Exploring Large Hyperdocuments: Fisheye Views of Nested Networks*

Emanuel G. Noik (noik@db.toronto.edu)
Computer Systems Research Institute
University of Toronto
Toronto, Canada M5S 1A1

Abstract

Browsing large hypertexts by following links can be difficult and confusing, especially if links span distant nodes. Often, a user would like to explore several regions of a network simultaneously, when studying the end points of one or more links, for example. Although this can be achieved by displaying each area of interest in a separate zoomed-in window, the union of such views is not always meaningful. In particular, valuable context showing the relationships between the views is lost.

By balancing local detail and global context, fisheye views display information at several levels of abstraction simultaneously. We have devised a novel technique for generating fisheye views of hierarchically nested structures with multiple variable strength focal points. In this paper we demonstrate its feasibility as a tool for exploring large nested hypertext networks.

Keywords: nested hypertext networks, navigation, overviews, fisheye views.

1 Introduction

Although hypertext and hypermedia systems have matured in terms of power and sophistication, many usability issues remain unresolved. One problem area that has been receiving growing attention is the scarcity of effective navigation techniques for exploring large or unfamiliar hyperdocuments [10, 15, 20, 26]. "Lost in hyperspace" is a common phenomenon cited by users who have felt disoriented when navigating through the information space. In response, numerous techniques including landmarks, timestamps, and footprints have been proposed [20]. These approaches were devised with the assumption that most users navigate hyperdocuments in order to locate some pertinent item of information. Unfortunately, these techniques are of limited benefit for users who set out to understand the overall structure of a large hyperdocument.

With respect to this latter goal, several techniques have been proposed, including multi-level overviews [20], nesting and link inheritance [6], and three-dimensional displays of networks [5]. Nonetheless, these have not been completely successful: overviews do not scale well; nesting provides a second level of organization that can aid users' understanding, but it too does not fare well when the information space grows in size; link-inheritance, although useful in decluttering a display by reducing the number of visible links, does not allow more information (greater number of nodes) to be displayed effectively; by harnessing the implicit effect of the geometry of point perspective, 3D displays provide a natural single focal point fisheye view of the information space, but this technique alone does not permit the simultaneous emphasis of two or more regions of a network.

The problem of visualizing large information spaces is not confined to the exploration of hypertext networks: the limitations of video display technology and the human visual system constrain the amount information that can be effectively shown and discerned on a computer console. To combat this problem, Furnas suggested an analogy to the wide angle or 'fisheye' lens: the generalized fisheye view [7]. By balancing local detail and global context, fisheye views (FEVs) display information at several levels of abstraction simultaneously.

We have devised a novel technique for generating FEVs of hierarchically nested structures, employing multiple variable-strength focal points. In this paper we describe how this technique is applied to the visualization of nested hypertext networks

*This paper appeared in modified form in *Hypertext '93: The 1993 ACM Conference on Hypertext and Hypermedia*, Seattle WA, Nov. 14–18, 1993.

[1, 6, 10, 29, 31, 32]. The paper proceeds as follows: Furnas' formulation of generalized FEVs is outlined; a sample hyperdocument is introduced and several overview visualizations created using existing techniques are demonstrated; our FEV approach is described; several exploration scenarios are suggested in order to demonstrate how our technique can be used to generate effective FEVs of the sample hyperdocument; the remainder comprises a few notes on implementation, directions for future work, and conclusions.

2 Fisheye views

Furnas' generalized FEV technique relied upon an abstract notion of a view *element* with an associated *a priori importance* (API), and a *distance* measure between any pair of elements. One element was selected to be the fisheye *focal point* (FP). A *degree of interest* (DOI) function computed a measure of the user's interest in each element; the DOI increased with API and decreased with distance to the FP. A FEV was obtained by displaying only elements whose DOI exceeded a threshold. The exact formulations of the DOI and distance functions were expected to vary with each application.

Recent applications of generalized FEVs can be loosely grouped into two categories: *filtering FEVs* and *distorting FEVs*.

Filtering FEV approaches use thresholding to constrain the display of information to relevant or interesting elements, and have remained more faithful to Furnas' original conception. For example, this approach was used to filter the display of aircraft maintenance data [18], to control the amount of output produced in response to boolean queries [8], and to determine the most "interesting" links in hypertext navigation [4]. Furnas described similar filtering FEVs of C programs and appointment calendars [7]. One graph FEV approach [5] described in the next section also belongs in this category.

Distorting FEV approaches, on the other hand, pursue the analogy to the fisheye lens further, but in doing so, they begin to depart from Furnas. The basic idea is to geometrically distort visualizations to obtain visual fisheye effects. Rather than filtering elements, distortion is used to alter elements' positions and sizes. Two graph FEV approaches [17, 25] described in the next section are of this type.

2.1 FEVs of graphs

Several approaches for obtaining FEVs of graphs have been documented [5, 17, 25].

Fairchild *et. al* described three filtering FEVs of the 3D networks of SemNet: clustering, three-dimensional point perspective, and sampling density [5]. In clustering, recursive subdivision of the 3D graph volume yielded an 8-ary tree with SemNet nodes partitioned among its leaves; tree distance (an approximation of euclidean distance) was used in the DOI function. The second strategy was based on the implicit effect of the geometry of point perspective in three dimensions, by which nearby points loom large and distant points appear small. Although not implemented, the third technique was motivated by the workings of the human retina which samples an image densely at its center of focus and in successively less detail for points further away. All three strategies were defined in terms of a single FP.

Sarkar and Brown described a distorting FEV of graphs in which a drawing of a graph was subjected to a distortion that altered the positions and sizes of nodes [25]. Although their solution could be implemented efficiently using special purpose graphics hardware, it was limited to non-nested graphs with a single FP.

Misue and Sugiyama described three distorting FEVs: fisheye, orthogonal fisheye, and biform [17]. These methods have been applied to drawings of compound digraphs [28] – a nested graph formalism. The first two method employed distortions similar to Sarkar and Brown's, while the third used uniform scaling. Although these methods supported multiple equal strength FPs, they could not handle variable strength FPs.

The main problem with all of these approaches is their reliance on a geometric interpretation of distance. Under this assumption, the resulting FEV depends completely on the original drawing of the graph: elements positioned near a FP receive a higher DOI rating. Such approaches are therefore only effective in cases where the abstract distance between nodes corresponds to their proximity in the drawing (*e.g.,* transportation networks where nodes are positioned using cartographic data). This basic flaw was also noted in previous work [5, 25].

The layout-independent FEV technique outlined in this paper overcomes this limitation.

2.2 Other related work

TreeMaps [13, 30] and *Cone Trees* [23] are two techniques for visualizing hierarchical structures that incorporate FEV concepts by using shape and size distortion and 3-D perspective respectively. Note that our FEV technique could be modified to calculate the sizes of the nested rectangles in TreeMaps. Although not strictly a distorting FEV, the *Perspective Wall*, used to display linear information [14], is a related visualization technique.

3 Nested hyperdocuments

3.1 Nested graphs and networks

Several nested graph and hypertext systems have been described in the literature; the following are some examples: Storyspace [1]; SemNet [5]; Interactive Graphical Documents (IGD) [6]; NoteCards [10]; state charts and higraphs [11]; the KJ method which employs compound digraphs [27, 28]; semantic knowledge networks [29]; and, Intermedia [32].

