Interactive Poster: A Point-Based Layout for Large Hierarchies

Hans-Jörg Schulz*　Steffen Hadlak†　Heidrun Schumann‡

University of Rostock, Germany

ABSTRACT

Space-filling layout techniques for tree representations are frequently used when the available screen space is small or the data set is large. In this paper, we propose a new approach to space-filling tree representations, which uses mechanisms from the point-based rendering paradigm. We also numerically evaluate our new technique together with two established space-filling techniques using the measures of the *Ink-Paper-Ratio* [2] and *overplotted%* [1].

Index Terms: I.3.8 [Computing Methodologies]: Computer Graphics—Applications

1 INTRODUCTION

Large hierarchies occur frequently in real world applications, among which are the life sciences and engineering. Yet, their graphical representation is problematic, as the tree might be heavily unbalanced, extremely wide but relatively shallow, or quite narrow but very deep. The TreeMap [4] and its successors are often used techniques for their representation, because they scale well above all known node-link-representations. This is due to the fact that they are space-filling techniques, meaning that they utilize the available screen space entirely. While many node-link-techniques try to optimize their usage of the available screen space, none of them can be called "space-filling" in the very sense of the word. With our point-based layout technique, we want to close this gap and present a first approach to a space-filling node-link overview visualization for large graphs.

For this, we propose the use of a very specific distribution of the nodes on the available screen space. Its fundamental idea to position nodes in between already existing nodes in an attempt to avoid overlap is taken from the point-based rendering paradigm [3]. An example of how our visualization technique treats an arbitrary tree is shown in Figure 1. It can be seen that our technique still leaves parts of the screen space empty, if the tree is unbalanced or partially very narrow. This is actually a very helpful property of an overview, as it allows to grasp certain tree characteristics (e.g. width, height, balancing) on a glance.

2 THE POINT-BASED LAYOUT TECHNIQUE

In the area of point-based graphics, points instead of triangles are used as fundamental graphics primitives. The challenge is to arrange them in such a manner, that they indeed make up a closed surface with no holes or gaps left in between. A very successful distribution strategy which achieves this is the $\sqrt{5}$-sampling [6]. As a space-filling, explicit tree layout technique would need to solve the very same problem to cover the entire screen space without leaving gaps, our technique uses the $\sqrt{5}$-sampling for positioning the tree's nodes. This sampling technique uses a hierarchical layout scheme for positioning points at unoccupied spots around other points. For each step of this technique, a starting grid will be refined by a rotation of approx. 27 degrees and a reduction of $1/\sqrt{5}$ of the distance between two adjacent grid points. Around every point, four new points will be inserted at the nearest position in the current grid. Figure 2 illustrates the steps of this algorithm. This shows nicely, how the overall density increases with every recursion step and that no gaps are left in between the points. In Figure 2, additional lines where included that are not part of the original $\sqrt{5}$-sampling method. These lines (edges) in between the points (nodes) already hint at a possibility to map a tree structure onto the resulting point positions.

*e-mail: hjschulz@informatik.uni-rostock.de

†e-mail: steffen.hadlak@uni-rostock.de

‡e-mail: schumann@informatik.uni-rostock.de

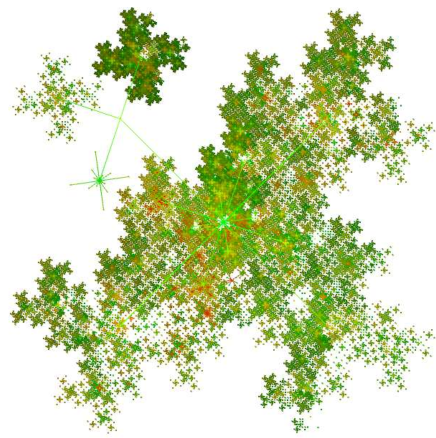

Figure 1: Point-based visualization of an unbalanced tree with 144.740 nodes, of which 134.310 are leaves.

This procedure is repeated until all nodes of the tree are positioned. Since not all siblings of a node can be positioned in the same step and thus on the same level, a green-to-red color scale is used to visualize the number of siblings as an indication of a subtree's width. In the example from Figure 2, this can be observed for the 16 red children of the root node: while the first four of them could be placed very prominently, the following steps can assign only decreasing areas to the remaining children. So, even though the last few of them do not really stick out anymore, it can be seen from the red color assigned to all of them that they have a lot of siblings and that the tree is quite wide at the first level. Also, the level of a node within the tree is mapped to the brightness of this color. This is done for the very same reason as mentioned above: siblings are not necessarily assigned the same amount of screen space, but should still be distinguishable from lower levels. Also, a clever implementation of our layout method includes a reordering of the subtrees by their size, so that larger subtrees are laid out into larger regions. Furthermore, edges can be drawn adaptively only in areas where there is enough space, effectively tying together sparse parts of the visualization without cluttering the dense parts.

So apart from the outer border, the described incremental refinement of the layout allows to use indeed every single pixel of the available space and hence to be space-filling in this sense. Yet, as this layout method predefines all possible node positions independently of the characteristics of the tree itself, the filling degree is tree dependent. This is not considered a limiting factor, though, as the fixed positioning allows for an easy comparison of large trees and the areas that remain empty carry a lot of information about a (sub-)tree's width and balancing – an advantage over other established space-filling techniques, which try to use the available space completely at the cost of this structural information.

