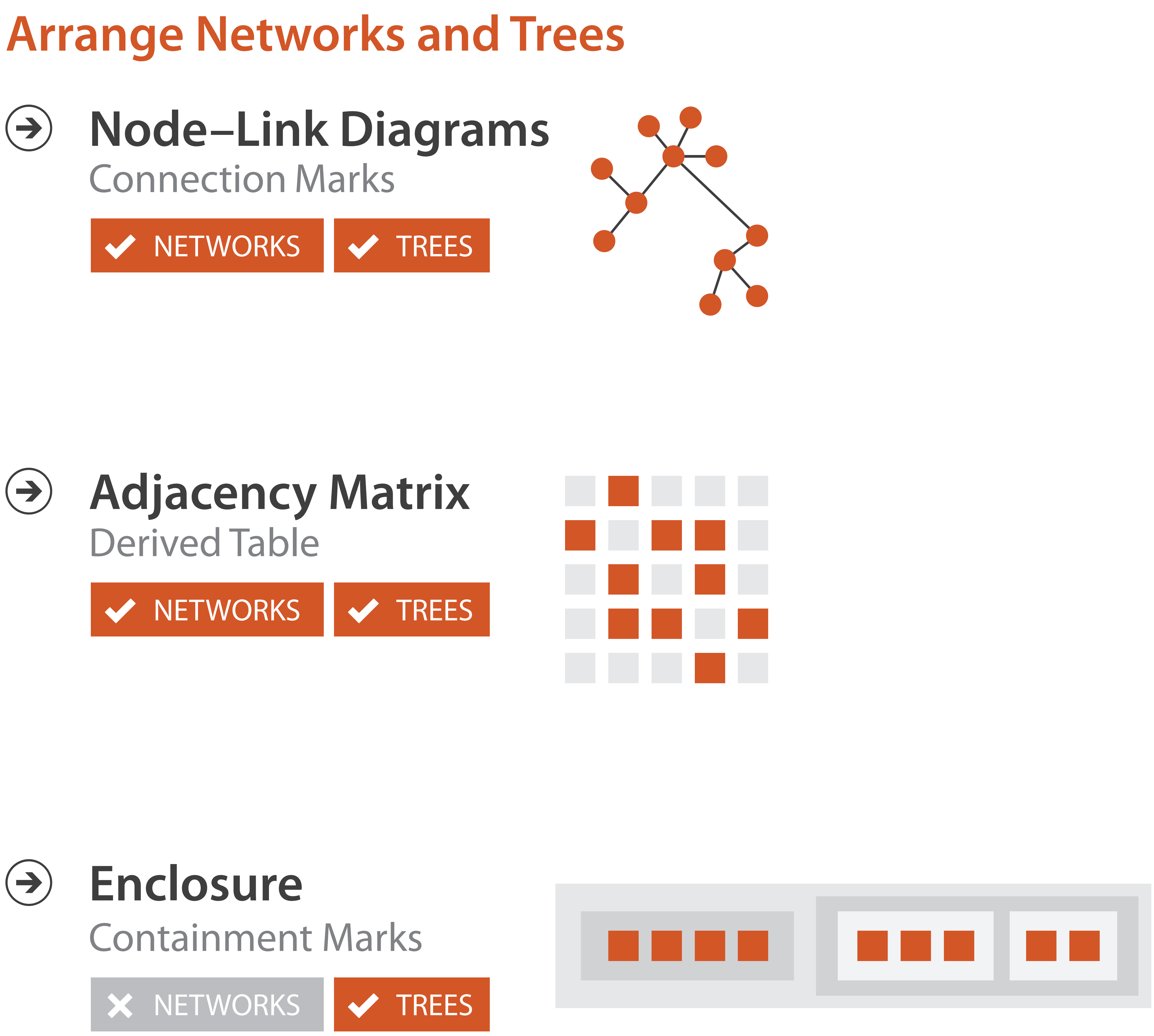
Data Visualization (CSCI 627/490)

Trees & Sets

Dr. David Koop

Networks

- Network: nodes and edges connecting the nodes
- Formally, G = (V,E) is a set of nodes V and a set of edges E where each edge connects two nodes.
- Nodes == items, edges connect items
- **Both** nodes and edges may have **attributes**



[Munzner (ill. Maguire), 2014]

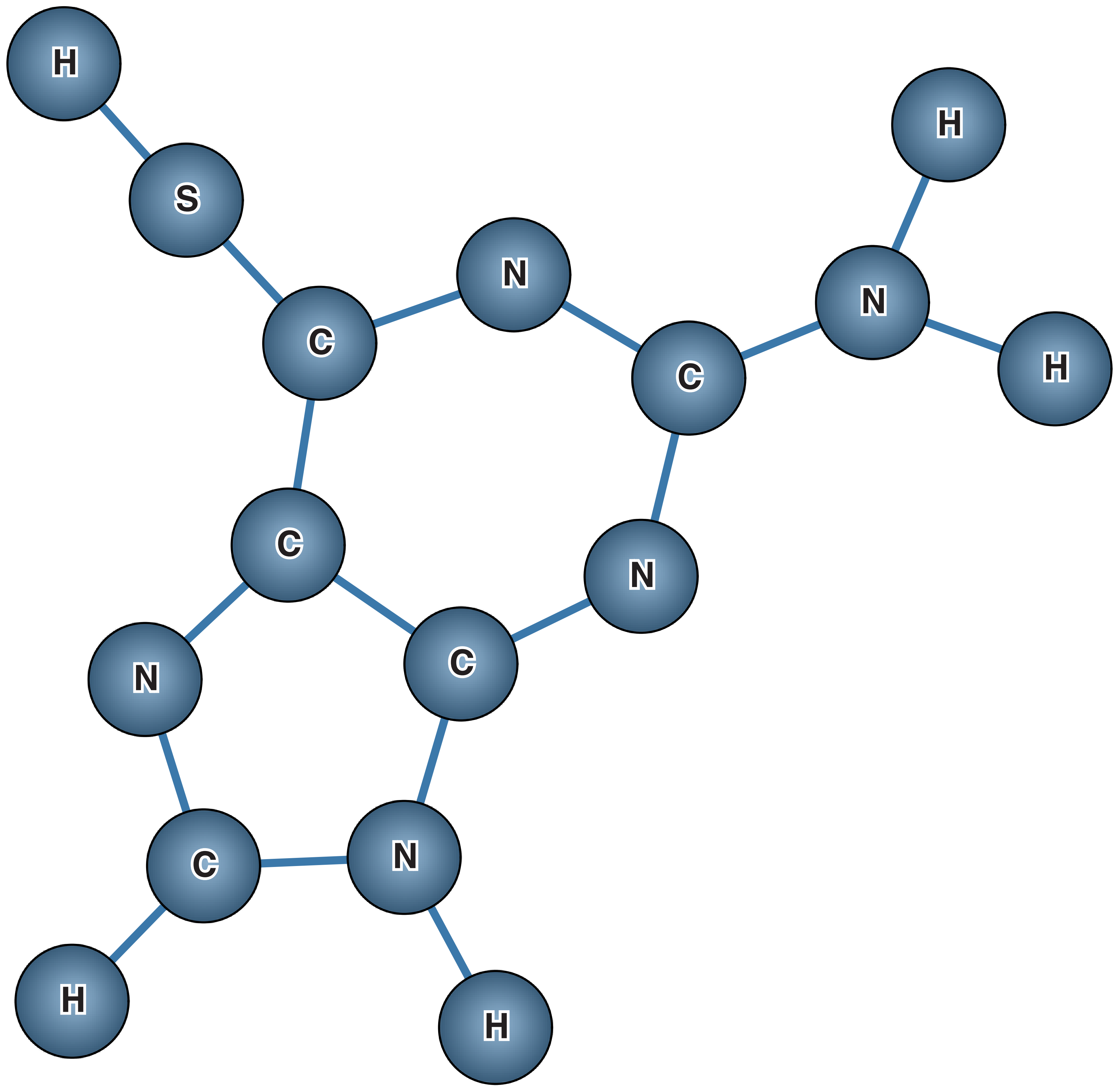
Network Data Represented in Tables

Nodes

ID	Atom	Electrons	Protons
0	N	7	7
1	C	6	6
2	S	16	16
3	C	6	6
4	N	7	7

Edges

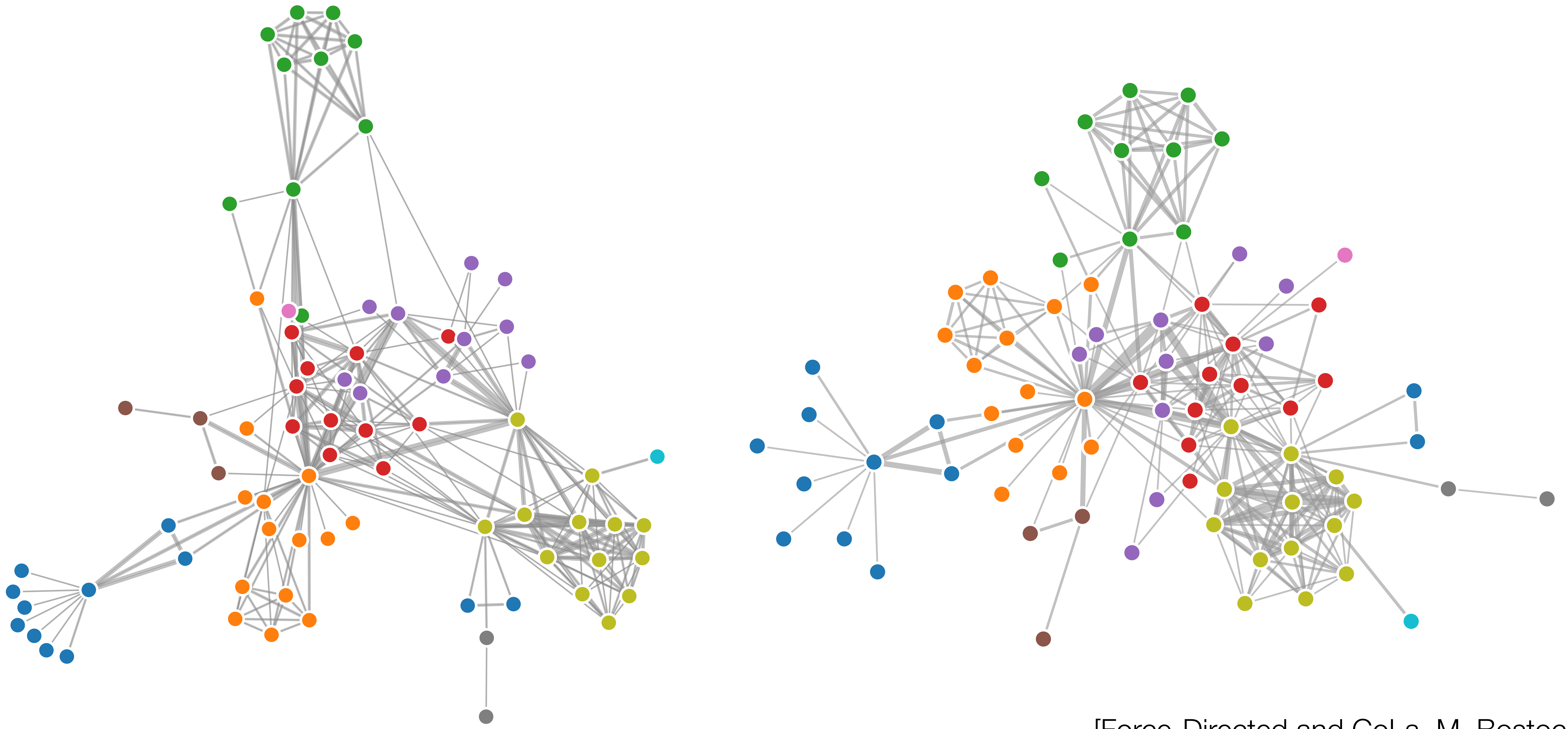
ID1	ID2	Bonds
0	1	1
1	2	1
1	3	2
3	4	1



Networks Need Layouts!

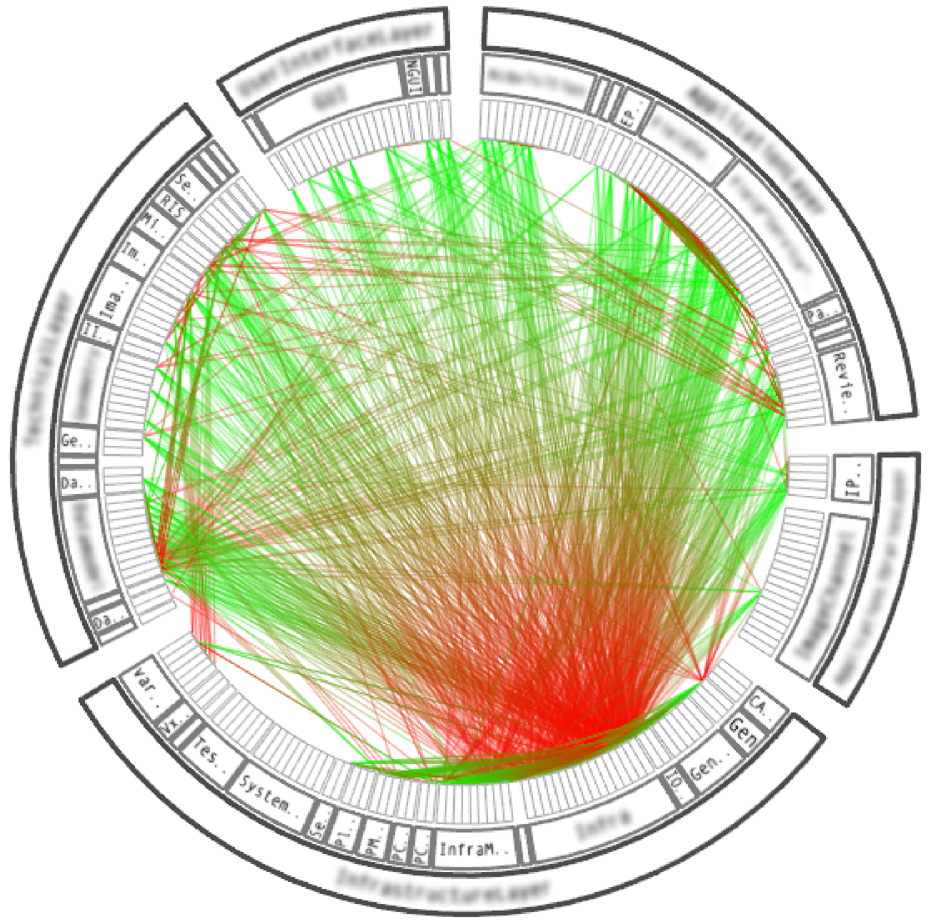
- Need to use spatial position when designing network visualizations
- Otherwise, nodes can **occlude** each other, links hard to distinguish
- How?
  - With bar charts, we could order using an attribute…
  - With networks, we want to be able to see connectivity and topology (not in the data usually)
- Possible metrics:
  - Edge crossings
  - Node overlaps
  - Total area

Layout Algorithms

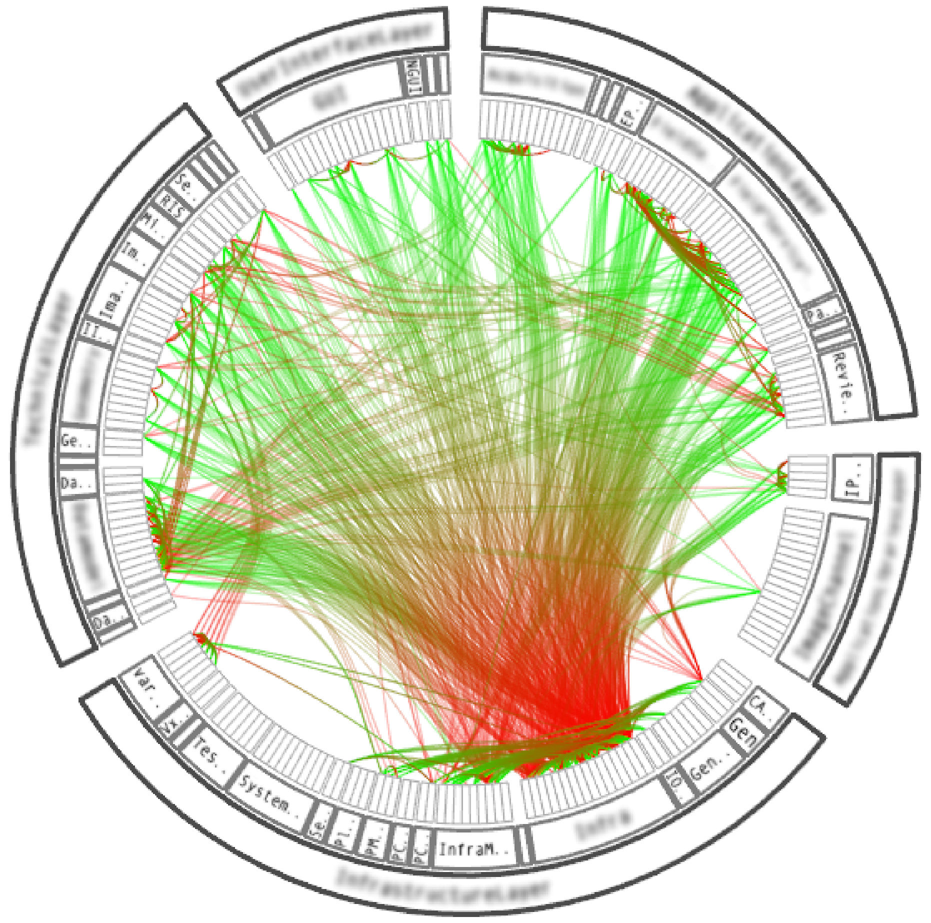


[Force-Directed and CoLa, M. Bostock]

