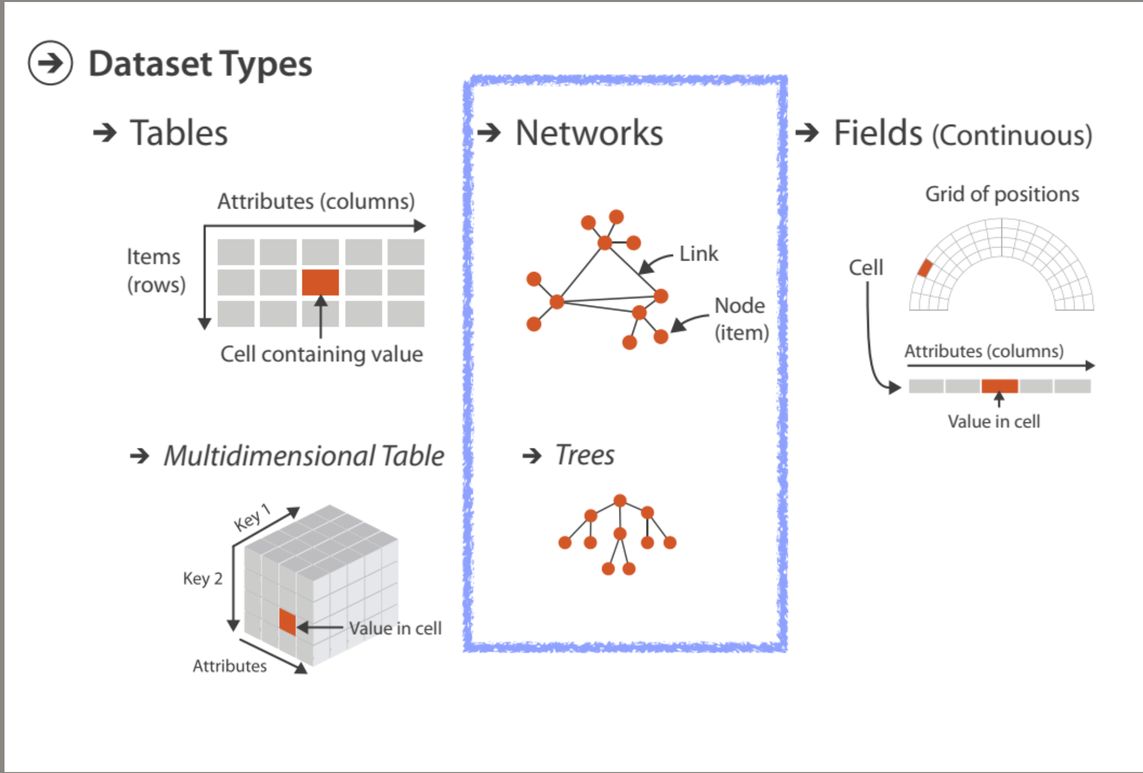
Spatial Layout:
Arrange Networks and Trees

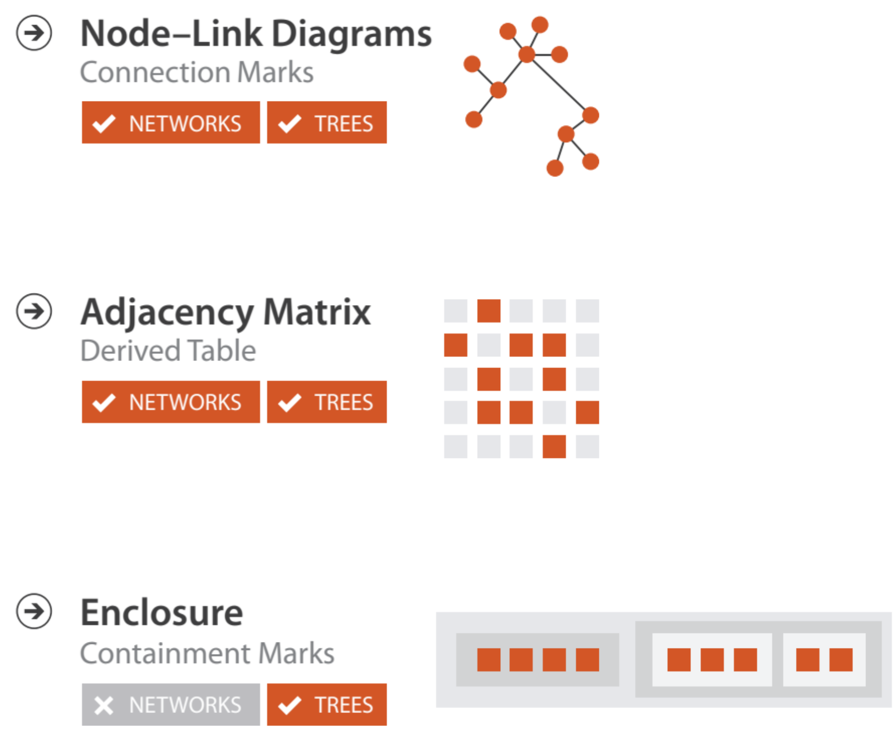
Arrange Network and Trees

- Network: model relationships between things
  - Graph
  - Both links and nodes can have attributes

- Tree
  - Special case of network
  - No cycles



Three Types of Network/Tree Visual Encoding



Node–Link Diagrams
Connection Marks

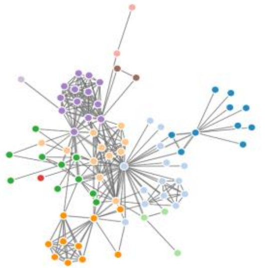
✓ NETWORKS ✓ TREES

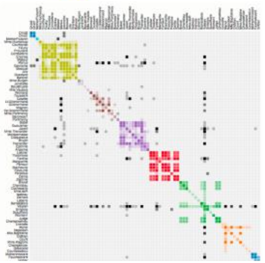
Adjacency Matrix
Derived Table

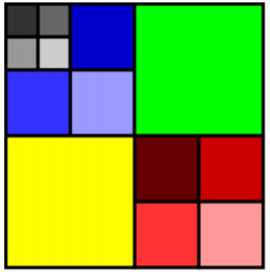
✓ NETWORKS ✓ TREES

Enclosure
Containment Marks

✗ NETWORKS ✓ TREES

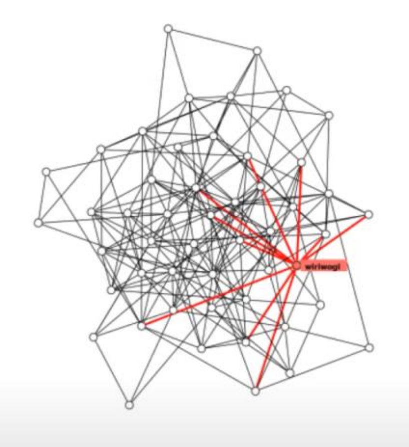






Network

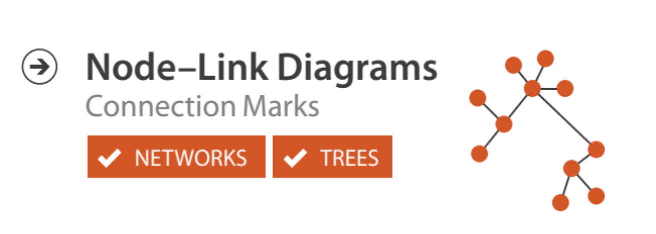
Network Tasks: Topology-based and Attribute-based

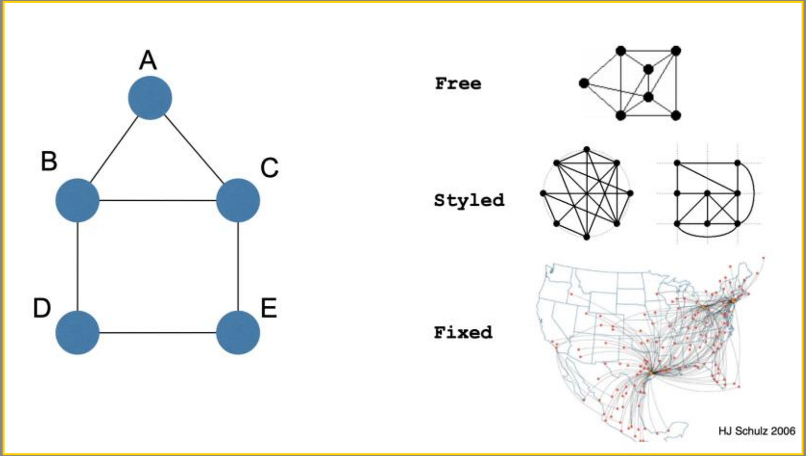


- Topology based task
  - Find path
  - Find topological neighbors
  - Compare important nodes
  - Identify clusters
- Attribute based task (similar to table data)
  - Find distributions, ….
- Combination tasks, incorporating both
  - Example: find friends-of-friends who like cats
  - Topology: find all adjacent nodes of given node
  - Attributes: check if has-pet (node attribute) == cat

Node-Link Diagram

- Nodes: point mark
- Links: line mark
  - Link: straight lines or arcs
  - Connect nodes
- Very very easy to understand
- Many variants



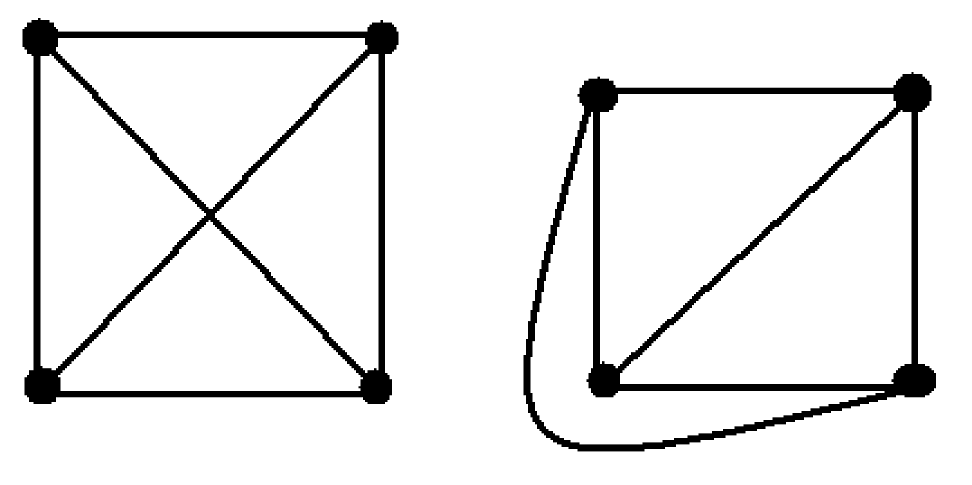


HJ Schulz 2006

S07-01

Which Network Diagram do You Like Most?

