EUROGRAPHICS - IEEE VGTC Symposium on Visualization (2005)
K. W. Brodlie, D. J. Duke, K. I. Joy (Editors)

ArcTrees: Visualizing Relations in Hierarchical Data

Petra Neumann[1], Stefan Schlechtweg[2], and Sheelagh Carpendale[1]

[1] Department of Computer Science, University of Calgary, 2500 University Drive NW, Calgary, AB, Canada T2N 1N4
[2] Department of Simulation and Graphics, Otto-von-Guericke University of Magdeburg, Universitätsplatz 2, D-39106 Magdeburg, Germany

Abstract

In this paper we present, ARCTREES*, a novel way of visualizing hierarchical and non-hierarchical relations within one interactive visualization. Such a visualization is challenging because it must display hierarchical information in a way that the user can keep his or her mental map of the data set and include relational information without causing misinterpretation. We propose a hierarchical view derived from traditional Treemaps and augment this view with an arc diagram to depict relations. In addition, we present interaction methods that allow the exploration of the data set using Focus+Context techniques for navigation. The development was motivated by a need for understanding relations in structured documents but it is also useful in many other application domains such as project management and calendars.*

Categories and Subject Descriptors (according to ACM CCS): H.5.2 [Information Interfaces and Presentation]: User Interfaces – Graphical user interfaces (GUI) I.3.6 [Computer Graphics]: Methodology and Techniques – Interaction techniques

Keywords: Information visualization, tree structures, relations, arc diagrams, user interface, Focus+Context

1. Introduction

Many types of data are either naturally hierarchical or amenable to a recursive grouping that can establish a hierarchy. However, the relations indicated by this natural or imposed hierarchical or tree structure are usually not the only types of relations of interest. While there are an increasing number of methods for drawing trees, almost no methods have been designed specifically to display the additional relations and reveal their connections to the hierarchy without resorting to general graph layout strategies. We present ARCTREES, a visualization method that integrates the display of additional relations with the hierarchical structure.

To illustrate this point, we mention a few of the many possible examples of information with a primarily hierarchical structure that also incorporate additional relations. For example, consider a text book. It has an explicit hierarchical structure composed of the book, its chapters, their sections, and the paragraphs and images that comprise those sections. There are also additional relations of interest. These include indicated forward and backward references, the sequence of readings for a course of instruction, and exactly which previous sections in the text are needed to understand a particular passage (cf. Figure 1). Another example can be taken from examining sports tournament data. During elimination rounds players advance through initial games and play-off rounds to quarter and semi finals with eventually the winner ending at the root of the hierarchy. Here, additional relations might include tracing players of a particular nationality, or discovering which players are playing at which facilities. As a third example, we may look at calendars. With a common display of a calendar it can be difficult to discover such things as start and end dates of recurring events or exceptions within an otherwise regularly occurring event.

ARCTREES can be used for any general graph by displaying a spanning tree as the hierarchical structure and the remaining edges as additional relations. However, it has been primarily designed as an informative interactive tool for viewing digital media, as, for example, digital or on-line documents. ARCTREES were developed to provide insight about relations within data through visual and interactive means. Therefore, the general problem is to combine two very different kinds of relational information in one visualization. These are the hierarchical parent-child relations, and vari-

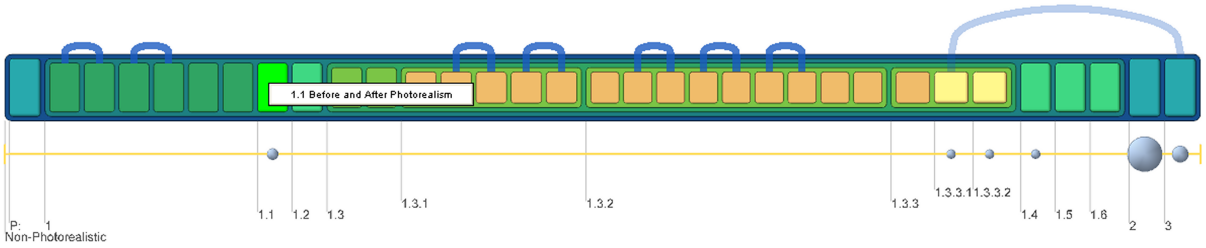

Figure 1: *An* ARCTREES *visualization of a scientific book. Arcs indicate references between images and paragraphs in the book. The circular glyphs indicate currently not visible relations.*

ous other kinds of relations that may provide links between nodes on different levels of the hierarchy. The goal of such a visualization is to provide a visual tool that enables a better understanding of the structure of the data and that supports interactive exploration of the data. As such it is important to keep the following constraints in mind:

- use as little screen space as possible because the majority of the space will be needed for the media itself, e.g., the document,
- reveal both the hierarchy and the additional relations,
- make explicit how the additional relations are linked to the hierarchy, and
- include navigational tools and methods that can reveal the navigation actions taken within the visualization and provide methods for reversing these actions.