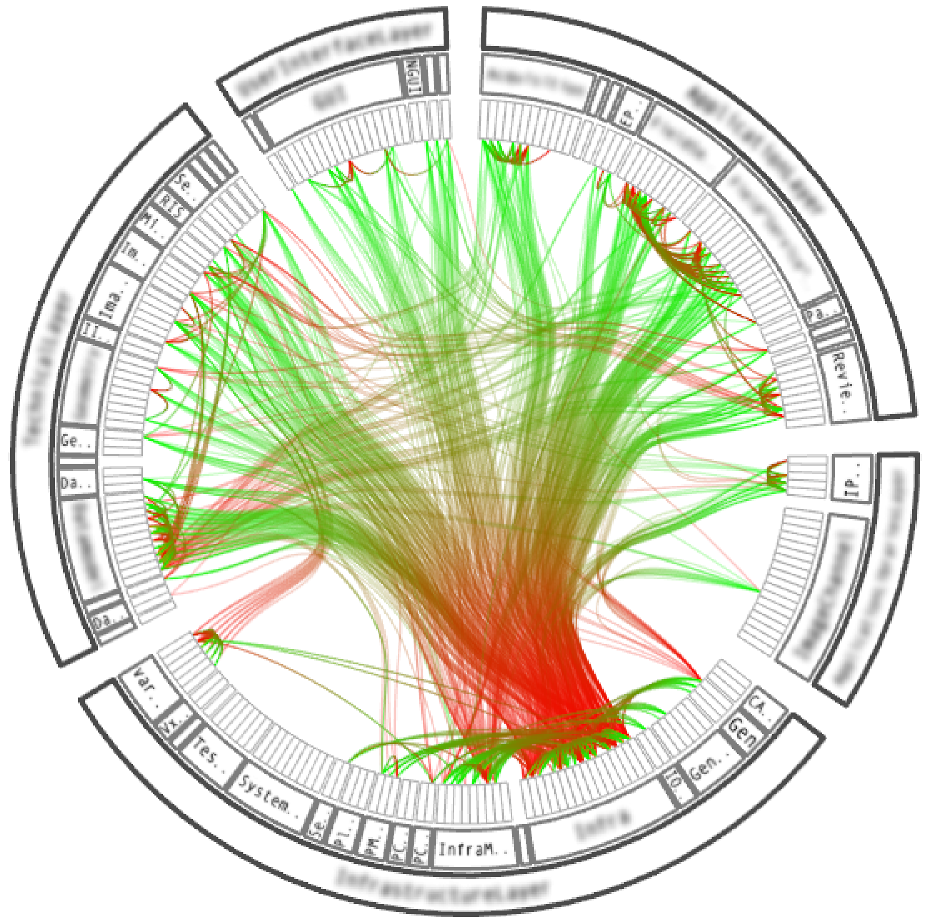
Bundling Strength



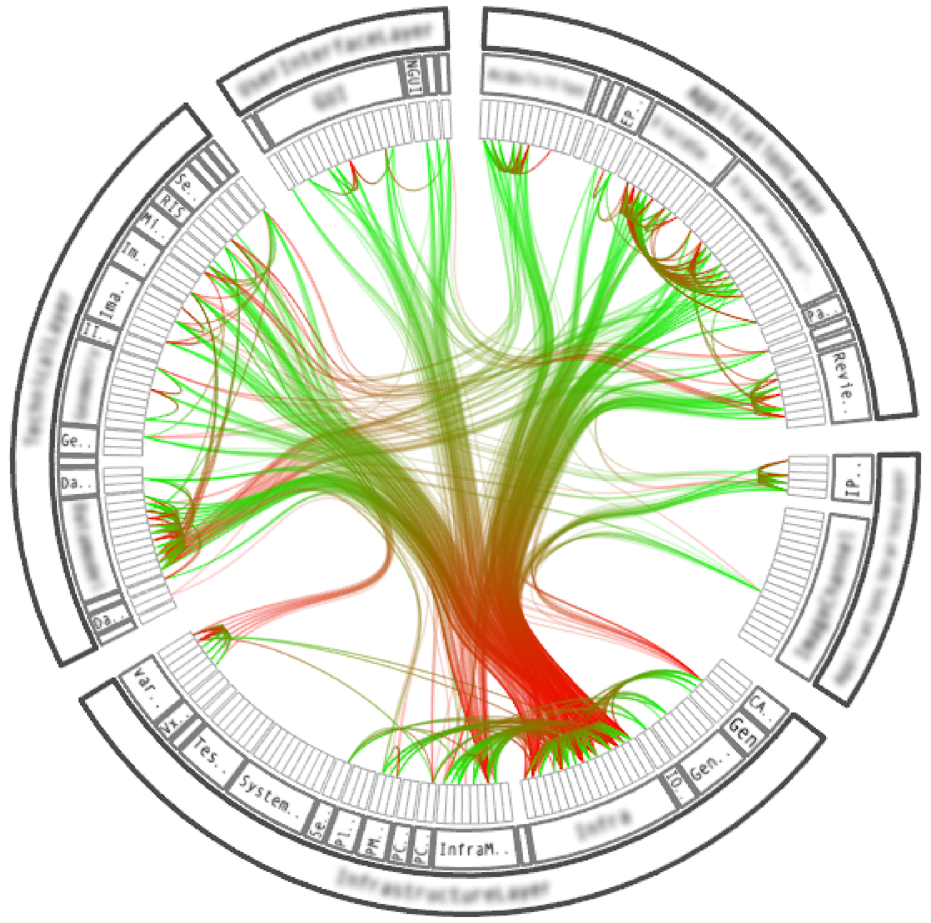
$\beta = 0$



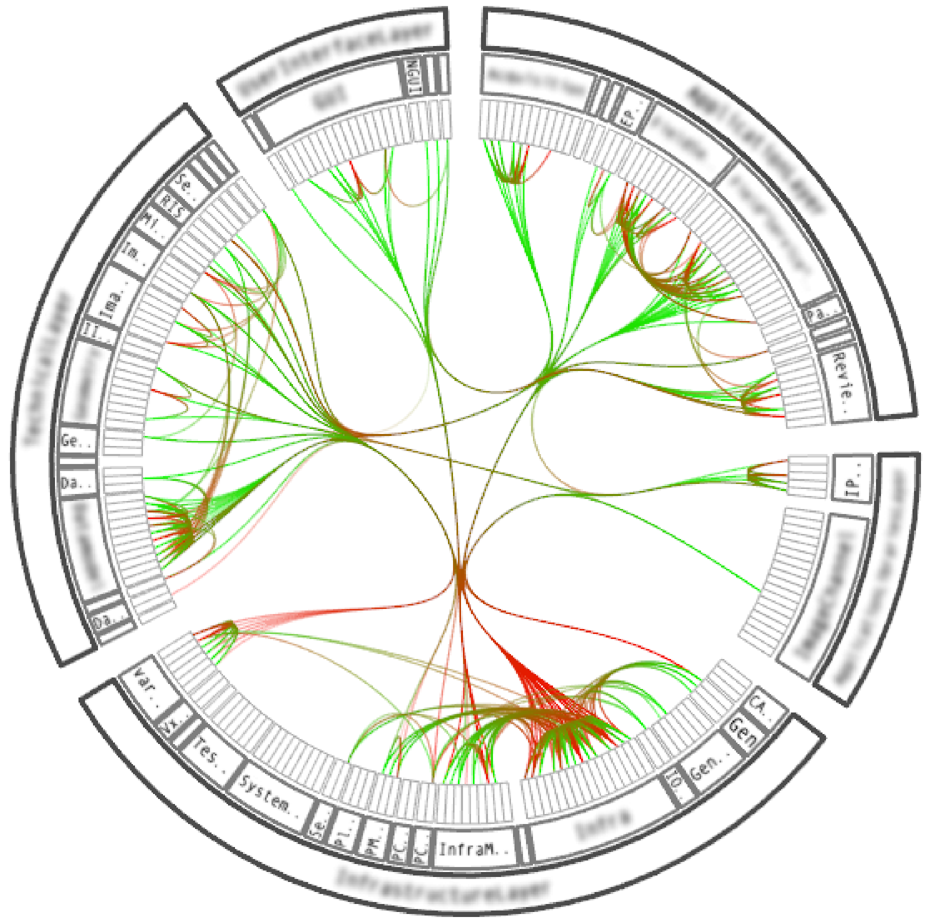
$\beta = 0.25$



$\beta = 0.5$



$\beta = 0.75$



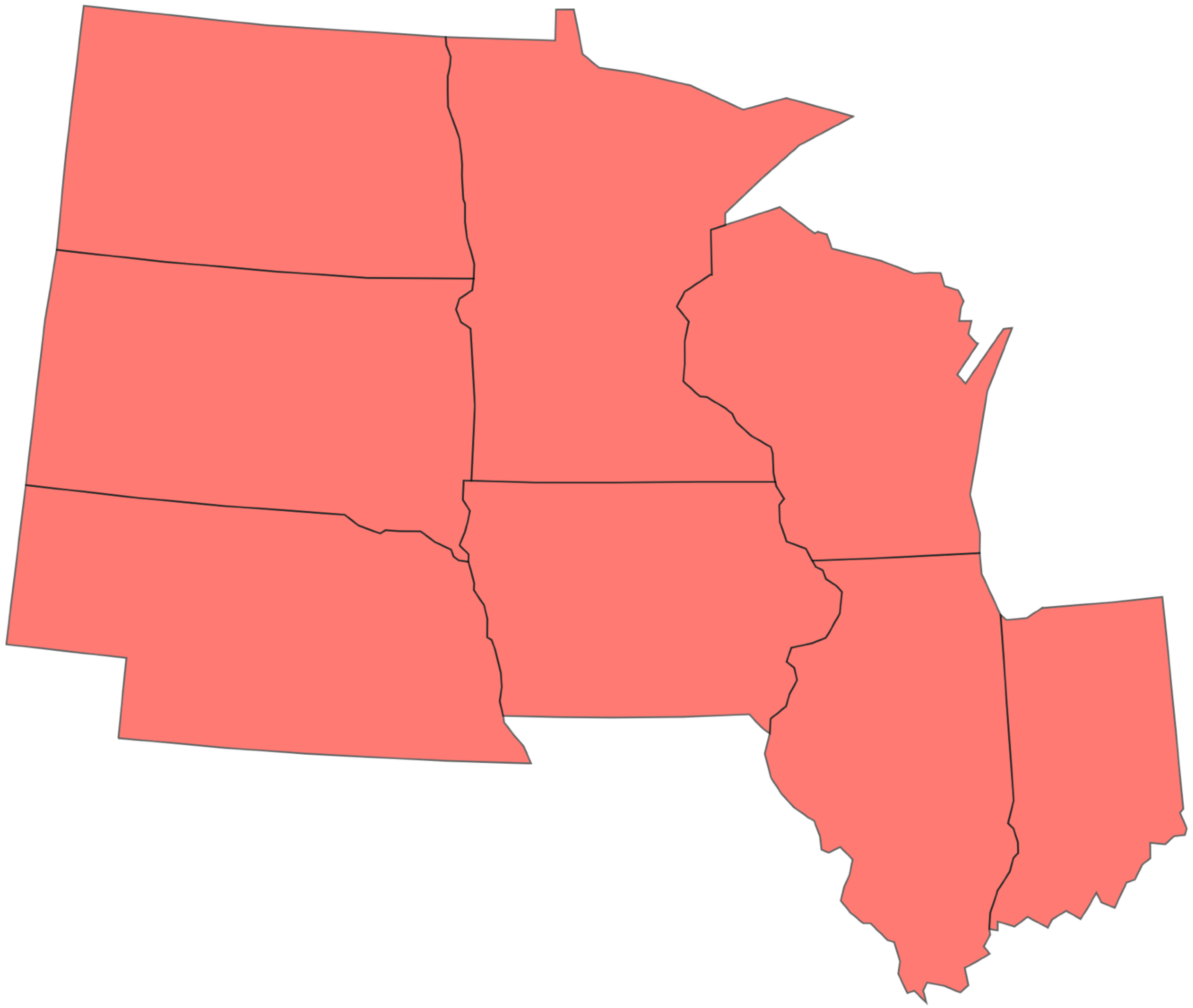
$\beta = 1$

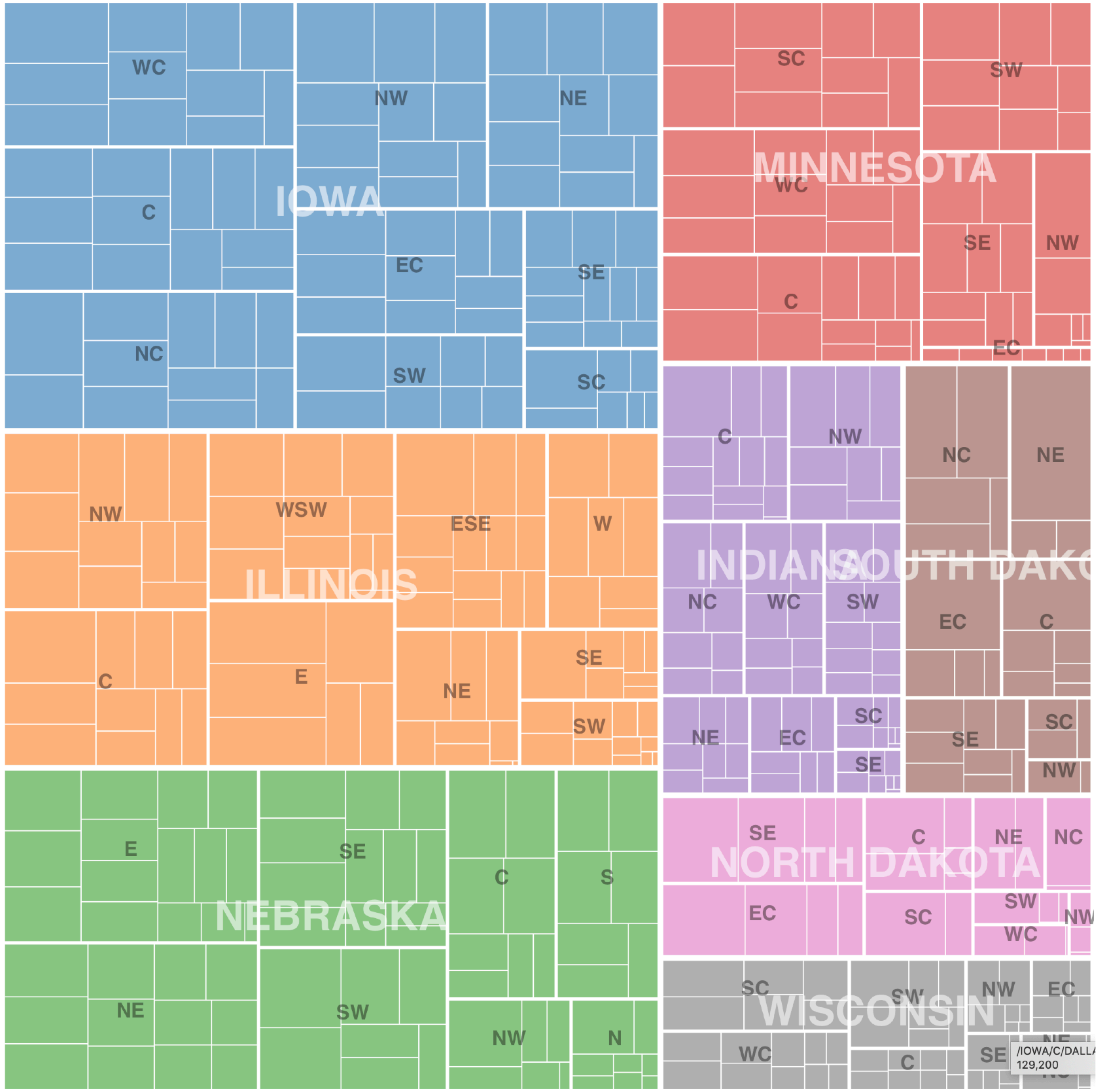
[Holten, 2006]

