and resulted in the rejection by the taxonomists of the least appealing prototype, the graph visualisation. During the second user test, initial qualitative usability testing was performed along with the probing of the visualisation's capabilities with regard to scalability. Further functionality was also verified and video and logging approaches to capturing user responses were carried out with differing degrees of success. In the third user test a second phase of further usability testing was performed with six users at the RBGE with a web-based applet version of the visualisation, with the focus almost exclusively on usability issues, though as stated any errors or omissions in the functionality would be tackled. The difficulty of displaying a data set on a smaller monitor was noted, and this led to the development of the linked focus+context technique that was employed in the final test. The final user test consisted of a quantitative experiment at the final prototype stage, and indicated that the set-based visualisation allowed users with little or no experience of it to perform quite complex tasks and queries on data sets that would prove tedious and difficult to follow on paper. As expected, a larger data set caused users more problems than a smaller one in testing. The visualisation also exhibited the property of *memorability*, in that previous users were able to pick up their knowledge of the system after a lengthy period from where they had left off, not needing to re-learn the interaction.

7 Other Information Domains

The visualisation had now been developed and tested to completion on two taxonomic data sets, the multiple classifications of *Apiaceae* and *Globba*. It would now be interesting to discover what other information domains the visualisation could act upon, as multiple overlapping hierarchies can be produced by, amongst others, differing clustering algorithms, ontologies, and information collections organised by multiple indices.

To begin with, a table of the common visualisation-based tasks performed by the taxonomists is presented in Table 7.1. These tasks are described with respect to their physical performance in the visualisation, and their abstract meaning in the underlying data model and set theory. These abstract operations then form a collection that can be applied to other specific information structures that could benefit from the visualisation, so that questions can be asked about whether the combinations of operations and new information types appear to be useful and meaningful, or, alternatively, make no sense whatsoever. Thus, it can be discovered whether candidate information structures would benefit from the visualisation.

Categorisable, hierarchical data sets from domains other than botanical taxonomy were sought out. Two sets of dissimilar information were obtained. The first was a set of human genome data that had been classified by two different clustering algorithms, kindly supplied by Allan Kuchinsky. The second was a set of multi-dimensional data, composed of 14 dimensions, and had previously been visualised by Lanning *et al* with their MultiNav [111] system.

USER TASKS	VISUALISATION INTERACTION	UNDERLYING MULTIGRAPH OPERATION	SET OPERATION
Track a taxon[*]	Select leaf node in visualisation	Selects one node within the graph.	Member select
Track a taxa group[*]	Select internal node in visualisation	Selects all nodes directly descended from the selected node in the classification concerned	Subset select
Track a taxon's sibling groups[*]	Set sibling option 'on'. Select leaf node in visualisation	Selects all nodes that are child nodes of any of the selected node's parents. Therefore, this selection set is all nodes that share a parent node in at least one classification with the chosen node.	Selection of all immediate subsets which contain a given member
Compare classifications by common ranks	Conceal/Reveal appropriate ranks	Amalgamate child collections by next highest retained rank	Union of appropriate subsets
Compare by certain classifications	Conceal/Reveal appropriate classifications	Marks as invisible or visible a particular tree / DAG within the multigraph.	Disregard sets in given classification
Locate taxa unique to a classification	Select top-level node in all classifications except the classification of interest. Invert final selection for clearer picture.	Select nodes with only one parent link, and that link is in given classification.	Find members located in only one classification superset
Locate taxa first used in a classification[T]	Select top-level node in all earlier classifications than the classification of interest. Invert final selection for clearer picture.	Select nodes if first parent link occurs in given classification	Find members of classification set not found in previous classification supersets.
Compare distinct ranks[*]	Visual non-interactive operation		
Compare entire classification[*]	Visual non-interactive operation		
* = original task T = hierarchies must be ordered temporally			

Table 7.1. Task to abstract set operation mapping.

7.1 Human Gene Data

The genome data consisted of 3,500 gene descriptions (summary text strings rather than the actual gene sequences) arranged by two clustering methods into two corresponding hierarchies, and presented the same main challenge as the large Globba specimen data set, namely that of *scale*. At 3,500 nodes each individual hierarchy was larger than the specimen hierarchies, and both hierarchies formed DAGs even in isolation, though the data set had only half the number of classifications as the specimen data set. Therefore, in total, the visualisation would have to deal with over 7,000 node representations, five times more than the original Apiaceae botanical taxonomy data set, and 25% more than the 5,500 node representations of the Globba data sets. This feature of the data would need to exploit the combined linking and focus + context methods described in Chapter 6.6.

Both hierarchies were shallow structures with only one intermediate level, which consisted of the cluster categories. The clustering algorithms used had not produced tree structures, and had instead generated DAG structures as some genes were classified in more than one parent category. When combined, the hierarchies still formed the DAMG structure described as the overall logical model, but from an aggregation of DAGs, not of trees. It was later discovered the non-tree nature of the data was an error in the clustering algorithms, as tree data had been expected for the individual clustering hierarchies.

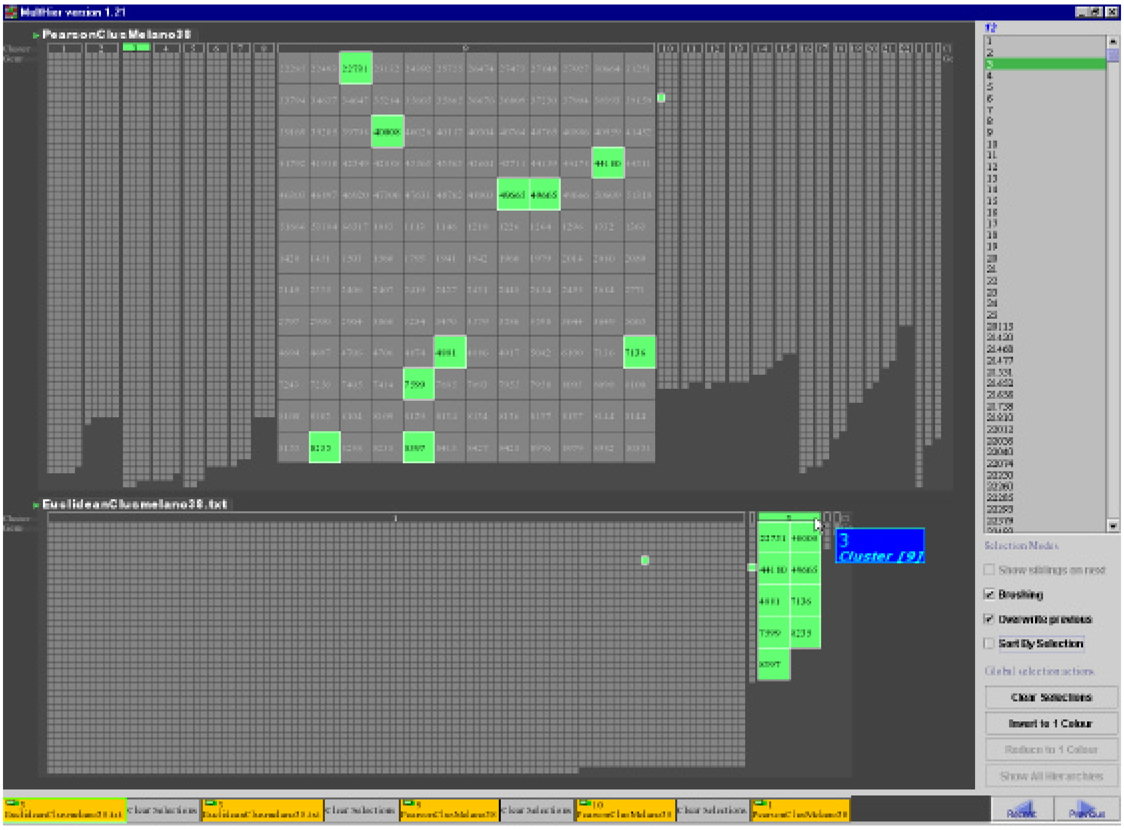


Figure 7.1. Gene data consisting of two hierarchies of approximately 3,500 nodes each. Nodes in the small highlighted group in the bottom hierarchy are also members of other groups in the same hierarchy i.e. this structure is a DAG.

The multiple set visualisation handles DAGs by having a many-to-one mapping of node representations from the view to the underlying model nodes, rather than a strict one-to-one relationship as used for tree display. Therefore, one node could be displayed at the appropriate multiple points in one tree representation, and interaction with any of those representations would affect the same node in the underlying DAG.

An example of DAG display is shown in the screenshot in Figure 7.1. Selection of a group in the bottom hierarchy has highlighted nodes in the top hierarchy, but also emphasised nodes in other groups in the bottom hierarchy. These other highlighted nodes are some of the same nodes as in the originally selected group, but are nodes with multiple parent categories and hence are given multiple representations. The visualisation shows the situation quite clearly.

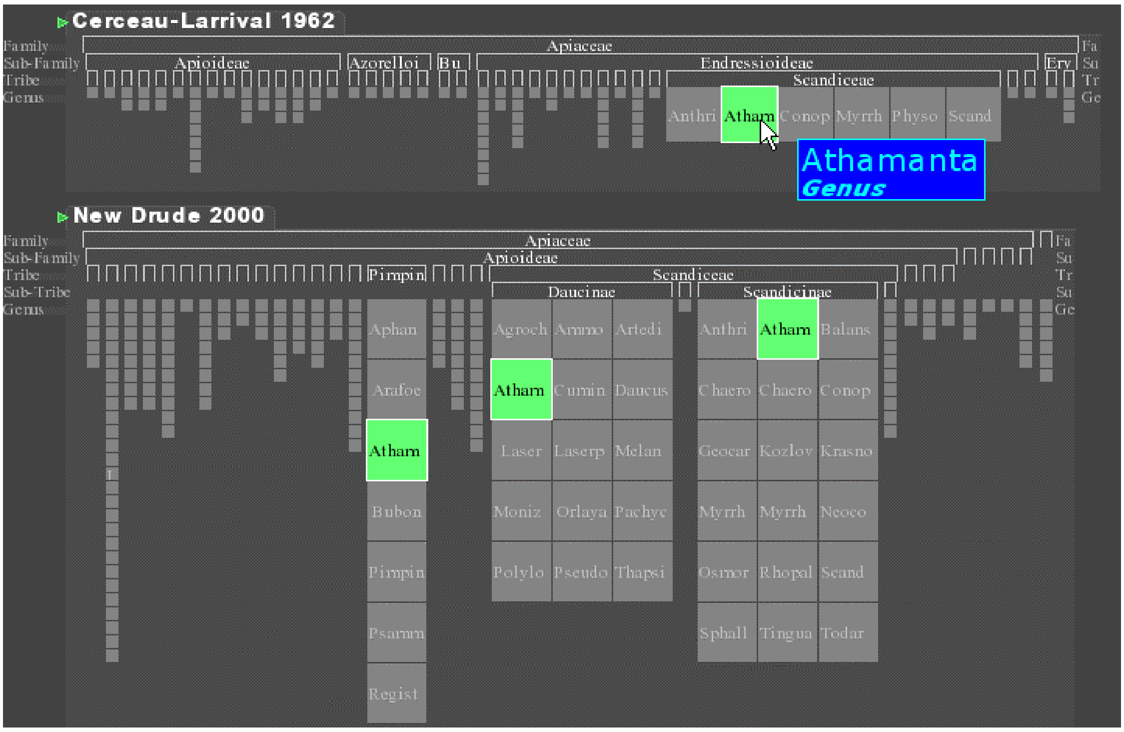


Figure 7.2. The genus *Athamanta* appears at several places within the New Drude taxonomy.

A similar situation can be seen in one of the Apiaceae taxonomies, New Drude 2000, in Figure 7.2. This was an incomplete taxonomy, with some taxa having undecided final classifications. The effects of not having the model adapted for multiple DAGs can be seen in Figure 6.3, where the New Drude classification contains several gaps where it appears that nodes should be present.

Therefore, the genome information can be displayed by the visualisation. Now, it must be decided whether the interaction operations are meaningful on such a data set. Taking the abstract set-based operations from the *Set Operation* column of Table 7.1, these can be analysed to see whether they would perform any useful function on the genome data.

Member Select would, in this context, track a single gene descriptor string between the representations of the different clustering algorithms, a simple but useful enough operation. *Subset Select* would track a group of these gene descriptions over the clustering algorithms, again a useful operation, having the same basic meaning as the function did in the botanical taxonomies.

Selection of all immediate subsets which contain a given member would indicate which gene descriptions another gene was ever grouped with. However, in the available data set, one hierarchy held a very large sub-group, containing 90%+ of the available data, such that most ‘sibling operations’ highlighted every representation within this group. For this particular information set, the operation was unwieldy, but could be manageable if data was obtained from other, more discerning clustering algorithms.

The *Union of appropriate subsets* operation was more problematic. Incorporated in the visualisation to enable the effects of internal taxonomic ranks to be disabled or enabled, it was found that hiding particular depths *across* the clustering hierarchies was not meaningful. Hiding levels such as *sub-family* is meaningful across multiple taxonomic classifications as taxonomies share a common framework of assigning meaning to depth in a structure. However, clustering methods work on a variety of principles, and thus the structure semantics resulting from the application of the different algorithms to the genome data will tend not to be the same. Merging subsets by removing certain depths or ranks will therefore only have an individual meaning internal to each hierarchical structure and not across the entire set of structures.

Disregard sets in given classifications had the same semantics as it did for the botanical taxonomies. Hierarchies of interest could be revealed or concealed visually, which in turn decided their participation in certain operations.

Find members located in only one classification superset was not applicable to this information set. All the gene descriptions were present in all the clustering algorithms, and therefore no gene descriptions occurred in only one clustering result. As such, the operation to find where nodes first occurred was doubly redundant, as on top of the fact that all the nodes occurred in all the hierarchies, there was no chronological ordering.

The purely visual operations such as comparing the number of levels in each clustering algorithm were still relevant.

The more complex interactions proved to be less meaningful. This is unsurprising, as while basic operations would tend to have common ground across data sets, more complex operations would have evolved to suit particular tasks on a particular data set.

7.2 Multi-Dimensional Data

The second data set was in contrast to the large-scale but simple classifications of the gene data. This data set consisted of less than 200 unique nodes, but they were nodes comprised of multi-dimensional data, specifically relating to various properties and attributes of digital cameras. In all, each item had up to 14 associated dimensions.

The MultiNav technique of Lanning *et al* visualised the data as individual dimensions, organised vertically in rows, with each item having a corresponding, ordered-by-value, horizontal position along each of the dimension 'rows'. Selection of an item or group of items in one dimension will highlight the node's positions within the other displayed dimensions. In their "Sliding Rod" prototype, the dimensional rows move horizontally (the sliding rods), in an automatic adjustment, in a manner that makes the dimensional attributes and values associated with a particular selected item form a vertical alignment at the centre of the screen. The dimensional rods can also be moved by hand, causing whatever item is at the centre of the screen to change, and thus causing all the other rods to adjust to bring that item's other attributes to the centre of the display.

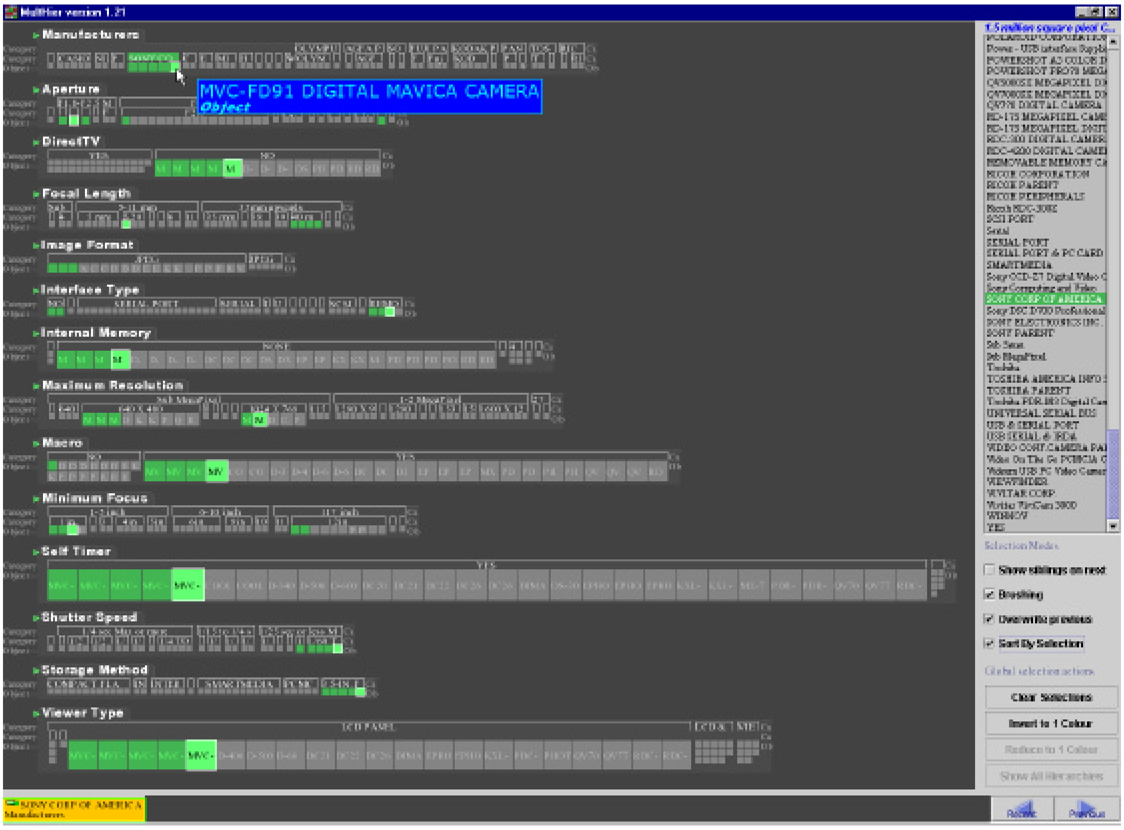


Figure 7.3. Multi-dimensional data set. Each hierarchy is a dimension, with 14 dimensions in total. This screenshot shows the attributes of Sony cameras in green, and the attributes of one particular camera is highlighted.

