Eurographics Conference on Visualization (EuroVis), Posters Track (2017), pp. 1–3
A. Puig Puig and T. Isenberg (Guest Editors)

Quantitative Comparison of Treemap Techniques for Time-Dependent Hierarchies

E.F. Vernier[1] & J.L.D. Comba[1] & A.C. Telea[2]

[1]Federal University of Rio Grande do Sul, Brazil
[2]University of Groningen, The Netherlands

Abstract

Hierarchical data in which leaf nodes have associated attributes appear in several applications. Rectangular Treemaps (RTs) were designed to display this type of data by tightly packing cells representing the tree nodes, with additional information stored in the cell sizes and/or colors. For dynamic hierarchies, RTs have to obey several requirements regarding the optimal aspect ratio of cells, but also the stability of layout in presence of data changes. While static RTs studied the first requirement well, far less information is available on how RTs behave vs *both requirements for dynamic data. We study how four known RT methods compare over several real-world dynamic hierarchies, and highlight how recent RT methods can be adapted to effectively handle optimal cell ratios and stability.*

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms

1. Introduction

Rectangular Treemaps (RTs) are space-filling methods that show large and complex hierarchies by a recursive subdivision of a starting rectangular domain [Tel15, Mun15]. Formally put, for a dataset $D = \{d_i\}$, a RT method is a function $T : D \rightarrow \mathbb{R}^2$ that maps a data element d_i to a rectangle $T(d_i)$. An important quality goal for RT methods is to maximize the aspect ratio of such rectangles to help their visibility [BHvW99, SW01]. However, using RTs for *dynamic* datasets $D(t) = \{d_i(t)\}$, $t \in \mathbb{R}^+$ is far more challenging: For such data, we want to ensure not only good cell aspect ratios, but also an amount of change in the layout $T(D(t))$ which reflects well the amount of actual change in the data $D(t)$. That is, small data changes should yield small layout changes (or *stable* layouts), while large layout changes should only occur when $D(t)$ changes a lot. While optimal aspect ratio in static RTs is well studied, only few insights exist for optimal aspect ratio *and* stability in dynamic RTs [SW01].

Recently, the NMap method [DSF*14] proposed a different way to construct RTs: For each $d_i \in D$, a point $\mathbf{x}_i$ is placed in 2D so that distances $\|\mathbf{x}_i - \mathbf{x}_j\|$ reflect the similarities of elements d_i and d_j, using dimensionality-reduction (DR) techniques. Next, cells $T(d_i)$ are built around $\mathbf{x}_i$ by iterative slice-and-scale. Interestingly, DR methods have shown, in a different context, very good ability to map time-dependent data in a stable way [RFT16]. Hence, it is interesting to compare NMap with more classical RT methods for the task of visualizing dynamic hierarchies. This is the goal of this paper: We compare four RT methods on several real-world dynamic hierarchies and measure cell aspect ratio and stability. We show that NMap better preserves these metrics than all other tested methods. As such, we believe that DR methods open new interesting ways for RT construction for dynamic data.

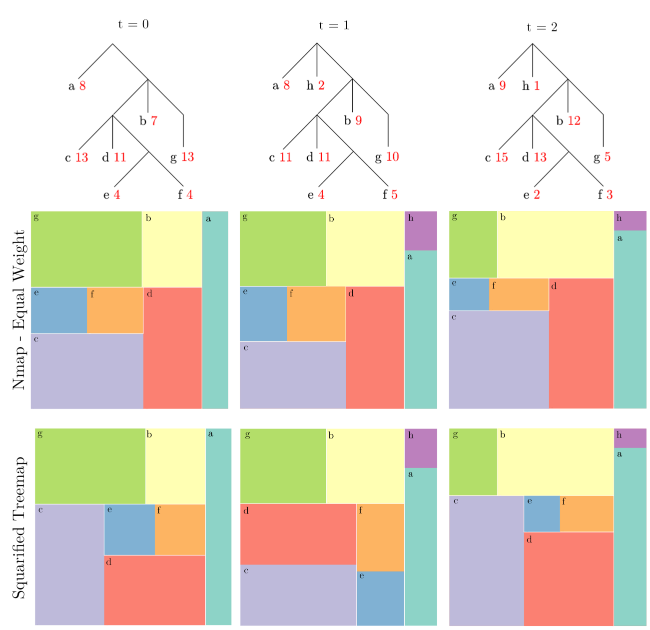

Figure 1: *Layouts generated by NMap - Equal Weight (NM-EW) and Squarified Treemaps (SQR) for a sample dataset. We see how NM-EW is more stable, and similar in aspect ratio, than/as SQR.*