Project

- Next steps:
  - Start thinking about the designs that help answer the questions
  - Tasks should drive your design
  - Different designs are great
    - Multiple views
    - Single view with details on demand
    - Interaction design (linked highlighting, navigation)
    - In general, don't force the user to make choices without first seeing an overview

Assignment 4

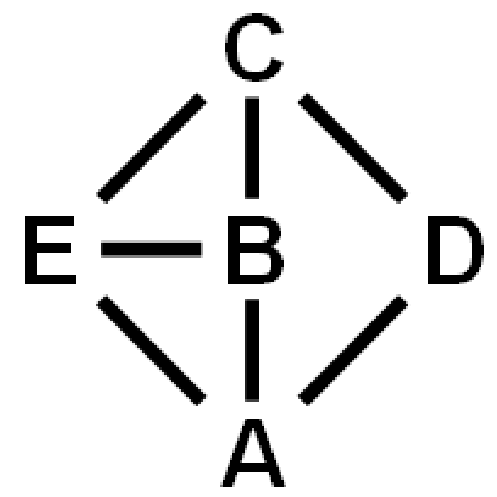


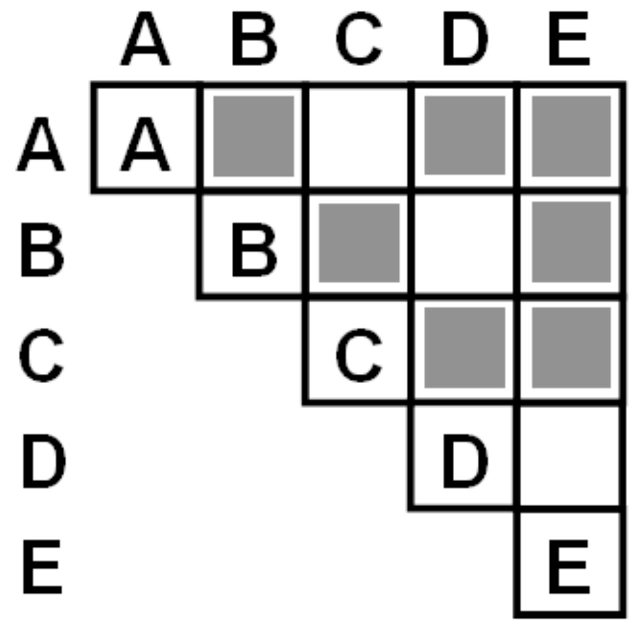


Adjacency Matrix

- Change network to tabular data and use a matrix representation
- Derived data: nodes are keys, edges are boolean values
- Task: lookup connections, find well-connected clusters
- Scalability: millions of edges

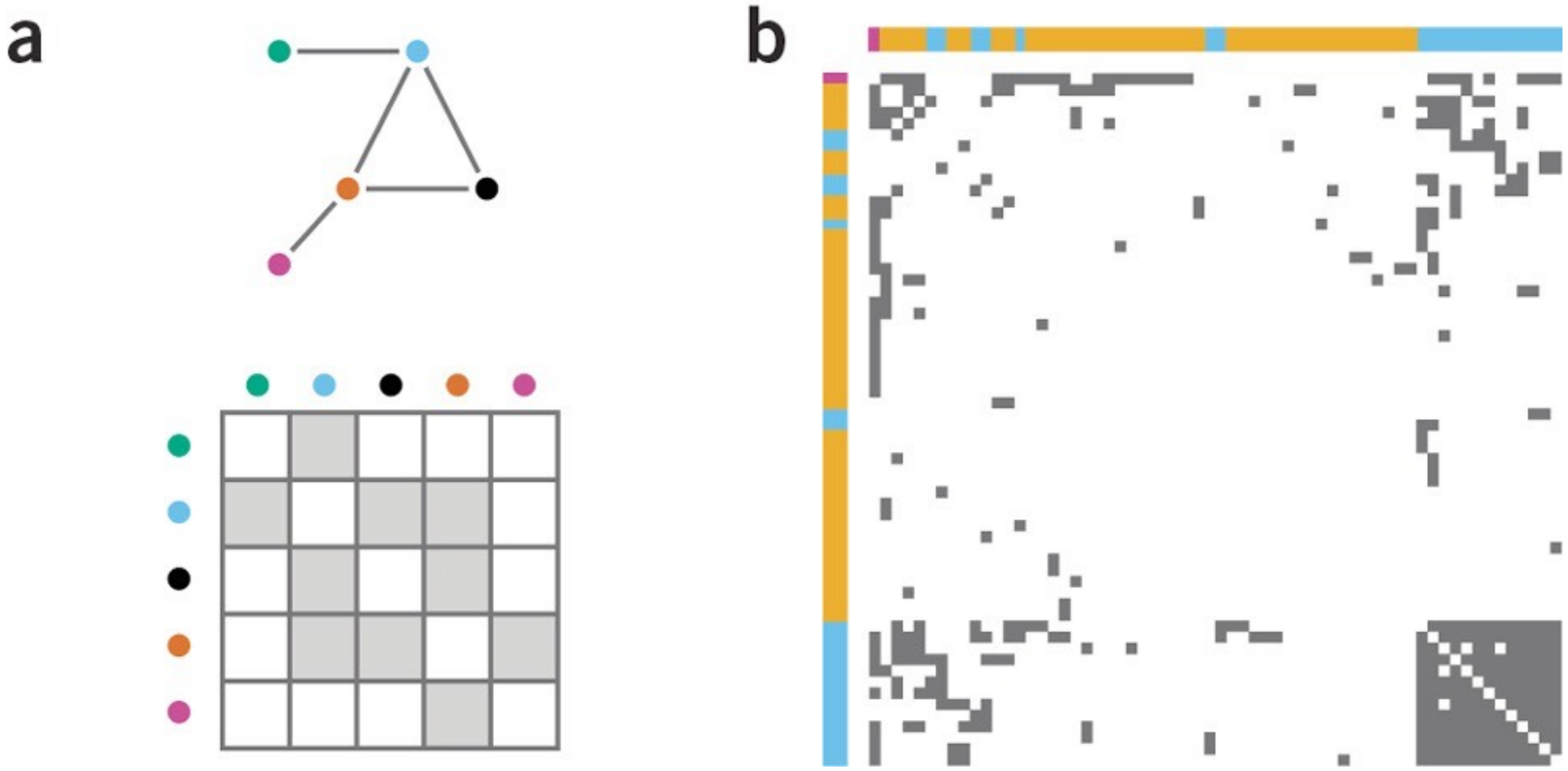
- Can encode **edge weight**, too





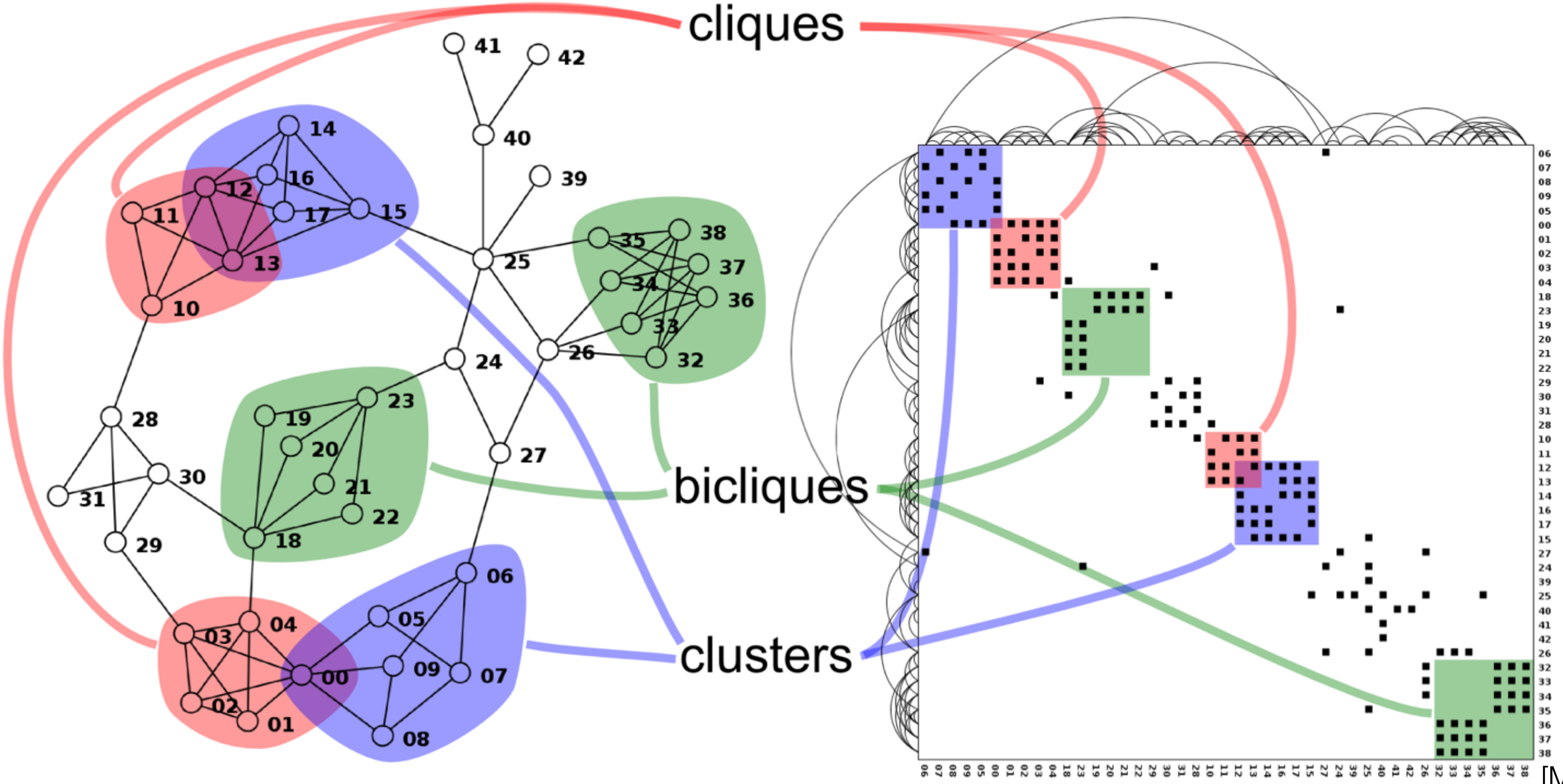
[Henry et al., 2007]

Cliques in Adjacency Matrices



[Gehlenborg and Wong]

Structures from Adjacency Matrices



[McGuffin]

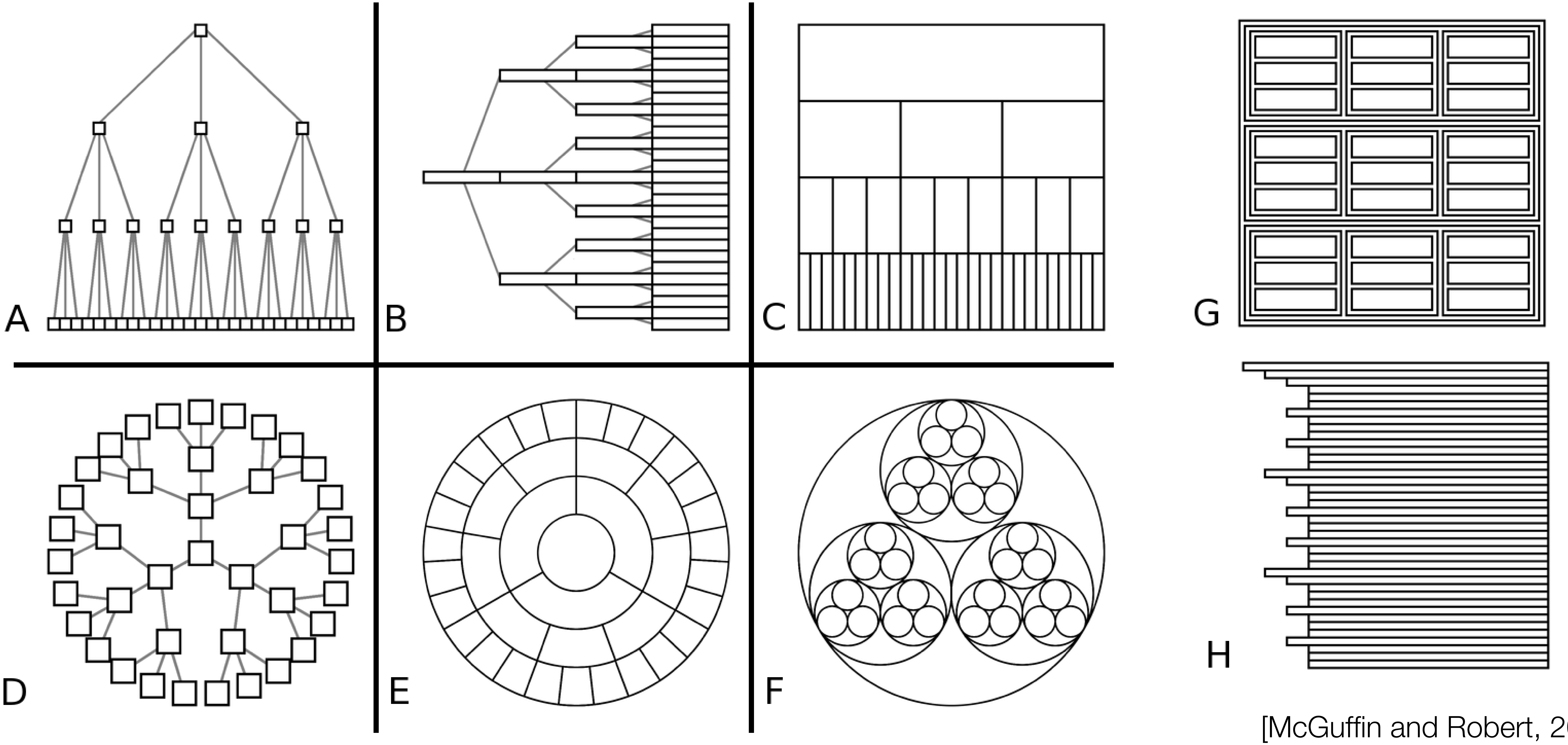
Node-Link or Adjacency Matrix?