The NoteCards model of hypertext, for example, is built around four basic constructs:

- *Notecards.* A notecard is an electronic generalization of the 3x5 paper notecard. Each card may contain some multi-media artifact such as text, graphics, image, sound, video. Each card also has a title.
- *Links.* Links are used to interconnect individual cards into networks. Links are directed, and typed by a user-chosen label that describes the nature of the relationship.
- *Browsers.* A browser is a specialized notecard that contains a diagram of a network of notecards.
- *Fileboxes.* A filebox is a specialized notecard that is used to organize or categorize large collections of notecards.

Browsers are used to display the structure of the network, while fileboxes provide additional hierarchical structure to aid in coping with large networks.

3.2 A sample hyperdocument

The Canada-U.S. Free Trade Agreement (FTA) is a medium-sized document that has been converted into electronic format for viewing with the lector flexible text display tool [22]. The document has a well-defined hierarchical structure that comprises: *parts, chapters, articles, sections, annexes*, and 4 levels of *itemization*. Two possible lector visualizations are shown in Figures 1(a) and 1(b). The first example shows a portion of the formatted, "parts, chapters, and articles" filtered view, while the second example displays a portion of the complete formatted text. Note that the size of the document does not permit a coherent textual overview without resorting to extreme information filtering. Furthermore, textual overviews are very one-dimensional – a simple textual section heading does not indicate how much structure is contained within.

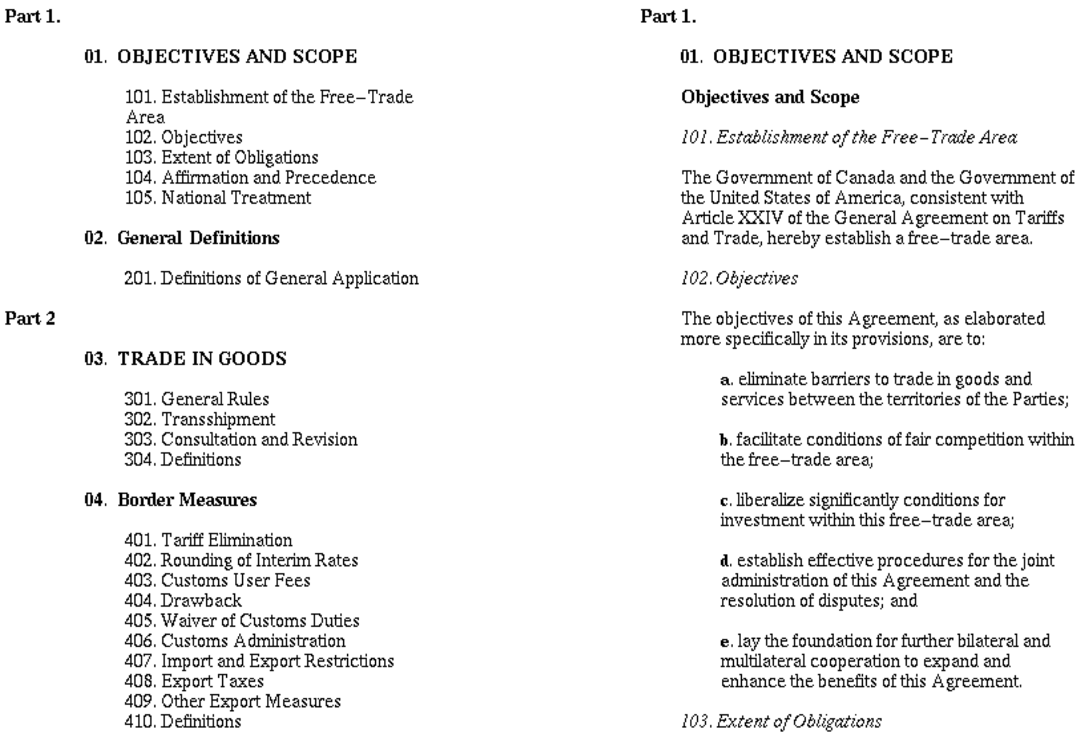
In an effort to produce more informative two-dimensional graphical overviews, we converted the lector marked-up text of the FTA into several types of graphs.

The first graph, shown in Figure 4, provides a flat (non-nested) overview of the document, and consists of 1680 nodes and 3852 links. The nodes of the graph represent the document headings. The labelled links of the graph are of three kinds: containment (*e.g.,* from a part to a contained chapter); sequencing (*i.e.,* the original linear ordering); and, semantic (*e.g.,* cross references). Note that the purpose of a graphical overview in a hypertext setting is to elucidate the structure of the information space and the user's location within it – to aid navigation and exploration. This technique was intended to complement text visualization tools – not to replace them. Viewing the text associated with the leaves of the overview graph, can be accomplished by integrating a tool like lector with a graphical graph browser, as done in the Hy+ system, for example [3].

Although the flat graph of Figure 4 is not overly large by present-day standards, it is clear that a simple one-level overview is totally ineffective, both as a navigation aid, and as a means of displaying the structure of the document. Since the FTA had a well-defined structure, we added nesting to the flat graph resulting in the graph shown in Figure 5. This graph consists of the original 1680 nodes, 247 "fileboxes", and 2180 links (sequencing and semantic only).

While the previous visualization is a substantial improvement over the original flat overview, it is still cluttered. To reduce this problem, we applied the *link inheritance* technique [6] to reduce the number of visible links. The resulting graph shown in Figure 6 contains 1512 visible links. A more extreme decluttering approach is available through information hiding: the contents of selected nodes can be hidden, and revealed as necessary. Figure 7 shows a graphical version of the "parts and chapters" filtered view. Unlike its textual counterpart (Fig. 1(a)), the graphical overview is more two-dimensional – the physical extent of a hidden region provides some indication of its size.

These examples provide convincing evidence that the combination of nesting and link inheritance can be used to generate more effective overviews of medium-sized hyperdocuments. Nevertheless, as the visualization of Figure 6 suggests, as the



(a) A "parts, chapters, and articles" view. (b) A full contents view

Figure 1: Two lector views of the FTA.

size of the information space grows, overviews require the addition of pan and zoom facilities to make it possible for users to read the node labels.

4 Layout-independent FEVs

We now describe our method for producing FEVs of hierarchically nested structures with multiple variable FPs. The crux of our approach is to generate FEVs by changing the size of each element in relation to its DOI before it is positioned by a layout algorithm, rather than by altering an existing layout. Since our technique does not filter or distort existing layouts, it does not properly belong to either of the filtering and distorting FEV categories.

Before presenting our generalized FEV formulation and the generic FP proximity algorithm, we give definitions of basic terms and give a sketch of some graph layout and related concepts.

4.1 Preliminaries

Graphs: A nested graph is $G = \langle G(V), G(A), G(C)\rangle$, where, $G(V)$ is a set of vertices, $G(A) \subseteq G(V) \times G(V)$ is a set of arcs, and $G(C) \subset G(V) \times G(V)$ is a set of containment arcs. Since we insist on hierarchical nesting, the nesting graph $G_n = \langle G_n(V) = G(V), G_n(A) = G(C), G_n(C) = \emptyset\rangle$ must be a forest of trees. A simple nested graph and its nesting forest are shown in Figure 2. For $v \in G_n(V)$, $anc(v)$ returns v's ancestors, $desc(v)$ v's descendants, $ch(v)$ v's children, and $anc(v) \cap desc(v) = \{v\}$.