2. Materials and Method

We tested four layout techniques and their variants: *Slice-and-Dice (SND)* [Shn92], *Squarified Treemap (SQR)* [BHvW99], *Ordered Treemap* (Pivot-by-Middle (OT-PBM) and Pivot-by-Size (OR-PBS) variants) [SW01], and *Nmap* (Alternate Cut (NM-AC) and Equal Weights (NM-EQ) variants) [DSF*14]. SND uses parallel lines to cut a rectangle (parent node) into smaller rectangles (children). As we descend the hierarchy, cut orientations – vertical *vs* horizontal – are switched. SND is inherently stable regarding cell positions, but often creates rectangles with a high aspect ratio, which are hard to see, select, and compare. SQR uses a heuristic that places cells so as to optimize aspect ratios. However, instabilities appear – small changes in D can yield large changes in $T(D)$. OT aims to compromise between the previous techniques, addressing both instability and bad aspect ratios. Finally, NM aims to place similar cells (given a user-selected distance $\delta : D \times D \rightarrow \mathbb{R}^+$) using a slice-and-scale strategy. While NM did not aim to handle dynamic data, we realized that its DR-based design *could* just do that intrinsically, given proven stability of related DR methods [RFT16].

Let us now consider *change* in $D(t)$. This occurs either in terms of added and/or deleted elements $d_i(t)$ or changes of their weights which are mapped to cell sizes. Let $\{t_j\}, 1 \leq j \leq N$ be the number of dynamic datasets $D(t_j)$ we have in a sequence. We measure the quality of a dynamic RT by two metrics: First, we measure how much centers of cells $T(d_i(t))$ move between two consecutive time moments t_j and t_{j+1} via

$$mov(D) = \frac{1}{N}\sum_{N}\left(\sum_{d_i \in D(t_i)\cap D(t_{i-1})} \|C(T(d_i(t)) - C(T(d_i(t-1)))\|\right), \quad (1)$$

where $C(\mathbf{r})$ is the center of a RT rectangle $\mathbf{r}$ and $\|\cdot\|$ is the Euclidean 2D distance. Secondly, we measure the aspect-ratios of generated treemaps (leaf and non-leaf cells):

$$aspect(D) = \frac{1}{N}\sum_{N}\left(\sum_{d_i \in D(t_i)} \min\left(\frac{W(T(d_i(t)))}{H(T(d_i(t)))}, \frac{H(T(d_i(t)))}{W(T(d_i(t)))}\right)\right), \quad (2)$$

where $W(\mathbf{r})$ and $H(\mathbf{r})$ are the width and height of a rectangle $\mathbf{r}$. For both Eqns. 1 and 2, we render the RTs $T(D(t))$ into a canvas of resolution 1000^2 pixels.

We used seven datasets in our experiments. The first five describe the evolution in size (*i.e.*, lines of code (LOC)) of open-source Github software projects, each having from 50 to 100 revisions. The projects are: *GIMP* (C++, 668 KLOC), *ExoPlayer* (Android, 64 KLOC), *iina* (Swift, 16 KLOC), *calcuta-android* (Android, 5.5 KLOC) and *bikedeboa* (JavaScript, 8 KLOC). The hierarchy is given by the package/directory/class organization. The sixth dataset is a yearly World Bank report [Wor15] active since 1960 that groups world countries into 7 geographical regions and gives the percentage of each country's GDP derived from the export of goods. The seventh dataset is a synthetic one that uses strings as seeds to generate a hierarchy and revision values using polynomial functions.

3. Findings

Figure 2 shows the results of our evaluation. We see that the Squarified Treemap (SQR) offers the best aspect ratios, but is the least stable of all techniques. Slice-and-Dice (SND) is at the other end,

