Visualisation of the file system with force-directed graph and Divide & Conquer Treemap.

Introduction

Visualisation is a perceptually and cognitively effective tool for the display and interaction of the data, when used effectively it can unveil patterns and abnormalities in the system (Kerren et al. 2008, p. 93; Liang et al. 2013, p 79).

In the real world, some of these data are in hierarchical forms like file system which have a large amount of elements and relationships of the elements. Graphs are used to describe the relationship between the entities. A graph refers to a set of nodes, represented by circle or boxes, and a set of edges, represented by lines, that connects the pairs of nodes to illustrate the relationship. Graphs can be categorized into undirected and directed graphs. A tree is a connected undirected graph without cycles between the nodes. (Di Battista et al. 1994, p 40; von Landesberger et al. 2011, p. 1721). The techniques used in the hierarchical visualisation are connection and enclosure. The connection method focuses on the relationship between the nodes while the enclosure method utilizes area and area containment for hierarchically structured data. The enclosure method, such as treemap, is suitable for use on large hierarchical data sets with attributed properties (Liang et al. 2015, p. 104-105).

Researchers (Bennett et. al 2007, p. 1; Eades et al. 2010, p.6-4; Hu 2005, p. 37; Purchase 2002, p. 502; Ware et al. 2002, p. 103) associate aesthetics with readabilities and readabilities with understanding. It is usually a challenge when drawing graphs with a very large data set (Bennett et. al 2007, p.1; Eades et al. 2010, p.6-4). The aesthetics criteria used by most are:

- minimize overlapping/crossing edges and nodes
- minimize edge length.
- reduce the number of edge bends, if unavoidable keep them uniforms with respect to bends' position.
- clustering of related nodes for huge data sets
- the vertex resolution of a drawing is the minimum distance between a pair of vertices
- symmetrical graph
- evenly distributed vertices

It is usually not practical, if not impossible, to satisfy all these criteria, one should decide which is the most relevant for the critical mass and possibly compromise on the least relevant (Purchase 2002, p. 502).

This paper will discuss the technical details, advantages and disadvantages of Divide and Conquer Treemap [D&C Treemap] and Collapsible force layout. It will also review and analyze the visualisation generated by the tools on the file system.

Treemap and its advantages and disadvantages.

Johnson and Shneiderman (1991, p. 284) first proposed the concept of the treemap, which makes high utilization of the available display space by mapping the hierarchical structure in its entirety onto a rectangular space in a space-filing manner. This makes treemap an excellent visualisation technique optimized for huge hierarchical data set. Treemap is developed to address the lack of good visualisation tools for the file system (Johnson & Shneiderman 1991).

Treemap algorithm recursively enclosed tree nodes in rectangular areas with the size of the area depending on the number of child nodes to be partitioned or the weight given to a particular node attributes such as file size for the file system. The whole process is repeated until all nodes are processed. The hierarchical structures of the data are implicitly built into the treemap by nesting the child nodes inside the parent. (Johnson & Shneiderman 1991, p. 286; Liang et al. 2013, p. 80). The processing time for treemap is $O(n)$ with the assumption of the data is pre-processed (Johnson & Shneiderman 1991, p. 289). It is more efficient than standard force-directed graph at $O(|n|^2)$.

Divide and Conquer Treemap and its advantages and disadvantages.

Liang et al. (2013;2015) proposed an enhanced version of the treemap called Divide and Conquer Treemaps to address the requirement of visualisation in the shape other than rectangular to mimic natural shapes. D&C Treemap built on the treemap paradigm by adopting containment which encloses child nodes in the parent node and encodes value using area in a different shaped container. It allows flexibility in the shape of both the container and the containment (Liang et al. 2015, p. 106). The procedures to construct D&C Treemap are (Liang et al. 2015, p. 107):

1. Weight calculation: Every node's weight values is calculated. The weight is defined as a value associated with the property of a node, for example, file size.
2. Area partitioning: Within the container, this procedure recursively partitions the display area into subareas based on the hierarchical nodes. The subarea's size is proportionate to the node's weight.
3. Node positioning: The positions of all the nodes are computed within their local regions. The position of the vertex is usually allocated at the center of its local region.
4. Attribute assignment: at this last stage, graphical attributes, such as size, label, colour, shapes are assigned. Visual cues are also added to enhance and clarify the presentation.