- Empirical study: For most tasks, node-link is better for small graphs and adjacency better for large graphs
- Multi-link paths are hard with adjacency matrices
- Immediate connectivity or neighbors are ok, estimating size (nodes & edges also ok)
- People tend to be more familiar with node-link diagrams
- Link density is a problem with node-link but not with adjacency matrices

Trees

- Trees are directed acyclic **networks**
  - each edge has a direction: the origin is the parent, the destination is the child
  - cannot get back to a node after leaving it
- …plus each node has **at most one** parent node
- A tree has a **root** (every other node hangs off it)
- Can consider enclosure in trees using parent-child relationships

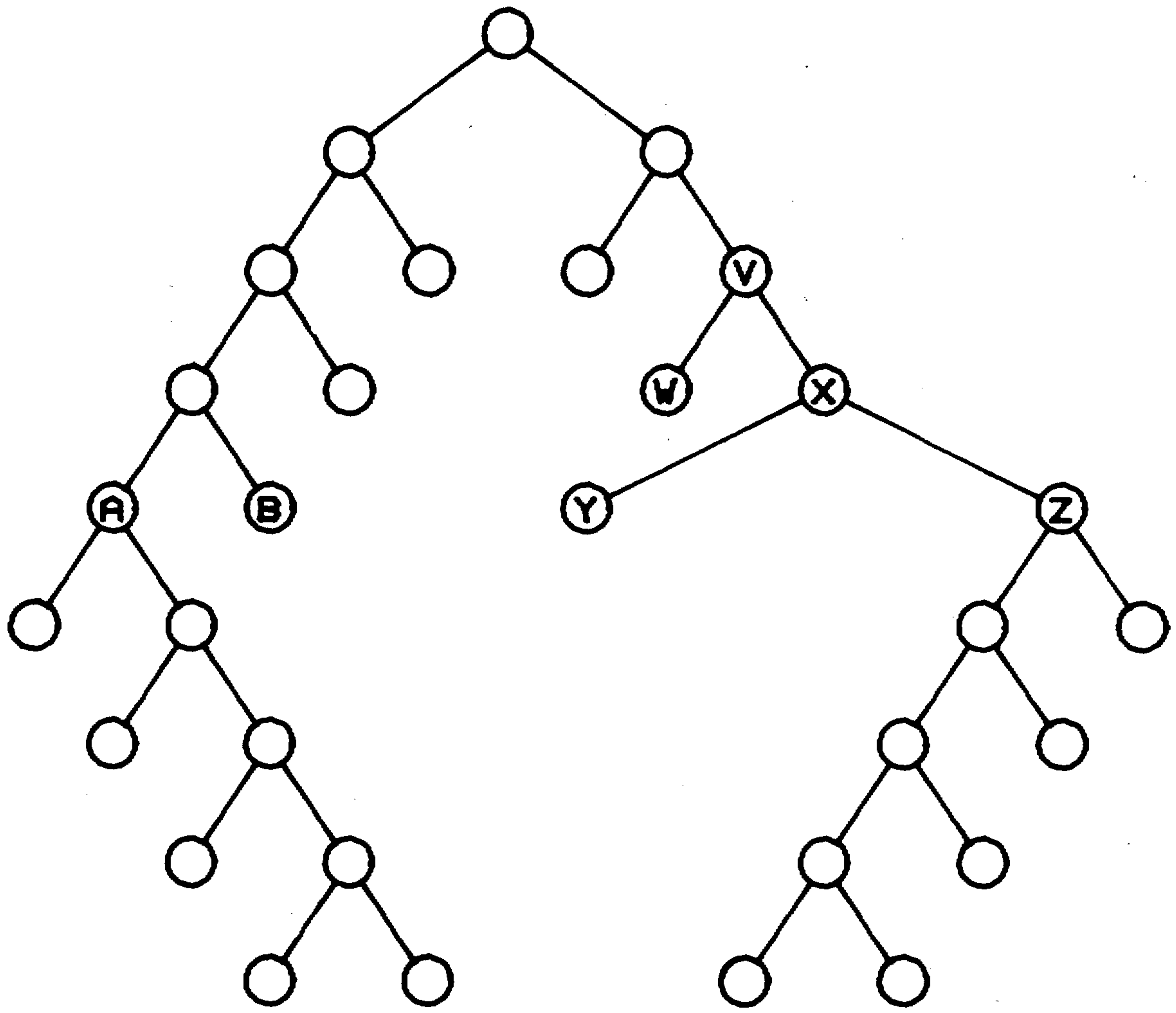
Tree Visualizations



[McGuffin and Robert, 2010]

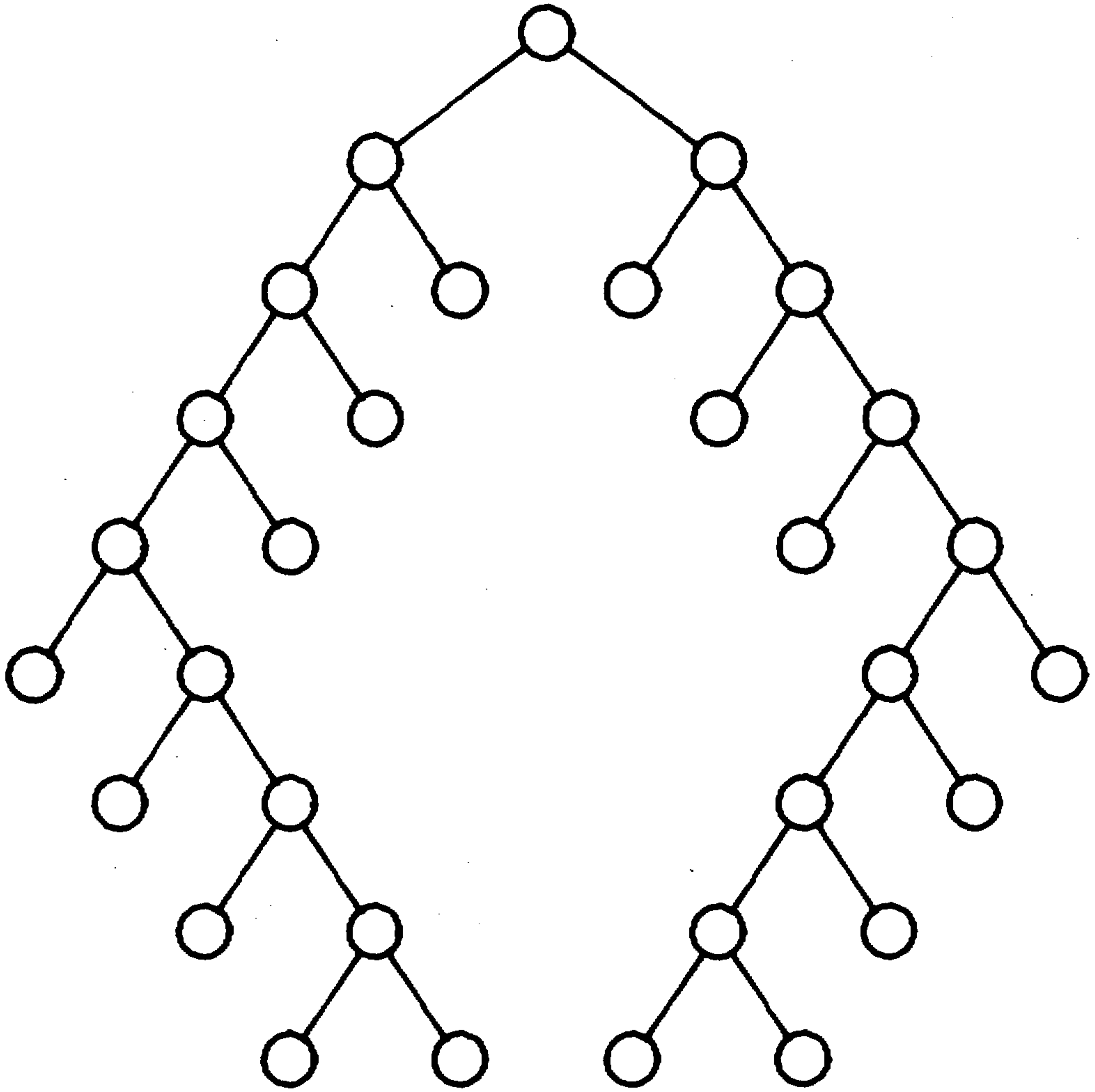
Node-Link Diagram

- Trees are graphs
- …but we have more structure
- Horizontal or vertical
- Idea 1: partition space for each node via recursion
- Idea 2: "Tidy" Drawing
    - Wetherell & Shannon: Don't waste space (overlapping parent nodes is ok)
    - Reingold and Tilford: Keep symmetry, subtrees look similar



[WS Alg., Reingold and Tilford, 1981]

Reingold-Tilford Algorithm

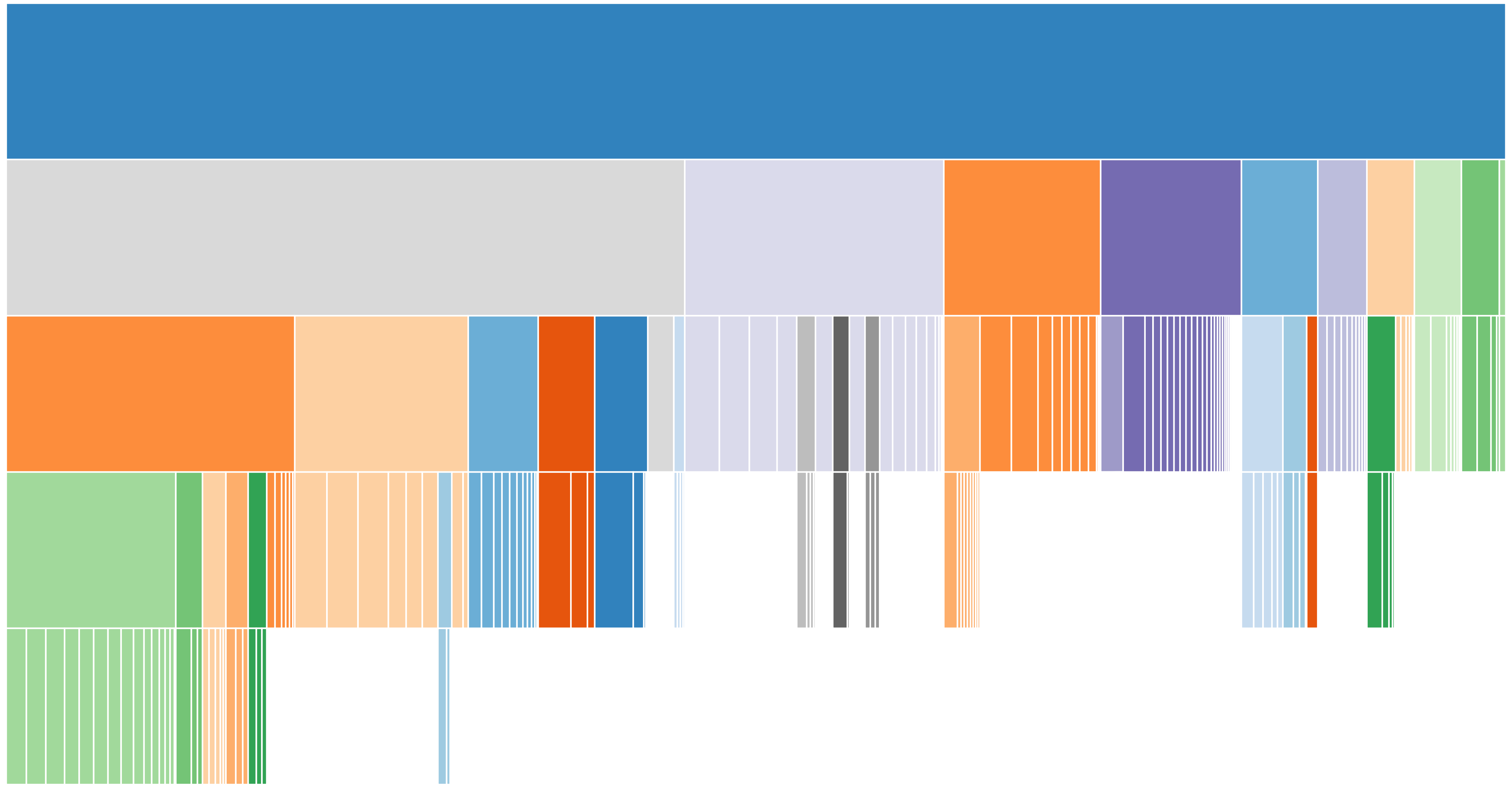


- Recurse on left and right subtrees
- Shift subtree over as long as it doesn't overlap
- Place parent centered above the subtrees
- Originally, only binary trees, extended by Walker

[Reingold and Tilford, 1981]

Icicle Plot

- Line marks
- Vertical position shows depth
- Horizontal position shows links and sibling order
- Scalability: 1 pixel leaves, but harder to label



[Bostock, 2011]

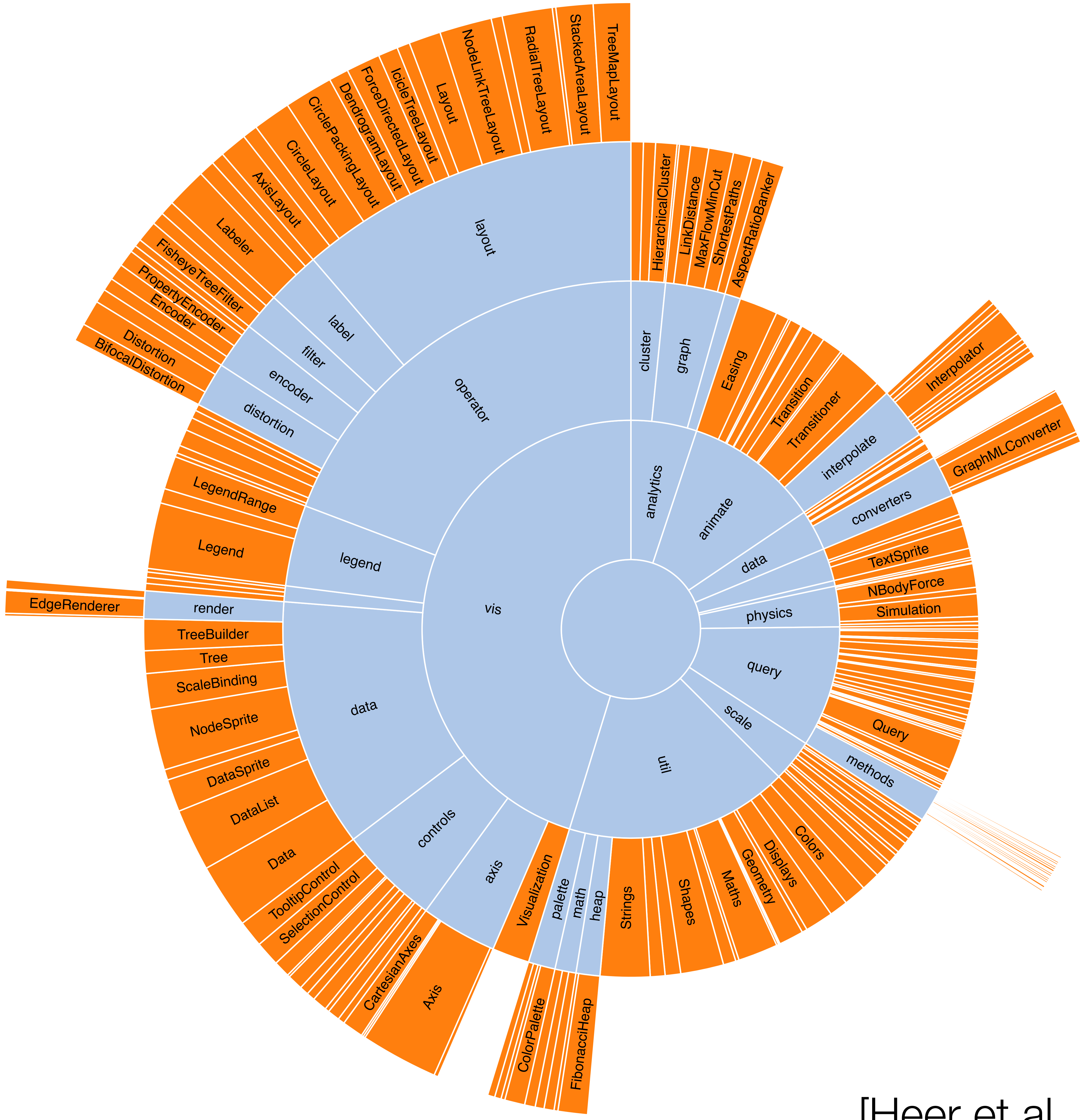
Radial Node-Link

- Use polar coordinates instead of rectilinear
- Same layout algorithms work (e.g. Reingold-Tilford)
- Benefit: space usage, labels

[M. Bostock, 2017]