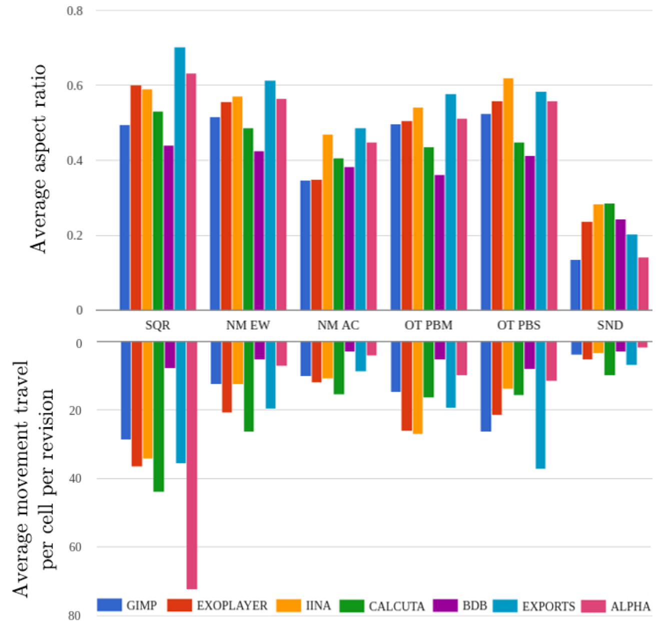

Figure 2: *Average movement and aspect ratios for seven datasets using Squarified Treemaps (SQR), NMap Equal Weight (NM-EW), NMap Alternate Cut (NM-AC), Ordered Treemaps Pivot-by-Middle (OT-PBM), Ordered Treemaps Pivot-by-Size (OT-PBS), and Slice-and-Dice (SND) layouts. See Sec. 3.*

being very stable but having poor aspect ratios and, thus, poor visibility for small cells. NMap and Ordered Treemaps offer a trade-off between stability and aspect ratios. More interestingly, we see that both NMap strategies (NM-AC and NM-EW) are stabler than the Ordered Treemap techniques (OT-PBM and OT-PBS). Out of the four techniques that offer this trade-off, NM-EW has the *best* aspect ratios, while NM-AC is significantly *stabler* than all others. Figure 1 shows this for a small dataset having $N = 3$ time steps: We see that NM-EW is more stable, and achieves similar aspect ratio (top row) than the well-known SQR method (bottom row). Overall, we conclude that *NMap* achieves good results on both stability and aspect ratios, so it is the best RT method (from the studied ones) for dynamic hierarchy visualization. It outperforms Ordered Treemaps, which is state-of-the-art in this context. We believe that this finding is worth being mentioned, therefore the aim of this paper.

To conclude: We have explored the quality of RT methods for showing dynamic hierarchies by measuring the average movement (stability) and average aspect ratio (readability) of treemap cells. Out of four state-of-the-art tested methods, we see that the recent, and far less well-known *NMap* [DSF*14] performs better in both metrics. The key interest in this result is that *NMap* was *not* designed, nor promoted, as a dynamic hierarchy visualization technique. Our results thus show that adapting DR methods such as *NMap* can potentially improve the quality of dynamic RTs. As a limitation of our work, we should mention that we only focused on *rectangular* treemaps; other treemap techniques, such as Voronoi ones, have recently shown marked abilities in preserving both aspect ratios and stability [vHH15,HTMD14]. Studying such alternative treemap methods, along classical RTs, is a focus of future work on dynamic hierarchy visualization.

References

[BHvW99] BRULS M., HUIZING K., VAN WIJK J.: Squarified treemaps. In *Proc. VisSym* (1999), pp. 33–42. 1, 2

[DSF*14] DUARTE F. S. L. G., SIKANSI F., FATORE F. M., FADEL S. G., PAULOVICH F. V.: Nmap: A novel neighborhood preservation space-filling algorithm. *IEEE TVCG 20*, 12 (2014), 2063–2071. 1, 2

[HTMD14] HAHN S., TRÜMPER J., MORITZ D., DÖLLNER J.: Visualization of varying hierarchies by stable layout of Voronoi treemaps. In *Proc. IVAPP* (2014). 2

[Mun15] MUNZNER T.: *Visualization analysis and design*. CRC Press, 2015. 1

[RFT16] RAUBER P., FALCÃO A., TELEA A.: Visualizing time-dependent data using dynamic t-SNE. In *Proc. EuroVis (short papers)* (2016). 1, 2

[Shn92] SHNEIDERMAN B.: Tree visualization with tree-maps: 2D space-filling approach. *ACM Trans. Graph. 11*, 1 (1992), 92–99. 2

[SW01] SHNEIDERMAN B., WATTENBERG M.: Ordered treemap layouts. In *Proc. IEEE InfoVis* (2001), pp. 73–81. 1, 2

[Tel15] TELEA A. C.: *Data visualization – Principles and Practice*, 2 ed. CRC Press, 2015. 1

[vHH15] VAN HEES R., HAGE J.: Stable Voronoi-based visualizations for software quality monitoring. In *Proc. IEEE VISSOFT* (2015), pp. 6–15. 2

[Wor15] WORLD BANK AND OECD NATIONAL ACCOUNTS: Exports of goods and services, 2015. data retrieved from World Development Indicators, http://data.worldbank.org/indicator/NE.EXP.GNFS.ZS. 2