D&C Treemap algorithm (Liang et al. 2015, p. 109-114) will first separate a set of data at the same level into two sets with similar weights. The algorithm is applied recursively to the subdivision. This algorithm contains three sub-algorithms, initialize the first point, divide and conquer. The property of the polygon determines the first side vertex for the starting point. For a convex polygon, the vertex with the largest angle among the polygon is chosen while for a concave polygon, the first concave angle with the negative value is used. The partitioning process continues with the recursive

structure of the original treemap. The divide algorithm is responsible for dividing the nodes into two set, whose combined weight should be close to half of the total weight, namely, Wg_1 and Wg_2. This is achieved by sorting the nodes by weights in ascending order and assigning them to the first set of data until the weight is, or almost, half of the total weight. The remaining nodes will be assigned to the second set of data. Then, the conquer algorithm constructs the polygon into two polygons with the area of the polygons proportionate to the weights of Wg_1 and Wg_2. The whole process is repeated until the last leaf of the tree. To ensure improve readability, minimum angular constraints at 15 degrees is applied. The D&C Treemap has the flexibility of representing hierarchical data using any shape of the container for the display area, and containment for the child. This could be used to put an emphasis on certain containment area for comparison of value between node. It could enhance the visual cue of the visualisation when applied correctly. The combined advantages of D&C Treemap makes it a great visualisation tool for large data set.

Force-directed graph and its advantages and disadvantages.

Eades (cited in Herman, Melançon & Marshall 2000) was first in proposing the use of force-directed approach. He modeled the nodes and edges of a graph as physical bodies tied with springs modeled with Hooke's law (Eades et al. 2010, p. 6-5). The method has since been revised and improved while keeping to Eades' original principal. The idea is to find a balanced layout of the vertices by minimizing the system energy which repels and attracts the vertices (Eades et al. 2010, p. 6-5; Hu 2005, p. 37). One of those derivatives is the algorithm of Fruchterman and Reigold (cited in Hu 2005, p. 37-38), which models the graphs with a springs system between neighboring vertices pulling them together with repulsive electrical force to push all vertices away from each other.

The standard force-directed graph algorithm is not the best candidate for large graph due to two limiting factors (Hu 2005, p. 38). The first factor is that the physical model usually has many local minimums, which could be improved by using a slow cooling schedule with more iterations of force balancing. The second factor is the computational complexity of the algorithm. According to Hu (2005, p. 38), the computational complexity of Fruchterman and Reigold's algorithm is $O(|V|^2)$ as for any given vertex, the repulsive force from all other vertices needed to be calculated.

Collapsible force layout and its advantages and disadvantages.

Bostock (2016) implemented a force-directed graph with collapsible nodes called collapsible force layout. This implementation uses position Verlet integration to allow simple constraints (Dwyer 2009) with the physical movement simulation based on Thomas Jakobsen algorithm (Jakobsen 2001). It uses a quadtree to accelerate charge interaction with the Barnes-hut approximation. A pseudo-gravity force was introduced to ensure the vertices stay in the visible display area and to avoid the expulsion of disconnected sub-graphs. The links or edges are fixed-distance geometric constraints. The use of Barnes-hut algorithm enables clustering and supernodes, which greatly improves the readability of the graph. Eades and Huang (2000, p. 158) defined clustering as groups of related nodes that are clustered into supernode which allows the user to see a summary of the graph. The use of the Barnes-hut approximation algorithm instead of the standard force-directed graph algorithm

also improved the performance significantly to *O(n log n)* (Hu 2005, p. 46). Barnes and Hut (1986) technique groups together nodes that are sufficiently nearby and recursively divides them into groups and stores them in a quad-tree. The forces between the nodes are then calculated locally and nodes that are nearby are treated as super-nodes which will have the mass of the group's total mass. This technique makes collapsible force layout better at handle big data set in comparison to the standard force-directed graph. However, when it is run with a 14k data set, the processing time is 2 minutes and the graph is too big for the display. Another disadvantage of the tool is the lack of labels for the nodes which becomes a major issue when it is used as file system analysis tool. The collapsible capability helps to improves readability by allowing information hiding.