In the set-based visualisation, the multi-dimensional information was visualised by converting each dimension into a shallow hierarchy, as shown in Figure 7.3. For instance, the manufacturer dimension was divided into EastMan-Kodak, Fuji, Minolta etc, and then sub-divided if necessary (Fuji Japan, Fuji US).

The abstract set operations had different, but applicable, meanings in the context of this information. *Member Select* simply mapped to tracking a particular object through the multiple dimensions i.e. picking out its full profile in the dimensional space. *Subset select* mapped to choosing a sub-group within a particular dimension and observing those groups' members positioning throughout the other dimensions. The *Selection of all immediate subsets which contain a given member* in this context would pick out all objects that shared at least one dimensional sub-grouping with the target object, in effect posing the query, "show all objects that share at least one close attribute with the target object." This would be useful in conjunction with the *disregard sets in given classification* operator. This operator itself would remove certain dimensions from participating in other operations, limiting them to a subset of chosen dimensions.

The multi-dimensional data set had the same problem with the rank-hiding mechanism (*union of appropriate subsets*) as the genome data set did, in that unless the depth semantics were constant across hierarchies the operation would not mean a great deal. As dimensions would be composed of different units and orderings, there could not be a constant method of decomposition across them all. Hence, it can be said that this particular operation is only useful when comparing hierarchies that have been re-organised within a common framework, as the taxonomy data is.

Find members located in only one classification superset, with this information set, found objects with a presence, or positioning, in only one dimension. This seems to have no practical application here, apart from to spot erroneous objects. Similarly, as the dimensions cannot be chronologically ordered, finding a first occurrence of an object within the dimension set is nonsensical.

The selection metaphor for the set-based visualisation differs from MultiNav as their visualisation performs selection as logical OR's within dimensions, but subsequent selections in other dimensions are treated as logical AND's, filtering down the previous selection of items. The set-based visualisation always performs selections as logical OR's, a result of the request for the taxonomists to compare different groups in different hierarchies. The metaphor used when selecting from multi-dimensional information as per Lanning *et al* is different however, in that the user will mostly wish to narrow down the amount of original selections to find an appropriate object, not expand the choice out.

The overall effect, as with Lanning *et al*, is strongly reminiscent of a line-free parallel co-ordinate display (as is observed by Gary Ng [129]), but one where the dimensions are divided into discrete sections as shown in Figure 7.3.

This last observation highlights a difficulty with using this data set in the multiple hierarchy visualisation. Some of the dimensions in the data set such as price are continuous variables, and the visualisation, indeed partitioning structures in general such as hierarchies, operate best on discrete information. The solution was to place ad hoc categories on top of the continuous dimensions, which would classify these variables according to range. For instance, price would be grouped into categories such as: under 200, 200-400, 400-600 etc. Conversely, when a dimension intrinsically clusters items in discontinuous bunches with meaningful qualities, the hierarchy imposed on it makes simple the selection of similar groups of objects within a dimension. Recent extensions of the parallel co-ordinates technique such as the hierarchical parallel co-ordinates metaphor developed by Fua *et al* [63] can achieve the same effect.

Another problem when using a dimension with a continuous scale can be seen in Figure 7.4. In the visualisation, leaf nodes and categories are ordered numerically for integer data sets, and alphabetically otherwise (so 12 doesn't come after 111 but before 132.) Thus, with a continuous scale, albeit snapped into digestible sections, it would be expected that there would be a directional precedence of order, as in a normal scale. However due to the wrap-around grid pattern that the visualisation uses for positioning leaf nodes into sets, this trend may not be truly reflected. Though it may not be a major problem, it does mean that visually comparing two leaf nodes solely by position in a continuous dimension can be deceiving, and care should be taken.

Figure 7.4. Ordering in grids is left-to-right and then wraps-around top-to-bottom. So, the green node on the right-hand side is ordered alphabetically before the coloured nodes on the left-hand side, as those nodes are positioned one row lower down.

In common with Inselberg and Dimsdale's technique, there can also be difficulty in differentiating between two objects when they share the same attribute in one dimension. Their problem was visualised as an ambiguous line-crossing problem, as when two lines crossed it was impossible to say which line was which after the intersection. A solution used by many parallel co-ordinate viewers since is to use colour or a brushing technique to distinguish different poly-lines in the dimensional space. In the set-based visualisation, the problem occurs when two nodes of the same colour are visualised. The node representations are always visualised unambiguously in spatial terms i.e. they do not share co-ordinate positions as Parallel

Co-ordinates do, but they don’t have the connecting lines that can link representations from one hierarchy or dimension to the other. Instead, the visualisation relies on using a different colour or the brushing technique used for the general viewing of multiple hierarchy data, and with the node representations having significant sizes when selected, brushing is more accurately performed than on the line-width representations of Parallel Co-ordinates. Also, thanks to the linked focus + context technique, selected nodes are given larger allocations of area, and in this area can be squeezed in the first letters of the node’s unique name. These can differentiate, depending on the similarity of the node names, identically coloured nodes across the hierarchies without brushing.

7.3 Application of Visualisation and Operations to New Domains

In conclusion it can be stated that the set-based visualisation managed to handle two very different types of information set with a varying degree of success. The gene data set, comprising of two hierarchies holding over 7,000 node representations in total, pushed the limits of the visualisation as to the amount of on-screen data it could handle, and also the type of individual structure it could use. The first observation benefited from the development of the linked focus + context techniques described previously in Chapter 6.6, a new method for efficiently using space when displaying overlapping, multiple hierarchies. The second feature of the data, the DAG hierarchies, along with the latter taxonomic structures, prompted the extension of the multiple hierarchy model to handle multiple overlapping DAGs as well as multiple overlapping trees.

The multi-dimensional data set allowed the visualisation to simulate a Parallel Co-ordinate style visualisation. There were problems such as multiple hierarchies and dimensions having different data expectations (discrete v. continuous), and differing styles of interaction, but the fact that the visualisation could perform the majority of tasks demonstrated by the MultiNav application was an unexpected bonus. It was not a situation, or use of the visualisation, that would have anticipated before the data was supplied.

Table 7.2 gathers together the meaning of the basic visualisation operations on the different data sets that were rendered in the set-based visualisation (plus their theoretical meaning on multiple file directories). All the listed operations, with the exception of the brushing actions, are additive logical ‘OR’ operations, as they accumulate on top of previous selections that have been made in the visualisation. As discussed previously, multi-dimensional IV exploration also benefits from logical ‘AND’ operations, in which intersections rather than unions of selections are made, in an effort to narrow down the selected item set. The set-based visualisation does not support such an operation mode, though could be programmed to do so.

The basic interactive operations, such as member and group selections, proved more transferable from the original taxonomy data sets to the new information structures. More complex operations, such as finding unique occurrences of nodes, appear to have evolved to suit the initial taxonomic information, but as they are built up from the more basic operations rather than being fundamental operations, this is no great drawback. Other information sets could thus take advantage of the basic operations, and build up useful complex operations to suit the particular information at hand.

Thus, after development, the set-based visualisation technique managed to demonstrate an ability to diversify from multiple botanical taxonomies to further information sets, such as the multiple gene clusters, though in some ways the tool was over-specialised for the task of viewing taxonomic structures. It was also extended to become a multi-dimensional interactive tool, in which objects could be variously classified according to distinct sets of hierarchical schema or attributes, but a specialised visualisation technique would naturally be expected to out-perform a tool developed for another domain. These are grounds for stating the visualisation has some merit as a versatile multiple hierarchy tool, supplying at least a subset of rational operations that can be applied across many similar information structures to the original multiple taxonomies.

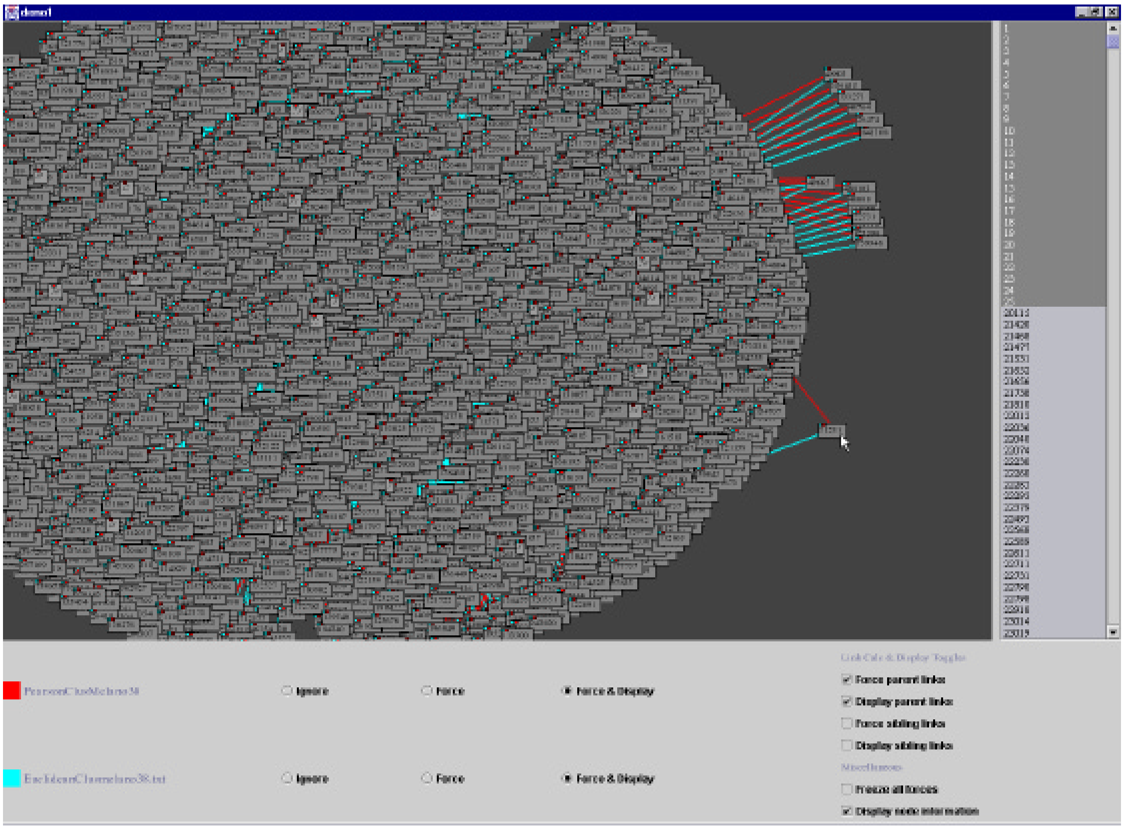
Multiple Tree Visualisation Operation	***Equivalent Underlying MultiGraph Operation***	***Meaning of Operation on Multiple Taxonomies***	***…Dimensions***	***…Gene Clusters***	***…File Directories***
1) Selection of one leaf node in one hierarchy.	*Selects one node within the graph.*	Shows where one taxon appears in the other hierarchies.	Show dimensional attributes of that object.	Shows where that gene occurs in the gene clusters generated by the different algorithms.	Shows where one file appears in the multiple file hierarchies.
2) Selection of a group of nodes in one hierarchy.	*Selects all nodes directly descended from the selected node in the classification concerned.*	Shows re-organisation of taxa in the context of other taxonomies.	Shows distribution of objects with similar values in one dimension throughout the other dimensions.	Shows correlation between different clustering algorithms.	Distribution of sub-tree directories in one file structure in other file structures.
3) Selection of all siblings of a particular node.	*Selects all nodes that are child nodes of any of the selected node's parents. Therefore, this selection set is all nodes that share a parent node in at least one classification with the chosen node.*	Shows associated taxa of a particular taxon for each hierarchy. Then shows the distribution of these taxa through all taxonomies.	Shows distribution of objects that are similar in at least one dimension to a given object node.	Shows distribution of the clusters, of which a given node is a member, throughout all the different clustering hierarchies.	Shows distribution of files that are in the same directory in at least one file structure as the given target file.
4) Hide / Show hierarchy.	*Marks as invisible or visible a particular tree / DAG within the multigraph.*	Hide / show a particular taxonomy. Toggle its participation in sibling operations.	Hide / show a particular dimension.	Hide / show a particular clustering hierarchy.	Hide / show a particular file structure.

Multiple Tree Visualisation Operation	***Equivalent Underlying MultiGraph Operation***	***Meaning of Operation on Multiple Taxonomies***	***…Dimensions***	***…Gene Clusters***	***…File Directories***
5) Hide / Show nodes at a given depth.	*Mark as invisible or visible all nodes with the particular chosen value for its depth property. This property is often not the same as depth from the root.*	Hide / show taxa at a particular rank, re-grouping lower level taxa as appropriate.	Hide / show a particular granularity of grouping. Not meaningful from dimension to dimension.	Hide / show a particular granularity of clustering. Not meaningful across clustering algorithms,	Meaningless. Depth in a file structure indicates distance from root but indicates nothing meaningful in terms of the objects at such a level.
6) Locate nodes unique to a hierarchy	*Select nodes with only one parent link, and that link is in given classification.*	Locate taxa unique to a classification	Locate object with only one dimension.	Locate object clustered by only one algorithm.	Locate file that appears in only one file structure.
7) Locate nodes first used in a hierarchy	*Select nodes if first parent link occurs in given classification*	Locate taxa first used in a classification	Not meaningful.	Not meaningful.	Locate files that first appear in this file structure snapshot.
8) Brush mode. Temporary action of operations 1), 2) or 3) when pointer is hovered over a node.	*Temporary marking of nodes. Same action as operations 1), 2), or 3). Visually, in the multiple tree representation brushed nodes are given extra brightness rather than a colour.*	Temporary selection of taxa.	Temporary selection.	Temporary selection.	Temporary selection.

Table 7.2. The meaning of the basic set-based visualisation operations on differing data sets

7.4 Graph Prototype Revisited

The new information structures demonstrated with the set-based visualisation also served to exacerbate the problems initially found with the graph-based visualisation, especially with regard to its interactivity. The larger data sets of the gene data and specimen revisions slowed the visualisation down to a point where large delays were introduced between issuing a command via the mouse and observing the visualisation respond. These data sets also took at least a couple of minutes until they settled down into a comfortable initial configuration, even with the adaptation of Chalmers' linear time algorithm. Interesting and visually arresting visualisations of the data sets could still be obtained, especially for the gradual revisions of the specimen taxonomy, where different stages of the emerging structure could be seen as being 'expelled' from the main body of nodes. Interaction, though, proved frustrating in the extreme.


🟥 ○ Ignore ○ Force ● Force & Display

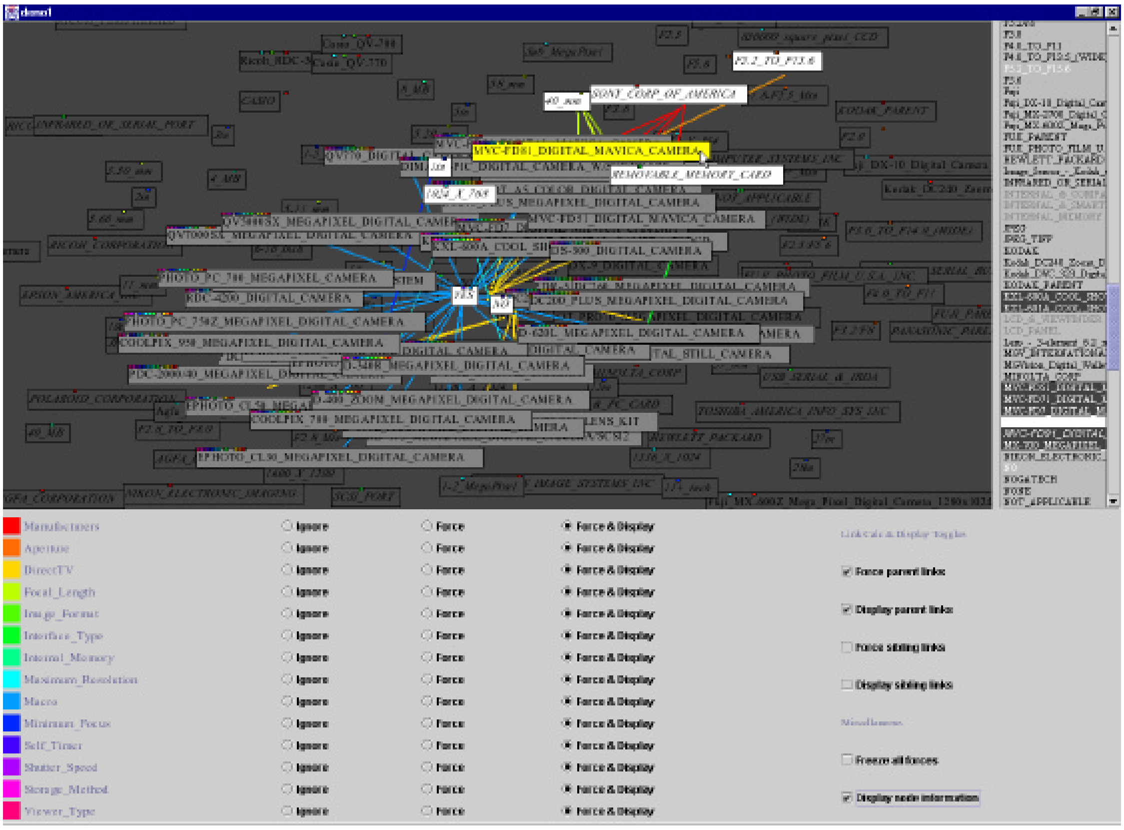
🟦 ○ Ignore ○ Force ● Force & Display

☑ Force parent links
☑ Display parent links
☐ Force sibling links
☐ Display sibling links
☐ Freeze all forces
☑ Display node information

Figure 7.5. "Death Star" - graph visualisation of the two massive gene data cluster groups.