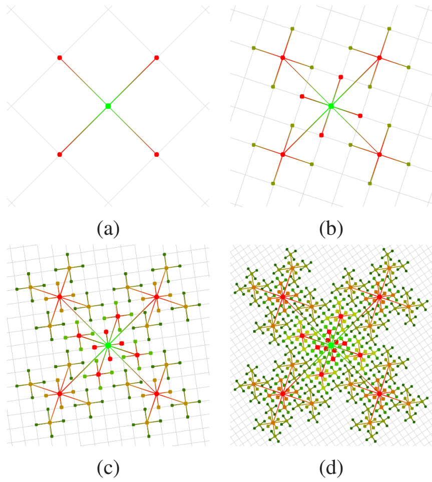

Figure 2: Four recursion steps of the $\sqrt{5}$-sampling method.

3 Evaluation

For the numerical evaluation, we took it as an indicator for the goodness of a layout, how well it utilizes the available screen space and thus minimizes overplotting artifacts. We measured these two properties using the *Ink-Paper-Ratio* [2] and *overplotted%* [1]. Besides our point-based layout, we investigated RINGS [7] and the Space-Optimized Tree Visualization [5] with regard to these measures. These two techniques were chosen, because they are explicit, space-filling techniques, too. To make the different techniques comparable, we fixed the node-size for all techniques to 1 pixel and used a fixed, quadratic screen space of 600×600, which consequently has 360.000 pixels. Since an optimal space-filling layout could draw as much 1-pixel nodes on such a screen-space, we constructed three different full trees with nearly as much nodes. These trees are fully balanced and each non-leaf node has the same amount of children. The chosen width-depth-combinations and their resulting overall sizes are listed in Table 1. The computed *Ink-Paper-Ratio* and *overplotted%* for the different layouts and trees are depicted in Figure 3.

depth	width	no. of nodes	no. of leaves thereof
6	8	299.593	262.144
7	6	335.923	279.963
9	4	349.525	262.144

Table 1: Overall sizes of the three evaluated width-depth-combinations.

For the *Ink-Paper-Ratio*, a higher value means a better usage of the space and is thus preferable. It can be noticed that the *Ink-Paper-Ratio* is steadily rising for the Space Optimized Tree layouts, while it stays about the same for our point-based technique. As for RINGS, in terms of the Ink-Paper-Ratio, it would behave similarly to our layout, if it where not for a special case in the RINGS layout for (sub-)trees with exactly 6 children, which leads to the better ratio for this case. Also, it is not surprising that the Space Optimized Tree layout achieved the best utilization for all of the trees as it is the most adaptable of the three techniques. It tries to maximize the screen usage by distributing the nodes evenly. But this evens out the overall characteristics of the tree and makes it hardly possible to relate the density of one subtree to another. Here, techniques like the point-based layout or RINGS have their strengths as they preserve tree characteristics at the expense of a lower screen utilization, resulting in blank, unoccupied spaces.

For the *overplotted%* values, a lower value means less clutter and is therefore desirable. It can be observed in the diagrams that the *overplotted%* values of our point-based and the RINGS layout are both rising with increasing depth and decreasing width. Interestingly, the *overplotted%* values for the Space Optimized Tree layout are even slightly falling with increasing depth and decreasing width. We believe, this is due to the fact that this technique is very much influenced by the width of the tree, because it partitions the available space according to the number of children. When the number of children is large, it produces many narrow partitions, which makes the layout at the next level even harder. Here again, like in the *Ink-Paper-Ratio* diagram, our point-based technique is mostly sandwiched in between the other two techniques. This overall distribution is related to the *Ink-Paper-Ratio*, since the use of more pixels has of course an effect on the degree of overplotting.

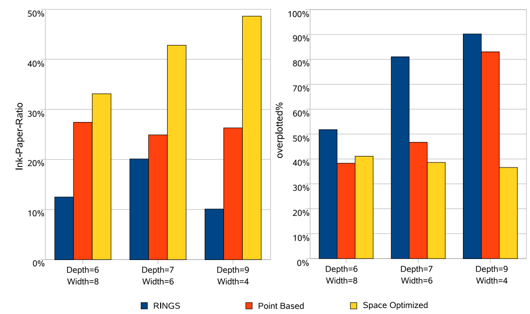

Figure 3: The *Ink-Paper-Ratios* and *overplotted%*-values for the three trees from Table 1.

4 Conclusion

Our point-based layout has a good trade-off between utilizing the screen and preserving the tree characteristics. So, it provides an overview visualization which allows to grasp fundamental tree properties at a glance while still being compact in size. Additional interaction techniques, which are shown in the accompanying video material, make use of the specific layout properties to provide drill-down functionality by zooming and filtering.

Acknowledgements

This work is supported by the DFG graduate school *dIEM oSiRiS*.

References

[1] G. Ellis and A. Dix. The plot, the clutter, the sampling and its lens: Occlusion measures for automatic clutter reduction. In *Proc. of AVI 2006*, pages 266–269, 2006.

[2] A. U. Frank and S. Timpf. Multiple representations for cartographic objects in a multi-scale tree – an intelligent graphical zoom. *Computers and Graphics*, 18(6):823–829, 1994.

[3] M. Gross and H. Pfister, editors. *Point-Based Graphics*. Morgan Kaufmann Publishers, 2007.

[4] B. Johnson and B. Shneiderman. Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proc. of the IEEE Vis 1991*, pages 284–291, 1991.

[5] Q. V. Nguyen and M. L. Huang. A space-optimized tree visualization. In *Proc. of the IEEE InfoVis 2002*, pages 85–92, 2002.

[6] M. Stamminger and G. Drettakis. Interactive sampling and rendering for complex and procedural geometry. In *Proc. of the EG Workshop on Rendering Techniques 2001*, pages 151–162, 2001.

[7] S. T. Teoh and K.-L. Ma. Rings: A technique for visualizing large hierarchies. In *Proc. of Graph Drawing 2002*, pages 268–275, 2002.