The power of interactively linked hierarchy visualizations

J Vis (2022) 25:857–873
https://doi.org/10.1007/s12650-021-00818-3

REGULAR PAPER

Michael Burch · Huub van de Wetering · Nico Klaassen

The power of interactively linked hierarchy visualizations

Received: 24 March 2021 / Revised: 30 August 2021 / Accepted: 17 October 2021 / Published online: 29 January 2022

Abstract This paper describes an interactive web-based tool for visualizing hierarchical data including the recently developed concept of space-reclaiming icicle plots and several more traditional hierarchy visualizations. The tool provides ways to upload, share, explore, and compare hierarchical data using a multitude of different linked hierarchy visualizations. The current version supports up to eight hierarchy visualizations, focusing on user-friendly interactive navigation by a dynamic graphical user interface with a variety of functions while each visualization is interactive by itself. Moreover, color deficiency issues are taken into account in the tool, further improving the usability for this user group. All of the visualizations can be shown in linked views while typical hierarchy parameters and visual variables can be changed on user demand. The interactive tool makes use of OpenGL and Angular, an industry standard JavaScript platform, and runs in a web browser. We illustrate the usefulness of the visualization tool by applying it to the NCBI taxonomy that consists of more than 300,000 hierarchically organized species while filtering for the tetrapoda subhierarchy as an application example. To further test the usability of the tool we did a performance test with increasing sizes of processed hierarchy datasets and even support a GPU implementation. Also user feedback on the eight visualizations was collected. Finally, we explain implementation details and discuss limitations and scalability issues of the linked visualization techniques.

Keywords Hierarchy visualization · Multiple coordinated views · Web-based visualization · Interaction techniques · GPU support

1 Introduction

Visual exploration of hierarchical data is important since it brings benefits for many application fields that deal with hierarchical organizations, like software engineering, bioinformatics, social networking, eye movement data, or architecture. If the number of elements goes into the thousands, a visual depiction of the data is required in order to rapidly understand the structures and to allow interactions with the visual output. Moreover, hierarchical representations are typically used as special interaction media to aggregate, filter, or explore data on different levels of hierarchical granularity, for example, if additional data are attached to a hierarchy like in software systems or in hierarchical clusterings.

Hierarchy visualization is an old, but still active research field in information visualization (Jürgensmann and Schulz 2010; Schulz 2011), and hence, there are at least four major visual metaphors following the principles of nesting, stacking, indentation, or explicit linking. Certain visualization candidates like treemaps (Shneiderman 1992), layered icicles (Kruskal and Landwehr 1983; Wetering et al. 2020), indented

M. Burch (✉) · H. van de Wetering · N. Klaassen
Eindhoven University of Technology, Eindhoven, The Netherlands
E-mail: michael.burch@fhgr.ch

plots (Burch et al. 2010), or node-link diagrams (Battista et al. 1999; Eades 1992) all have benefits and drawbacks focusing on certain data exploration perspectives. For example, treemaps are a space-filling approach using the available display space efficiently, but negatively, the hierarchical structure (branching and depth) is unclear when it comes to larger data. On the other hand, node-link diagrams clearly show the hierarchical structure, but due to the exponentially growing hierarchies, those typically waste display space in the deeper hierarchy levels if shown in a perceptually effective top-to-bottom layout (Burch et al. 2011).

In this paper we introduce linked hierarchy visualizations to not leave alone the observer with only one perspective, but instead to provide the opportunity to take, depending on the data properties, the best from several views to gain insights in the data. We link the hierarchy views by simple coordination principles highlighting the interactive modifications not only in one view to also explore the interesting visual patterns from another perspective. To reach this goal we developed a web-based tool (https://ngl.cornelissen.pw/) that is easy to use as a visual interface (see Fig. 1) by adapting and changing certain parameters. To this end we support eight visualization techniques, but not all of them have to be selected. The user has the free choice of using as many as desired, by also deciding in which order to position them on the monitor.

To illustrate the usefulness of the interactive tool we apply it to a subhierarchy in an NCBI dataset (Sayers et al. 2009) consisting of more than 300,000 hierarchically organized species. We show typical interaction techniques to inspect the data on several hierarchical granularities. Moreover, we discuss scalability issues and limitations, also in the light of implementation details and educational purposes.

This article is an extension of a paper that was originally published at the international symposium on visual information communication and interaction (VINCI) (Burch et al. 2020). We focus on several improvements in the original work which are summarized in the following list:

- Apart from just linking the hierarchy visualizations, we now support many more interactions in each individual hierarchy visualization which greatly improves the user experience in the tool (see Sect. 3.3.)
- We added many more additional features to improve the user experience, in particular, to get better visual results that can easily be exported as an image file for further use. Moreover, it is possible to restart the tool at the latest configuration, in case an exploration process has to be ended due to some reason (see Sect. 3.4.)
- To provide a certain solution for people suffering from color deficiencies, we provide options to switch the color coding based on those visual disorders (see Sect. 6.2.)
- We did a performance test with hierarchy datasets of increasing sizes to find out the limitations of our tool, in particular, for the loading times and the interaction techniques. Moreover, to improve the

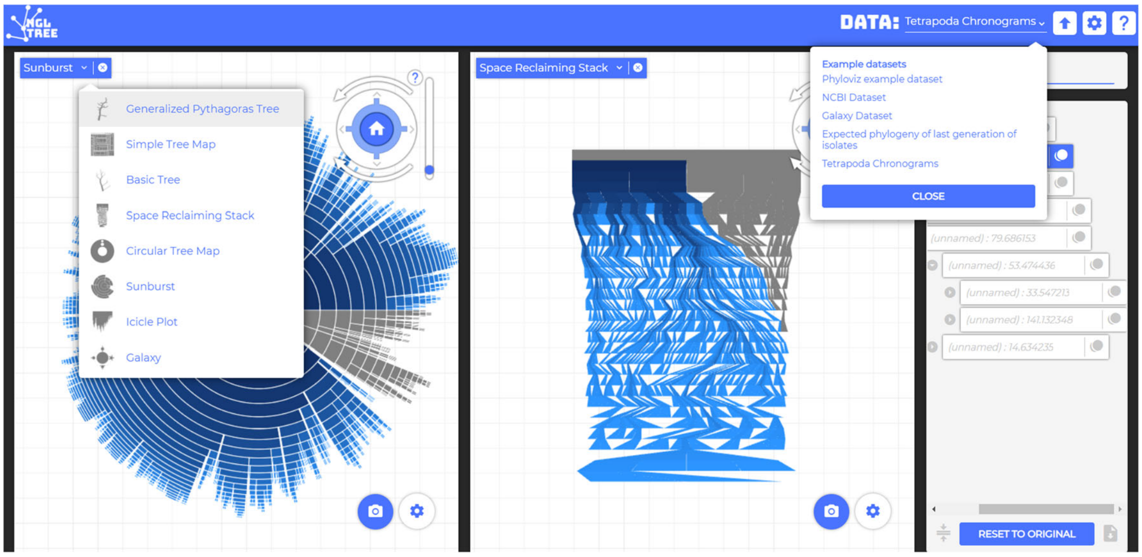


Fig. 1 Two visualizations of the same dataset: A sunburst visualization (left view) and a space-reclaiming icicle plot (right view). The visualizations are linked showing the same selected subhierarchy in blue color and can be exchanged by six more visualization candidates. The web-based visualization tool supports data analysts to upload, share, compare, and explore their hierarchy data in different views

performance we support now also a GPU implementation and only render the nodes not smaller than pixel size (see Section 6.4).
- We recruited several participants to give feedback to all of the eight hierarchy visualizations in order to ascertain the most aesthetically appealing one (see Sect. 6.5.)

2 Related work

Hierarchy visualizations have been developed for many years and an extensive survey is given by Jürgensmann and Schulz (2010); Schulz (2011). All of the existing visualizations for this kind of data typically focus on space efficiency or on depicting the hierarchical structure of the data in a clear and comprehensible manner. However, there is a trade-off in both criteria which causes space-efficient hierarchy visualizations to provide unclear views on the structure and vice versa, structure-preserving visualizations provide a less space-efficient visualization. Moreover, in many hierarchy visualizations we can identify the challenge of selecting individual nodes, in particular, in deeper substructures which is not an ideal situation if it comes to the application of interaction techniques.