To address these issues, the Collapsible force layout was enhanced to include the display of the file name and file size together with the change in the node size and color when the mouse is hovering over the node. Cascading Style Sheet (CSS) is used to ensure the graph can be displayed in its totality within the display area, which has been optimized for 1400x900 display. The nodes are also colored to indicate the file type. Zooming capability for the whole graph is added to facilitate the possible use of the tool on a larger display screen.

File System visualisation, analysis and findings.

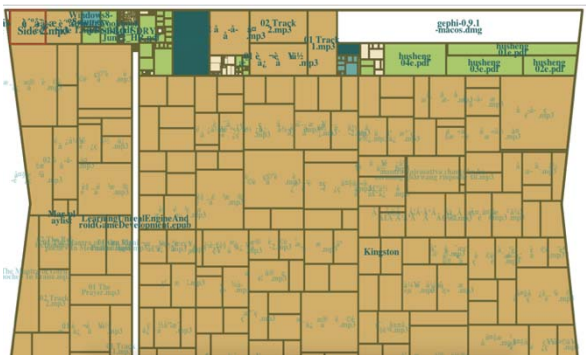

Figure 1. D&C Treemap visualisation for data set with 1,000

File system visualisation was carried out using D&C Treemap and Collapsible force layout. Three sets of data, with 1.8k, 6k and 14k nodes, were used.

The first data set was run with D&C Treemap, figure 1; and, Collapsible force layout, figure 2. D&C Treemap outperformance Collapsible force layout in the time taken to display the result. This is expected as Collapsible force layout

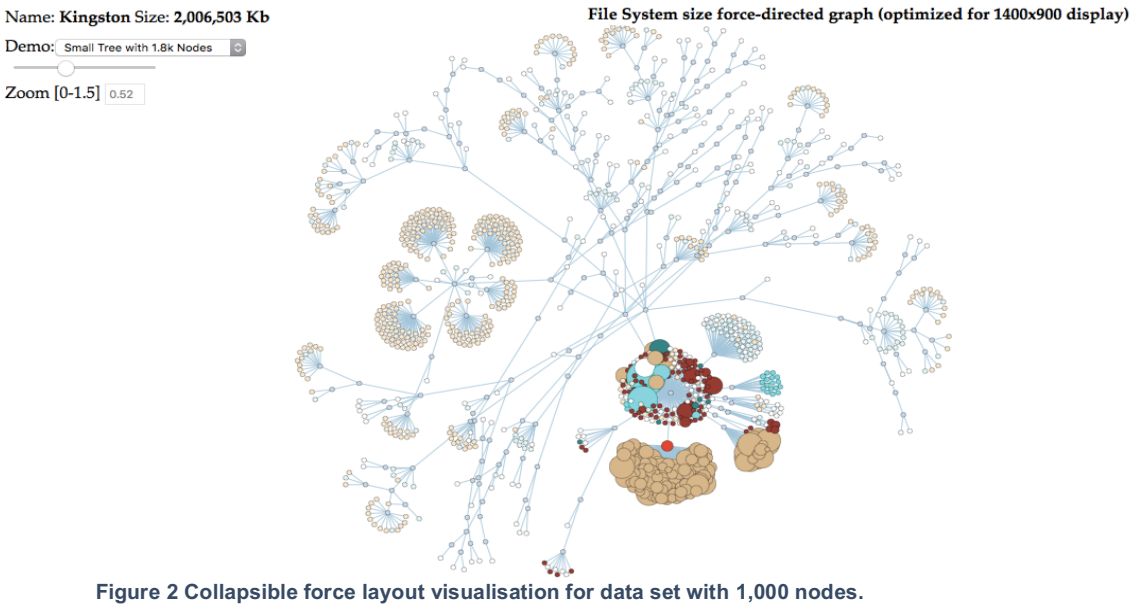

Figure 2 Collapsible force layout visualisation for data set with 1,000 nodes.

algorithm has a higher computation processing time than D&C Treemap. Collapsible force layout took approximate13s to stabilize the drawing completely while D&C Treemap completed it in less than 1s. The platform of the software should be taken into consideration as the D&C Treemap program is compiled into java byte code while the Collapsible force layout code was written in JavaScript which is interpreted at runtime by the browser. Both visualisations offer good indication of the majority of the file type in the directories and sub-directories through effective use of color. Both figures clearly showed that the majority of the files are multimedia files. In figure 2, 5 and 6, one can notice that most of the sub-directories have similar file type except the clustering at the center of the graph in figure 2, which is the download directory. The Collapsible force layout with clustering and super-node was able to allow a quick discovery of the directory holding those files of interests. D&C Treemap achieved the same with the use of selection level to highlight the boundary of the parent node.