ARCTREES address three main points: the visualization of the hierarchical structure of the data, the visualization of additional relations, and interaction techniques that allow the exploration of the data.

The remainder of the paper is organized as follows. After a discussion of related research in Section 2 we explain the three main issues of our approach: the visualization of the hierarchical structure in Section 3.1, the visualization of additional relations in Section 3.2, and interaction techniques in Section 4. In Section 5 we illustrate the use of ARCTREES with practical examples and conclude the paper in Section 6.

2. Related Work

Visualizing relations between data items is an essential information visualization task. Relations can represent an inherent attribute of the data such as a linear structure of time or another metric, or a hierarchical parent-child relation. Other types of relations might be given through additional parameters or attributes of data items like their size, date, or type. Typical encoding mechanisms for relations are color, shape, orientation, proximity, position, and connection. Position is often used to encode relations in the visualization of linearly structured data sets. Chronological and line-oriented textual data items are usually presented in a linear order implying before and after-relations. If data items are related in some other way they are often encoded with similar color, shape, or texture. A visualization of linear data that breaks away from this pattern is Arc Diagrams, a recent visualization technique for repetition patterns in strings [Wat02]. Direct connections with arcs visualize relations between repeating parts in the string. This visualization has inspired Thread Arcs [Ker03], a technique that shows relations in email conversations that have a direct connection.

The visualization of relations through direct connections with lines or arrows, for example, is a well-known technique. Given a set of data items and a set of relations between these items, a common approach is to use a node-link diagram in the form of a graph. Here the items are represented as nodes and the relations become the links or edges between the nodes. There are many possible graph drawing techniques yielding a variety of different node link diagrams (cf. [HMM00, BETT99]). The parent-child relations of hierarchical data have been depicted with connections in node link diagrams (cf. [HMM00]), with containment in Treemaps [Shn92], and with position and proximity in a variety of configurations including Sunburst [SZ00], Info Slices [AH98], InterRing [JORP01], or Icicle Plots [KL83]. While the latter two kinds of tree visualizations are superior to the traditional node-link diagram in the usage of screen-real-estate, it is typically more difficult to understand their hierarchical relations. However, Barlow and Neville have shown in a study [BN01] that Icicle Plots compared favorably to other visualization techniques with respect to communicating topology and comparison of node size. Furthermore, Icicle Plots of large but not very deep hierarchies result in layouts with a rational use of screen space.

Visualizations specifically tailored to the combination of both hierarchical and non-hierarchical relations in one view are rare. Similarity relations between nodes that are related to other nodes are sometimes encoded through similar color, shape, or texture. Fekete et al. [FWD*03] presented a technique where a Treemap provides the hierarchy visualization and additional relations were overlaid as Bézier arcs connecting two Treemap regions. While this visualization does offer an example that combines a hierarchy with additional relations, it suffers from many edge crossings which makes it

hard to read and uses a considerable amount of screen space. Balzer et al. recognize this problem within their visualization of large software systems [BNDL04]. They state that very unclear representations would be the result of representing multiple relations as direct connections in 2D. Therefore, they suggest building a three-dimensional visualization which can be explored interactively by the user. Since the view parameters can be changed freely, an exploration of the relations is possible by selecting parameters that keep occlusion and overlapping at a minimum.

3. The ARCTREES Visualization

The purpose of ARCTREES is to graphically represent hierarchical data together with additional relations in order to trigger discoveries necessary for insight or decision making. Therefore, the hierarchical structure must be clear and the explicit details of the external relations apparent.

3.1. Visualizing the Hierarchy

Since ARCTREES are intended to be used as a modular components of an information display such as adjacent to a related text display, minimum and consistent use of screen space is important. We chose to represent the hierarchical structure via containment since this offers effective use of screen space that does not expand during interaction. Also, to honor the fact that the leaf nodes of text information often have significant linear ordering, such as chapters in a book, we combined the ideas of Treemaps [Shn92] and Icicle Plots [KL83] to develop a "one-dimensional Treemap". Each node n is represented by a rectangle on the screen. Leaf nodes are aligned linearly and sequentially. The width of the root node is set to the available display space and the height of the root rectangle is influenced by the depth of the tree. The height is calculated such that the rectangles for child nodes are contained within their parent's rectangle. Some offset is reserved for visual clarity and interaction while ensuring that the deepest leaf node is still visible. Within a level the width of a node is determined by a node weight $w(n)$ while the height depends on the level of n.

The hierarchy layout algorithm partitions the rectangular display space for the root into required parts according to the number of root's children. For each child, its horizontal display space is then again divided according to the number of children at the next level. This algorithm establishes an ordered tree layout and provides a spatially constrained display. The visual metaphor involves viewing a stack of nodes, similar to an upside-down Icicle Plot, from the top (Figure 2).