For example, treemaps (Shneiderman 1992) and most of their variants (Wijk and Wetering 1999; Balzer et al. 2005; Nocaj and Brandes 2012) provide a space-efficient view on the data, but negatively, the hierarchical structure is less clear than, for example, in corresponding node-link representations (Battista et al. 1999; Eades 1992) and their variations like bubble layouts (Grivet et al. 2004; Lin and Yen 2007). However, node-link diagrams typically have visual scalability problems due to the fact that hierarchies grow exponentially while subtrees in a top-down layout (Burch et al. 2011) get less space to be represented. Indented diagrams (Burch et al. 2010) on the other hand focus on space efficiency, but they have the drawback of not clearly depicting the hierarchical structure, nor is it possible to show symmetries in the data that easily. Another negative aspect is that data elements are visually depicted by tiny graphical primitives making it hard to select them.

There have also been user studies to judge the usefulness of tree diagrams (Burch et al. 2011, 2013) while the research by McGuffin and Robert (2010) analyzed the space efficiencies of different hierarchy visualizations and their benefits and drawbacks.

Layered icicle plots (Kruskal and Landwehr 1983; Wetering et al. 2020) are also beneficial since they clearly show the hierarchical structure (if read from top to bottom), provide a representative element for every node, inner node as well as leaf node, but negatively, do not use the display space efficiently. Icicle plots have been extended a lot, for example, by using radial versions (Andrews and Heidegger 1998; Stasko and Zhang 2000; Yang et al. 2003). Also based on the concept of stacking, generalized Pythagoras trees (Beck et al. 2014) produce aesthetically appealing visualizations that are structurally clear, but suffer from overplotting, later enhanced by a force-directed overlap-free variant (Munz et al. 2019).

There have been attempts to focus on this trade-off between space efficiency and structural clarity (Hlawatsch et al. 2014). For example, in the hyperbolic browser (Lamping et al. 1995; Lamping and Rao 1996), a focus and context technique, hyperbolic geometry is used for visualizing large hierarchies. More display space is assigned to portions of the hierarchy while those are embedded in the context of the entire hierarchy. However, this is built on node-link diagrams and not in a top-to-bottom reading strategy making them challenging to understand from a structural perspective. The Extended Grid Embedding (Youn and Singh 1988) is an old but useful approach in which it is tried to use screen space efficiently, but also in this technique trees are not readable from top to bottom and hence, difficult to interpret. There are various other hierarchy visualization techniques focusing on making the diagrams more scalable and space-efficient, but mostly they do not follow an easy-to-interpret reading direction (Burch et al. 2011), and hence, a linked combination of them is a powerful concept.

Also the indented pixel tree browser (Burch et al. 2011) focuses on using space more efficiently for indented plots by letting the user interact with the diagram. However, it is still not possible to see the entire tree in one view and, additionally, provide extra complementary hierarchy visualizations in a multiple coordinated view (Roberts 2003).

Some approaches even rely on 3D, for example, cone trees (Carrière and Kazman 1995) or botanical trees (Kleiberg et al. 2001). But although they typically show the hierarchies in a more natural and aesthetic way, they can also suffer from occlusion problems, requiring to rotate the 3D visualization. With our web-based visualization tool we try to bring together eight hierarchy visualizations and interactively link them, allowing the user to inspect the hierarchy data from several perspectives. In particular, the space-reclaiming icicle plots have several benefits. They use the same visual metaphor in every hierarchy branch (unlike hybrid representations Nguyen and Huang 2005; Zhao et al. 2005). Moreover, we provide a visualization with which space efficiency and structural clarity can be focused on at the same time based on user-defined interactive parameters. Additionally, the hierarchy is read from top to bottom and it is possible to see all hierarchy elements without overlaps while they are still large enough to interact with.

Several comparative user evaluations have been conducted in the past with the goal to figure out which benefits and drawbacks hierarchy visualizations have, based on certain user tasks. A famous example of such a study is the one by Barlow and Neville (2001) who made two experiments including organization charts (which are similar to top-to-bottom node-link diagrams), icicle plots, treemaps, and the tree rings. Their results show that the participants preferred the organization charts as well as the icicle plots, a result that is different from our aesthetics study in which the node-link diagrams and icicle plots were rated on the lower end of the scale. In the study by Kobsa (2004) the treemaps seemed to be the best hierarchy visualization technique for the given tasks while they outperformed techniques like SequoiaView, Beam Trees, Star Trees, the Tree Viewer, and the Windows Explorer. In our aesthetics study we could find the treemap visualization on a third place of the eight given ones. Maybe it is a good idea to use treemaps since they seem to support certain hierarchy tasks and have some kind of aesthetic appealing. Wang et al. (2006) evaluated three tree visualizations, namely RINGS, treemaps, and the Windows explorer. Also in their results the treemap technique led to good user performances for many of the given tasks. Song et al. (2010) experimented with variants of node-link diagrams like traditional, list, and multi-column approaches while the multi-column was liked by the users. However, their study showed that node-link hierarchy visualizations are readable in general. Based on the findings in these studies and our own aesthetics study, we can argue that the chosen eight hierarchy visualization techniques are useful candidates for including in our web-based tool; however, each typically focuses on one or two of the relevant criteria like space efficiency, structural clarity, and aesthetics. There is no unique solution that supports all of those three criteria at once.

As a compromise between space efficiency, structural clarity, and aesthetics we have chosen four visualization candidates that follow the criterion of space efficiency (Fig. 3b, c, e, and h) while the other four are intended to provide users with aesthetically pleasing overviews of hierarchical data (Fig. 3a, d, f, and g). Additionally focusing on structural clarity is also important (Fig. 3a, d, and h); hence, a mixture of hierarchy visualizations can be interactively chosen following the users' tasks and intentions when exploring the data.

3 Hierarchy visualization tool

We designed a web-based hierarchy visualization tool providing eight different linked visualization techniques with the goal to show the data in several perspectives.

3.1 Tool design decisions

Figure 1 shows a screenshot of our tool with certain visual features or elements, for which we will give a brief description in the following.

(1) *Visualization area:* Here, the selected visualization will be shown for the uploaded dataset. The user is able to move around in this window in the form of panning, zooming, and rotating. In the viewcube a simple interaction can be applied that enables users to perform different kinds of interactions. A viewcube is a means to allow direct rotations and resizing in each view in the tool. The settings button allows users to change parameters based on which visualization is shown while the screenshot button provides a way to download a screenshot of the currently shown visualization.

(2) *Tree navigator:* Here, a simple visualization is shown complementary to the major view and as extra interaction support. Subtrees are offered as an option to visualize only a part of the subtree by defining that subtree as a root node. The reference to the original tree is kept which can be restored by clicking the restore button. From within the tree navigator it is possible to select a node, after which the subtree, rooted at that node, is highlighted in all views. When visualizing a subtree, the tool offers a functionality that exports the opened subtree in the Newick format.

(3) *Additional buttons:* By pressing the upload button the user is able to select and upload a dataset to visualize. Each file that the user uploads is saved in the browser in order to enable the user to quickly access those files in future sessions. Furthermore, to enable the user to get started quickly with the tool

we added a few useful hierarchical datasets. The user can also decide about which visualization to use for the uploaded dataset. Moreover, a tour can be started that explains each of the elements given on the webpage. Several aspects of the website can be changed on user's demand. It currently contains various options regarding the user interface and colors.

Figure 2 depicts the webpage in its start configuration. We have anticipated on several scenarios regarding the user's experience. Firstly, it is possible that the users have never worked with our tool before, and therefore a quick tour highlighting all features would benefit them. Secondly, one could find oneself in a situation where he or she quickly wants to visualize own data, which is what our tool was designed for. Lastly, if someone quickly wants to see the capabilities of the tool, we added a functionality that provides users with ready-to-use data. To show the user the flow in the application, we added a helper arrow, to point at the next step in the process of visualizing data with our application.

