

# Data Visualization (CSCI 627/490)

---

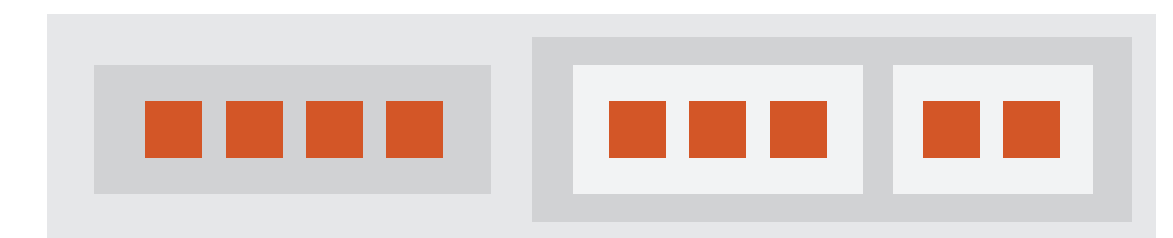
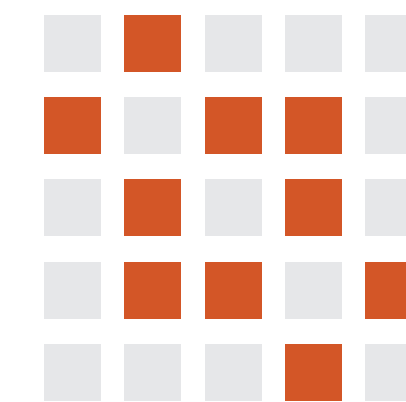
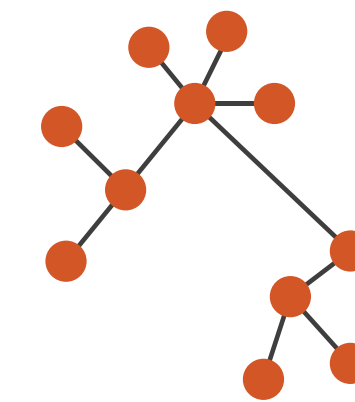
Trees

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**

# Arrange Networks and Trees



[Munzner (ill. Maguire), 2014]

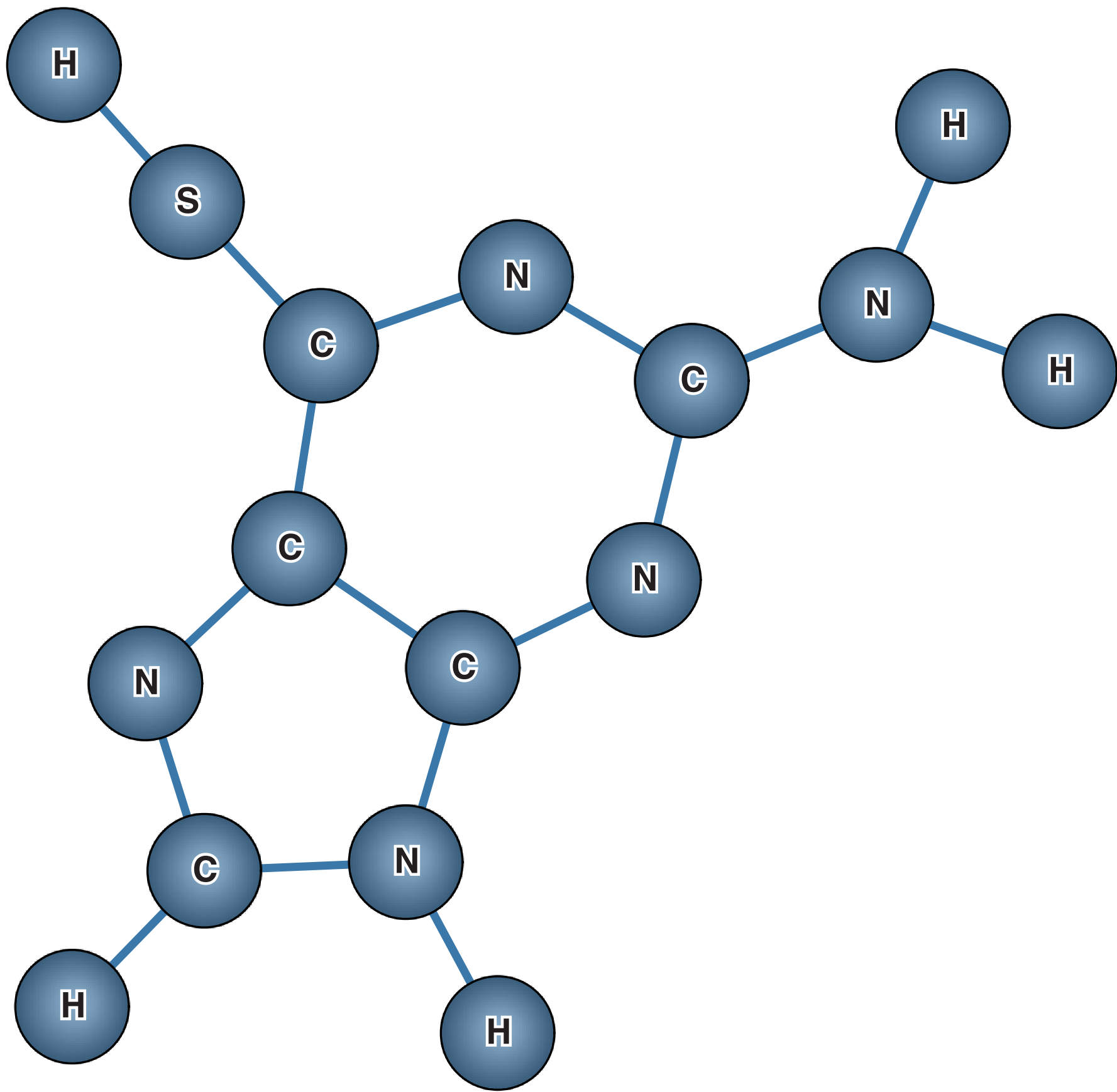
# Network Data Represented in Tables

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



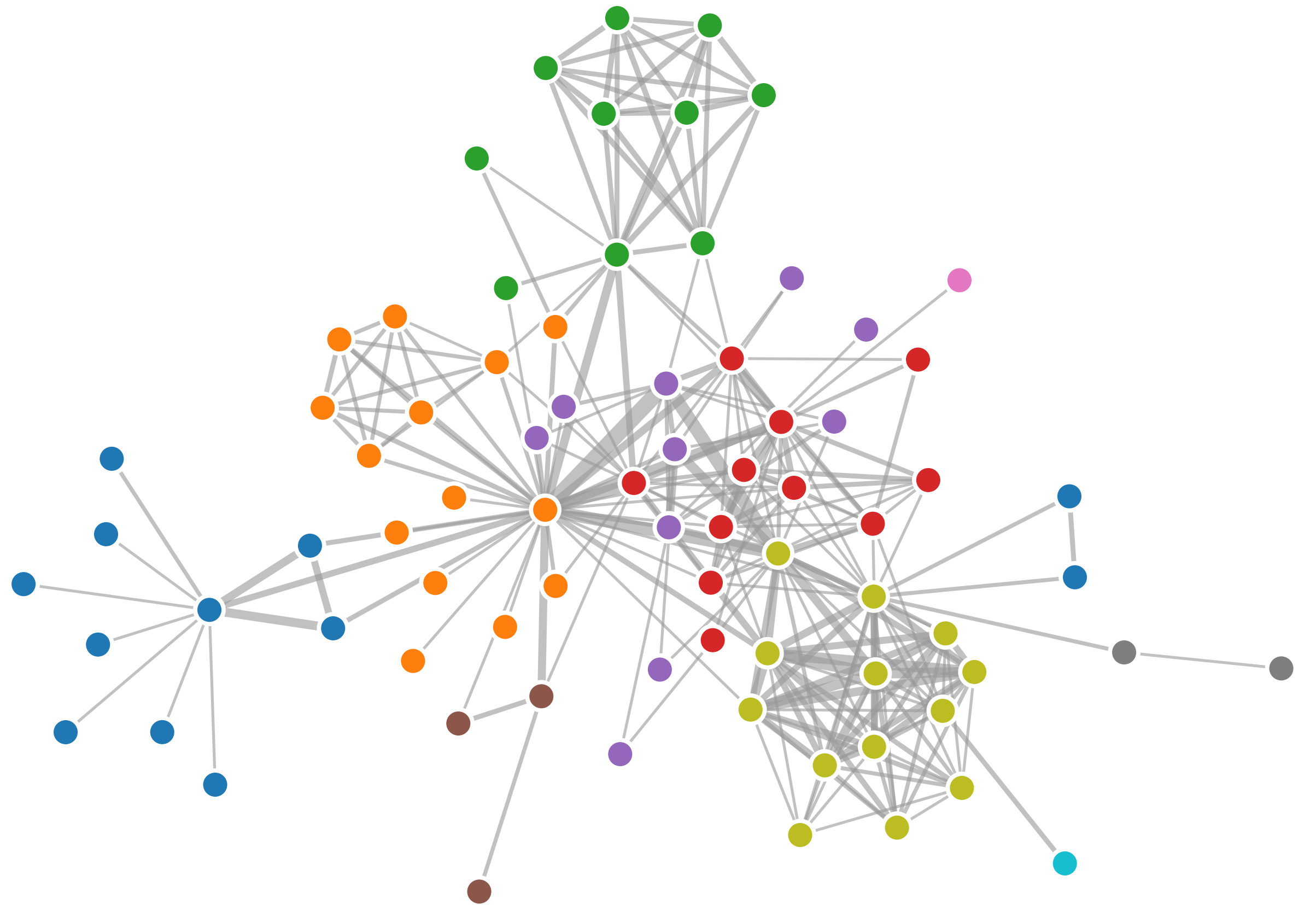
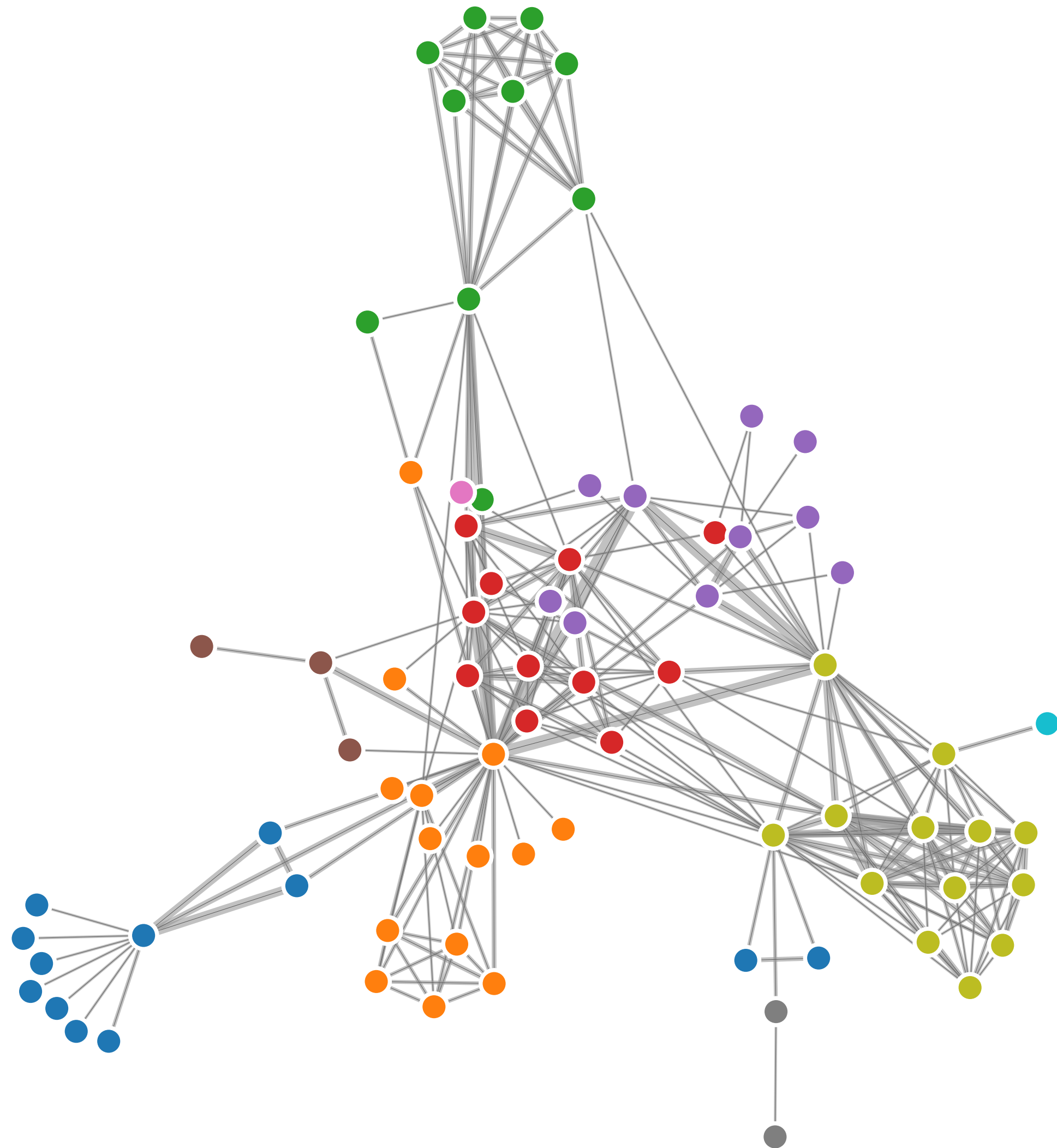
# Networks Need Layouts!

---

- 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



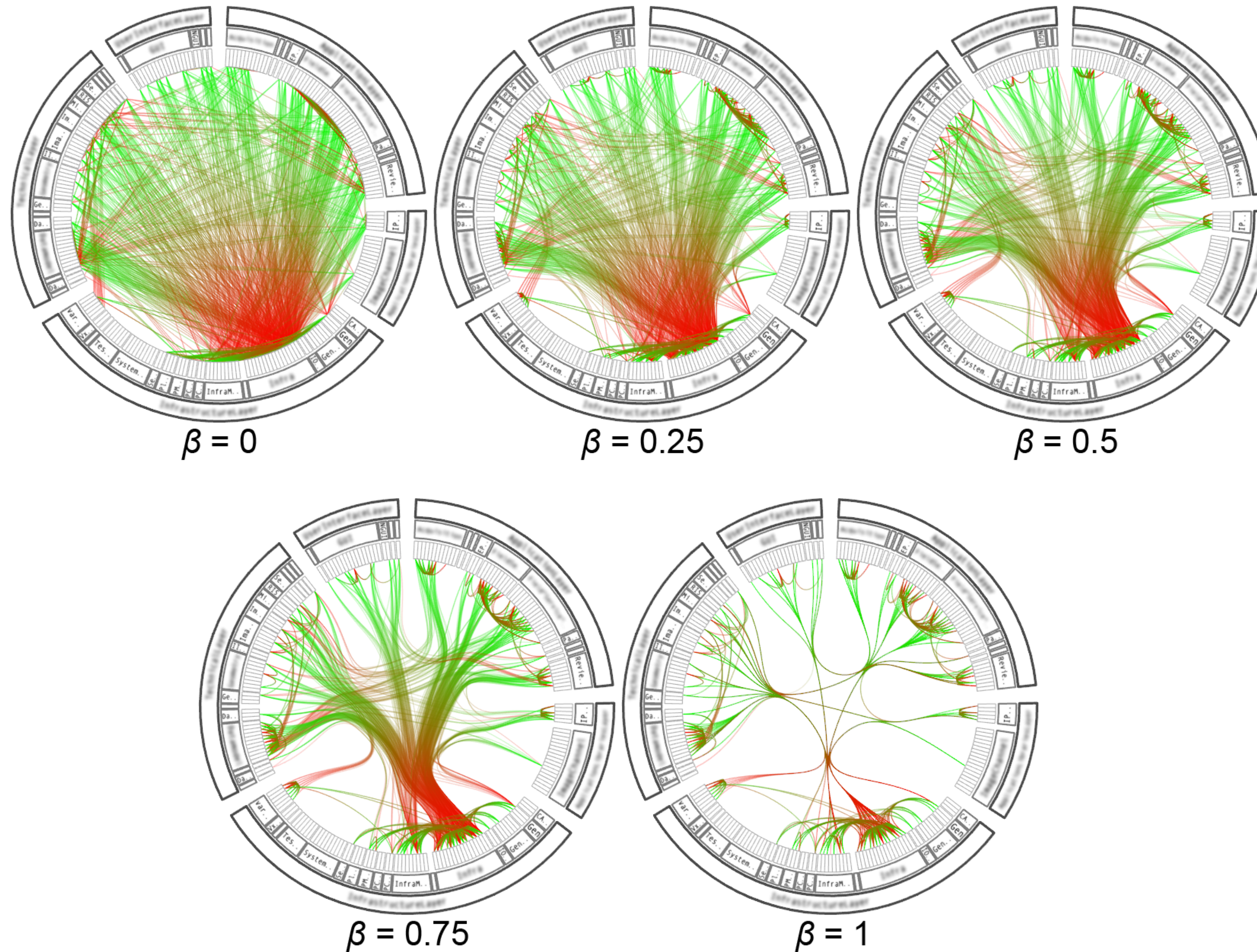
# Layout Algorithms



[Force-Directed and CoLa, M. Bostock]



# Bundling Strength

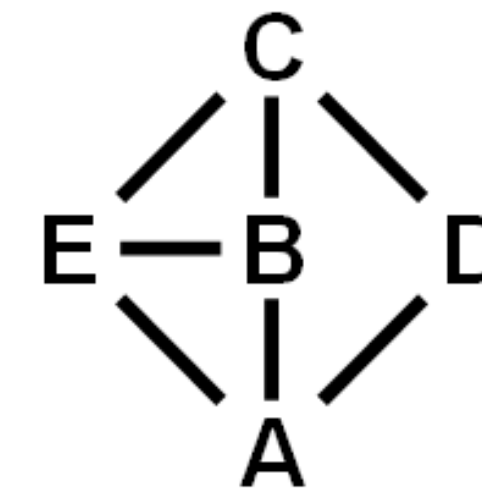


[Holten, 2006]



# Adjacency Matrix

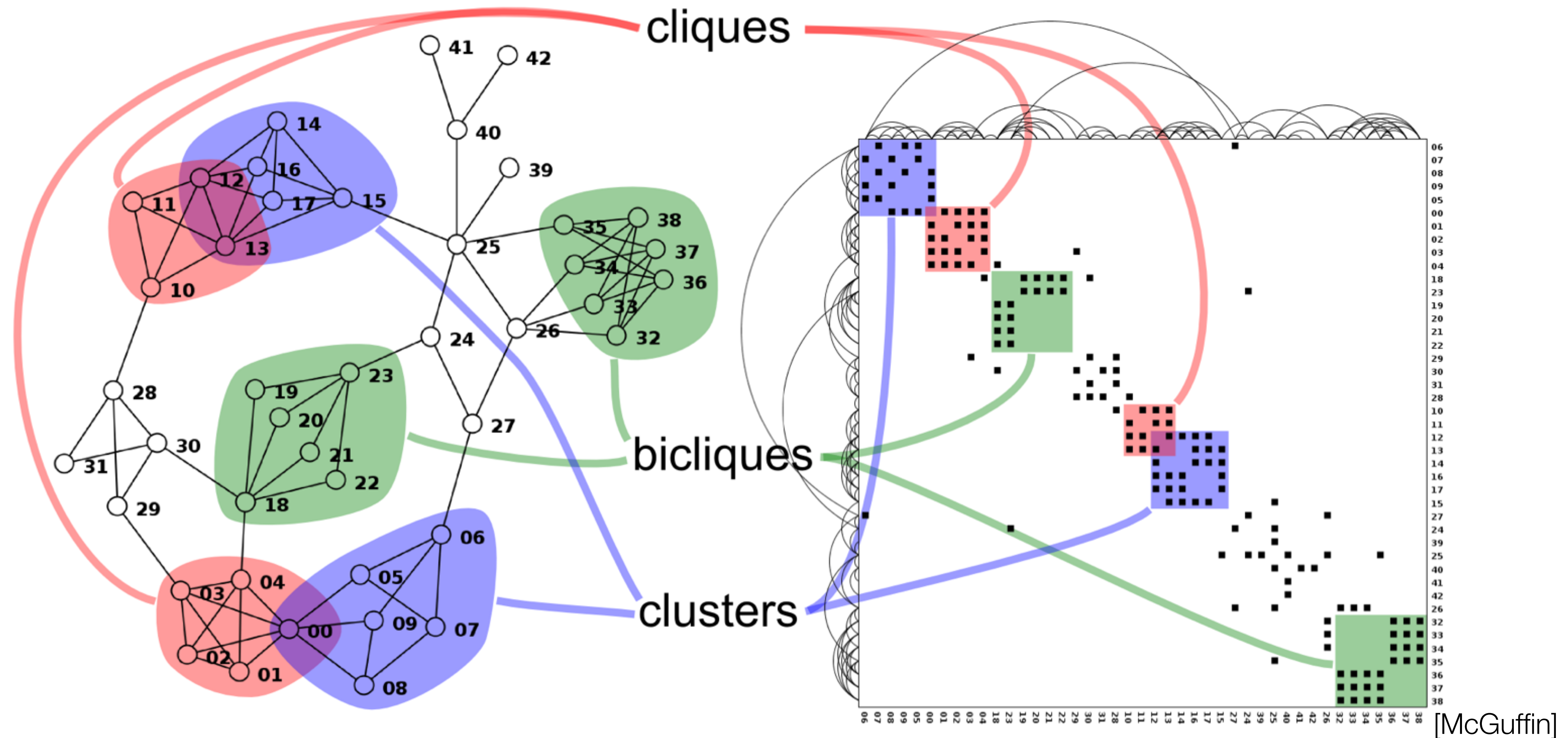
- Change network to tabular data and use a matrix representation
- Derived data: nodes are keys, edges are boolean values
- Task: lookup connections, find well-connected clusters
- Scalability: millions of edges
- Can encode **edge weight**, too



	A	B	C	D	E
A	A				
B		B			
C			C		
D				D	
E					E

[Henry et al., 2007]

# 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

# Project

---

- Working through grading these to provide feedback
- Initial Feedback
  - Some tasks are not tasks
  - Some tasks are technically tasks but are phrased in terms of a visualization
  - Think about the question "Why would someone care?"
- Example: Is there a correlation between the season and types of storms in regions?
  - Who cares?
  - Why do they care?
  - Are there specific instances where we can see how people might use info?

