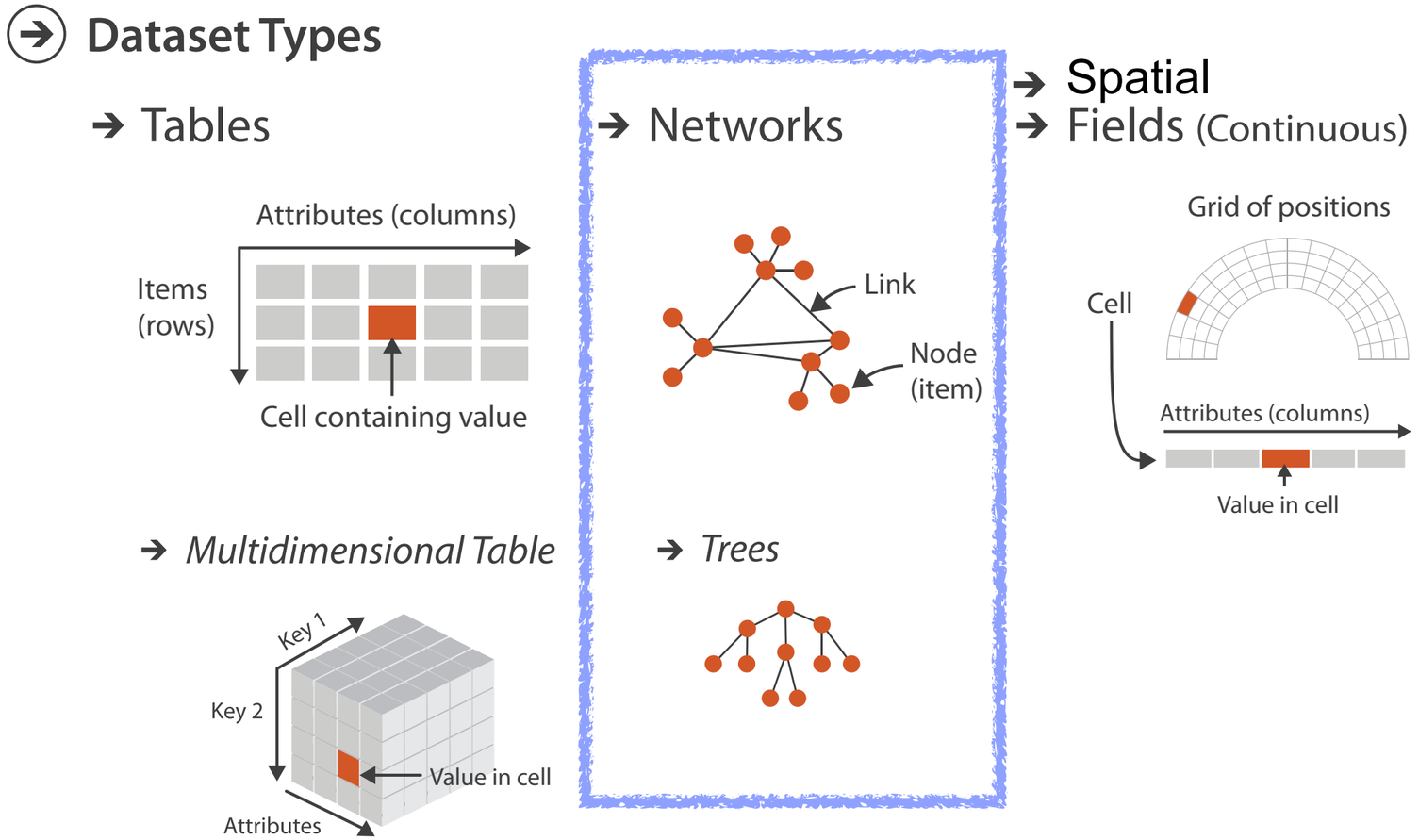
Networks and Graphs!

Adapted from Tamara Munzner

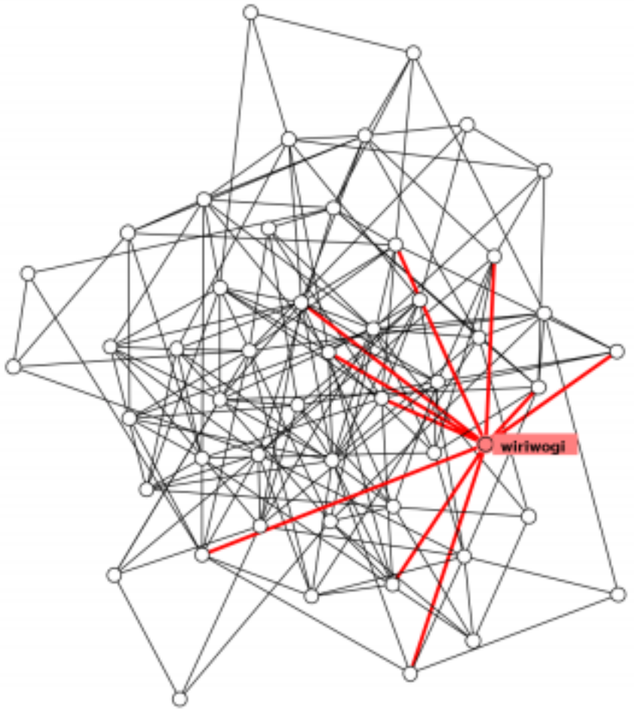
Network data

- networks
  - model relationships between things
    - aka graphs
  - two kinds of items, both can have attributes
    - nodes
    - links
- tree
  - special case
  - no cycles
    - one parent per node



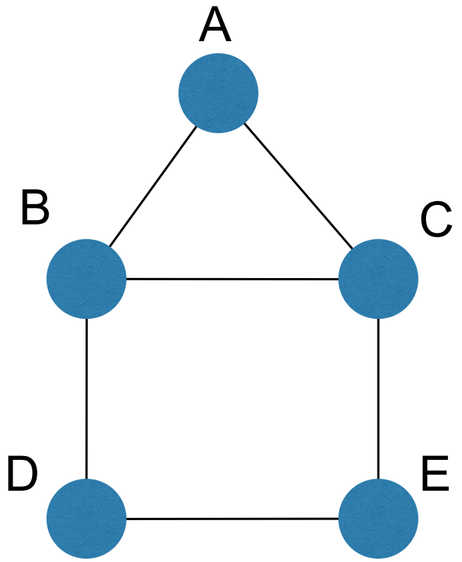
Network tasks: topology-based and attribute-based

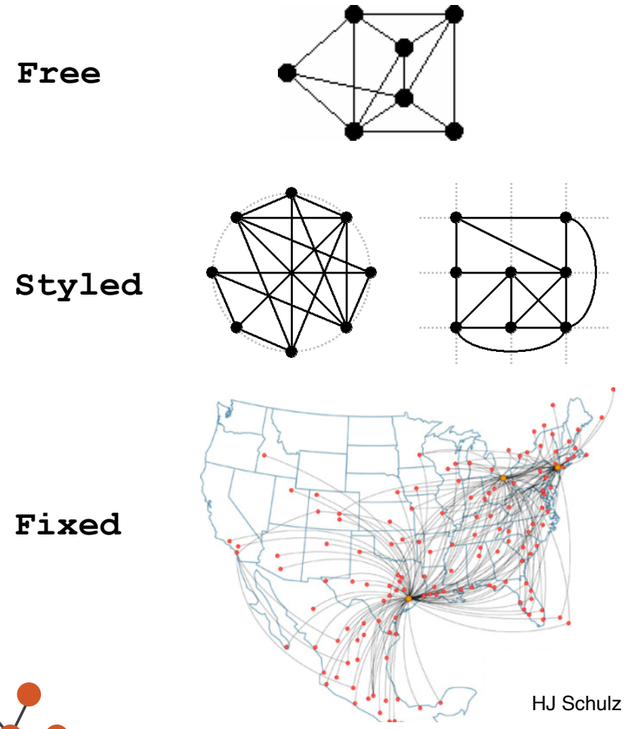
- topology based tasks
  - find paths
  - find (topological) neighbors
  - compare centrality/importance measures
  - identify clusters / communities
- attribute based tasks (similar to table data)
  - find distributions, ...
- combination tasks, incorporating both
  - example: find friends-of-friends who like cats
    - topology: find all adjacent nodes of given node
    - attributes: check if has-pet (node attribute) == cat



