Circular Hierarchy

Description

The *Circular Hierarchy* algorithm produces a circular visualization of the output produced by a multi-level aggregation method. It is used to visualize the community detection done by the aggregation algorithm.

Implementation Details

The algorithm aims to create a circular visualization when provided with hierarchy information for each node. Different communities information can also be provided using the levels. The algorithm is as follows,

1. Collect information about the nodes including their ids, labels, strengths & color values.
2. Then strengths of the nodes are normalized.
3. Edge information is collected next.
4. Hierarchy information is collected from the appropriate columns in the nwb file. Appropriate mapping of Node ID to Level ID are collected.
5. After this angles for each node are calculated using the nodes & level information.
6. The calculated angles are then converted into Cartesian system.
7. These node positions are then reconfigured/normalized based on the radius of the proposed circular visualization.
8. Next level angles are computed followed by its position.
9. Next computations are performed on edges. After normalizing the edge weights, the edge control points are computed. To do this, the ancestor hierarchy is computed & then normalized.
10. Afterwards the edges are sorted so that in-community edges are drawn first.
11. Next edges are sorted from thickest to thinnest & then drawn in that order.
12. Next dividers - separating different communities are calculated for their angles, position.
13. Color values in the form of RGB are computed for each node. If no color information is provided then the default color of Grey (203, 203, 203) is assumed.

All this data is then output into a ps file in appropriate format.

Usage Hints

The algorithm requires a file storing the information about the network. This file should have information about at least one hierarchy level. This contains the information about community to which a particular node belongs to. The hierarchy can have at most 4 levels. The ordering of these hierarchies is also important and can be specified at the time of input.

The other optional information like node color attribute values, node strengths, edge weights can also be provided. In its absence the default values of 1.0 are considered for all of these parameters.

Hierarchical edge bundling is based on the principle of visually bundling adjacency edges together analogous to the way electrical wires and network cables are merged into bundles along their joint paths and fanned out again at the end, in order to make an otherwise tangled web of wires and cables more manageable. This value can also be provided. It must be between 0.0 and 1.0. It's default value is 0.75.

Once these parameters are provided the system performs the computations and a PostScript (.ps) file is generated.

Links

- Source Code: Link

Acknowledgments

The original Round Russell algorithm was authored and later implemented in Python by Russell Duhon. This circular visualization algorithm was then implemented in Java, integrated and documented by Chintan Tank. Also thanks to Russell Duhon for providing the original python source code and helping throughout the implementation. Many thanks to Micah Linnemeier as well for providing guidance during implementation.

References

1. Holten, Danny. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. Link

See Also