The problems were even worse when visualising the two gene cluster data sets as in Figure 7.5. Interaction was slow, the display was swamped, and many nodes were completely occluded at a zoom level that allowed an overview. The large group seen in the second hierarchy of the set-based visualisation of Figure 7.1 acts to pull the nodes together in the graph visualisation into a uniform clump. Only the tiny clusters present in the second hierarchy can be distinguished, escaping off to the right.

The multi-dimensional data set provided different problems. Objects that shared the same values in certain categories could be found by selecting the relevant parent node, and dimensions could be included or ignored by switching on or off their participation in the display and spring-mass algorithm. Similarly to other multi-dimensional visualisations adapted to use a force-based metaphor, objects that shared common attributes tended to be clustered together. In this respect, the graph visualisation performed well, and the small size of the data set and interactive nature of the visualisation combined to allow new layouts and thus comparison according to dimensional subsets to be made quickly and easily.



Manufacturers ○ Ignore ○ Force ● Force & Display
Aperture ○ Ignore ○ Force ● Force & Display
DirectTV ○ Ignore ○ Force ● Force & Display
Focal_Length ○ Ignore ○ Force ● Force & Display
Image_Format ○ Ignore ○ Force ● Force & Display
Interface_Type ○ Ignore ○ Force ● Force & Display
Internal_Memory ○ Ignore ○ Force ● Force & Display
Maximum_Resolution ○ Ignore ○ Force ● Force & Display
Macro ○ Ignore ○ Force ● Force & Display
Minimum_Focus ○ Ignore ○ Force ● Force & Display
Self_Timer ○ Ignore ○ Force ● Force & Display
Shutter_Speed ○ Ignore ○ Force ● Force & Display
Storage_Method ○ Ignore ○ Force ● Force & Display
Viewer_Type ○ Ignore ○ Force ● Force & Display

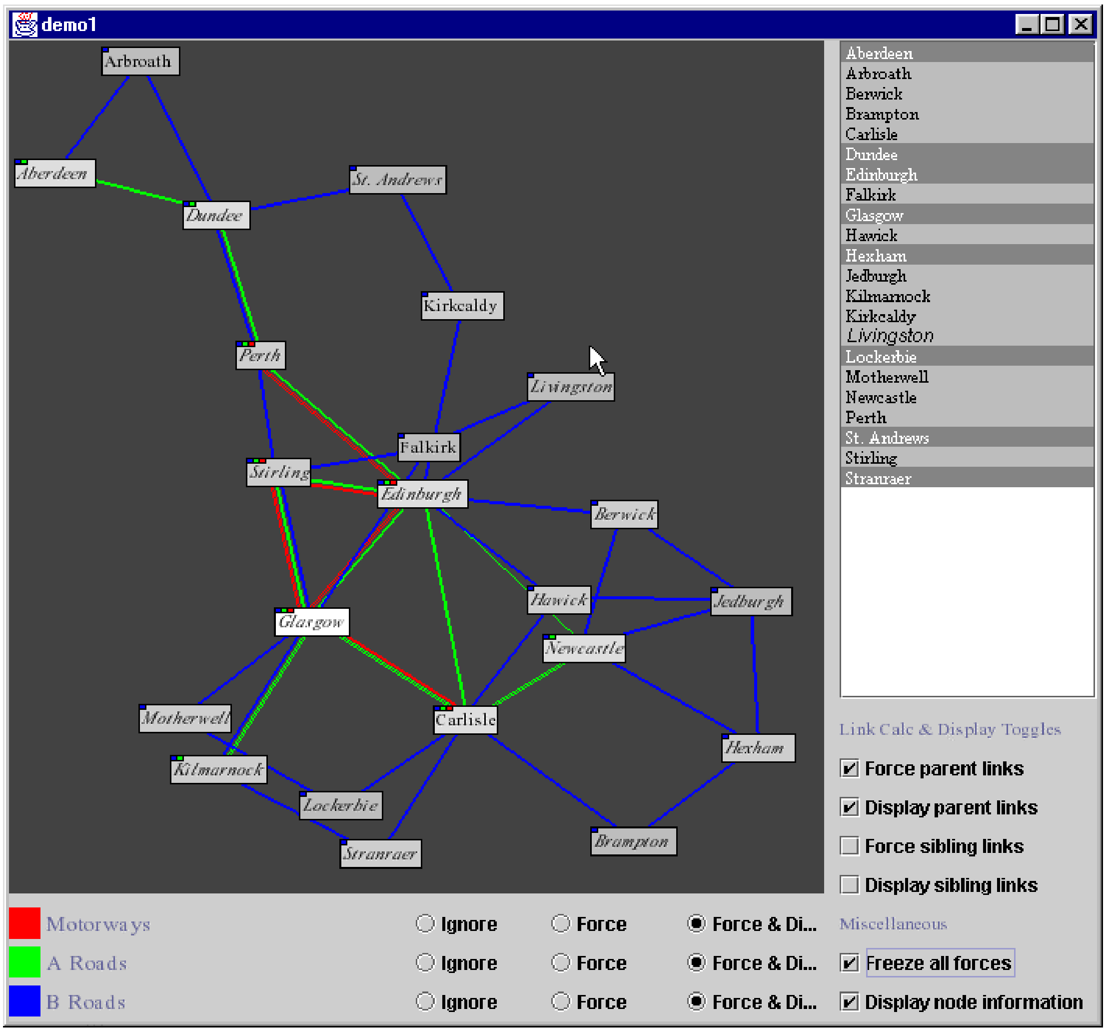
Link Calc & Display Toggles
☑ Force parent links
☑ Display parent links
☐ Force sibling links
☐ Display sibling links

Miscellaneous
☐ Freeze all forces
☑ Display node information

Figure 7.6. Hard to distinguish colours along with label occlusion make this graph visualisation of the multi-dimensional data set difficult to interpret.

The problem for this data set visualisation was the use of colour to differentiate between different hierarchies, or, as in this case, dimensions. Human perception experiments such as Healey's [82] have shown that the human eye can only successfully differentiate about seven colours, possibly nine, when the colours are not placed adjacently for direct visual comparison. As the multi-dimensional data set had fourteen dimensions, each colour used in the resulting visualisation was hard to distinguish from a range of similar colours, as shown in Figure 7.6. Consequently, it became difficult to discern which links corresponded to which dimensions.

There was one feature the graph-based prototype possessed that the set-based visualisation could not stretch to. It was found that the graph prototype could be extended to handle multiple overlapping *graphs*, where each graph formed a partially overlapping subset of a larger structure. This was something the set-based prototype could not show, restricted as it was to displaying individual hierarchical representations. An example of overlapping graphs can be seen in Figure 7.7, showing overlapping topographical networks of roads and towns.



🟥 Motorways ○ Ignore ○ Force ● Force & Di...

🟩 A Roads ○ Ignore ○ Force ● Force & Di...

🟦 B Roads ○ Ignore ○ Force ● Force & Di...

Link Calc & Display Toggles

☑ Force parent links

☑ Display parent links

☐ Force sibling links

☐ Display sibling links

Miscellaneous

☑ Freeze all forces

☑ Display node information

Figure 7.7. Overlapping non-hierarchical graphs in the graph visualisation prototype.

In summary, the graph metaphor eventually succumbed to technical and perceptual limits, as well as the poor initial reception it received in the first stage of user testing. The colour difficulties could possibly be overcome by using other cues such as texturing (dotted or dashed lines), or assigning colours only to a dimensional sub-group of interest. However, the technical difficulties of speed and scale are more intrinsic to the force-based approach of the visualisation. Given that the algorithm has already been heavily optimised, it appears that the highly interactive, force-based approach to graph visualisation must wait for the hardware to develop before it can tackle data sets on a larger scale.

So, while the new information sets highlighted some of the strengths of the set-based visualisation, the graph-based visualisation was observed to buckle in different directions under the demands of the genome and multi-dimensional data. Whilst it could then also be extended to handle multiple overlapping graphs, the flaws that worked against the graph visualisation with the *Apiaceae* data set became all too obvious when exposed to the additional information sets here.

8 Conclusion

This thesis has described the successful design, development and testing of a visualisation technique to explore multiple, overlapping hierarchies, using plant taxonomies as reference data sets. This final chapter discusses two aspects of the work that merit further examination and discussion. Firstly, the positioning of the set-based and graph-based visualisations within the population of other IV techniques and applications, and secondly, to delineate the contributions the prototype visualisations and their development have made to the IV field.

8.1 Placement in IV Space

One question to be asked is where the visualisations sit within the existing body of IV research. As stated in Chapter 2, there is no agreed, overall structuring to the IV field. Techniques and visualisations can be categorised according to a variety of attributes, so a selection of IV classification spaces are assessed for their suitability in describing both visualisations developed in this thesis.

The visualisations are designed to display a type of graph structure that exists between a tree and a full graph, so applying a general classification criterion such as Shneiderman's [155] does not reveal anything that isn't already known. Indeed, the fact that the set-based visualisation can also reasonably handle some multi-dimensional data sets means that it straddles two of Shneiderman's data type categories. As far as tasks are concerned, both visualisations allow overviews of the information sets, along with 'details-on-demand', and both are fundamentally designed to relate (link) similar information. The graph-based visualisation employs both zoom and filter mechanisms, while the set-based visualisations focus+context mechanism can be loosely classified as zoom and filter too, enhancing as it does the relevant areas of information. Only the set-based visualisation supports a history function of previous actions.

Later IV classification schemas such as the IV Design Space of Card *et al* [29] and Chi's IV taxonomy [39] cover a more general swathe of IV techniques and data domains. This gives them more scope but also reduces the coverage and detail they give to tree and graph structures. For instance, Chi's IV taxonomy groups all tree-drawing methods together, as it classifies techniques by the stages of data transformations and abstractions used to produce a final visualisation from the raw data. The vast majority of tree visualisation systems only differ in the final presentation stages, and as such he aggregates their descriptions. In his classification both the set-based and graph visualisations appears quite undistinguished, as the data structure to be visualised is, for the main extent, unchanged until the final presentation of the data.

IV Design Space provides a framework for analysing and describing how differing types of raw data sets are converted into visual properties on-screen. Trees and graphs are treated as having the same basic structure of nodes and interconnections between those nodes, and for the most part are described as being mapped onto an unquantified combination of spatial variables. As can be seen from the resulting descriptions of how three of the most common tree visualisations map their tree structures to on-screen representations, there is not much to distinguish between them. The fact that TreeMaps displays a tree as a nested structure is marked simply by the asterisk in the '[]' column, and Cone Trees' 3D technique is distinguished by the 'XYZ' co-ordinates of the transformed data as compared to the 'XxY' of the Hyperbolic Tree and TreeMap. Interactive properties are not considered. In short, the IV Design Space classification tends to distinguish multi-dimensional data sets more clearly than tree or graph visualisations.

Reproduced from Card *et al* [29]:

Cone Tree

Name	D	F	D'	X	Y	Z	T	R	-	[]	CP
Files	NxN	Tree	XYZ	*	P	*			L		tx

Tree Map

Name	D	F	D'	X	Y	Z	T	R	-	[]	CP
Files	NxN	Tree	XxY	*	*					*	

Hyperbolic Tree

Name	D	F	D'	X	Y	Z	T	R	-	[]	CP
Set	NxN	Hb	XxY	*	*				L		

Noik's [134] earlier work on the classification of graph and tree visualisations was based solely on presentation and basic interaction techniques, but this is suitable for us as presentation techniques is where the bulk of this thesis work has been concentrated.

Approach	Date in literature	Transformation gv/vv	Emphasis technique i/f/d/a	Priorities s/A/D/a/d	FP's 0/1/n	Ani?	Inputs s/h/g/n/*
Non-Graph Oriented							
Tree Maps [96]	10/1991	¦ ?	? ? ¦ ?	¦ ? ? ? ?	⦿○○	?	¦ ¦ ? ? ?
Perspective Wall [117]	04/1991	¦ ?	? ? ¦ ?	? ? ¦ ? ?	○⦿○	¦	¦ ? ? ? ?
Graph Oriented							
Set-Based Vis	05/2000	¦ ?	? ? ¦ ¦	? ¦ ¦ ? ?	○○⦿	?	¦ ¦ **?** ? ?
Cone Trees [150]	04/1991	? ?	¦ ? ? ?	? ? ? ? ?	○⦿○	¦	¦ ¦ ? ? ?
Graph Vis	05/2000	¦ ?	? ¦ ? ?	? ? ? ? ?	⦿○○	¦	¦ ¦ **?** ? ?
Continuous Zoom [11]	05/1994	? ¦	? ¦ ¦ ?	¦ ¦ ¦ ? ?	○○⦿	¦	¦ ¦ ¦ ¦ ?
ZTree [12]	05/2000	¦ ?	? ¦ ¦ ?	¦ ¦ ¦ ? ?	○○⦿	¦	¦ ¦ ¦ ? ?
MultiTree + TreeMap [174]	12/1995	¦ ?	? ? ¦ ¦	¦ ? ? ? ?	⦿○○	?	¦ ¦ **?** ? ?
Web Ecology Vis [40]	04/1998	¦ ?	¦ ? ? ¦	? ? ? ? ?	○⦿○	¦	¦ ¦ **?** ? ?

Table 8.1. Common tree and graph visualisations as classified by Noik's dimensions

As such, an investigation was made of where the visualisations would fit within Noik's dimensions of *transformations, emphasis techniques, priorities, focal points, animation,* and *inputs*, as shown in Table 8.1. These six dimensions, in theory, combine to give 43,200 different descriptions of a graph visualisation, though Noik acknowledges that some of them are not orthogonal. As such there can be expected to be considerably less than 40,000 graph visualisations waiting to be constructed, possibly of an order or two of magnitude less, given his classification.

The *transformation* dimension is based on whether the visualisation is obtained from raw graph data, as is the case of both the set-based and graph visualisations, or from a technique that allows enhancement of a standard graph view, such as Bederson and Hollan's Pad++ environment [18]. Both result in what Noik terms an 'emphasised view' where, either implicitly or through interaction, the user can emphasise particular pieces of information. It can also be seen that the latter type of transformation, that of an enhanced view being an adaptation of a standard view, is a more general notion of the first-order / second-order descriptions of normal and focus+context views put forward later by Björk *et al* [21]. Both the graph and set visualisations are obtained from the underlying graph model, and are thus classed as *gv* (graph ? visualisation) as opposed to *vv* (visualisation ? visualisation).

The *emphasis* dimension describes how the visualisation gives visual priority to certain subsets of information. Noik lists three classes of emphasised view: *focused (d), filtered (f),* and *adorned (a)* (retinal variables such as colour, texture and shape), along with an *implicit (i)* category, such as those found in 3D projections where objects at the front of the field of vision attract the users' attention more readily. It can of course be argued that 3D techniques could come under the umbrella of focus techniques, simply being a special case of the technique that humans are familiar with. The graph visualisation uses filtering through the inclusion and exclusion of certain hierarchies and the filtering out of all links apart from those directly concerned with a chosen node. The set-based visualisation combines a distorted, focused view with an adorned display, and is unusual in that it is one of only a few to use an adorned view, namely the colouring of groups of selected nodes. Noik pointed this out, stating that very few classifications at the time used texture, colour or shape to mark out information. Since Noik's classification, the bulk of new graph visualisations have concentrated on focus+context techniques and clustering methods, so this situation has not changed.

The third dimension, *priorities*, describes an ordered scale of the methods used to calculate a focus+context view. Specifically, it describes whether the effect is either set at initialisation (*s*), updated via the application (*A/D*), or described programmatically by the user (*a/d*). It divides the latter two choices into Furnas's API and distance functions [65], where objects are given 'a priori importance's' (APIs) and these are then used as inputs to the distance functions which calculate Degree of Interest values (DOIs) via proximity to selected objects. The graph visualisation has no focus+context mechanism, and so doesn't register in this dimension, and the set-based visualisation, strictly speaking, does not use API values, all nodes are rated as being of equal importance to begin with. It does however build up historical APIs through the course of an interaction, and therefore the set-based visualisation registers in two columns in this dimension.

Fourthly, visualisations and techniques are ranked according to the number of *focal points* they support, either none, one, or multiple points. As the graph visualisation has no focusing technique, it obviously supports no focal points, but the set-based visualisation can support multiple independent and linked focal areas. Conversely, in the animation dimension, which is represented as a simple Boolean yes/no selection, the graph visualisation fulfils the criteria. Visualisation of the updates in the underlying spring-mass model intrinsically animates changes in layout due to removal or addition of hierarchies. The set-based visualisation has no animating properties.

Finally, each visualisation technique is ranked along a scale related to the complexity of the graph structures it can display, namely sequences (*s*), hierarchies (*h*) (trees and forests), full graphs (*g*), nested graphs (*n*), and 'beyond nested graphs' (*). The structure described at each rank on the scale is a superset of the previous structure. Both the visualisations handle Directed Acyclic MultiGraphs (DAMGs), and their ranking on this scale is uncertain. They certainly fulfil the hierarchy qualification, but are not general graphs, existing at a point between the two structures, as do normal DAGs. However, the multigraph nature of the structures is not something that is normally classed as a feature of a general graph.

Noik's technique allows a more detailed comparison with other common tree and graph visualisation techniques such as Cone Trees and TreeMaps. For instance, in Table 8.1 it can be observed that the graph and set-based visualisations are both derived from the underlying graph structure, as are TreeMaps and Cone Trees. Unsurprisingly, the ZTree and Continuous Zoom approaches appear to be very similar, which is to be expected as the ZTree is based on the Continuous Zoom technique, the main difference being that the ZTree must also calculate the initial layout for the zoom technique to work on.

