Laying Out and Visualizing Large Trees Using a Hyperbolic Space

John Lamping and Ramana Rao

Published in Proceedings of the ACM Symposium on User Interface Software and Technology. November 1994. ACM Press. Pages 13-14.

Laying out and Visualizing Large Trees Using a Hyperbolic Space

John Lamping and Ramana Rao

Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
<lamping,rao>@parc.xerox.com

ABSTRACT

We present a new focus+context (fisheye) scheme for visualizing and manipulating large hierarchies. The essence of our approach is to lay out the hierarchy uniformly on the hyperbolic plane and map this plane onto a circular display region. The projection onto the disk provides a natural mechanism for assigning more space to a portion of the hierarchy while still embedding it in a much larger context. Change of focus is accomplished by translating the structure on the hyperbolic plane, which allows a smooth transition without compromising the presentation of the context.

KEYWORDS: Hierarchy Display, Information Visualization, Fisheye Display, Focus+Context Technique.

INTRODUCTION

The amount of information that can coherently be displayed on the screen of an interactive computer system can dramatically affect the ease of interacting with a large information structure. We present a new technique for displaying and manipulating large hierarchies.[1] This technique, illustrated in Figure 1, provides a smoothly varying "focus+context" or "fisheye" view well suited for visualizing trees. The display space allocated to a node falls off as a continuous function of its distance in the tree from the focus of attention. The context always includes several generations of parents, siblings, and children, limited only by display resolution, making it easy for the user to explore the hierarchy without getting lost.

The tree is initially displayed with its root at the center, but the display can be smoothly transformed to bring other nodes into focus, as illustrated in Figure 2. We have developed effective procedures for manipulating the focus using selection, as well as pointer dragging, and for smoothly animating transitions across such manipulation.

[1] Xerox Corporation is seeking patent protection for technology described in this paper.

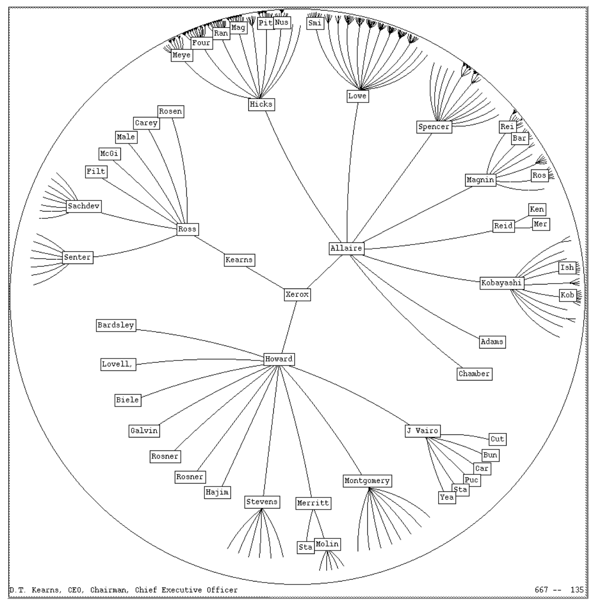

Figure 1: An organization chart displayed by uniformly embedding the tree on a hyperbolic plane and using the Poincaré mapping of the plane onto a disk.

This technique compares favorably with conventional hierarchy displays. In a 600 pixel by 600 pixel window, a conventional 2-d browser can typically display 100 nodes (w/ 3 character text strings). The hyperbolic browser can display 1000 nodes of which about the 50 nearest the focus can show from 3 to dozens of characters of text. Thus the hyperbolic browser can display up to 10 times as many nodes while providing more effective navigation around the hierarchy.

HYPERBOLIC TECHNIQUE

The essence of our approach is to lay out the hiearchy on the hyperbolic plane, and then map this mathematical construct onto a circular display region. The hyperbolic plane is a non-Euclidian geometry where parallel lines diverge from one another. This leads to the convenient property that the circumference of a circle grows exponentially with its radius, which means that exponentially more space is available with increasing distance. Thus hierarchies—which tend to expand exponentially with depth—can be laid out uniformly in hyperbolic space, so the distance between parents, children, and