Vertex geometry: For layout purposes, the geometry of a vertex is specified relative to the geometry of its parent in G_n. This makes it possible to resize all vertices in the subtree rooted at v, simply by changing the extents of v. The bounding box

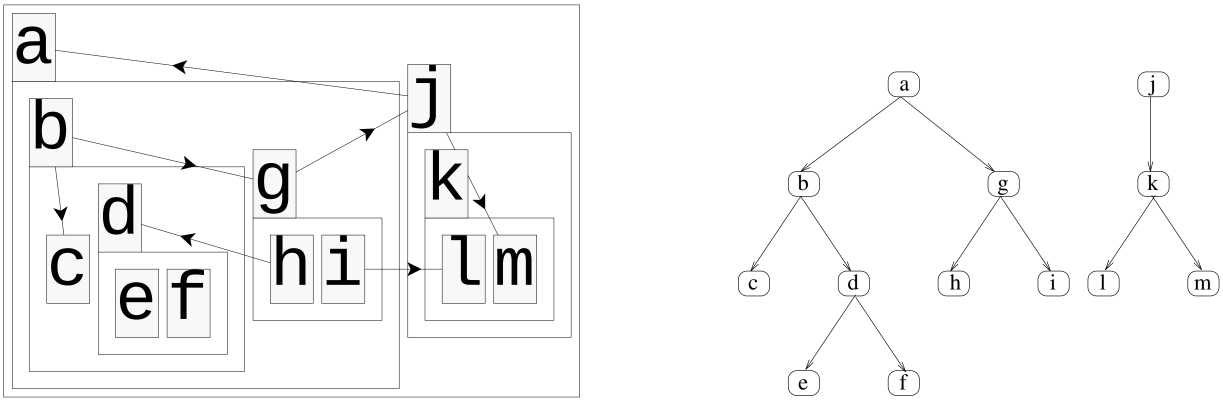


Figure 2: A sample graph and its nesting forest.

of v is computed as the smallest rectangle that fully encloses v's label and subgraph of contained vertices.

Graph layout: In drawing nested graphs, topological nesting is naturally depicted by visual containment. A simple, bottom-up algorithm is:

1. foreach $v \in G_n(V)$ in postorder wrt $G_n(A)$
2. layout graph $\langle ch(v), (ch(v) \times ch(v)) \cap G(A), \emptyset \rangle$
3. compute bounding box for v
4. endfor
5. layout non-nested vertices

Step 2 positions all vertices nested within a given vertex relative to one another. Similar layout algorithms have been described in the literature [5, 12, 16, 19, 28]. As an aside, notice that a composite layout strategy [12] can be realized simply by associating a specific layout algorithm with each vertex.

4.2 Generalized FEVs of nested graphs

In this formulation, the abstract view elements are vertices, thus $F \subseteq G(V)$ is a set of FPs. The DOI of $v \in G(V)$ is expressed as:

$$doi(v) = api(v) \cdot (p \cdot wt(v) + (1-p) \cdot proxim(v))$$

The functions api, wt and $proxim$ return non-negative real values: $api(v)$ is v's API, $wt(v)$ is a generic measure of v's proximity to the FPs, $proxim(v)$ is an application specific measure of v's proximity to the FPs, and $0 \leq p \leq 1$ reflects the tradeoff between generic and domain-specific notions of distance. Note that api and $proxim$ are fixed by the semantics of the application, while wt is computed by our algorithm. To generate a FEV, the following line must be added to the nested layout algorithm:

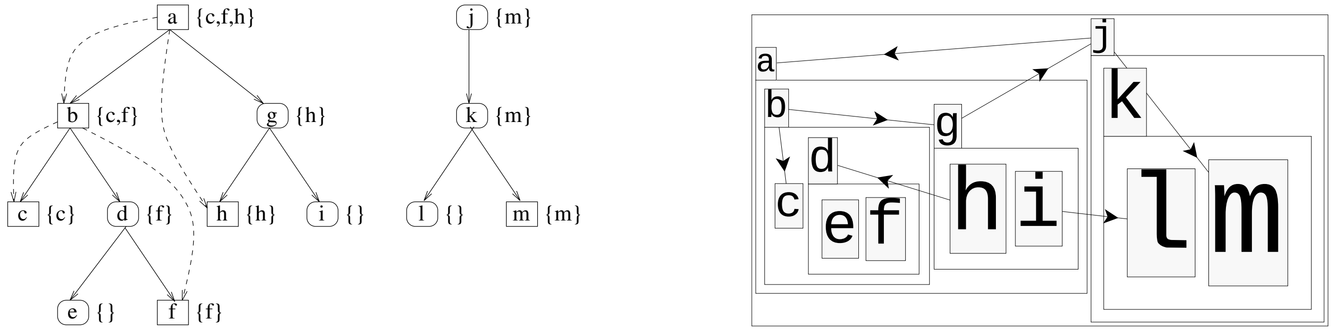
3.5 uniformly scale the bounding box of v by $\sqrt{doi(v)}$

which has the effect of increasing the total area of v's bounding box by $doi(v)$, consistent with the interpretation that the amount of screen space allocated to an element should reflect its DOI. The function mag maps a FP to its non-negative real-valued magnification. For example, $mag(f) = 4.0$ implies that the area allocated to vertex $f \in F$ in the FEV should be four times greater than what it would have been in a regular view.

4.3 Generic FP proximity algorithm

Given G_n, F, api, and mag, the algorithm determines a mapping wt such that

$$\forall\, f \in F, \prod_{v \in anc(f)} wt(v) = mag(f).$$



(a) The marked-up nesting forest (the labels of FP vertices have square corners while the rest have rounded corners).

(b) The resulting FEV.

Figure 3: The sample graph example continued.

Note that due to the relative geometry of vertices, the final size of a vertex depends on the product of the weights of its ancestors. The above formula, therefore, says that in the absence of domain-specific distance metrics (*i.e.,* $p = 1$ in $doi(f)$), and in addition to $api(f)$ contribution, each FP f will have its total area magnified by $mag(f)$.[1] The trick, then, is to distribute the value $mag(f)$ among the ancestors of f in such a way that the resulting layout will exhibit the desired FEV characteristics. Now, we present the algorithm.

Step 1: First, for each vertex $v \in G(V)$ we compute $key(v) = desc(v) \cap F$. Figure 3(a) shows the keys associated with the vertices in the sample graph given $F = \{c, f, h, m\}$. Vertices that share the same key form a path in the nesting forest, and are associated with the same FP: the path's sink, $s \in G_n(V)$; if $s \notin F$, s is a composite FP (a least common ancestor of two or more FPs). In our example, there are two composite FPs: a and b. The mapping $fpSet(f)$ returns the set of vertices associated with FP f. For example, $fpSet(m) = \{j, k, m\}$ and $fpSet(f) = \{d, f\}$.

Step 2: Having obtained the keys, we generate the FP containment graph $G_c = \langle G_c(V), G_c(A), \emptyset \rangle$. Each $v \in G_c(V)$ is a FP that either contains one or more FPs or is contained by a FP. $G_c(A) \subset G_c(V) \times G_c(V)$ is the set of (u, v) arcs such that u contains v and there is no w for which u contains w and w contains v. Following our example, G_c is the subgraph induced by the dashed arcs in Figure 3(a).