- They are the same data, but different diagram layout

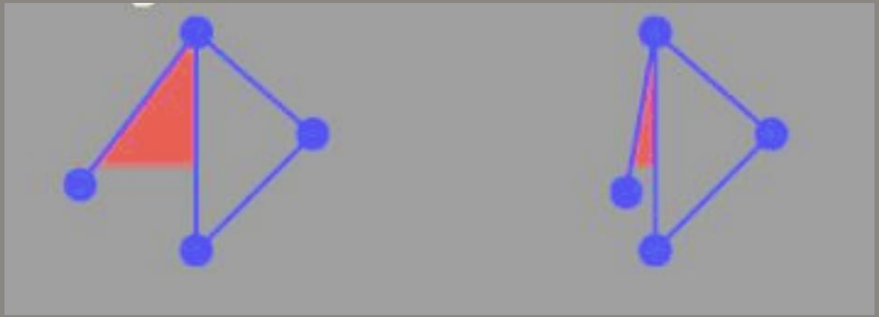


drawback edge crossing

bending edge,
different edge length

Which Network Diagram do You Like Most?

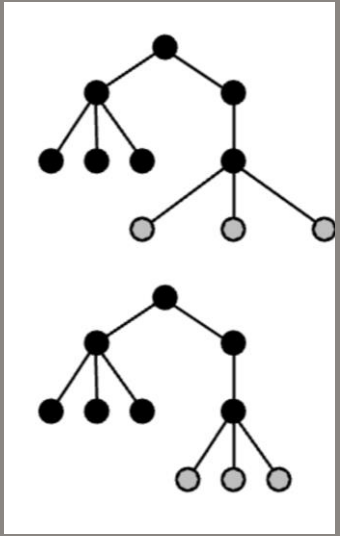
- They are the same data, but different diagram layout



Drawback: different angular distance between edges

Which Network Diagram do You Like Most?

- They are the same data, but different diagram layout

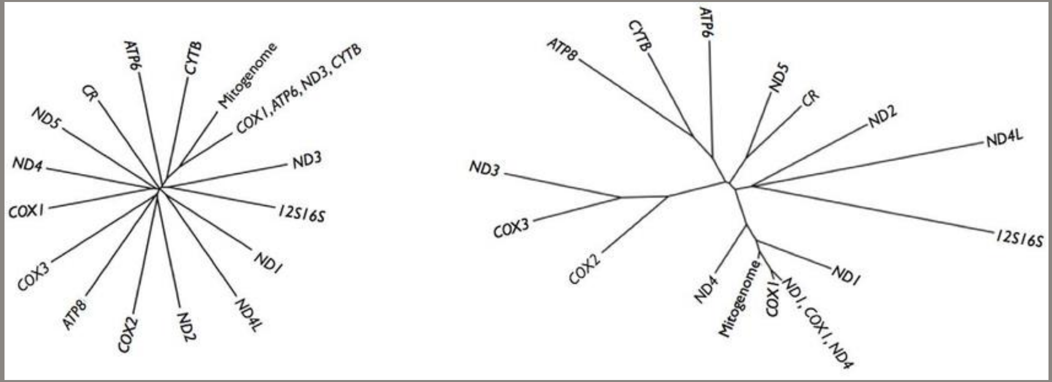


Drawback: similar topology structure looks different

Drawback: worse space utilization

Which Network Diagram do You Like Most?

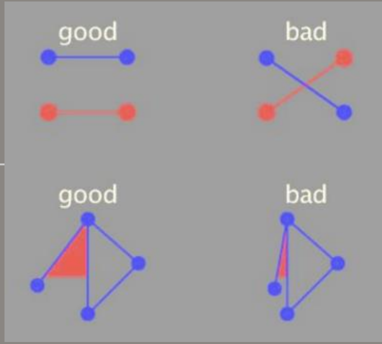
- They are the same data, but different diagram layout



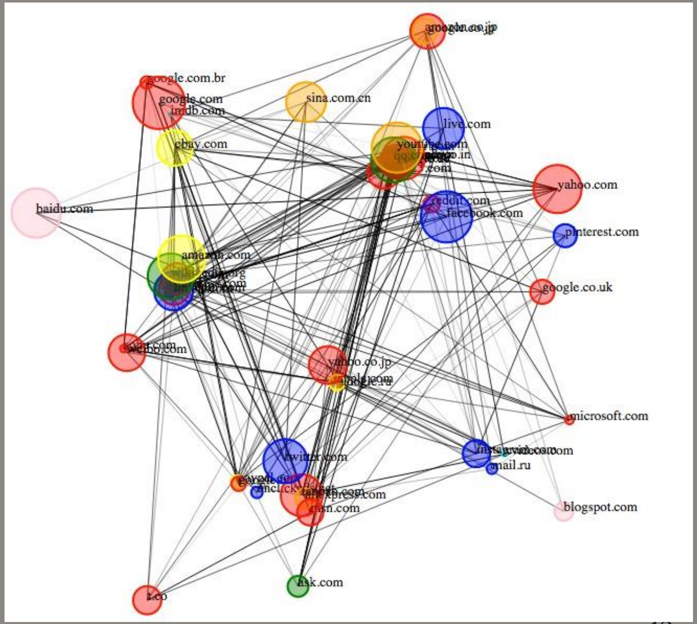
Drawback: do not emphasize the topology distance

Drawback: use more space

Good Node Link Diagram



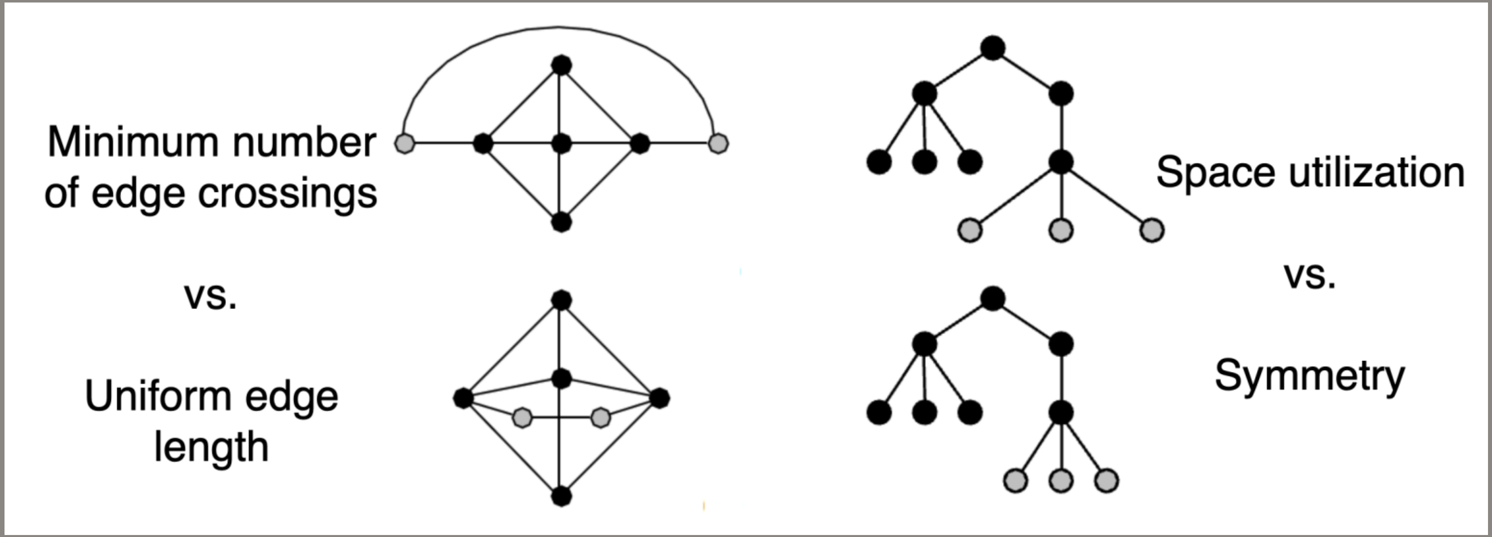
- Minimize
  - Edge crossing
  - Distances between topological neighbor node
  - Total drawing area
  - Edge bends
  - Edge length disparities
- Maximize
  - Angular distance between edges
- Emphasize symmetry
  - Similar graph structures should look similar in layout