3.2 Visualization techniques for comparison

The tool currently supports eight different hierarchy visualization techniques (see Fig. 3). We will describe them briefly and compare the seven of them with the most novel one among them, i.e., the space-reclaiming icicle plots. Some visualizations are subject to some limitations that the space-reclaiming icicle plots are able to solve on the cost of bringing new challenges into play.

3.2.1 Generalized Pythagoras tree (Fig. 3a)

One of the reasons we chose the generalized Pythagoras tree (Beck et al. 2014; Munz et al. 2019) is the clarity that it provides for hierarchical data. From a generalized Pythagoras tree it is easy to see the relation from one node to another one since those are also based on a stacking principle like in the space-reclaiming icicle plots. The drawback of the Pythagoras trees is that a lot of overdraw and occlusion can occur, in case the tree grows large and deep which got improved by Munz et al. (2019). Moreover, due to the splitting effect of angles into smaller angles summing up to 180 degrees, the visual representatives for the deeper hierarchy levels soon reach pixel size and are not recognizable and selectable any more. This size problem is mitigated in the space-reclaiming icicle plots by reclaiming horizontal space.

3.2.2 Treemap (Fig. 3 b)

Treemaps are considered as useful hierarchy representations since they use the display space efficiently and allow the visual encoding of additional quantitative metrics that sum up meaningfully if one climbs up the hierarchy. Color coding is often used as additional indicator of a categorical information like in visualizing file systems where the file types might be worth investigating together with the file and directory sizes. However, from a hierarchical clarity perspective, treemaps are not that useful while they also typically do not easily depict inner nodes of a hierarchy. Nodes are placed on top of each other and cannot be selected easily which is solved in a better way in the space-reclaiming icicle plots. Moreover, in treemaps there is no clear reading direction comparable to the top-to-bottom reading strategy for space-reclaiming icicle plots.

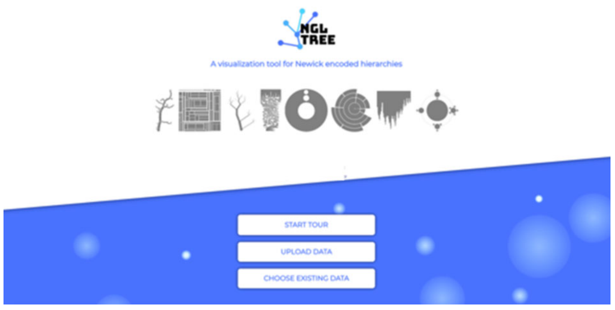


Fig. 2 Start configuration of the tool's webpage

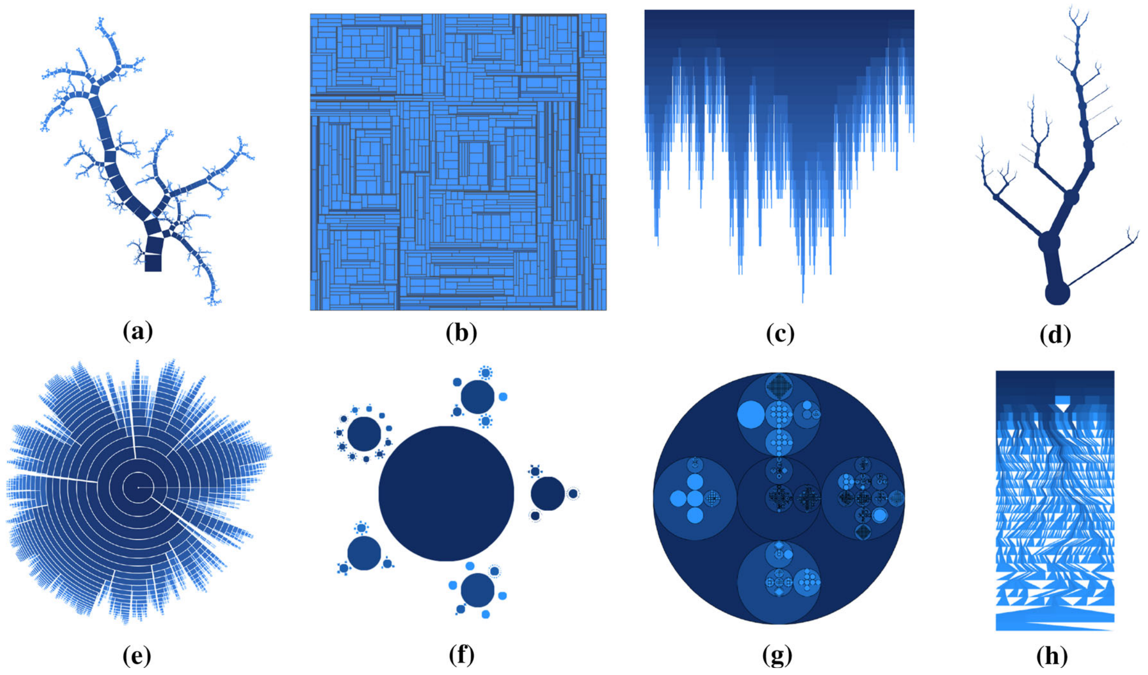


Fig. 3 Several hierarchy visualizations of the NCBI taxonomy with more than 300,000 hierarchically organized species provided by the tool: **a** A Generalized Pythagoras tree. **b** A treemap. **c** An icicle plot. **d** A basic node-link tree. **e** A sunburst. **f** A galaxy plot. **g** A circular treemap. **h** A space-reclaiming icicle plot

3.2.3 Icicle plot (Fig. 3 c)

Icicle plots follow the principle of stacking and provide a clear overlapping-free representation of hierarchical data. Moreover, each node is visually represented and selectable. However, in deeper hierarchy levels a lot of display space is wasted which leads to the fact that hierarchy nodes located deeper in the tree can hardly be selected since they become one-pixel-wide rectangular shapes, i.e., vertical lines. This is especially a problem for deeper hierarchy levels where the chances are higher to waste display width. This problem is remedied by the design of the space-reclaiming icicle plots.

3.2.4 Basic node-link tree (Fig. 3d)

A node-link diagram in a top-to-bottom or bottom-to-top layout is a good way to represent hierarchical data. The direct links easily indicate the parent-child relationships and do not produce overlaps and occlusions. However, if the hierarchy becomes deeply structured, node-link diagrams waste a lot of display space. Unlike icicle plots, they waste display space in the lower as well as in the deeper hierarchy levels. Node-link diagrams could also be adapted to the space-reclaiming strategy, but the links cannot be followed that easily as in space-reclaiming icicle plots where the parent-child relationships are visually modeled by color-coded band-like geometric shapes and by juxtaposed rectangles.

3.2.5 Sunburst (Fig. 3 e)

There have been attempts to radialize a hierarchy visualization, i.e., let the hierarchy grow from the circle center towards the circle circumference. This has the benefit that more layout space is gained for deeper hierarchy levels. But trees may grow exponentially with their depth and the circle circumference is just $2\pi r$ making the layout space to place hierarchy elements only linearly in the depth. Hence, the benefit of increasing layout space is limited. Moreover, radial tree diagrams are somewhat difficult to interpret since the reading direction is not from top-to-bottom but from inside-to-outside (like in a sunburst visualization) which makes it difficult to compare subhierarchy levels.

3.2.6 Galaxy plot (Fig. 3 f)

The galaxy plot can be regarded as a way to place subhierarchies like satellites around a planet, i.e., around their parent node. It follows similar principles like in bubble or balloon layouts for node-link diagrams. This variant of a hierarchy visualization is derived from the Sierpinski Carpet Fractal, but the squares have been replaced by circles to allow for more than eight child nodes. Although this generates a nicely looking visualization, it has many drawbacks concerning readability and interpretability. Moreover, visual scalability is restricted a lot. However, nodes can be selected easily and the hierarchical structure is understandable.