Node-link diagrams

- nodes: point marks
  - level 1
- links: segment connecting marks
  - straight lines or arcs
  - level 2: inherit node locations
    - unavailable: position/order
    - unavailable: length (1D size)
      - show connections between nodes, not express length of quant attribute
- intuitive & familiar
  - most common
  - family of idioms
    - many many variants
    - differences with channel usage

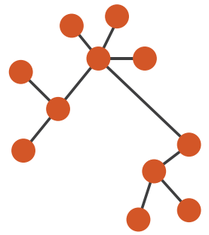




HJ Schulz 2006

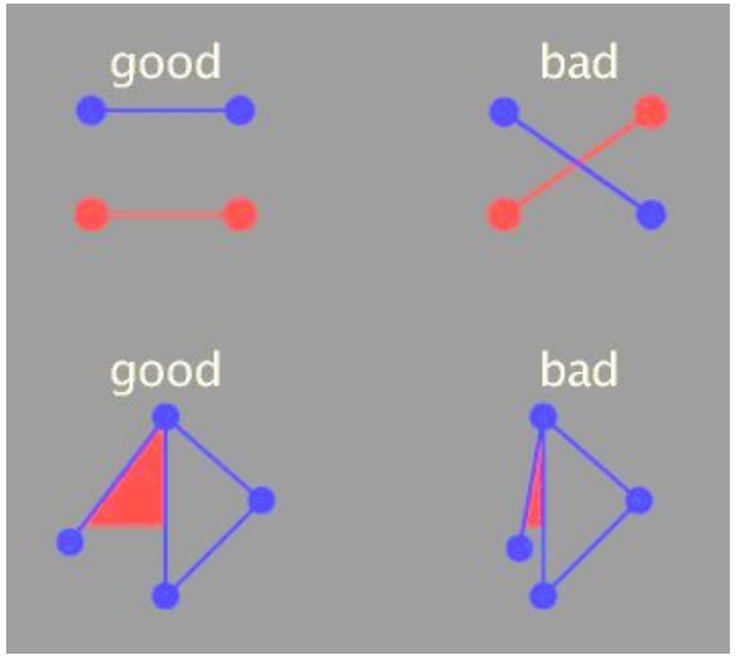
Node–Link Diagrams
Connection Marks

✔ NETWORKS ✔ TREES



Criteria for good node-link layouts

- minimize
  - edge crossings, node overlaps
  - distances between topological neighbor nodes
  - total drawing area
  - edge bends
- maximize
  - angular distance between different edges
  - aspect ratio disparities
- emphasize symmetry
  - similar graph structures should look similar in layout



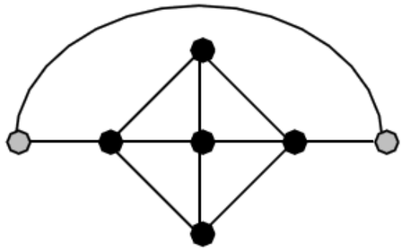
Criteria conflict

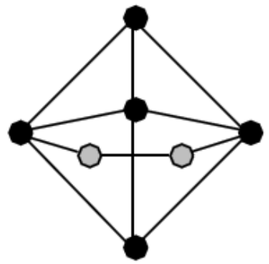
- most criteria NP-hard individually
- many criteria directly conflict with each other

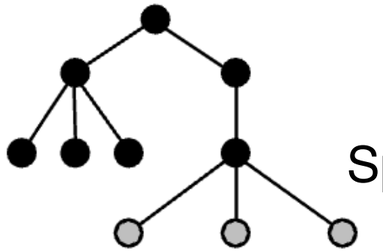
Minimum number of edge crossings

vs.

Uniform edge length



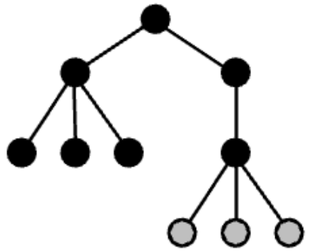




Space utilization

vs.

Symmetry



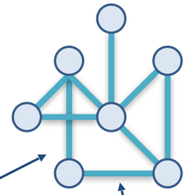
Schulz 2004

Optimization-based layouts

- formulate layout problem as optimization problem
- convert criteria into weighted cost function
  - F(layout) = a*[crossing counts] + b*[drawing space used]+...
- use known optimization techniques to find layout at minimal cost
  - energy-based physics models
  - force-directed placement
  - spring embedders

Force-directed placement

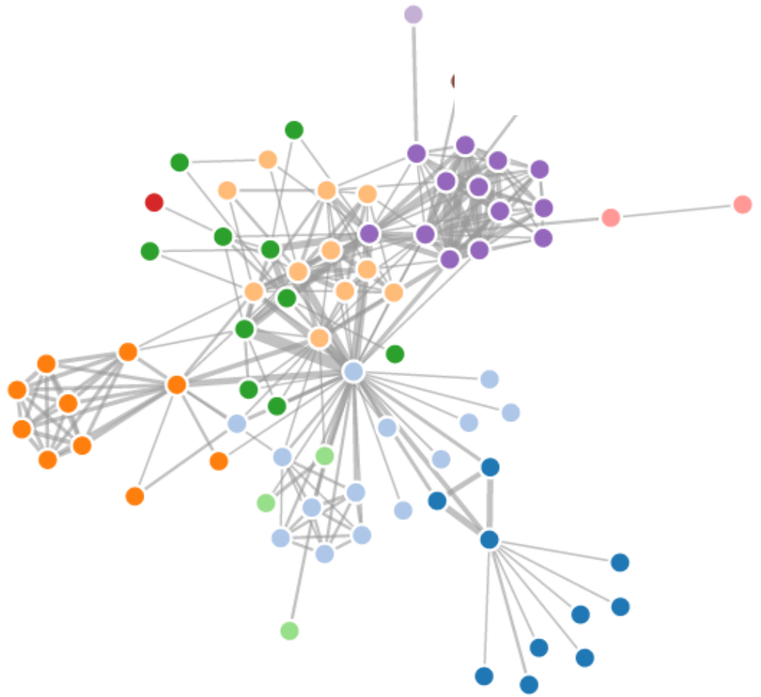
- physics model
  - links = springs pull together
  - nodes = magnets repulse apart



Expander
(pushing nodes apart)

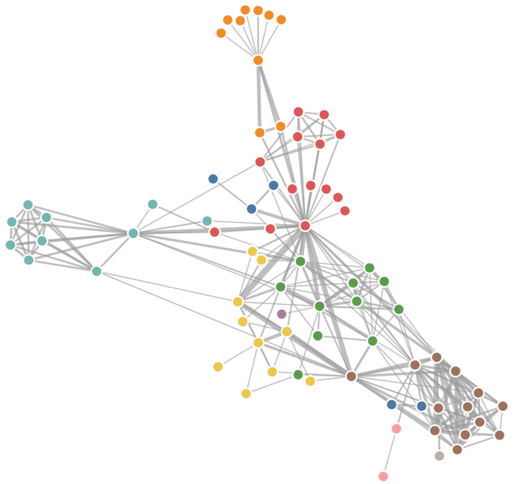
Spring Coil
(pulling nodes together)