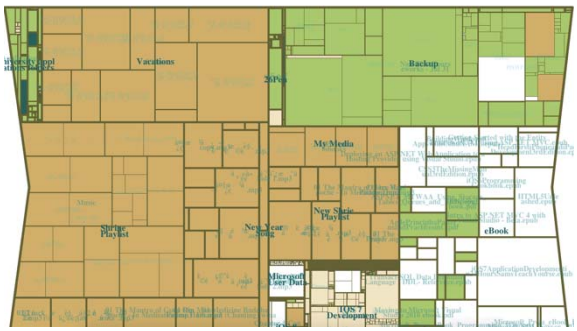

Figure 3 D&C Treemap with 6k nodes.

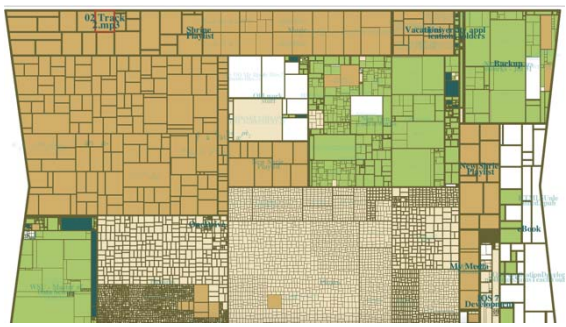

Figure 4 D&C Treemap with 14K nodes.

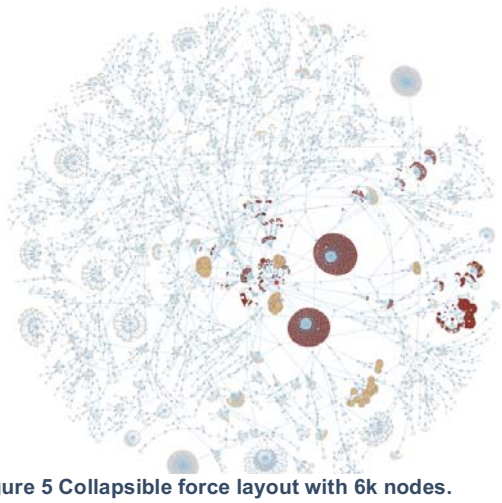

Figure 5 Collapsible force layout with 6k nodes.

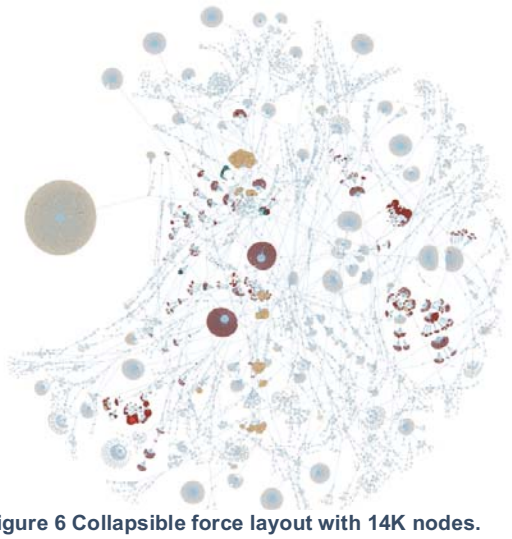

Figure 6 Collapsible force layout with 14K nodes.

D&C Treemap processing time for both 6k and 14k nodes still maintained at less than 1s while the processing time for Collapsible Force layout grew to around 1 minute and 2 minutes respectively. Most of the earlier observation remain for the same for the other 2 data sets as shown in figure 3-6.

Conclusion

The result indicated that D&C Treemap and Collapsible Force layout can be used to display file system hierarchical data. The D&C Treemap has the good processing time. It is an excellent tool for file system visualisation. If the performance of the visualisation is important, D&C Treemap should be

used as Collapsible force layout does not have a good processing time. The user control functionality enhancement on the Collapsible force layout made it more user-friendly. There are so many visualisation methods available, one needs to understand the nature of the data and the capabilities of the visualisation tool to have the best match.