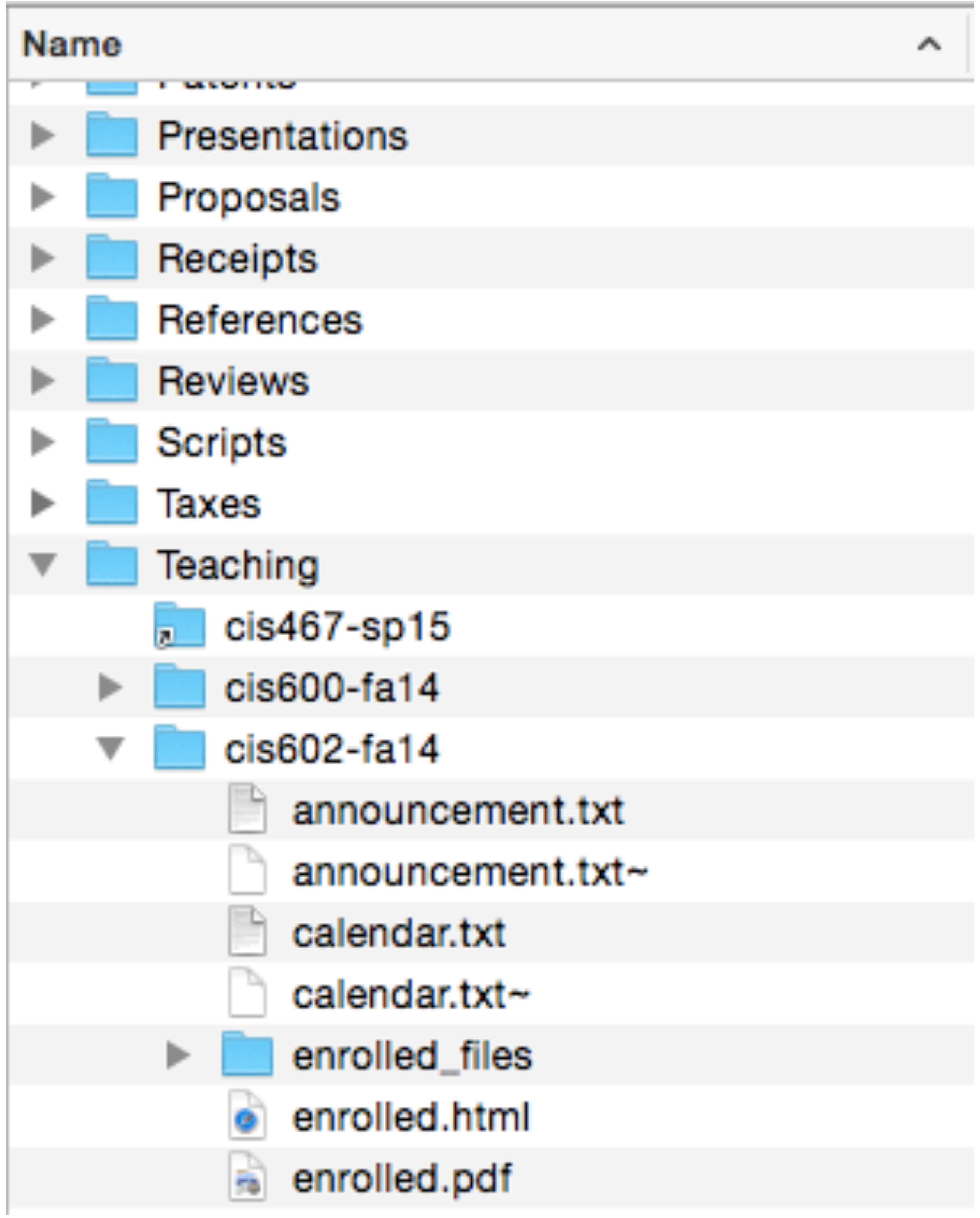
Sunburst

- Icicle plot in a radial layout
- Reading labels?
- Intuitive navigation



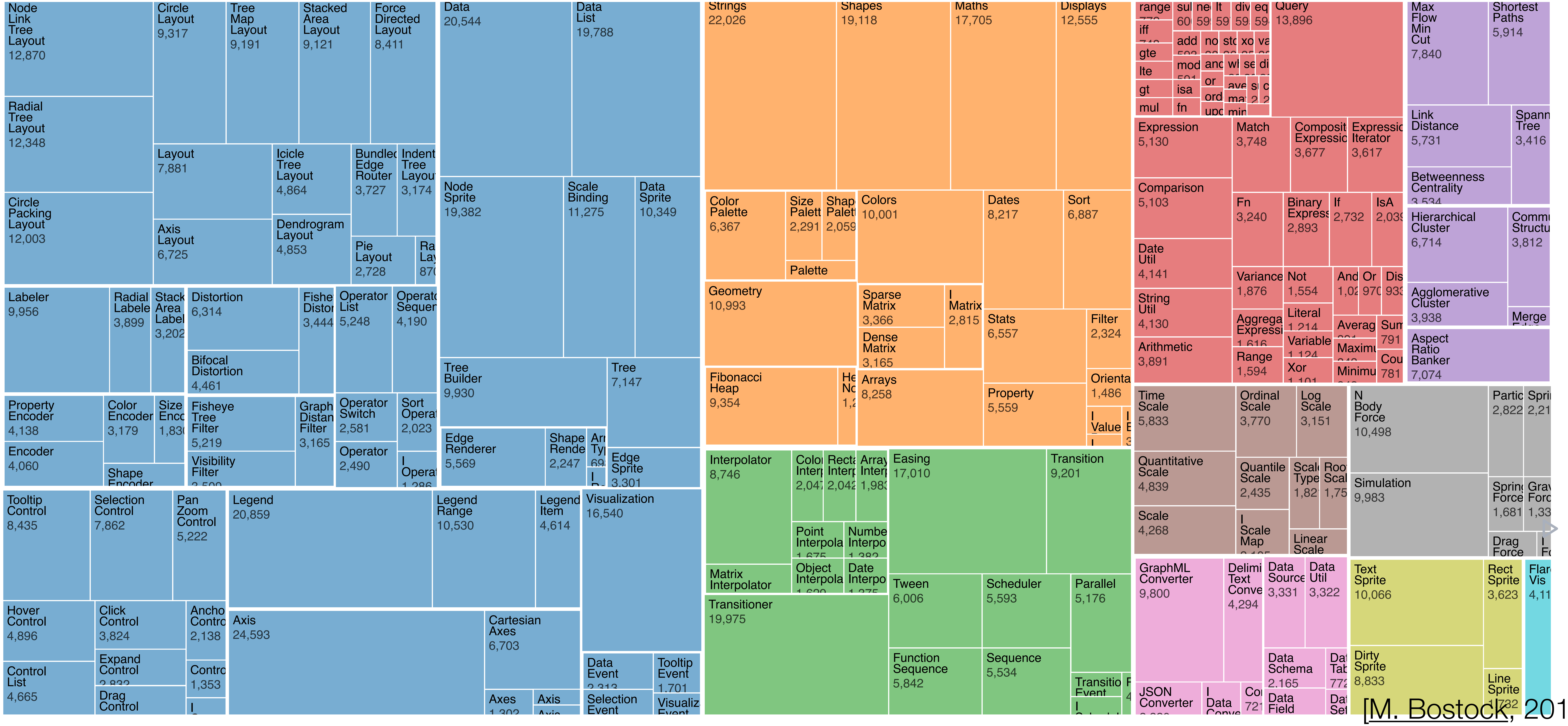
[Heer et al., 2012]

Indented Outline



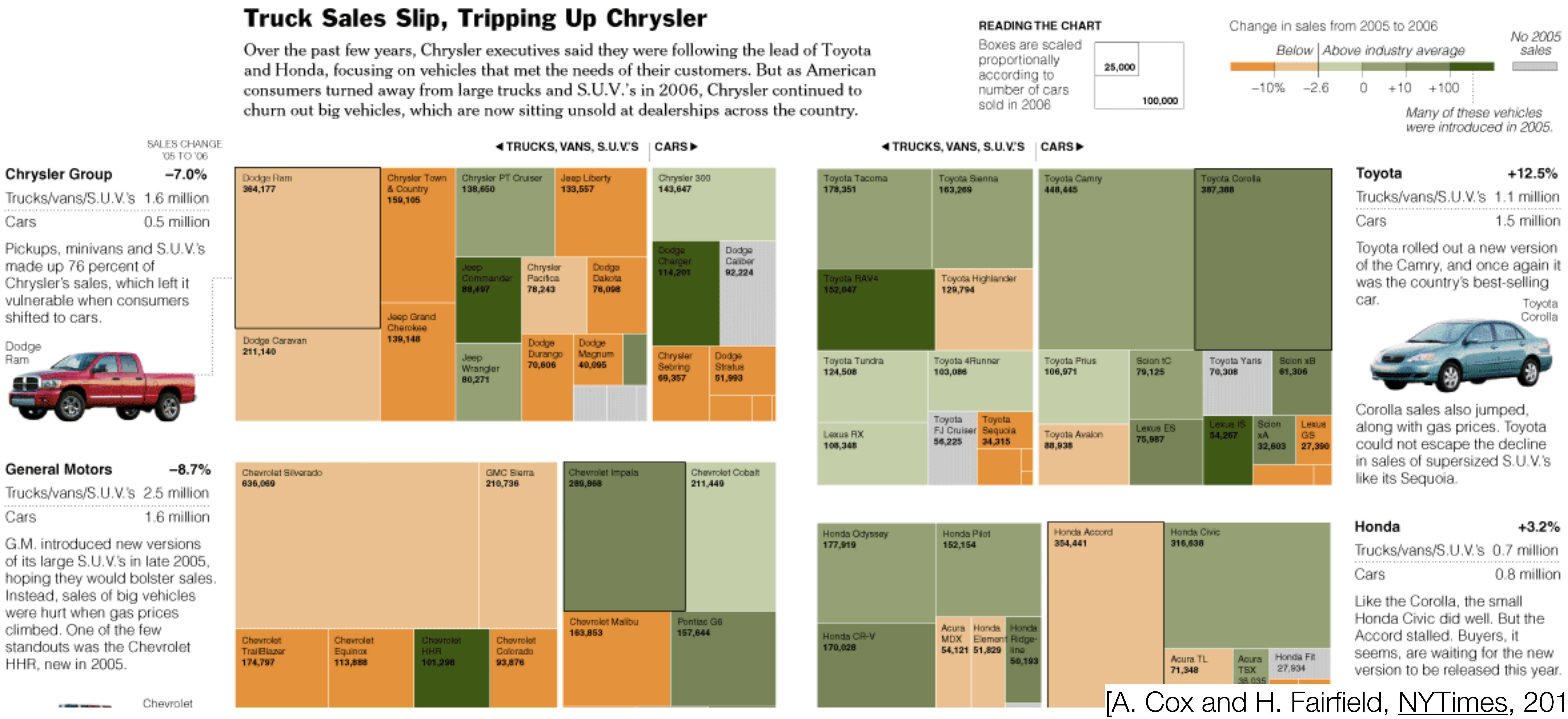
- Like a filesystem tree
- Use horizontal position to show depth, vertical positions show sibling/order

Treemap



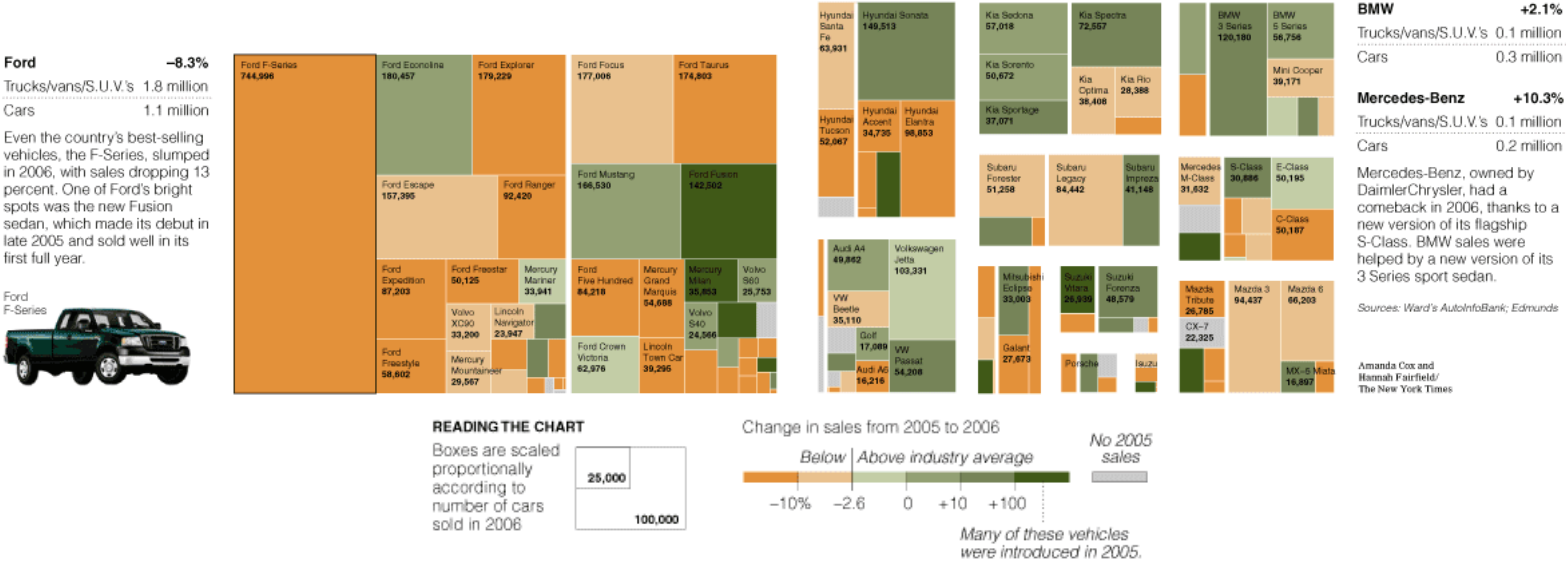
[M. Bostock, 2017]

Car/Truck Treemap



[A. Cox and H. Fairfield, NYTimes, 2012]

Car/Truck Treemap



[A. Cox and H. Fairfield, NYTimes, 2012]

Treemap

- Containment marks instead of connection marks
- Encodes some attribute of the items as the **size** of the rectangles
- Not as easy to see the intermediate rectangles
- Scalability: millions of leaf nodes and links possible

- Need a layout algorithm!

Layout Algorithms

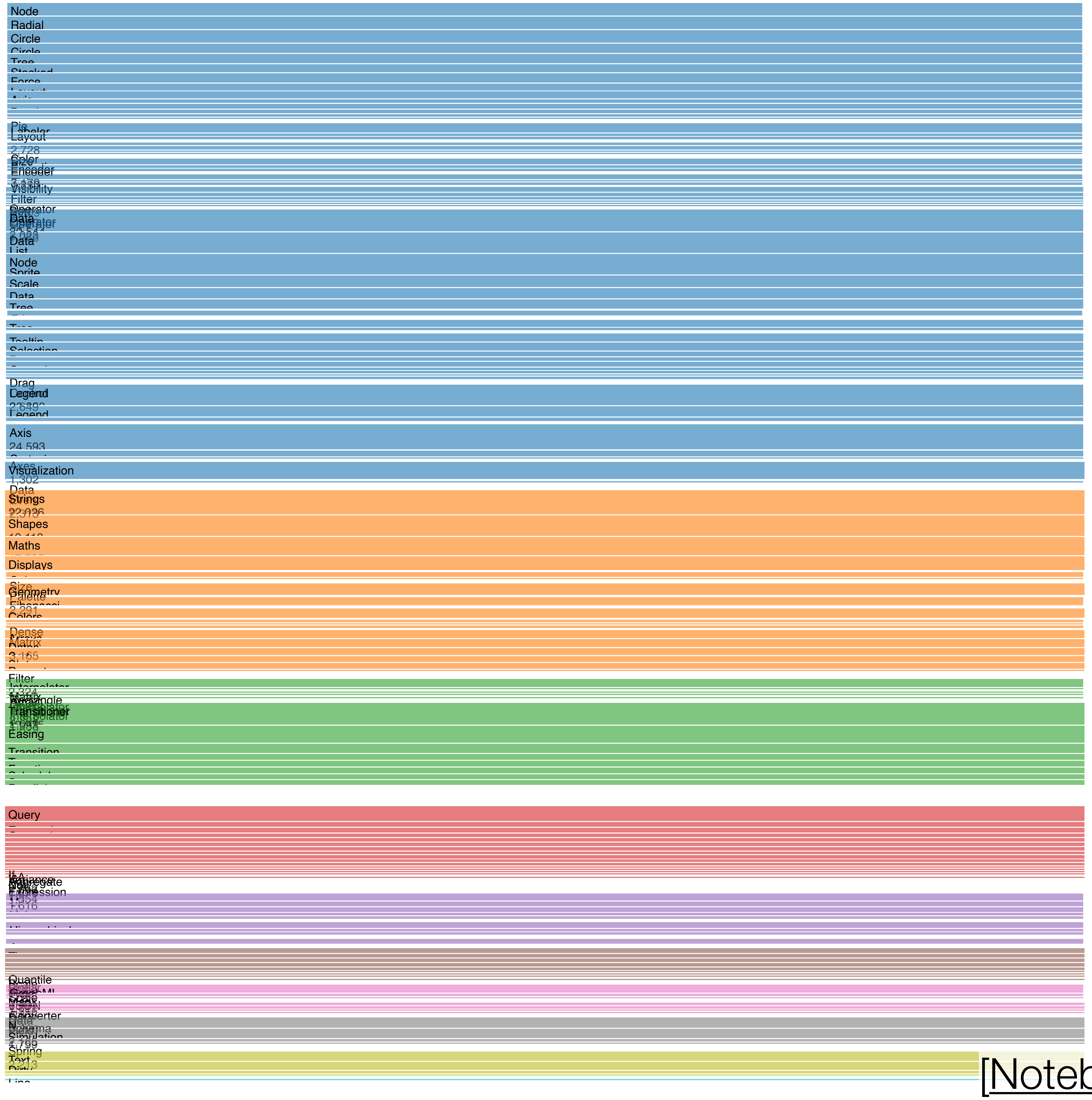
- How do we generate the area marks?
- What considerations should we try to keep in mind?