Noik proposed that a visualisation that could fulfil all the possible conditions, and the highest ranks of the focal point and input structure categories, would provide the most powerful graph visualisation possible. It would certainly be the most wide-ranging in scope, but whether it would be the 'best' is open to question. Visualisation techniques tend to be more usable and powerful when focused on a tightly defined domain. For instance, the set-based visualisation can manage trees, multiple trees, and multiple DAGs, and can also work as a multi-dimensional visualisation. However, in the case of the multi-dimensional visualisation, there are definitely more powerful techniques aimed solely at visualising such data sets. Visualisation *frameworks* become more powerful if they are able to generalise, but a jack-of-all-trades visualisation *tool* risks excelling at none of its intended tasks.

All these classifications demonstrate to a greater or lesser extent where both visualisations fit within the standard range of tree, graph, and N-D structures etc. Noik's classification gives a more detailed map of the graph visualisation realm, but to demonstrate more clearly where the visualisations make a valuable contribution, yet another classification must be developed.

This classification is based on the fact that the visualisations are not novel in handling trees, but in handling, displaying and linking between multiple tree representations simultaneously, specifically showing the overlap in structure and content between these trees. The classification in Table 8.2 focuses on how visualisations that display multiple, linked representations of larger structures have developed.

		UNIFIED DATA TYPE				
		N-d	**Tree**	**DAG**	**DAMG**	**Graph**
Multiple Linked Representations	**1-D**	Parallel Co-ordinates [91]				
	2-D	Linked Scatterplots [15]		Klingner's MDS scaling of evolutionary trees [103]		
	Tree	Single spanning tree with multiple scatterplots [104]	Cone Trees, Tree Maps etc (i.e. single tree model = single tree visualisation)	MultiTree based multiple TreeMap [174] Web Ecologies [40]	My set and graph-based visualisations	
	DAG				My graph-based visualisation	
	Graph					My graph-based visualisation

Table 8.2. Space of linking multiple smaller structures from a larger structure.

It shows that linking techniques have been concentrated in reducing multi-dimensional data to smaller 1-D or 2-D linked representations. Koschat and Swayne [104] link one spanning tree with scatterplots that represent multi-dimensional data whilst Klingner [103] displays sets of trees as scatterplots using multi-dimensional scaling methods. Wittenburg *et al's* [174] multiple Treemap representations of the MultiTree DAG structure begins to explore the space of using less complicated linked structures to represent a more complicated overall artefact, but its unsuitability for the multiple taxonomy requirements was documented in Chapter 4.

The visualisations described in this thesis explore this space further, up to the final development of the graph prototype that linked together multiple graphs. It is noticeable that systems stick close to the diagonal that marks a visual structure mirroring the actual underlying structure. This shows that complicated structures tend to be reduced to only slightly simpler, linked structures, rather than to radically different structures. For instance, a full multigraph would be tricky to reduce to a set of scatterplots. The other side of the diagonal marks improbabilities such as a DAG being represented by multiple, linked full graph visualisations, and while multi-dimensional data can be turned into overall graph form it will tend not be turned into multiple, linked graphs.

8.2 Contribution

Previous attempts to visualise multiple overlapping hierarchies were either limited in scale, scope or interaction possibilities. The set-based visualisation presented here allows users for the first time to see detailed correlations between separate trees. In general terms, the set-based visualisation does for trees what Becker and Cleveland's work [15] did for scatterplots, and in the course of this work a number of advances were made over existing IV techniques.

The original contributions of this thesis occurred as existing display and interaction techniques of IV were required to be applied and extended to a relatively unexplored type of information structure, multiple overlapping hierarchies, and then tested on a truly representative user population.

8.2.1 Novel Hybrid Tree Layout

Firstly, a new variation of IV tree layout was developed and implemented for the set-based visualisation. Individually, as a representation, each tree combines the logical to spatial depth mapping that the Cone Tree and traditional node-link layouts use along with the space-efficiency of Treemap style layouts. These properties enable multiple tree representations to be fitted into the screen space, as with the multiple Treemaps of Wittenburg *et al* [174], but also allows their structures and depths to be easily compared at a glance, something that cannot be done with Treemap representations. Lately, other techniques such as the radial space-filling techniques, exemplified by the web ecologies of Chi *et al* in Figure 4.2, have also combined these two properties. However, there are two potential drawbacks of these representations with regard to visualising multiple hierarchies. Primarily, the leaf nodes, the most numerous groups of nodes, are still arranged in a 1D linear fashion in the radial techniques, leading to thin stalk-like representations. The tree drawing technique presented here displays a group of leaf nodes as square representations in a 2D grid, similar to Treemaps. Also, with regard to the space efficiency of displaying multiple trees, it is harder to tessellate the circular structures of the radial techniques within a rectangular screen space, such that the maximum amount of space is used for data presentation.

8.2.2 Extension of Interaction Techniques

Secondly, particular interaction techniques were extended upon and improved to operate on more complicated structures than had previously been attempted in IV. Linking and brushing techniques were extended from between simple scatterplots, as with Becker and Cleveland, or from between scatterplots and a single tree representation as in Koschat and Swayne [104], to being able to link and brush between multiple tree representations simultaneously. Also, the data set this linking and brushing was acting on was a restricted graph (the DAMG), compared to the multi-dimensional data sets that linking techniques had been confined to in these other examples.

Linking techniques were also extended past the simpler acts of propagation of highlighting or filtering attributes, to encompass focus and context effects in the form of linked Degree of Interest (DOI) values. This combination caused an object’s representations in all views to achieve a level of focus when just one of its representations was selected, allowing linked objects affected by a selection to increase in size and achieve greater prominence at the expense of unselected objects. To current knowledge, this is a novel combination of linking and focus+context techniques.

8.2.3 Meaningful Interaction with Multiple Hierarchies

The basic interactive techniques were only practical because of the combination of space-efficient display and the novel focus+context policy. Even with quite small data sets, other multiple tree visualisations have struggled to provide basic interaction merely because of the size of the on-screen representations, being for the most part, output-only channels of communication to a user or viewer as Fitts’ Law asserted itself. The properties of the set-based visualisation allow, for the first time, questions to be asked of the information by the user, rather than just acting as an almost static representation of the information to hand. Therefore, one of the contributions of this work is that a meaningful interaction dialogue can be conducted with a complete visual display of such an information set for the first time. In effect, this thesis accomplishes a merging of the intricate selecting and brushing interactivity of the scatterplot visualisations with the more structurally complex representations and data models of the various multiple tree visualisations.

With the original taxonomic data sets, the visualisations represent novel techniques for bioinformatics, in which multiple taxonomies can be compared and investigated. Other work on visualising multiple taxonomic classifications such as that by Amavizca *et al* [4] and Klingner [103] had either sacrificed detail or scale to display such structures simultaneously. To expand beyond this one area, the visualisations were also tested with further data sets on top of the original botanical taxonomy data. The multi-dimensional information in particular gave sound indications of the set-based technique’s versatility, enabling as it did a quasi-Parallel Co-ordinates type behaviour. This allowed the visualisation to stake a claim to being able to visualise internal relationships in sets of objects that can be organised according to multiple properties, either by attributes or by classification schemas.

The graph-based visualisation, though reported as being less usable than the set-based representations, could also be extended in an original manner to allow multiple overlapping *general graphs* to be displayed together. Whilst other spring-mass visualisation systems such as those by Donath [51] and Huang *et al* [89] can animate addition and deletion of nodes, the graph visualisation presented here can animate the effect of the removal and addition of entire graph structures.

The work also provides a useful case study into the development of an original visualisation from its beginnings, and offers several findings as to the effectiveness of some standard testing methods with regard to IV development. As with any user-centred development, iterative design and testing is necessary, as user actions cannot be predicted or specified accurately in advance. It was also found that the initial tasks changed during the course of the thesis, as the visualisations and perceived tasks engaged in a form of feedback loop. What the taxonomists observed at one point effected their thoughts as to what tasks the visualisation could possibly extend to with further stages of development. This type of behaviour is not uncommon in information visualisations, as early prototypes or systems enable users to see data, patterns and then possibilities they would have hitherto neither imagined nor entertained.

The preference of the user population for the set-based over the graph-based visualisation further supports the theory that people prefer to handle trees rather than complex networks. Mukherjea *et al* [125] demonstrated this to be the case when they deconstructed a graph structure down to one tree representation that represented one particular viewpoint. However this thesis also shows those users can, and prefer to, handle *multiple* tree representations, and hence *multiple* viewpoints on the overall structure, as compared to the one overall graph structure. Of course it helps if these viewpoints are viable and sensible, and the overall structure of the information sets discussed here are formed from amalgamations of well-defined individual structures, rather than the viewpoints being extracted from a pre-formed superstructure.

In summary, this thesis presents an advance in the visualisation of multiple overlapping hierarchies, and also makes contributions towards multiple focus + context mechanisms, two aspects of IV that Robertson [149] has identified to be in need of particularly urgent development.

This work is not a complete solution to visualising multiple hierarchies; no hierarchy visualisation to date has been that, which is of course why development and research has continued and will continue. It is a visualisation that has evolved and adapted to suit a particular type of data, namely taxonomic data, and it has also shown an ability to handle other types of related information sets, and as such makes a definite and concrete contribution to the IV research field.

8.3 Future Work

As the previous paragraph states that this thesis is not *the* solution to the problem of visualising multiple overlapping hierarchies, it should be stated in which possible directions further work to increase and refine the usefulness and generality of such a visualisation should proceed.

Given the original difficulties in obtaining data sets and finding suitable data formats, one obvious progression of this work is to integrate it with a database that can deliver multiple overlapping hierarchies for comparison. Indeed this is now the subject of an ongoing research effort at Napier. The fundamental problem of reconciling database retrieval times with the speed necessary for interactive operations such as brushing is being explored by other research groups also, such as documented in Stroe *et al* [163].

With regard to the set-based visualisation itself, it would be useful to incorporate some other features that are presently found in single hierarchy visualisations. Drill-down techniques for example, if adapted properly, could allow extremely large data sets of more than ten or so levels to be explored. Currently the visualisation has the ability to hide certain levels within the hierarchies, but this is a breadth-wise operation that affects all nodes at that level. This operation doesn't allow particular sub-trees to be displayed or concealed and makes little sense on some data sets such as file structures, and so it may be desirable to use drill-down techniques to remove or target depth-wise portions of the hierarchies under consideration. 'Zooming' in like this on portions of ultra-large hierarchies, such as digital libraries as detailed in Shneiderman *et al* [156], allows users to focus on areas deep within a hierarchy and deal with much larger data sets. The problem here for multiple hierarchies is how to provide display and exploration facilities for the resulting re-distribution of one portion of one hierarchy when those elements could be scattered widely across the whole structures of other hierarchies. It is unclear how a dispersed collection of elements could be drilled down to, or shown in context, in an extremely large hierarchy.

The linked multiple focus + context mechanism worked well generally, but during the final test one taxonomist observed that its bias towards favouring sets with a high ratio of selected objects may not always be the desired effect. They pointed out those occasionally outlying selections or peculiar distributions would be of most interest. Inverting the selection using the present controls would bring these groups to the fore, but give even more prominence to groups with no selected members. Hence, a focus + context mechanism that enhanced both very strong and very weak (but not non-existent) correlations would be welcome, at the expense of middling distributions. In effect, the idea comes across as a type of focus + context "graphic equaliser", requiring the ability to interactively set focal effects on groups according to the frequency or ratio of selected objects within them. Would such a control mechanism be feasible and useful or too involved to be workable?

During user testing, it was found that software logging was more useful to subsequent analysis of user behaviour than video recording for the space-efficient and tightly packed set-based visualisation. Direct screen capture software in combination with a tape record of user comments was put forward as a possible replacement for video recording, but lately Java has introduced a `Robot` class that allows input devices to be mimicked under programmatic control. If a suitable Robot class was constructed, and a richer software log that captured input events such as mouse movements and key presses was developed, these should theoretically be able to combine to re-enact a user's actions on the prototype. This would then perform the same function as the direct screen capture software but at a much lower cost in resources.

Also, the set-based visualisation performed well when handing multiple trees or DAGs. However, visualising multiple overlapping graphs proved more problematic, as the graph prototype that could achieve this was stated to be harder to use and understand compared to the set-based visualisation's hierarchical layouts, drawbacks it has in common with other graph visualisations. Could there be a way of either extending the set-based visualisation to deal with graphs, perhaps by breaking them down into spanning trees, or by improving the legibility and interactivity of the graph visualisation?

Finally, and probably a request of most IV researchers, there is a need for a more robust and complete framework to describe the IV space. General frameworks and taxonomies that currently exist can describe static, single instances of representations such as a scatterplot or Cone Tree, but cannot deal with dynamic visualisations easily. Similarly, Chuah and Roth's [41] taxonomy of visualisation interactions can describe many common and complex interactions, but it is not easy to connect these interactions back to the display characteristics of a visualisation.

Smaller, more detailed, taxonomies exist for specific areas, such as Noik's [134] for graph structures, which can capture subsets of interaction and display qualities of visualisations, but it was still difficult to fully describe the properties of the visualisations developed in this thesis within their classification. Thus, a more extensible and comprehensive description of the IV space is needed, perhaps developed from scratch or, as seems more likely, from unifying and generalising the existing examples.

9 References

[1] Abello, J. and Korn, J. (2000). Visualizing Massive Multi-Digraphs. Proc. of IEEE Symposium on InfoVis 2000 (October 9-10, Salt Lake City, Utah, USA), IEEE Computer Society Press, 39-48.

[2] Abello, J. and Krishnan, S. (1999). Graph Surfaces. Proc. of International Congress on Industrial and Applied Math (July 5-9, Edinburgh, UK), 234-244.

[3] Ahlberg, C. and Shneiderman, B. (1994). Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. Proc. of ACM Conference on Human Factors in Software, CHI '94 (April 24-28, Boston, MA, USA), ACM Press, 313-317.

[4] Amavizca, M., Sanchez, J. A. and Abascal, R. (1999). Visualization of large and complex information spaces in the Floristic Digital Library. Proc. of ENC '99 (September 12-15, Pachuca, Hidalgo, Mexico).

[5] Andrews, K. and Heidegger, H. (1998). Information Slices: Visualising and Exploring Large Hierarchies using Cascading, Semi-Circular Discs. Proc. of IEEE InfoVis '98, Late Breaking Hot Topics (October 19-20, Research Triangle, North Carolina, USA), IEEE Computer Society Press, 9-12.

[6] Andrews, K., Pichler, M. and Wolte, J. (1997). Information Pyramids: A New Approach to Visualising Large Hierarchies. Proc. of Late Breaking Hot Topics, IEEE Visualization '97 (October 19-24, Phoenix, Arizona, USA), IEEE Computer Society Press, 49-52.

[7] Asahi, T., Turo, D. and Shneiderman, B. (1995). Using treemaps to visualize the Analytic Hierarchy Process. Information Systems Research, 6 (4), 357-375.

[8] Baldonado, M. Q. W., Woodruff, A. and Kuchinsky, A. (2000). Guidelines for Using Multiple Views in Information Visualizations. Proc. of AVI 2000 (May 24-26, Palermo, Italy), ACM Press, 110-119.

[9] Bartram, L. (1997). Perceptual and interpretative properties of motion for information visualization. Proc. of NPIV '97. New Paradigms in Information Visualization and Manipulation (November 13-14, Las Vegas, Nevada, USA), ACM Press, 3-7.

[10] Bartram, L. (1998). Enhancing Visualizations with Motion. Proc. of IEEE InfoVis '98: Late Breaking Hot Topics (October 19-20, Research Triangle, North Carolina, USA), IEEE Computer Society Press, 13-16.

[11] Bartram, L., Ho, A., Dill, J. and Henigman, F. (1995). The Continuous Zoom: A Constrained Fisheye Technique for Viewing and Navigating Large Information Spaces. Proc. of ACM UIST '95 (November 14-17, Pittsburgh, USA), ACM Press, 207-215.

[12] Bartram, L., Uhl, A. and Calvert, T. (2000). Navigating Complex Information with the ZTree. Proc. of Graphics Interface 2000 (May 15-17, Montreal, Quebec, Canada), Morgan Kaufmann, 11-18.

[13] Bartram, L., Ware, C. and Calvert, T. (2001). Moving Icons: Detection and Distraction. Proc. of INTERACT 2001 (July 9-13, Tokyo, Japan), North-Holland.

[14] Beaudoin, L., Parent, M.-A. and Vroomen, L. C. (1996). Cheops: A Compact Explorer for Complex Hierarchies. Proc. of IEEE Visualization '96 (October 27 - November 1, San Francisco, USA), Computer Society Press, 87-92.

[15] Becker, R. A. and Cleveland, W. S. (1987). Brushing Scatterplots. Technometrics, 29 (2), 127-142.

[16] Bederson, B. B. (2001). PhotoMesa: A Zoomable Image Browser using Quantum Treemaps and Bubblemaps. Proc. of UIST 2001 (November 11-14, Orlando, Florida, USA), ACM Press, 71-80.

[17] Bederson, B. B. and Boltman, A. (1999). Does Animation Help Users Build Mental Maps of Spatial Information? Proc. of InfoVis '99 (October 25-26, San Francisco, California, USA), IEEE Computer Society Press, 28-35.

[18] Bederson, B. B. and Hollan, J. D. (1994). Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. Proc. of UIST '94 (November 2-4, Marina Del Ray, California, USA), ACM Press, 17-26.

[19] Benedikt, M. (1991). Cyberspace: some proposals. In M. Benedikt (eds). Cyberspace: First Steps, 119-224, MIT Press.

[20] Bentley, C. L. and Ward, M. O. (1996). Animating Multidimensional Scaling to Visualize N-Dimensional Data Sets. Proc. of IEEE Information Visualization 1996 (October 28-29, San Francisco, California, USA), IEEE Computer Society Press, 72-73.

[21] Björk, S., Holmquist, L. E. and Redström, J. (1999). A Framework For Focus+Context Visualization. Proc. of IEEE InfoVis '99 (October 25-26, San Francisco, California, USA), IEEE Computer Society Press, 53-56.