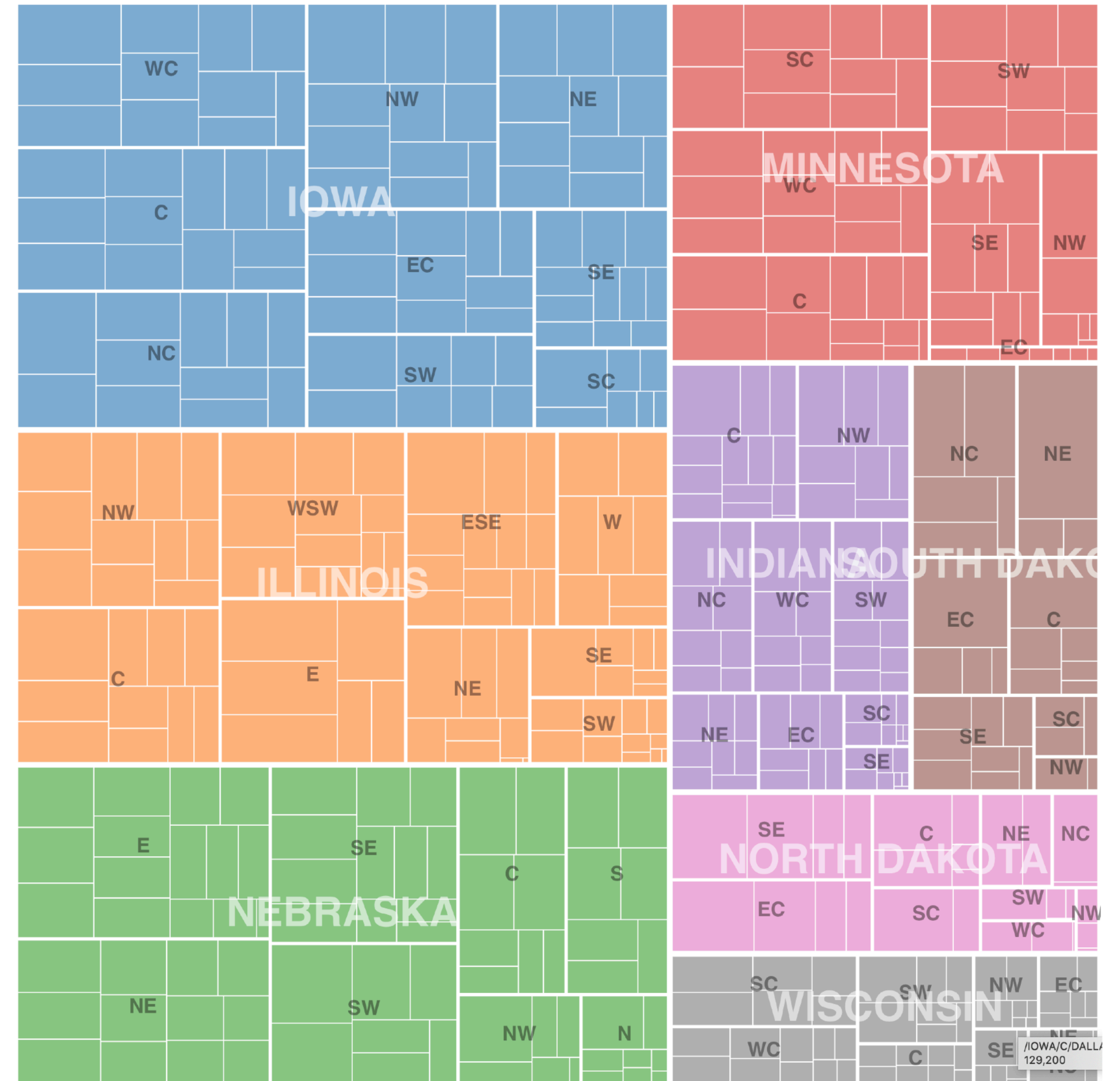
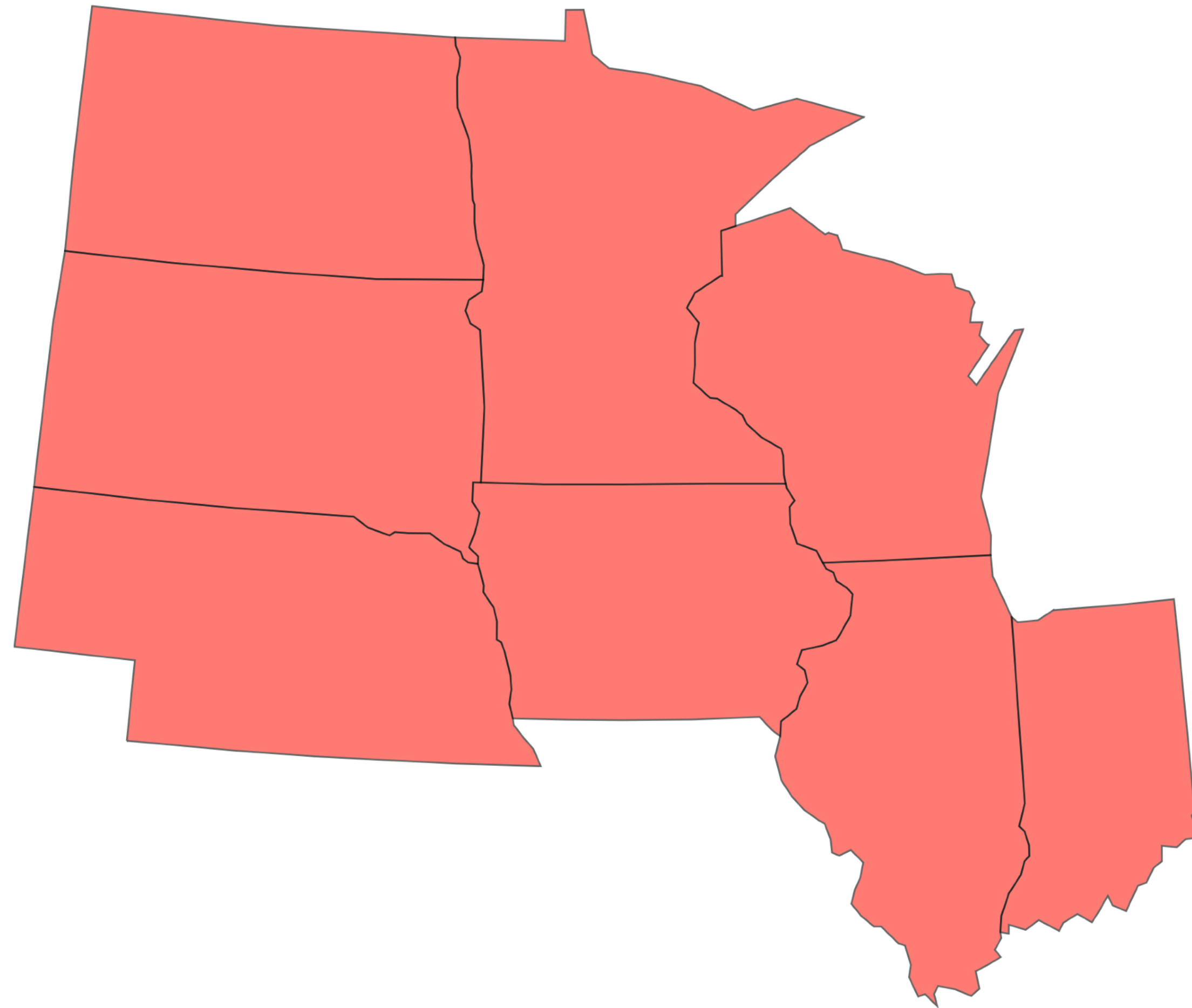


