Graph, Network, and Tree Visualization

Álvaro Figueira, PhD.
MSc in Data Science - Data Visualization

[dcc] DEPARTAMENTO DE CIÊNCIA DE COMPUTADORES
FACULDADE DE CIÊNCIAS DA UNIVERSIDADE DO PORTO

FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO U.PORTO

285

285

Introduction

- In addition to presenting values associated with data instances, visualization techniques may also represent **relationships:**
  - Part/sub-part, parent/child, or other hierarchical relationship
  - Connectedness, such as cities connected by streets
  - Derivations, such as a sequence of steps or stages
  - Similarities in values/attributes, such as temporal/spatial

- Relationships can be simple or complex
  - Directed/non-directed
  - Weighted/unweighted
  - Certain/uncertain

Relationships may provide more richer information than the contained in the data values.

286

286

Visualizing Trees and Hierarchies

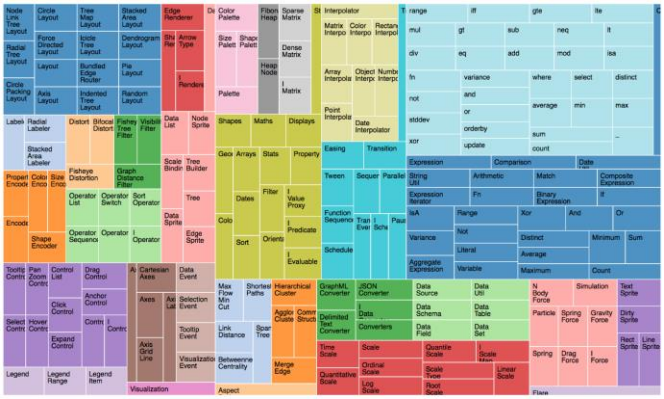
- The most basic are the hierarchical relationships
- We can split the techniques for visualizing trees/hierarchies into
  - Space filling methods
  - Non-space filling methods

Space Filling Methods

- Space filling techniques seek to use as much as possible the available visual space
  - Juxtaposition (side-by-side) is used to represent the connection between data objects.
  - As opposed to, for example, using edges to convey relations.
- The most common approaches are:
  - Rectangular, and
  - Radial representations

Space Filling Methods

- Treemaps is the most popular rectangular representation
  - A rectangle is recursively divided into pieces, alternating horizontal and vertical cuts, based on the sub-trees size at a given level

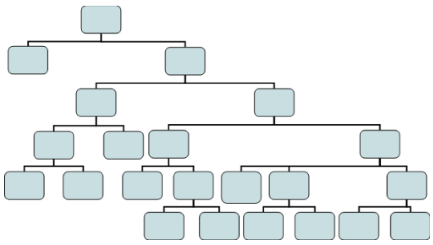


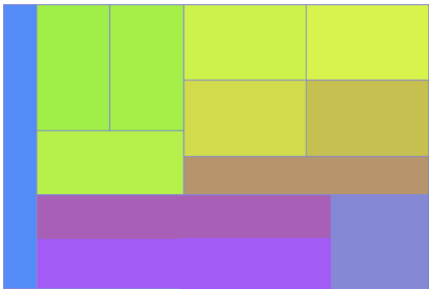
https://bl.ocks.org/mbostock/4063582

289

289

Treemap ("Slice-and-Dice")





290

290

Space-filling Methods

- There are other variations of the basic Treemap method (“slice-and-dice”)
  - Cushion treemap - uses lighting effect to improve hierarchy identification
  - Squarified treemap - reduces long and thin rectangles
  - Nested treemap - emphasizes the hierarchical structure
  - Voronoi treemap - employs Voronoi diagrams instead of rectangles
  - Circular treemap - employs circles within circles

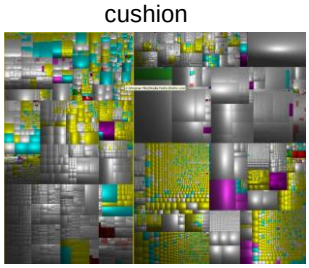
- More Information in
  - http://www.cs.umd.edu/hcil/treemap-history/