[22] Brath, R. (1997). Concept demonstration metrics for effective information visualization. Proc. of IEEE Symposium on Information Visualization 1997 (InfoVis '97) (October 20-21, Phoenix, Arizona, USA), IEEE Computer Society Press, 108-111.

[23] Brazell, M. C. and Jeffrey, C. L. (1997). Tree Structure Detection and Visualization, Technical Report CS-97-7, Division of Computer Science, University of Texas, August 6, 1997, 1997.

[24] Brodbeck, D., Chalmers, M., Lunzer, A. and Cotture, P. (1997). Domesticating Bead: Adapting an Information Visualization System to a Finanical Institiution. Proc. of IEEE InfoVis '97 (October 20-21, Phoenix, Arizona, USA), IEEE Computer Society Press, 73-80.

[25] Brooke, J. (1996). SUS: A Quick and Dirty Usability Scale. In P. W. Jordan, B. Thomas, B. A. Weerdmeester and I. L. McClelland (eds). Usability Evaluation in Industry, 189-194. London, UK, Taylor & Francis.

[26] Bruls, M., Huizing, K. and Wijk, J. J. v. (2000). Squarified Treemaps. Proc. of VisSym '00 (May 29-31, Amsterdam, The Netherlands), SpringerWienNewYork, 33-42.

[27] Buja, A., McDonald, J. A., Michalak, J. and Stuetzle, W. (1991). Interactive Data Visualization Using Focusing and Linking. Proc. of IEEE Visualization '91 (October 22-25, San Diego, California, USA), IEEE Computer Society Press, 156-163.

[28] Camtasia (2000). http://www.camtasia.com, TechSmith Corporation.

[29] Card, S. K. and Mackinlay, J. D. (1997). The structure of the information visualization design space. Proc. of IEEE Symposium on Information Visualization 1997 (October 20-21, Phoenix, Arizona, USA), Computer Society Press, 92-99.

[30] Card, S. K., Mackinlay, J. D. and Shneiderman, B., Eds (1999). Readings in Information Visualization: Using Vision to Think. The Morgan Kaufmann Series in Interactive Technologies. Morgan Kaufmann, San Francisco, 1999.

[31] Card, S. K., Robertson, G. G. and Mackinlay, J. D. (1991). The Information Visualizer, an Information Workspace. Proc. of ACM CHI '91 (April 27 - May 2, New Orleans, Louisiana, USA), ACM Press, 181-188.

[32] Carpendale, M. S. T., Cowperthwaite, D. J. and Fracchia, F. D. (1997). Extending Distortion Viewing from 2D to 3D. IEEE Computer Graphics & Applications, 17 (4), 42-51.

[33] Carrière, J. and Kazman, R. (1995). Interacting with Huge Hierarchies: Beyond Cone Trees. Proc. of IEEE InfoVis '95 (October 30-31, Atlanta, USA), Computer Society Press, 74-81.

[34] Carroll, J. (1995). Scenario-based Design. John Wiley.

[35] Chalmers, M. (1994). Designing Information Workspaces. Proc. of ECHT '94 (European Conference on Hypermedia Technology) Workshop on Spatial Metaphors (September 18, Edinburgh, Scotland).

[36] Chalmers, M. (1996). A Linear Iteration Time Layout Algorithm for Visualizing High-Dimensional Data. Proc. of IEEE Information Visualization '96 (October 28-29, San Francisco, California, USA), IEEE Computer Society Press, 127-132.

[37] Chalmers, M. and Chitson, P. (1992). Bead: Explorations in Information Visualization. Proc. of ACM SIGIR '92 (June 21-24, Copenhagen, Denmark), ACM Press, 330-337.

[38] Chen, C. (1997). Structuring and visualising the WWW by generalised similarity analysis. Proc. of Eighth ACM Conference on Hypertext (Hypertext '97) (April 6-11, Southampton, UK), ACM Press, 177-186.

[39] Chi, E. H. (2000). A Taxonomy of Visualization Techniques using the Data State Reference Model. Proc. of IEEE InfoVis 2000 (October 9-10, Salt Lake City, Utah, USA), IEEE Computer Society Press, 69-75.

[40] Chi, E. H., Pitkow, J., Mackinlay, J., Pirolli, P., Gossweiler, R. and Card, S. K. (1998). Visualizing the Evolution of Web Ecologies. Proc. of ACM CHI '98 (April 18-23, Los Angeles, California, USA), ACM Press, 400-407.

[41] Chuah, M. C. and Roth, S. F. (1996). On the Semantics of Interactive Visualizations. Proc. of IEEE InfoVis '96 (October 28-29, San Francisco, California, USA), IEEE Computer Society Press, 29-36.

[42] Cockburn, A. and McKenzie, B. (2000). An Evaluation of Cone Trees. Proc. of BCS HCI 2000: People and Computers XIV (September 5-8, Sunderland, UK), Springer-Verlag, 425-436.

[43] Colby, G. and Scholl, L. (1991). Transparency and blur as selective cues for complex visual information. Proc. of SPIE Image Handling and Reproduction Systems Integration (February 26, San Jose, California, USA), SPIE, 114-125.

[44] Colebourne, A., Mariani, J. A. and Rodden, T. (1996). Q-PIT: a populated information terrain. Proc. of Visual Data Exploration and Analysis III (January 31 - February 2, San Jose, California, USA), SPIE Press, 12-22.

[45] Coutaz, J. (1987). PAC, an object oriented model for dialog design. Proc. of INTERACT '87 (September 1-4, Stuttgart, West Germany), North-Holland, 431-436.

[46] Csinger, A. (1992). The Psychology of Visualization, Technical Report TR-92-28, University of British Columbia, 1992.

[47] Dantzich, M. v. (1997). Visualization is a state of mind. Proc. of NPIV '97 (November 13-14, 1997, Las Vegas, USA), ACM Press, 29-31.

[48] Davidson, R. and Harel, D. (1996). Drawing graphs nicely using simulated annealing. ACM Transactions on Graphics, 15 (4), 301-331.

[49] Dill, J. (1998). Special Introduction. IEEE Computer Graphics and Applications, 18 (4 - July/Aug), 16.

[50] Dominguez, E. and Wheeler, Q. D. (1997). Taxonomic Stability is Ignorance. Cladistics, 13 (4), 367-372.

[51] Donath, J. S. (1995). Visual Who: Animating the affinities and activities of an electronic community. Proc. of ACM MultiMedia '95 (November 5-9, San Francisco, USA), ACM Press, 99-108.

[52] Drevermann, H. and Travis, D. (1998). Visualization using Colour: visual presentation of events in particle physics. Behaviour & Information Technology, 17 (1), 18-26.

[53] Eades, P. and Huang, M. L. (2000). Navigating Clustered Graphs using Force-Directed Methods. Journal of Graph Algorithms and Applications, 4 (3), 157-181.

[54] Eick, S. G. (1994). Data Visualization Sliders. Proc. of UIST '94 (November 2-4, Marina del Ray, California, USA), ACM Press, 119-120.

[55] Eick, S. G., Steffen, J. L. and Sumner, E. E., Jr. (1992). Seesoft - A Tool For Visualizing Line Oriented Software Statistics. IEEE Transactions on Software Engineering, 18 (11), 957-968.

[56] Ellis, J. B., Rose, A. and Plaisant, C. (1997). Putting visualization to work: ProgramFinder for youth placement. Proc. of CHI '97: Conference proceedings on Human Factors in Computing Systems (March 22-27, Atlanta, Georgia, USA), ACM Press, 502-509.

[57] Espinosa, O. J., Hendrickson, C. and Garrett, J. H. (1999). Domain Analysis: A Technique to Design A User-Centred Visualization Framework. Proc. of IEEE InfoVis '99 (October 25-26, San Francisco, California, USA), IEEE Computer Society Press, 44-52.

[58] Feiner, S. and Beshers, C. (1990). Worlds within Worlds: Metaphors for Exploring n-Dimensional Virtual Worlds. Proc. of ACM UIST '90 (October 3-5, Snowbird, Utah, USA), ACM Press, 76-83.

[59] Felsenstein, J. (1999). The Newick tree format. HTML Document at http://evolution.genetics.washington.edu/phylip/newicktree.html.

[60] Fishkin, K. and Stone, M. C. (1995). Enhanced Dynamic Queries via Movable Filters. Proc. of ACM CHI '95 (May 7-11, Denver, Colorado, USA), ACM Press, 415-420.

[61] Fowler, A. (2000). A Swing Architecture Overview, Web-based Technical Report http://java.sun.com/products/jfc/tsc/articles/architecture/index.html, Sun Microsystems Co., 21st June, 2000, 2000.

[62] Fruchterman, T. M. J. and Reingold, E. M. (1991). Graph Drawing by Force-directed Placement. Software - Practice and Experience, 21 (11), 1129-1164.

[63] Fua, Y.-H., Ward, M. O. and Rundensteiner, E. A. (1999). Hierarchical Parallel Coordinates for Exploration of Large Datasets. Proc. of IEEE Visualization '99 (October 24-29, San Francisco, California, USA), IEEE Computer Society Press, 43-50.

[64] Fua, Y.-H., Ward, M. O. and Rundensteiner, E. A. (1999). Navigating Hierarchies with Structure-Based Brushes. Proc. of InfoVis '99 (October 25-26, San Francisco, California, USA), IEEE Computer Society Press, 58-64.

[65] Furnas, G. W. (1986). Generalized Fisheye Views. Proc. of ACM CHI '86 (April 13-17, Boston, Massachussetts, USA), ACM Press, 16-23.

[66] Furnas, G. W. and Zacks, J. (1994). Multitrees: Enriching and Reusing Hierarchical Structure. Proc. of ACM CHI '94 (April 24-28, Boston, Massachusetts, USA), ACM Press, 330-336.

[67] Gall, H., Jazayeri, M. and Riva, C. (1999). Visualizing Software Release Histories: The Use of Color and Third Dimension. Proc. of International Conference on Software Maintenance (ICSM '99) (August 30 - September 3, Oxford, England), IEEE Computer Society Press, 99-108.

[68] Gansner, E. R. and North, S. C. (1998). Improved Force-directed Layouts. Proc. of Graph Drawing '98 (August 13-15, Montreal, Canada), Springer-Verlag, 364-373.

[69] Gershon, N., Card, S. K. and Eick, S. G. (1998). Information Visualization Tutorial. Proc. of ACM CHI '98: human factors in computing systems (April 18-23, Los Angeles, USA), ACM Press, 109-110.

[70] Gershon, N. and Eick, S. G. (1995). Visualizations new tack: Making sense of information. IEEE Spectrum, 32 (11), 38-56.

[71] Gershon, N. and Eick, S. G. (1997). Information Visualization. IEEE Computer Graphics & Applications, 17 (4), 29-31.

[72] Gershon, N., Eick, S. G. and Card, S. K. (1998). Design: Information Visualization. Interactions, 5 (2), 9-15.

[73] Ghoniem, M. and Fekete, J.-D. (2001). Animating Treemaps. Proc. of 18th HCIL Symposium - Workshop on Treemap Implementations and Applications (May 31, University of Maryland, College Park, Maryland, USA).

[74] Gilmour, R. (2000). Taxonomic markup language: applying XML to systematic data. Bioinformatics, 16 (4), 406-407.

[75] Gould, J. D. and Lewis, C. (1985). Designing for usability: key principles and what designers think. Communications of the ACM, 28 (3), 300-311.

[76] Graves, A., Hand, C. and Hugill, A. (1997). Interactive Visualisation of Musical Form using VRML. Proc. of Fourth UK VR-SIG Conference (November 1, Brunel University, England), 98-109.

[77] GVU WWW User Surveys (1998). http://www.gvu.gatech.edu/user_surveys/, GVU Center, Georgia Tech, October 1998, 1998.

[78] Hao, M. C., Hsu, M., Dayal, U. and Krug, A. (2000). Navigating Large Hierarchical Space Using Invisible Links. Proc. of Visual Data Exploration and Analysis VII (January 24-26, San Jose, California, USA), SPIE Press, 73-83.

[79] Harel, D. (1988). On Visual Formalisms. Communications of the ACM, 31 (5), 514-530.

[80] Harel, D. and Koren, Y. (2000). A Fast Multi-Scale Method for Drawing Large Graphs. Proc. of Graph Drawing 2000 (September 20-23, Williamsburg, Virginia, USA), Springer Verlag, 183-196.

[81] Hascoët-Zizi, M., Ahlberg, C., Korfhage, R., Plaisant, C., Chalmers, M. and Rao, R. (1996). Where is Information Visualization Technology going? Proc. of User Interface Software and Technology (UIST) '96 (November 6-8, Seattle, Washington, USA), ACM Press, 75-77.

[82] Healey, C. G. (1996). Choosing Effective Colours for Data Visualization. Proc. of IEEE Visualization 1996 (October 27 - November 1, San Francisco, California, USA), IEEE Computer Society Press, 263-270.

[83] Healey, C. G., Booth, K. S. and Enns, J. T. (1995). Visualizing real-time multivariate data using preattentive processing. ACM Transactions on Modelling and Computer Simulation, 5 (3), 190-221.

[84] Healey, C. G., Booth, K. S. and Enns, J. T. (1996). High-speed Visual Estimation using Pre-attentive processing. ACM Transactions on Human-Computer Interaction, 3 (2), 107-135.

[85] Hendley, R. J., Drew, N. S., Wood, A. M. and Beale, R. (1995). Narcissus: Visualising Information. Proc. of IEEE Information Visualisation Symposium 95 (October 30-31, Atlanta, Georgia, USA), Computer Society Press, 90-97.

[86] Herman, I. and Marshall, M. S. (2000). GraphXML - a graph description language. Proc. of Graph Drawing '00 (September 20-23, Williamsburg, Virginia, USA), Springer Verlag, 52-62.

[87] Herman, I., Melançon, G. and Marshall, M. S. (2000). Graph Visualisation and Navigation in Information Visualisation: a Survey. IEEE Transactions on Visualization and Computer Graphics, 6 (1), 24-43.

[88] Herman, I., Melançon, G., de Ruiter, M. M. and Delest, M. (1999). Latour - a tree visualisation system. Proc. of Graph Drawing '99 (September 15-19, Stirin Castle, Prague, Czech Republic), Springer-Verlag, 392-399.

[89] Huang, M. L., Eades, P. and Wang, J. (1998). On-line Animated Visualization of Huge Graphs using a Modified Spring Algorithm. Journal of Visual Languages and Computing, 9 (6), 623-645.

[90] Hubona, G. S., Shirah, G. W. and Fout, D. G. (1997). The effects of motion and stereopsis on three-dimensional visualization. International Journal of Human-Computer Studies, 47 (5), 609-627.

[91] Inselberg, A. and Dimsdale, B. (1990). Parallel Coordinates: A Tool for Visualizing Multidimensional Geometry. Proc. of IEEE Visualization 1990 (October 23-25, San Francisco, California, USA), IEEE Computer Society Press, 361-378.

[92] Inxight Software Inc. (1999). Star Tree Demos. Java applets at http://www.inxight.com/products/star_tree/demos.html, Inxight Software Inc.

[93] Jeffrey, C. (1982). An introduction to plant taxonomy. Cambridge University Press, Cambridge, UK.

[94] Jeong, C.-S. and Pang, A. (1998). Reconfigurable Disc Trees for Visualizing Large Hierarchical Information Space. Proc. of IEEE InfoVis '98 (October 19-20, Research Triangle, North Carolina, USA), Computer Society Press, 19-25.

[95] Jern, M. (1996). Information Visualization enables interaction with large datasets on the web. Proc. of 3D and Multimedia on the Internet, WWW and Networks. (Proceedings published in book form as "The Internet in 3D: Information, images and interaction") (April 16-18, Bradford, UK), Academic Press, 223-236.

[96] Johnson, B. and Shneiderman, B. (1991). Treemaps: A Space-Filling approach to the visualization of hierarchical information structures. Proc. of IEEE Visualization '91 (Oct 22-25, San Diego, California, USA), IEEE Computer Society Press, 284-291.

[97] Kamada, T. and Kawai, S. (1989). An algorithm for drawing general undirected graphs. Information Processing Letters, 31 (1), 7-15.

[98] Kaugars, K. (1999). OTree: A Tree Visualization using Scaling and Omission. Proc. of Interact '99 (August 30 - September 3, Edinburgh, UK), 491-496.

[99] Kazman, R. and Carrière, J. (1996). An Adaptable Software Architecture for Rapidly Creating Information Visualizations. Proc. of Graphics Interface '96 (May 22-24, Toronto, Ontario, Canada), Morgan Kaufmann, 17-27.

[100] Keahey, T. A. and Robertson, E. L. (1997). Nonlinear Magnification Fields. Proc. of IEEE InfoVis '97 (October 20-21, Phoenix, Arizona, USA), IEEE Computer Society Press, 38-45.

[101] Keim, D. A. and Herrmann, A. (1998). The Gridfit Algorithm: An Efficient and Effective Approach to Visualizing Large Amounts of Spatial Data. Proc. of IEEE Visualization 1998 (October 18-23, Research Triangle, North Carolina, USA), IEEE Computer Society Press, 181-188.

[102] Keim, D. A., Kriegel, H.-P. and Anderst, M. (1995). Recursive Pattern: a technique for visualizing very large amounts of data. Proc. of IEEE Visualization 1995 (October 29 - November 3, Atlanta, Georgia, USA), IEEE Computer Society Press, 279-286.

[103] Klingner, J. (2001). Visualizing Sets of Evolutionary Trees, Technical Report TR01-26, Dept. of Computer Science, University of Texas, May 2001, 2001.

[104] Koschat, M. and Swayne, D. F. (1996). Interactive Graphical Methods in the Analysis of Customer Panel Data. Journal of Business and Economics Statistics, 14 (1), 113-132.