# Project

---

- 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



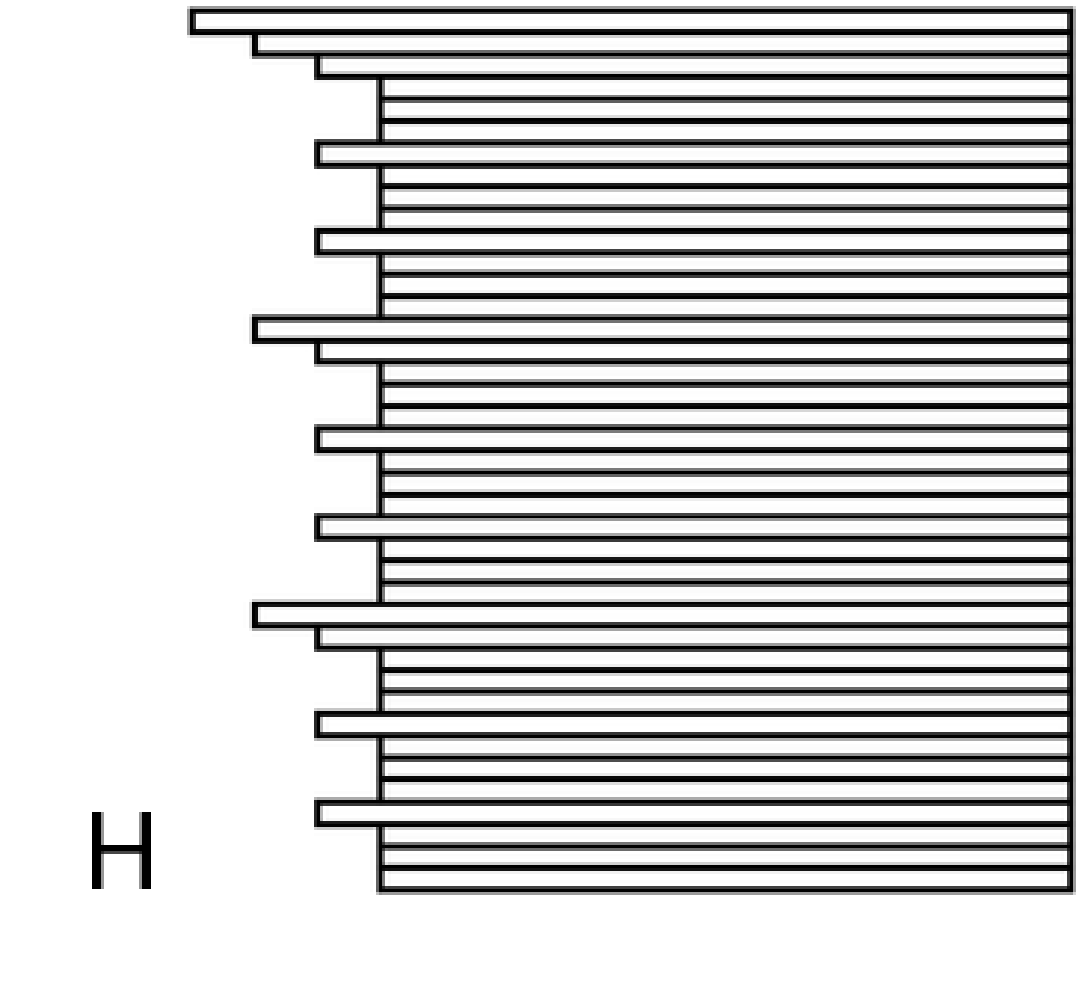
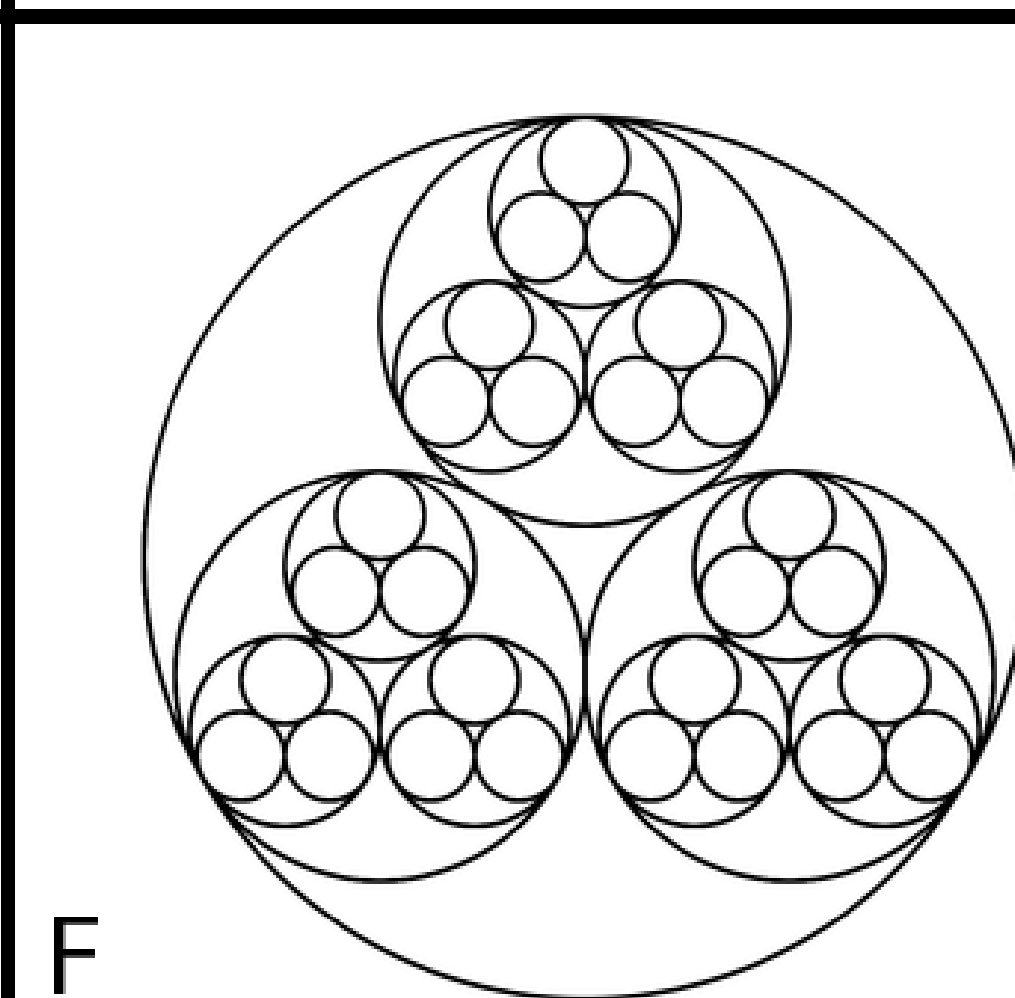
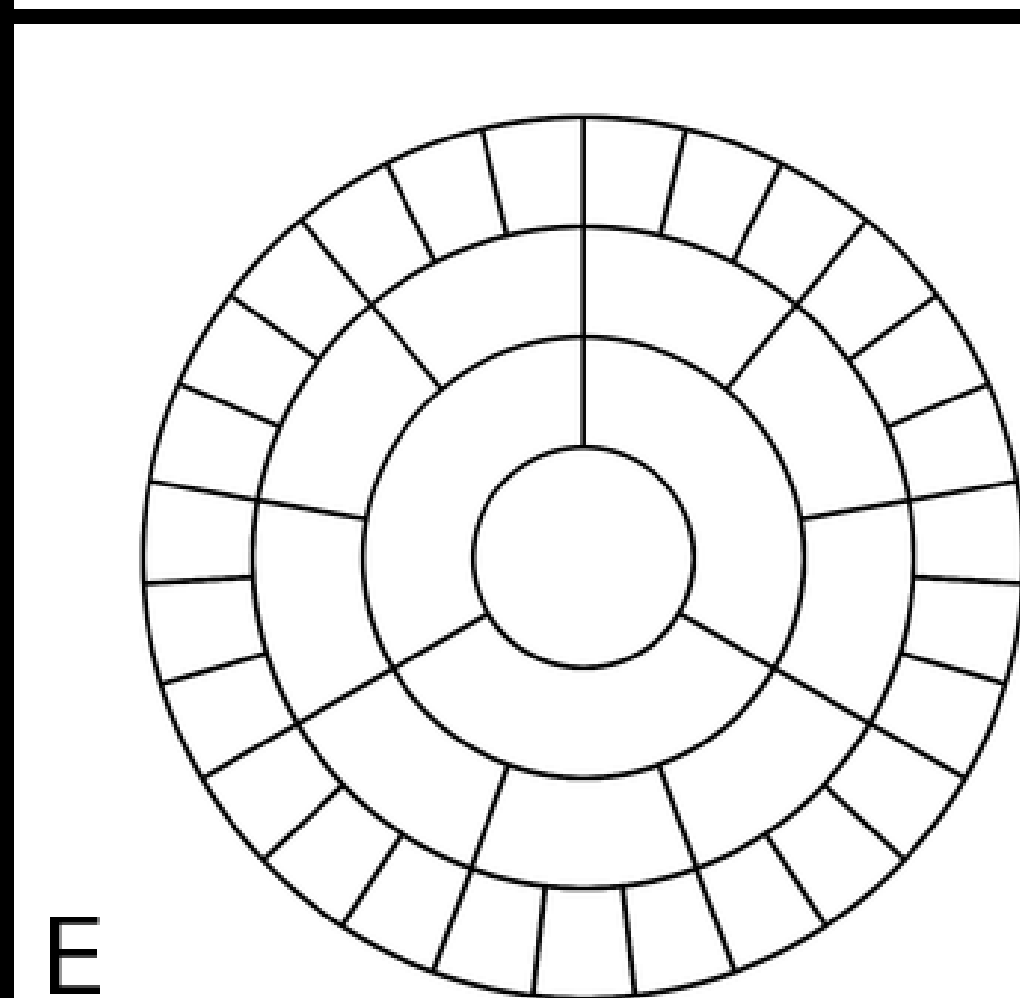
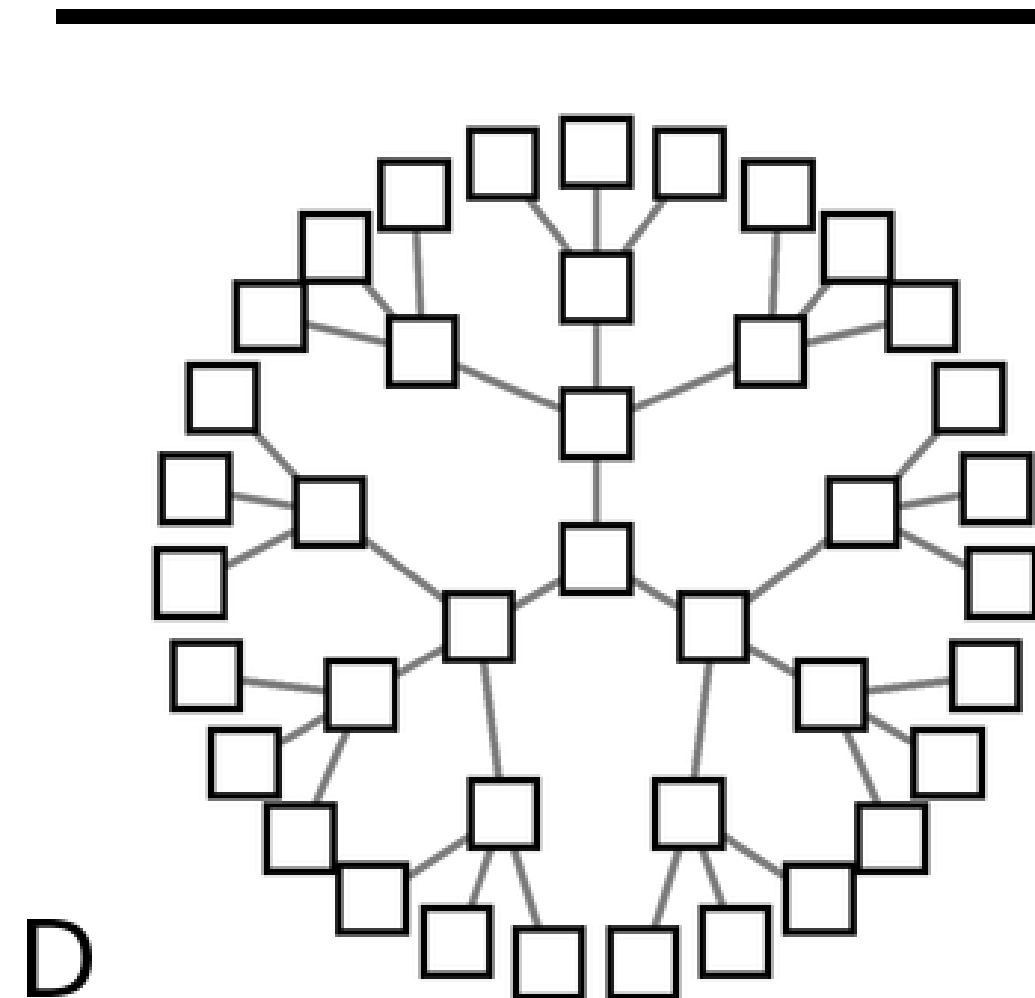
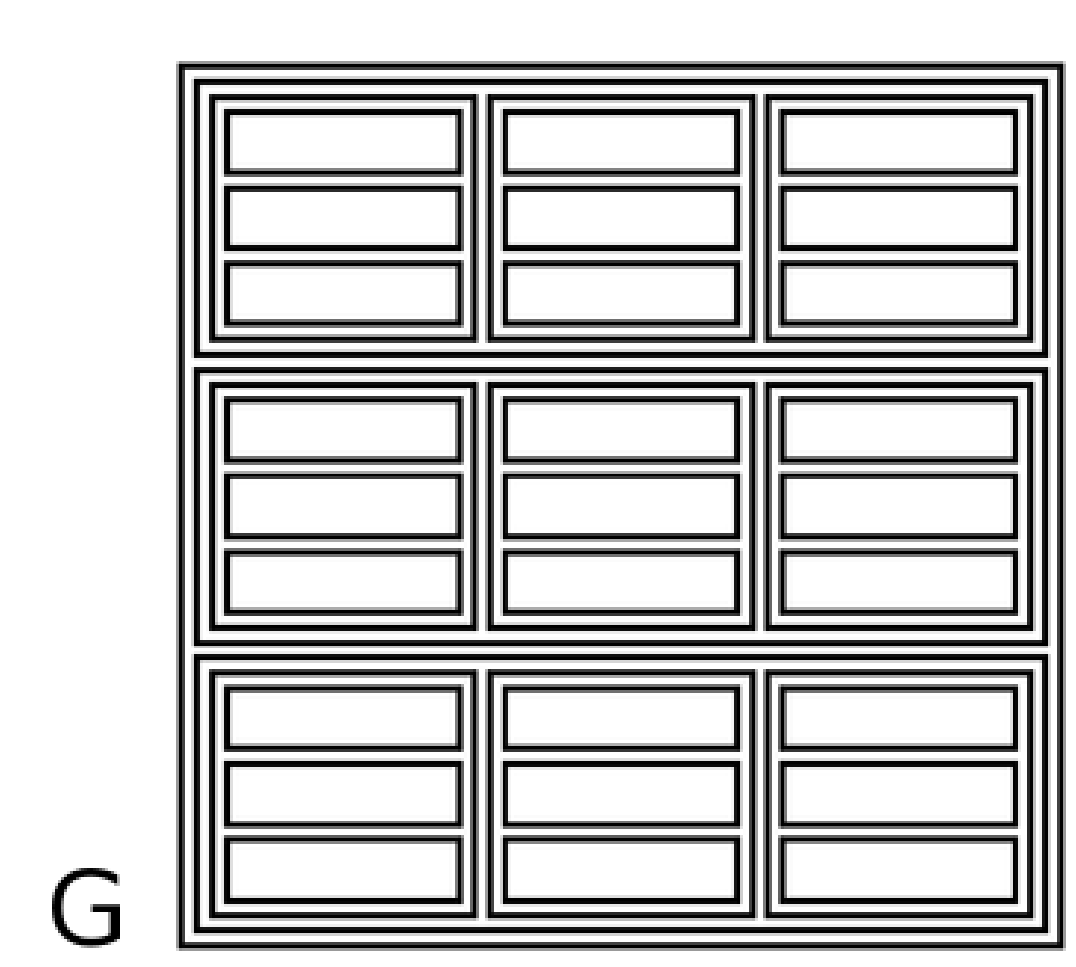
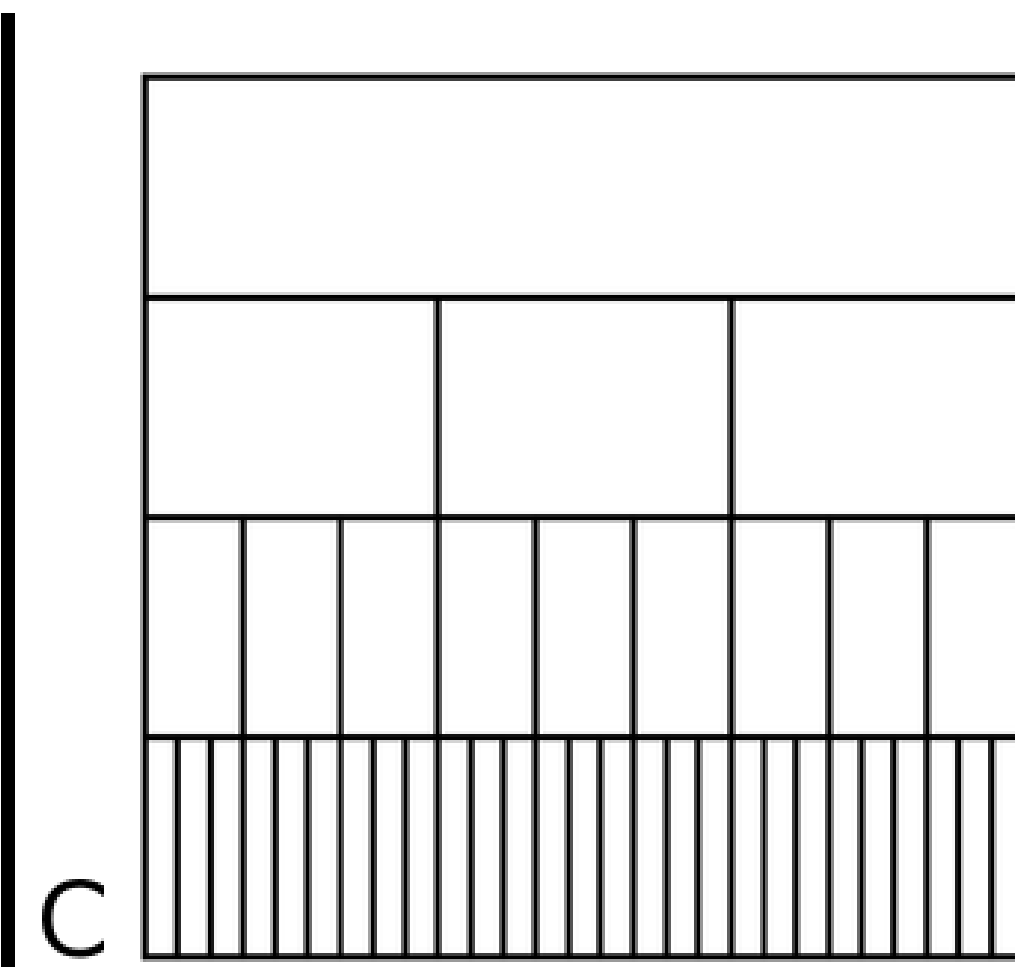
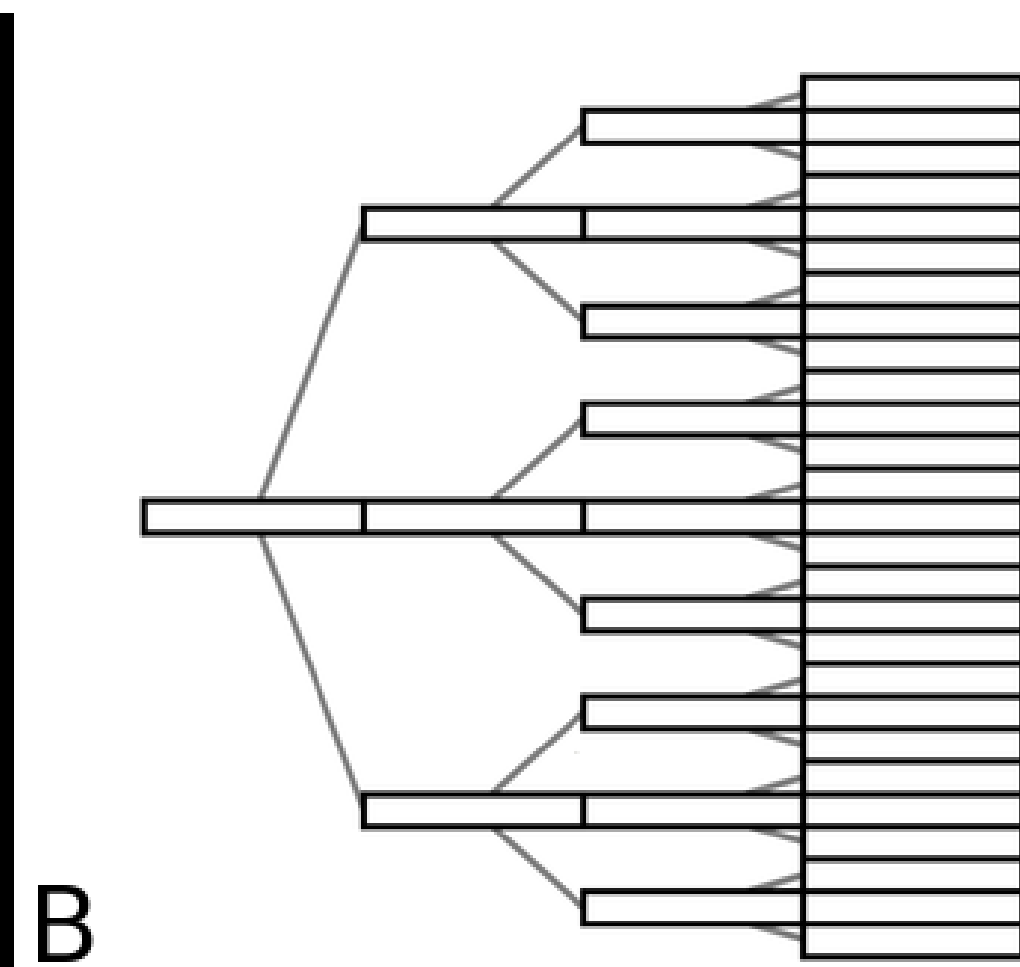
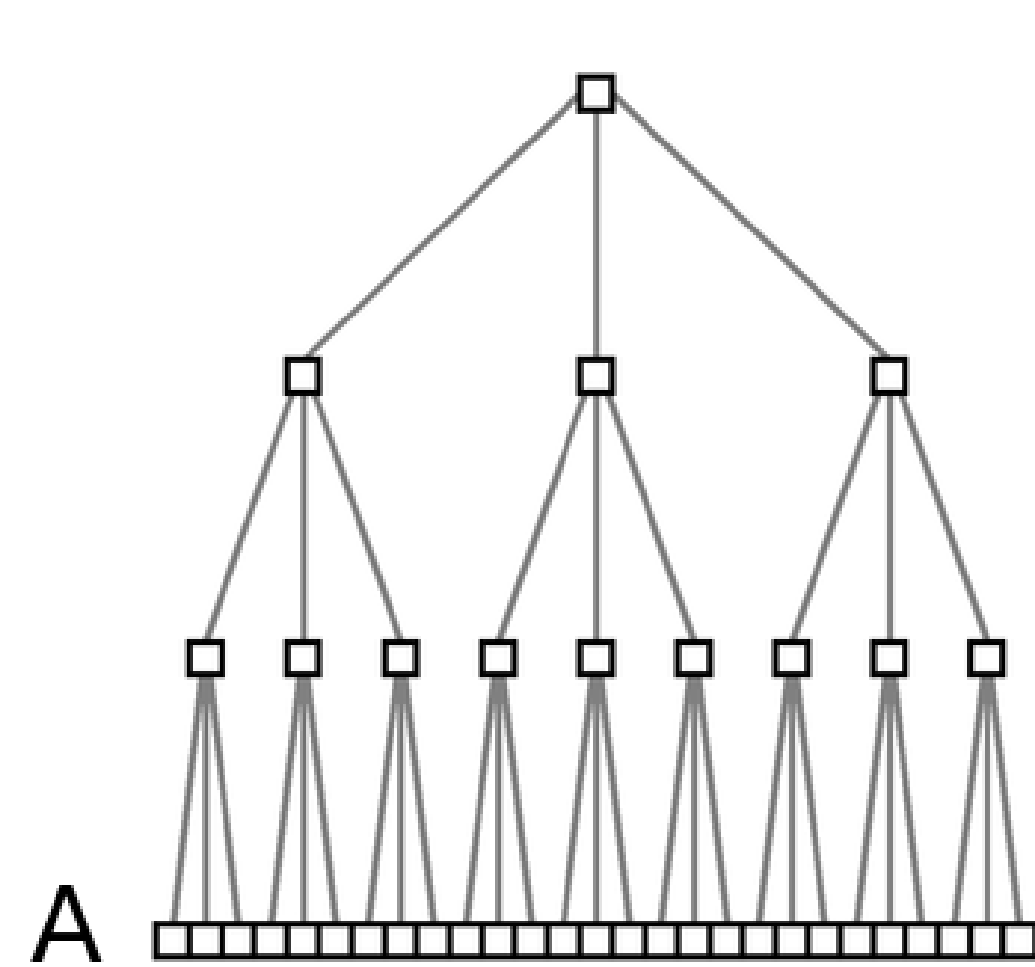


# Trees

---

- 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

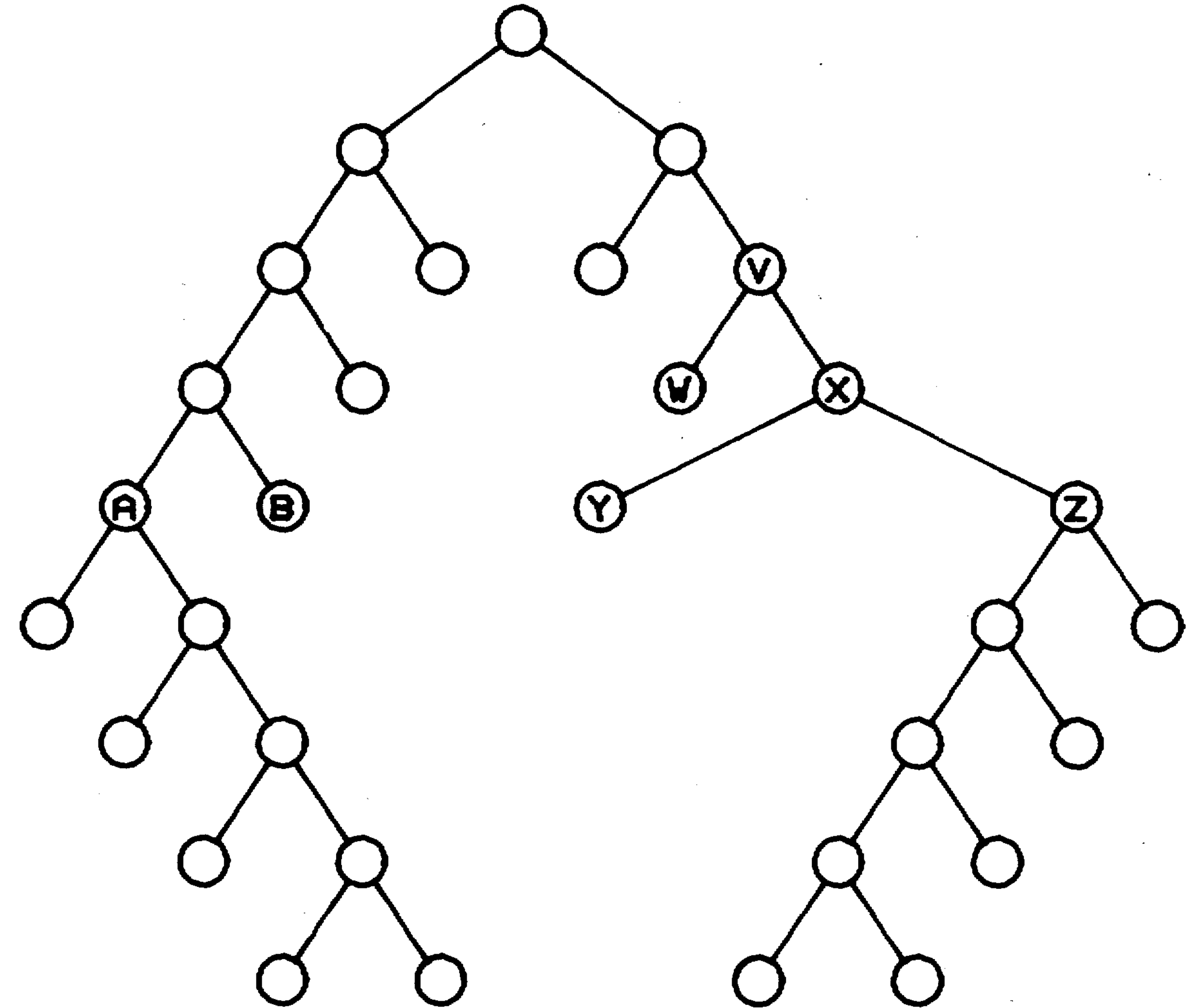
# Tree Visualizations



[McGuffin and Robert, 2010]