3.2.7 Circular treemap (Fig. 3 g)

To get rid of the rectangular geometrical shapes occurring in standard treemap visualizations, so-called circular treemaps or bubblemaps are useful since they can better provide an overview about the hierarchical structure. But this effect comes at the cost of wasting display space required to display deeply nested substructures. However, circle sizes can still be used to visually encode a quantitative metric and color coding can show an extra categorical attribute, but visual scalability is the major issue for this approach. No space can be reclaimed in deeper hierarchy levels.

3.2.8 Space-reclaiming icicle plots (Fig. 3 h)

The novel idea of a space-reclaiming icicle plot (Wetering et al. 2020) tries to overcome lots of the aforementioned drawbacks like overlaps and occlusion, no clear top-to-bottom reading direction, and no representative graphical elements for each hierarchy element. Moreover, interacting with the diagram is still possible even in deeper hierarchy levels due to the fact that space is reclaimed causing smaller elements to grow larger. For example, by adapting the reclaiming parameter we could find a suitable visual configuration for the hierarchy dataset at hand.

The space-reclaiming icicle plots bring new challenges into play; for example, it is unclear how much horizontal space should be reclaimed and how large the gaps between the neighbored nodes should be chosen in order to create a readable tree diagram. Furthermore, it is not clear what the best hierarchy traversal is at the moment. Different orders of handling the child nodes result in different tree diagrams with more or less meandering. This *meandering effect* in space-reclaiming icicle plots is caused by long links crossing the display with the goal to reclaim the space (see, for example, Fig. 4). It seems best to place the

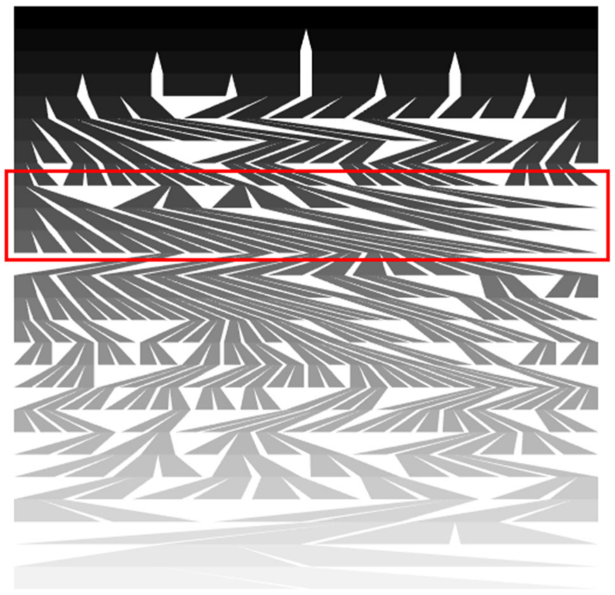


Fig. 4 The skewness of links impacts the meandering effect. The strength of this effect should be reduced by a layout algorithm focusing on this aesthetic criterion in order to make the space-reclaiming icicle plots more readable

largest and deepest subtrees always in the center, causing a more balanced and symmetrically looking diagram, i.e., a more aesthetically pleasing representation.

3.3 General interaction techniques

In our original hierarchy visualization tool we already had plenty of interaction techniques integrated, but we mostly focused on the novel technique of the space-reclaiming icicle plots Wetering et al. (2020). In the extended version of this line of research we experimented with various other interaction techniques, in particular, taking into account each hierarchy visualization separately. In the following we are going to describe several major interaction techniques that are now available in the tool and for which we think that they are useful for exploring hierarchical data.

- *Expand and collapse:* Since each hierarchy visualization technique has a different geometry we do not include the expand/collapse interaction feature directly in the visual representation of each hierarchy, but instead, support this in an additional hierarchy view, similar to the Microsoft File Explorer where also label information can be read in a horizontal reading direction.
- *Hierarchy margins:* For each visualization we allow to add margins between the hierarchy levels (see Fig. 5 for an example in a sunburst visualization). Moreover, margins are also allowed between sibling nodes.
- *Hierarchy visualization rotation:* Each hierarchy visualization can be rotated on users' demand. This helps to bring hierarchy elements to a certain side on which they can be better compared with other views, or they can be rotated due to reasons of an appropriate screenshot for dissemination purposes (see Fig. 6 for a rotated treemap visualization).
- *Color coding for each visualization:* Apart from the standard color palettes we now allow each hierarchy visualization to be modified by a suitable color coding (see Fig. 7 for different color coding of the

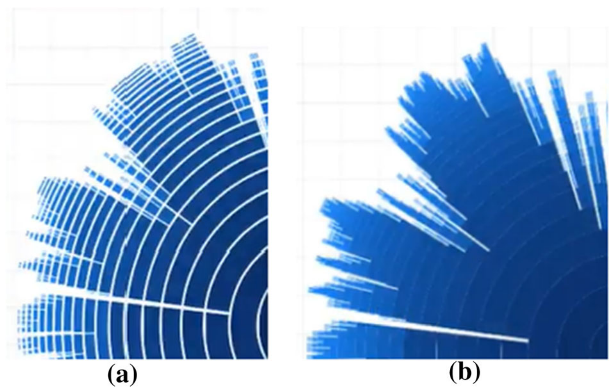


Fig. 5 Modifying the margin between hierarchy levels: **a** With margin. **b** Without margin

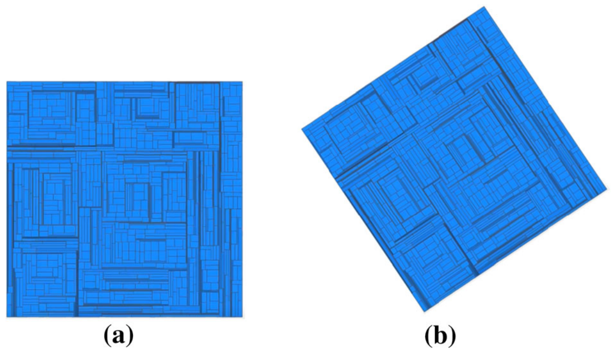


Fig. 6 Rotating a treemap can help to put hierarchy elements to different locations while preserving the layout: **a** Standard treemap representation. **b** The same treemap rotated a few degrees

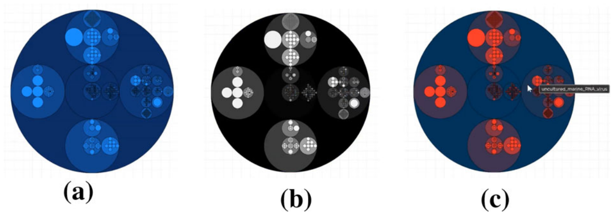


Fig. 7 Three different color coding for the circular treemap. Also labels can be shown

circular treemap visualization). Furthermore, we support reverse palette colors, gradients per subtree, gradient types, as well as an inverted HSV gradient.