[105] Krasner, G. E. and Pope, S. T. (1988). A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System. Journal of Object-Oriented Programming, 1 (3), 26-49.

[106] Kratzner, K. (1987). The Desktop Metaphor - Visualisation of Office Applications on a Desktop. Proc. of IEEE Computer Society Office Automation Symposium (April 27-29, Gaithersburg, Maryland, USA), IEEE Computer Society Press, 128-136.

[107] Kreuseler, M., Lopez, N. and Schumann, H. (2000). A Scalable Framework for Information Visualization. Proc. of InfoVis 2000 (October 9-10, Salt Lake City, Utah, USA), IEEE Computer Society Press, 27-36.

[108] Kumar, A. and Fowler, R. H. (1994). A Spring Modeling Algorithm to Position Nodes of an Undirected Graph in Three Dimensions, Technical CS-94-7, University of Texas - Pan American, May 1994, 1994.

[109] Kumar, H. P., Plaisant, C. and Shneiderman, B. (1996). Browsing hierarchical data with multi-level dynamic queries and pruning. International Journal of Human-Computer Studies, 46 (1), 103-124.

[110] Lamping, J., Rao, R. and Pirolli, P. (1995). A focus + context technique based on hyperbolic geometry for visualizing large hierarchies. Proc. of ACM CHI '95 (May 7-11, Denver, Colorado, USA), ACM Press, 401-408.

[111] Lanning, T., Wittenburg, K., Heinrichs, M., Fyock, C. and Li, G. (2000). Multidimensional Information Visualization through Sliding Rods. Proc. of AVI 2000 (May 23-26, Palermo, Italy), ACM Press, 173-180.

[112] Leung, Y. K. and Apperley, M. D. (1994). A review and taxonomy of distortion-oriented presentation techniques. ACM Transactions on Human-Computer Interaction, 1 (2), 126-160.

[113] Levkowitz, H. and Herman, G. T. (1992). Color Scales for Image Data. IEEE Computer Graphics & Applications, 12 (1), 72-80.

[114] Lux, M. (1997). Visualization of financial information. Proc. of NPIV '97. New Paradigms in Information Visualization and Manipulation (November 13-14, Las Vegas, Nevada, USA), ACM Press, 58-61.

[115] MacDonald, L. W. (1999). Using Color Effectively in Computer Graphics. IEEE Computer Graphics & Applications, 19 (4 - July/August), 20-35.

[116] Mackinlay, J. D. (2000). Opportunities for Information Visualization. IEEE Computer Graphics and Applications, 20 (1 - Jan/Feb), 22-23.

[117] Mackinlay, J. D., Robertson, G. G. and Card, S. K. (1991). The perspective wall: detail and context smoothly integrated. Proc. of ACM CHI '91 (April 27 - May 2, New Orleans, Louisiana, USA), ACM Press, 173-176.

[118] Maddison, D. R., Swofford, D. L. and Maddison, W. P. (1997). NEXUS: An extensible file format for systematic information. Systematic Biology, 46 (4), 590-621.

[119] Marshall, M. S., Herman, I. and Melançon, G. (2001). An object-oriented design for graph visualization. Software: Practice and Experience, 31 (8), 739-756.

[120] McCrickard, D. S., Catrambone, R. and Stasko, J. T. (2001). Evaluating Animation as a Mechanism for Maintaining Peripheral Awareness. Proc. of INTERACT 2001 (July 9-13, Tokyo, Japan), North-Holland, 148-156.

[121] Melançon, G. and Herman, I. (2000). DAG Drawing from an Information Visualization Perspective. Proc. of VisSym '00 - joint Eurographics and IEEE TCVG Symposium on Visualization (May 29-31, Amsterdam, The Netherlands), Springer-Verlag, 3-12.

[122] Miller, N., Hetzler, B., Nakamura, G. and Whitney, P. (1997). The Need For Metrics In Visual Information Analysis. Proc. of New Paradigms in Information Visualization and Manipulation 1997 (NPIV '97) (November 13-14, Las Vegas, USA), ACM Press, 24-28.

[123] Misue, K., Eades, P., Lai, W. and Sugiyama, K. (1995). Layout Adjustment and the Mental Map. Journal of Visual Languages and Computing, 6 (2), 183-210.

[124] Monk, A. F. and Wright, P. C. (1991). Observations and inventions: new approaches to the study of human-computer interaction. Interacting with Computers, 3 (2), 204-216.

[125] Mukherjea, S., Foley, J. D. and Hudson, S. (1995). Visualizing Complex Hypermedia Networks through Multiple Hierarchical Views. Proc. of ACM CHI '95 (May 7-11, Denver, Colorado, USA), ACM Press, 331-337.

[126] Muñoz, R., Miller-Jacobs, H. H., Spool, J. M. and Verplank, B. (1992). In Search Of The Ideal Prototype. Proc. of ACM CHI '92 (May 3-7, Monterey, California, USA), ACM Press, 577-579.

[127] Munzner, T. (1998). Exploring Large Graphs in 3D Hyperbolic Space. IEEE Computer Graphics & Applications, 18 (4), 18-23.

[128] National Computation Science Alliance (1998). Movies from the Edge of Space-Time.

[129] Ng, G. (2001). Information Visualization Resources: People. HTML Document at http://www.cs.man.ac.uk/~ngg/InfoViz/People/.

[130] Nielsen, J. (1990). Miniatures versus icons as a visual cache for videotex browsing. Behaviour & Information Technology, 9 (6), 441-449.

[131] Nielsen, J. (1993). Usability Engineering. Academic Press Professional, Boston.

[132] Nielsen, J. (1994). Guerrila HCI: Using Discount Usability Engineering to Penetrate the Intimidation Barrier. In R. G. Bias and D. J. Mayhew (eds). Cost-Justifying Usability, Chapter 11, Academic Press Professional.

[133] Nielsen, J. and Levy, J. (1994). Measuring Usability: Preference vs. Performance. Communications of the ACM, 37 (4), 66-75.

[134] Noik, E. G. (1994). A Space of Presentation Emphasis Techniques for Visualizing Graphs. Proc. of Graphics Interface '94 (May 16-20, Banff, Alberta, Canada), Morgan-Kaufmann Publishers, 225-234.

[135] North, C. and Shneiderman, B. (1997). A Taxonomy of Multiple Window Co-ordinations, Technical Report CS-TR-3854, University of Maryland, December 1997, 1997.

[136] North, C. L. (2000). A User Interface for Co-ordinating Visualization Based On Relational Schemata: Snap-Together Visualization, PhD Thesis, Department of Computer Science, Univeristy of Maryland, College Park. pp 179.

[137] Osawa, N. (2001). A Multiple-Focus Graph Browsing Technique Using Heat Models and Force-Directed Layout. Proc. of IV 2001 (July 25-27, London, UK), IEEE Computer Society Press, 277-283.

[138] Parker, G., Franck, G. and Ware, C. (1998). Visualization of Large Nested Graphs in 3D: Navigation and Interaction. Journal of Visual Languages and Computing, 9 (5), 299-317.

[139] Parunak, H. V. D. (1989). Hypermedia Topologies and User Navigation. Proc. of ACM Hypertext '89 (Nov. 5-8, Pittsburgh, USA), ACM Press, 43-50.

[140] Parunak, H. V. D. (1991). Don't Link Me In: Set Based Hypermedia for Taxonomic Reasoning. Proc. of ACM Hypertext '91 (December 15-18, San Antonio, Texas, USA), ACM Press, 233-242.

[141] Plunkett, G. M., Soltis, D. E. and Soltis, P. S. (1996). Evolutionary patterns in Apiaceae: inferences based on *matK* sequence data. Systematic Botany, 21, 477-495.

[142] Pullan, M. R., Watson, M. F., Kennedy, J. B., Raguenaud, C. and Hyam, R. (2000). The Prometheus Taxonomic Model. Taxon, 49 (1), 55-75.

[143] Purchase, H. C., Cohen, R. F. and James, M. (1995). Validating Graph Drawing Aesthetics. Proc. of Graph Drawing '95 (September 20-22, Passau, Germany), Springer-Verlag, 435-446.

[144] Raguenaud, C., Graham, M. and Kennedy, J. (2001). Two approaches to representing multiple overlapping classifications: a comparison. Proc. of SSDBM 2001 (July 18-20, Fairfax, Virginia, USA), IEEE Computer Society Press, 239-244.

[145] Raguenaud, C., Kennedy, J. and Barclay, P. J. (2000). The Prometheus Taxonomic Database. Proc. of 12th International Conference on Scientific and Statistical Database Management (July 26-28, Berlin, Germany), IEEE Computer Society Press, 250-252.

[146] Rekimoto, J. and Green, M. (1993). The Information Cube: Using Transparency in 3D Information Visualization. Proc. of Third Annual Workshop on Information Technologies & Systems (WITS'93) (December 5, Orlando, Florida, USA), 125-132.

[147] Robert, L. and Lecolinet, E. (1998). Browsing hyperdocuments with multiple focus+context views. Proc. of ACM Hypertext '98 (June 20-24, Pittsburgh, Pennsylvania, USA), ACM Press, 293-294.

[148] Roberts, J. C. (1998). On Encouraging Multiple Views for Visualization. Proc. of IEEE International Conference on Information Visualization '98 (July 29-31, London, UK), IEEE Computer Society Press, 8-14.

[149] Robertson, G. G. (2000). From Hierarchies to Polyarchies: Visualizing Multiple Relationships, Invited Talk, User Interface Research, Microsoft Research, November 30, 2000, 2000.

[150] Robertson, G. G., Mackinlay, J. D. and Card, S. K. (1991). Cone Trees: Animated 3D Visualizations of Hierarchical Information. Proc. of CHI '91 : Human Factors in Computing Systems (April 27 - May 2, New Orleans, Louisiana, USA), ACM Press, 189-194.

[151] Ruths, D. A., Chen, E. S. and Ellis, L. (2000). Arbor 3D: an interactive environment for examining phylogenetic and taxonomic trees in multiple dimensions. Bioinformatics, 16 (11), 1003-1009.

[152] Sarkar, M. and Brown, M. H. (1994). Graphical fisheye views. Communications of the ACM, 37 (12), 73-84.

[153] Schaffer, D., Zao, Z., Greenberg, S., Bartram, L., Dill, J., Dubs, S. and Roseman, M. (1996). Navigating hierarchically clustered networks through fisheye and full-zoom methods. ACM Transactions on Human-Computer Interaction, 3 (2), 162-188.

[154] Shackel, B. (1991). Usability - Context, Framework, Definition, Design and Evaluation. In B. Shackel and S. Richardson (eds). Human Factors for Informatics Usability, 21-37. Cambridge, Cambridge University Press.

[155] Shneiderman, B. (1996). The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. Proc. of IEEE Visual Languages '96 (September 3-6, Boulder, Colorado, USA), IEEE Computer Society Press, 336-343.

[156] Shneiderman, B., Feldman, D., Rose, A. and Grau, X. F. (2000). Visualizing Digital Library Search Results with Categorical and Hierarchical Axes. Proc. of Digital Libraries '00 (June 2-7, San Antonio, Texas, USA), ACM Press, 57-66.

[157] Shneiderman, B. and Wattenberg, M. (2001). Ordered Treemap Layouts. Proc. of InfoVis 2001 (October 22-23, College Park, Maryland, USA), Human-Computer Interaction Lab, University of Maryland, 73-78.

[158] Siirtola, H. (2000). Parallel Coordinate Explorer. Java 1.1 Applet at http://www.cs.uta.fi/~hs/pce/.

[159] Smith, S. P. and Duke, D. J. (2000). Binding Virtual Environments to Toolkit Capabilities. Computer Graphics Forum, 19 (3), 81-89.

[160] Sprenger, T. C., Gross, M. H., Eggenberger, A. and Kaufman, M. (1997). A Framework for Physically-Based Information Visualization. Proc. of Visualization in Scientific Computing '97 (April 28-30, Boulogne-sur-Mer, France), SpringerWienNewYork, 71-83.

[161] Stasko, J., Guzdial, M. and McDonald, K. (1999). Evaluating Space-Filling Visualizations for Hierarchical Structures. Proc. of IEEE InfoVis '99, Late Breaking Hot Topics (October 25-26, San Francisco, California, USA), IEEE Computer Society Press, 35-38.

[162] Stasko, J. and Zhang, E. (2000). Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations. Proc. of IEEE Symposium on InfoVis 2000 (October 9-10, Salt Lake City, Utah, USA), IEEE Computer Society Press, 57-65.

[163] Stroe, I. D., Rundensteiner, E. A. and Ward, M. O. (2000). Scalable Visual Hierarchy Exploration. Proc. of Database and Expert Systems Applications (DEXA) 2000 (September 4-8, London, UK), Springer-Verlag, 784-793.

[164] Tognazzini, B. (1992). User testing on the cheap. TOG on Interface, 79-89, Chapter 14, Addison-Wesley.

[165] Tufte, E. R. (1991). Envisioning Information. Graphics Press, Cheshire, Connecticut.

[166] Turo, D. and Johnson, B. (1992). Improving the Visualization of Hierarchies with Treemaps: Design Issues and Experimentation. Proc. of IEEE Visualization '92 (October 19-23, Boston, Massachusetts, USA), Computer Society Press, 124-131.

[167] Tversky, O. J., Snibbe, S. S. and Zeleznik, R. (1993). Cone Trees in the UGA Graphics System: Suggestions for a More Robust Visualization Tool, Technical Report CS-93-07, Brown University, February 1993, 1993.

[168] Tweedie, L., Spence, R., Dawkes, H. and Hua, S. (1996). Externalising abstract mathematical models. Proc. of ACM CHI '96 (April 13-18, Vancouver, Canada), ACM Press, 406-412.

[169] Upson, C., Faulhaber Jr, T., Kamins, D., Laidlaw, D., Schlegel, D., Vroom, J., Gurwitz, R., et al. (1989). The Application Visualization System: A Computational Environment for Scientific Visualization. IEEE Computer Graphics & Applications, 9 (4 - July), 30-42.

[170] Ware, C. and Franck, G. (1996). Evaluating Stereo and Motion Cues for Visualizing Information Nets in Three Dimensions. ACM Transactions on Graphics, 15 (2), 121-140.

[171] Weiss-Lijn, M., McDonnell, J. T. and James, L. (2001). Visualising Document Content with Metadata to Facilitate Goal-directed Search. Proc. of IV 2001 (July 25-27, London, UK), IEEE Computer Society Press, 71-76.

[172] van Wijk, J. J. and van de Wetering, H. (1999). Cushion Treemaps: Visualization of Hierarchical Information. Proc. of IEEE InfoVis '99 (October 25-26, San Francisco, California, USA), IEEE Computer Society Press, 73-78.

[173] Wills, G. J. (1997). NicheWorks - interactive visualization of very large graphs. Proc. of Graph Drawing '97 (September 18-20, Rome, Italy), Springer-Verlag, Berlin, 403-414.

[174] Wittenburg, K., Das, D., Hill, W. and Stead, L. (1995). Group Asynchronous Browsing on the World Wide Web. Proc. of Fourth International World Wide Web Conference (December 11-14, Boston, Massachusetts, USA), 51-62.

[175] Wittenburg, K. and Sigman, E. (1997). Visual Focusing and Transition Techniques in a Treeviewer for Web Information Access. Proc. of Visual Languages '97 (September 23-26, Capri, Italy), IEEE Computer Society Press, 20-27.

[176] Wood, A., Drew, N., Beale, R. and Hendley, B. (1995). HyperSpace: Web Browing with Visualization. Proc. of Third International World-Wide Web Conference (April 10-14, Darmstadt, Germany), 21-25.

[177] Xiong, R., Smith, M. and Drucker, S. (1998). Visualizations of Collaborative Information for End-Users, Technical Report MSR-TR-98-52, Microsoft Research, October 7th 1998, 1998.

[178] Zhai, S., Buxton, W. and Milgram, P. (1996). The Partial-Occlusion effect: Utilizing Semi-Transparency in 3D Human-Computer Interaction. ACM Transactions on Human-Computer Interaction, 3 (3), 254-284.

Appendix A – List of Tasks for First User Test

Test Tasks

1. Where does genus *Athamanta* appear in the classifications?

2. Who uses *Mulineae* at rank tribe?

3. Who first used genus *Foeniculum*?

4. What are the members of *Selineae* in Berchtold & Presl's classification?

5. In which tribes or sub-tribes do these genera appear in the other classifications (from Q. 4)?

6. What is the spread of other genera that the classifications have grouped with these genera (from Q. 5)?

7. How many levels more does Bentham's classification use compared to Berchtold & Presl's?

8. What is the distribution of all sibling genera of genus *Kundmannia*?

9. What groupings of genera in Koch's classification remain together in De Candolle's classification? (i.e. are not re-distributed among two or more groupings)

10. What groupings of genera in Koch's classification are **exactly** the same in De Candolle's classification?

11. Which genera are unique to Berchtold & Presl's classification?

12. Which genera are first used in De Candolle's classification?

Observations and Inventions from the First Test

This document describes the observations made when users encountered problems during the first test. Using Monk and Wright's methodology, each observation is countered with an *invention*, put forward as a putative solution to the observed problem.

Graph Interface

Observation: Overlapping nodes caused problems, especially when selecting a certain node.
Invention: Try to distribute nodes with regard to their overall shape (i.e. the nodes are always wider than tall). Always have the chosen node visible at the 'top' of the display.

Obs: Non-linked but highlighted nodes found to be confusing rather than helpful when viewing only a subset of the trees in the graph.
Inv: Stop highlighting sibling nodes if the tree they are related in is not displayed (and hence the links connecting it are not displayed)

Obs: Lack of spatial ordering of different ranks caused difficulties in rank-related tasks.
Inv: Perhaps some indicator within the node to show rank (apart from italicisation for genera). Spatial ordering of the ranks within a force directed graph is too restrictive, as shown by the 'rainbow'-style graph visualisation.