- algorithm
  - place vertices in random locations
  - while not equilibrium
    - calculate force on vertex
      - sum of
        - » pairwise repulsion of all nodes
        - » attraction between connected nodes
    - move vertex by c * vertex_force



http://mbostock.github.com/d3/ex/force.html

Typical network data format: Node and Edge lists



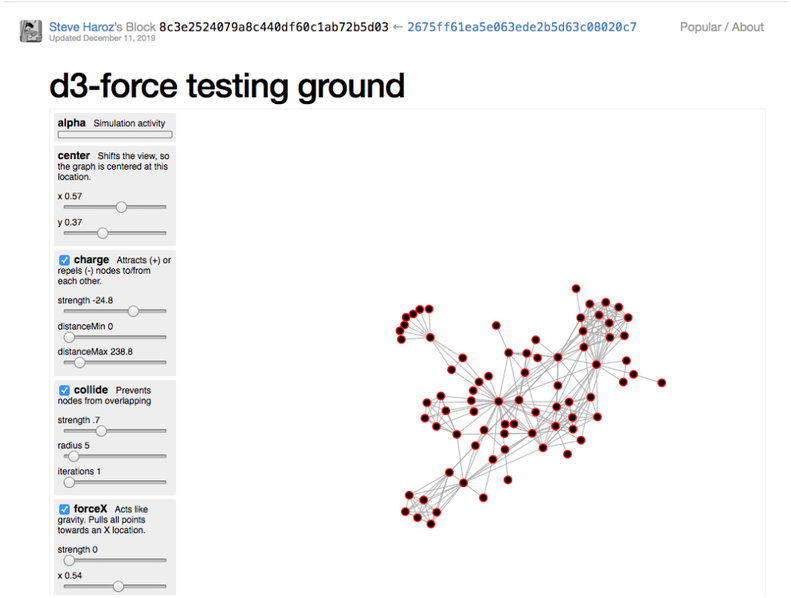
```
miserables = ▼Object {
  nodes: ▼Array(77) [
    0: ▶Object {id: "Myriel", group: 1}
    1: ▶Object {id: "Napoleon", group: 1}
    2: ▶Object {id: "Mlle.Baptist
    3: ▶Object {id: "Mme.Magloire
    4: ▶Object {id: "CountessdeLo
    5: ▶Object {id: "Geborand", group: 1}
    6: ▶Object {id: "Champtercier", group: 1}
    7: ▶Object {id: "Cravatte", group: 1}
    8: ▶Object {id: "Count", group: 1}
    9: ▶Object {id: "OldMan", group: 1}
    10: ▶Object {id: "Labarre", group: 2}
    11: ▶Object {id: "Valjean", group: 2}
    12: ▶Object {id: "Marguerite", group: 3}
    13: ▶Object {id: "Mme.deR", group: 2}
    14: ▶Object {id: "Isabeau", group: 2}
    15: ▶Object {id: "Gervais", group: 2}
    16: ▶Object {id: "Tholomyes", group: 3}
    17: ▶Object {id: "Listolier", group: 3}
    18: ▶Object {id: "Fameuil", group: 3}
    19: ▶Object {id: "Blacheville", group: 3}
    … more
  ]
```

Edges reference node table

```
  links: ▼Array(254) [
    0: ▶Object {source: "Napoleon", target: "Myr
    1: ▶Object {source: "Mlle.Baptistine", targe
       "Mme.Magloire", target:
       "Mme.Magloire", target:
    4: ▶Object {source: "CountessdeLo", target: "Myriel", value: 1}
    5: ▶Object {source: "Geborand", target: "Myriel", value: 1}
    6: ▶Object {source: "Champtercier", target: "Myriel", value: 1}
    7: ▶Object {source: "Cravatte", target: "Myriel", value: 1}
    8: ▶Object {source: "Count", target: "Myriel", value: 2}
    9: ▶Object {source: "OldMan", target: "Myriel", value: 1}
    10: ▶Object {source: "Valjean", target: "Labarre", value: 1}
    11: ▶Object {source: "Valjean", target: "Mme.Magloire", value: 3}
    12: ▶Object {source: "Valjean", target: "Mlle.Baptistine", value: 3}
    13: ▶Object {source: "Valjean", target: "Myriel", value: 5}
    14: ▶Object {source: "Marguerite", target: "Valjean", value: 1}
    15: ▶Object {source: "Mme.deR", target: "Valjean", value: 1}
    16: ▶Object {source: "Isabeau", target: "Valjean", value: 1}
    17: ▶Object {source: "Gervais", target: "Valjean", value: 1}
    18: ▶Object {source: "Listolier", target: "Tholomyes", value: 4}
    19: ▶Object {source: "Fameuil", target: "Tholomyes", value: 4}
    … more
  ]
```

Force-directed placement properties

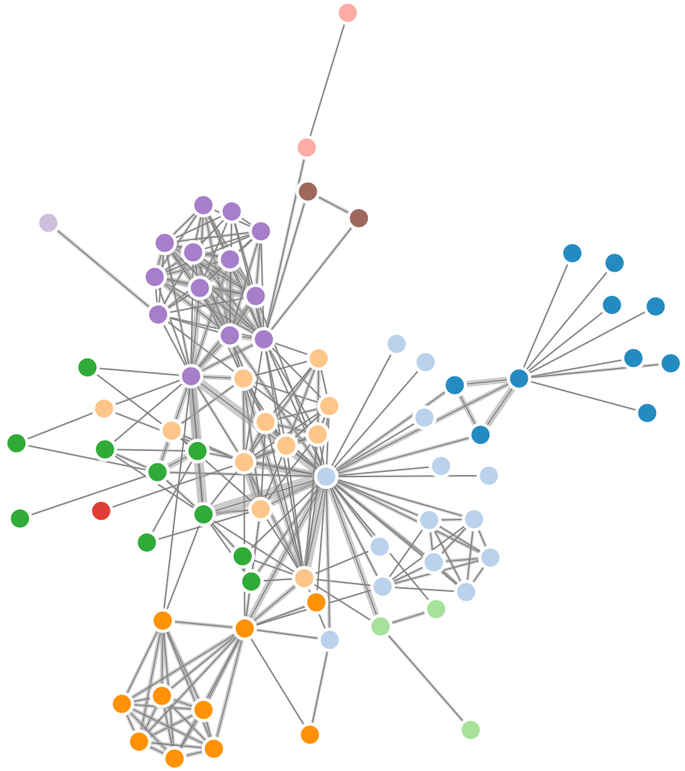
- strengths
  - reasonable layout for small, sparse graphs
  - clusters typically visible
  - edge length uniformity
- weaknesses
  - nondeterministic
  - computationally expensive: O(n^3) for n nodes
    - each step is n^2, takes ~n cycles to reach equilibrium
  - naive FD doesn't scale well beyond 1K nodes
  - iterative progress: engaging but distracting