- *Zoom in and out:* Zooming in and out is supported, for example, to explore subhierarchies. Moreover, we provide geometric as well as semantic zooming as a way to differ between the pure geometrical representation and the one filled with extra finer-granular information that cannot be shown in the fully zoomed out overview for a hierarchical dataset (see Fig. 8 for a zoomed in and color coded subhierarchy in the circular treemap). The user can also pan and zoom automatically to a searched node in a hierarchy.
- *Background mode:* For a better user experience we added an option to switch between dark and bright background mode. Moreover, we show grid lines to better orientate in the diagram, we added tabs, as well as better side-by-side views.
- *Node search and automatic smooth panning:* If a user searches for a specific node this can be done by a textual search as well as by adding additional information on the node, for example its approximate depth in the hierarchy or its subhierarchy it belongs to. The user is then smoothly guided to the node if it is found.
- *Tree export:* Each hierarchy, either complete or parts of it, can be exported as a text file in a Newick format or as an image file. The user can load such a file in the tool and the tool's internal state is set back to the incorporated file parameters expressing the hierarchy values included.
- *Details-on-demand* (Fig. 9 (a)): Additional information is displayed when a node is hovered or clicked. The related information is shown as a tooltip, e.g., a textual label to the interesting hierarchy node can be requested. This details-on-demand feature is difficult in treemap visualizations or Pythagoras trees where overlaps occur. Moreover, it is challenging to apply in hierarchy visualizations that show graphical elements in pixel size.
- *Width/height change:* Depending on the available display space, the user can decide to change width or height parameters interactively, resulting in the hierarchy visualization scaling accordingly in horizontal and vertical direction.
- *Reclaimed space:* If too much space is wasted and the diagram is not space-efficient anymore the user can adapt the space-reclaiming parameter interactively between values of 0 and 100 percent.
- *Gap size:* If neighbored hierarchy elements look similar due to a similar color coding, they can be separated by a horizontal gap. The size of the gap can be adjusted by the user, but should not be too large

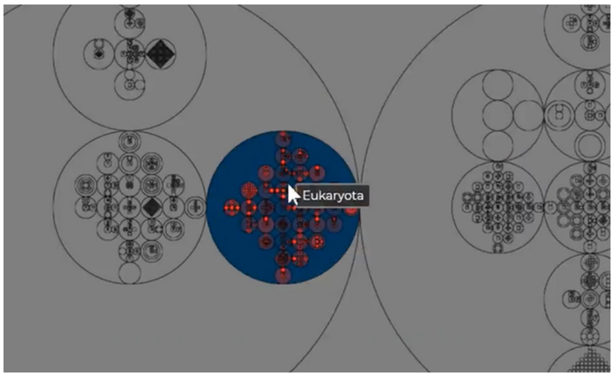


Fig. 8 Zooming is possible in any of the hierarchy visualizations. A geometrical as well as a semantic zooming is supported showing extra details about the subhierarchy elements

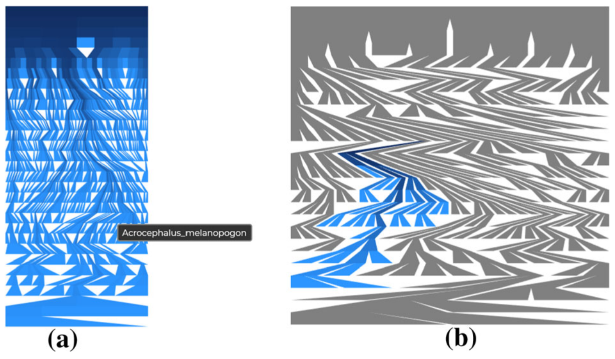


Fig. 9 Interactions: **a** Hovering or clicking an element at a deep level in the hierarchy is possible and gives additional information about the element in the form of a tooltip. **b** Selecting a node in a space-reclaiming icicle plot and highlighting the subtree to the deepest tree level

in order to create a space-efficient diagram. However, larger gaps may improve the readability, but tend to increase meandering.

- *Node appearance and path tracking* (Fig. 9 (b)): The color or outlines of nodes can be changed on demand. This is useful if the path to a node needs to be tracked or just highlighted.
- *Alignment of reclaiming:* Choosing the alignment for the reclaiming location has an impact on the visual output. In our default setting we always distribute the used space around the horizontal center of the visualization.
- *Linking:* Selecting hierarchy elements in any of the hierarchy visualizations is a useful feature since the selected data can also be highlighted in any of the other simultaneously shown hierarchy visualizations. This linking results in different views and perspectives on the dataset like in multiple coordinated views.
- *Subhierarchy filter:* Selecting a certain subhierarchy in a certain hierarchy visualization leads to a highlight of this hierarchy and can be used as a filter to restrict the dataset to the highlighted hierarchy.
- *Data export:* Selected subhierarchies in a hierarchy visualization can be exported in the well-known Newick format and might be visualized separately for further data exploration, maybe also in a different hierarchy visualization. The same feature works for any supported hierarchy visualization.
- *Image export:* If a certain hierarchy visualization is of particular interest, for example, useful to be shown in a presentation or to be included in a publication, this image can be produced by the export functionality.

It may be noted that this is just a short list of the most important interaction techniques. Figure 9a and b illustrates the impact of some of the major interactions if they are applied to a space-reclaiming icicle plot.

3.4 Improving the user experience

To improve the user experience in our hierarchy visualization tool, we follow the eight golden design rules by Shneiderman et al. (2016). This means we strive for consistencies in the user interface, we support shortcuts, the user gets informative feedback, or we allow undo operations to mention a few.

First of all, we support various color coding as well as an adaptation of the background color to make it useful for different groups of people, for example, when suffering from color deficiency problems. Moreover, users can step back to the latest configuration if a session has ended abruptly or even on purpose. A help tool can be used to get extra information when clicking through the functionality provided by the tool. This is given by textual descriptions that explain the operations in detail. Different layouts are possible, for example manageable by side-by-side views or via tabs. We have both, mouse and keyboard support, for example allowing zooming in and out by keyboard shortcuts. The users can even upload their own hierarchical datasets and share them with others to receive feedback about the data and the found insights. Last but not least, an export functionality with annotated files is powerful for dissemination purposes but also to set back the tool to an original state to start a data exploration process from a certain point.

4 Tetrapoda application example

Since we already showed many features of the space-reclaiming icicle plots as well as additional comparative hierarchy visualizations, we will discuss a specific application of this novel technique in the following.

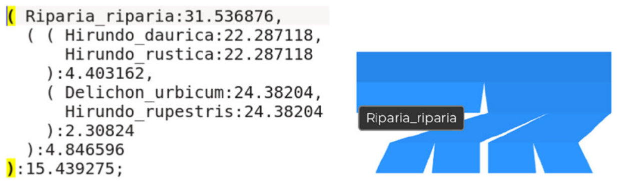
The superclass tetrapoda, meaning four feet, contains the four-limbed vertebrates known as tetrapods. This class describes living and extinct amphibians, reptiles like dinosaurs and thus birds, mammals like primates, and all hominid subgroups like humans, as well as earlier extinct groups. It consists of more than 33,000 species in its entirety and is already structured in a deeply nested hierarchy. The tetrapoda hierarchy is a subset of the NCBI taxonomy dataset (Sayers et al. 2009). Figure 10 shows for a small subhierarchy of the tetrapoda hierarchy, both the Newick format and the space-reclaiming icicle plot.

Figure 11 shows space-reclaiming icicle plots for this hierarchical dataset while the space-reclaiming parameter is interactively changed by the data analyst. The space-reclaiming icicle plot in (a) is pretty useless since the hierarchical structure is not visible while the one in (e) uses the entire horizontal space to unfold the hierarchy. We can easily see that the hierarchy consists of many levels (31) and has a varying branching factor over the hierarchy levels. Moreover, in the deeper hierarchy levels the branching factor is pretty small, i.e., not many species are contained in subhierarchies in deeper levels.

Figure 12 shows three different scenarios in which the data analyst was interested in three different subhierarchies. For this, an inner node has to be selected leading to a highlight of the entire subhierarchy. This interaction feature can be useful to export certain subparts of the dataset to, for example, explore later on in more detail, maybe with an external tool. On the other hand, this data part could also be visualized as a space-reclaiming icicle plot, but then it is separated from the rest of the entire dataset, meaning it can be explored individually or shared with other data analysts using the web-based hierarchy visualization tool.