Step 3: We now compute the values $mag(f)$ for each composite FP f in a single postorder traversal of G_c. The magnification of each composite FP is a function of the set of FPs that it contains. For example, $mag(b) := \mu(\{mag(c)/\pi(c, b), mag(f)/\pi(f, b)\})$, where

$$\pi(u, v) = \prod_{w \in fpSet(u) \cup \{v\}} \delta(w, u, anc(u))$$

computes the product of vertex weights along the v to u path assuming that $F = \{u\}$, and $\delta(v, f, S)$ is a function that assigns a weight to v assuming that $mag(f)$ is distributed among the vertices in S. For the examples shown in this paper, $\mu(S)$ returns the minimum element in S, while $\delta(v, f, S)$ distributes $mag(f)$ evenly among all of the vertices in S (*i.e.,* $\delta(v, f, S) = mag(f)^{\frac{1}{|S|}}, \forall\, v \in S$).

Next, with a single postorder traversal of G_c, for every $(u, v) \in G_c(A)$, we scale FP magnifications as follows: $mag(v) := mag(v) \,/\, mag(u)$.

Step 4: Now we are ready to compute wt. First, wt is initialized as follows: $\forall\, v \in G(V)$, $wt(v) := 1.0$. Then, using the scaled magnification values, for every FP f (both composite and original) and for every vertex $v \in fpSet(f)$, we compute $wt(v) := \delta(v, f, fpSet(f))$.

The resulting FEV is shown in Figure 3(b).

[1] Note that $0 < mag(f) < 1$ could also be used to explicitly reduce f's area.

5 FEVs of nested networks

We now present two simple scenarios that demonstrate how FEVs can be used when exploring large nested hypertext networks.

5.1 Scenario 1

In our first example, a user navigates through a hyperdocument by following links, and uses an overview diagram as a "you are here" -style guide map. When, by following a link, the vantage point shifts to a distant or unfamiliar part of the network, the user would like to get a more detailed overview of the new region. Pan and zoom facilities can be used to do just that, however, it becomes more difficult to simultaneously display overviews of both, the original and target regions. The main deficiency of zoomed-in views is the loss of valuable intervening context that shows how such views are related – global context is exchanged for local detail.

Following our FTA example, suppose a user is exploring 'Article 2004' in 'Chapter 20' of the document, and discovers that there is a link to 'Annex 5' in 'Chapter 14'. The user would like to ascertain whether following this link will prove fruitful in his search, and wishes to examine an informative overview that shows both of these items and their neighbourhoods simultaneously. Since the current location is deemed by the user to be more important, a FEV with FPs: {*(2004 Intellectual*,20),(5 Monopolies,10)*} is produced. These settings result in the former item's total area being increased by 2000 percent and the latter's area increased ten-fold, in relation to the original nested overview diagram, as shown in Figure 8.

This visualization successfully combines local detail and global context: the relationships among 1680 nodes and 1512 links are preserved while two regions have been simultaneously magnified so that contained labels can be easily discerned.

5.2 Scenario 2

A second type of task that is quite common in the context of hypertext exploration is the discovery of common features in the network. Powerful text searching engines such as pat [9], developed for the new Oxford English Dictionary project, can be used to explore the textual information in a hyperdocument.

Returning to our FTA example, suppose that a user is interested in finding all the occurrences of the word *trade* in a full document heading. As it turns out, there are four such instances; centering a FP on each one results in the diagram shown in Figure 9. Once again, the details of the four regions are clearly visible, as are the features of the global structure.

6 Implementation

All Figures shown in this paper were produced by *graphite*: a suite of Unix programs developed at the University of Toronto [21]. Graphite comprises approximately 15K lines of C++ source code, and includes programs that can perform composite graph layout of nested graphs, produce several types of FEV visualizations (including those obtained by the technique described in this paper), and convert graphs into Encapsulated PostScript. Several graphite components have been incorporated into existing systems, including G+/Hy+ [2] and 4Thought [3, 24].

7 Future directions

The application of our algorithm to nested hypertext networks suggests several directions for future research.

Our generic FP proximity algorithm ignores the arcs of the nested graph. We are currently investigating formulations of the DOI that capture link-sensitive distance metrics through appropriate choices of the application specific $proxim$ function. In particular, we are studying the generation of dynamic FEV navigation maps in which a node's DOI also reflects its abstract distance from the current node (user's location in the hyperdocument); that is, in addition to $wt(v)$, $proxim(v)$ may approximate v's connectivity to the current node. The computation of such a proximity measure requires the formalization of the DOI contributions of: a single link, a set of links on a path, and a set of paths (*e.g.,* the contribution of a single link may vary according to its type, direction, and proximity to the current node).

A second problem with our approach is that of *layout stability*. It is very common to position the nodes displayed in an overview diagram by an automatic layout algorithm. Since many such algorithms are sensitive to the sizes and shapes of the objects that they position, changing nodes' sizes (as done when generating FEVs) can greatly influence the output of the layout algorithm. During the process of creating a FEV, then, many nodes may be repositioned in relatively arbitrary ways, causing the user to become disoriented. It would be invaluable to determine the class of incremental, stable layout algorithms that could be used to automatically position the nodes of a nested network.

Another usability issue with the present version of the algorithm, is the requirement that the user supply exact magnification values when specifying FPs. An inexperienced user cannot easily predict the effect of FPs' magnifications on the resulting visualization – the effect is a function of the topology and size of the network – larger networks require "stronger" FPs to magnify interesting areas to an acceptable size. A more user-friendly solution would have the user supply a list of FPs that are partially ordered with respect to magnification, from which the exact magnification values could be determined algorithmically.

8 Conclusion

Exploring large hyperdocuments can be difficult in the absence of effective visualization facilities. Although the overview and pan and zoom techniques can help a user to understand the structure of large networks, these approaches do not support some forms of exploration. In particular, these approaches are inadequate at presenting several regions of a network in detail while simultaneously displaying the network's global structure. As one strategy for coping with this problem, we described a generalized approach for generating layout-independent fisheye views of nested graphs with multiple variable focal points, and demonstrated its feasibility as a tool to aid in the exploration of large nested hypertext networks.

References

[1] M. Bernstein, J.D. Bolter, M. Joyce, and E. Mylonas. Architectures for Volatile Hypertext. In *Hypertext '91*, pages 243–260, San Antonio, Tx, Dec. 1991. ACM.

[2] M.P. Consens, I.F. Cruz, and A.O. Mendelzon. Visualizing Queries and Querying Visualizations. *ACM SIGMOD Record*, 21(1):39–46, Mar. 1992.

[3] M.P. Consens, A.O. Mendelzon, and A.G Ryman. Visualizing and Querying Software Structures. In *14th Intl. Conf. on Software Engineering*, pages 138–156, 1992.

[4] G. Dittrich and K. Tochtermann. Browsing in unstrukturierten Hyperdokumenten. In *Hypertext/Hypermedia '91*, pages 75–86, Graz, Austria, Mar. 1991. Springer-Verlag.