“not so good” example

However: Criteria Conflict

- Example



Optimization-based Layout

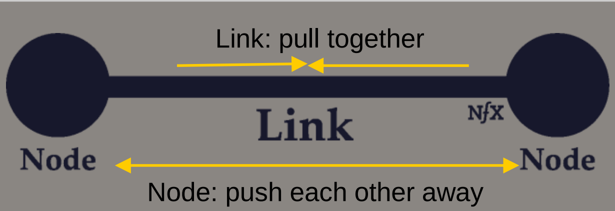
- Optimization problem
- Define a cost function by the above criteria
  - Ex: F (Layout) = a*crossCount + b*[space used] + …

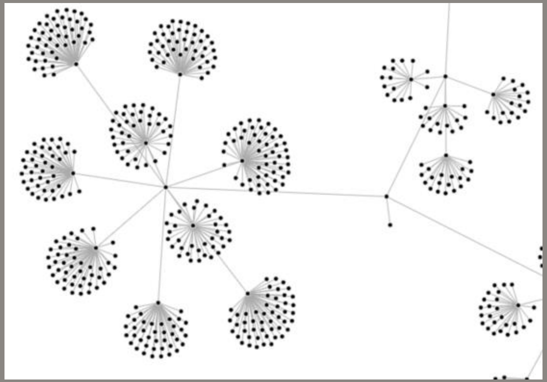
- Use known algorithm to find the layout with minimum cost
  - Energy-based physics model
  - Force-directed placement
    - Popular
      - introduced in D3 tutorial if we have time
  - Spring embedded

Forced-directed Layout

- Model
  - Link: springs pull together
  - Nodes: magnets repulse apart

- Algorithm
  - Place vertices in random positions
  - While not equilibrium
    - Calculate force on vertex
      - Sum of pairwise repulsion of all nodes and attraction between connected nodes
    - Move vertex by c*vertex_force

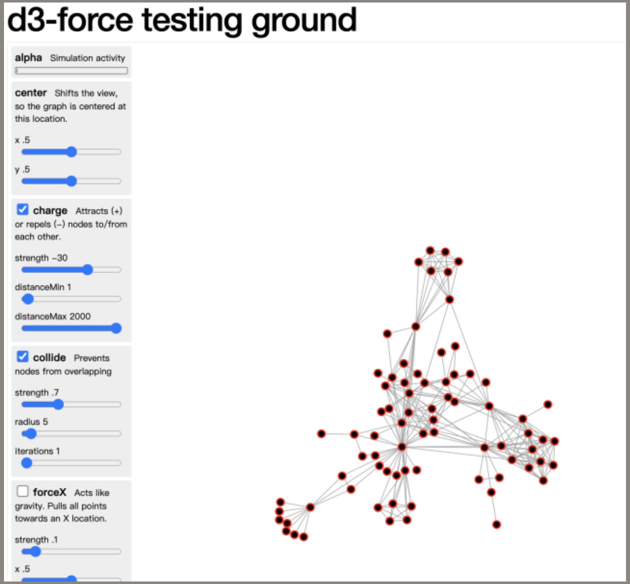




Forced-directed Layout

- Procs
  - Good layout for small, sparse graphs
  - Clusters typically visible
  - Uniform edge length

- Cons
  - Nondeterministic
  - Computational expensive ~O(nodes^3)
  - Cannot scale up well beyond 1k nodes
  - Visualize Iterative progress: distract and not useful information



d3-force testing ground

alpha Simulation activity

center Shifts the view, so the graph is centered at this location.

x .5

y .5

☑ **charge** Attracts (+) or repels (–) nodes to/from each other.

strength –30

distanceMin 1

distanceMax 2000

☑ **collide** Prevents nodes from overlapping

strength .7

radius 5

iterations 1

☐ **forceX** Acts like gravity. Pulls all points towards an X location.

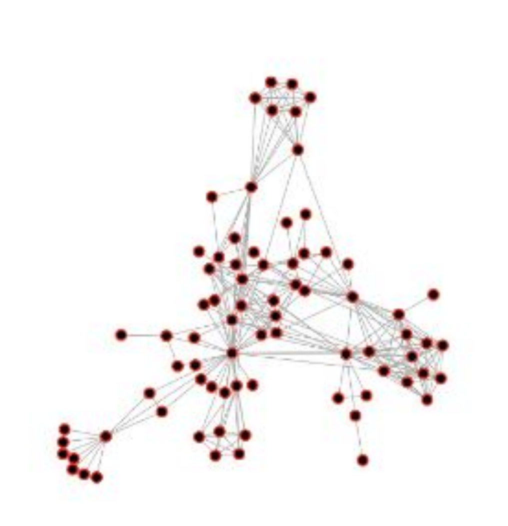
strength .1

x .5

D3: https://bl.ocks.org/steveharoz/8c3e2524079a8c440df60c1ab72b5d03

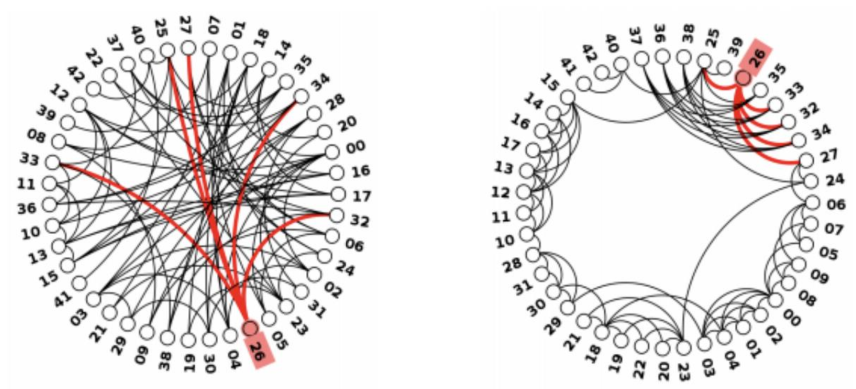
Forced-directed Layout

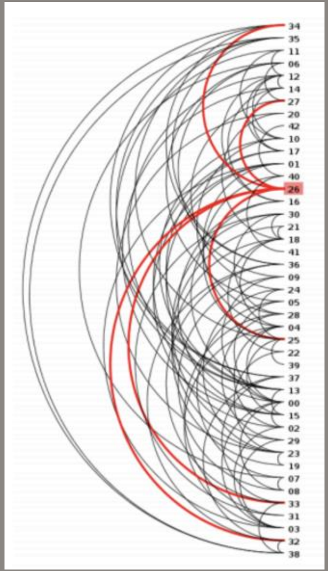
- Visual encoding
  - Line mark for link, point mark for nodes
  - Encode more attributes by visual channels of points and lines
- Considerations
  - Spatial position: no meaning
  - Proximity? “Sometimes” meaningful
  - Long edges more visually salient
- Tasks
  - Explore topology, locate path, cluster
- Scalability
  - Node/edge density: E<4N

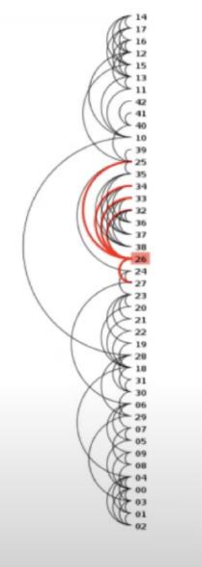


Restricted Layout: Circular/Arc

- Layout nodes around circle or along line
  - Circular layouts
  - Arc diagrams
- Data
  - Original: network
  - Derived: node ordering attribute (global computation)
- Node ordering crucial to avoid excessive clutter from edge crossings
  - Barycentric ordering before & after
  - Derived attribute: global computation