5 Implementation details

We chose to make use of the Angular framework written in the popular language Javascript/Typescript. After careful consideration, we found out that it would integrate tightly with our goal to easily add multiple visualizations. The framework takes care of a lot of page manipulations that would normally take dozens of lines of code. Angular enriches the developer experience by enforcing a code style that is component-based. Although Angular does add a little bit of overhead vs. plain Javascript, we chose to take this for granted because in comparison with the memory usage and processing power of the visualizations, the overhead is



```
( Riparia_riparia:31.536876,
  ( ( Hirundo_daurica:22.287118,
      Hirundo_rustica:22.287118
    ):4.403162,
    ( Delichon_urbicum:24.38204,
      Hirundo_rupestris:24.38204
    ):2.30824
  ):4.846596
):15.439275;
```

Fig. 10 Small subhierarchy of the tetrapoda dataset in the Newick format (left); icicle plot with tooltip (right)

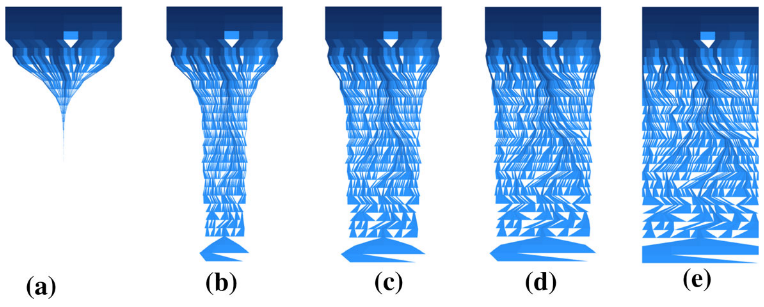


Fig. 11 The space-reclaiming parameter has an impact on the appearance of the hierarchy visualization: **a** 0 percent. **b** 25 percent. **c** 50 percent. **d** 75 percent. **e** 100 percent

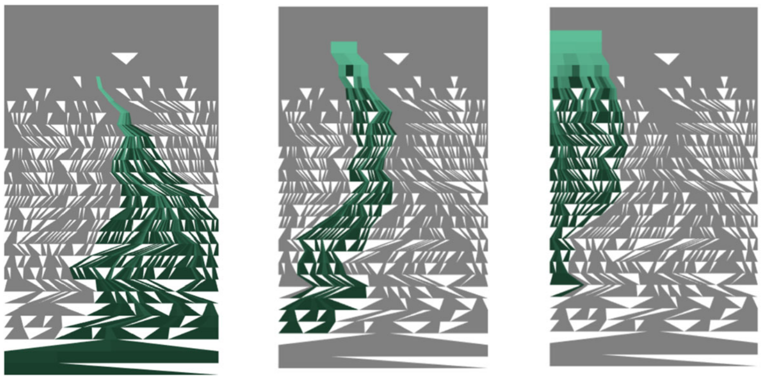


Fig. 12 Selecting subhierarchies starting from a certain inner node helps to identify the entire subhierarchy and how it is contained in the whole tree. Moreover, it can give insights about similarities between the subhierarchies or the dataset can be filtered for this subhierarchy and the smaller dataset can be exported

negligible. We chose to make use of the AOT (Ahead Of Time) option for Angular, meaning lots of dependencies are compiled at build time. Resources like templates are all stored in the Javascript files, rather than fetched via XHR, as was the case in earlier versions of AngularJS.

The main reason we have chosen to use the language Typescript, is because it is an Angular standard. Typescript is a flavor of Javascript, differing in the fact that it adds extra functionality like type checking and classes. At build time, all those extra functionalities are converted to normal Javascript that can run cross-browser. Because Typescript adds type checking, it is less likely that certain kinds of bugs sneak into the code of the application, because Typescript can easily detect mismatching types.

One of the features we thought of during the project are Web Workers. This is a relatively new web technology that enables developers to program multi-threaded. Because Javascript is normally single-threaded, the calculations that are made while calculating the visualization freeze the entire user interface. This means that we cannot show any animated progress indicator. We have thought a long time on whether to use it or not, because it has some drawbacks. One of them is that all code that runs must be self-contained. It cannot access any variables or classes, except the input that is put in, which cannot contain any functionality. While developing a Web Worker solution for the generation of visualization, we ran into this problem multiple times. We solved it by passing over objects that, in our case, represent draw calls.

A lot of aspects of our website have been created over time. There would be little gain in spending a lot of time recreating something like a Newick parser, if we could just as well use a tested solution and spend the time on improving the rest of the website. Therefore, we used libraries where possible and practical, here we will discuss the reasons, benefits, and downsides for using each of the libraries.

We wanted to have a clear way to explain how the website works to a new user. Generally, using a big wall of text explaining each part of the website does not work very well, so we looked further. Initially, we stumbled upon a way of showing an overlay over the website where there would appear text next to each button giving a brief explanation. This was better, but only allowed for a short sentence per button. Finally, we found Intro.js. This is a popular library used by big websites like Amazon and Nestle. We found that this mostly satisfied our wishes. It did not allow for any interaction during the tour though. For example, we wanted to open a visualization in the tour, but Intro.js did not allow that, luckily, though it gave us the tools to implement this ourselves.

We use NewickJS for parsing Newick datasets into a data structure that is easier to work with. This library suits our needs perfectly. We can give it the Newick file, and it creates for us a data structure that our visualizations can efficiently make use of.

For the user it might be annoying to be required to upload the same dataset several times. To resolve this we included a feature where the website stores the dataset in the browser for later usage. There were some challenges in doing this, namely the browser only allows for approximately five megabytes of storage in our target browsers (Chrome, Safari, and Firefox). Large Newick datasets like the NCBI dataset tend to require a large amount of storage, almost seven megabytes for the NCBI dataset. Luckily, the Newick dataset is highly compressible, the NCBI dataset can be compressed down to around two megabytes in about a hundred milliseconds on an average computer.

6 Limitations and improvements

In this section we discuss some limitations of the interactive tool focusing on typical aspects like scalability, color deficiencies, performances, or usability.

6.1 Scalability

The space-reclaiming icicle plots try to find a way to the trade-off between space efficiency and structural clarity for hierarchy visualizations. Although we think this novel concept is quite useful, we are aware of the fact that also novel challenges and drawbacks are introduced.

- *Algorithmic scalability:* The generation of a space-reclaiming icicle plot and the final rendering is fast and linear in the number of nodes. Consequently, interactions for modifying the layout of a space-reclaiming icicle plot are responsive even for large hierarchies, like the tetrapoda dataset. However, if more advanced ordering strategies have to be considered, such a layout might become time-consuming which is also the case for several other hierarchy visualization candidates like squarified treemaps or Voronoi treemaps.
- *Visual scalability:* If a hierarchy grows more in size than in depth or if it is nearly perfectly balanced, there does not remain much space to reclaim. Hence, a standard layered icicle plot might be the better choice to represent such a hierarchical dataset. The space-reclaiming icicle plots work better for unbalanced and deeply structured hierarchies in which much space is lost due to the low depth in many subhierarchies.
- *Perceptual scalability:* When applying color coding to a space-reclaiming icicle plot, e.g., to indicate the hierarchy depth, we are aware that this can have an impact on the readability of a hierarchy visualization. Furthermore, the meandering effect can have a bad influence on the readability of them indicating parent-child relationships leading over several hierarchy layers.
- *Data scalability:* The space-reclaiming icicle plots are built for rooted and finite hierarchies. However, unrooted trees or forests might be visualizable but by many modifications to the standard layout. Forests might be drawn by multiple versions of a space-reclaiming icicle plot, but if comparisons between several trees have to be done this might be enhanced by additional layout algorithms. Moreover, dynamic hierarchies are not supported by the standard approach.

6.2 Color deficiency issue

We circumvent the issue of users suffering from color deficiency problems by providing color scales created for these purposes. For example, each hierarchy visualization can be modified in a way to adapt the color coding, even for individual subhierarchies. Popular color coding can be the tool-specific blue, min-max-red, -green, -blue, -grayscale, vaporwave, malachite, candy, golden blue, neon, purple and orange, and many more.

To reach this goal we incorporate many color coding from color brewer as well from popular research (Silva et al. 2007). Not only the visualizations can be adapted to certain user-defined color scenarios, we even allow the background color to be changed on users' demands. The background color can be important for certain application fields; for example, in the SciVis community a black background is typically used while the InfoVis community mostly prefers a white background. Hence, such a background color adaptation might be useful for presenting the visual results for a specific audience.