[5] K.M. Fairchild, S.E. Poltrock, and G.W. Furnas. SemNet: Three-Dimensional Graphic Representations of Large Knowledge Bases. In R. Guindon, editor, *Cognitive Science and its Applications for Human-Computer Interaction*, pages 201–233. Lawrence Erlbaum Associates, 1988.

[6] S. Feiner. Seeing the Forest for the Trees: Hierarchical Display of Hypertext Structure. In *ACM OIS '88*, pages 205–212. ACM, March 23–25 1988.

[7] G.W. Furnas. Generalized Fisheye Views. In *ACM CHI '86*, pages 16–23, Boston, MA, Apr. 1986. ACM.

[8] R. Godin, J. Gecsei, and C. Pichet. Design of a browsing interface for information retrieval. In *ACM SIGIR '89*, pages 32–39. ACM, Jun. 1989.

[9] G. Gonnet. PAT 3.1: An efficient text searching system. Technical report, UW Centre for the New OED, U. of Waterloo, 1987.

[10] F.G. Halasz. Reflections on Notecards: Seven Issues for the Next Generation of Hypermedia Systems. *Communications of the ACM*, 31(7):836–852, Jul. 1988.

[11] D. Harel. On Visual Formalisms. *Communications of the ACM*, 31(5):514–530, May. 1988.

[12] T.R. Henry and S.E. Hudson. Interactive Graph Layout. In *UIST '91*, pages 55–64. ACM, 1991.

[13] B. Johnson and B. Shneiderman. Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures. In *IEEE Visualization '91*, pages 284–291, 1991.

[14] J.D. Mackinlay, G.G. Roberson, and S.K. Card. The Perspective Wall: Detail and Context Smoothly Integrated. In *ACM CHI '91*, pages 173–179. ACM, Apr. 1991.

[15] K.C. Malcolm, S.E. Poltrock, and D. Schuler. Industrial Strength Hypermedia: Requirements for a Large Engineering Enterprise. In *Hypertext '91*, pages 13–24, San Antonio, Tx, Dec. 1991. ACM.

[16] E.B. Messinger, L.A. Rowe, and R.R. Henry. A Divide-and-Conquer Algorithm for the Automatic Layout of Large Directed Graphs. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(1):1–12, 1991.

[17] K. Misue and K. Sugiyama. Multi-viewpoint Perspective Display Methods: Formulation and Application to Compound Graphs. In *4th Intl. Conf. on Human-Computer Interaction*, volume 1, pages 834–838, Stuttgart, Germany, Sep. 1991. Elsevier.

[18] D.A. Mitta. A Fisheye Presentation Strategy: Aircraft Maintenance Data. In *INTERACT '90*, pages 875–880. IFIP, Elsevier, 1990.

[19] F. Newbery Paulisch and W.F. Tichy. EDGE: An Extensible Graph Editor. *Software–Practice and Experience*, 20(S1):63–88, Jun. 1990.

[20] J. Nielsen. The Art of Navigating through Hypertext. *Communications of the ACM*, 33(3):296–310, Mar. 1990.

[21] E.G. Noik. Graphite: A Suite of Hygraph Visualization Utilities. In A.O. Mendelzon, editor, *Declarative database visualization: recent papers from the Hy+/GraphLog project*. Forthcoming technical report, CSRI, U. of Toronto, Jul. 1993.

[22] D.R. Raymond. Flexible text display with lector. *IEEE Computer*, 28(8):49–60, 1992.

[23] G.G. Robertson, J.D. Mackinlay, and S.K. Card. Cone Trees: Animated 3D Visualizations of Hierarchical Information. In *ACM CHI '91*, pages 189–194. ACM, Apr. 1991.

[24] A.G. Ryman. Foundations of 4Thought. In *CAS '92*, pages 133–155, Toronto, Canada, Nov. 1992.

[25] M. Sarkar and M.H. Brown. Graphical Fisheye Views of Graphs. In *ACM CHI '92*, pages 83–91, Monterey, CA, May. 1992. ACM.

[26] B. Schneiderman. Future Directions for Human-Computer Interaction. *International Journal of Human-Computer Interaction*, 2(1):73–90, 1990.

[27] K. Sugiyama and K. Misue. 'Good' Graphic Interfaces for 'Good' Idea Organizers. In *Human-Computer Interaction – INTERACT '90*, pages 521–526. IFIP, Elsevier, 1990.

[28] K. Sugiyama and K. Misue. Visualization of Structural Information: Automatic Drawing of Compound Digraphs. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(4):876–892, 1991.

[29] M. Travers. A Visual Representation for Knowledge Structures. In *Hypertext '89*, pages 147–158, Pittsburgh, PA, Nov. 1989. ACM.

[30] D. Turo and B. Johnson. Improving the Visualization of Hierarchies with Treemaps: Design Issues and Experimentation. In *IEEE Visualization '92*, pages 124–131, Boston, MA, 1992.

[31] K. Utting and N. Yankelovich. Context and Orientation in Hypermedia Networks. *ACM Transactions on Information Systems*, 7(1):58–84, Jan. 1989.

[32] N. Yankelovich, B.J. Haan, N. Meyrowitz, and S.M. Drucker. Intermedia: The Concept and the Construction of a Seamless Information Environment. *IEEE Computer*, 21(1):81–96, Jan. 1988.