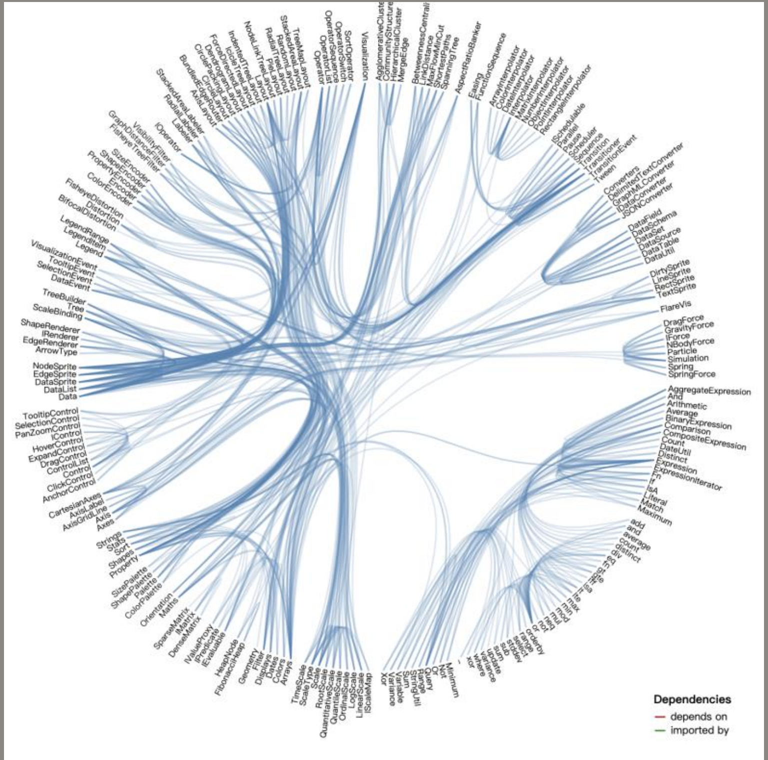


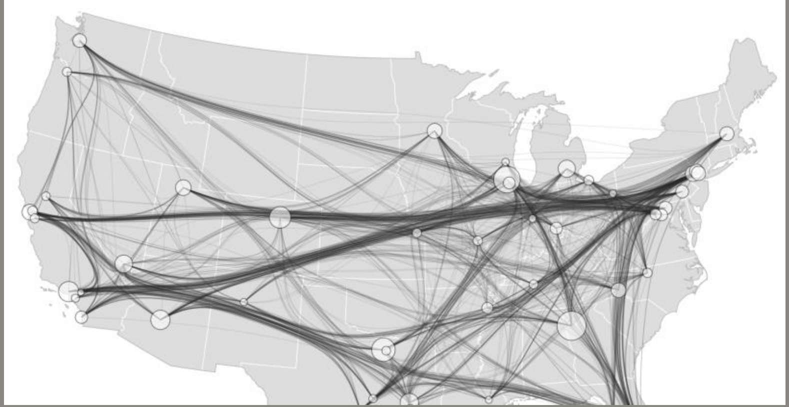




Reduce Edge Visual Clutter: Edge Bundling

- Demonstration: https://vega.github.io/vega/examples/edge-bundling/

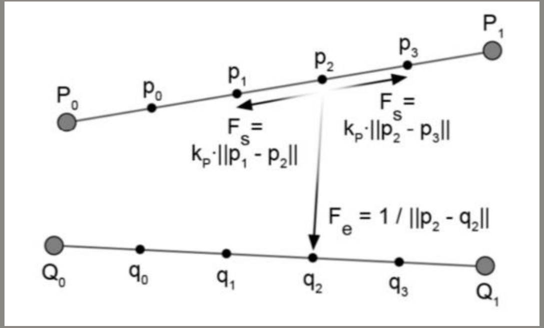


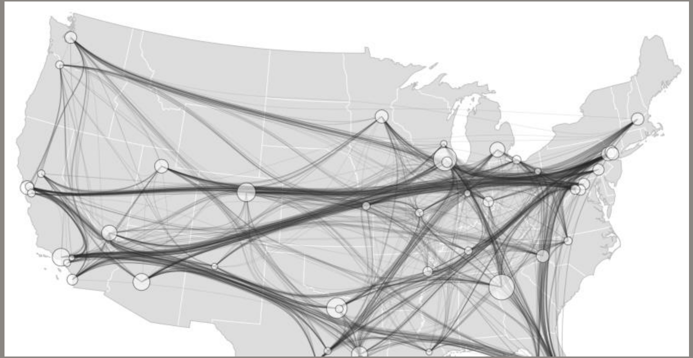


D3: https://www.d3-graph-gallery.com/bundle

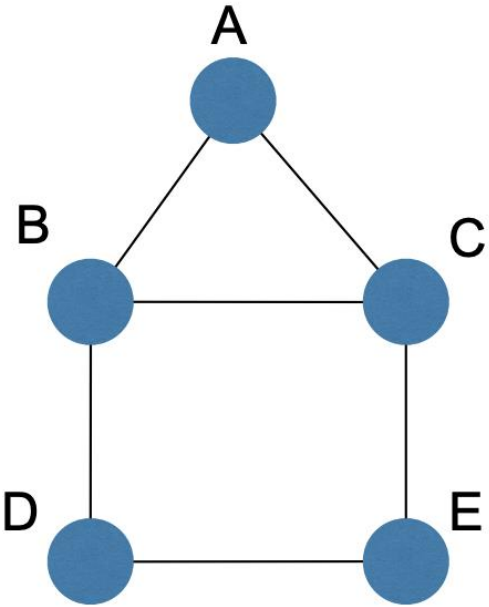
Reduce Edge Visual Clutter: Edge Bundling

- Too many methods for edge bundling
- Example: force-directed edge bundling
  - Idea:
    - add nodes to an edge
    - Corresponding nodes on two edge attract each other
  - Exception: no force cases (two edges). (1) almost perpendicular with each other (2) difference of length are too large (3) center nodes are too far away


P_1
p_3
p_2
p_1
p_0
P_0
$F_s = k_p \cdot \|p_2 - p_3\|$
$F_s = k_p \cdot \|p_1 - p_2\|$
$F_e = 1 / \|p_2 - q_2\|$
Q_0
q_0
q_1
q_2
q_3
Q_1

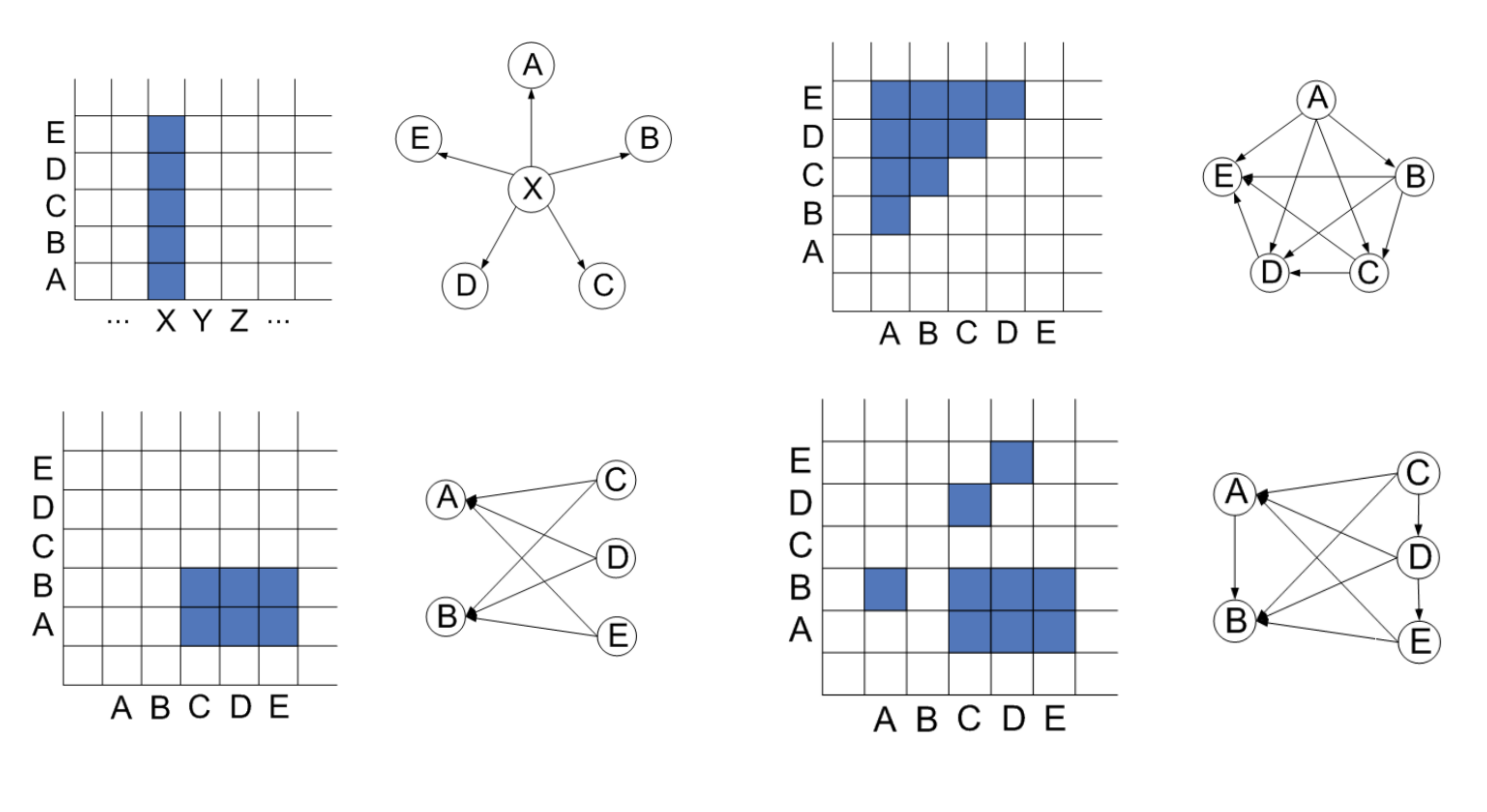


Adjacency Matrix Representation



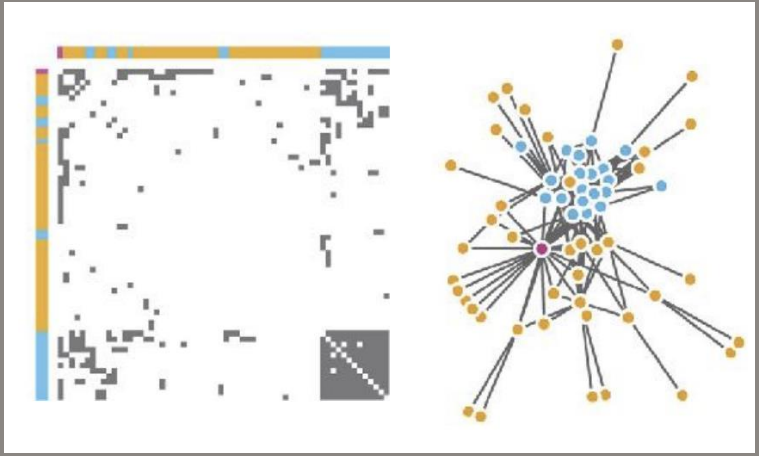
	A	B	C	D	E
A		■	■		
B	■		■	■	
C	■	■			■
D		■			■
E			■	■	