6.3 Educational purposes

Actually, the simple hierarchy visualizations that we support in our tool are useful for educational purposes; however, some prior knowledge and experiences have to be learned beforehand. The techniques are useful to showcase small hierarchy datasets and to see a visual pattern from different hierarchical perspectives, but negatively, we cannot expect that the visual depictions are understandable right from the beginning. The tool also does not support the animated generation of an entire hierarchy from its building blocks as a way to step-by-step understand the hierarchy layout algorithm in a visual form. However, a teacher in information visualization might use the tool in a lecture to spot the differences between the different hierarchy views and indicate the algorithmic concepts by visual depictions of the data.

One negative issue is that we only support eight hierarchy visualizations from a large repertoire of possible candidates; hence, the lecturer has to focus on one or several of the provided candidates. There is currently no good way to extend the tool by further techniques including the lecturer or the students which would be a powerful idea for future work. The current version of the tool can only be extended by its developers; however, this is actually a simple task, depending on the fact whether the new hierarchy visualization is clearly specified.

6.4 Performance test and GPU support

The tool has OpenGL GPU support and has been tested and compared for standard laptops and high-end computers. OpenGL can handle the data more efficiently, and hence, a performance improvement is achieved (see Fig. 13).

Moreover, since only the data are updated that are really required we again win performance for our algorithms. This is, in particular, useful for rendering the hierarchy visualizations, i.e., we only render those visual objects that are really visible on screen and only those that have undergone changes affecting the visual output (see Fig. 14 for a large generalized Pythagoras tree). Also only those objects are incorporated in an algorithm that are explicitly affected by an interaction technique. As another add-on we now use specific shaders for rendering visual shapes, i.e., much of the formerly CPU computation is now offloaded to the GPU which again frees the CPU a lot. As another extra, particularly for zooming, the GPU allows a fast, smooth, and continuous zoom into a hierarchy, not like the stuttering effects happening when only the CPU is made use of.

6.5 User feedback

We also requested feedback about our eight hierarchy visualization techniques in an online questionnaire by providing eight visual stimuli showing the different visualizations for the same dataset with similar visual characteristics, i.e., using the same color coding, the same hierarchy expansion level, the same rotation style, and so on.

The participants had to fill out a form about personal details about age, gender, experience with visualizations, and some more. They got a short tutorial about the visualization tool as well as about the hierarchy visualizations implemented. Moreover, to keep the study setup simple, the participants only had to respond to one task of ordering the eight hierarchy visualizations for aesthetic appealing in decreasing order.

The online questionnaire ran for 2 weeks and we collected the hierarchy visualization orderings based on the opinions of 107 participants. As result and summary we had 34 male and 73 female participants whose age ranges varied between 11 and 67 with an average age of 34 years. Eighty-nine participants mentioned that they are not familiar with visualization, 18 said that they are familiar with them.

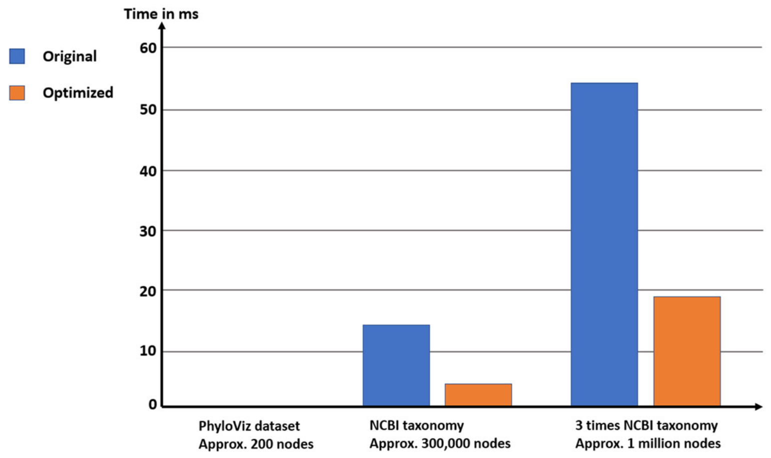


Fig. 13 Letting run the visualization tool in different scenarios, original vs. optimized, with varying hierarchy sizes shows the differences in the runtime in seconds. The datasets under performance exploration are a dataset with approximately 200 nodes (which we call PhyloViz dataset) (0.3 and 0.2 ms) which is much smaller compared to the NCBI taxonomy dataset with 300,000 nodes (14 and 4 ms) and the tripled version of it (55 and 19 ms)

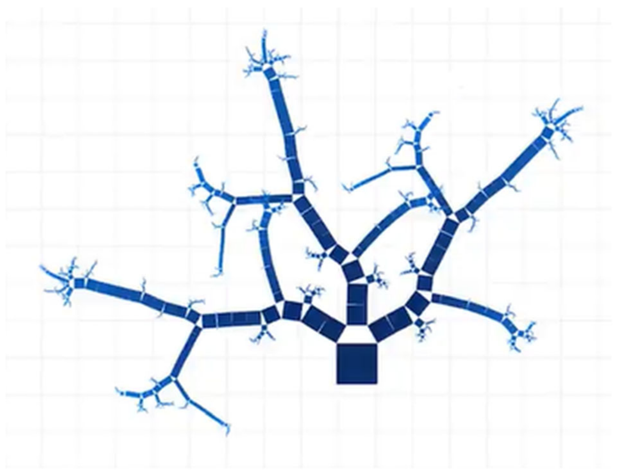


Fig. 14 The visual output for the generalized Pythagoras tree for a large artificially generated hierarchy dataset to check for performance issues

To evaluate the provided orderings we computed the average position each hierarchy visualization was positioned in all orderings which gave as an indication about how aesthetically appealing a hierarchy visualization is. One interesting result was that the generalized Pythagoras trees were voted on the first place in 88 out of 107 voting. Those were followed by the sunburst, the treemap, the basic node-link tree, the standard icicle plot, the space-reclaiming icicle plot, the galaxy plot, and finally, the circular treemap.

We took the opinions and additional feedback by those study participants to further improve some of the hierarchy visualization features. For example, we removed some of the hierarchy element borders, we also took into account aspects like color deficiencies that were mentioned by the participants, and we increased the circle sizes in the galaxy plot, to mention a few.

We also have to mention that the aesthetics study did not provide any insights about data exploration aspects; for example, although the Pythagoras trees seemed to be aesthetically appealing, they might not be good at typical analytical tasks, in particular tasks that have to be solved by biologists working with phylogenetic data. We are aware of the fact that the strength of a hierarchy visualization is not only dependent on the data properties, the user tasks, but even on the application domain. For example, the opentreeoflife.org website provides a hierarchy visualization (based on a node-link metaphor similar to the one researched by Buchheim et al. (2002)) which might be useful for biologists, but actually, just one view is provided and loading hierarchy data from another application domain (like software system data) might not doable with this specific tool. Our intention was to provide several hierarchy views that can be individually selected by the users, independently from the data properties, the tasks, and the application domain, however, providing views for all parameters is a challenging, even impossible task.

7 Conclusion and future work

In this extended paper we described a web-based and interactive visualization tool for hierarchy data supporting eight visualization techniques. We discussed the benefits and drawbacks of the linked visualizations, explained the layout algorithms and adaptable parameters, described typical interactions, applied it to the tetrapoda dataset which is a subset of the NCBI taxonomy, and provided some implementation details. Moreover, we discussed space-reclaiming icicle plots in comparison with the other hierarchy visualizations. Finally, algorithmic, visual, perceptual, and data scalability issues are explained, describing problems that are challenging and that could not be solved with all hierarchy visualizations. As an additional step to find out which hierarchy visualizations are liked and aesthetically appealing we requested feedback from 107 participants in an online experiment. For future work we plan to add additional dataset scenarios like unrooted trees, forests, or dynamic hierarchies, all of them might be worth considering for extending the tool. We also plan to take into account the dataset properties as well as the user tasks for the visual encoding; for example, each technique supports the flexibility for side to side or up and down views, in cases the hierarchy might be deep or even if the task demands for a hierarchical top-to-bottom flow detection. Also further visual variables might be included like categorical information for data properties and the like. Moreover, a comparative user evaluation, maybe with eye tracking, belongs to future work.