Node weights are influenced by a given node metric $f(n)$. A node metric measures or quantifies certain abstract features associated with a node. Node weights are used to determine the display size of a node in relation to its siblings and ancestor nodes. In Figure 3, several examples are given

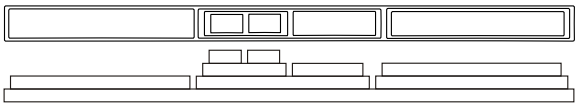

Figure 2: ARCTREES *layout (top) and metaphor (bottom).*

including the most simple metric that assigns the same constant value to each node so that on each level a uniform subdivision is achieved. More meaningful structural metrics calculate the size of a rectangle according to the number of direct children or the size of the subtree rooted at a particular node. We can also work with semantic metrics where the size depends on the data contents of the node or some value provided via interaction (degree of interest). Node weights can be interactively adjusted (see Section 4).

Color coding is used to portray structural information and highlight certain aspects of the visualized hierarchy. Figure 4 shows two examples. To allow for a better understanding of the hierarchy itself, a color scale is used to assign equal colors to nodes on an equal level in the hierarchy (top image in Figure 4). The bottom image in Figure 4 shows a semantic coding where the leaf nodes are colored according to their type. The group nodes on higher levels of the hierarchy are alternately colored white and gray so that the level information becomes immediately apparent.

3.2. Visualizing Relations

Relations are given as pairs of nodes and each relation may also be defined with a type parameter so that different kinds of relations can be visualized. Inspired by Arc Diagrams (cf. [Wat02, Ker03]), for each relation we draw an arc between the horizontal centers of the two nodes. Each arc is drawn above the tree visualization with its two endpoints extending down to the connected nodes using straight lines. Using semicircles as arcs results in a visualization with a sub-optimal use of screen space as can be seen in Figure 5 (top). For instance, a semi-circle relation spanning the complete length of the hierarchy would result in an arc height of half the length of the rectangle. Therefore, we use Chaikin's Algorithm [Cha74], to draw curves which can be controlled in height as can be seen in Figure 5 (bottom). Drawing half-ellipses would be another alternative but the resulting curves are less flexible in shape.

Additional information can be encoded using the opacity and width of the arcs. All arcs are drawn somewhat transparent to make the crossings clear and understandable (see Figure 6). An even more decreased opacity encodes connections between collapsed nodes (see Section 4). The width of an arc is determined by a metric that specifically calculates a weight value for each relation. Similarly to a node metric the relational metric also depends on structural or semantic information. Furthermore, if more than one relation connects

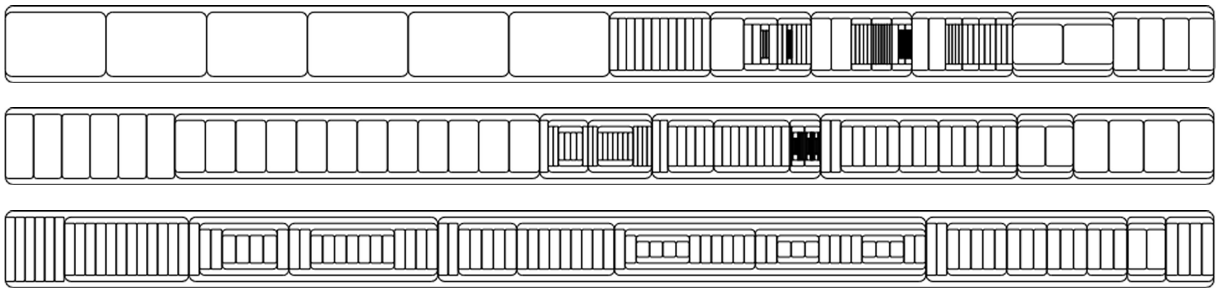

Figure 3: *Different tree layouts depending on the node metric. All leaf nodes having the same width (top), node width depending on the number of direct children (middle), node width depending on the size of the subtree (bottom)*

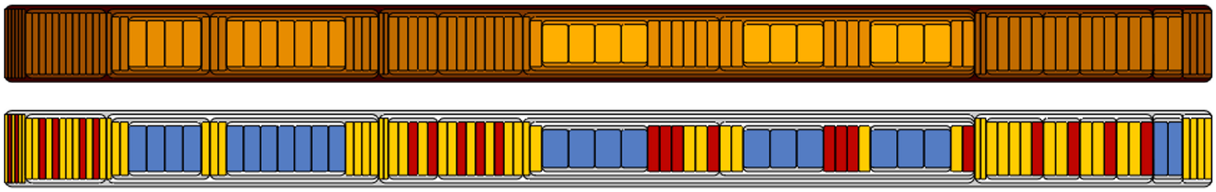

Figure 4: *Color coding according to the hierarchy levels (top) and based on semantic information (bottom). Here, an XML file is visualized using different colors for different tag types in the leaf nodes. Inner nodes all represent the same kind of grouping tag so that level information can be given by alternating dark and bright rectangles.*

two nodes, only one arc is drawn and the weights of all relations are added to calculate the arc's width. This resembles the approach taken by Balzer et al. in [BNDL04], where the width of an arc represents the number of relations that are present. Color could also be used to encode additional information. However, since the choice of color needs to be coordinated with the colors that are in use within the hierarchy visualization, currently color is not being used as an encoding technique for additional information about relations.