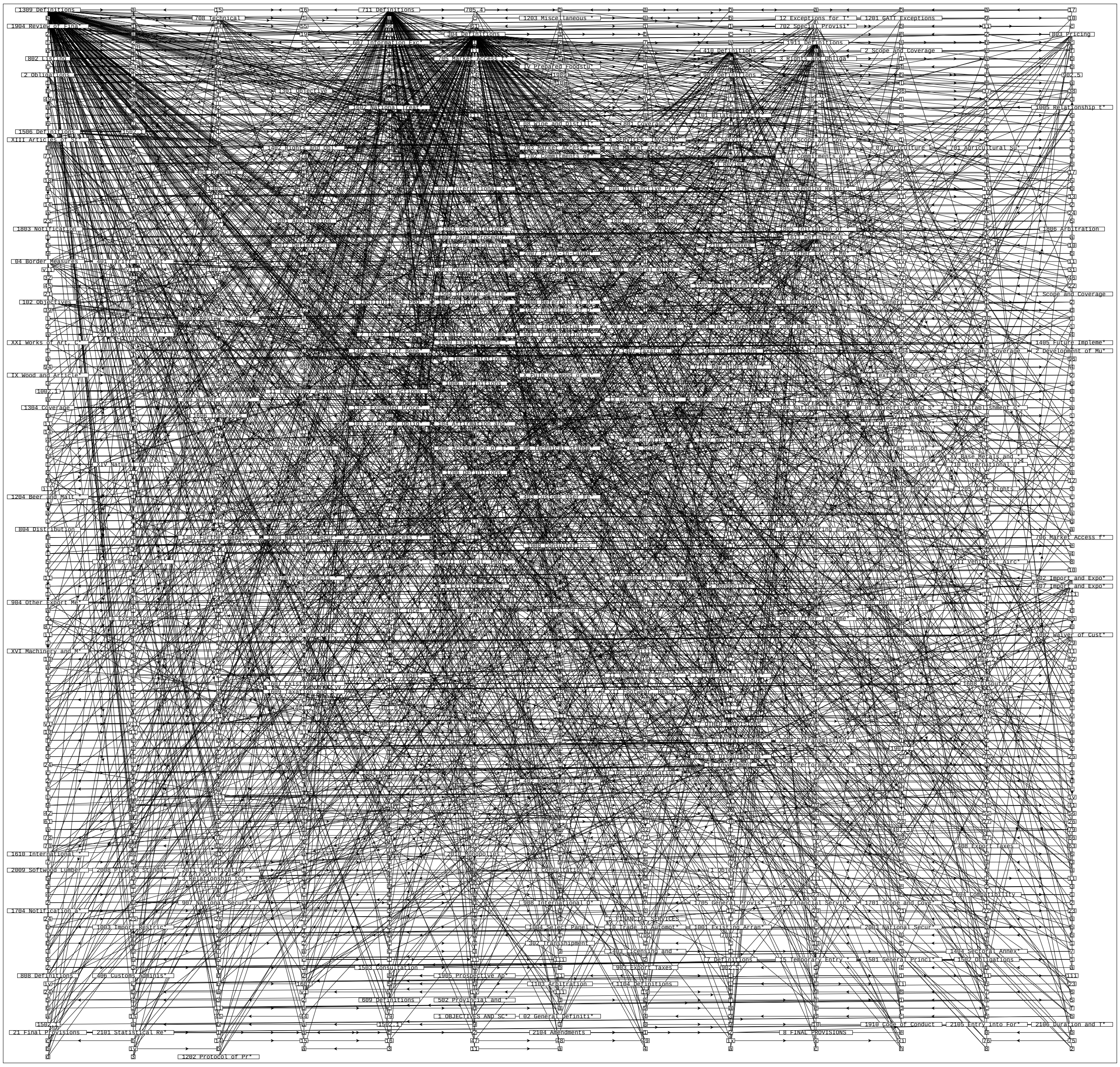


Figure 4: Original non-nested FTA hyperdocument.

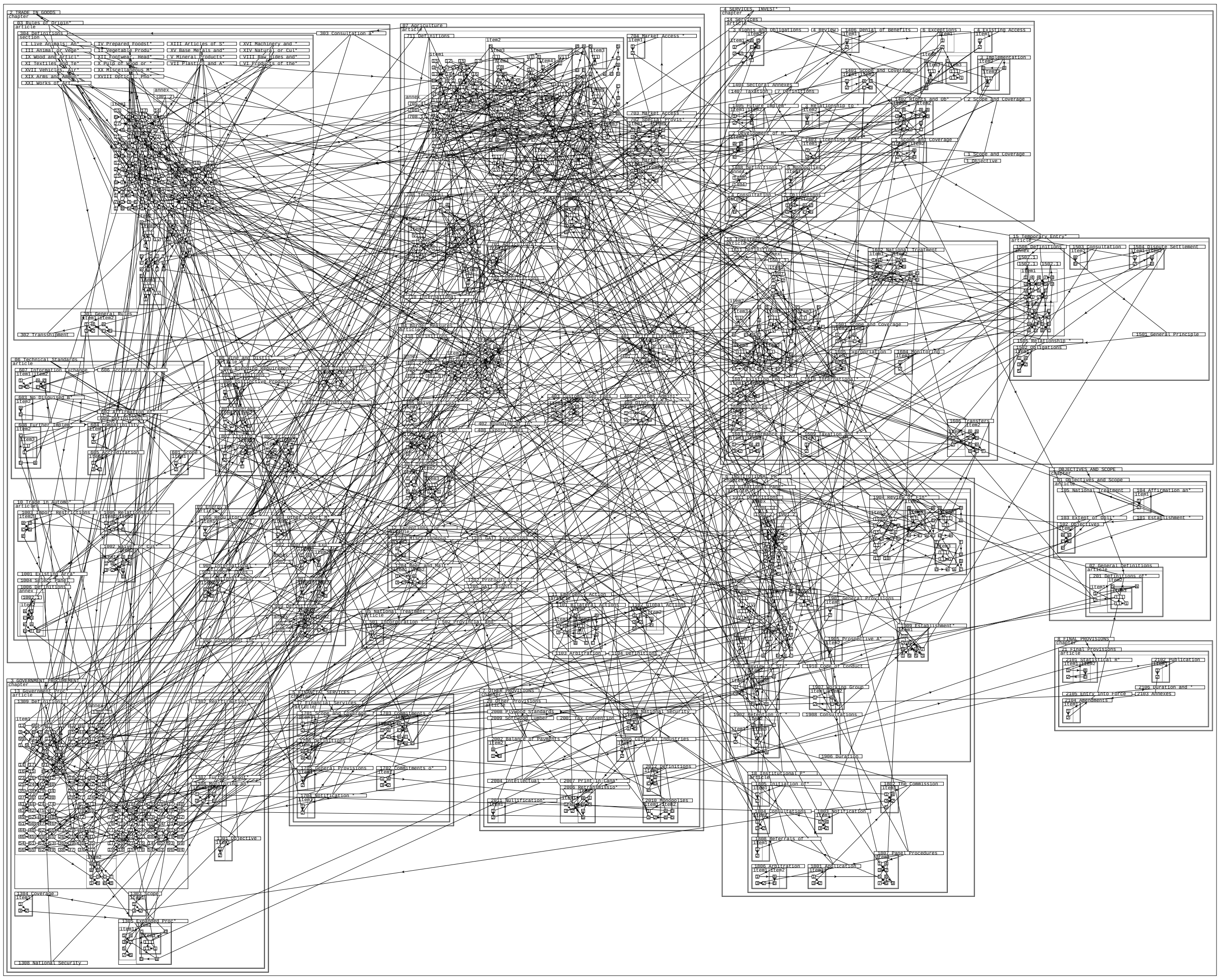


Figure 5: A nested version of the FTA hyperdocument.