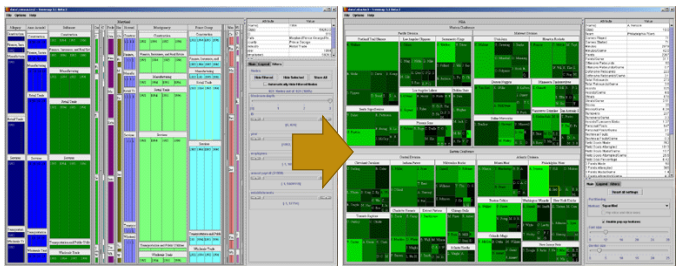
Treemap Examples

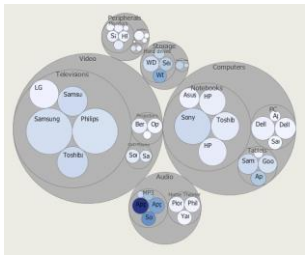
cushion

non-squarified

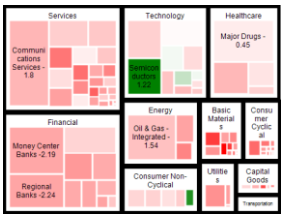
squarified



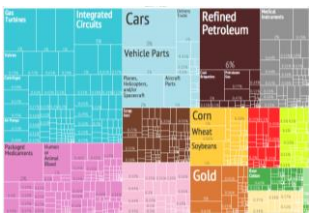




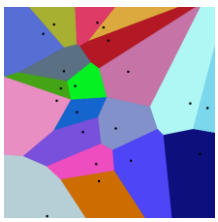
circular



nested

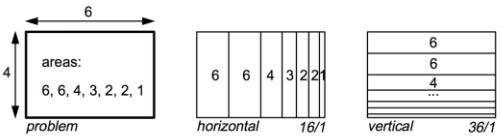


poorly-nested

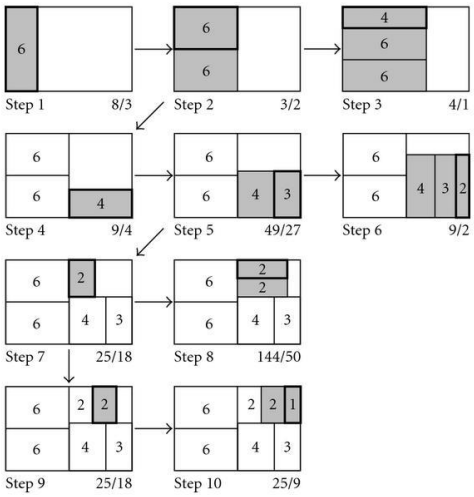


voronoi

Squarified Treemap

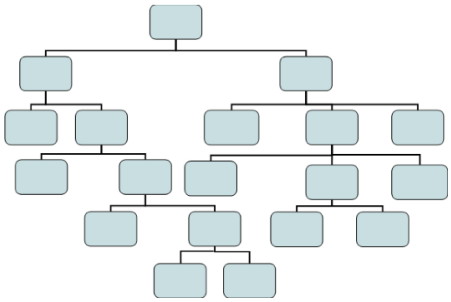


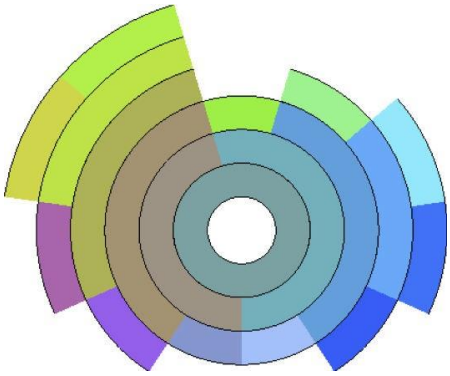
$$AspectRatio = \frac{\max(width, height)}{\min(width, height)} \approx 1$$



Space Filling Methods

- Radial space filling techniques, also known as **sunburst** displays. They place the root of the hierarchy in the center of the display and employ nested rings to represent the hierarchy





https://bl.ocks.org/mbostock/4348373

Space Filling Methods

- Note:
  - Unlike Treemaps, the sunburst technique employs the visual space to model the intermediate (non-terminal) nodes