Figure 6 shows an example that uses this encoding scheme. The transparent arcs on the right hand side connect a collapsed node (far right) with other nodes. The different widths of the more opaque arcs show different weight values that have been computed from the number of relations that are present between the particular nodes.

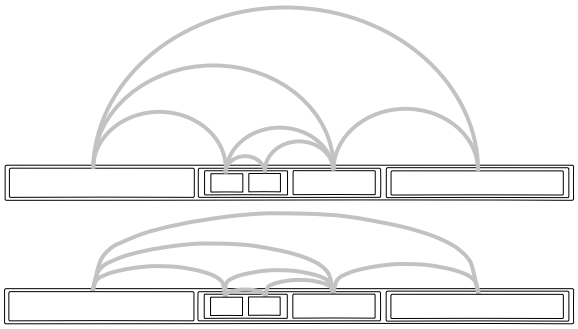

Figure 5: *Drawing circular arcs (top) or Chaikin curves (bottom) to visualize relations.*

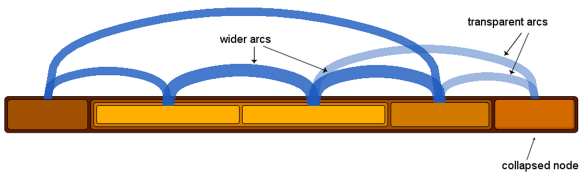

Figure 6: *Encoding techniques used for relations.*

4. Interaction

To enable the exploration of large hierarchical data structures, interaction techniques are needed to augment the spatial layout. Appropriate interaction tools are criticial because the number of leaf nodes that can be displayed with the proposed tree layout is bounded by the display resolution. As with all layouts, as the number of nodes gets larger the visual clutter increases and individual nodes may get too small to be discernible (Figure 4). To address this problem we provide a variety of interaction techniques including zoom and filter techniques and Focus+Context techniques for both the hierarchy and the relations. Also, a Focus+Context navigation can be combined with zoom and filter operations that depend on user-specified nodes of interest. This allows for larger trees and relations to be explored in detail at subtree

resolution while providing a contextual overview for the remainder of the tree.

4.1. Zoom and Filter Interactions

We implemented a simple zoom and filter mechanism by expanding and collapsing subtrees based on user interaction, as can be seen in Figure 7. A node that can be further expanded is represented as a button with shading to indicate that it can be pushed. A fully expanded or leaf node is drawn as flat. A subtree that has been collapsed will be represented as a button indicating that this operation can be reversed.

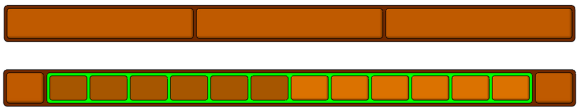

Figure 7: *A tree containing three subtrees (top). All three nodes are collapsed. After interaction, the middle subtree is expanded one level (bottom).*

4.2. Focus+Context: Interactions with the Hierarchy

Subtrees may also be expanded or collapsed according to their assigned degree of interest. The degree of interest values for each node n are computed based on

1. the distance $d(n,r)$ of the node n from the root node r as an "a priori importance" according to Furnas' terminology (cf. [Fur86])
2. the distance $d(n,f)$ of the node n to the focus node f as the "a posteriori importance"

Both are combined as $\mathrm{DOI}(n) = -(d(n,f)+d(n,r))$. Note that the distance in both the above cases is defined as the length of the path between the two nodes in question. After having selected a node as focus node, thresholding on the DOI determines those nodes that need to be collapsed (i.e., the context) and those nodes that need to be expanded (i.e., the focus area). The DOI values can also be used to obtain a node metric which then determines the size of the rectangles to be drawn (cf. Section 3.1), thus, providing a full Focus+Context technique. Figure 8 shows the same tree as in Figure 3, this time using a DOI based metric.

4.3. Focus+Context interactions for relations

Providing a Focus+Context visualization for the relations requires additional care. Three issues are of importance:

1. the arc layout may have to change during interaction,
2. relations between two nodes should contribute to the DOI value, and
3. relations might be used to initiate the interaction.