Adjacency Matrix Examples

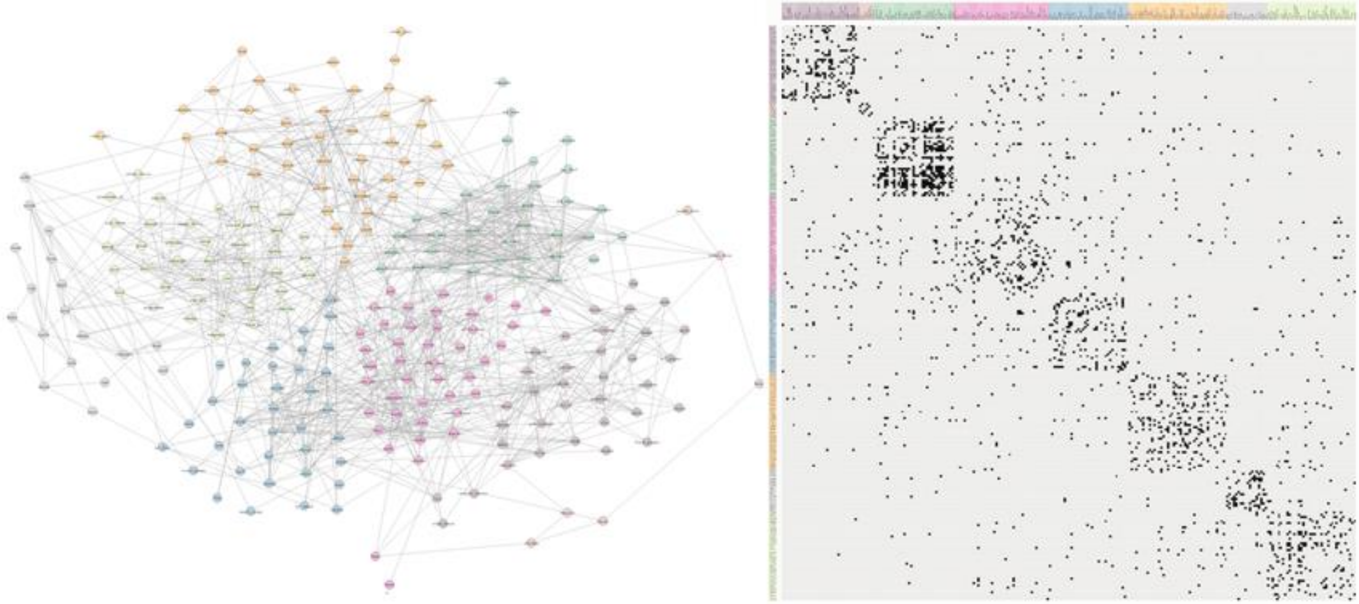


Idiom: Adjacency Matrix View

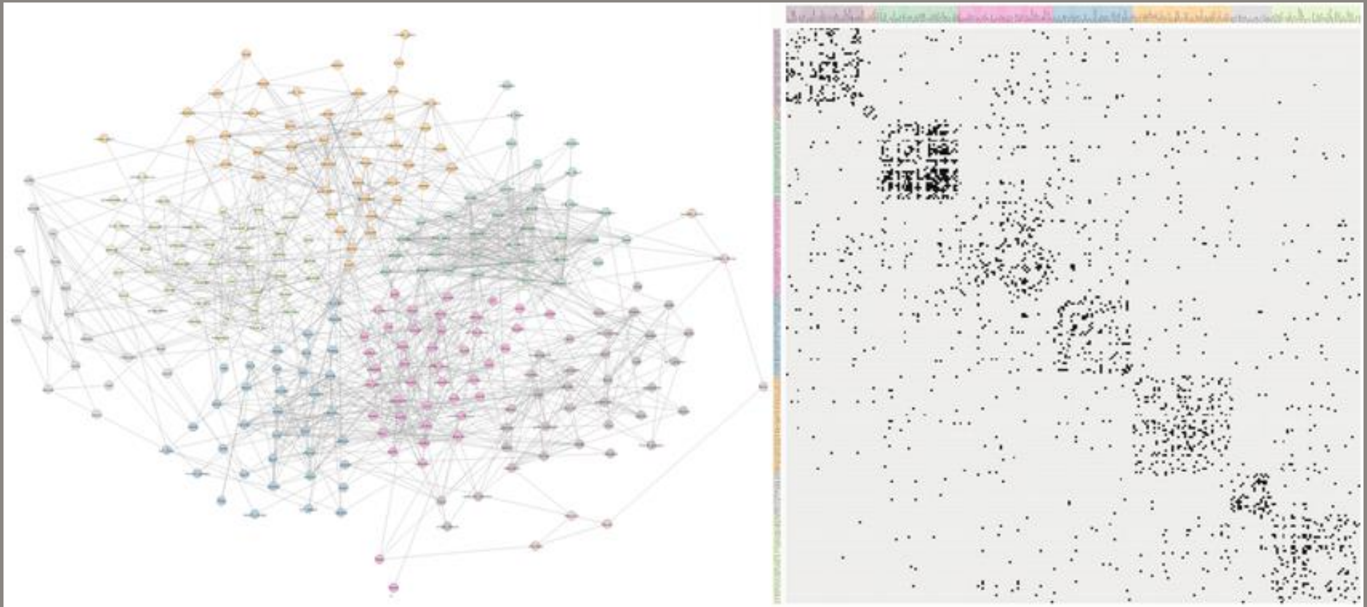
- Data: network
  - Transform into same data/encoding as heatmap
- Derived data: table from network
  - 1 quantitative attribute
    - Weighted edge between nodes
  - 2 categorical attributes:
    - Node list * 2
- Visual encoding
  - Cell shows presence/absence of edge (or weight of the edge)
- Scalability
  - 1K nodes, 1M edges



Which one is better visualization?

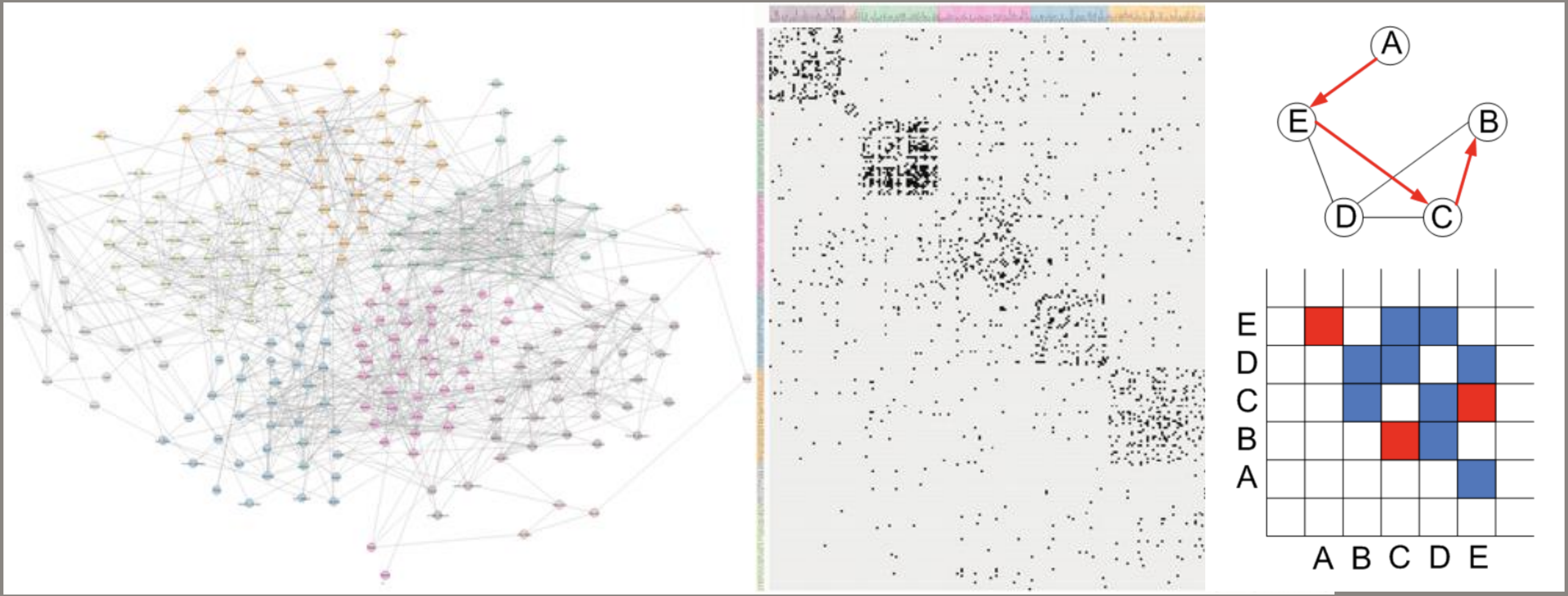


Which one is better visualization?



- What you want to do?
  - Path tracing?