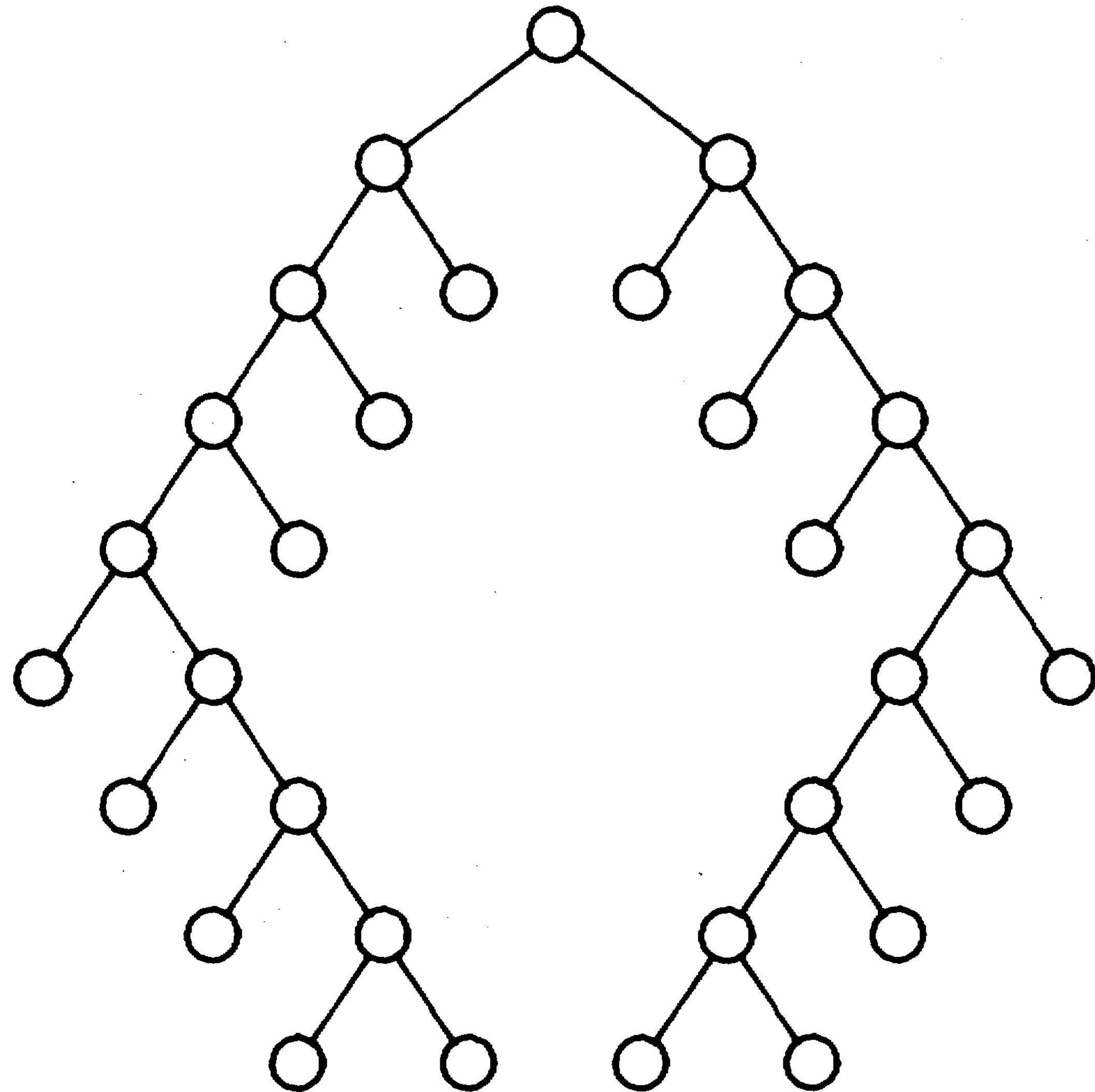
# Node-Link Diagram

- 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

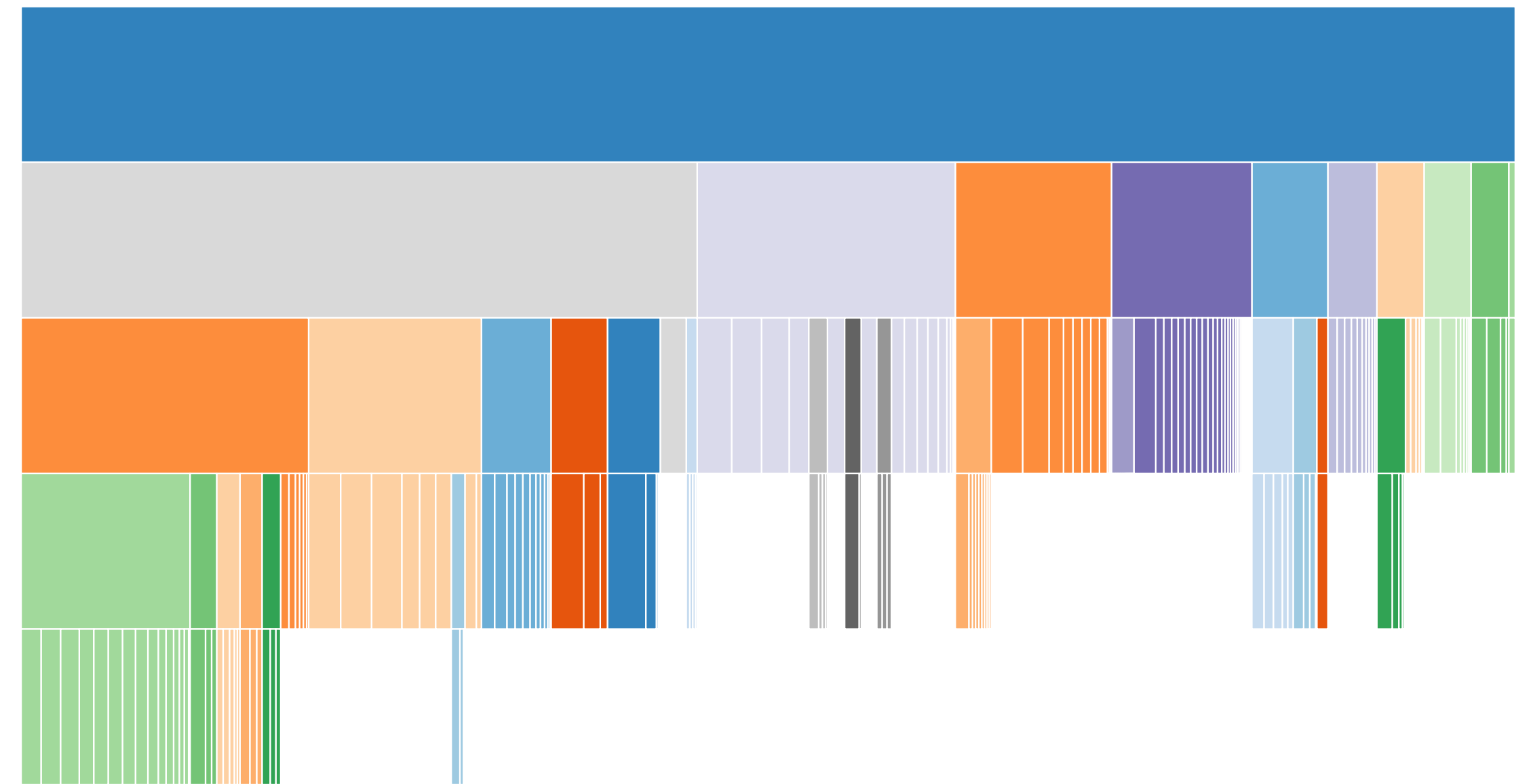


- Recurse on left and right subtrees
- Shift subtree over as long as it doesn't overlap
- Place parent centered above the subtrees
- Originally, only binary trees, extended by Walker

[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

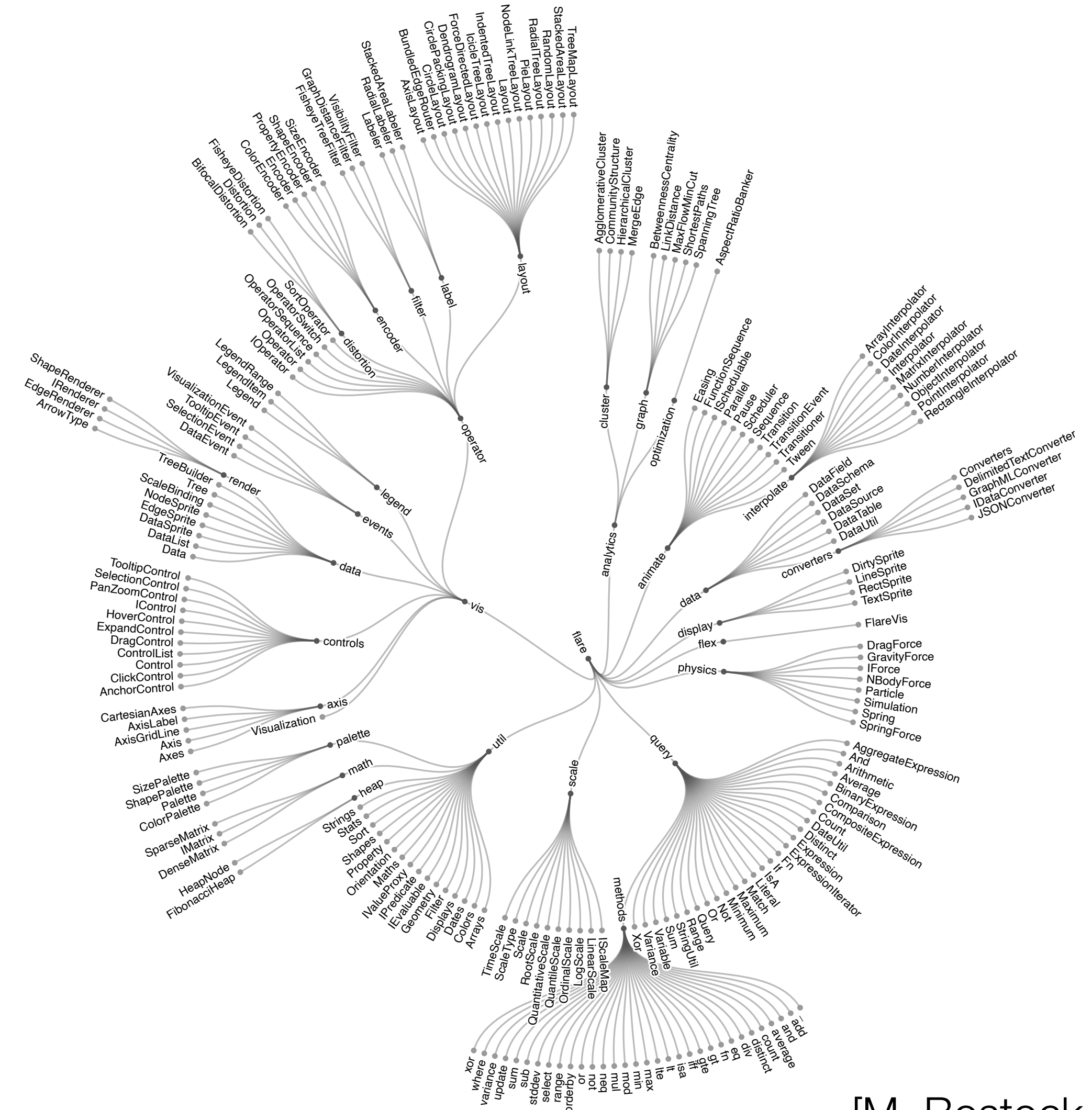


[Bostock, 2011]



# Radial Node-Link

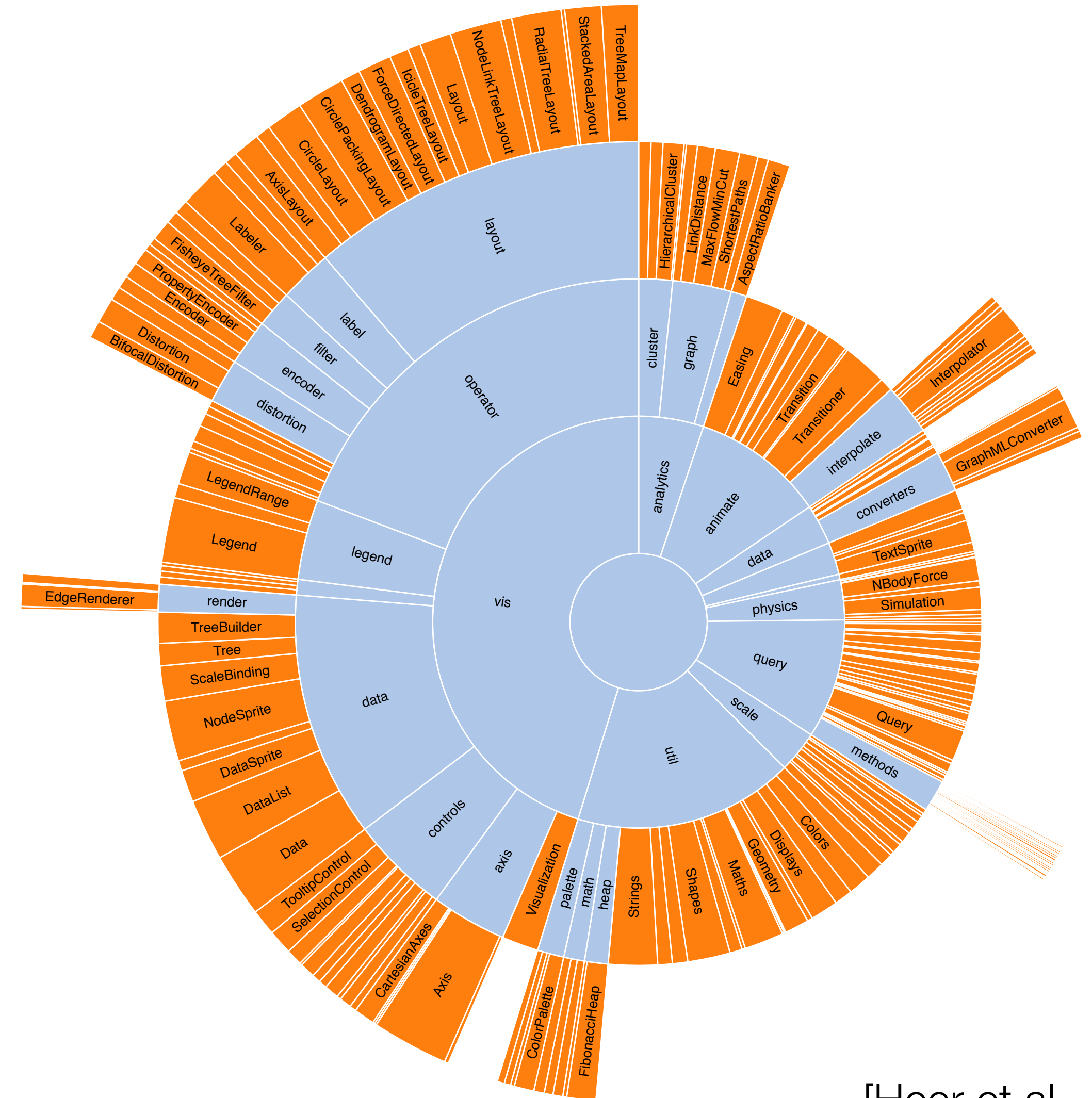
- Use polar coordinates instead of rectilinear
- Same layout algorithms work (e.g. Reingold-Tilford)
- Benefit: space usage, labels



[M. Bostock, 2017]

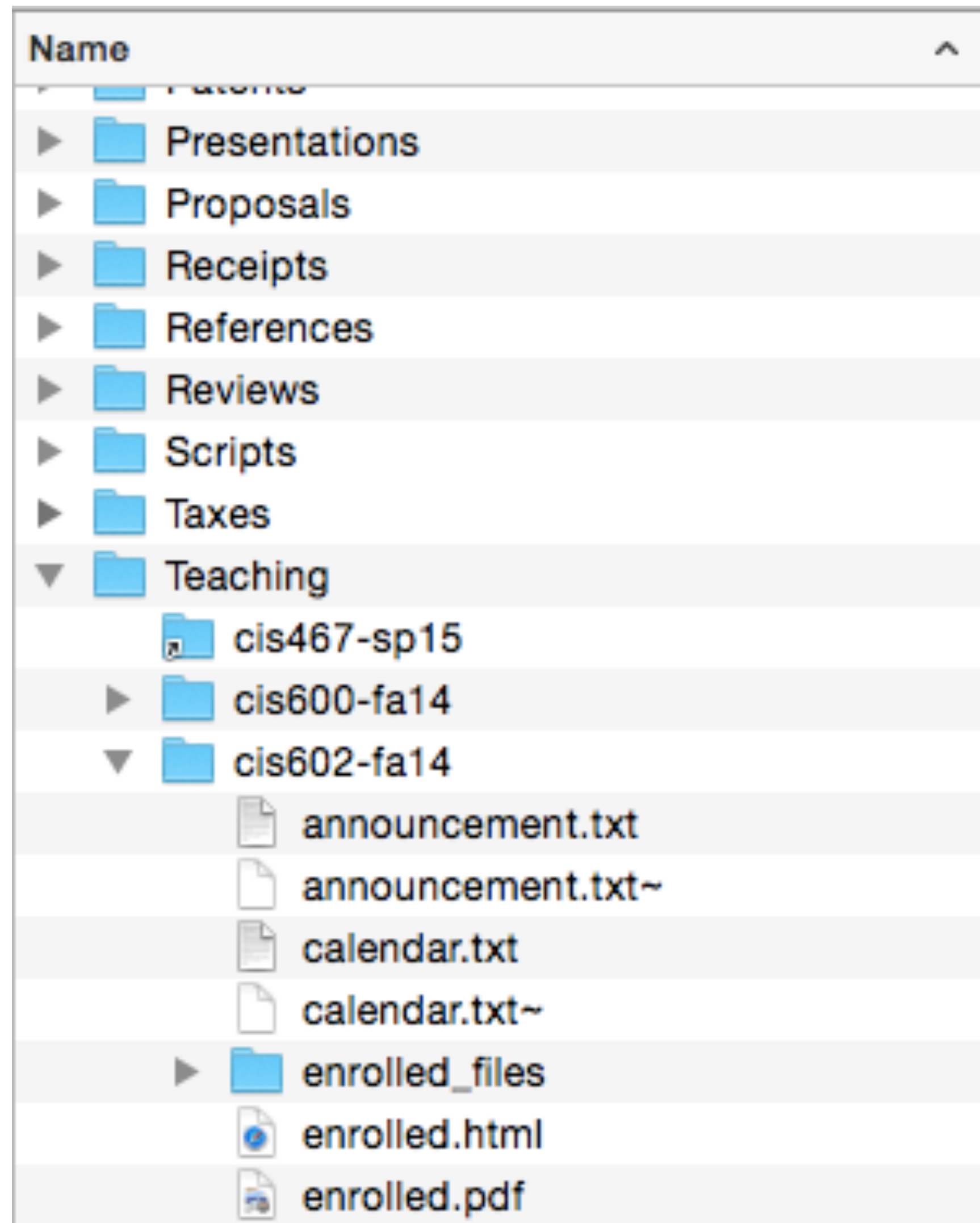
# Sunburst

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



[Heer et al., 2012]

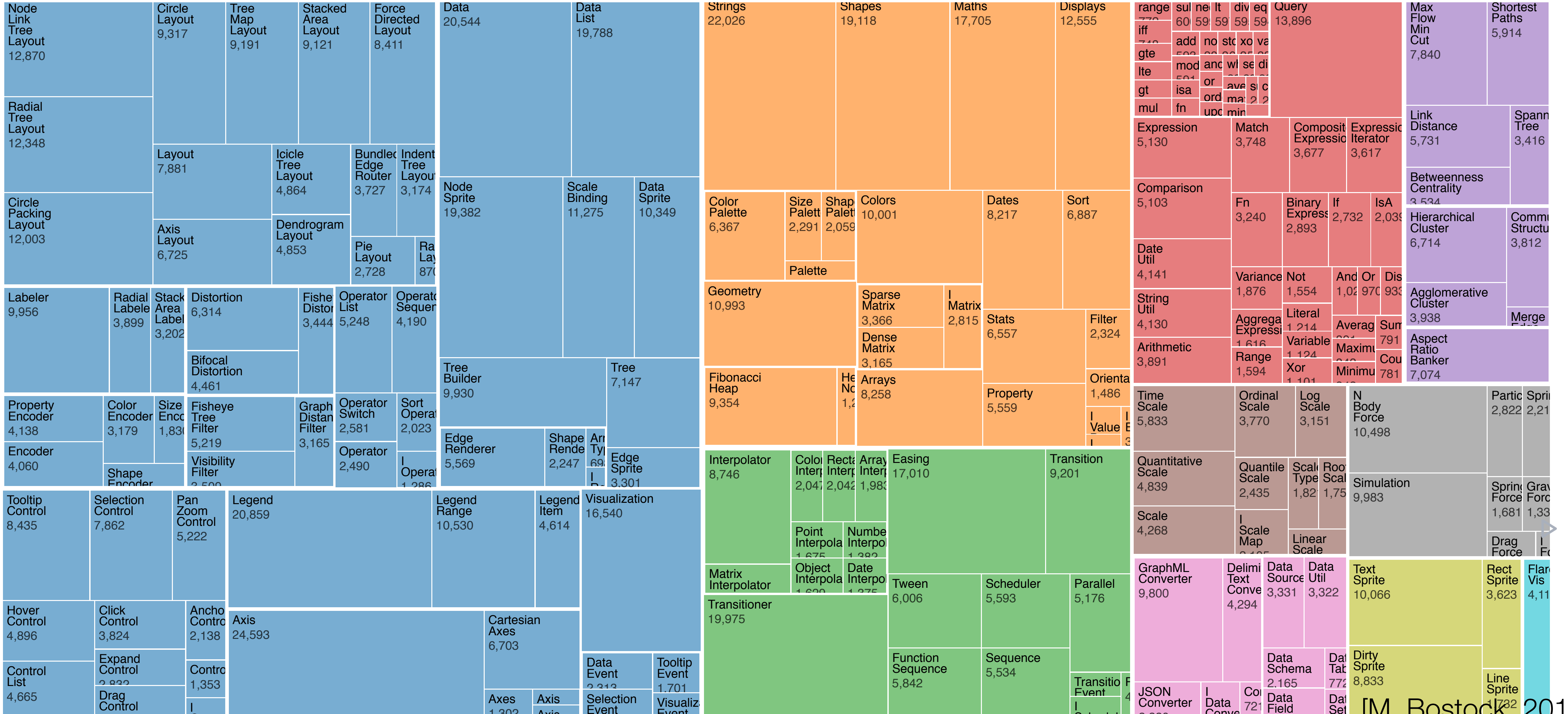
# Indented Outline



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



# Treemap



[M. Bostock, 2017]