When interacting with the tree visualization, nodes that are connected via relations may become invisible (when a parent node is collapsed) or nodes may become visible (when a parent node is expanded) that need to be directly connected via relations. To avoid information loss during interaction a visualization is needed that encodes relations connecting to hidden child nodes of a collapsed node. When a node connected by a relation becomes hidden during interaction, its connecting arc is drawn to the collapsed parent. If at least one of the nodes directly connected by a relation is hidden inside a collapsed node, the arc representing this relation is drawn using a lower degree of opacity (see Figure 6). A relayout is also necessary when an *indirect connection* (leading to a collapsed node) becomes a *direct connection* (leading to visible nodes). In this case, arcs need to be connected to the respective visible nodes and the opacity needs to be increased. The case in which a relation is completely hidden inside a collapsed subtree is indicated by a circular glyph below the collapsed node to indicate that hidden relations exist (cf. Figure 9).

To explore the data based on relational information, the relation arcs may be used directly for interaction. Selecting a relation will cause an increase in the DOI value of the connected nodes, i.e., both connected nodes are regarded as focus nodes. Therefore, they are (a) expanded and (b) enlarged in size. To achieve this, external relations are considered when calculating the DOI. An initial DOI is calculated as described in Section 4.2. Then, the following four steps adjust these DOI values to consider relational connections:

1. Collect all relations currently connected to a focus node directly or indirectly.
2. Follow these relations recursively and adjust DOI values if needed.
3. If a new DOI value was set for a node adjust its ancestors if needed.
4. Repeat the previous steps for all nodes in the focus area.

This procedure enlarges the DOI values for all nodes that are connected via external relations directly or indirectly to the focus node. Due to the recursive application, the distance along these relations is automatically taken into account. The recursion stops when reaching a node without further relations or when returning to the focus node in question (circular relational references are possible). The adjustment of the ancestor nodes is needed since no parent node can have a lower DOI than a child node (cf. [Fur86]).

Figure 9 shows an example for layout changes when interacting with relations. The top image shows the start situation. The user interacts with the left arc that connects a visible and a collapsed node. After interaction the relations need to be updated since the collapsed node has expanded due to DOI changes (it becomes part of the focus since it is connected to the selected relation). Note that the selected relation splits up into two relations to different child nodes of the formerly collapsed node (highlighted in green). Other relations appear

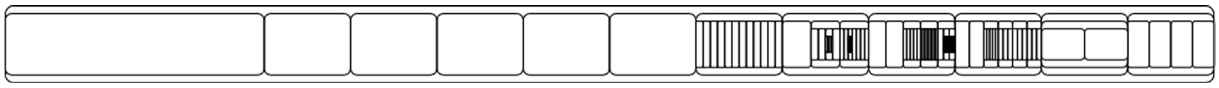

Figure 8: *DOI based node metric. The focus is on the first node on the left. Note that no Focus+Context techniques have been applied here, this is just a visualization of the node metric.*

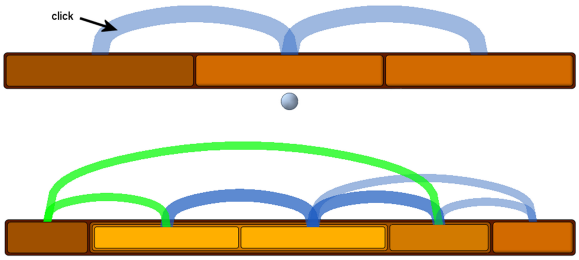

Figure 9: *Direct interaction with relations.*

within the middle node as well as from the middle one to the right node. Since the middle node is expanded, the right relation arc also needs to be split to display the now visible relations to the child nodes. All changes finally result in the bottom image.

Hidden relations are encoded as circular glyphs below the tree visualization. The diameter of the circular glyph stands for the number of hidden relations it encodes. Glyphs are drawn with shading to provide push-able affordance indicating interaction possibilities. Interaction with the circular glyph buttons expands the respective tree nodes until at least one relation becomes visible. In addition, our system provides tooltips to display detail on demand for relations and nodes as can be seen in Figure 10.

5. Examples

In this section we use an example of a scientific text to illustrate the intended use of ARCTREES. Scientific books are seldomly read from cover to cover, instead, they are consulted to learn about a specific topic. Reading a chapter that deals with this particular topic may require background information from other chapters. Assume an electronic book where such information in the form of relations named "requires knowledge of" or "is prerequisite for" is present. As an example we have chosen a book on Computer Graphics (cf. [FvDFH90]) and modeled the structure and the above mentioned relations. Figure 11(a) shows an overview of the 21 chapters that build this book. (Note that we have not modeled all chapters, therefore, only a few of them are expandable.)