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

Idiom: **force-directed placement**

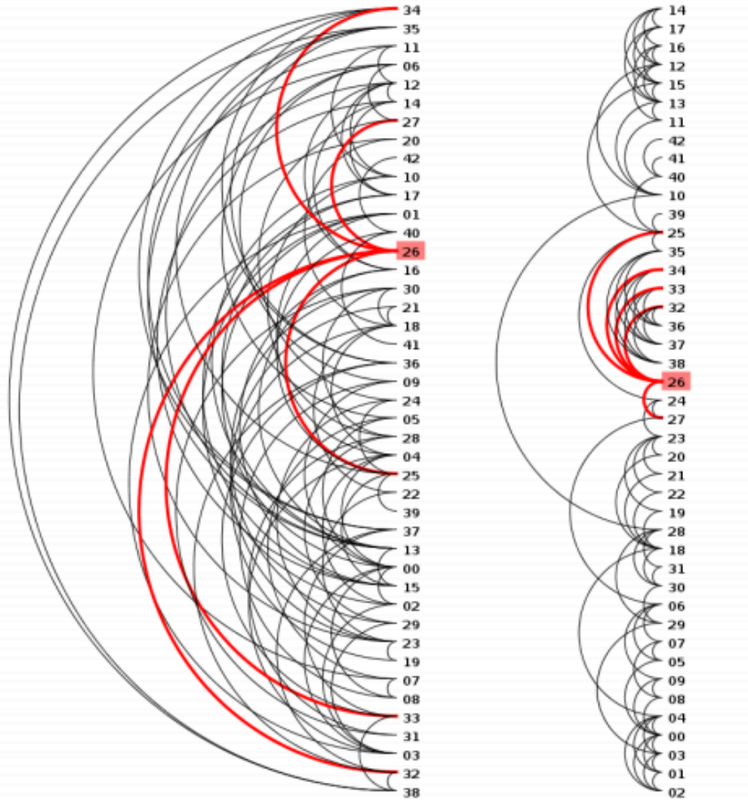
- visual encoding
  - link connection/segment marks, node point marks
  - multilevel: nodes at level 1, links at level 2 with positions given by nodes
  - no shared boundaries
- channels
  - unavailable: position/order & orientation
    - no meaning directly encoded
    - used by algorithm to optimize parameters (minimize crossings, etc)
  - proximity semantics?
    - sometimes meaningful, sometimes arbitrary algorithm outcome
    - tension with length: long edges more visually salient than short
  - free: color, symbol/shape (shape for nodes, line style for links), size (1D width for links, 2D area for nodes)
- tasks
  - explore topology; locate paths, clusters
- scalability
  - node/edge density E < 4N

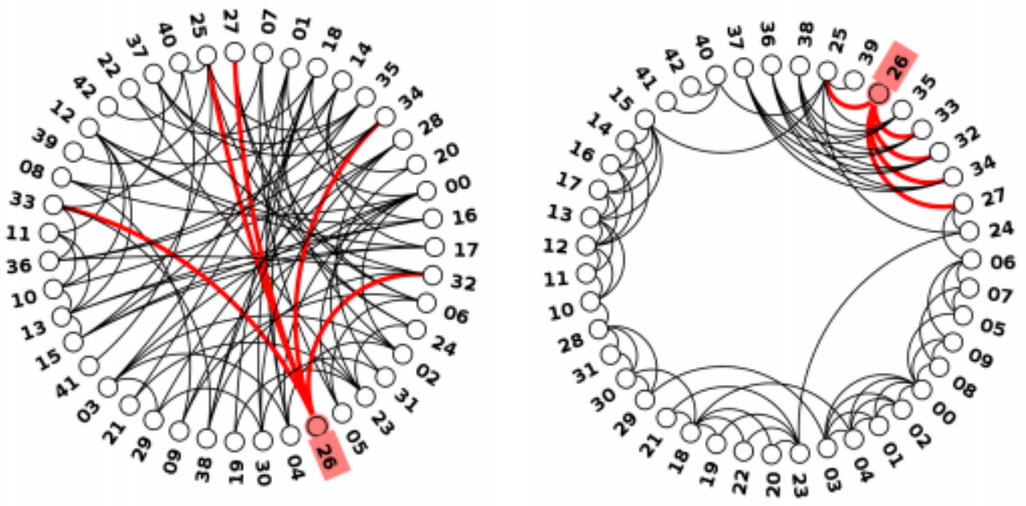


http://mbostock.github.com/d3/ex/force.html

Idiom: **circular layouts / arc diagrams (node-link)**

- special case of node-link layouts with restrictions: lay out nodes around circle or along line
- data
  - original: network
  - derived: node ordering attribute computed to minimize crossings (global algorithm)
- channels for nodes
  - unavailable: position/order (used by algorithm, but no intrinsic meaning)
  - free: color, symbol/shape, orientation
- considerations: node ordering crucial to avoid crossing clutter
  - examples: before & after barycentric ordering

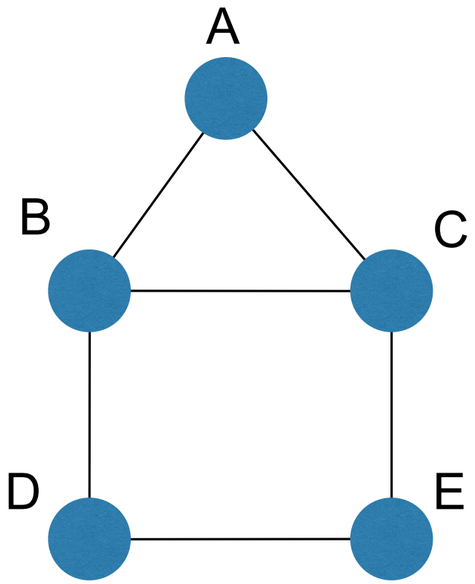


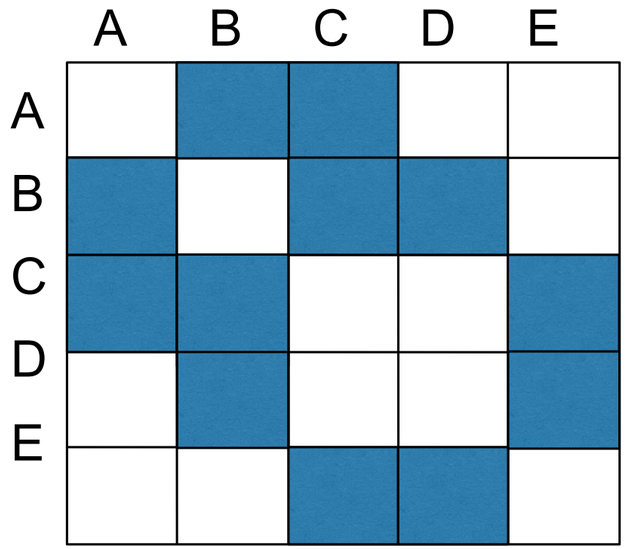


http://profs.etsmtl.ca/mmcguffin/research/2012-mcguffin-simpleNetVis/mcguffin-2012-simpleNetVis.pdf