Layout Algorithms

- How do we generate the area marks?
- What considerations should we try to keep in mind?
  - area true to quantitative value
  - show hierarchy
  - aspect ratio
- Also…
  - ordering
  - stability

Treemap Layouts: Slice

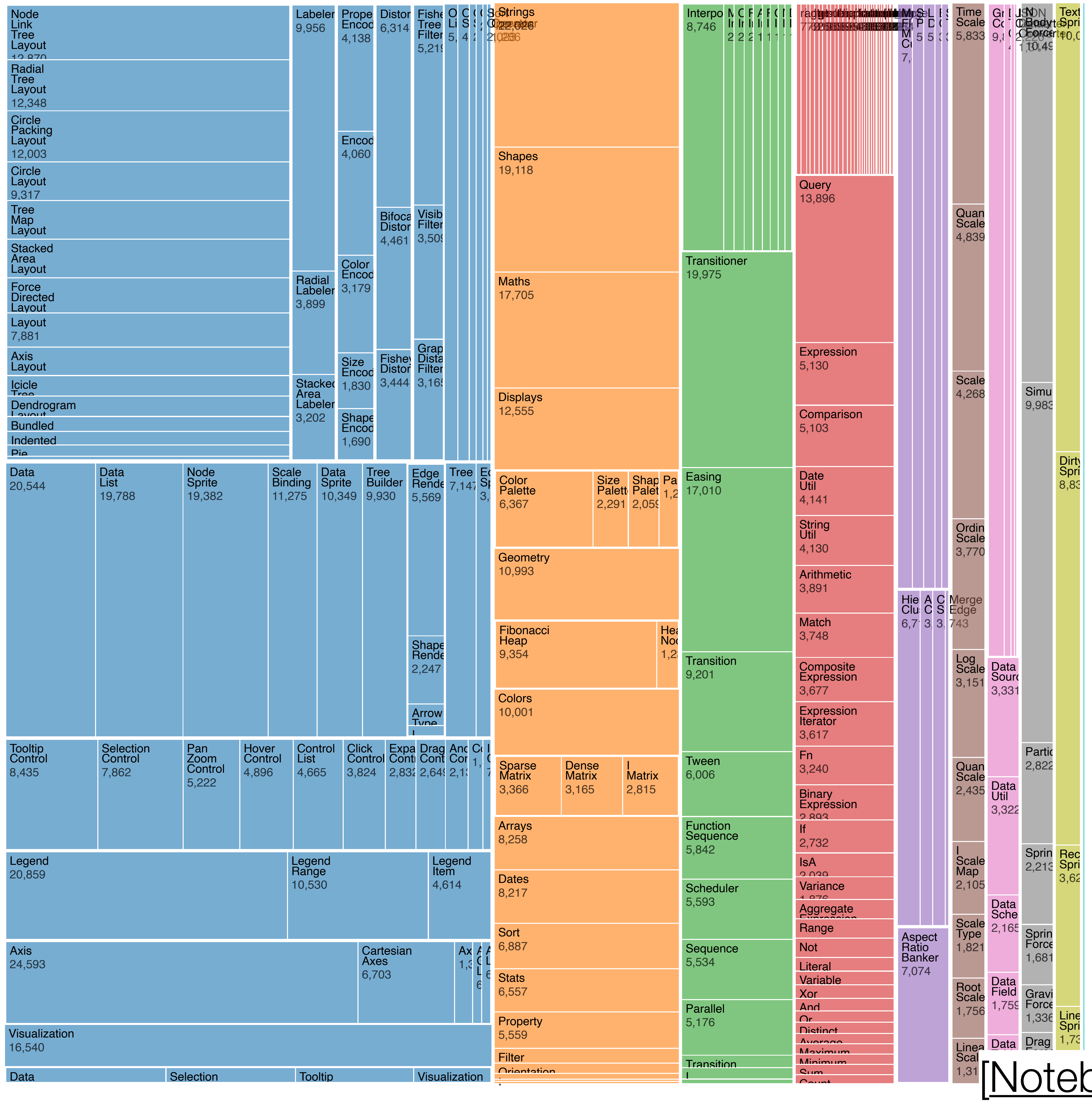
- Just divide horizontally
- Dice is similar, just vertical
- Problem: Bad aspect ratio!
  - Very skinny rectangles
  - Makes it harder to compare sizes, see labels, select rectangles
  - Want rectangles that are closer to squares
  - Aspect ratio = width/height



[Notebook]

Treemap Layouts: Slice & Dice

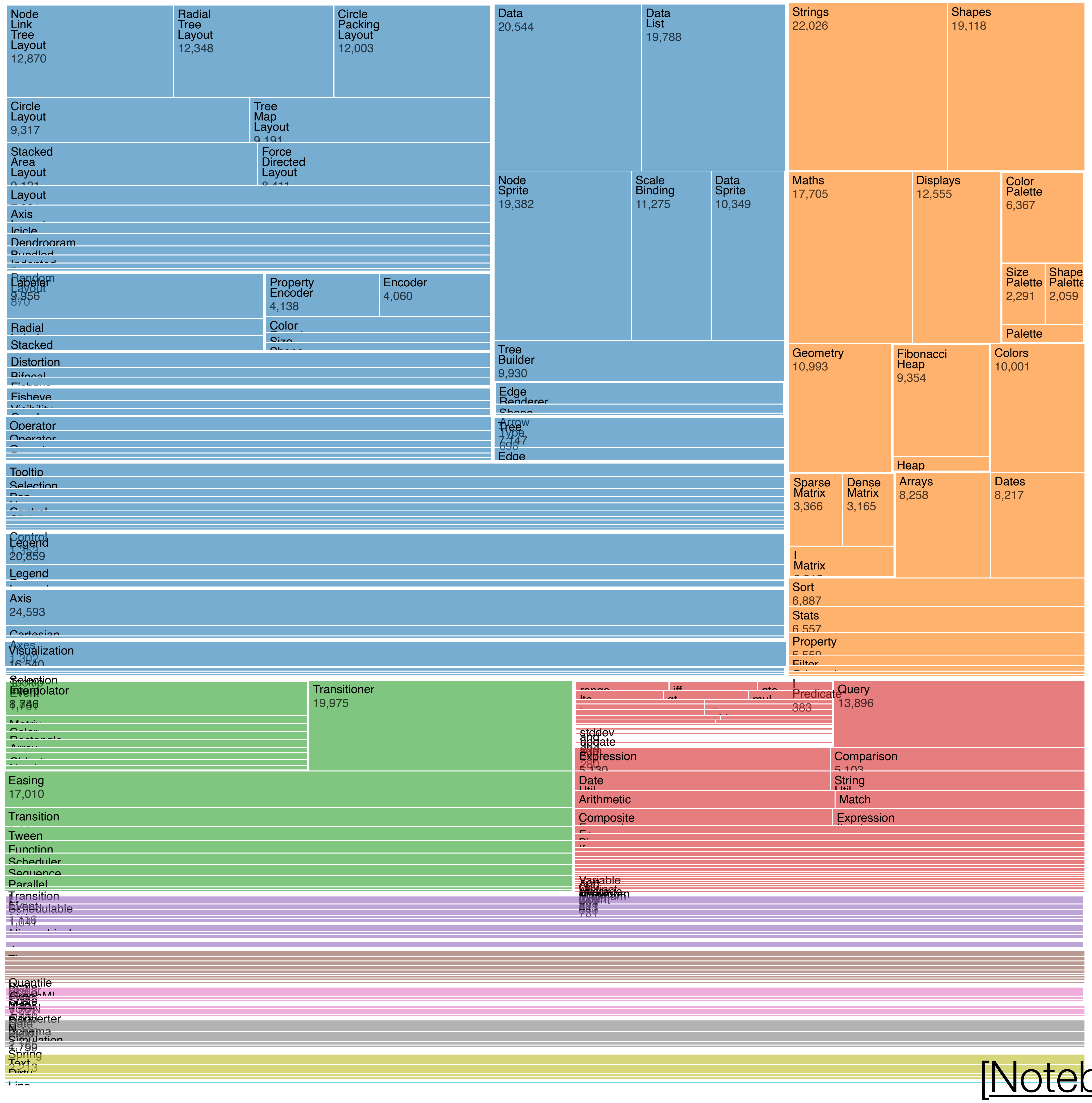
- Split at each level into strips
- At each step, orientation of division (horizontal/vertical) changes
- Better, but some rectangles still have bad aspect ratio



[Notebook]

Treemap Layouts: Strip

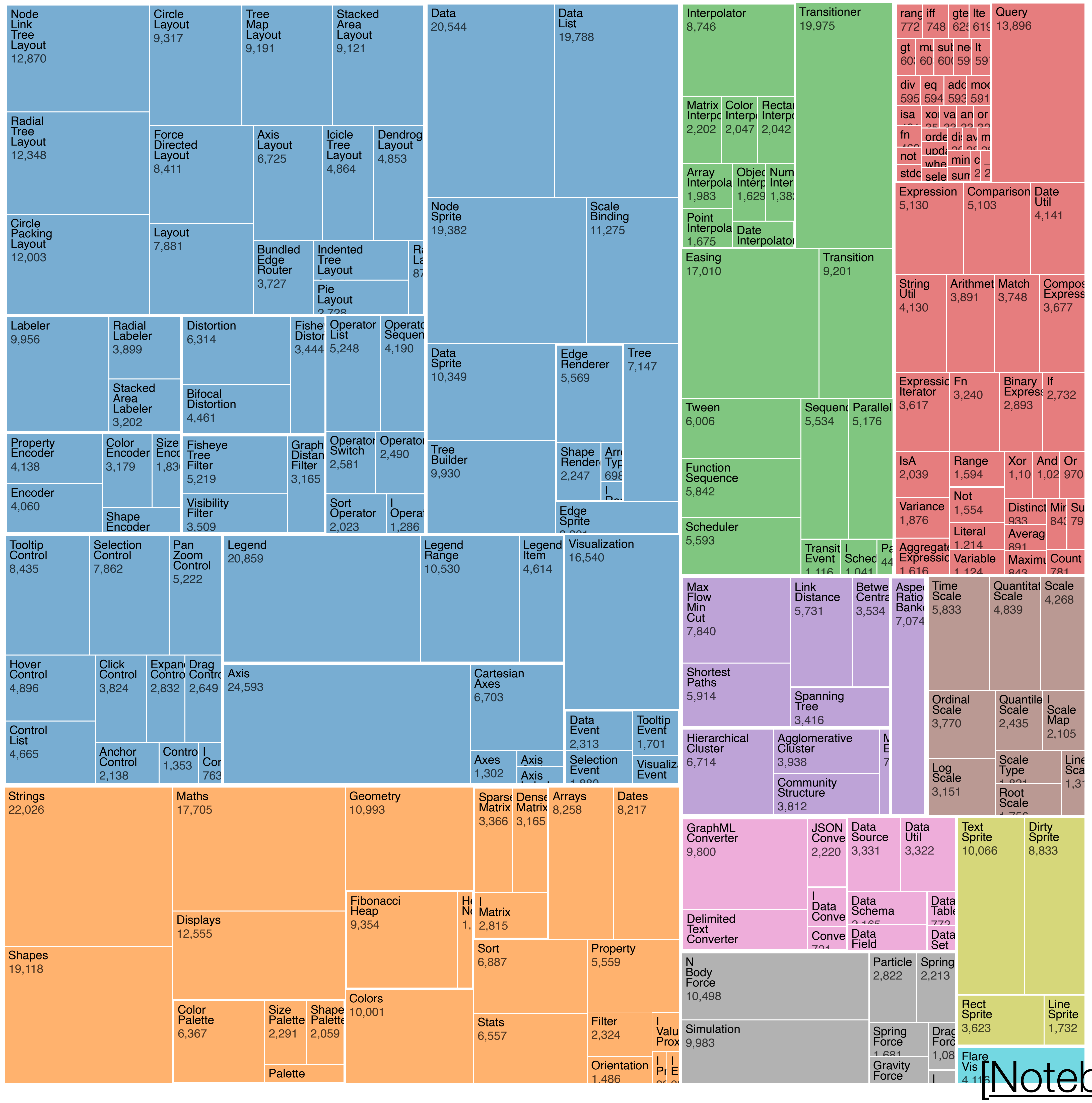
- Consider aspect ratio when adding rectangles
- Do one row at a time by processing rectangles in sorted order by size
  - Check if adding the next rectangle to the row improves aspect ratio
  - When it doesn't, go to next row
- Problem: Last rectangles have bad aspect ratios
- Solution: Look ahead to decide if would be better to add to previous row



[Notebook]

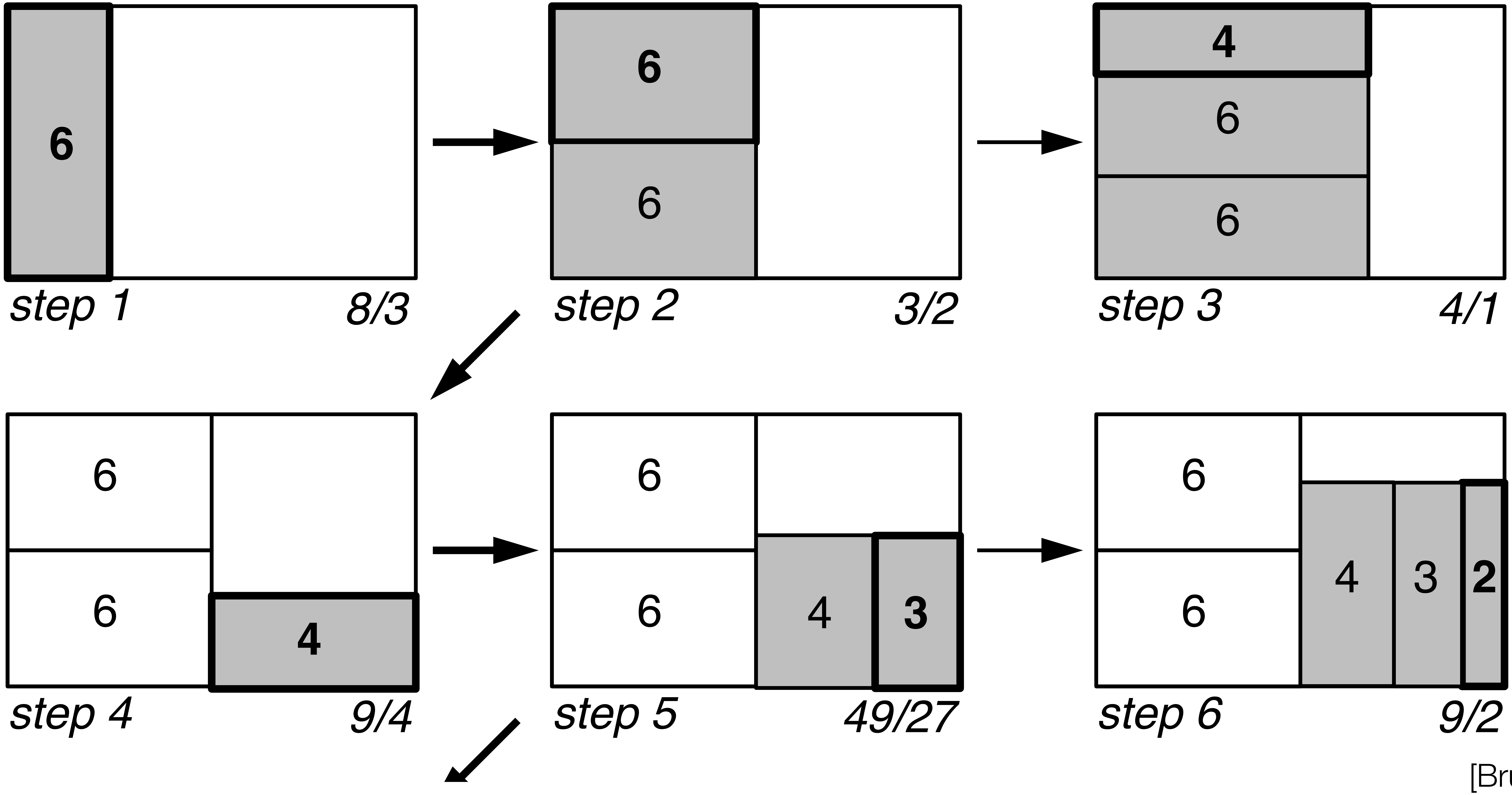
Treemap Layouts: Squarify

- Slice & Dice and Strip can lead to bad **aspect ratios**
- Solution: Strip only uses rows, allow columns to be used, too
- Choose divisions (x/y) based on the width/height of region in order to maintain good aspect ratios
  - Use left and right side
  - Process large rectangles first
- Ordering not preserved which may cause issues if the data is updated