Now, the reader looks up shadow algorithms and the table of contents tells him or her that Section 16.4 deals with this topic. Having no previous knowledge about shadow algorithms he or she might not be familiar with all the background information needed to understand the algorithms covered in this section. Figure 11(b) shows additional information describing "requires knowledge of" relations for Section 16.4. The circular button below Chapter 16 tells the reader that some sections in the same chapter are required reading. The reader will also notice a relatively wide arc between Chapter 15 and 16 which represents more than one connection between these chapters. He or she might then first request additional information on Chapter 15 and find out through the tooltip that Chapter 15 is concerned with visible surface determination which seems to be an important aspect of shadow algorithms. A click on the arc connecting to Chapter 15 reveals four relations between both chapters. In a Focus+Context view eight focus nodes are placed in the visualization. This leads to the eight connected nodes being displayed larger in comparison to their siblings giving a clearer picture of the connected nodes in Figure 11(c) Through detail on demand the reader will notice that the knowledge of four algorithms is needed for implementing shadows: Appel's Algorithm, a z-Buffer Algorithm, Scan-Line Algorithms, and the Weiler-Atherton Algorithm. Each of these algorithms is connected to one different shadow implementation covered in Section 16.4. Based on this information the reader can now decide to concentrate on a shadow algorithm for which he or she already has the required background, to concentrate on a specific one, or to go through all algorithms after learning the required background information.

6. Discussion, Directions and Conclusions

In this paper we have presented ARCTREES, an interactive visualization tool that integrates hierarchical and relational information in one view. Since ARCTREES were designed to be used as an accompanying tool with another information display, the space usage has been successfully limited to a narrow strip approximately one tenth the size of a normal desktop display. ARCTREES:

- require only a small amount of screen space,
- integrate additional relations with a hierarchical representation without obscuring the structure of the hierarchy,
- use 3D button affordances to indicate available interactions and hidden information,
- provide Focus+Context interactions that consider the importance of the structure in the hierarchy and the structure in the relations, and

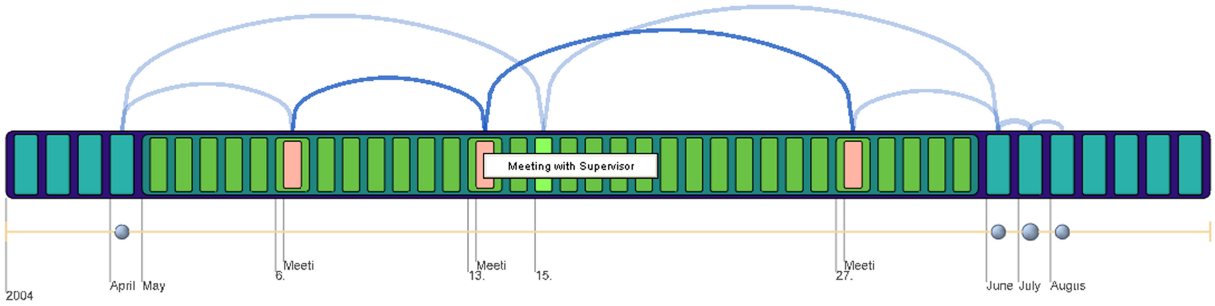

Figure 10: *The complete interactive visualization with all additional tools present. This example shows the exploration of a person's calendar. Relations represent recurring events.*