Adjacency matrix representations

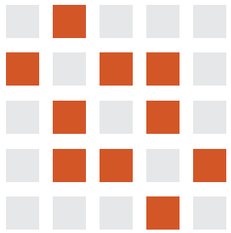
- derive adjacency matrix from network



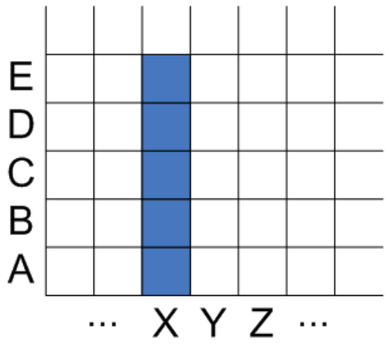


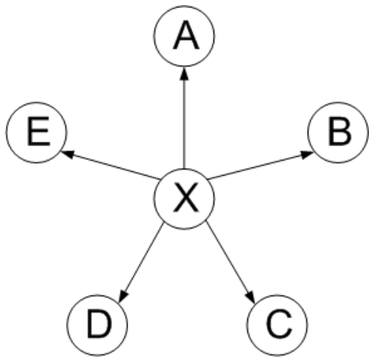
Adjacency Matrix
Derived Table

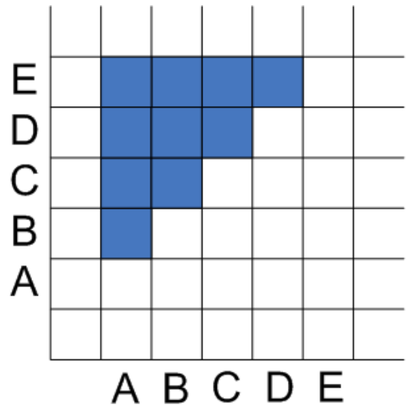
✔ NETWORKS ✔ TREES

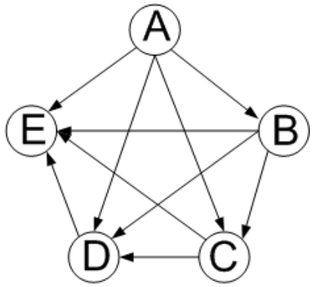


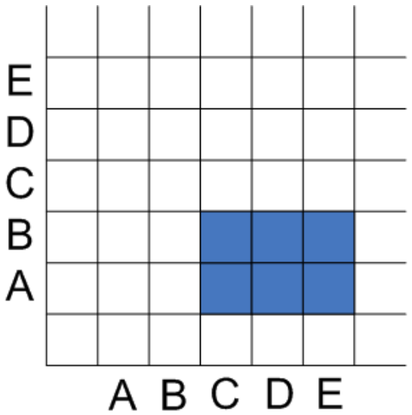
Adjacency matrix examples

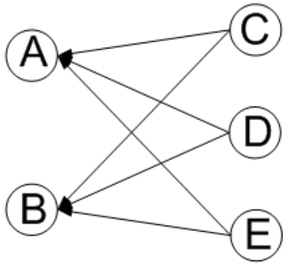


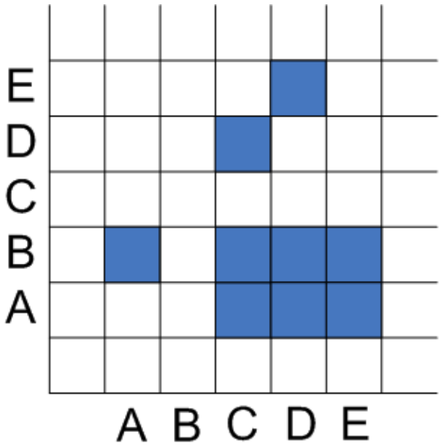


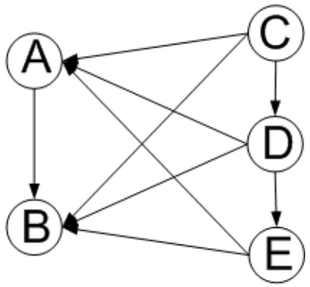






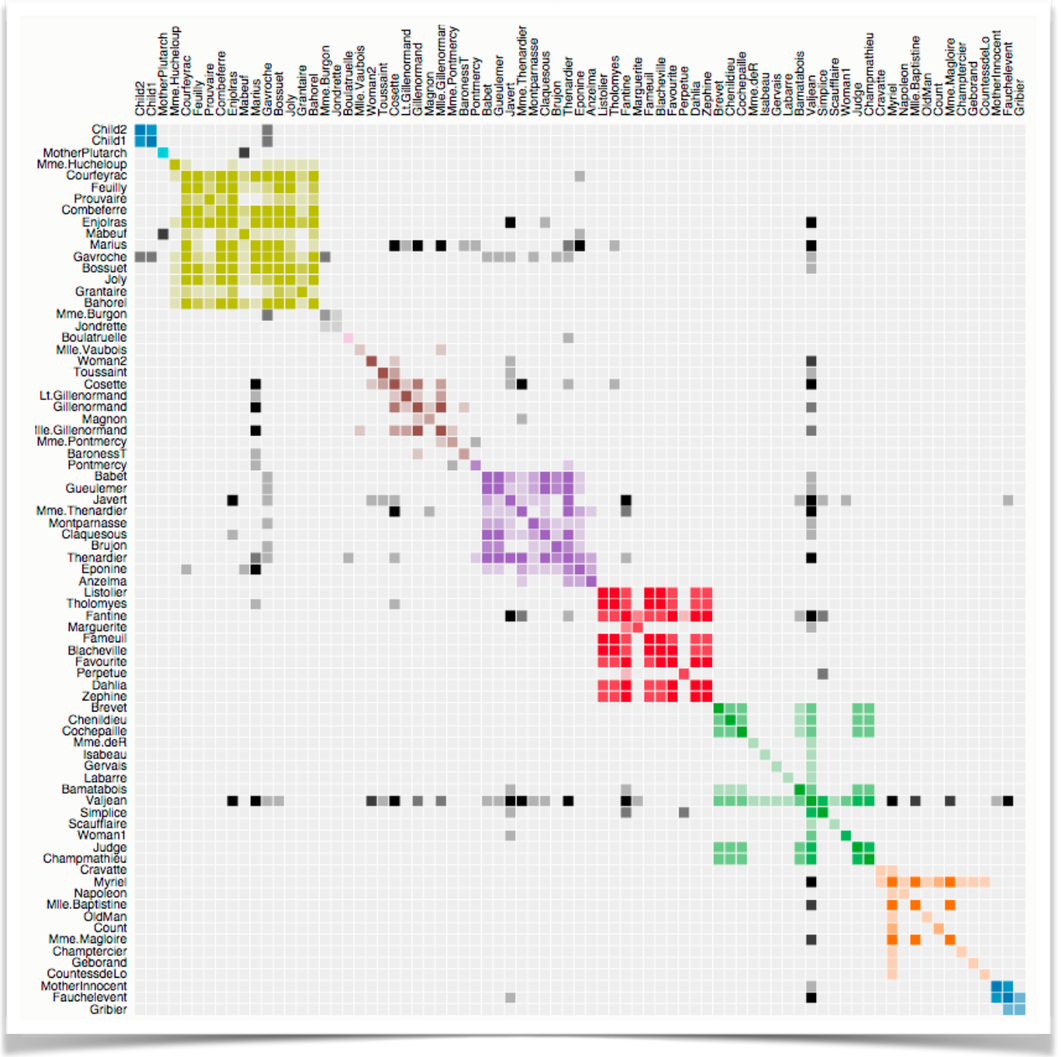






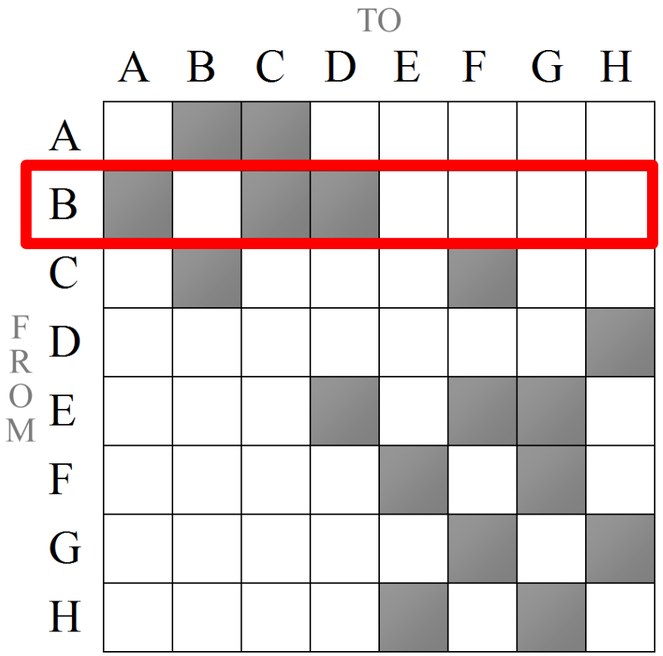
HJ Schulz 2007

Node order is crucial: Reordering

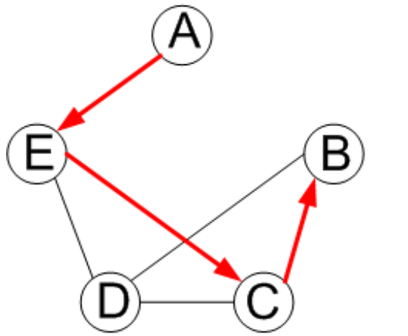


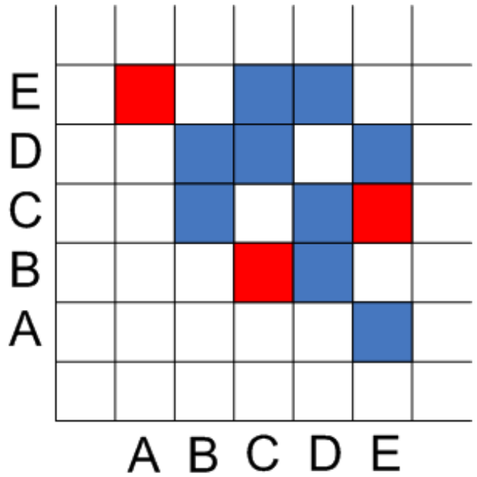
https://bost.ocks.org/mike/miserables/

Adjacency matrix



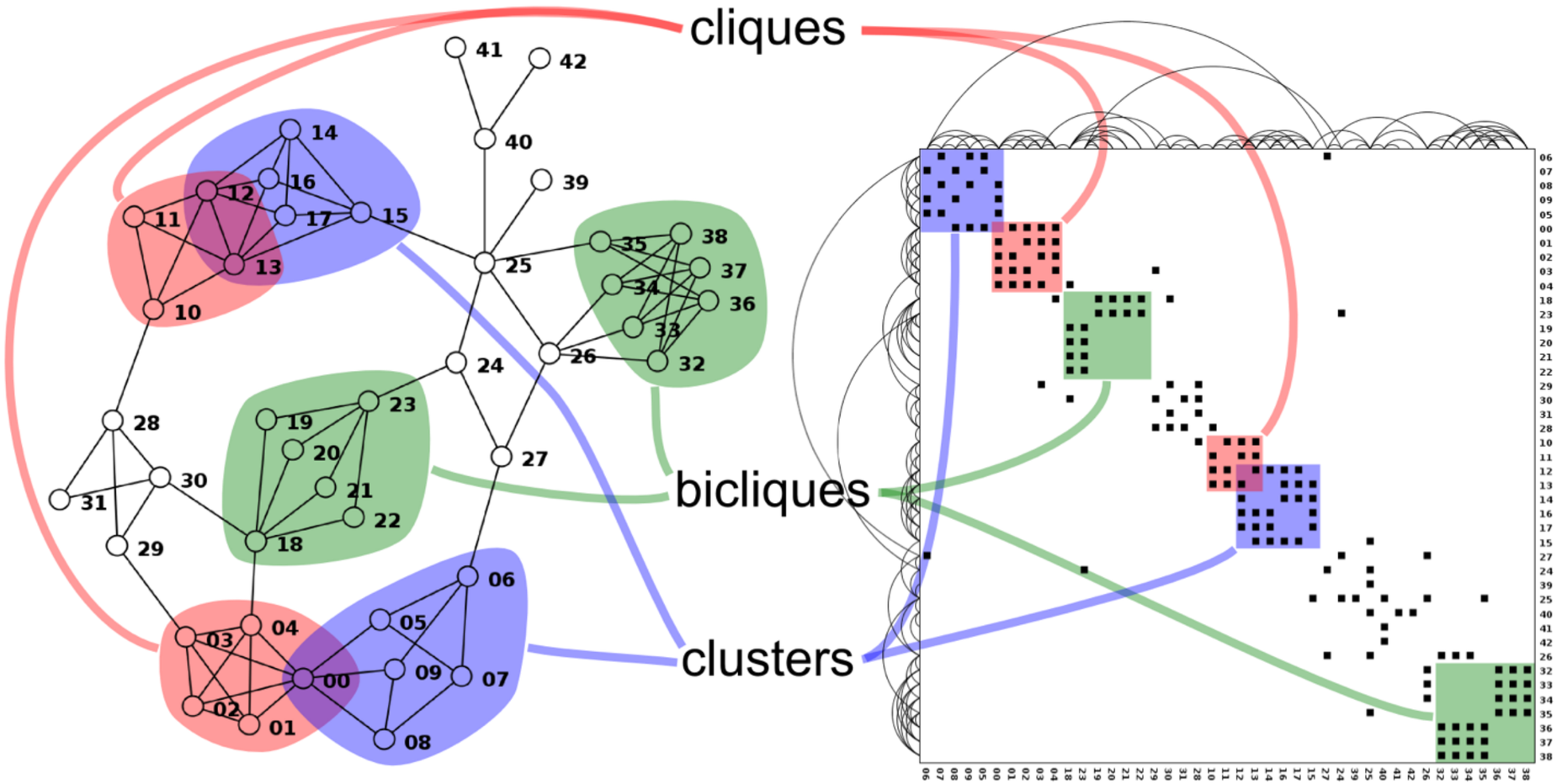
good for topology tasks
related to neighborhoods
(node 1-hop neighbors)





bad for topology tasks
related to paths