References

Andrews K, Heidegger H (1998) Information Slices: visualising and exploring large hierarchies using Cascading, semicircular disks. In: Proceedings of IEEE symposium on information visualization, pp 9–11

Balzer M, Deussen O, Lewerentz C (2005) Voronoi Treemaps for the Visualization of Software Metrics. In: Proceedings of software visualization, pp 165–172

Barlow T, Neville P (2001) A comparison of 2-D visualizations of hierarchies. In: Proceedings of the IEEE symposium on information visualization, INFOVIS, Andrews K, Roth SF, and Wong PC (Eds.). IEEE Computer Society, pp 131–138

Di Battista G, Eades P, Tamassia R, Tollis IG (1999) Graph drawing: algorithms for the visualization of graphs. Prentice-Hall

Beck F, Burch M, Munz T, Di Silvestro L, Weiskopf D (2014) Generalized pythagoras trees: a fractal approach to hierarchy visualization. In: Proceedings of the international joint conference on computer vision, imaging and computer graphics - theory and applications, VISIGRAPP, pp 115–135

Buchheim C, Jünger M, Leipert S (2002) Improving Walker's algorithm to run in linear time. In: Kobourov SG and Goodrich MT (Eds), Proceedings of 10th international symposium on graph drawing, GD (Lecture Notes in Computer Science), Vol 2528. Springer, pp 344–353

Burch M, Andrienko GL, Andrienko NV, Höferlin M, Raschke M, Weiskopf D (2013) Visual task solution strategies in tree diagrams. In: Proceedings of the IEEE pacific visualization symposium. pp 169–176

Burch M, Konevtsova N, Heinrich J, Höferlin M, Weiskopf D (2011) Evaluation of traditional, orthogonal, and radial tree diagrams by an eye tracking study. IEEE Trans Vis Comput Graph 17(12):2440–2448

Burch M, Raschke M, Weiskopf D (2010) Indented pixel tree plots. In: Proceedings of international symposium on visual computing, pp 338–349

Burch M, Schmauder H, Weiskopf D (2011b) Indented pixel tree browser for exploring huge hierarchies. In: Proceedings international symposium on advances in visual computing, ISVC, pp 301–312

Burch M, van de Wetering H, Klaassen N (2020) Multiple linked perspectives on hierarchical data. In: Michael B, Westenberg MA , Nguyen QV, and Zhao Y (Eds.) Proceedings of the 13th international symposium on visual information communication and interaction, VINCI, ACM, 3:1–3:8

Carrière J, Kazman R (1995) Research report: interacting with huge hierarchies: beyond cone trees. In: Proceedings of information visualization, pp 74–81

Eades P (1992) Drawing free trees. Bull Inst Comb Appl 5(1992):10–36

Grivet S, Auber D, Domenger JP, Melançon G (2004) Bubble tree drawing algorithm. In: Proceedings of international conference on computer vision and graphics, pp 633–641

Hlawatsch M, Burch M, Weiskopf D (2014) Bubble hierarchies. In: David M (Ed.). Proceedings of 10th international symposium on computational aesthetics in graphics, visualization, and imaging, CAe@Expressive, ACM, pp 77–80

Jürgensmann S, Schulz H (2010) A visual survey of tree visualization. IEEE Visweek 2010 Posters (2010)

Kleiberg E, van de Wetering H, van Wijk JJ (2001) Botanical visualization of huge hierarchies. In: Proceedings of information visualization, pp 87–94

Kobsa A (2004) User experiments with tree visualization systems. In: Ward MO and Munzner T (Eds.) Proceedings of the 10th IEEE symposium on information visualization, InfoVis, IEEE Computer Society, pp 9–16

Kruskal J, Landwehr J (1983) Icicle plots: better displays for hierarchical clustering. Am Stat 37(2):162–168

Lamping J, Rao R (1996) The hyperbolic browser: a focus + context technique for visualizing large hierarchies. J Vis Lang Comput 7(1):33–55

Lamping J, Rao R, Pirolli P (1995) A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In: Proceedings of the SIGCHI conference on human factors in computing systems, pp 401–408

Lin CC, Yen HC (2007) On balloon drawings of rooted trees. Graph Algorithms Appl 11(2):431–452

McGuffin MJ, Robert JM (2010) Quantifying the space-efficiency of 2D graphical representations of trees. Inf Vis 9(2):115–140

Munz T, Burch M, van Benthem T, Poels Y, Beck F, Weiskopf D (2019) Overlap-free drawing of generalized pythagoras trees for hierarchy visualization. In: Proceedings of IEEE visualization conference, VIS IEEE, pp 251–255

Nguyen QV, Huang ML (2005) EncCon: an approach to constructing interactive visualization of large hierarchical data. Inf Vis 4(1):1–21

Nocaj A, Brandes U (2012) Computing voronoi treemaps: faster, simpler, and resolution-independent. Comput Graph Forum 31(3):855–864

Roberts JC (2003) Guest editor's introduction: special issue on coordinated and multiple views in exploratory visualization. Inf Vis 2(4):199–200

...Sayers EW, Barrett T, Benson DA, Bryant SH, Canese K, Chetvernin V, Church DM, DiCuccio M, Edgar R, Federhen S, Feolo M, Geer LY, Helmberg W, Kapustin Y, Landsman D, Lipman DJ, Madden TL, Maglott DR, Miller V, Mizrachi I, Ostell J, Pruitt KD, Schuler GD, Sequeira E, Sherry ST, Shumway M, Sirotkin K, Souvorov A, Starchenko G, Tatusova TA, Wagner L, Yaschenko E, Ye J (2009) Database resources of the national center for biotechnology information. Nucl Acids Res 37(suppl 1):D5–D15

Schulz HJ (2011) Treevis.net: a tree visualization reference. IEEE Comput Graph Appl 31(6):11–15

Shneiderman B (1992) Tree visualization with tree-maps: 2-D space-filling approach. ACM Trans Graph 11(1):92–99

Shneiderman B, Plaisant C, Cohen M, Jacobs S, Elmqvist N (2016) Designing the user interface - strategies for effective human-computer interaction, 6th Edition. Pearson

Silva SS, Madeira J, Santos BS (2007) There is more to color scales than meets the eye: a review on the use of color in visualization. In: Proceedings of 11th international conference on information visualisation, IV. IEEE Computer Society, pp 943–950

Song H, Kim BH, Lee B, Seo J (2010) A comparative evaluation on tree visualization methods for hierarchical structures with large fan-outs. In: Mynatt ED, Schoner D, Fitzpatrick G, Hudson SE, Edwards WK, and Rodden T (Eds.). Proceedings of the 28th international conference on human factors in computing systems, CHI, ACM, pp 223–232

Stasko JT, Zhang E (2000) Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In: Proceedings of the IEEE symposium on information visualization, pp 57–65

van de Wetering H, Klaassen N, Burch M (2020) Space-reclaiming icicle plots. In: Proceedings of 2020 IEEE pacific visualization symposium, PacificVis. IEEE, pp 121–130

van Wijk JJ, van de Wetering H (1999) Cushion Treemaps: visualization of hierarchical information. In: Proceedings of the IEEE symposium on information visualization, pp 73–78

Wang Y, Teoh ST, Ma KL (2006) Evaluating the effectiveness of tree visualization systems for knowledge discovery. In: BS Santos, T Ertl, and KI Joy (Eds.). Proceedings of 8th joint eurographics - IEEE VGTC symposium on visualization, EuroVis, Eurographics Association, pp 67–74

Yang J, Ward MO, Rundensteiner E A, Patro A (2003) InterRing: a visual interface for navigating and manipulating hierarchies. Inf Vis 2(1):16–30

Youn HY, Singh AD (1988) near optimal embedding of binary tree architecture in VLSI. In: Proceedings of the international conference on distributed computing systems, pp 86–93

Zhao S, McGuffin MJ, Chignell MH (2005) Elastic hierarchies: combining treemaps and node-link diagrams. In: Proceedings of IEEE symposium on information visualization 8