Non-space Filling Methods

- The most common method for displaying hierarchical relationships is the **node-link diagram**
- In the design of algorithms for drawing diagrams some factors need to be taken into account:
  - Draw conventions - straight edges, polygonal lines or curves; position of nodes in a grid; all siblings in the same vertical position, etc.
  - Restrictions - positioning a particular node in the center of the drawing, positioning groups of nodes close to each other, etc.
  - Aesthetics - several rules that greatly influence the final interpretation (next slide)

Non-space Filling Methods

- Aesthetic rules can be:
  - Minimize cross-lines
  - Keep nice aspect ratio
  - Minimize total drawing area
  - Minimize overall edge size
  - Minimize the number of distinct angles and curvatures
  - Try to create a symmetric structure

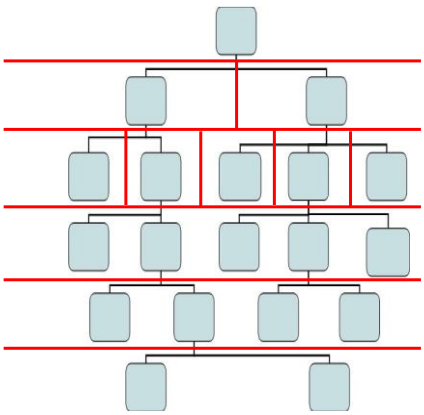
297

297

Non-space Filling Methods

- For **trees** it is relatively easy to respect these rules
- A simple algorithm could be:

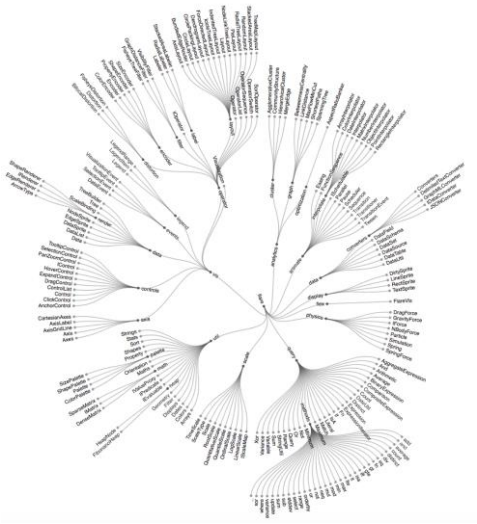
1. Split the drawing area into slices of equal height, considering the height of the tree
2. Divide each slice into rectangles of equal sizes, considering the number of nodes at each level
3. Draw each node in the center of its corresponding rectangle
4. Draw a line between the center-bottom of each node to the center-top of its child nodes