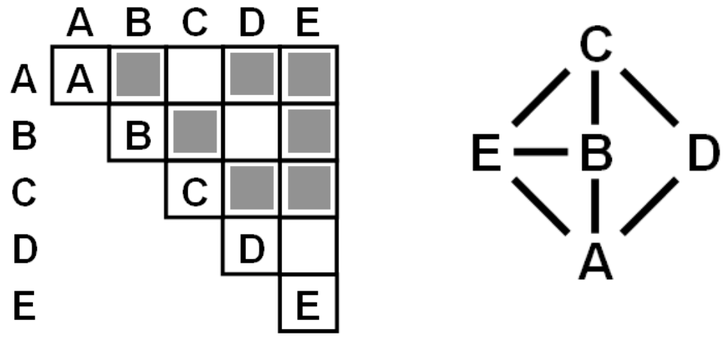
Structures visible in both



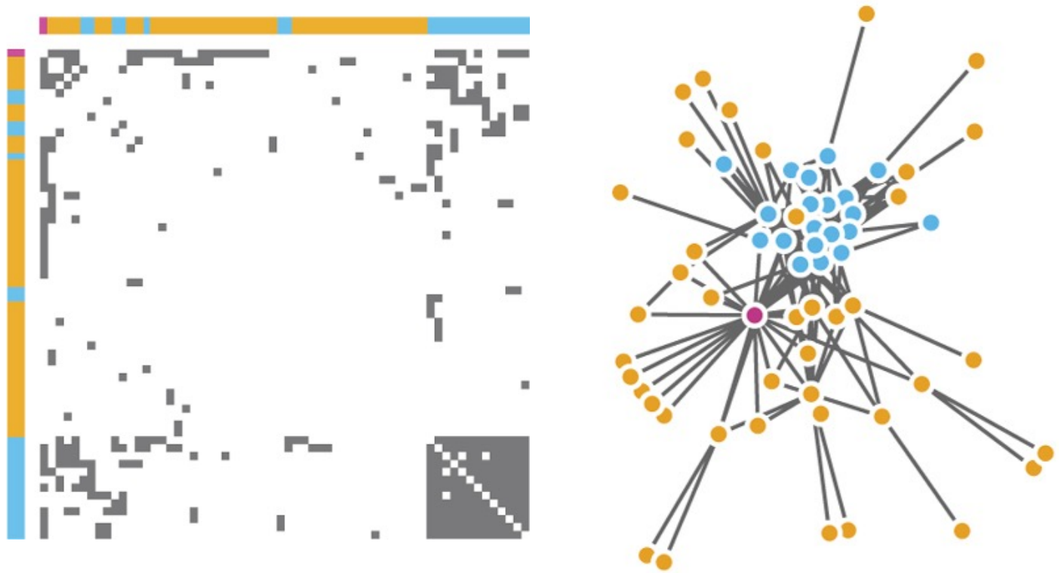
http://www.michaelmcguffin.com/courses/vis/patternsInAdjacencyMatrix.png

Idiom: **adjacency matrix view**

- data: network
  - transform into same data/encoding as heatmap
- derived data: table from network
  - 1 quant attrib: weighted edge between nodes
  - 2 categ attribs: node list x 2
  - 1 quant attrib: node ordering
    - computed from topology to maximize visible patterns
- visual encoding
  - point mark with shared boundaries in grid
    - cell shows presence/absence of edge
  - channels
    - unavailable: position/order (if node order algorithm in use)
    - unavailable: size, orientation, symbol/shape (shared boundaries)
    - free: color (could show weighted counts vs binary)
- scalability
  - 1K nodes, 1M edges



[NodeTrix: a Hybrid Visualization of Social Networks. Henry, Fekete, and McGuffin. IEEE TVCG (Proc. InfoVis) 13(6):1302-1309, 2007.]