# Car/Truck Treemap

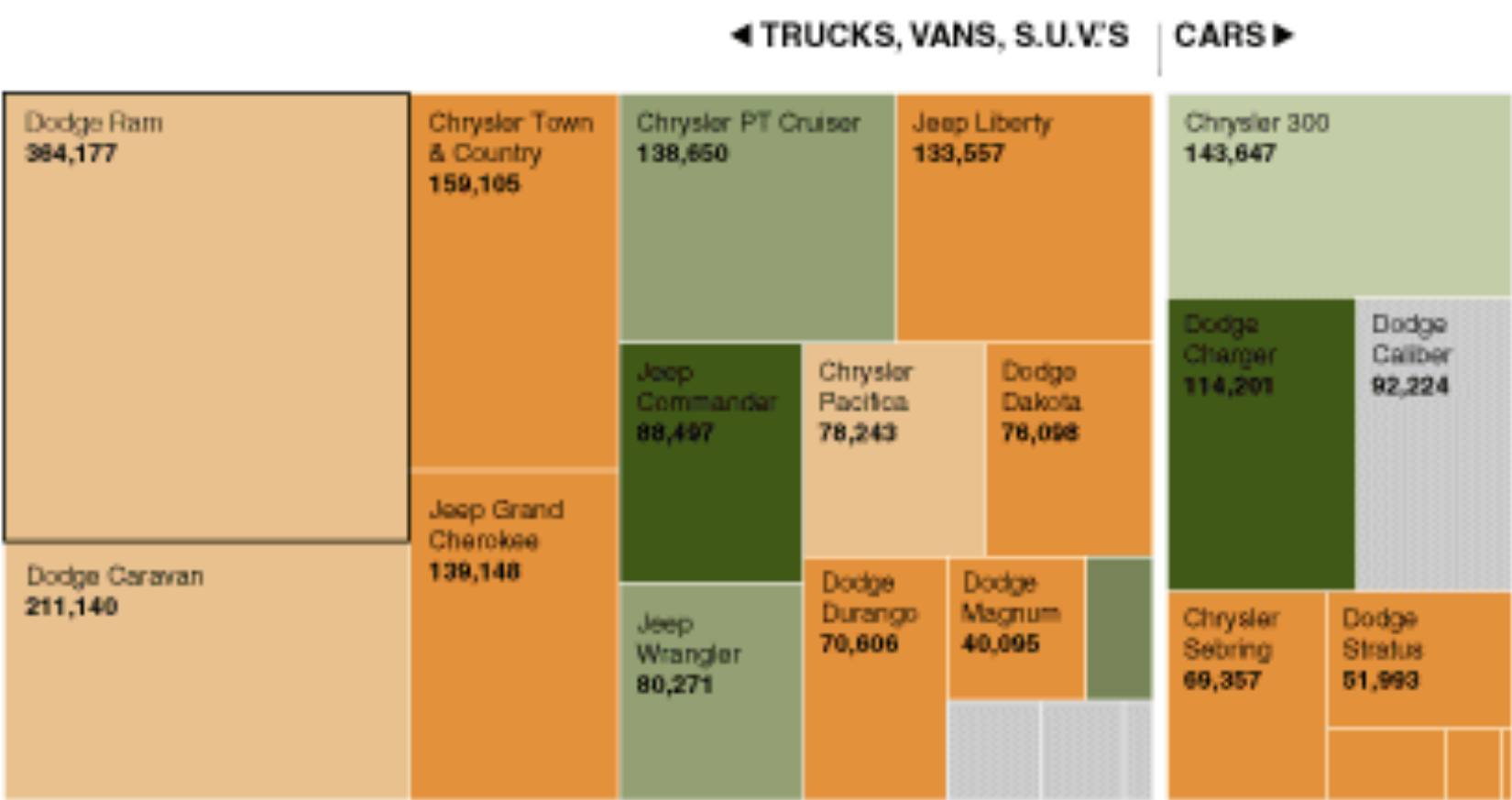
## Truck Sales Slip, Tripping Up Chrysler

Over the past few years, Chrysler executives said they were following the lead of Toyota and Honda, focusing on vehicles that met the needs of their customers. But as American consumers turned away from large trucks and S.U.V.'s in 2006, Chrysler continued to churn out big vehicles, which are now sitting unsold at dealerships across the country.

**Chrysler Group** **-7.0%**  
Trucks/vans/S.U.V.'s 1.6 million  
Cars 0.5 million

Pickups, minivans and S.U.V.'s made up 76 percent of Chrysler's sales, which left it vulnerable when consumers shifted to cars.

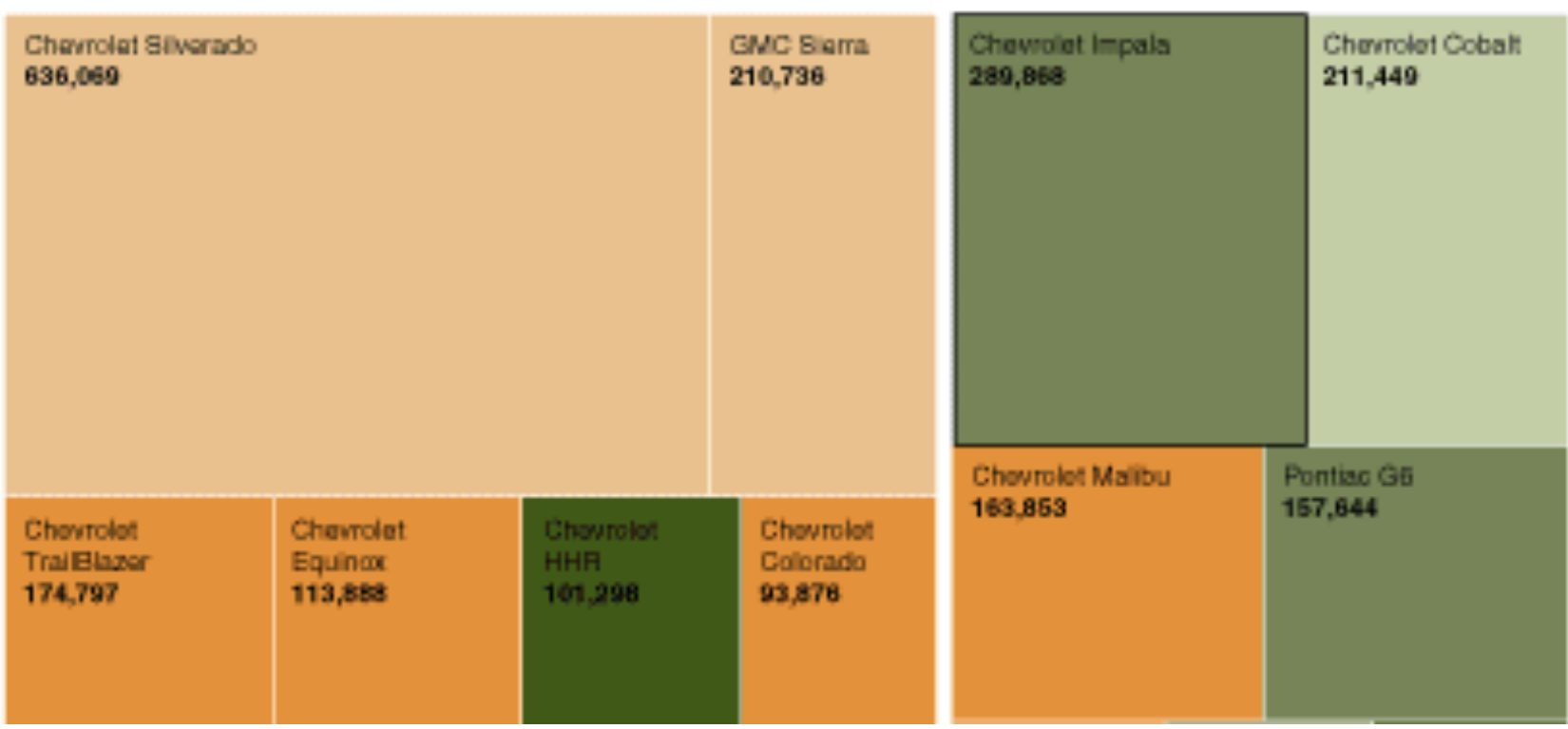
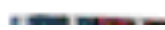
Dodge Ram



**General Motors** **-8.7%**  
Trucks/vans/S.U.V.'s 2.5 million  
Cars 1.6 million

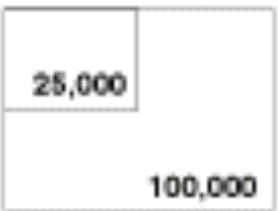
G.M. introduced new versions of its large S.U.V.'s in late 2005, hoping they would bolster sales. Instead, sales of big vehicles were hurt when gas prices climbed. One of the few standouts was the Chevrolet HHR, new in 2005.

Chevrolet

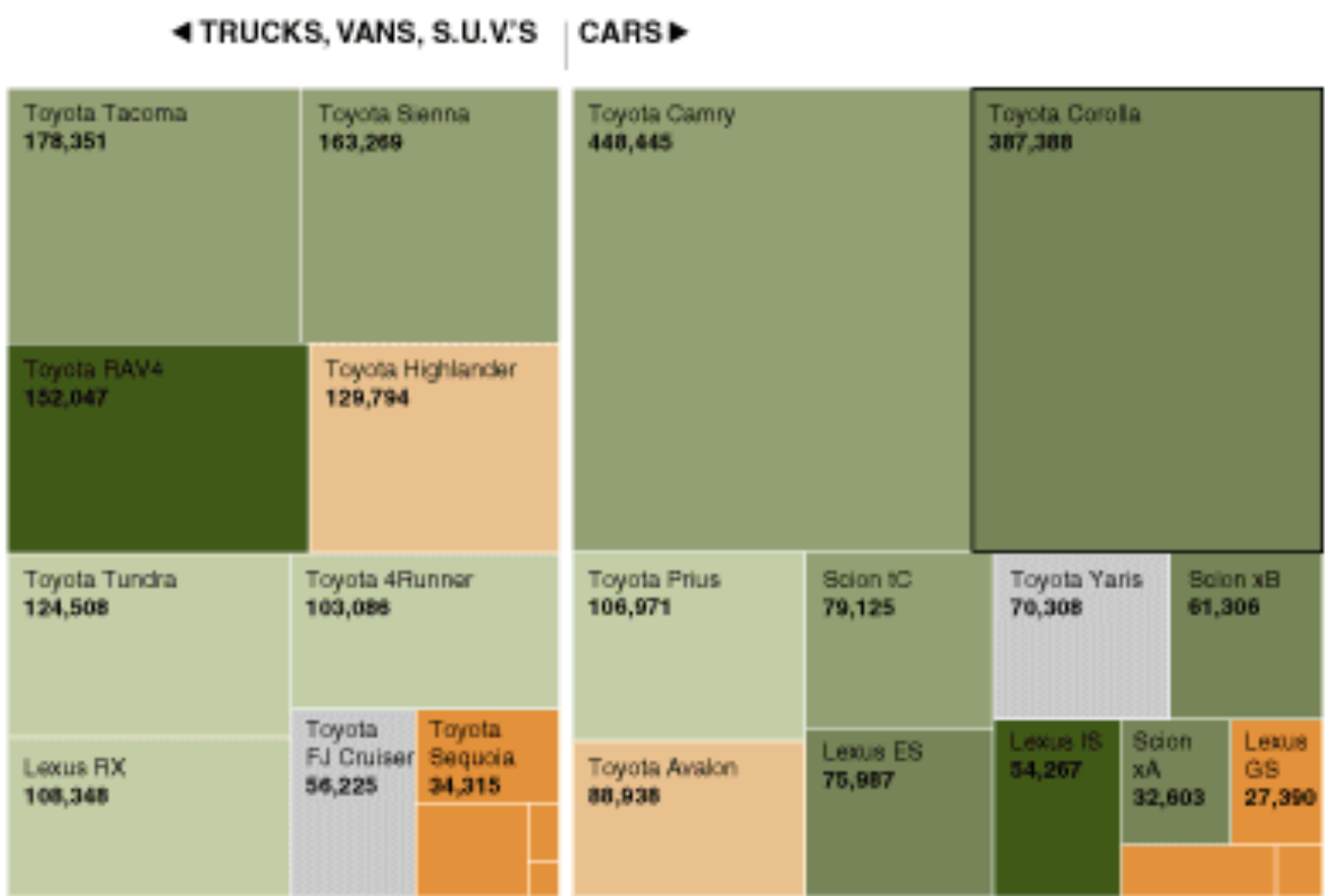


### READING THE CHART

Boxes are scaled proportionally according to number of cars sold in 2006



### Change in sales from 2005 to 2006



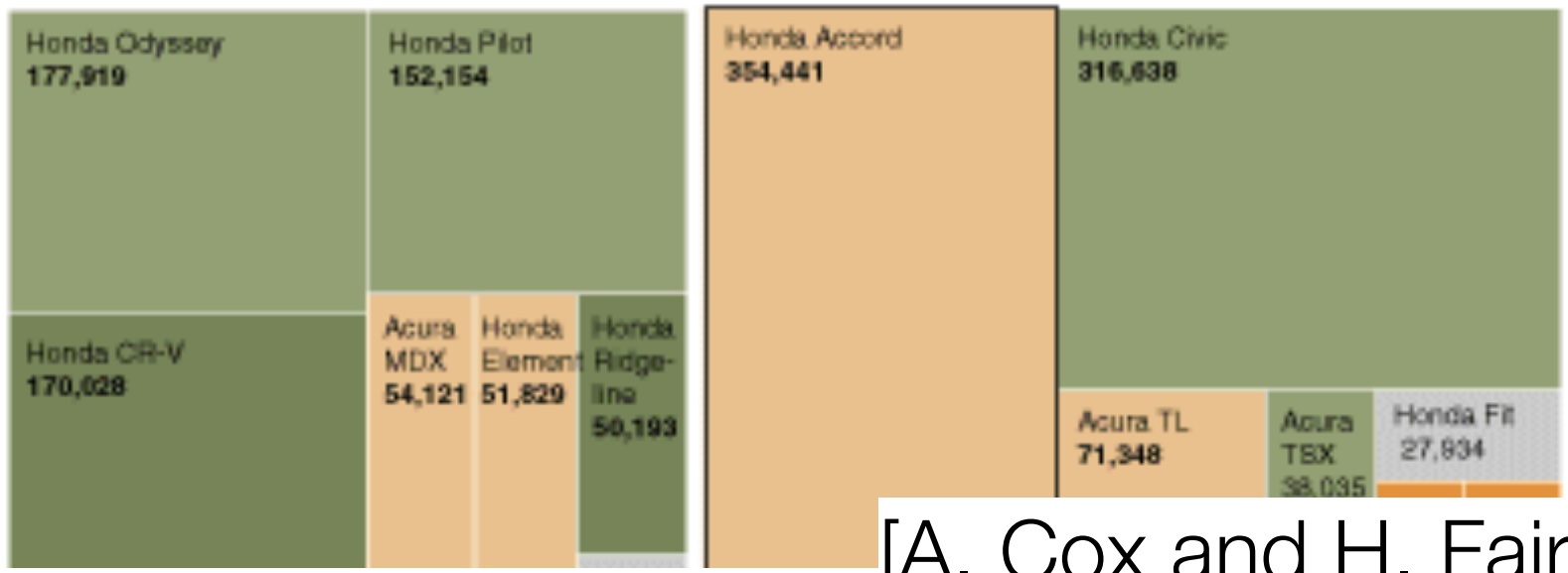
**Toyota** **+12.5%**  
Trucks/vans/S.U.V.'s 1.1 million  
Cars 1.5 million

Toyota rolled out a new version of the Camry, and once again it was the country's best-selling car.

Toyota Corolla



Corolla sales also jumped, along with gas prices. Toyota could not escape the decline in sales of supersized S.U.V.'s like its Sequoia.



**Honda** **+3.2%**  
Trucks/vans/S.U.V.'s 0.7 million  
Cars 0.8 million

Like the Corolla, the small Honda Civic did well. But the Accord stalled. Buyers, it seems, are waiting for the new version to be released this year.

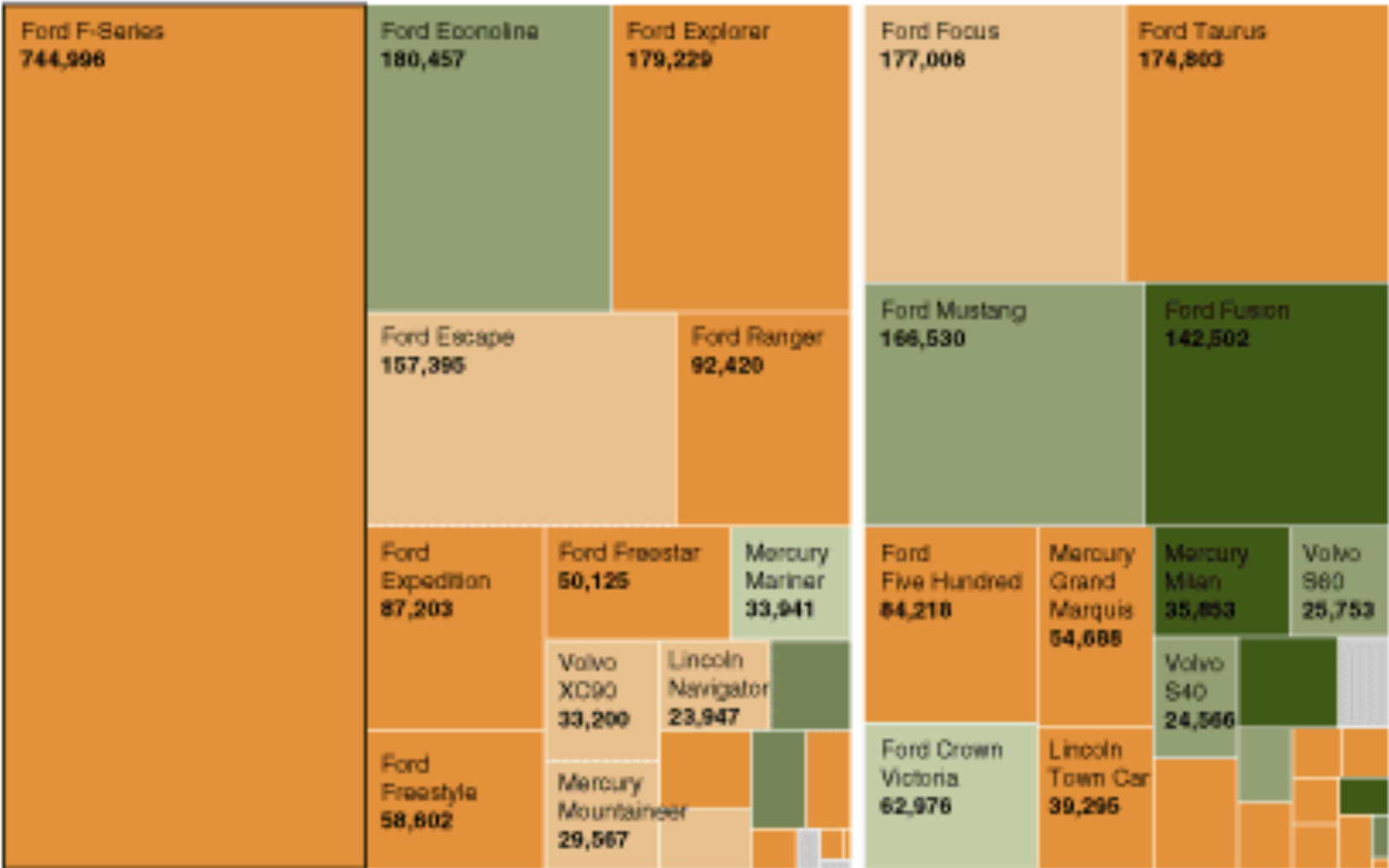
[A. Cox and H. Fairfield, NYTimes, 2012]



# Car/Truck Treemap

**Ford** **-8.3%**  
Trucks/vans/S.U.V.'s 1.8 million  
Cars 1.1 million

Even the country's best-selling vehicles, the F-Series, slumped in 2006, with sales dropping 13 percent. One of Ford's bright spots was the new Fusion sedan, which made its debut in late 2005 and sold well in its first full year.