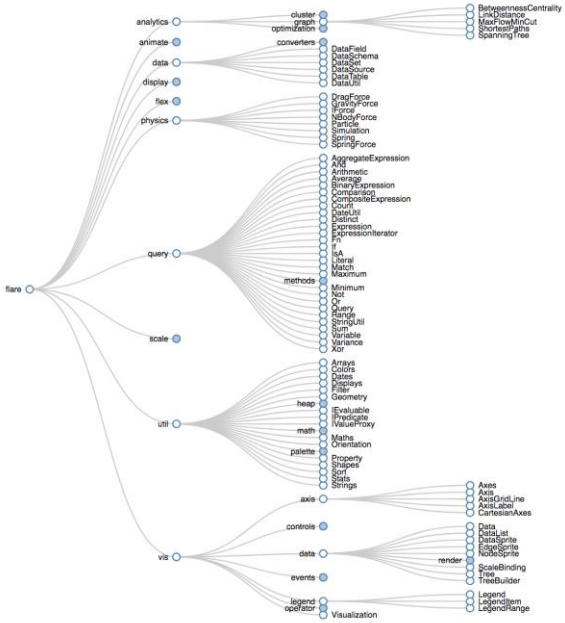


298

298

Some Tree Layouts





https://bl.ocks.org/mbostock/4339083
https://bl.ocks.org/mbostock/4063550

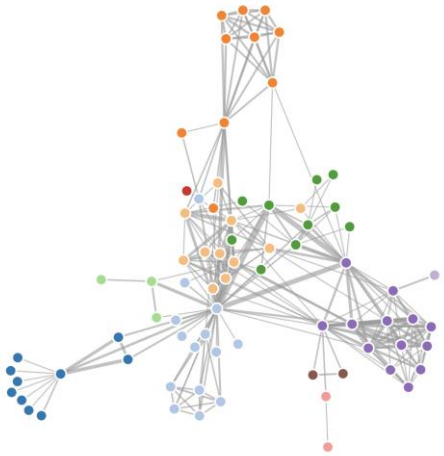
Graphs and Networks

- Trees are a specific type of graphs which are
  - Connected, unweighted and acyclic
- There are specific techniques for **visualizing** general graphs, two examples are
  - Node-link diagrams
  - Matrix representations

Node-Link Force-based Representations

Example: *Les Misérables*

What can we say by looking at the graph?



304

304

Node-Link Force-based Representations

- Force-based methods use a spring analogy to represent the edges, with nodes' positions iteratively changed until a stable state is reached
- There are two forces between nodes
  - f_{ij}: spring (attraction/repulsion) force between connected nodes
  - g_{ij}: repulsion force to avoid neighbor nodes to get too close
- The simplest model uses Hook's laws to represent the force of a spring and the inverse square law to represent the repulsion

305

305

Node-Link Force-based Representations

- If *d(i,j)* is the distance between two nodes, s_{ij} is the size of the spring (at rest), and k_{ij} is the spring tension, the *x*-component of the spring force can be calculated as

$$f_{ij}(x) = (k_{ij} \times (d(i,j) - s_{ij})) \times (x_j - x_i) / d(i,j)$$

- If *rij* is the repulsion tension, the *x*-component of repulsion force will be

$$g_{ij}(x) = (r_{ij} / d(i,j)^2) \times (x_i - x_j) / d(i,j)$$

306

306

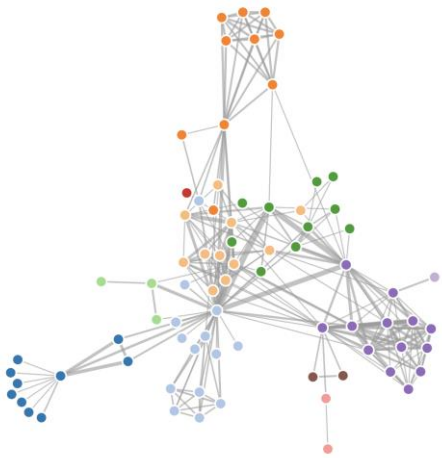
Node-Link Force-based Representations

- Start position can be random
- In each iteration the forces on each node are calculated and their positions are updated
- To avoid oscillations the forces should decrease over time
  - The result can converge to a local minimum

307

307

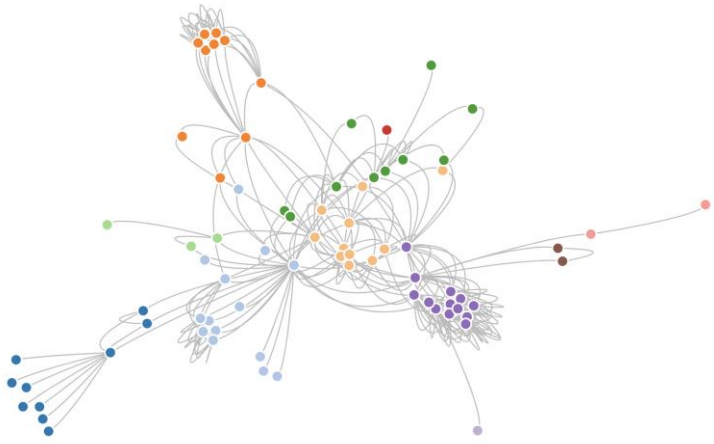
Force-based Method



Try it at:
https://bl.ocks.org/mbostock/4062045

308

Force-based Method (Curved Edges)