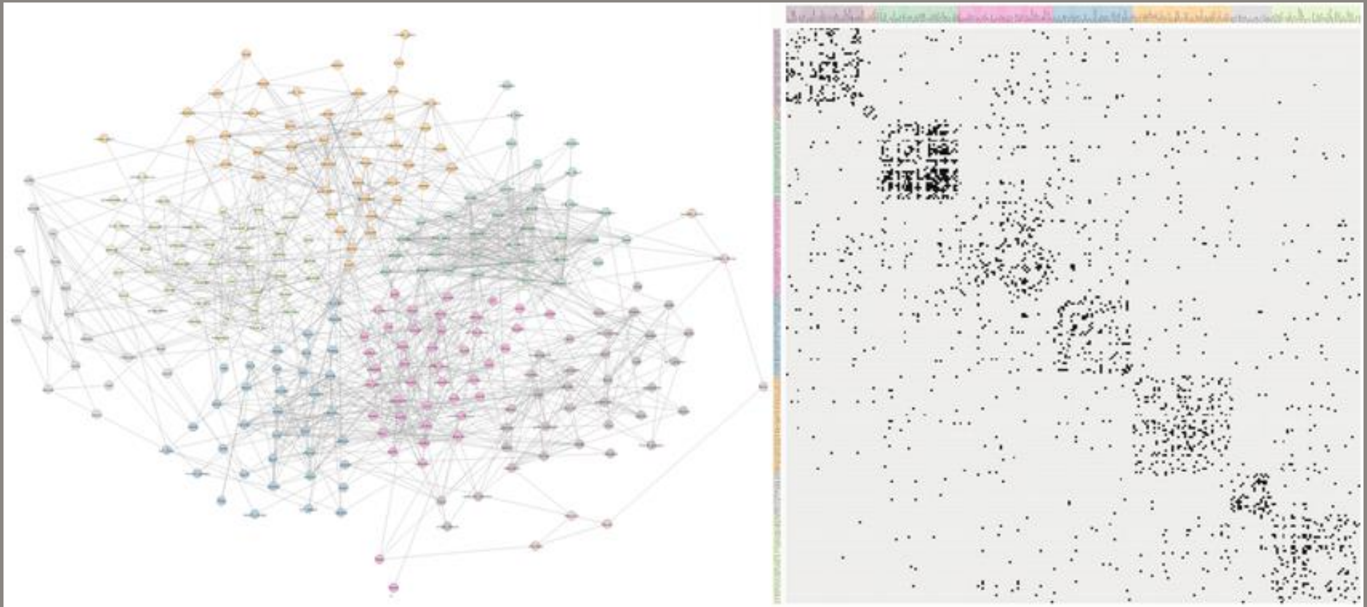
Which one is better visualization?



	A	B	C	D	E
E	🟥		🟦	🟦	
D		🟦	🟦		🟦
C		🟦		🟦	🟥
B			🟥	🟦	
A					🟦

- What you want to do?
  - Path tracing?

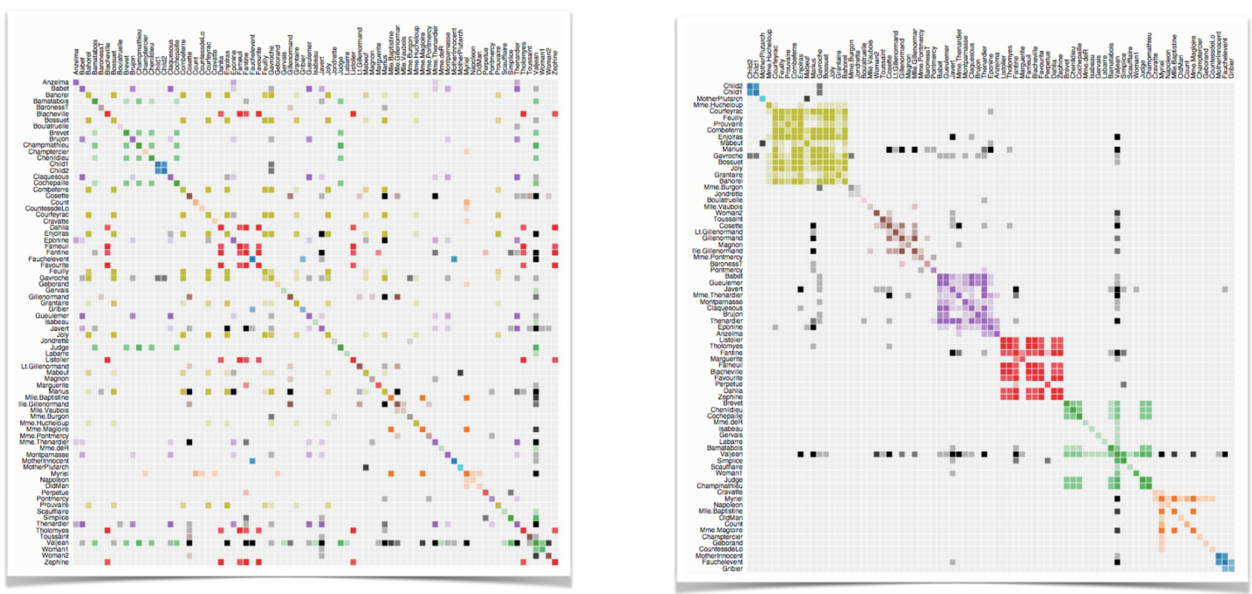
Which one is better visualization?



- What do you want to do?
  - Recognize topological clusters in complex network (good reordering is needed)

Order is Crucial: Reordering

- Easy to find cluster with good order

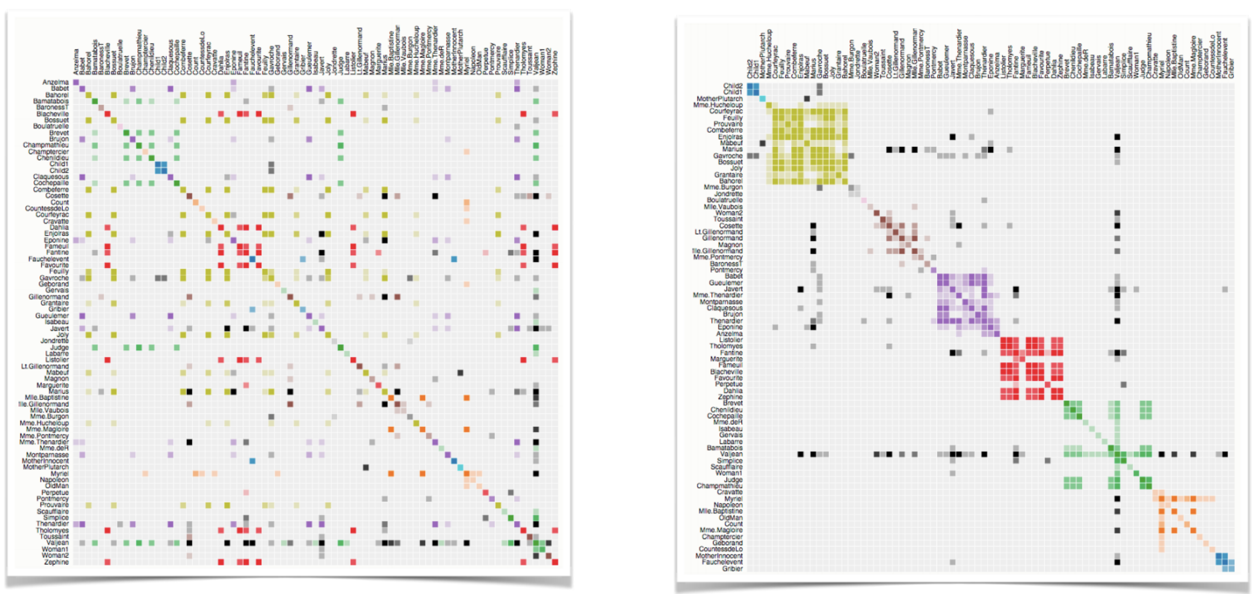


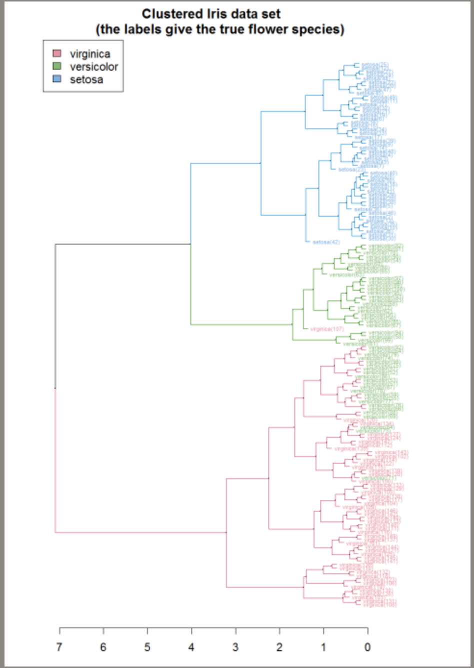
Demo: https://bost.ocks.org/mike/miserables/

Order is Crucial: Reordering

- Easy to find cluster with good order

One way to determine the order
Hierarchical clustering





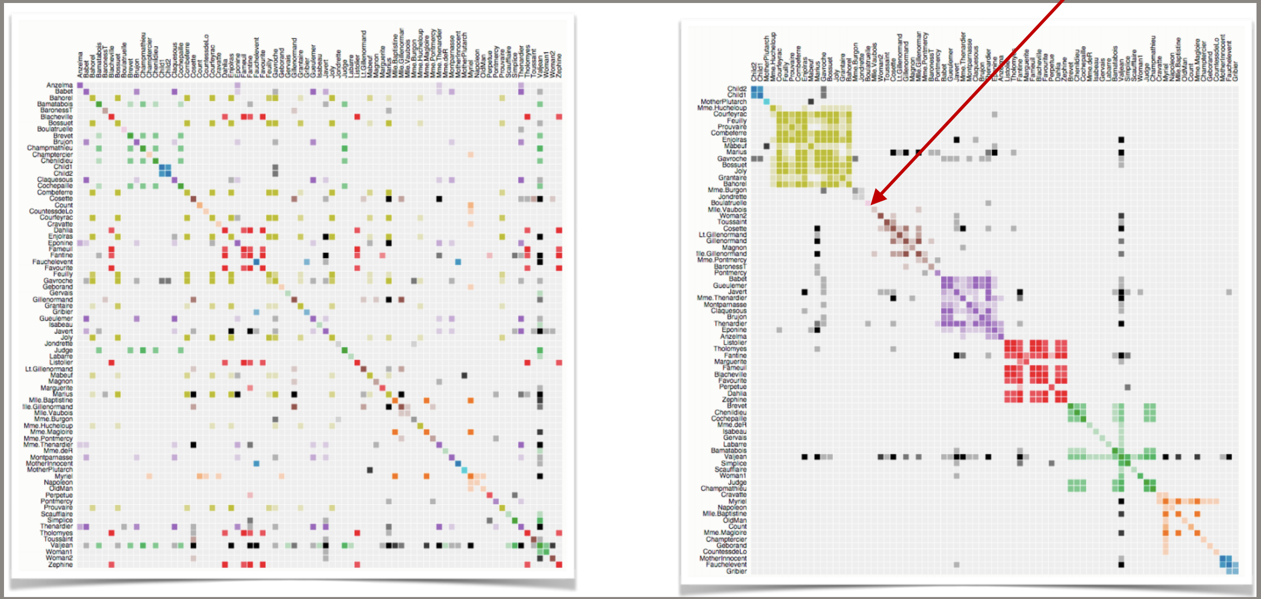
Demo: https://bost.ocks.org/mike/miserables/