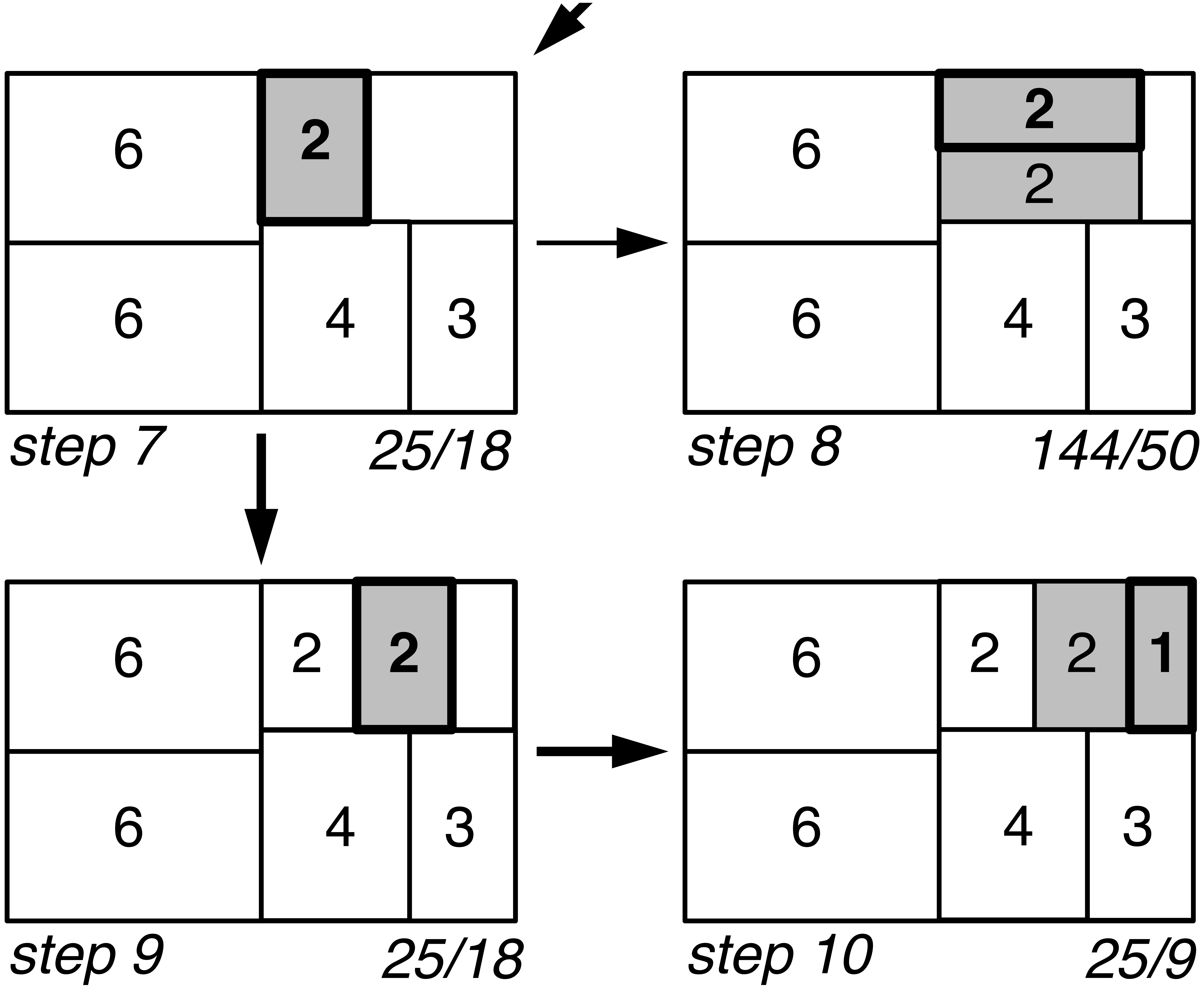
[Notebook]

Squarification Algorithm



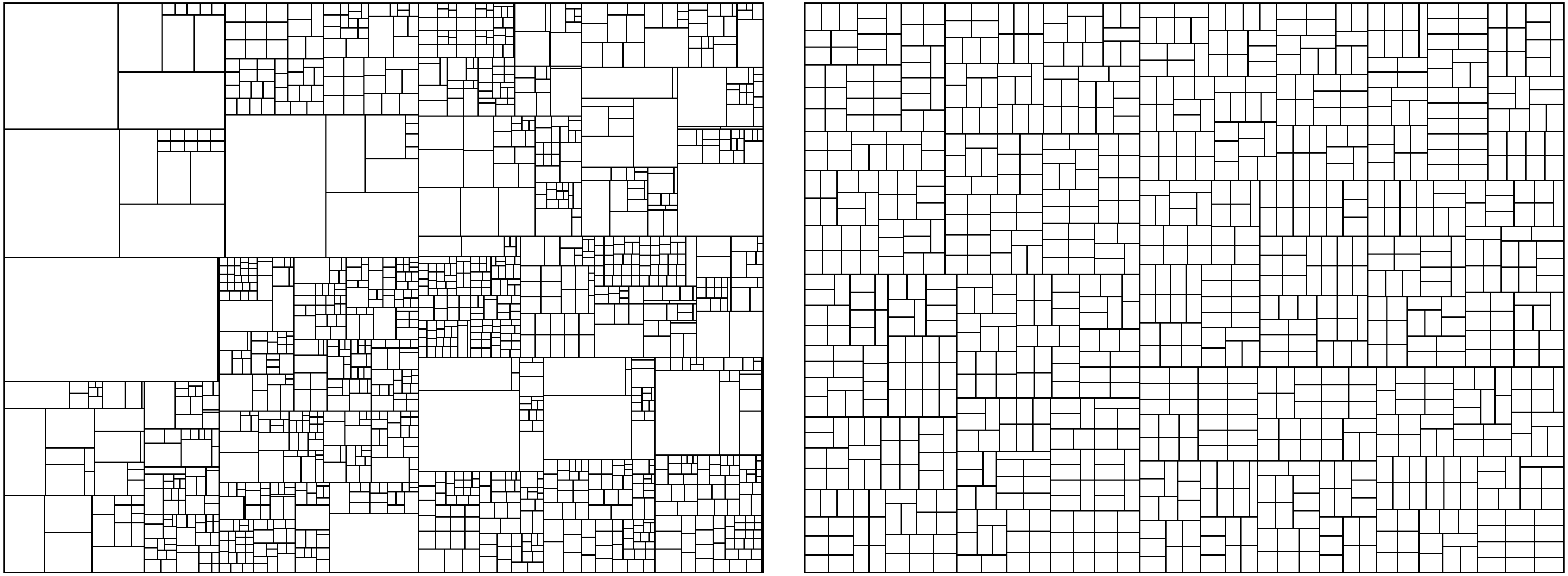
[Brus et al., 1999]

Squarification Algorithm



[Brus et al., 1999]

Squarified Treemaps



(a) File system (b) Organization

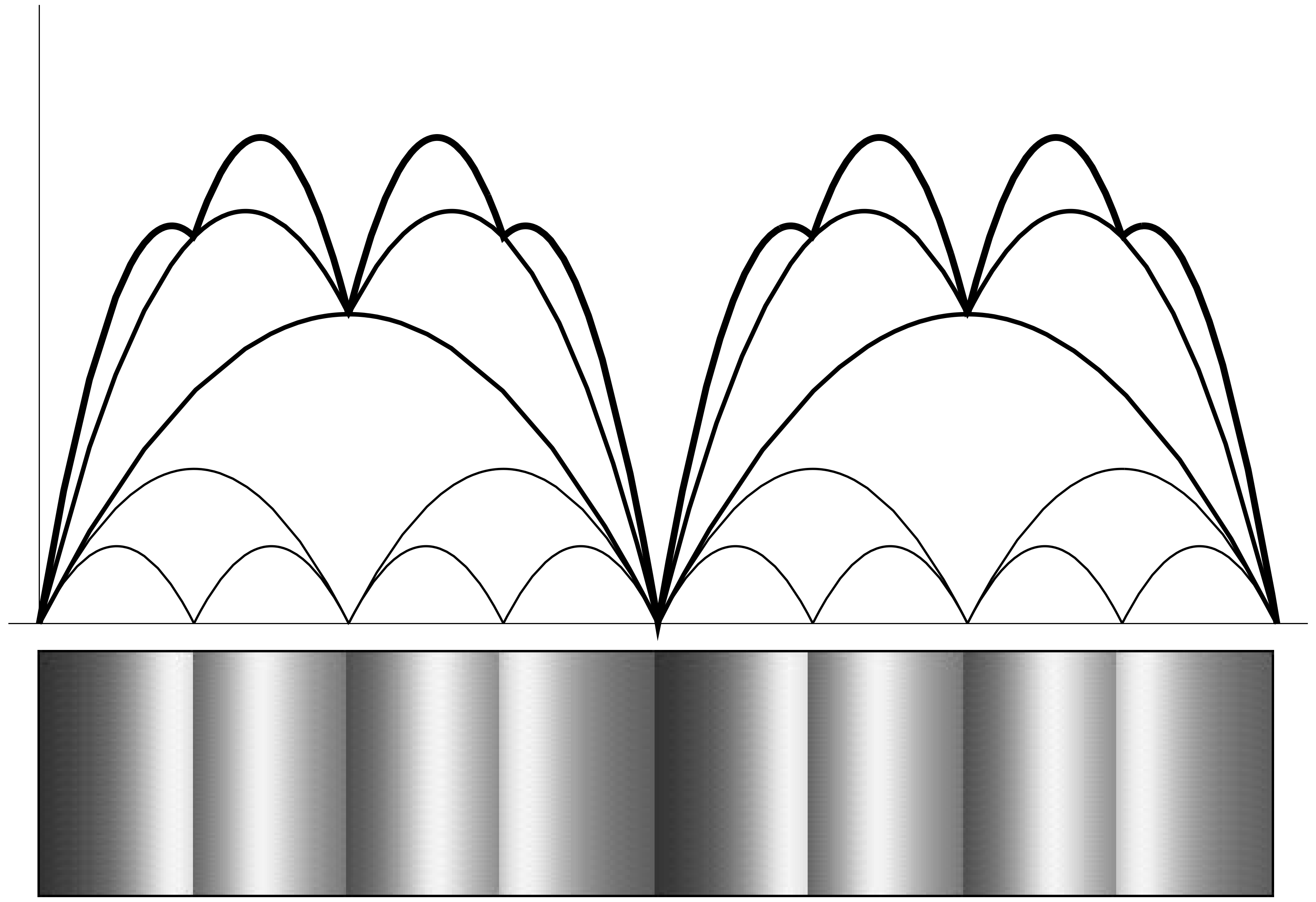
[Brus et al., 1999]

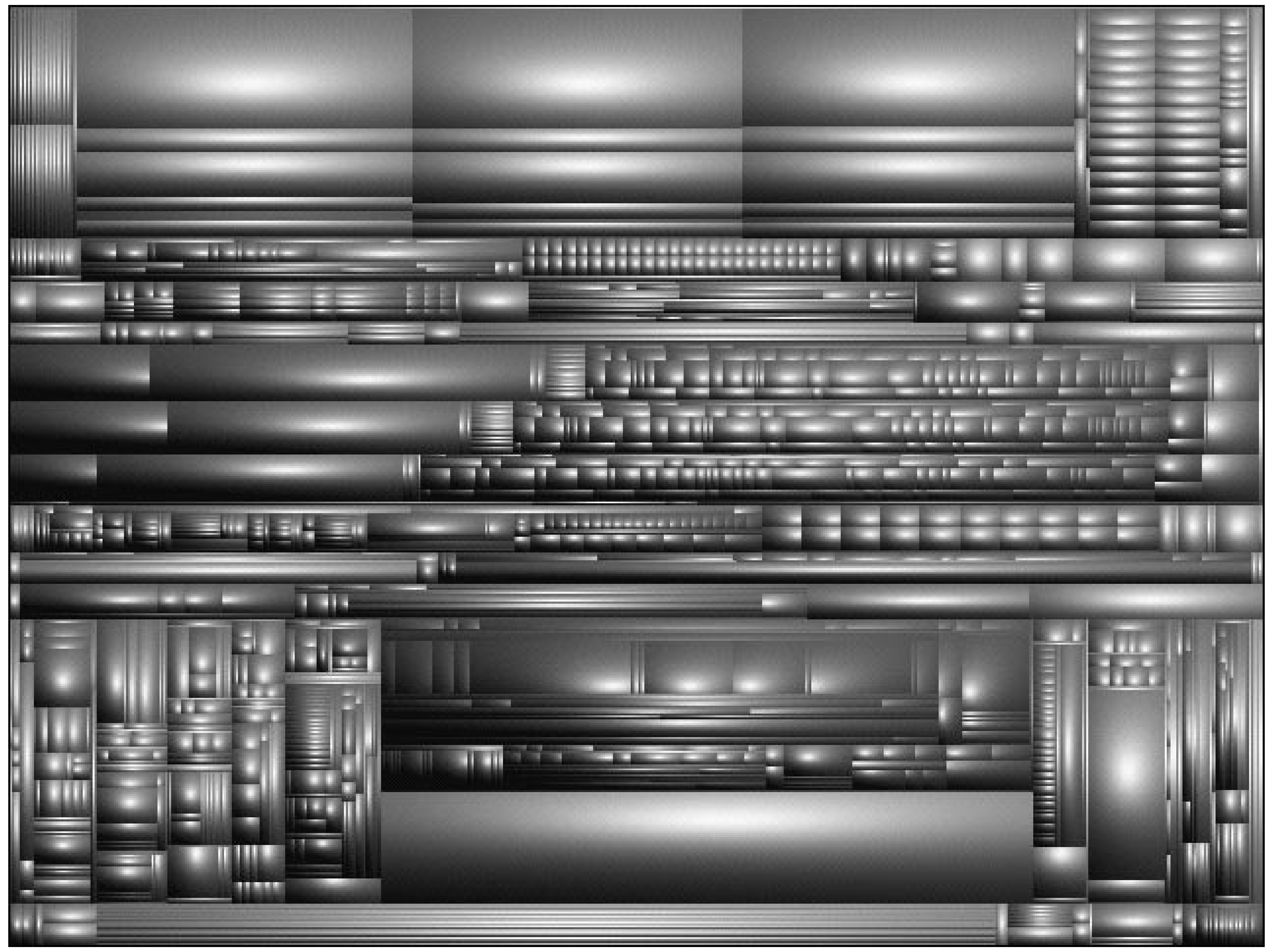
Squarified Layout

- Sort values
- Switch orientation whenever necessary to obtain best aspect ratios

Improving Treemaps (Cushion)

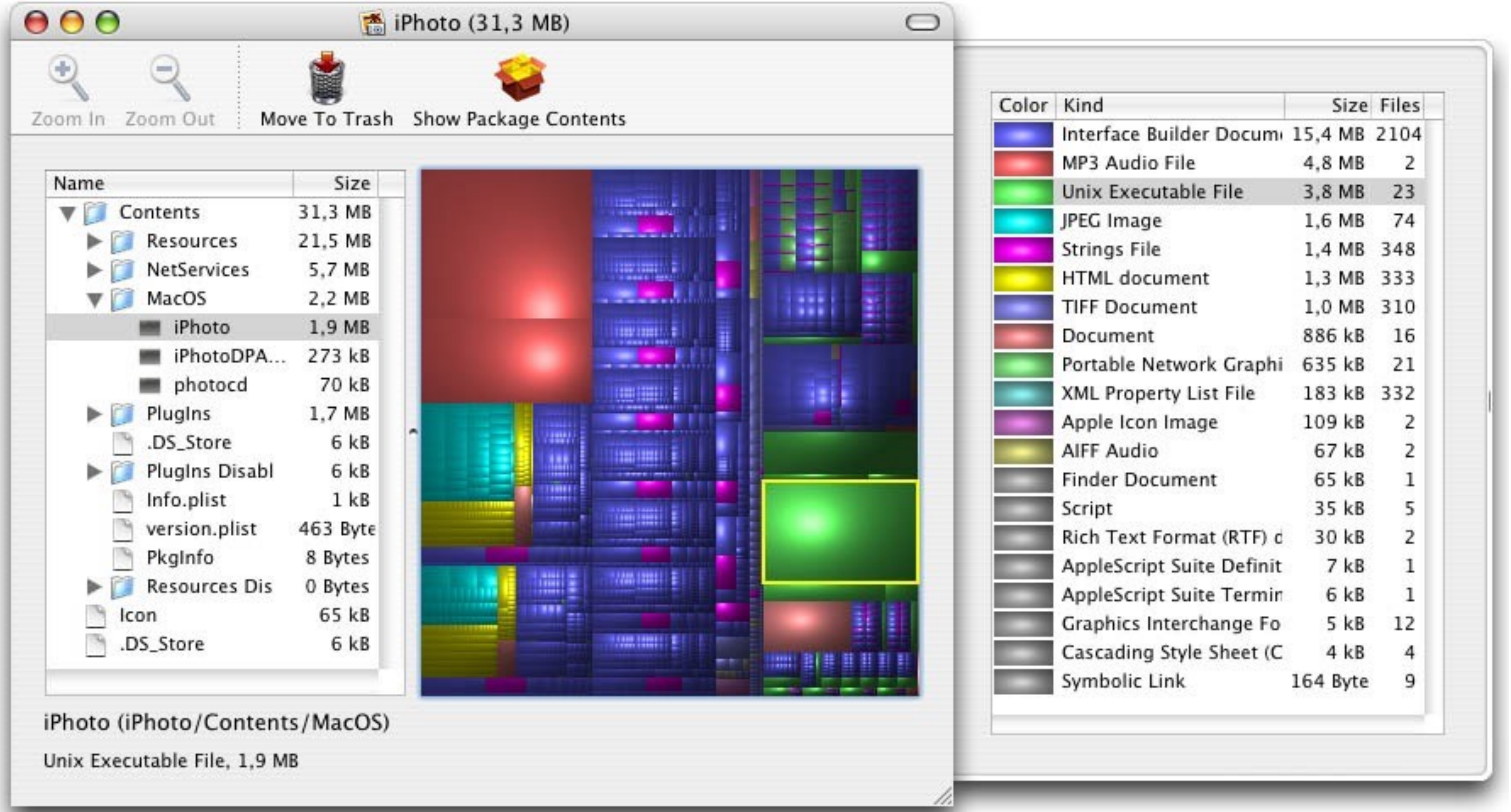
- Leaves are ok, but it can be difficult to find the hierarchy
- Encode this as shading information
- More effective to understand hierarchy