https://bl.ocks.org/mbostock/4600693
https://bost.ocks.org/mike/fisheye/

309

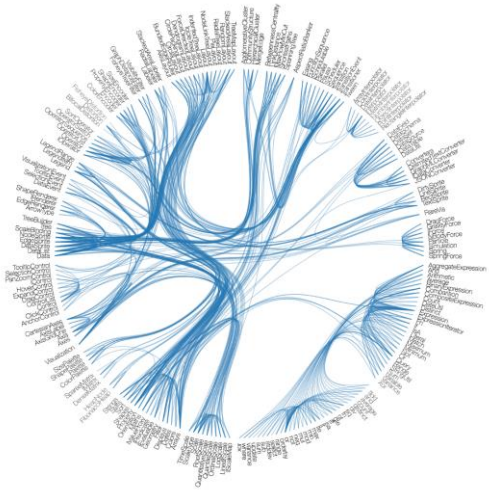
Circular Layouts

http://mbostock.github.io/d3/talk/20111116/bundle.html

310

310

Hierarchical Edge Bundling

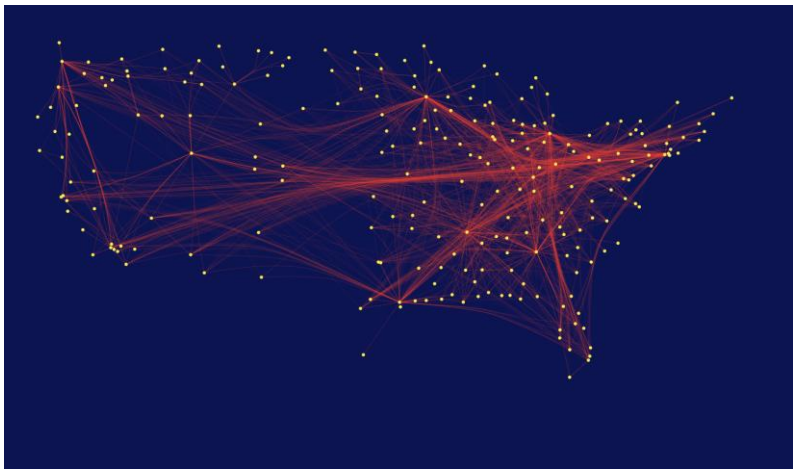


http://mbostock.github.io/d3/talk/20111116/bundle.html

311

311

Force-based Edge Bundling



http://bl.ocks.org/upphiminn/6515478

Matrix Representation

- An alternative representation is the adjacency matrices
  - A grid $n \times n$ where the positions (i,j) represent the existence (or not) of edges between nodes i and j
- This method overcomes the edge-crossing problem, but has issues with scalability (with thousands of nodes)
- Research began to seek for reordering's that could depict structures in the data

Matrix Representation

- There are specific strategies for reordering, from user-centric to automatic (it is a NP-complete problem)

	a	b	c	d	e	f	g	h
a		●	●			●		
b	●			●		●		
c	●				●		●	●
d	●	●				●		
e			●				●	●
f	●	●		●				
g			●		●			●
h			●		●		●	

(...)

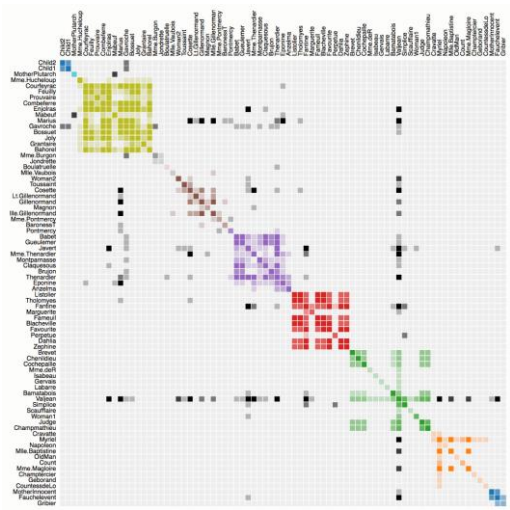
	p	q	r	s	t	u	v	w
p		●	●	●				
q	●		●	●				
r	●	●		●				
s	●	●	●		●			
t				●		●	●	●
u					●		●	●
v					●	●		●
w					●	●	●	

314

314

Matrix Representation

reorder



https://bost.ocks.org/mike/miserables/

315

315

Reference

- Ward, M., Grinstein, G. G., Keim, D. **Interactive data visualization foundations, techniques, and applications**. Natick, Mass., A K Peters, 2010.

320