References

Barnes, J & Hut, P 1986, 'A hierarchical O (N log N) force-calculation algorithm', *Nature,* vol. 324, no. 6096, pp. 446-49, DOI:10.1038/324446a0.

Bennett, C, Ryall, J, Spalteholz, L & Gooch, A 2007, 'The aesthetics of graph visualization', *Computational Aesthetics in Graphics, Visualization, and Imaging*, pp. 57-64, viewed 12 April 2016, <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.225.7974&rep=rep1&type=pdf>.

Bostock, M 2016, *Force Layout*, GitHub, viewed 21 April 2016, <https://github.com/mbostock/d3/wiki/Force-Layout>

Di Battista, G, Eades, P, Tamassia, R & Tollis, IG 1994, 'Algorithms for drawing graphs: an annotated bibliography', *Computational Geometry*, vol. 4, no.5, pp.235-82.

Dwyer, T 2009, 'Scalable, versatile and simple constrained graph layout', *Computer Graphics Forum,* vol. 28, no. 3, pp. 991-98, DOI:10.1111/j.1467-8659.2009.01449.x.

Eades, P, Gutwenger, C, Hong, SH & Mutzel, P 2010, 'Graph drawing algorithms', in *Algorithms and theory of computation handbook*, Chapman & Hall/CRC, Boca Ranton, Florida, pp. 21-25.

Eades, P & Huang, ML 2000, 'Navigating clustered graphs using force-directed methods', *Journal of Graph Algorithms and Applications,* vol. 4, no. 3, pp. 157-81.

Herman, I, Melançon, G, Marshall, MS 2000, 'Graph visualization and navigation in information visualization: A survey', *IEEE Transactions on Visualization and Computer Graphics,* vol. 6, no. 1, pp. 24-43, DOI:10.1109/2945.841119.

Hu, Y 2005, 'Efficient, high-quality force-directed graph drawing', *Mathematica Journal,* vol. 10, no. 1, pp. 37-71.

Huang, W, Eades, P, Hong, SH & Lin, CC 2010, 'Improving force-directed graph drawings by making compromises between aesthetics', *2010 IEEE Symposium on Visual Languages and Human-Centric Computing*, IEEE, pp. 176-83, DOI:10.1109/VLHCC.2010.32.

Jakobsen, T 2001, 'Advanced character physics', *Game Developers Conference*, pp. 383-401.

Johnson, B & Shneiderman, B 1991, 'Tree-maps: a space-filling approach to the visualization of hierarchical information structures', *Visualization, 1991. Visualization '91, Proceedings., IEEE Conference on,* IEEE, pp. 284-91, DOI:10.1109/VISUAL.1991.175815.

Kerren, A, Stasko, J, Fekete, JD & North, C (eds.) 2008, *Information visualization: human-centered issues and perspectives (Vol. 4950)*, Springer, Berlin/Heidelberg.

Liang, J, Nguyen, QV, Simoff, S & Huang, ML 2015, 'Divide and conquer treemaps: visualizing large trees with various shapes', *Journal of Visual Languages and Computing*, vol. 31, pp. 104-27, DOI:10.1016/j.jvlc.2015.10.009.

Liang, J, Simoff, S, Nguyen, QV & Huang, ML 2013, 'Visualizing large trees with divide & conquer partition', *Proceedings of the 6th International Symposium on Visual Information Communication and Interaction,* ACM, New York, pp. 79-87, DOI:10.1145/2493102.2493112.

Purchase, HC 2002, 'Metrics for graph drawing aesthetics', *Journal of Visual Languages and Computing*, vol. 13, no. 5, pp. 501-16, DOI:10.1006/jvlc.2002.0232.

von Landesberger, T, Kuijper, A, Schreck, T, Kohlhammer, J, van Wijk, JJ, Fekete, JD et al. 2011, 'Visual analysis of large graphs: state-of-the-art and future research challenges', *Computer Graphics Forum,* vol. 30, no. 6, pp. 1719-49, DOI:10.1111/j.1467-8659.2011.01898.x.

Ware, C, Purchase, H, Colpoys, L & McGill, M 2002, 'Cognitive measurements of graph aesthetics', *Information Visualization,* vol. 1, no. 2, pp. 103-10, DOI: 10.1057/palgrave.ivs.9500013.