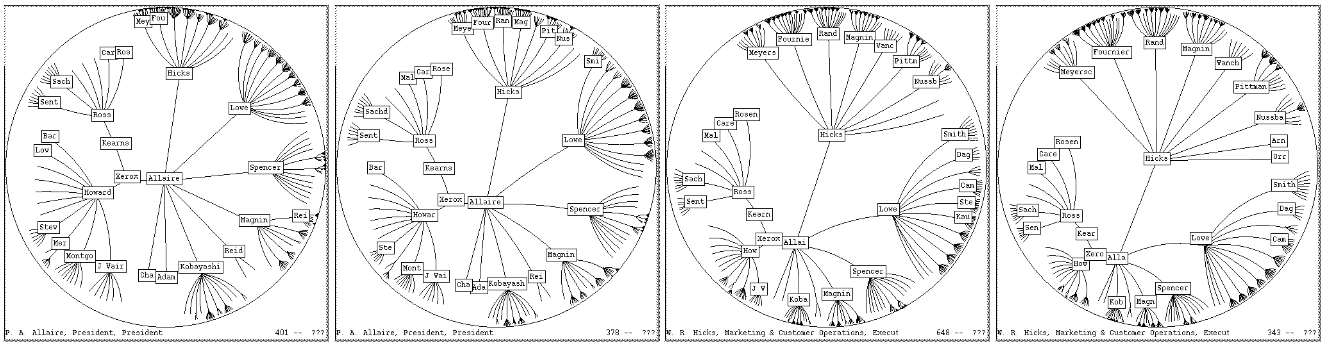

Figure 2: The user has clicked on the node “Hicks” which is the child of and above the central node. The animated transition brings that node to the center in a series of steps exposing more of its descendant structure.

siblings is roughly the same across the hierarchy.

We use a recursive algorithm that lays out each node based on local information. A node is allocated a wedge of the hyperbolic plane, angling out from itself, to put its descendants in. It places all its children along an arc in that wedge, at an equal distance from itself, and far enough out so that the children are some minimum distance apart from each other. Each of the children then gets a sub-wedge for its descendants. Because of the divergence of parallel lines in hyperbolic geometry, each child will typically get a wedge that spans about as big an angle as does its parent’s wedge.

The hyperbolic plane can be mapped in a natural way onto the unit disk, which provides a means for displaying it on a 2-D (Euclidean) display. There are two canonical ways of mapping the hyperbolic plane to the unit disk, the Klein model (projective) and the Poincaré model (conformal). In both of them, one vicinity in the hyperbolic plane is in focus at the center of the disk while the rest of the hyperbolic plane fades off in a perspective-like fashion toward the edge of the disk. We have found that the Poincaré model (illustrated in the figures) works much better for visualizing hierarchies, because it preserves the shapes of fan-outs at nodes and does a better job of using the screen real-estate.

CHANGING FOCUS

A change of focus is implemented as a translation of the structure on the hyperbolic plane, providing a continuous geometric model. The user can change focus either by clicking on any visible point to bring it into focus at the center, or by dragging any visible point to any other position (with immediate feedback). In either case, the rest of the display transforms appropriately. Regions that approach the center become magnified, while regions that were in the center shrink as they are moved toward the edge. Thus the user can browse the tree structure while maintaining the visual context.

The user’s understanding of a jump to a new focus is greatly improved if the transition is animated. This is done by calculating an “nth-root” transformation, a transformation that when applied n times will have the same effect as the original, and using this transformation successively to generate intermediate frames, as illustrated with an n of 3 in Figure 2.

Doing successive translations in the hyperbolic plane will, in general, cause rotations, which users find utterly counter-intuitive. To avoid this, our system adds a rotation component to translations so that the node in focus on the display will have a canonical orientation. The canonical orientation that we have found most effective is to have the parent of the focus node be in a canonical direction.

Responsive display performance is crucial for supporting interactive dragging of the structure and for supporting animated transitions on jump changes. This can be a problem for large hierarchies on standard hardware. We achieve rapid redisplay on standard hardware by compromising on display quality during animation and interactive dragging. Fortunately, the rendering compromises don’t affect the coherence of the motion and are often unnoticed at animation speeds. Two effective compromises are to draw less of the fringe and to draw lines, rather than arcs. Another compromise we considered was to drop text during animation, but it destroyed coherence of motion.

CONCLUSION

Hyperbolic geometry provides an elegant solution to the problem of providing a focus+context display for large hierarchies. The hyperbolic plane has the room to lay out large hierarchies uniformly, and the Poincaré map provides a natural, continuously graded, focus+context mapping from the hyperbolic plane to a display.

The hyperbolic browser supports interaction with larger hierarchies than conventional browsers with modest computational requirements. It supports up to 10 times as many nodes while providing more effective navigation around the hierarchy. It should be more systematically compared with other novel browsers such as the Cone Tree[2] and Treemaps [1].

REFERENCES

1. B. Johnson and B. Shnedierman. Tree-maps: A space-filling approach to the visualization of hierarchical information. In *Visualization 1991*, pages 284–291. IEEE, 1991.

2. G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: Animated 3d visualizations of hierarchical information. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 189–194. ACM, April 1991.