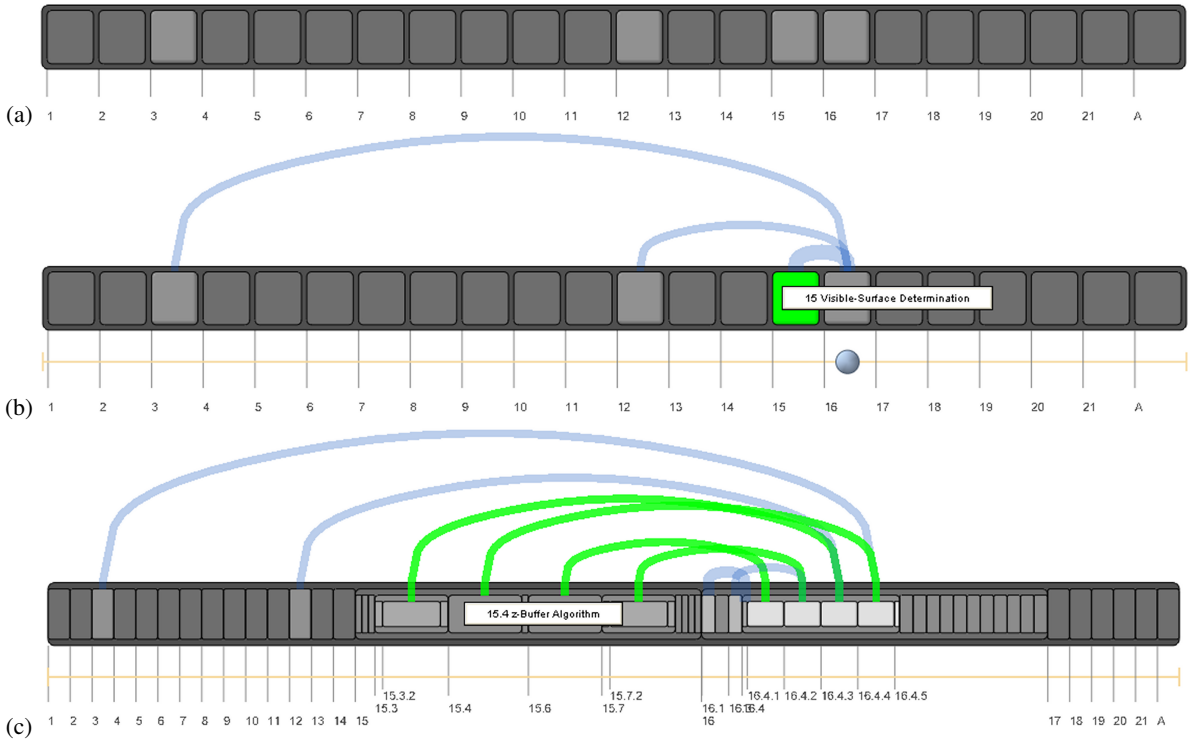

Figure 11: *Using* ARCTREES *to explore a scientific book. (a) shows the general structure of the book being divided into 21 chapters and an appendix. (b) shows that the topic that is treated in Chapter 16 builds on information from Chapters 3, 12, and 15. Selecting the wide arc finally results in (c), which allows a detailed exploration of the connections between both chapters.*

- augment these interactions with users requested zoom and filter, labels, tooltips and additional text.

Both classes of relations, i.e., hierarchical and non-hierarchical, are often encountered in information spaces. Electronic documents are only a prominent example. Here, the document structure defines the hierarchical relations while a wealth of other relations may connect parts of the text. For navigation in and exploration of such an information space, the understandable depiction of both types of relations is important. ARCTREES provide an explicit visualization of the hierarchy and how the additional relations are connected to it.

The effectiveness of ARCTREES as a visualization tool depends on several aspects. One is whether the structural relations are understandable. Our goal was to provide a visualization that used much less space than Treemaps and maintained the same degree of readability. Initial comparative pilot studies suggest that the layout of ARCTREES can

be read significantly faster at equal error rate when compared in several tasks to the traditional slice-and-dice Treemap layout while using approximately 1/8 of the space of Treemaps for large trees.

We are also interested in enriching the visualization of relations. For instance, color schemes to indicate relation type could be integrated with the color schemes in use for the hierarchy. Also some relations are directional. Visual indications of direction could be included. One possible method would be to make arcs become gradually darker in the direction of the relation.

Significant layout changes occur when expanding and contracting nodes—especially if many hidden relations exist. While animation of these changes is important, it is possible that other techniques might be developed that better communicate the changes to the user.

ARCTREES might, for example, be used as a modular component of an electronic reading environment. Given the space constraints that were observed in the design of this visualization, it is well imaginable that such a visualization might be part of, for example, Acrobat Reader as the thumbnail overview is today.

Acknowledgments

We gratefully thank our funding providers: Natural Sciences and Engineering Research Council (NSERC), Alberta's Informatics Circle of Research Excellence (iCORE), and Alberta Ingenuity. We would also like to thank the researchers from the Department of Simulation and Graphics at the University of Magdeburg and the Interactions Lab at the University of Calgary for their insightful comments on this work.

References

[AH98] ANDREWS K., HEIDEGGER H.: Information Slices: Visualising and Exploring Large Hierarchies using Cascading, Semi-Circular Discs. In *Proc. InfoVis '98, Late Breaking Hot Topics* (1998), IEEE Press, pp. 9–12. 2

[BETT99] BATTISTA G. D., EADES P., TAMASSIA R., TOLLIS I. G.: *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, Upper Saddle River, NJ , USA, 1999. 2

[BN01] BARLOW T., NEVILLE P.: A Comparison of 2-D Visualizations of Hierarchies. In *Proc. InfoVis'01* (2001), IEEE Press, pp. 131–138. 2

[BNDL04] BALZER M., NOACK A., DEUSSEN O., LEWERENTZ C.: Software Landscapes: Visualizing the Structure of Large Software Systems. In *Data Visualization 2004. Eurographics/IEEE TCVG Visualization Symposium Proceedings* (2004), Eurographics Association, pp. 261–266. 3

[Cha74] CHAIKIN G. M.: An Algorithm for High Speed Curve Generation. *Computer Graphics and Image Processing 3*, 12 (1974), 346–349. 3

[Fur86] FURNAS G. W.: Generalized Fisheye Views. In *Proceedings of CHI'86* (New York, 1986), ACM SIGCHI, pp. 16–23. 5