**BMW** **+2.1%**  
Trucks/vans/S.U.V.'s 0.1 million  
Cars 0.3 million

**Mercedes-Benz** **+10.3%**  
Trucks/vans/S.U.V.'s 0.1 million  
Cars 0.2 million

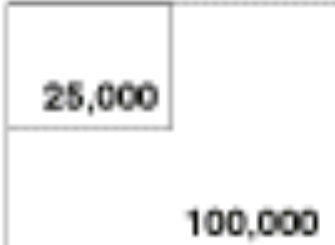
Mercedes-Benz, owned by DaimlerChrysler, had a comeback in 2006, thanks to a new version of its flagship S-Class. BMW sales were helped by a new version of its 3 Series sport sedan.

Sources: Ward's AutoInfoBank; Edmunds

Amanda Cox and Hannah Fairfield/  
The New York Times

### READING THE CHART

Boxes are scaled proportionally according to number of cars sold in 2006



### Change in sales from 2005 to 2006



[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?

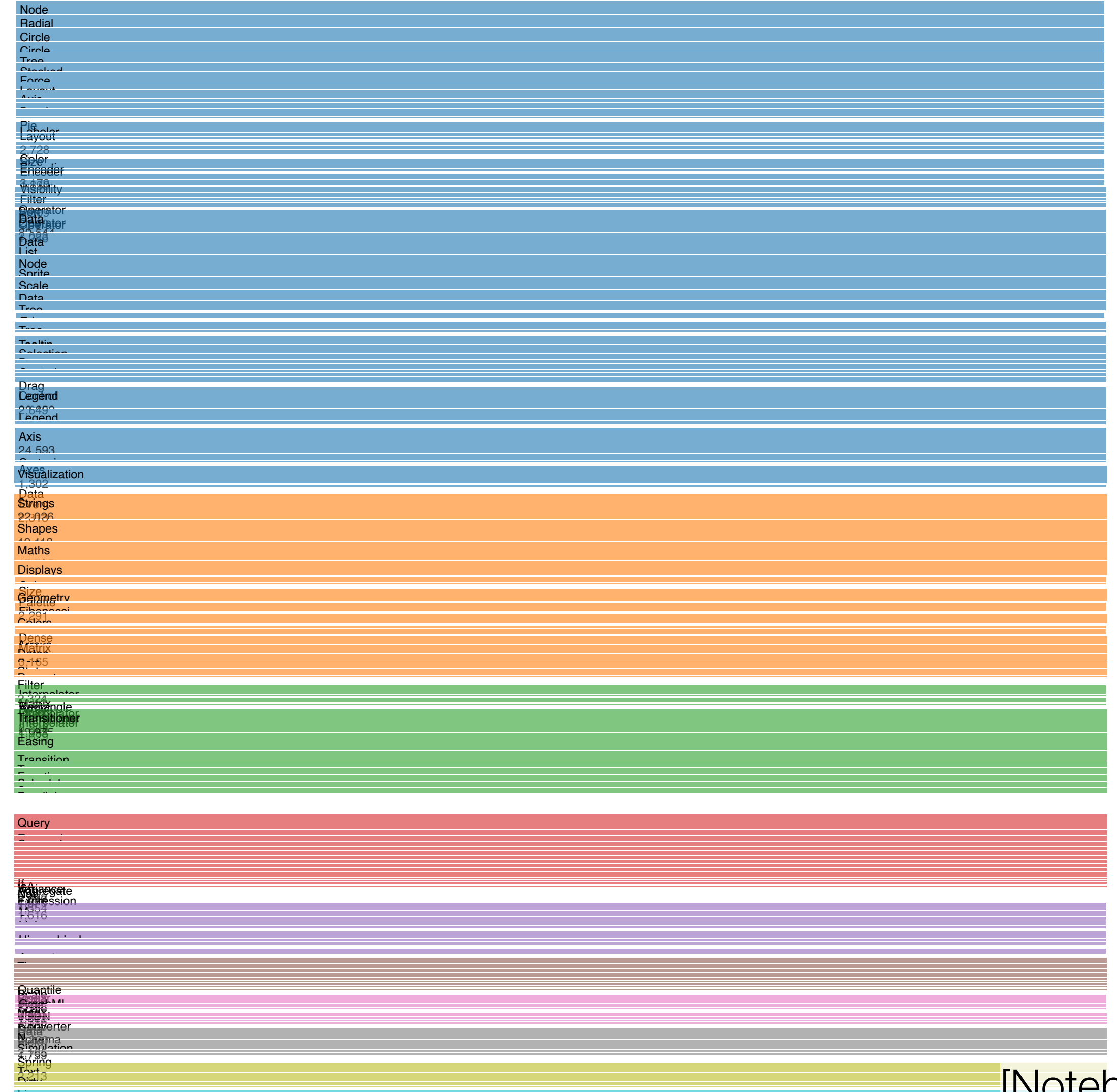
# Layout Algorithms

---

- 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

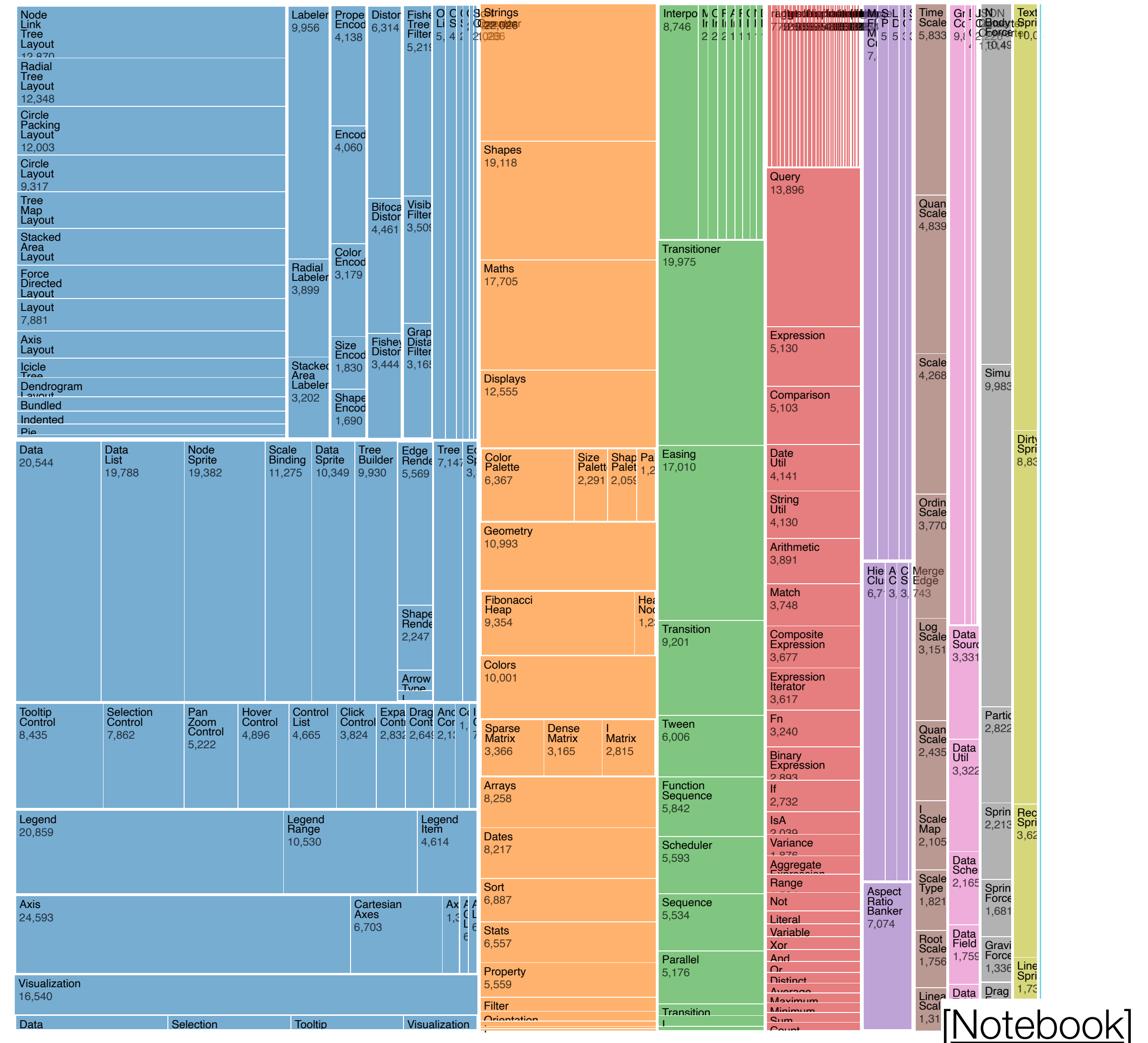


[Notebook]



# Treemap Layouts: Slice & Dice

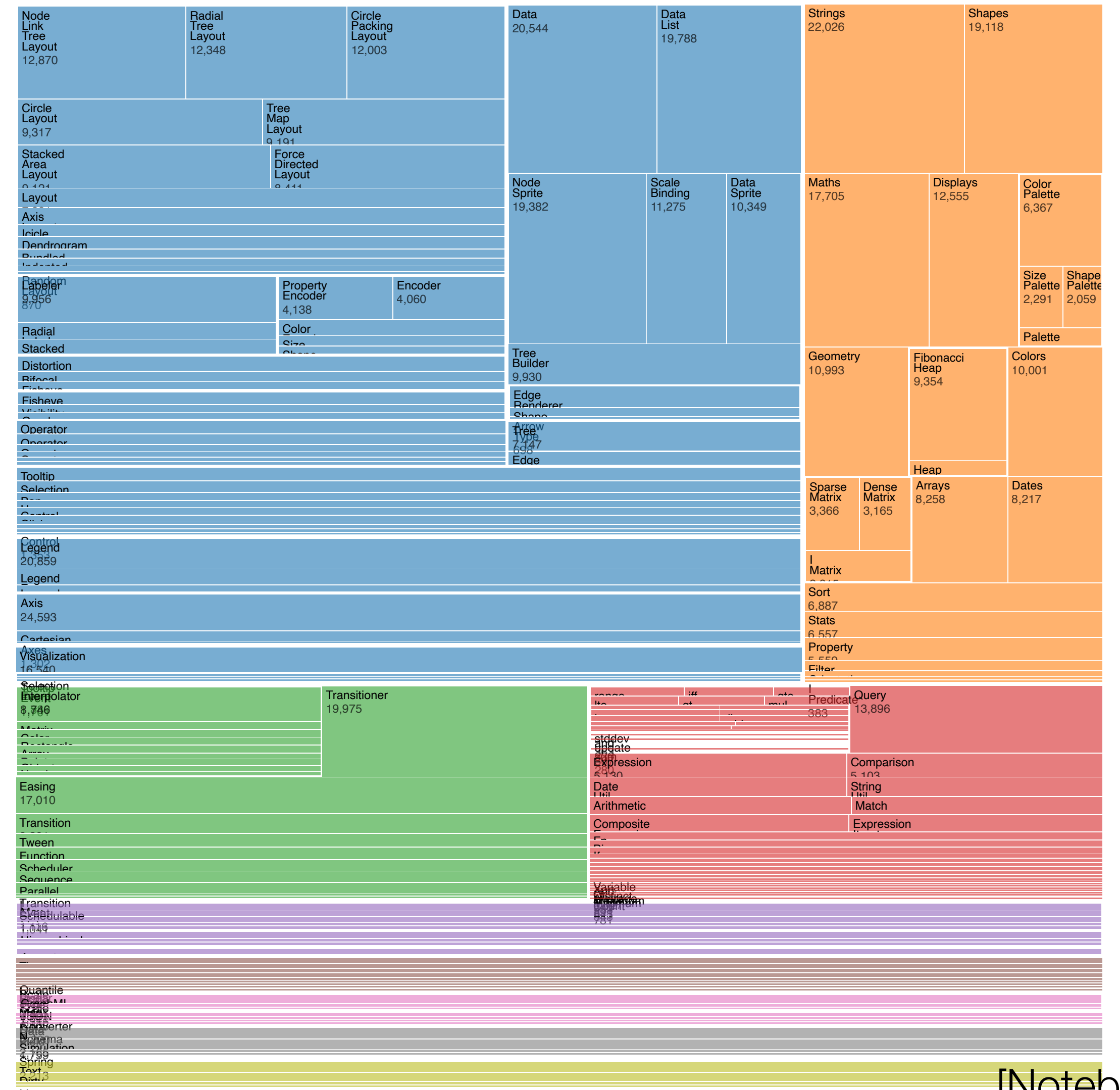
- Split at each level into strips
- At each step, orientation of division (horizontal/vertical) changes
- Better, but some rectangles still have bad aspect ratio





# Treemap Layouts: Strip

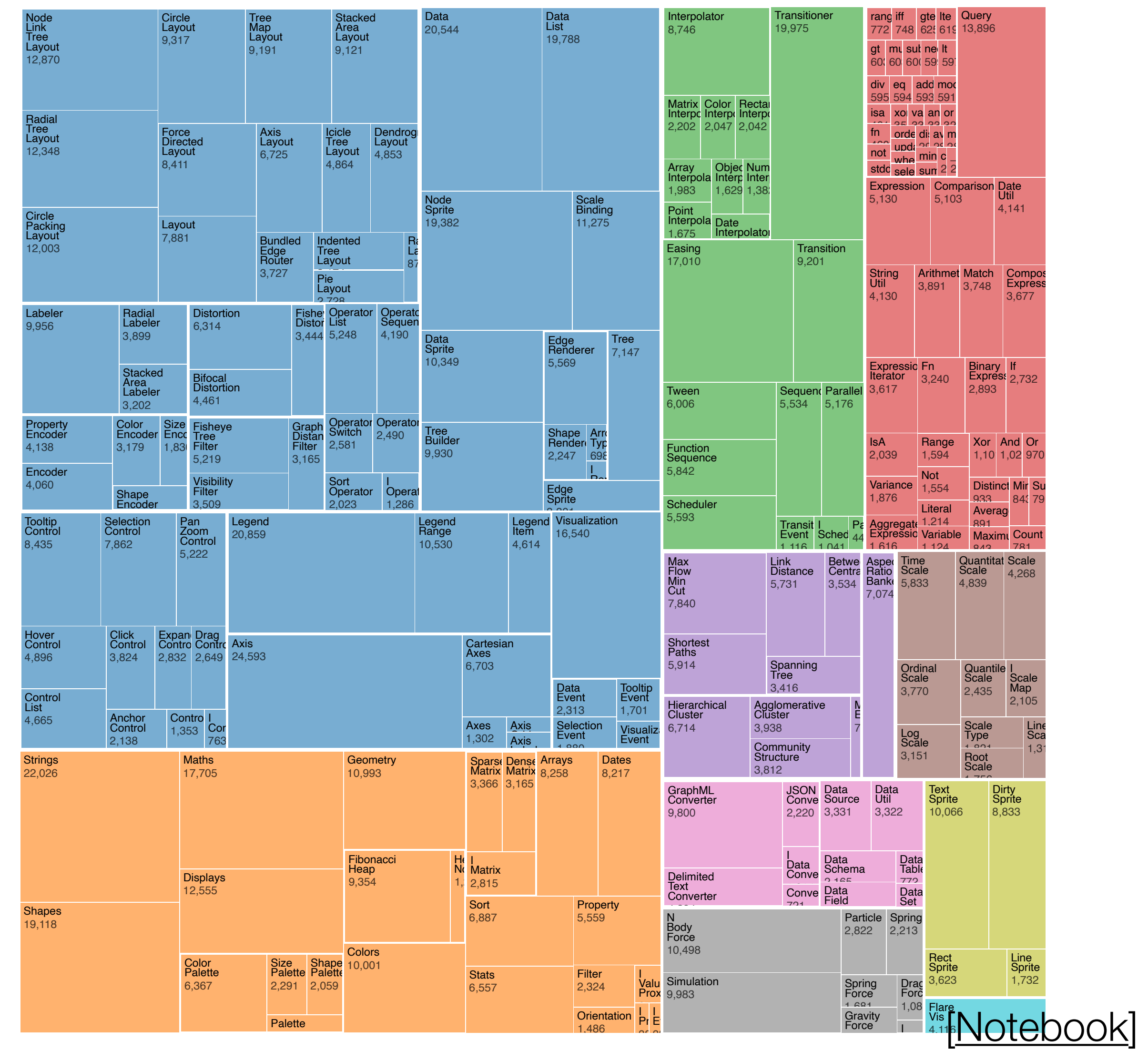
- Consider aspect ratio when adding rectangles
- Do one row at a time by processing rectangles in sorted order by size
  - Check if adding the next rectangle to the row improves aspect ratio
  - When it doesn't, go to next row
- Problem: Last rectangles have bad aspect ratios
- Solution: Look ahead to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



[Notebook]