Obs: When shown, the users preferred to visualise one tree at a time for most tasks
Inv: Bluntly, appears to show that the idea of merging all the trees into one graph wasn't a good one.

Obs: Selecting some nodes, specifically the non-leaf nodes, could produce an overwhelming amount of links.
Inv: Perhaps indicates the same as the above! A problem all graph/network visualisations appear to suffer from, even when filtering is used.

Obs: Finding unique/"first classified" nodes in the classifications was achievable by filtering out the trees. Also by looking at the glyphs in the corner of the nodes. Though it did take some individual inspection due to the overlapping problem.
Inv: Perhaps a function for highlighting unique nodes and reduction of overlapping.

Bug:
Sometimes clicking with left mouse button on the background highlighted a node. Need to restrict node picking to the actual node and not the nearest node within a certain distance.

Multiple Tree Interface

Obs: Endings of names in taxonomy indicate rank, but these are abbreviated in the display. Require a way to view rank.
Inv: Indication of rank by the side of each level in the display

Obs: Some tasks require the distribution of two genera/families to be interpreted (for example Q.6)
Inv: Allow multiple choices on-screen that merge a number of distributions using colour. This could only be allowed for single colour choices, as showing a number of visualisations that utilise multiple colours side-by-side would be unintelligible. In effect, we require a sibling function for non-genera nodes.

Obs: Choosing a non-genus from the list will highlight the genera it contains and not the actual non-genera node itself, leading to confusion as to which node in which tree on-screen was actually picked. Can also indicate that the non-genera node is present in more classifications than it actually is, if its genera are spread across more classifications.
Inv: Highlight the chosen non-leaf node in the chosen tree.

Obs: Sibling node function was found to be confusing.

Inv: The function was found to be useful when its purpose was understood. A more detailed explanation of the function is necessary on the display.

Obs: Users tended to use single colour distribution, though preferred the multiple colour distributions when shown them.
Inv: Make it clearer by showing an example that multiple colour distributions can be produced.

Obs: Could only find nodes unique to each classification by individual inspection (very long-winded)
Inv: Have a different grey-scale colouring for unique nodes, and even for nodes that are first recognised within a particular classification.

Obs: When viewing the distribution of genera within a certain rank, if new sub-ranks were present in other classifications, the genera were spread across several sub-sets, indicating a wider distribution than was actually the case.
Inv: Ability to remove the effect of certain ranks to compare like against like. In effect, rolling up the genera towards the top of the classifications.

Appendix B – Scenarios for Second User Test

Scenario

The aim of this test of the visualisation is to concentrate on finding problems in the user interface that need tackling. To do this, there are 3 scenarios to follow that involve the use of all the parts of the interface we'd like to test. The scenarios also cover all the tasks that were covered during the first test.

You can quit or skip parts, but we'd like you to try and tackle problems you are experiencing, as this is where we derive a lot of the information on a particular problem from.

As you progress, we'd like you to think-aloud during the test, describing why certain situations are problematic, and why you have chosen to execute certain actions.

1) Start by using the control panel at the bottom of the screen to display all the available classifications. Once they have appeared, and looking at the general structure of the displayed hierarchies, which appears to hold the most information? Which appears to have the deepest structure?

   Deciding to at first experiment with the diagram, use the list at the right-hand side of the screen, and select the *Mulineae* taxon to discover how many classifications use this name at the tribe rank. Which is the first classification to use this taxon. Using the mouse pointer to investigate the diagram on-screen, can you discover the names of the genera that make up this first example of *Mulineae*? You then decide to inspect two different, individual genera, such as *Athamanta* and *Foeniculum*. Select these using the list.

2) During the course of your work, you decide that you need to discover the distribution of the *Molopospermum* genus across taxonomies previous to 1930. Using the visualisation from its starting state, attempt to derive the visualisation that will show this information. You will need to display the relevant taxonomies, and select the *Molopospermum* genus from the list on the right-hand side of the screen.

   From this you can discover the tribes and sub-tribes *Molopospermum* is classified under in the different taxonomies. Does there appear to be any over-riding pattern to *Molopospermum's* classification?

   You decide that more information is needed to discover the context of *Molopospermum's* distribution, and you decide that viewing the distribution of its sibling nodes could show rationale behind its classification. After activating the sibling mode function, view this information by left-clicking on one of *Molopospermum's* representations within the main diagram.

Can you distinguish any pattern or lack of pattern to the visualisation that appears?

What does Cerceau-Larrival's classification show with regard to *Molopospermum*?

3) Later, you decide that a swift comparison of Koch's and De Candolle's classifications is required. Making sure the brush mode is on and that the sibling mode is off, examine Koch's classification at the tribe level for groupings of genera that appear to match **exactly** one of De Candolle's tribes. Don't mouse-click on any of the nodes on the screen or within the list. Then re-examine Koch's classification for tribes whose genera are wholly contained within a group in De Candolle's classification i.e. are not split between two or more tribes.

    Following on from this, you wish to discover which genera De Candolle first classified. Firstly, remove from the display all the classifications after De Candolle. Using the on-screen diagram, click on the top-level ranks of Berchtold & Presl's and Koch's classifications. From this diagram can you see the genera first used by De Candolle?

    Use the 'Invert to 1 Colour' button to swap the appearance of all the highlighted and greyed-out nodes.

    Is it apparent from the list of taxons on the right-hand side which are presently or not presently highlighted in De Candolles' classification?

    Display all the available classifications.

    Now it occurs that although De Candolle's is the later and bigger classification, there may be nodes in Koch's classification that De Candolle did not use, and could be in fact totally unique to Koch's classification. Click with the left mouse button on each of the top level ranks in the classifications apart from Koch's. You may clear the previous selections by using the button on the on-screen control panel or by making your first top-rank selection with the mouse's right-button.

    What genera appear to be unique to Koch? You may use the control panel buttons to reduce the colours on screen or invert the display if you think it would make the display easier to understand.

    Is it apparent from the list of taxons on the right-hand side which are presently or not presently highlighted in Koch's classification?

Problems found as a result of the Second User Test

Major:

- List – redo totally
    - Need focus with mouseOver() not mouseClick().
    - Ability for multiple background colours.
    - Extra selection cue (arrow? Bold font?)
    - Represent input string for keypresses (similar to Word / Kawa lists.)

- Classification hiding / displaying
    - To be done on-screen (in visualisation.)
    - Names and classifications to be visually tied together by tab-style or card display.
    - Names to remain on-screen if classification hidden, acting as reminders.
    - Have separate ‘display all hierarchies’ button.

- Rename ‘clear colours’ to ‘clear selections’

- Brighter colour spectrum
    - Linear colour scale appears dark and non-linear in brightness.

- Step-back undo function
    - To complement or replace history bar. Maybe have last five or ten selection states stored so user can retrieve them with undo button/dialog. Keep a maximum of five/ten last clicked nodes for easy reference (too many means it’s too much trouble to look through them all).
    - Left/right mouse click on bar decides whether to display just selected node or all states up to selected node.

Minor:

- Grey out sibling checkbox when over non-genera. Don’t remove the tick.

- Shortened rank indicators on right-hand side of classification (first letter, abbreviation?)

- Re-label sibling mode as ‘sibling mode on next selection’?

- Examine whether name -> rank -> classification indicator is necessary. If it is, associate first classification with it, not last, in arbitrary situations.

- User wanted to in direct selection:
    - Re-colour directly selected node if already coloured.
    - But another user wanted semantics to be that previously selected nodes were not to be overwritten with new selection colour. (Maybe enforce this rule with exception for explicitly-selected nodes? Or another mode checkbox?)

- Have mouse info offset adjust itself according to mouse (x, y) position on screen, so that no positioning of the mouse pointer will push the mouse-tip info off-screen.

Appendix C – Scenarios for Third User Test

Scenario

The aim of this test of this taxonomy visualisation is to concentrate on finding problems in the user interface that need tackling. To do this, there are 3 scenarios to follow that involve the use of all the parts of the interface we'd like to test.

You can quit or skip parts, but we'd like you to try and tackle problems you are experiencing, as this is where we derive a lot of the information on a particular problem from.

As you progress, we'd like you to think-aloud during the test, describing why certain situations are problematic, and why you have chosen to execute certain actions.

1) Start by using the control panel at the right-hand side of the screen to display all the available classifications. Once they have appeared, and looking at the general structure of the displayed hierarchies, which appears to hold the most information? Which appears to have the deepest structure?

   Deciding to at first experiment with the diagram, use the list at the right-hand side of the screen, and select the *Mulineae* taxon to discover how many classifications use this name at the tribe rank. Which is the first classification to use this taxon. Using the mouse pointer to investigate the diagram on-screen, can you discover the names of the genera that make up this first example of *Mulineae*? You then decide to inspect two different, individual genera, such as *Athamanta* and *Foeniculum*. Select these using the list.

2) During the course of your work, you decide that you need to discover the distribution of the *Molopospermum* genus across taxonomies previous to 1930. Using the visualisation from its starting state, attempt to derive the visualisation that will show this information. You will need to display the relevant taxonomies, and select the *Molopospermum* genus from the list on the right-hand side of the screen.

   From this you can discover the tribes and sub-tribes *Molopospermum* is classified under in the different taxonomies. Does there appear to be any over-riding pattern to *Molopospermum's* classification?

   You decide that more information is needed to discover the context of *Molopospermum's* distribution, and you decide that viewing the distribution of its sibling nodes could show rationale behind its classification. After activating the sibling mode function, view this information by left-clicking on one of *Molopospermum's* representations within the main diagram. All the groups that *Molopospermum* is associated with through the set of classifications are displayed using different colours.

Can you distinguish any pattern or lack of pattern to the visualisation that appears?

What does Cerceau-Larrival's classification show with regard to *Molopospermum*?

Using the bar on the bottom of the screen, see if you can re-select the non-sibling, previous, *Molopospermum* appearance of the visualisation.

3) Later, you decide that a swift comparison of Koch's and De Candolle's classifications is required. Make sure the brush mode indicator on the right-hand control panel is on (ticked) and that the sibling mode indicator is off (empty). Examine Koch's classification at the tribe level for groupings of genera that appear to match **exactly** one of De Candolle's tribes, but don't highlight the nodes by pressing the mouse buttons. Then re-examine Koch's classification for tribes whose genera are wholly contained within a group in De Candolle's classification i.e. are not split between two or more tribes.

   Following on from this, you wish to discover which genera De Candolle first classified. Firstly, shrink/hide all the classifications after De Candolle. Using the on-screen diagram, click on the top-level ranks of Berchtold & Presl's and Koch's classifications. From this diagram can you see the genera first used by De Candolle?

   Use the 'Invert to 1 Colour' button to swap the appearance of all the highlighted and greyed-out nodes.

   Is it apparent from the list of taxons on the right-hand side which are presently or not presently highlighted in De Candolles' classification?

   Display all the available classifications.

   Now it occurs that although De Candolle's is the later and bigger classification, there may be nodes in Koch's classification that De Candolle did not use, and could be in fact totally unique to Koch's classification. Click with the left mouse button on each of the top level ranks in the classifications apart from Koch's. You may clear the previous selections by using the button on the right-hand control panel or by making your first top-rank selection with the mouse's right-button.

   What genera appear to be unique to Koch? You may use the control panel buttons to reduce the colours on screen or invert the display if you think it would make the display easier to understand.

   Is it apparent from the list of taxons on the right-hand side which are presently or not presently highlighted in Koch's classification?

User 1

1) Problem

Observation
Still having Mulineae-Mulinum confusion. User thinks they have double clicked

Solutions
- Stronger highlighting for primary selection.
- Indication of rank may remove the confusion. Could user then see that Mulinum is a genus of Mulineae?
- Organise list by category, and alphabetically within category?

2) Problem

Observation
Keypresses used to navigate list, but mouse used to make final selection. User is either not seeing current node name at top of list, or is unsure of whether they can make a selection by pressing return.

Solutions
- Stronger colour / bigger font for current node name.
- Icon after name indicates possibility of pressing return.

3) Problem

Observation
Second colour (blue) on selected scale is too dark.

Solutions
- Darken background.
- Iso-luminant colour scale.
- Remove second colour choice from colour list.

4) Problem

Observation
Brushing occasionally gave impression of more colours in sibling mode, rather than one particular set of colours at two levels of brightness.

Solution
- Ignore brush mode in sibling mode?
- Better selection of colour scale (iso-luminant again?) – see Problem 3

5) Problem

Observation
User had no knowledge that selecting with the right-hand mouse button performed a ‘clear selection’ operation prior to the chosen selection.

Solutions
- Mention this in instructions?
- Some sort of appropriate cursor?

6) Problem

Observation
Browsing behaviour causes same problem in history bar as browsing in WWW does. Pushes previous states off stack and cannot backtrack after so many selections.

Solutions

7) Problem

Observation
Sometimes user clicked on nodes when trying to close hierarchies. Names too close to nodes?

Solutions

- Have mouse tool-tip on show / hide hierarchies area?
- Change cursor over appropriate area?

8) Problem

Observation

Grey-scale brushing occasionally doesn’t attract attention at edge of user’s visual field.

Solutions

- Blinking behaviour after half a second or so of same brush selection?
- Blinking compulsory at certain distance from mouse pointer (‘x’ pixels+). Sounds nasty!

9) Problem

Observation

User found meaning of greyed-out nodes in list to be unintuitive. Understood once explained.

Solutions

User 2

10) Problem

Observation

User didn’t use keypresses at all on list. This, however, is a common mechanism, especially in Windows.

Solutions

- User doesn’t have to use keypresses. May be too much hassle for some switching between keyboard and mouse.

11) Problem

Observation

User appeared to try and find nodes in an instance of Mulineae by clicking on it, and expecting the nodes to expand out from it.

Solutions

- The Zoom / DOI interface?

12) Problem

Same as Problem 4

13) Problem

Observation

Found fonts hard to read. Had to lean forward.

Solutions

- Anti-aliased fonts.
- Extend Zoom/DOI interface to categorynodes and font sizes?

14) Problem

Same as Problem 3

15) Problem

Observation

User couldn’t find previous state from history bar, even though it was there.

Solutions

- Arrow of time on history bar: Recent -> Previous.
- Have last selection on left-hand side of history bar (to avoid last state drifting off right-hand side of screen.)
- Better indication of sibling / simple selections:
- One colour for simple node selections.

- Multi-colour swatch for sibling node selections.
- Grey background for control panel actions.

16) Problem

Observation

Hiding and displaying classifications not intuitive. But user found them due to lack of other choices.

Solutions

- Have HIDE-SHOW tags prefixed to layer names in visualisation, depending on current visibility state of layer.
- Mouse tooltip? But then only visible whn over area.

17) Problem

Observation

Layer names tended to be associated with classifications above them in some situations, which is incorrect.

Solutions

- Reduce gap between layer name and correct hierarchy. Increase gap to hierarchy above with this saved space.
- Make tab graphic more noticeable.

Note: This was an introduced error after user test 2. New_Drude classification had 2 top-level nodes, which required a virtual top node to tie them together in the model. Unfortunately, this virtual node was then carried through to the visualisation, resulting in a gap below the layer name.

18) Problem

Observation

User kept re-starting task that involved finding unique nodes in the classifications. Discovered that user thought that too much change was occurring on-screen after a high-level node selection.

Solutions

- Have users perform mid-rank-level tasks. The difference to the user between selecting one or ten nodes to selecting hundreds of nodes is a bit too much of a jump to be taken easily.

User 3

19) Problem

Observation

Language mismatch between ‘classifications’ and ‘hierarchies’. A classic “speak the user’s language” problem. cf Nielsen’s heuristics.

Solutions

- Rename button appropriately.
- Have resource file with button names etc, similar to X-Windows resource files. (sounds like a big undertaking.)

20) Problem

Observation

User kept one classification hidden by accident. Didn’t notice till later in the test.

Solutions

- Perhaps the hide-show indicators as stated in Problem 16 would help, as the rogue classification would be the only one saying ‘show’ instead of ‘hide’.

21) Problem

Same as Problem 1.

22) Problem

Observation

Smaller screen resolution causing problems. Had to scroll display.

Solutions
- Squeeze hierarchies on Y-axis, similar to X-axis squeeze? Could make visualisation too small.
- Stop panning in X-axis as it is now unnecessary as information is guaranteed to fit horizontally. Can only push information off-screen if we keep this behaviour.

23) Problem
Observation
User stated that they would only be interested in sometimes seeing one particular rank, and wouldn't want to see its contents.

Solutions
- Incorporate level/rank hiding mechanism. Make bottom visible nodes to be leaves.

24) Problem
Observation
Slow computer made user double-click as it was slow to respond. Typical error in interactive systems, and Java isn't well suited for low-spec machines.

Solutions

25) Problem
Observation
User expected list to work with cursor keys, and to move current node to centre of list.

Solutions
- Implement this behaviour, apart from…
- …Move current node in keypress behaviour to mouse y-coord, not centre of list.

26) Problem
Observation
User wondered why some nodes were appended with 'ST'.

Solutions
- Leftover notation from start of project. Remove it.

27) Problem
Same as Problem 13.

28) Problem
Observation
Some relevant information was pushed off the bottom of the screen, due to low resolution.

Solutions
- Set node size relative to screen resolution and monitor size.
- Also see Problem 22.

29) Problem
See Problem 15. User confused as to colours of history bar nodes.

30) Problem
Observation
Text greying and white/black colouring in list was not intuitive to user.

Solutions
- Have white background in list replaced by grey background for unselected nodes, to match unselected nodes in main visualisation. Will stop confusion over text colour being related to selection colour.

31) Problem
Observation

User was required to scroll entire list to find unique nodes in a classification after task was completed.

Solutions

- Could just investigate them in the visualisation.

32) Problem

Observation

User found ‘unique’ nodes that were not present in any of the classifications available to them. (These were nodes from New_Drude, which wasn’t available to them.)

Solutions

- Bar nodes from list that aren’t referenced in given node-pair files.

User 4

33) Problem

See Problem 19.

