Animating Treemaps

Mohammad Ghoniem and Jean-Daniel Fekete
Ecole des Mines de Nantes
4, rue Alfred Kastler
44300 Nantes – France
Tel: 33-2-51-85-82-41
E-mail: jdf@emn.fr; mghoniem@emn.fr

ABSTRACT
Treemaps have so far been used to display static views of hierarchical information mainly for exploratory purposes. Most of the time, the data at hand are multivariate and the user is often interested in seeing alternative views of the same hierarchy by changing the visible variables used for weighing or filling. Consequently, some areas of the display shrink while other areas expand so much so that the transition between the alternative views of the same hierarchy are impossible to follow. This paper describes early attempts at using animation with two treemap layouts.

KEYWORDS: Visualization, animation, treemaps, zoom

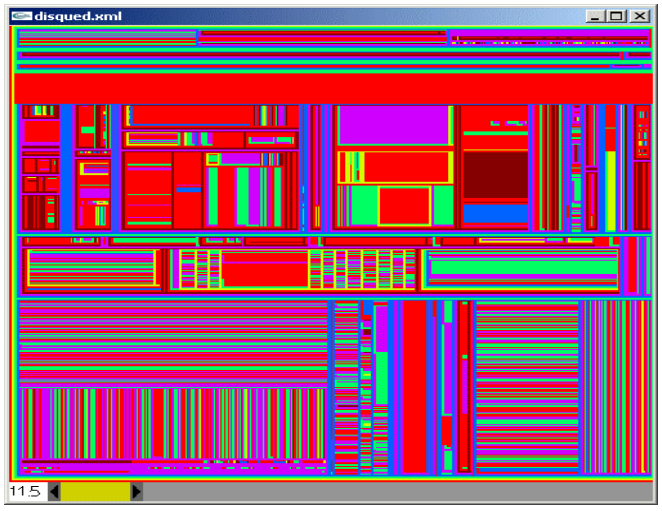
INTRODUCTION
Hierarchical information structures are everywhere. The need for an efficient way to visualize such structures, especially large trees, has given birth to the treemaps technique as developed by Shneiderman and Johnson [2]. Other variants such as the nested treemaps [2], squarified treemaps [1], and cushion treemaps [3] dealt with some shortcomings of the initial slice and dice technique.

Animating Treemaps
Handling multivariate data implies that there are several representations of the same structure depending on the variable chosen to weight the nodes and the one used for filling it. Since the weight of a node is proportional to the area of the rectangle associated to it in the visualization, changing the weighing variable makes some areas of the visualization shrink while other areas may expand considerably. Thus, the user may have hard time matching various representations of the same structure. This lead us to try and animate the transition between two alternative representations of the same tree.

Moreover, while visualizing large structures the user often needs to zoom in and out on certain regions of the display. Once more, animation proves interesting in the course of such a process since it helps the user keep track of the observed area. However, one must consider implementing a deformation algorithm that would hide/restitute the context gradually.

It has always been assumed that the data structures being visualized are static i.e. they do not evolve during the visualization. However, there are many cases where nodes may be added or removed from the hierarchy bringing about changes in the balance of the tree. These changes could be spotted and tracked thanks to animation. For instance, this may apply to real-time problems and help understand the behavior of certain algorithms through direct observation.

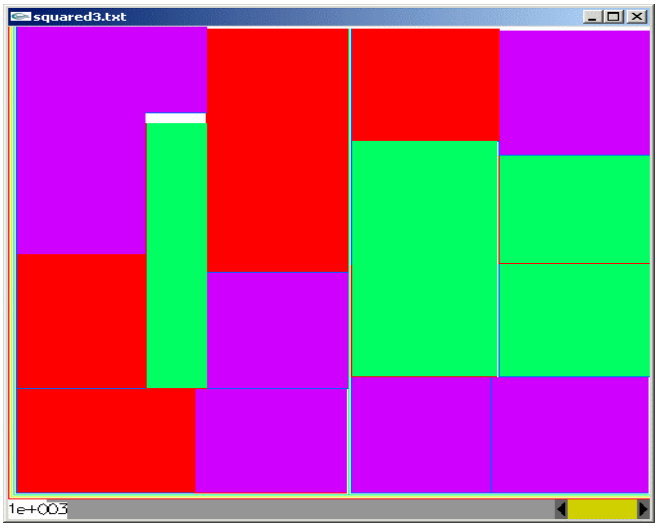
Materials
We have implemented such animations using OpenGL on a Pentium III 600MHz provided with an accelerated Geoforce2 video card. They were tested on various hierarchies ranging from 10 to 600,000 nodes. In order to speed up the display, we chose not to display any node (and its subsequent children) whose dimensions are less than one pixel.