[FvDFH90] FOLEY J. D., VAN DAM A., FEINER S. K., HUGHES J. F.: *Computer Graphics. Principle and Practice*, 2. ed. Addison Wesley Publishing Company, Reading, MA, 1990. 6

[FWD*03] FEKETE J.-D., WANG D., DANG N., ARIS A., PLAISANT C.: Overlaying Graph Links on Treemaps. In *Proc. InfoVis'03, Poster Compendium* (Aug. 2003), IEEE Press, pp. 82–83. 2

[HMM00] HERMAN I., MELANÇON G., MARSHALL M. S.: Graph Visualization and Navigation in Information Visualization: A Survey. *IEEE Transactions on Visualization and Computer Graphics 6*, 1 (2000), 24–43. 2

[JORP01] J.YANG, O.WARD M., RUNDENSTEINER E. A., PATRO A.: InterRing: A Visual Interface for Navigating and Manipulating Hierarchies. In *Proc. InfoVis'01* (2001), IEEE Press, pp. 16–30. 2

[Ker03] KERR B. J.: *THREAD ARCS: An Email Thread Visualization*. Tech. Rep. RC22850, IBM Research, Cambridge, MA, USA, 2003. 2, 3

[KL83] KRUSKAL J. B., LANDWEHR J. M.: Icicle Plots: Better Displays for Hierarchical Clustering. *The American Statistician 37*, 2 (May 1983), 162–168. 2, 3

[Shn92] SHNEIDERMAN B.: Tree Visualization With Treemaps: a 2-d Space-Filling Approach. *ACM Transactions on Graphics 11*, 1 (1992), 92–99. 2, 3

[SZ00] STASKO J. T., ZHANG E.: Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations. In *Proc. InfoVis'00* (Oct. 2000), IEEE Press, pp. 57–65. 2

[Wat02] WATTENBERG M.: Arc Diagrams: Visualizing Structure in Strings. In *Proc. InfoVis'02* (2002), IEEE Press, pp. 110–116. 2, 3

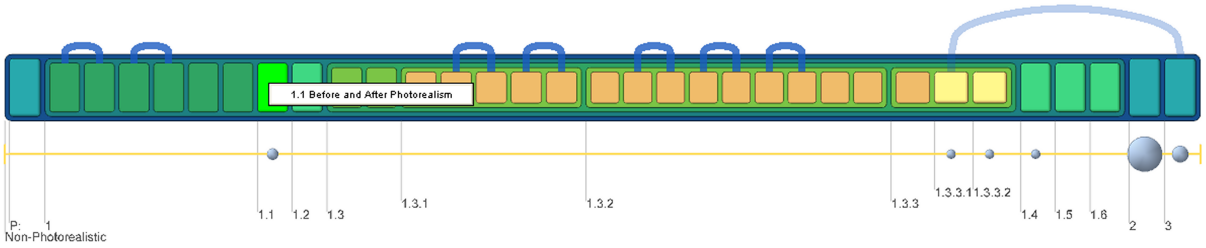

Figure 1: *A complete* ARCTREES *visualization.*

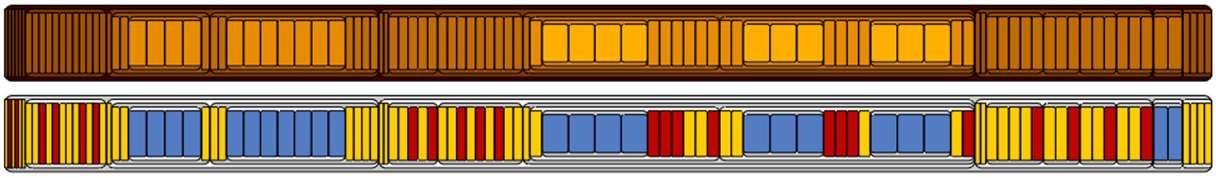

Figure 4: *Ordinal (top) and semantic (bottom) color coding.*

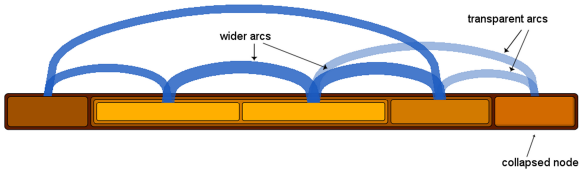

Figure 6: *Encoding techniques used for relations.*

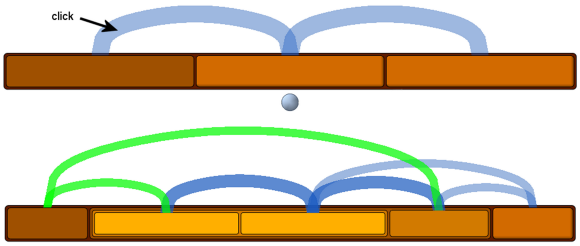

Figure 9: *Direct interaction with relations.*