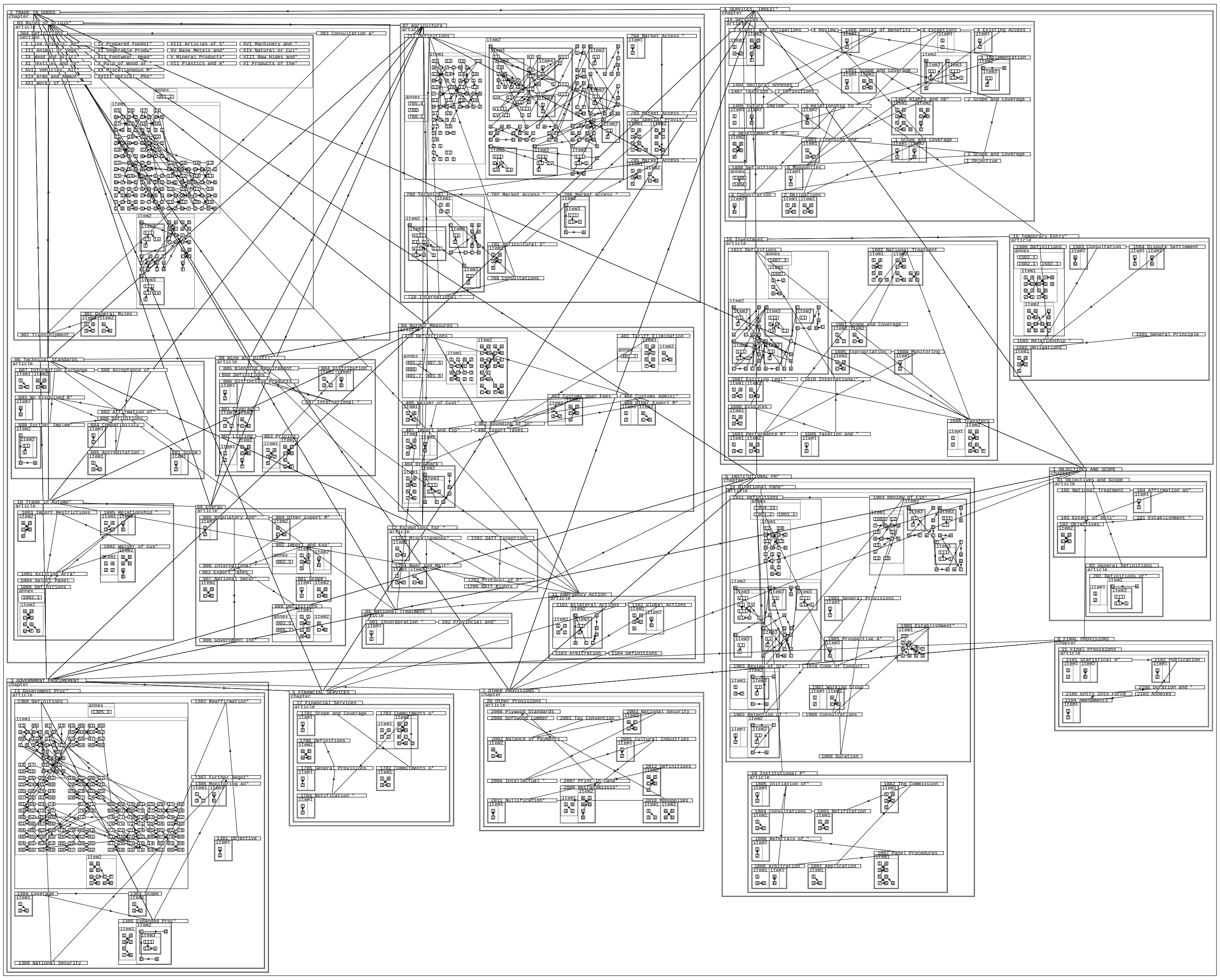


Figure 6: The nested version of the FTA with *link inheritance*.

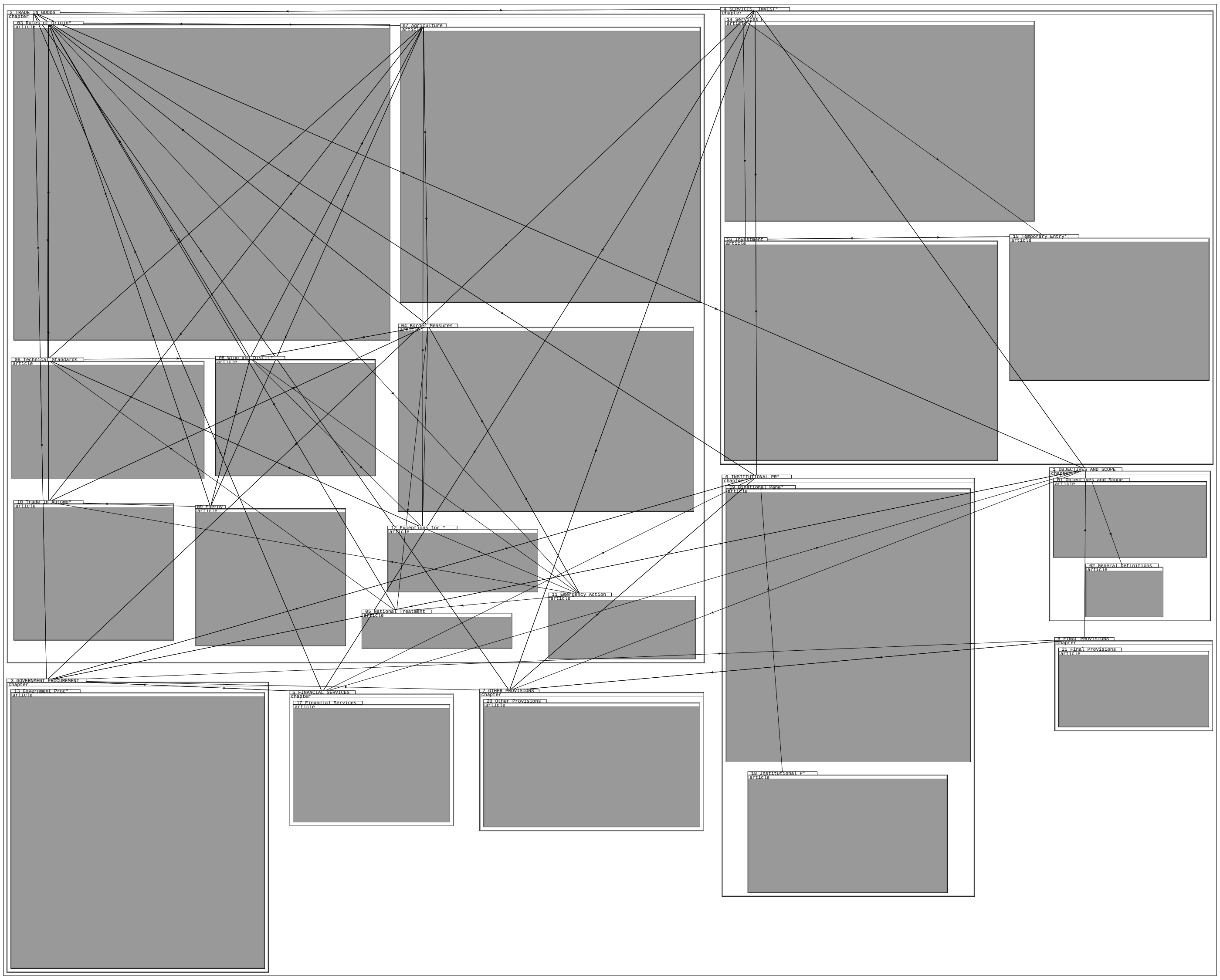


Figure 7: A parts and chapters view of the nested version of the FTA. Size of each part or chapter reflects the amount of content found within.