34) Problem

See Problem 17.

35) Problem

See Problem 16.

36) Problem

Observation

Some greyed-out nodes in the list still caused brush highlighting effects on visualisation. This was due to them being higher-level nodes that contain nodes visible in other classifications.

Solutions

- See Problem 1 for indication of rank in list. If brushed node is high-level node and user can see that lower-level nodes are being highlighted as a result of brushing on it, this may help them understand the situation.
- Greyed-out nodes to be unselectable by brush mode? After all, they are not selectable in the main visualisation, due to them being hidden.

37) Problem

See Problem 18.

User 5

38) Problem

See Problem 19.

39) Problem

See Problem 16.

40) Problem

Observation

User attempted to match nodes singly, rather than by sets at a time. Discovered they could match by sets after mouse fortuitously moved upwards.

Solutions

- Reckon this will be found by experience.

User 6

41) Problem

Observation

Keypresses in list resets after so many seconds. No indication of this to user. Also, pressing just return causes the list to select whatever node is at the halfway index in the list.

Solutions

- Have timebar or fade-out on selected name label at top of list.
- Stop the anomalous ‘return’ key behavior.

42) Problem

See Problem 1.

43) Problem

Observation

User suggests having rank indicated in pop-up mouse tip, so they don’t have to scan to sides of visualisation to discover rank information.

Solutions

- Implement this behaviour. This will be especially handy for category nodes.
- Have rank ‘pop-up’ after half a second or so?

44) Problem

Observation

User made incorrect but intuitive association of greyed-out text nodes in list, with grey nodes in visualisation.

Solutions

- Dark grey background in list for unselected nodes. Will need light-grey text to show up against this background/.

45) Problem

See Problem 1.

46) Problem

See Problems 1 & 36.

Appendix D – Questionnaires & Scenario for Fourth User Test

Multiple Taxonomy Interface Test : Pre-Questionnaire

Hello, and thanks for agreeing to be part of my interface testing! This is a final round of testing to validate the development of a tool that visualises correlations between multiple taxonomies based on the same data sets. Before the sit-down practical testing of the tool, I'd be grateful if you would fill in this pre-questionnaire. When you've finished, could you e-mail it back to me, or if completed by hand, pass it on to Mark W?

Martin Graham (marting@dcs.napier.ac.uk)

Pre-Questionnaire

1) Is comparing different taxonomies of the same family/genera an activity you would consider undertaking using paper-based information? If not, why not?
A:

2) Is comparing different taxonomies of the same family/genera/etc something you would explore if feasible? If so, in what context would this activity be undertaken?
A:

3) Would you think of multiple taxonomies as one merged object/structure or as many related but separate structures, or as something else?
A:

4) Which is the more important consideration for your work, time taken or accuracy?
A:

5) Do you think a computer-based interface tool to query multiple classifications would be useful in your work?
A:

6) Have you previously encountered any computer programs or interfaces that display a taxonomy graphically?
A:

The actual test (dates to be arranged) should take no more than an hour, and will consist of a number of tasks using our visualisation tool, and a questionnaire of approximately twice this size afterwards**.**

The results of the test will form part of my PhD at Napier, but no individual user identities will be disclosed!

Post-Questionnaire

1) Which tasks did you find especially easy or difficult? Why?

A:

2) Which tasks/features did you find especially useful or irrelevant? Why?

A:

3) In general, which did you find more difficult: understanding the input controls to perform a certain task, or understanding the visualisation display after you selected the necessary inputs for a task?

A:

4) What, in your view, are the bad points of this visualisation?

A:

5) What, in your view, are the good points of this visualisation?

A:

6) Would this tool encourage you to attempt more work with multiple taxonomies?

A:

7) Further general or specific comments on the visualisation tool:

A:

Apiaceae Task Set

Scenario

Having recently generated a new taxonomy for Apiaceae based on DNA analysis, it occurs that it may be useful to compare the new classification against previous classifications of the same family. This will display where the DNA character data matches, at least partially, previous classifications, which may or may not have been regarded as important until now. Such matches, when combined with the methodology behind the previous classifications, could give information on how the plant sequencing influences certain aspects of the plant's characteristics.

General Information

By looking at and then interacting with the visualisation, can you tell…

- Which classification has the most taxa?
- Which classification has the most taxonomic ranks?
- Where is the biggest group of genera in the classifications? And how many genera are in that group?

Single Selections

Select `Peucedanum` from the list.

- Which classification does it not occur in?
- What is the most common name of the tribe that `Peucedanum` belongs to across the classifications?
- What is unusual about `Peucedanum` in the New Drude classification?

Set the overwrite option box so it is ticked.
Clear the previous selections of coloured taxa.
Select `Athamanta` from the list.

- What is unusual about `Athamanta` in the New Drude classification?

Clear the overwrite option box.
Select the 3 groups in which `Athamanta` occurs in New Drude.

- Does `Athamanta` have any strong previous association with these groups?

Group Selections

Set the overwrite option box so it is ticked.
Clear the previous coloured taxa selections.
Select the `‘Scandiceae’` tribe in the New Drude classification.

- How many classifications use this name at the *tribe* rank?
- In Koso-Poljansky's classification, what proportion of taxa under `Scandiceae` are coloured green, roughly a third, a sixth or a tenth?.
- After attempting this first time, hide the sub-tribe and grex ranks and set the ‘Sort by Selection’ tick-box to ‘ticked’. What is your guess at the proportion now?
- Which genera of Berchtold & Presl's `Scandiceae` are present in the New Drude classification?

Sibling selections

Reset the 'Sort by Selection' option box to empty. Make the sub-tribe and grex ranks visible again.
Clear all selections.
Set the 'Show siblings on next' option box to 'ticked'.
Select `Molopospermum` from the list.

- Which classification does `Molopospermum` not occur in? Hide this classification.
- Can you see a pattern to the way `Molopospermum` is grouped with other genera across the classifications, or a lack of one?

Hide the Sub-Tribe and Grex ranks.

- Does this make a sibling pattern appear in any of the classifications?

Brush Manipulations

Noticing a possible correlation emerging between the historical Koch and De Candolle classifications, you decide to investigate further by matching distribution of genera by tribe between the two classifications. You can do this by 'brushing' the classifications i.e. by moving the mouse pointer over a single taxon or a group of taxa, which will highlight the same taxa temporarily in other classifications.

Clear all selections.
Clear the sibling mode option box.
Make all ranks visible.
Hide all hierarchies except Koch and De Candolle.

- *Brush* all the groups of genera in Koch, one at a time. Find which groups *exactly* (no missing and no extra) match groupings in De Candolle.

- Re-examine groups in Koch for groups whose genera, present or not in De Candolle, are not split between two or more groups in De Candolle i.e. are full or partial subsets of groups in De Candolle.

- Now for each of the groups found in the previous question, brush the corresponding (containing) groups in De Candolle. Judge whether these groups in De Candolle take genera from other places in Koch beside the original group, or consist of the original Koch group with additional new genera names, or a mixture of both cases.

Classification Manipulations

Highlight the taxa De Candolle first classified by selecting ('clicking' on) the top level `Apiaceae` rank of Berchtold & Presl and then of Koch, and then finally inverting the highlighted display.

- What proportion of taxa in De Candolle are new names, roughly 25, 50, or 75%?
- How are the taxa first used by De Candolle highlighted in the list?

Re-display all the classifications.
Clear selections.
Select ('Click') the `Apiaceae` taxon at the Family rank in all classifications *except* New Drude.
Invert the highlighted nodes.
This should now show all the unique names in New Drude.

- How many genera are unique to New Drude?
- What are their names?
- Which Family name is unique to New Drude?

Globba Task Set

Scenario

You are currently revising the specimen set within the Thai Globba genus and wish to answer some questions about the evolution of the revisions you have completed so far. This will allow you to discover what specimens were classified when and how the revision has developed over time.

General Information

By looking at and then interacting with the visualisation, can you tell:

- Which classification has the most specimens?
- Which classification has the least taxonomic ranks?
- Where is the biggest group of specimens in the classifications? And how many specimens are in that group?

Single Selections

Select “`AAU 76-467 J. F. Maxwell (1976)`” from the right-hand side list.

- Which classification or revision does it not occur in?
- In which of the modern revisions was “`AAU 76-467 J. F. Maxwell (1976)`” classified outside of the `incertae-sedis` unassigned rank?

Set the ‘Overwrite’ option box to clear.
Select the section taxa in which “`AAU 76-467 J. F. Maxwell (1976)`” occurs in Newman’s 10/10/2000 revision.

- In which revision are most of the highlighted specimens pulled out of the unassigned group and classified?
- Which specimen in the ‘`mn-sector sectcer`” section in the 3/10/2000 revision appears to have been dropped in the later 10/10/2000 revision?

Select “`LINN Cat.Linn.Herb. 45/1`” from the list.

- What species is “`LINN Cat.Linn.Herb. 45/1`” assigned to, when it is not in the unassigned group?

Group Selections

Set the ‘Overwrite’ option box to ‘ticked’.
Clear the previous selections of coloured taxa.
Select “`Globba section Globba L.`” in Newman’s Thai Globba 10/10/2000 revision.

- How many revisions use this name at the Section rank?
- In Newman’s 3/10/2000 revision, what proportion of specimens under “`Globba section Globba L.`” are coloured green, roughly a third, a half or two-thirds? After attempting this first time, hide the Species rank and set the ‘Sort by Selection’ tick-box to ‘ticked’. What is your guess at the proportion now?
- Which two specimens of Newman’s 10/10/2000 revision are present under “`Globba section Globba L.`” in the 3/10/2000 revision?

Sibling selections

Reset the ‘Sort by Selection’ option box to empty.
Make the Species rank visible again.
Clear the previous coloured taxa selections.
Set the ‘Show siblings on next’ option box to ‘ticked’.
Select the first listing of “`K.s.n. Micholitz`” from the list.

- Which classification / revision does “`K.s.n. Micholitz`” not occur in? Hide this classification.
- Can you see a pattern to the way “`K.s.n. Micholitz`” is grouped with other specimens across the revisions, or a lack of one?

Hide the Species rank.

- Does this make a pattern appear in any of the revisions?

Brush Manipulations

Noticing a possible correlation emerging between Newman’s 3/10/2000 and 10/10/2000 revisions, you decide to investigate further by matching distribution of specimens species by species between the two classifications. You can do this by ‘brushing’ the classifications i.e. by moving the mouse pointer over a single taxon or a group of taxa, which will highlight the same taxa temporarily in other classifications.

Clear all previous coloured taxa selections.
Clear the sibling mode option box.
Make all ranks visible.
Hide all revisions except Newman 3/10/200 and Newman 10/10/2000.

- *Brush* all the groups of specimens in Newman 3/10/2000, one at a time. Find which groups *exactly* match groupings in Newman 10/10/2000.
- Re-examine groups in Newman 3/10/2000 for groups whose specimens, present or not in Newman 10/10/2000, are not split between two or more groups in Newman 10/10/2000.
- Now for each of the groups found in the previous two questions, brush the corresponding (containing) groups in Newman 10/10/2000. Judge which groups take specimens from other places in Newman 3/10/2000, or consist of Newman 3/10/2000 groups with additional new specimen names, or a mixture of the two.

Classification Manipulations

Highlight the taxa Newman 3/10/2000 first classified by selecting (“clicking” on) the taxa at the genus rank (“`Globba L.`” or “`mn-genGlobba`”) of *all* the preceding classifications and revisions, and then inverting the colours in the display.

- How many specimens in Newman 3/10/2000 are new additions to the revision?
- What are their names?
- How are the taxa first used by Newman 3/10/2000 highlighted in the list?

Select the last occurrence of ‘M. F. Newman in M. F. Newman…’ from the history bar at the bottom of the screen.
Select the “`Globba L.`” genus rank in Newman 3/10/2000.
Invert the highlighted nodes.
This should now show all the unique names in Newman 10/10/2000.

- What specimens are unique to Newman 10/10/2000?

	General			Single					Group				Sibling			Brush			Classification			Manipul		Totals	**Average**	Prev?	Globba or Apiaceae?
U1	1	1	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1	1	10	**43.48**	TRUE	A
U2	1	0	0	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	0	1	1	11	**47.83**	FALSE	G
U3	1	1	1	1	1	1	1	0	0	1	0	0	1	0	0	1	1	0	1	1	1	1	1	16	**69.57**	TRUE	A
U4	1	1	0	1	1	1	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	1	13	**56.52**	FALSE	G
U5	1	1	0	1	1	1	1	0	1	1	0	1	1	0	0	1	1	0	1	0	1	1	1	16	**69.57**	FALSE	A
U6	1	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	20	**86.96**	TRUE	A
U7	1	1	0	1	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	18	**78.26**	FALSE	A
U8	0	0	0	0	1	0	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0	1	1	7	**30.43**	FALSE	G
U9	1	0	0	1	1	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0	0	1	1	12	**52.17**	FALSE	G
U10	0	0	0	1	1	1	0	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	11	**47.83**	FALSE	G
U11	1	1	0	1	1	1	1	0	0	1	1	1	1	1	0	1	0	0	1	0	1	1	1	16	**69.57**	FALSE	A
U12	1	1	0	1	1	0	1	0	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	18	**78.26**	FALSE	A
U13	0	1	0	1	1	0	1	0	1	1	0	1	1	0	0	1	1	1	0	0	0	0	0	11	**47.83**	FALSE	G
U14	1	1	0	1	1	0	1	1	1	0	1	1	1	1	0	1	0	0	1	0	1	1	1	16	**69.57**	FALSE	A
U15	0	0	0	1	1	0	0	0	1	0	0	0	1	1	0	1	0	0	0	0	0	1	1	8	**34.78**	FALSE	G
U16	1	1	1	1	1	0	1	0	1	1	0	1	1	0	0	1	1	0	0	0	1	1	1	15	**65.22**	FALSE	A
U17	0	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	1	1	0	0	16	**69.57**	TRUE	G
U18	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	20	**86.96**	TRUE	G
U19	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	1	1	0	1	1	18	**78.26**	TRUE	G
Q Total	14	12	5	18	18	11	13	9	14	14	11	16	16	12	4	15	8	5	10	5	10	16	16	272	**62.24**	16.9045799	
Section Sums			62					138				110			64			56					114				
Prev Avg																Mean for previous									**72.46**		
New Avg																Mean for new									**57.53**		
Apiaceae Avg																Mean for Apiaceae									**70.05**		
Globba Avg																Mean for Globba									**55.22**		
New with Apiaceae Avg																Mean for new Apiaceae users									**71.74**		
New with Globba Avg																Mean for new Globba users									**45.34**		
Prev with Apiaceae Avg																Mean for previous Apiaceae users (or									**66.67**		
Prev with Globba Avg																Mean for previous Globba users (only									**78.26**		

System Usability Scale

Strongly disagree Strongly agree

1. I think that I would like to use this system frequently	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
2. I found the system unnecessarily complex	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
3. I thought the system was easy to use	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
4. I think that I would need the support of a technical person to be able to use this system	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
5. I found the various functions in this system were well integrated	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
6. I thought there was too much inconsistency in this system	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
7. I would imagine that most people would learn to use this system very quickly	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
8. I found the system very cumbersome to use	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
9. I felt very confident using the system	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
10. I needed to learn a lot of things before I could get going with this system	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5

SUS scale results	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	**User Total**	**User % Total**	Prev User	Apiaceae or Glc
U1	4	3	4	4	3	3	4	4	4	4	**37**	**92.5**	TRUE	A
U2	1	2	1	1	2	3	2	2	2	1	**17**	**42.5**	FALSE	G
U3	2	4	3	4	3	4	3	4	3	4	**34**	**85**	TRUE	A
U4	2	3	3	1	3	4	3	4	1	1	**25**	**62.5**	FALSE	G
U5	2	2	2	1	3	3	3	3	1	2	**22**	**55**	FALSE	A
U6	3	3	3	3	2	2	3	3	3	2	**27**	**67.5**	TRUE	A
U7	2	3	3	0	3	3	2	3	2	3	**24**	**60**	FALSE	A
U8	2	1	2	3	3	2	4	2	1	1	**21**	**52.5**	FALSE	G
U9											**0**	**0**	FALSE	G
U10	2	1	1	1	4	4	2	3	1	1	**20**	**50**	FALSE	G
U11	2	3	3	3	2	3	3	3	2	2	**26**	**65**	FALSE	A
U12	4	4	4	4	4	4	4	4	3	3	**38**	**95**	FALSE	A
U13											**0**	**0**	FALSE	G
U14	0	2	1	3	2	2	3	2	1	1	**17**	**42.5**	FALSE	A
U15											**0**	**0**	FALSE	G
U16	3	2	2	3	1	2	3	2	2	1	**21**	**52.5**	FALSE	A
U17	3	3	2	1	4	3	4	3	2	3	**28**	**70**	TRUE	G
U18	2	4	4	4	4	4	2	4	3	4	**35**	**87.5**	TRUE	G
U19	3	4	3	4	3	4	2	4	4	4	**35**	**87.5**	TRUE	G
Q Total	**37**	**44**	**41**	**40**	**46**	**50**	**47**	**50**	**35**	**37**	**427**			
Mean	2.313	2.75	2.563	2.5	2.875	3.125	2.938	3.125	2.188	2.313	**26.6875**	**66.71875**		
Median	2	3	3	3	3	3	3	3	2	2	**25.5**			
Prev Median	3	3.5	3	4	3	3.5	3	4	3	4	**34.5**			
New Median	2	2	2	3	3	3	3	3	1.5	1	**21.5**			
Apiaceae Median	2.5	3	3	3	3	3	3	3	2	2	**26**			
Globba Median	2	3	2	1	3	4	2	3	2	1	**25**			
New Apiaceae Medi	2	2.5	2.5	3	2.5	3	3	3	2	2	**23**			
New Globba Mediar	2	1.5	1.5	1	3	3.5	2.5	2.5	1	1	**20.5**			