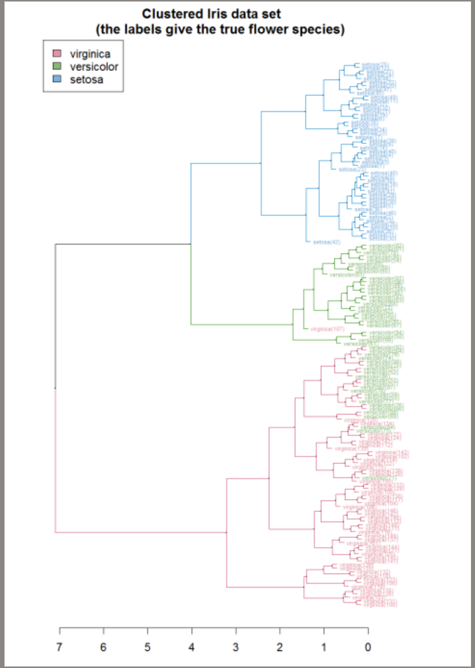
Order is Crucial: Reordering

- Easy to find cluster with good order

More about visual encoding: diagonal of the matrix may to available to encode extra information

One way to determine the order
Hierarchical clustering



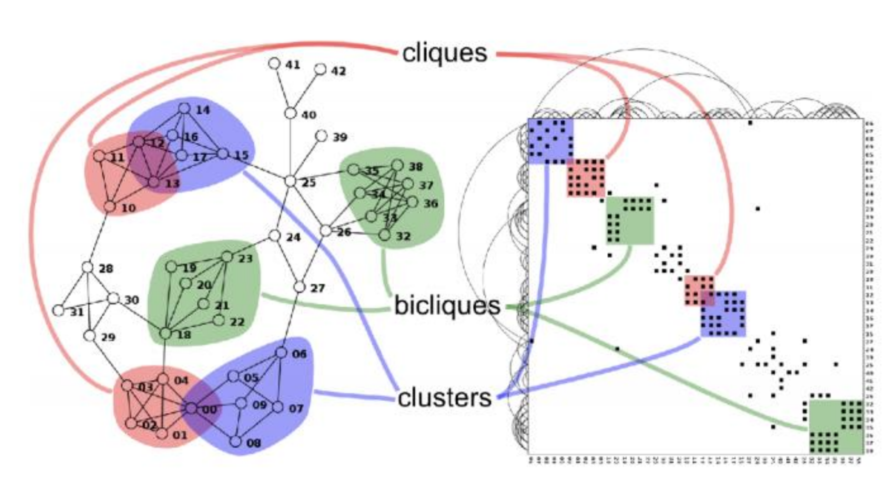


Demo: https://bost.ocks.org/mike/miserables/

S07-02

Node-link vs. Matrix

- Node-link diagram strengths
  - Topology understanding, path tracing
  - Intuitive, flexible, no training needed
- Adjacency matrix strength
  - Focus on edge rather than nodes
  - Layout straightforward (reordering needed)
  - Predictability, scalability
  - Some topology task trainable
- Empirical study
  - Node-link best for small networks
  - Matrix bests for large networks
    - If tasks do not involve path tracing

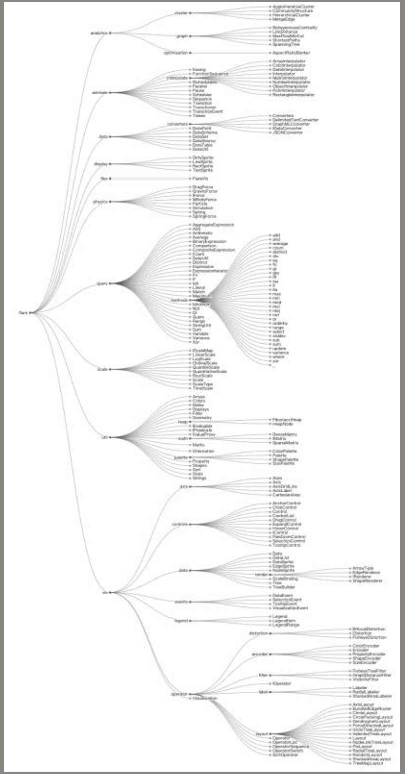


Tree

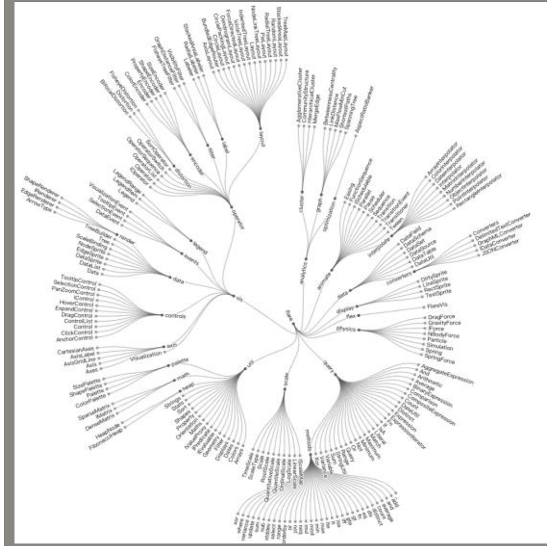
Idiom: Node-Link Trees

- Node-link tree
  - Tidy drawing
  - Clear parent/child structure
  - Compact without overlap
  - Rectilinear and radial variants

- Data: tree
- Encoding:
  - Link connection marks, point node marks
  - Distance from root: depth in the tree
  - Angular (radial), horizontal(regular) proximity: siblings
- Tasks
  - Understanding topology, following paths
- Scalability
  - Regular: several dozens - hundreds nodes
  - Radial: 1K – 10K nodes



D3: https://observablehq.com/@d3/tidy-tree

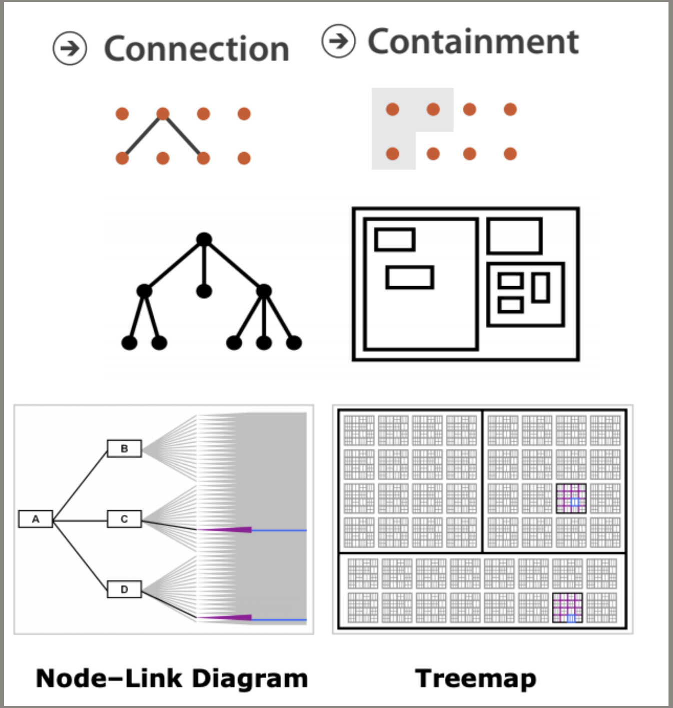


D3: https://observablehq.com/@d3/radial-tidy-tree

Two Ways to Represent Links

- Connection
  - All node-link diagrams
  - Emphasize topology, path tracing
  - Networks and trees

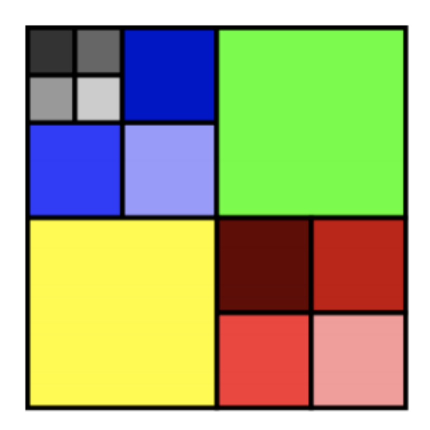
- Containment
  - All treemap/sunburst/icicle variants
  - Emphasize **attribute** values at **leaves**
  - Only trees



Containment Tree Layout

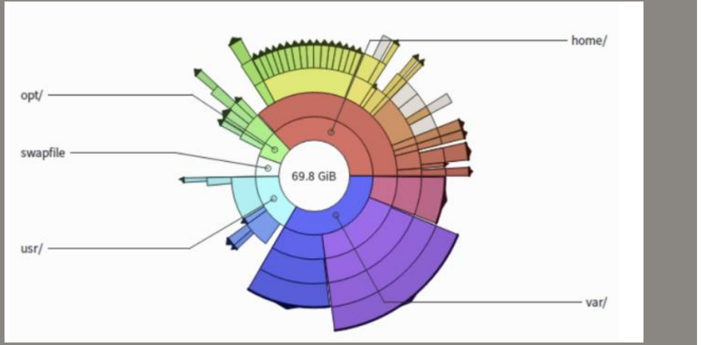
- Implicitly visualize the tree structure