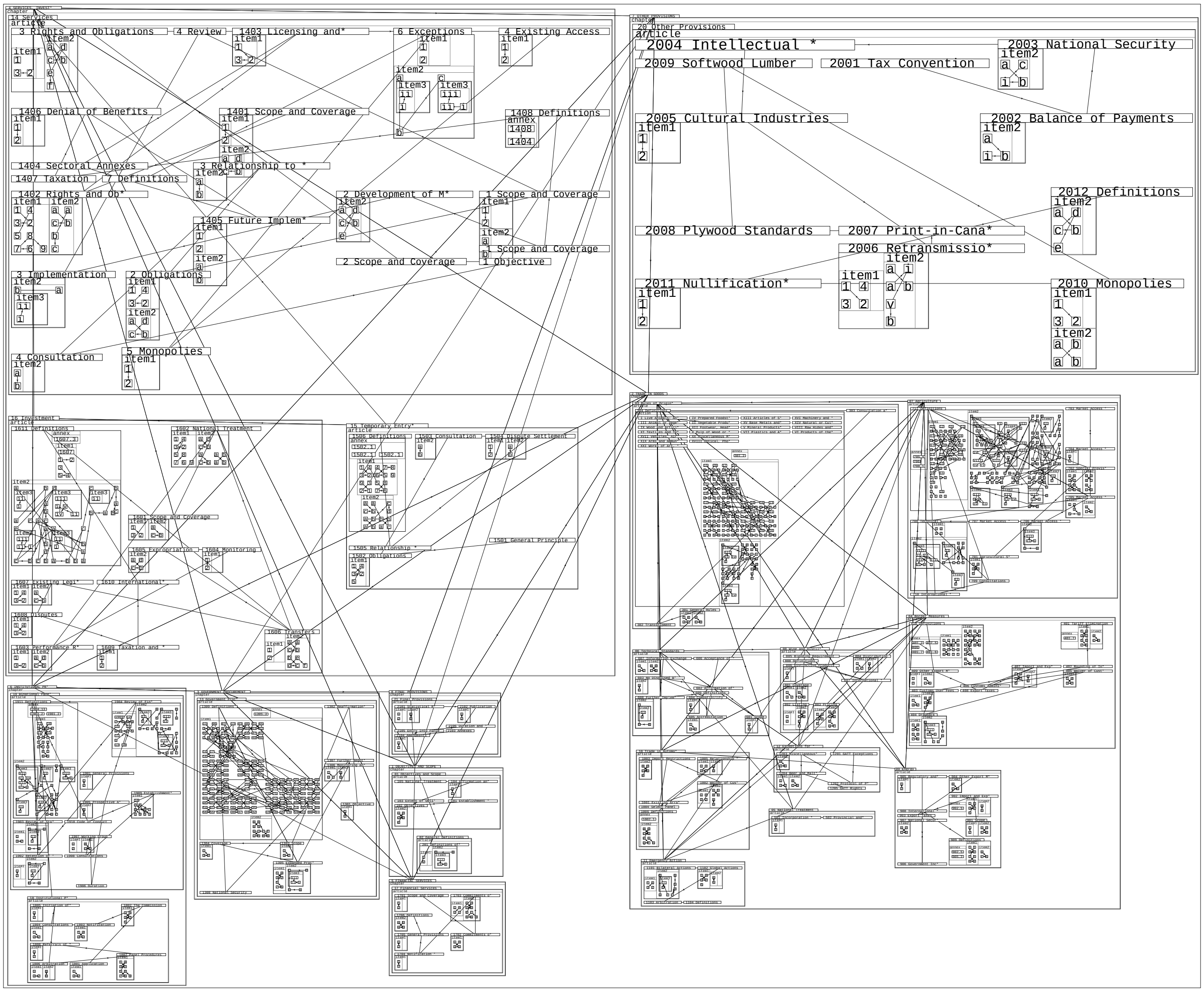


Figure 8: A fisheye view of the nested version of the FTA with *link inheritance*. Focal points are *2004 Intellectual ** (mag. 20) and *5 Monopolies* (mag. 10).

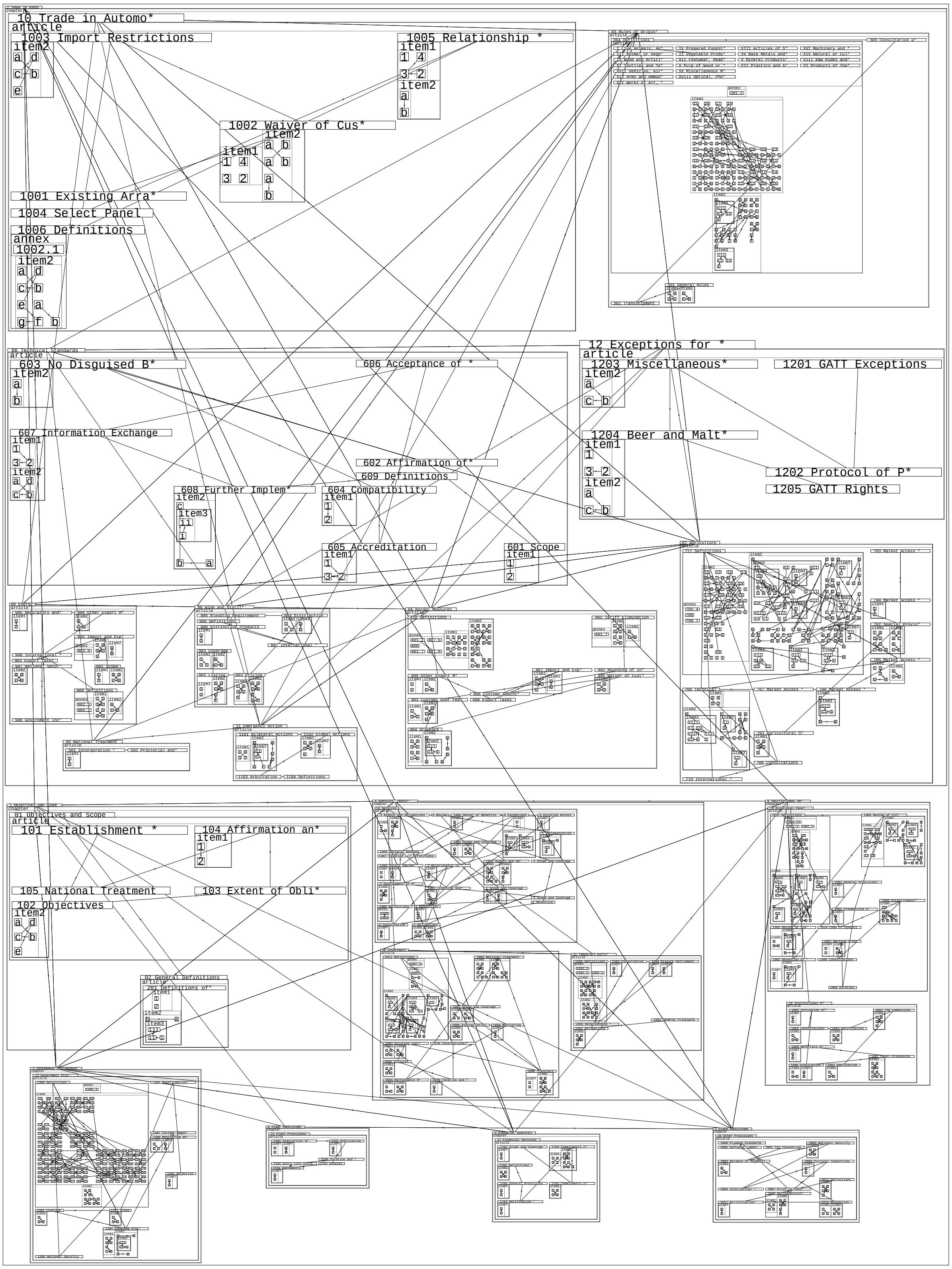


Figure 9: A different fisheye view of the nested version of the FTA. Here, focal points are: *101 Establishment o**, *603 No Disguised B**, *10 Trade in Automo**, and *12 Exceptions for ** (all have mag. 15).