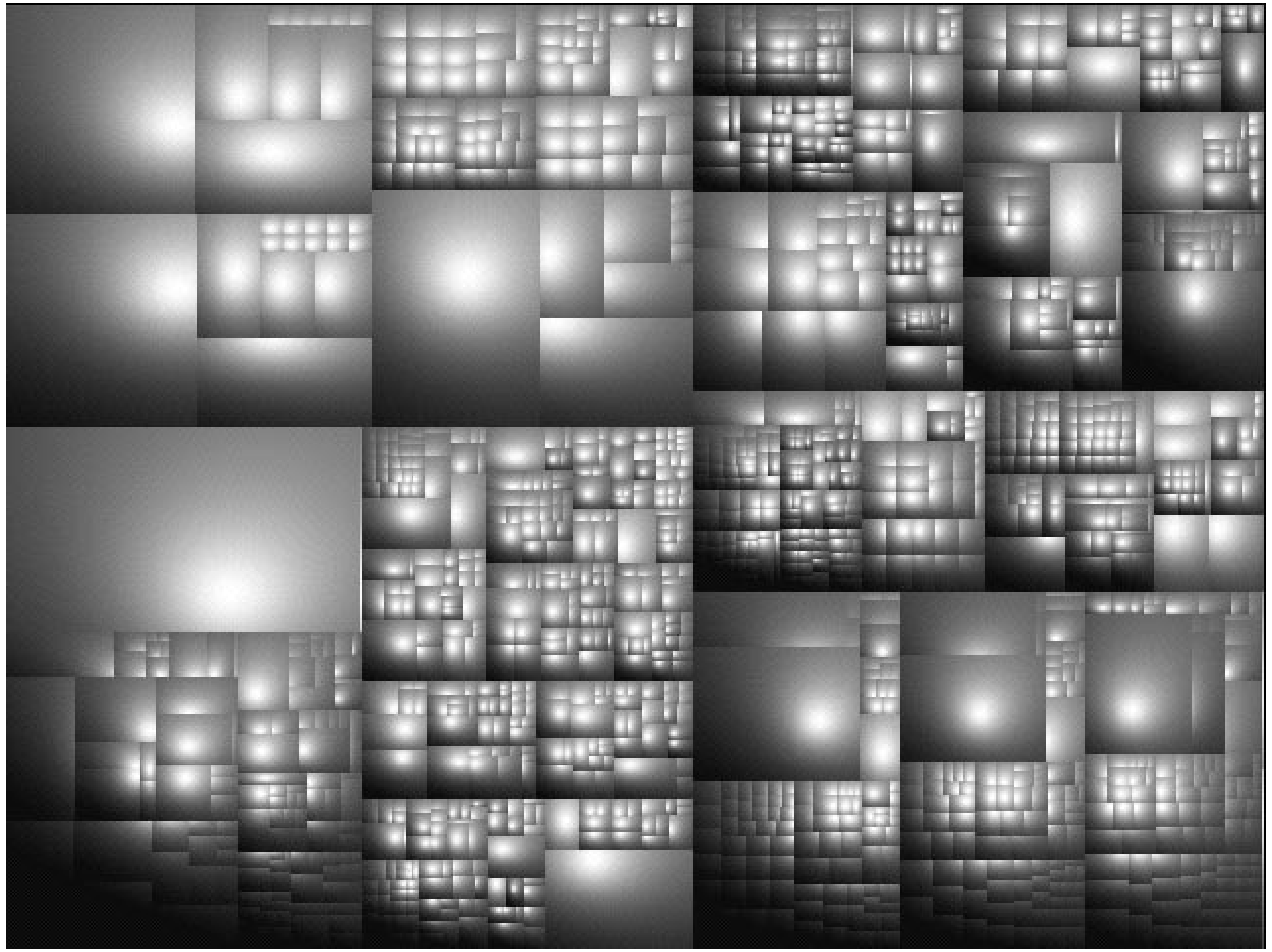
[van Wijk and van de Wetering, 1999]

Disk Inventory

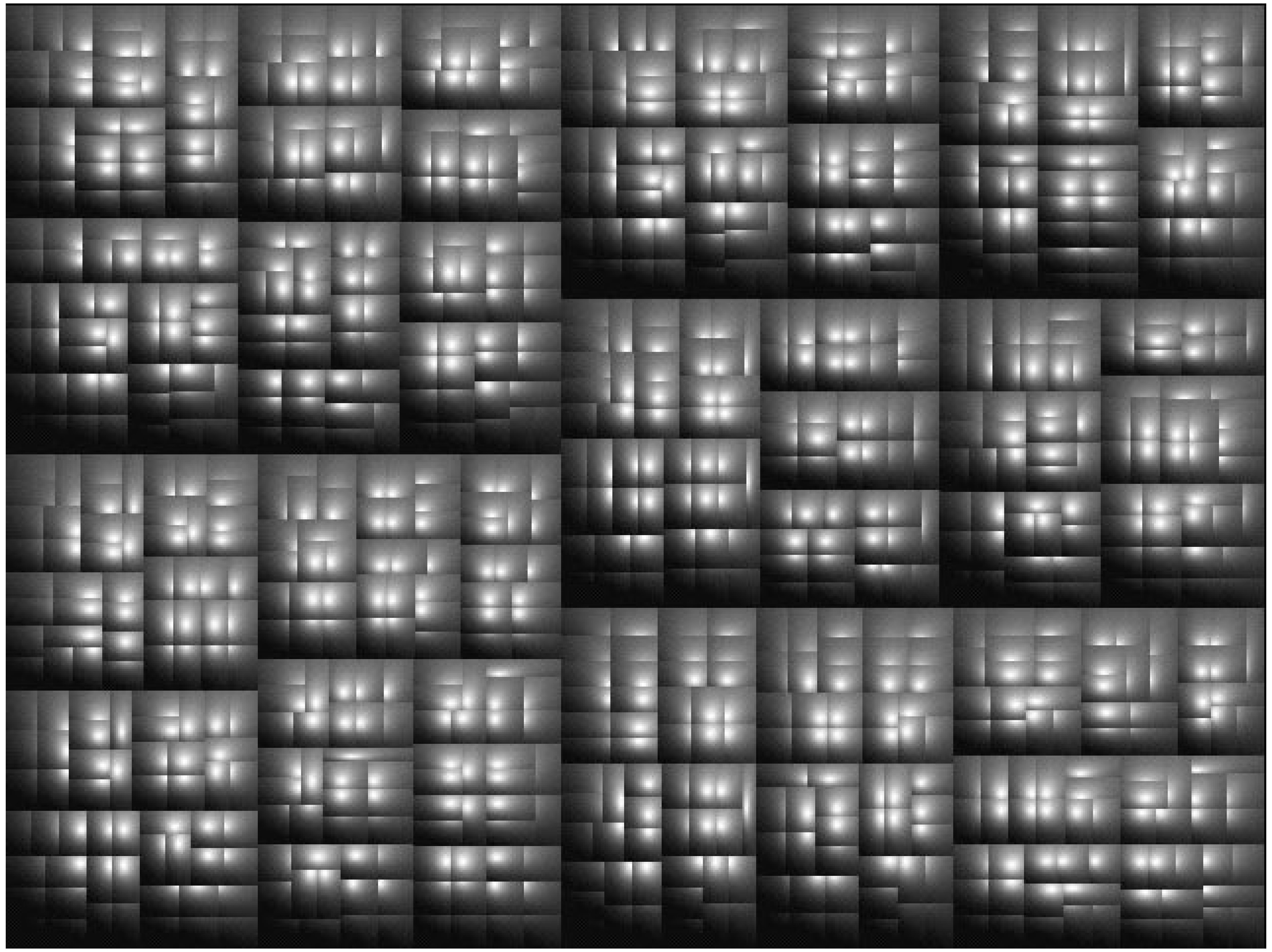


[Disk Inventory X]

Squarified + Cushioned Treemaps



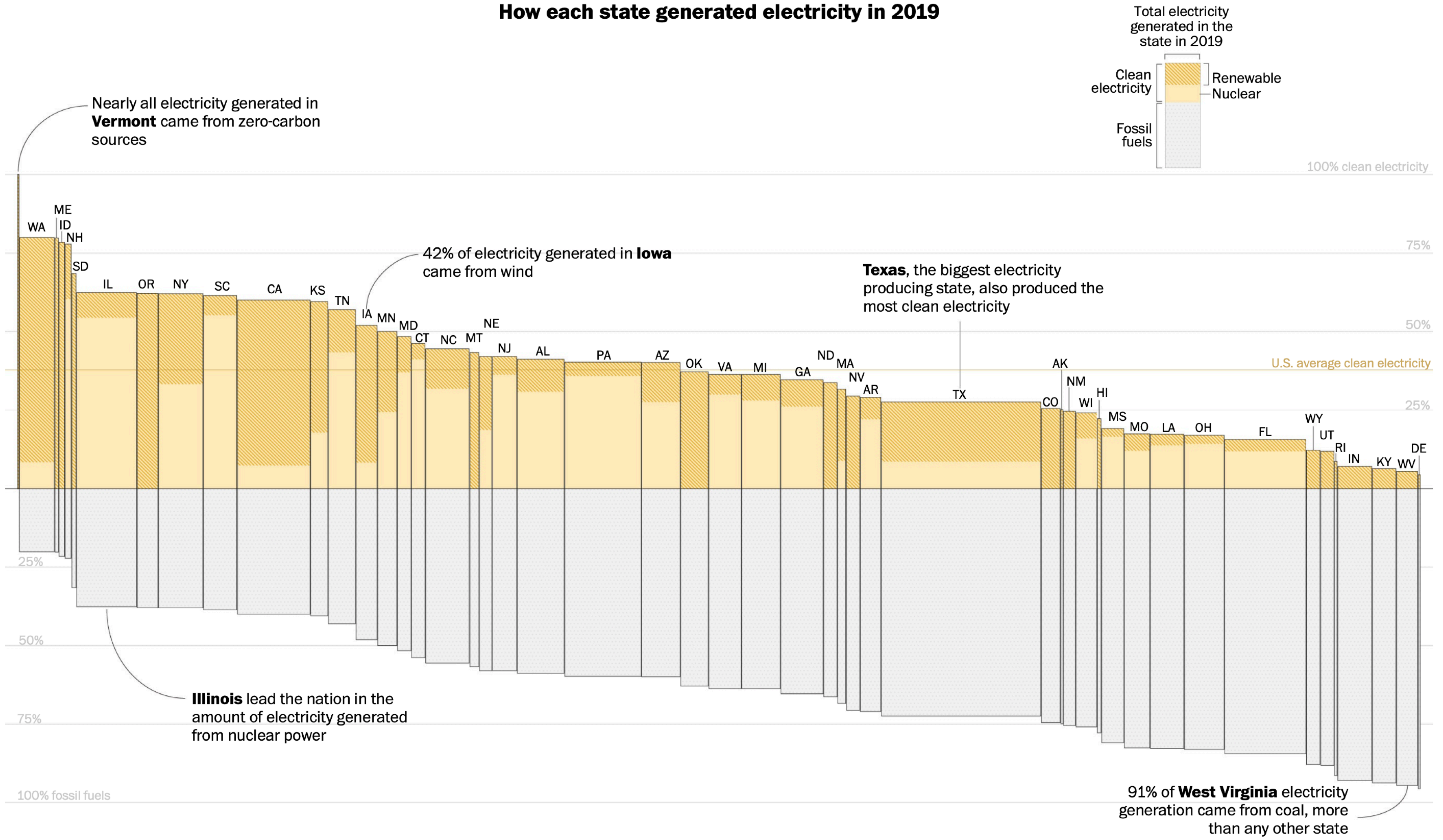
(a) File system



(b) Organization

[Brus et al., 1999]

Variations: Marimekko Chart



[J. Muyskens, Washington Post]

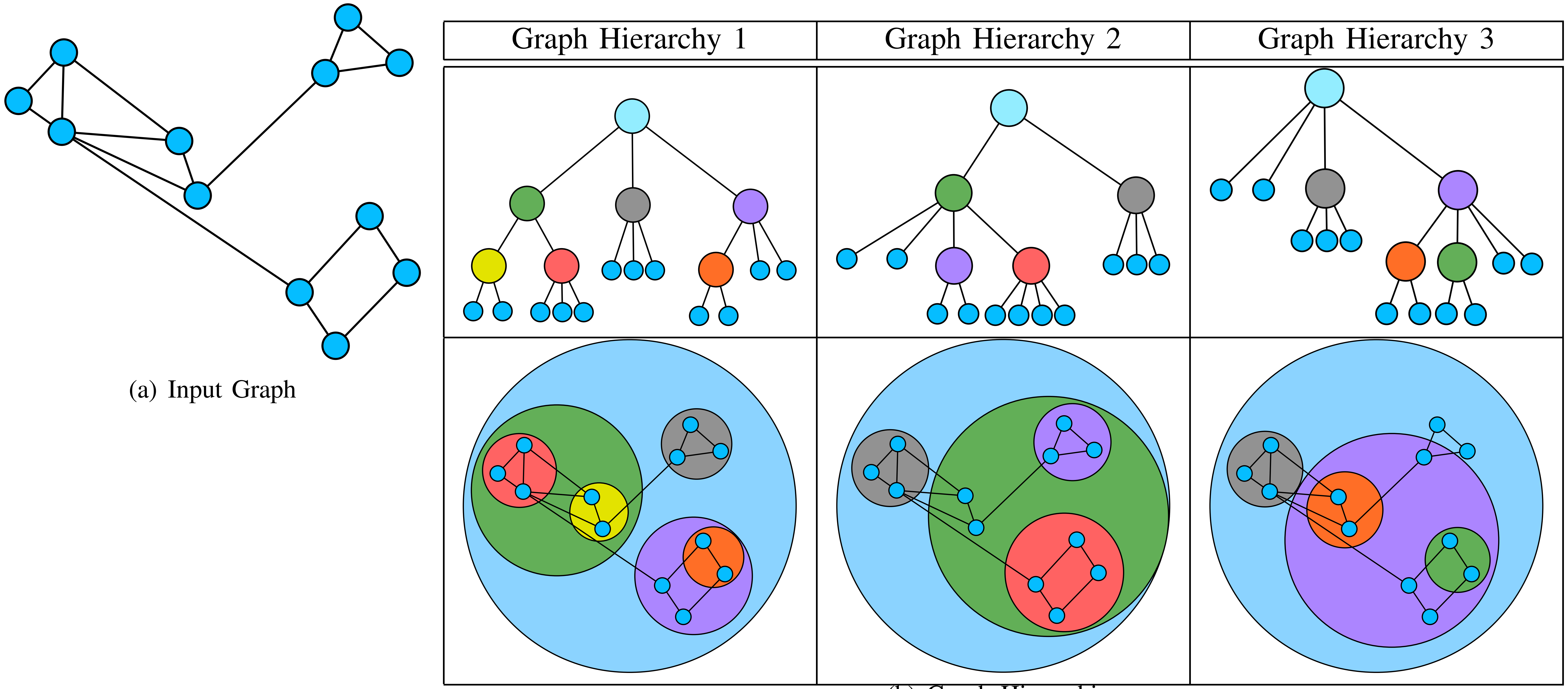
Nested Circles

- Looks more like cluster diagram, but shows hierarchy
- Containment shown by the layering of semi-transparent circles
- Labeling becomes more difficult

[Bostock, 2012]

Compound Networks

- Add a hierarchy to the network (e.g. from clustering)
- GrouseFlocks: uses nested circles with colors



(a) Input Graph

(b) Graph Hierarchies

[Archambault et al., 2008]