Tree map



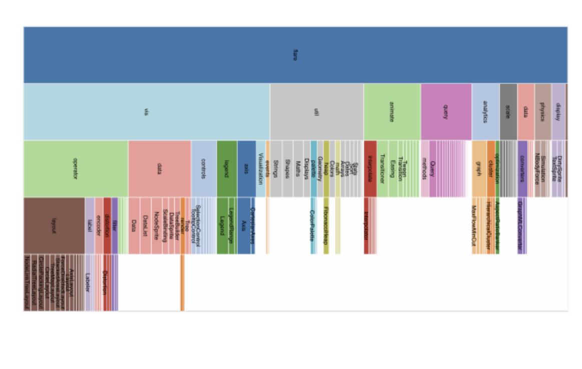
D3: https://www.d3-graph-gallery.com/treemap

Sunburst



D3: https://observablehq.com/@d3/zoomable-sunburst

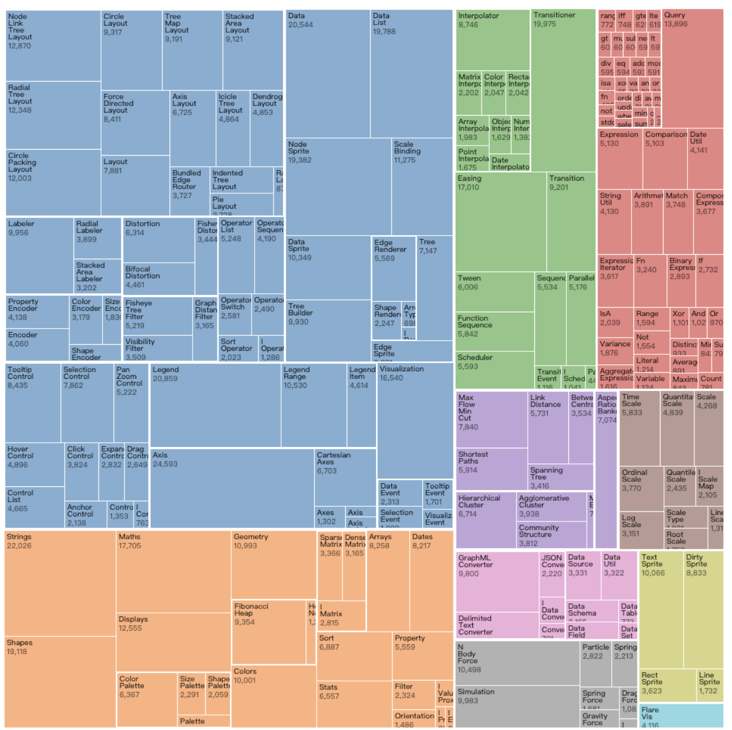
Icicle Plot



D3: https://observablehq.com/@d3/zoomable-icicle

Idiom: treemap

- Data
  - Tree
  - 1 quantitative attribute at leaf nodes
- Encoding
  - Area containment marks for hierarchical structure
  - Rectilinear orientation
  - Size encodes quantitative attribute
- Tasks
  - Query attribute at leaf nodes
- Scalability
  - 1M leaf nodes

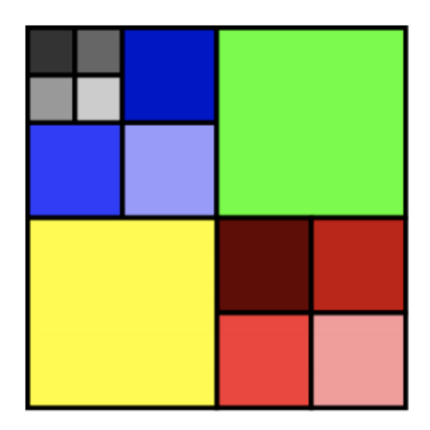


Containment Tree Layout

- Implicitly visualize the tree structure

Only leaf node visible

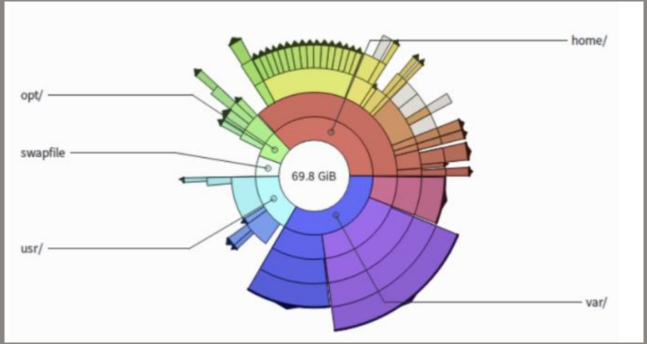
Tree map



D3: https://www.d3-graph-gallery.com/treemap

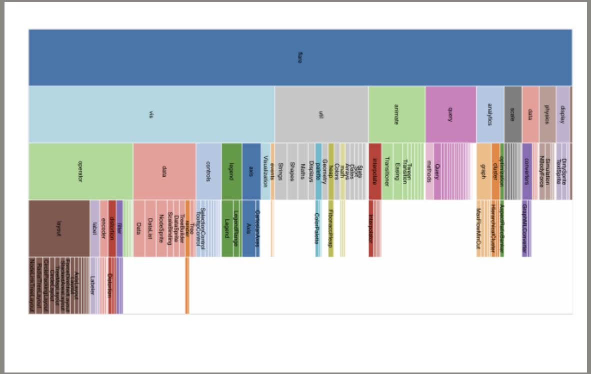
Inner node and leaf node visible

Sunburst



D3: https://observablehq.com/@d3/zoomable-sunburst

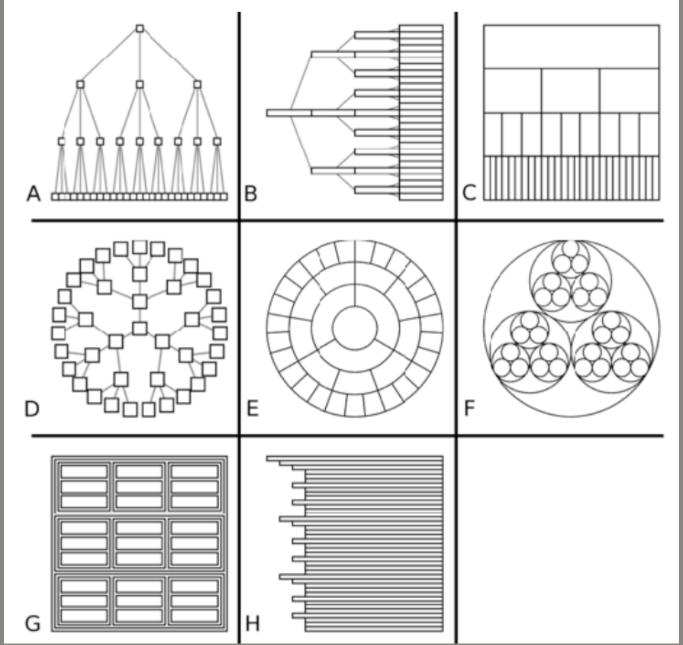
Icicle Plot



D3: https://observablehq.com/@d3/zoomable-icicle

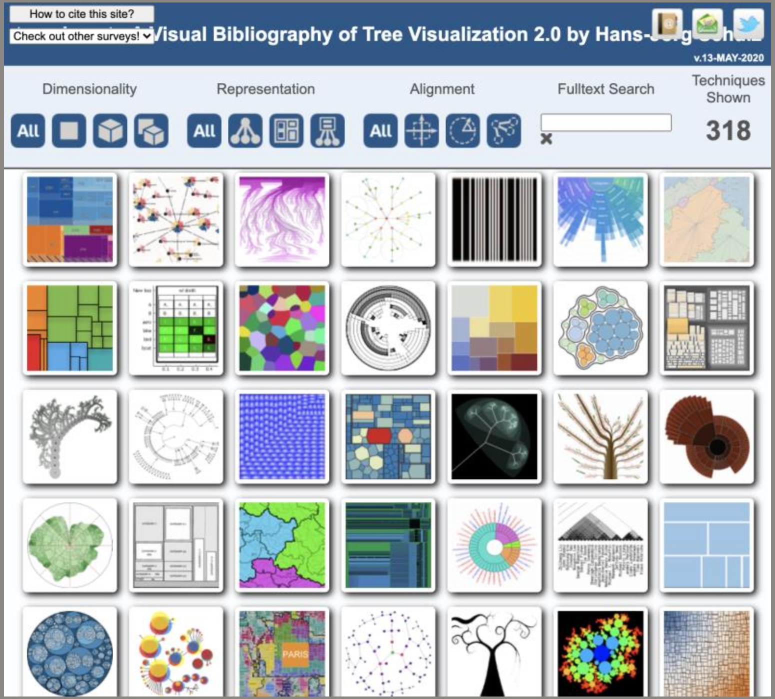
Tree Drawing Idioms Choice

- What you want to shown
  - Link relationships
  - Tree depth
  - Sibling order
- Design choices
  - Connection or containment for links
  - Rectilinear vs radial layout
  - Spatial position channels
- Considerations
  - Information density
    - Avoid wasting space
    - Consider where to fit **labels**



treevis.net

- Check to see more examples for tree vis



S07-03