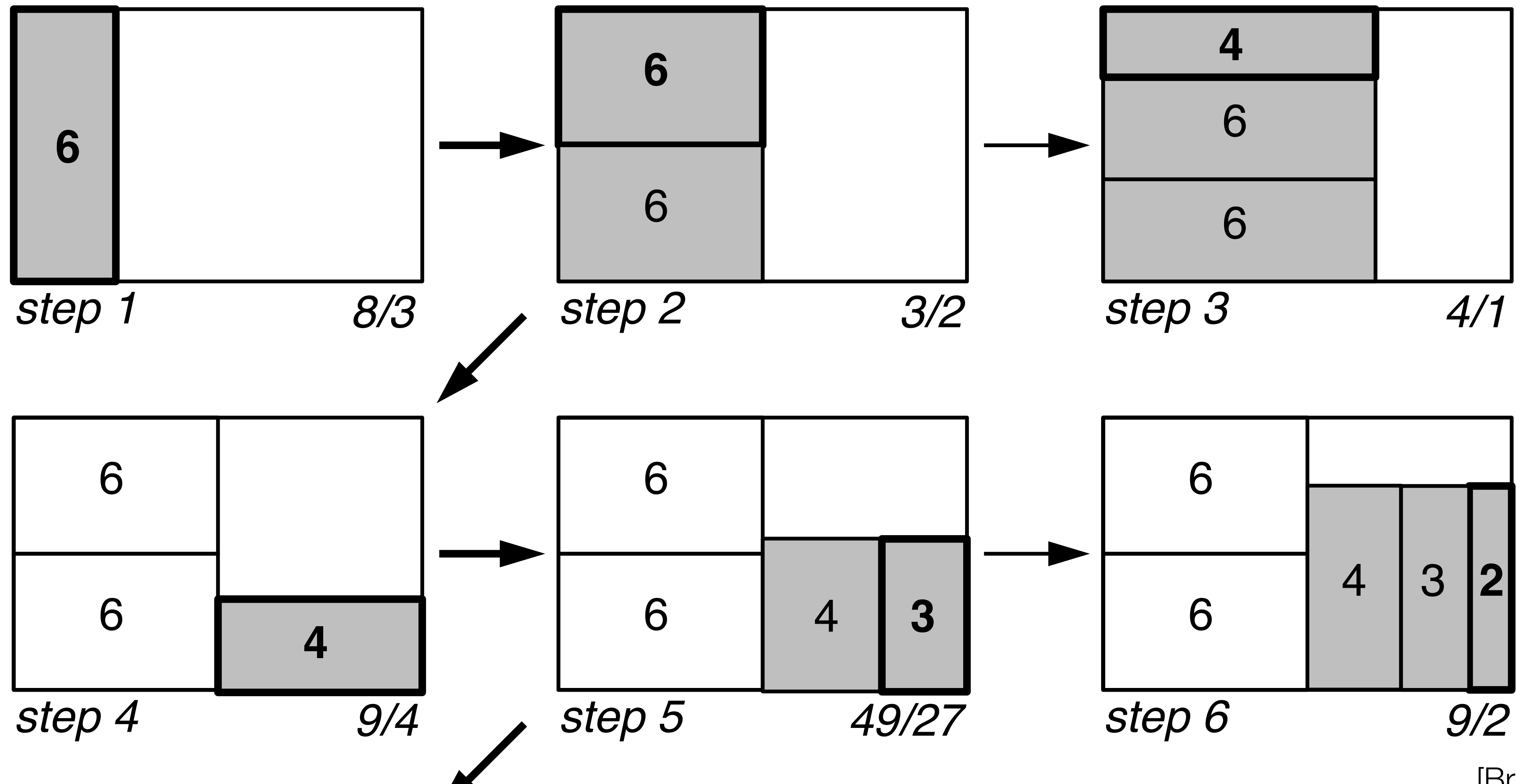
# Treemap Layouts: Squarify

- 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



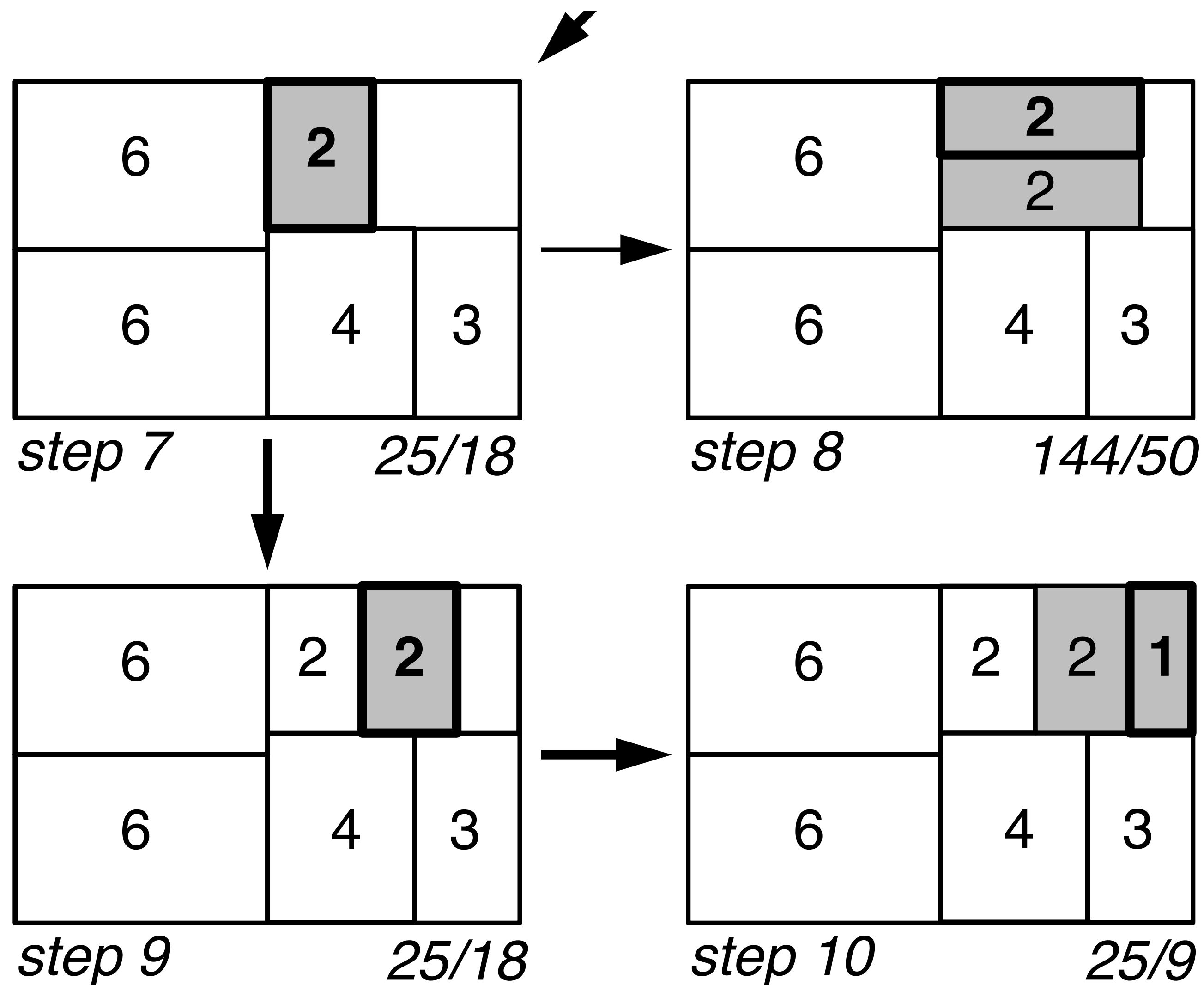
[Notebook]

# Squarification Algorithm



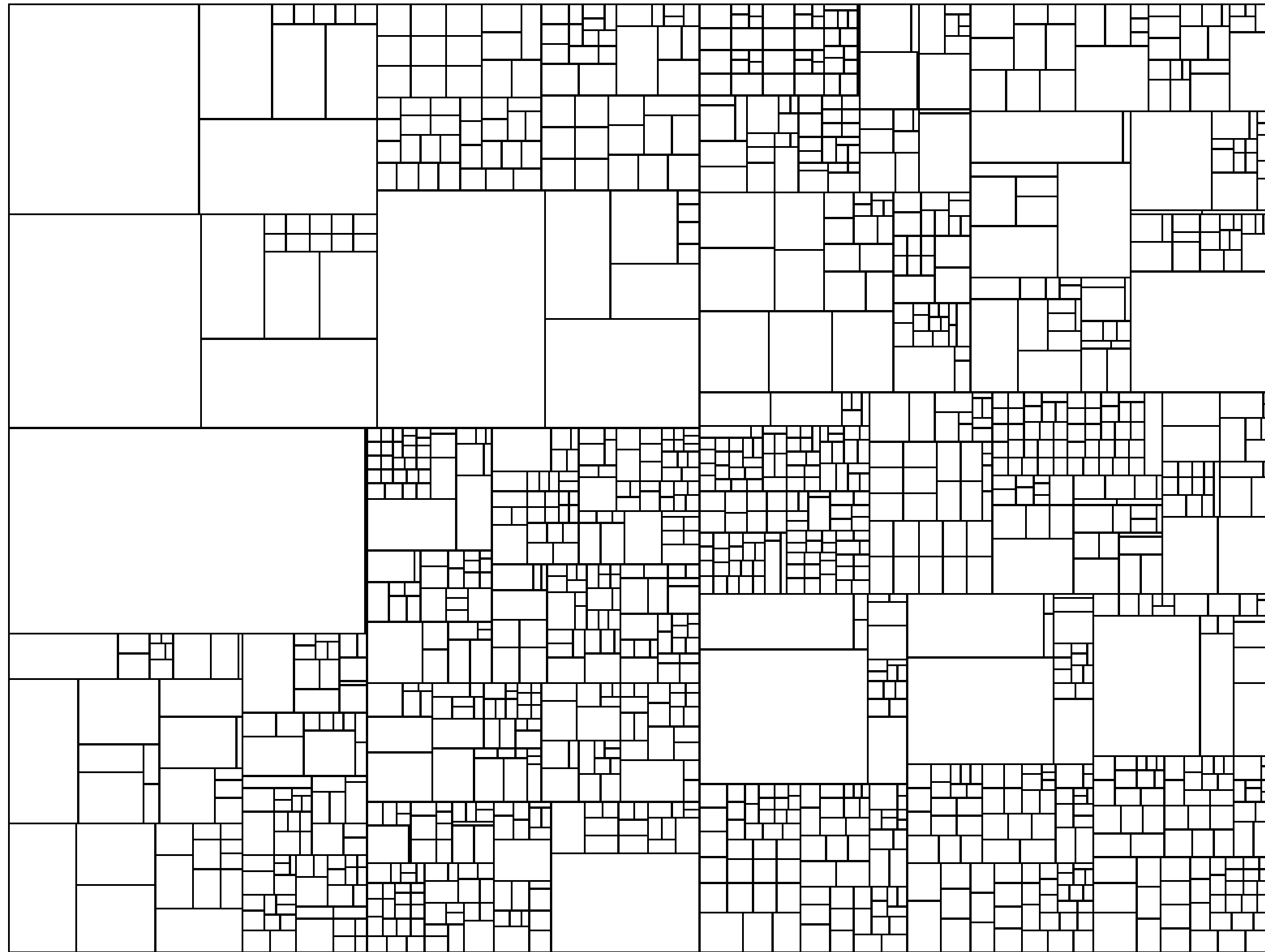
[Brus et al., 1999]

# Squarification Algorithm

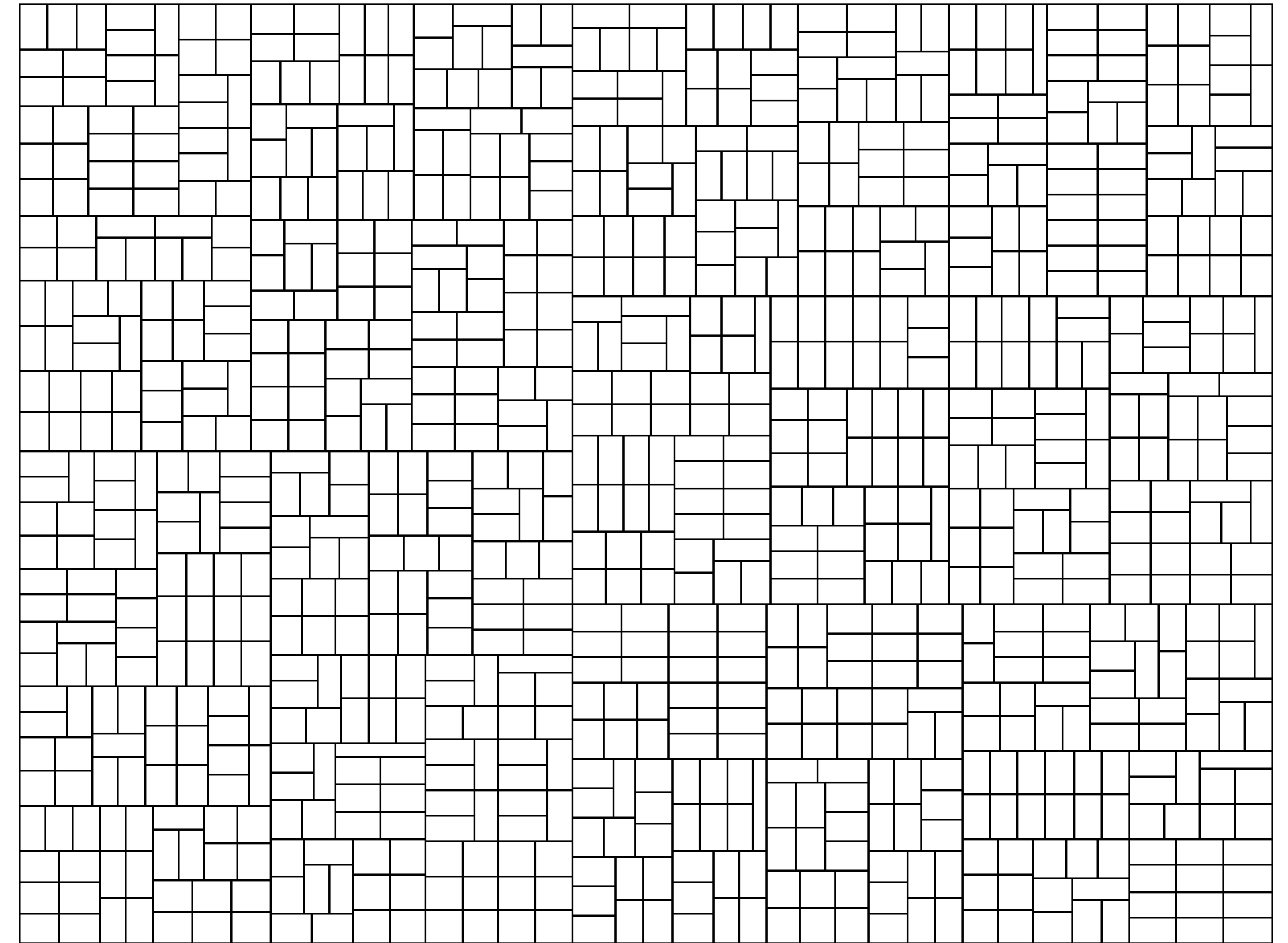


[Brus et al., 1999]

# Squarified Treemaps



(a) File system



(b) Organization

[Brus et al., 1999]

# Squarified Layout

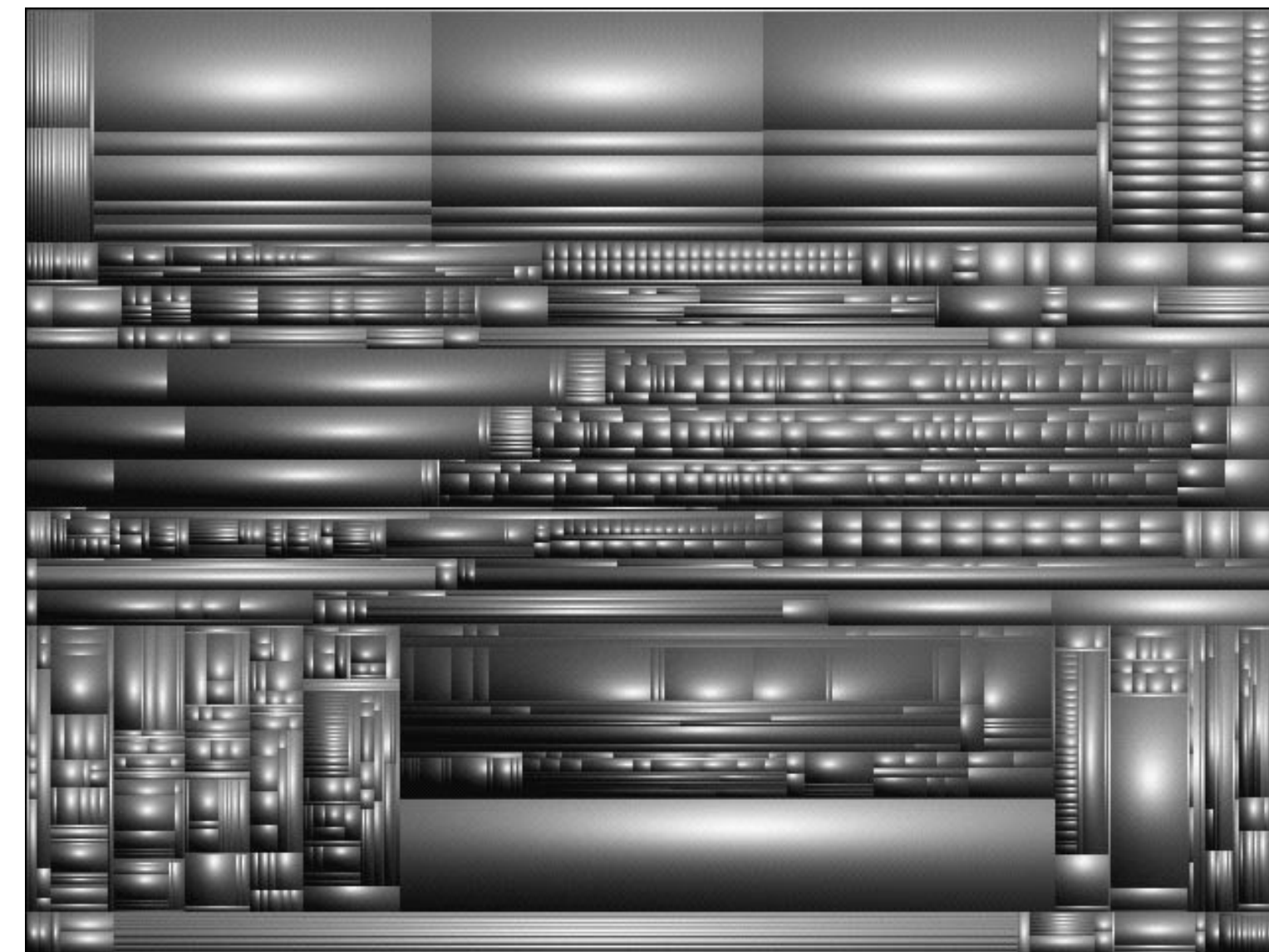
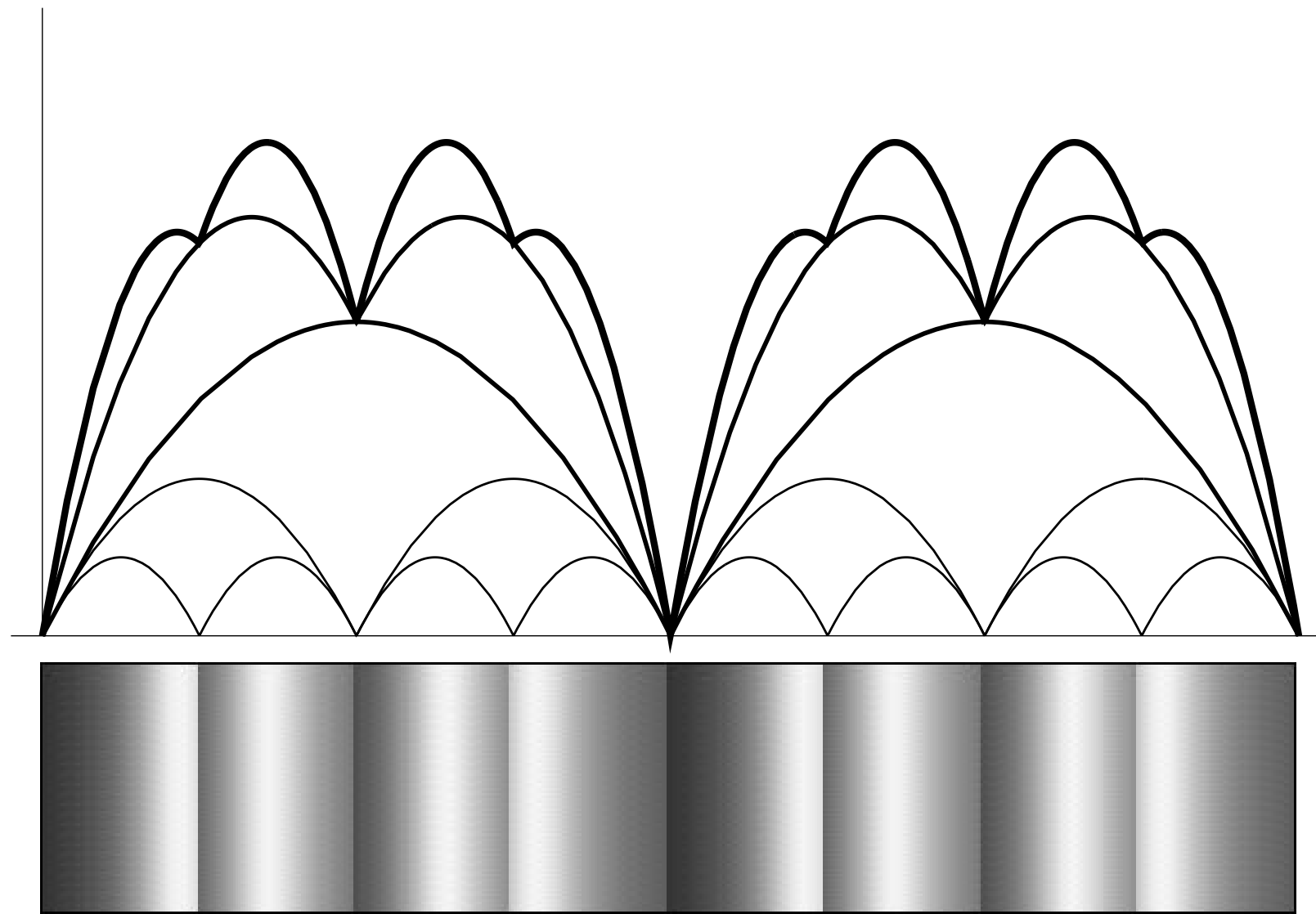
---