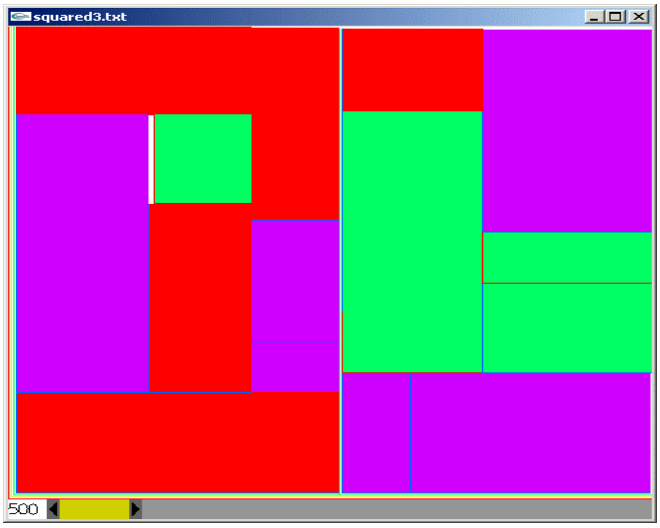


Results & Discussion

For as many as 60,000 nodes, the animation is performed at 30 fps approximately during the animation. However, when we take 10 times as many nodes, the animation is carried out at about 6 fps.

The animation of the transition between two weight distributions in a tree structure is quite smooth and makes it possible for the user to match the various regions of the structure at hand on both visualizations. We yet have to test it on a significant population to its impact.





We also applied animation to squarified treemaps. As one may expect, leaps occurred when we animated the visualization because, while some nodes expanded, the squarification algorithm would shift them from their actual column to the next row and vice versa. In order to minimize the effect of such leaps during the animation, we decided that rows and columns would alternate within a spiral counterclockwise. Consequently, instead of leaping, the rectangles rotate according to the schema : west, south, east, north etc. Thus, we can preserve a continuity during the animation.

Another advantage of this shell-like variant of squarified treemaps is the restitution of a certain notion of order that is present in the slice and dice algorithm and faints during the squarification process. Therefore, the presence of color patterns may still have a chance to be detected at least on limited structures. We yet have to experiment the validity of this assumption on a neutral population with a wide range of hierarchies.

Conclusion

Animating treemaps provide smooth transition between alternative views of the same hierarchy. This is particularly helpful during the visualization of multivariate data structures. It also seems to be interesting while the user zooms in and out on a particular region of the structure. And, of course, it opens the use of treemaps visualizations to real-time data. Further research and experimentation have yet to be carried out in this domain.

REFERENCES

1. Bruls, Huizing & van Wijk, Eindhoven, Squarified Treemaps, Eurographics/IEEE TVCG 2000 Symposium, University of Technology, Dept. of Mathematics and Computer Science, The Netherlands, 2000.

2. Johnson & Shneiderman, Treemaps : A Space-Filling Approach to the Visuaization of Hierarchical information Structures, Proceedings of IEEE Visualization 91, IEEE Computer Soc. Press, Los Alamitos, California, October 1991, pp. 189-194.

3. Van Wijk & Van de Wetering, Cushion Treemaps – visualization of hierarchical information, Proceedings 1999 IEEE Symposium on Information Visualization (InfoVis'99), pp. 73-78, October 1999.