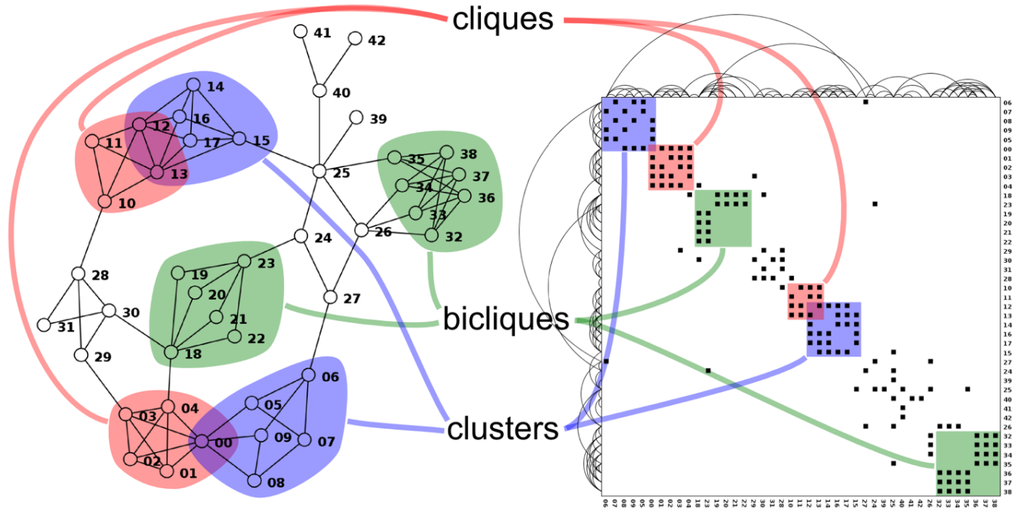


[Points of view: Networks. Gehlenborg and Wong. Nature Methods 9:115.]

Node-link vs. matrix comparison

- node-link diagram strengths
  - topology understanding, path tracing
  - intuitive, flexible, no training needed
- adjacency matrix strengths
  - focus on edges rather than nodes
  - layout straightforward (reordering needed)
  - predictability, scalability
  - some topology tasks trainable
- empirical study
  - node-link best for small networks
  - matrix best for large networks
    - if tasks don't involve path tracing!

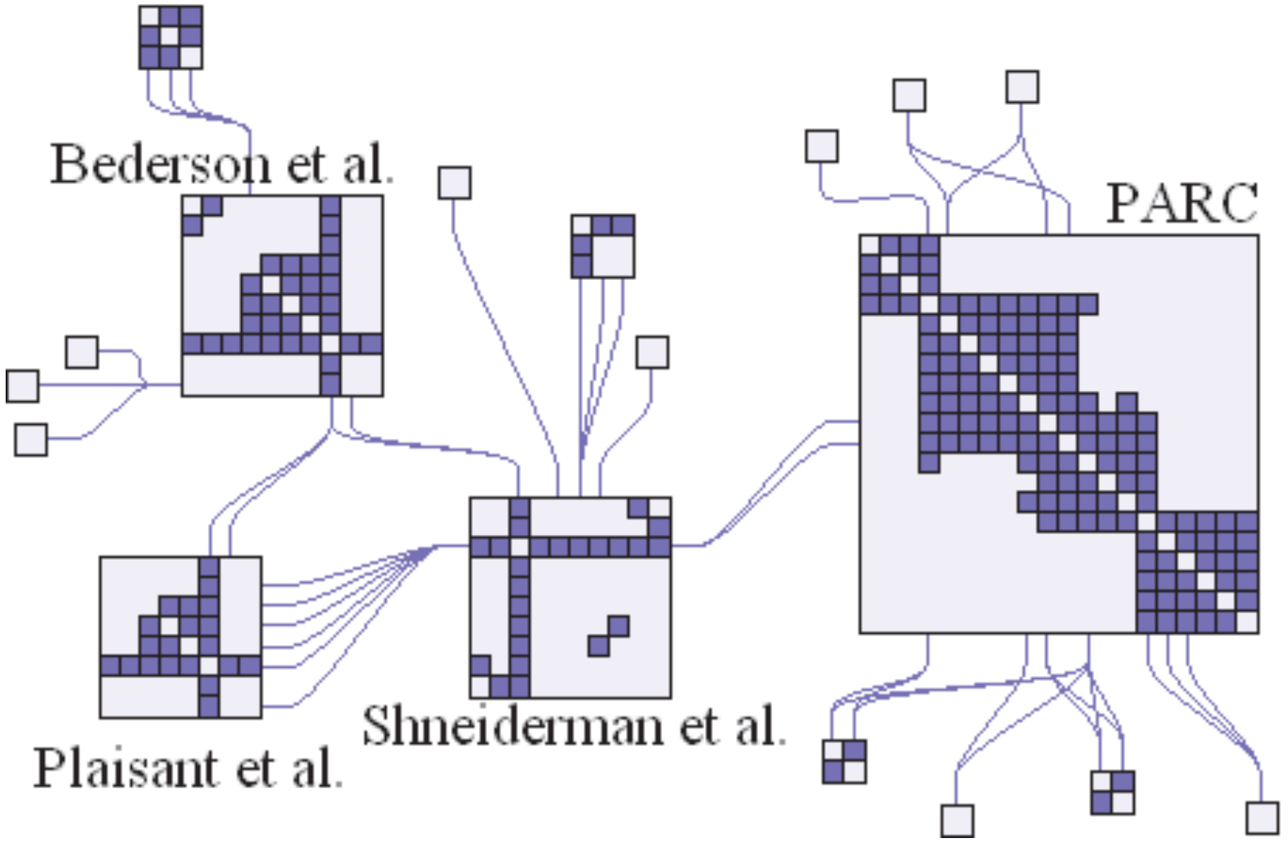
[On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. Ghoniem, Fekete, and Castagliola. Information Visualization 4:2 (2005), 114–135.]



http://www.michaelmcguffin.com/courses/vis/patternsInAdjacencyMatrix.png

Idiom: **NodeTrix**

- hybrid nodelink/matrix
- capture strengths of both



[NodeTrix: a Hybrid Visualization of Social Networks. Henry, Fekete, and McGuffin. IEEE TVCG (Proc. InfoVis) 13(6):1302-1309, 2007.]