- 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

+

-

Zoom In

Zoom Out

Move To Trash

Show Package Contents

Name	Size
Contents	31,3 MB
Resources	21,5 MB
NetServices	5,7 MB
MacOS	2,2 MB
iPhoto	1,9 MB
iPhotoDPA...	273 kB
photocd	70 kB
PlugIns	1,7 MB
.DS_Store	6 kB
PlugIns Disabl	6 kB
Info.plist	1 kB
version.plist	463 Byte
PkgInfo	8 Bytes
Resources Dis	0 Bytes
Icon	65 kB
.DS_Store	6 kB

iPhoto (iPhoto/Contents/MacOS)

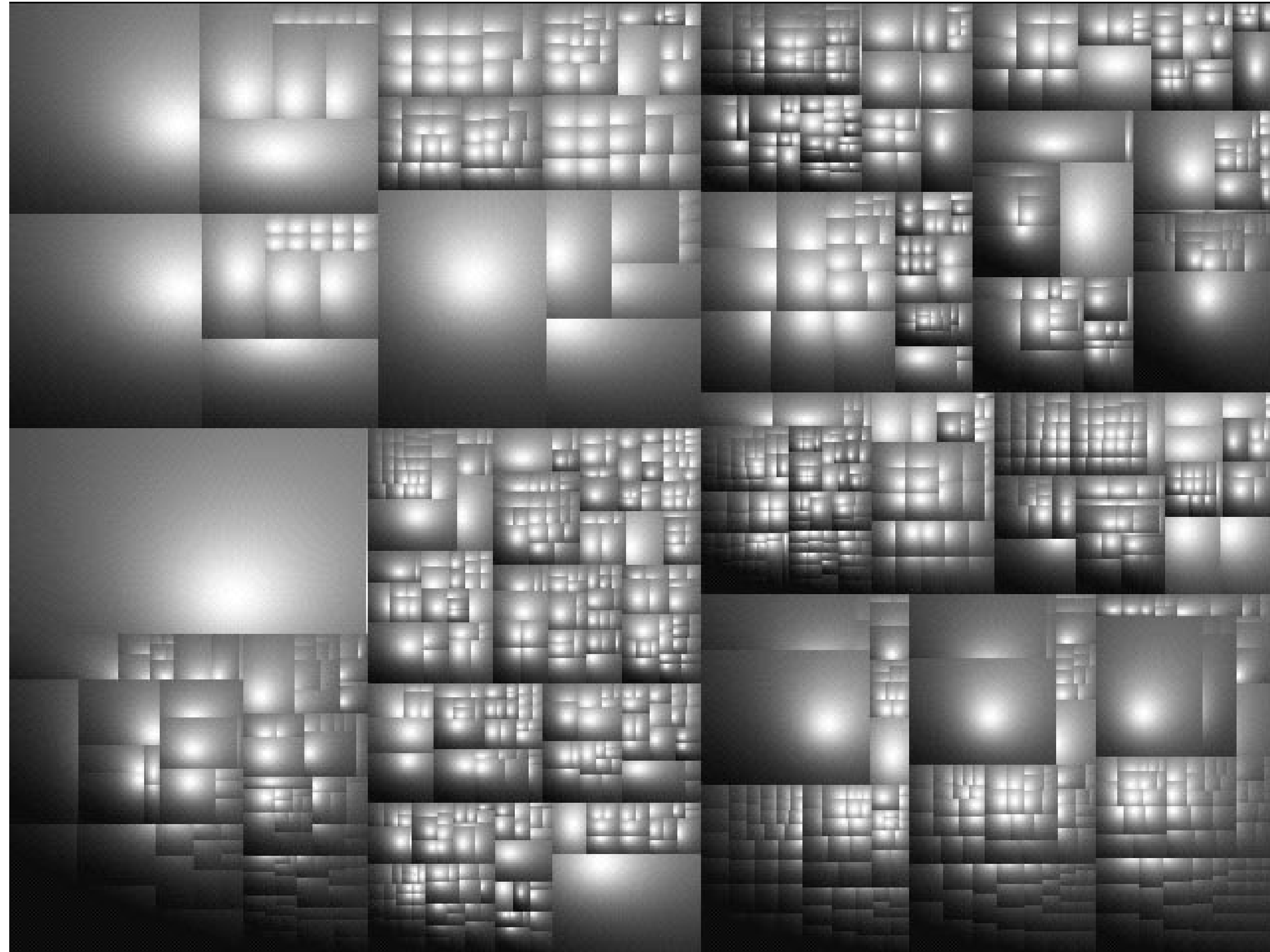
Unix Executable File, 1,9 MB

Color	Kind	Size	Files
Blue	Interface Builder Document	15,4 MB	2104
Red	MP3 Audio File	4,8 MB	2
Green	Unix Executable File	3,8 MB	23
Cyan	JPEG Image	1,6 MB	74
Magenta	Strings File	1,4 MB	348
Yellow	HTML document	1,3 MB	333
Dark Blue	TIFF Document	1,0 MB	310
Brown	Document	886 kB	16
Light Green	Portable Network Graphi	635 kB	21
Teal	XML Property List File	183 kB	332
Purple	Apple Icon Image	109 kB	2
Gold	AIFF Audio	67 kB	2
Grey	Finder Document	65 kB	1
Dark Grey	Script	35 kB	5
Light Grey	Rich Text Format (RTF) d	30 kB	2
Very Light Grey	AppleScript Suite Definit	7 kB	1
Very Light Grey	AppleScript Suite Termin	6 kB	1
Very Light Grey	Graphics Interchange Fo	5 kB	12
Very Light Grey	Cascading Style Sheet (C	4 kB	4
Very Light Grey	Symbolic Link	164 Byte	9

[Disk Inventory X]



# Squarified + Cushioned Treemaps



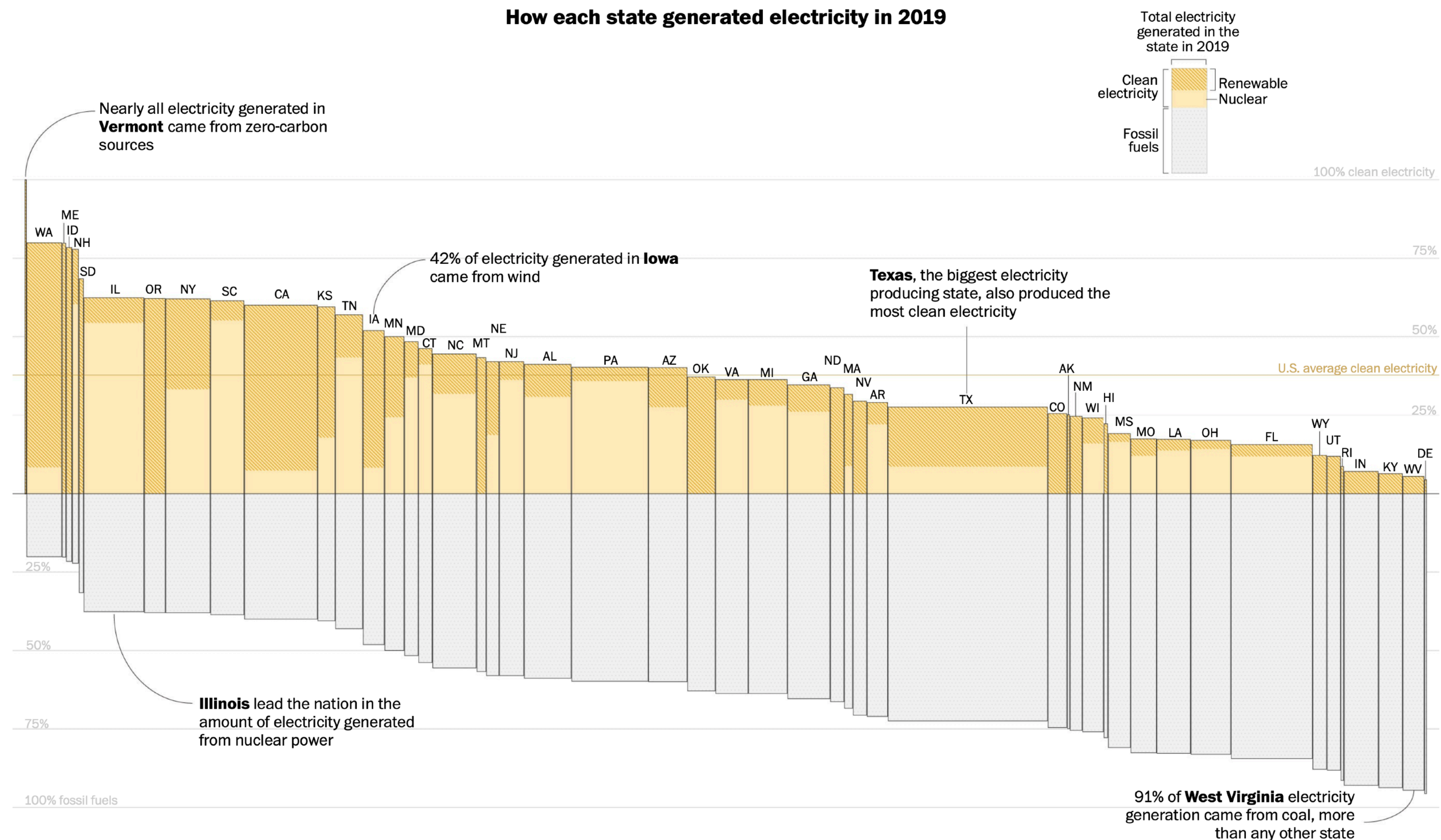
(a) File system



(b) Organization

[Brus et al., 1999]

# Variations: Marimekko Chart

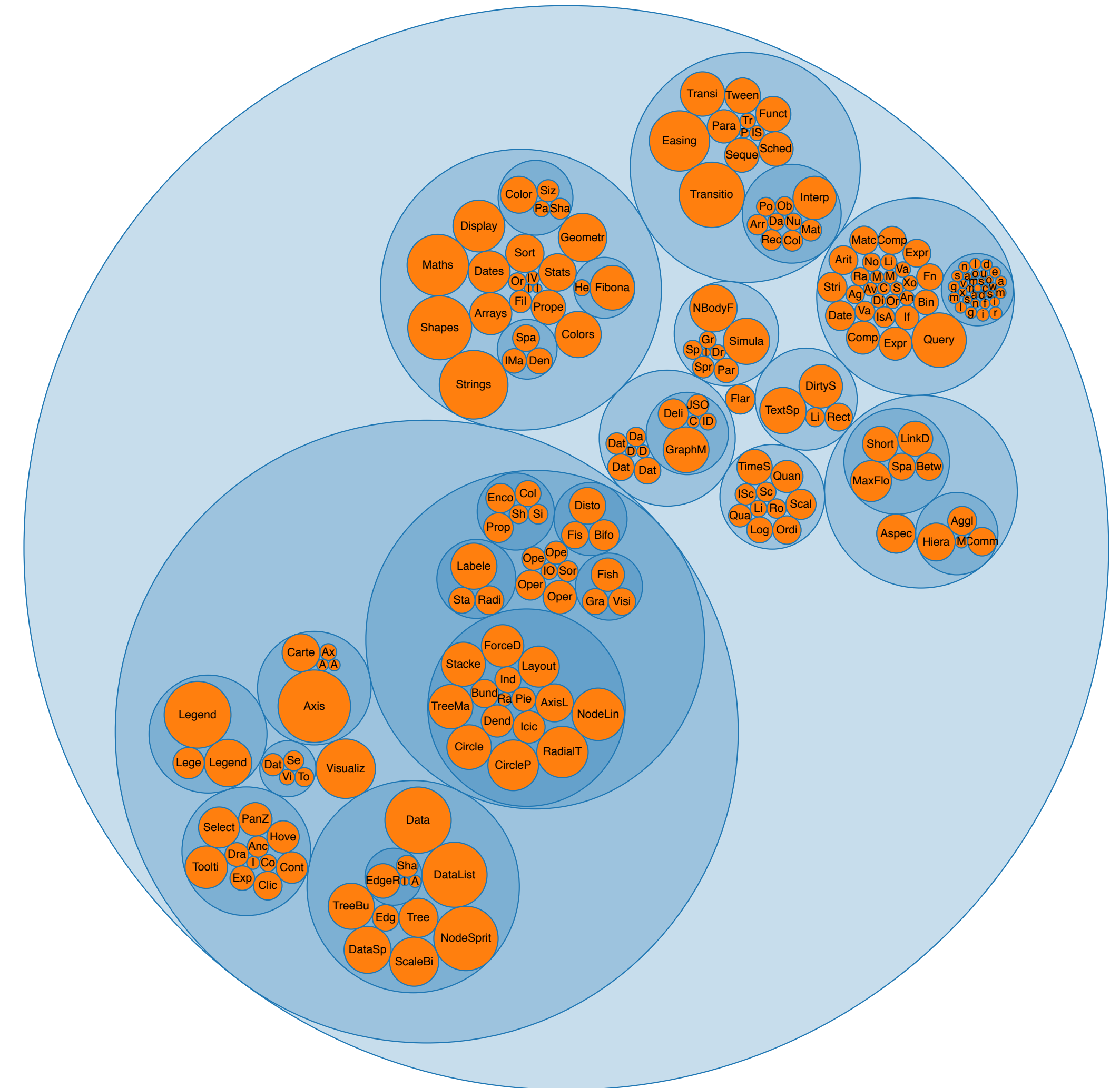


[J. Muyskens, [Washington Post](#)]



# Nested Circles

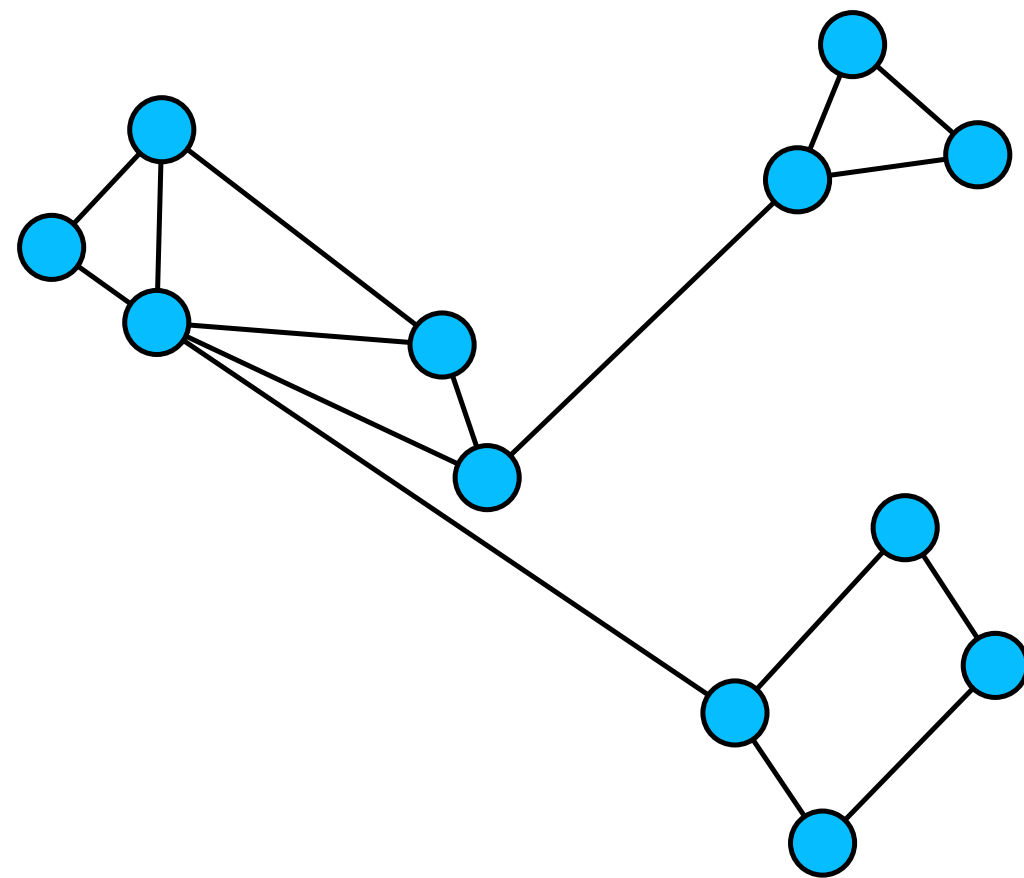
- Looks more like cluster diagram, but shows hierarchy
- Containment shown by the layering of semi-transparent circles
- Labeling becomes more difficult



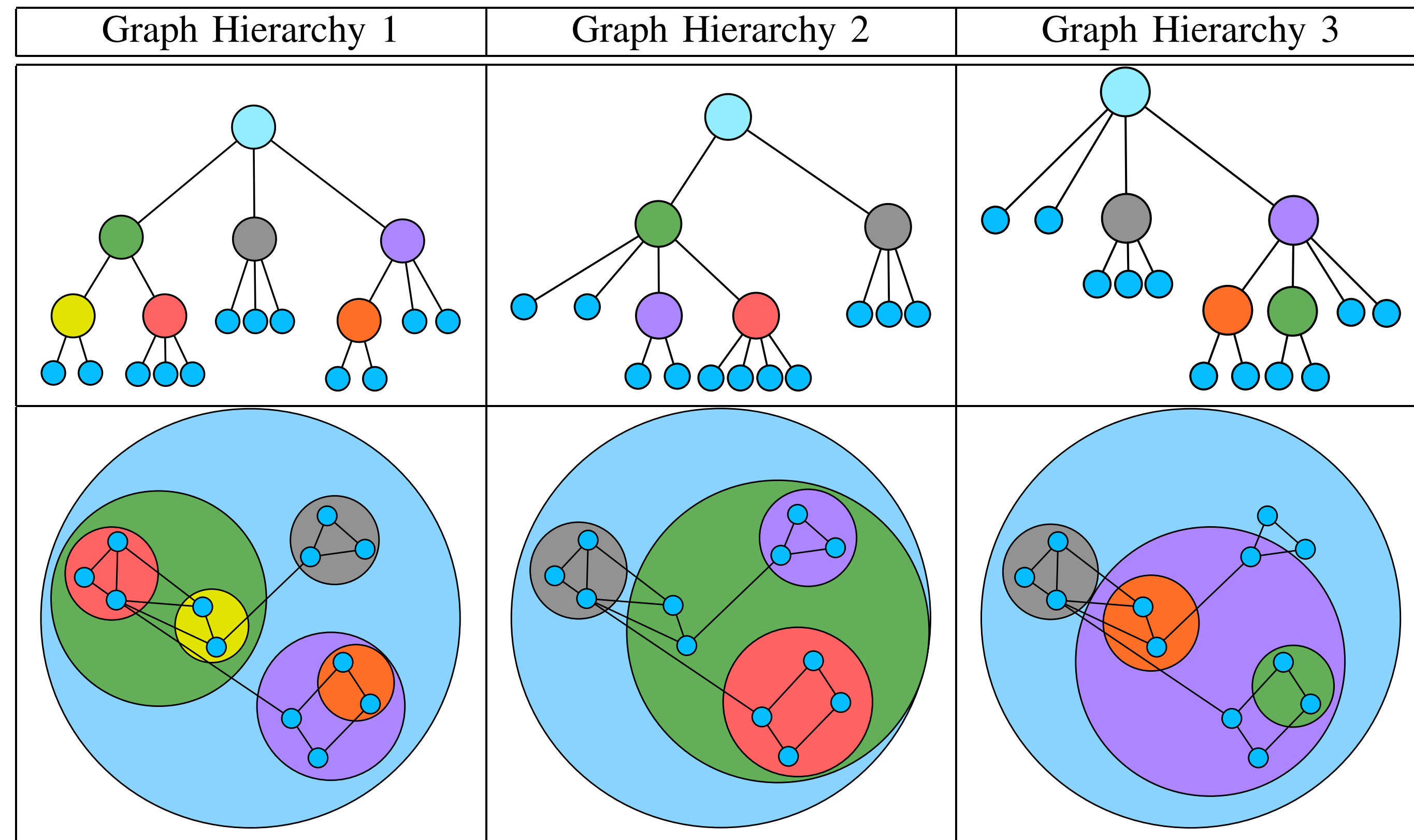
[Bostock, 2012]

# 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]