Trees

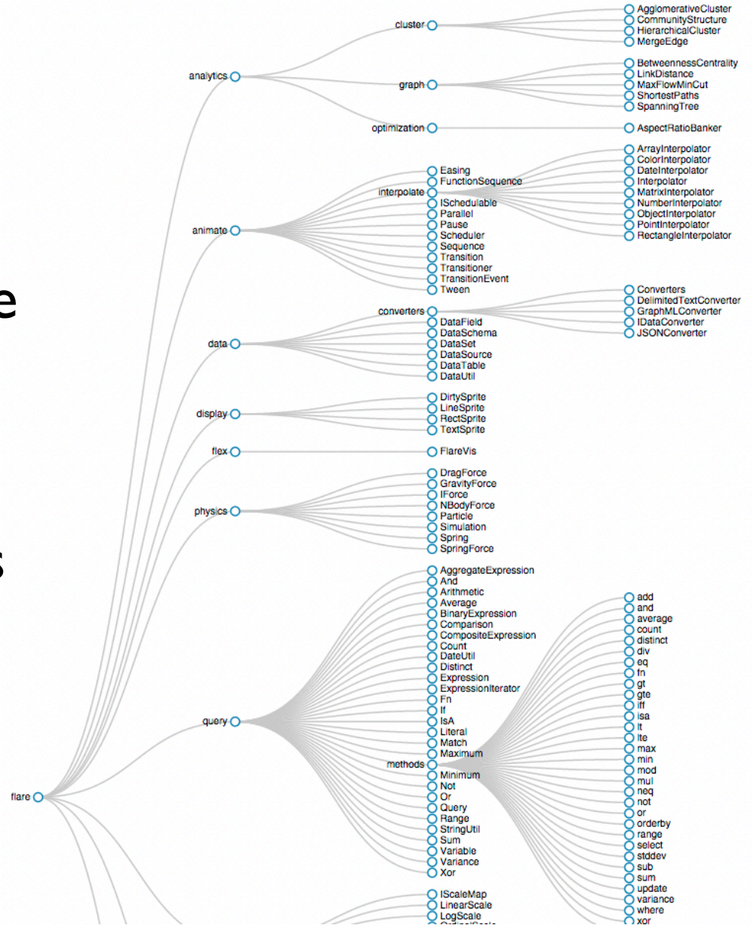
Node-link trees

- Reingold-Tilford
  - tidy drawings of trees
    - exploit parent/child structure
  - allocate space: compact, but without overlap
    - rectilinear and radial variants

[Tidier drawing of trees. Reingold and Tilford. IEEE Trans. Software Eng., SE-7(2):223–228, 1981.]

  - nice algorithm writeup

http://billmill.org/pymag-trees/

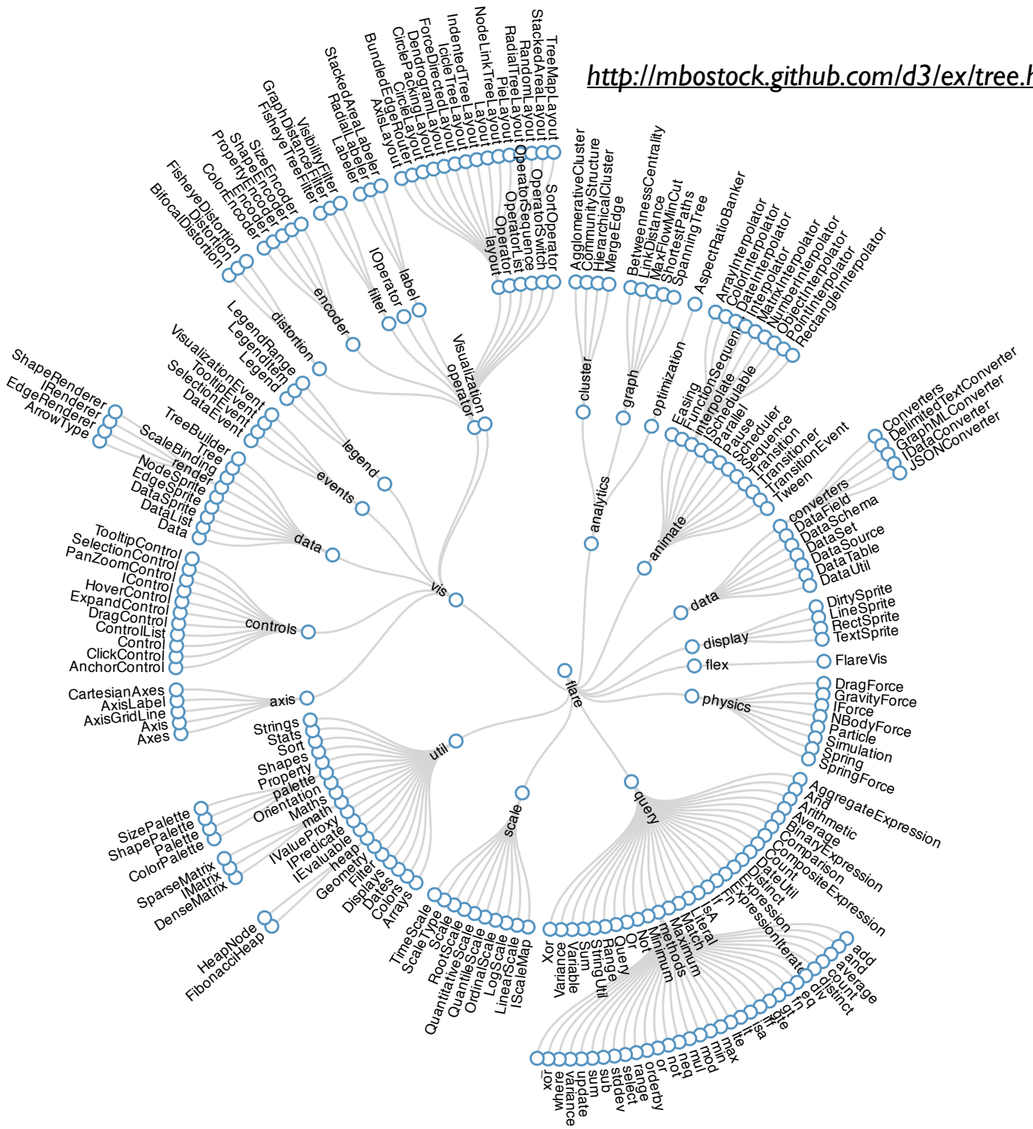


http://bl.ocks.org/mbostock/4339184

http://bl.ocks.org/mbostock/4063550

Idiom: **radial node-link tree**

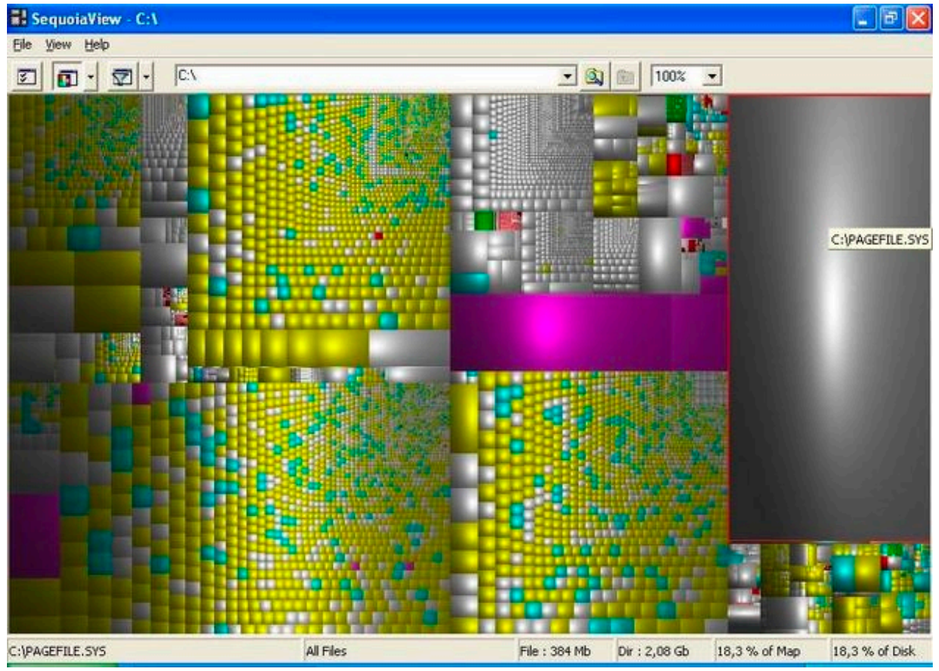
- data: tree
- marks
  - points: node marks (level 1)
  - links: segment connection marks (level 2)
- channels
  - encode: radial position/order, depth in tree
  - encode: node angular order, siblings of same parent
    - ordered by some attribute (alphabetical by label, or other)
  - unavailable: angular position, algorithm controls but no meaning
  - unavailable: segment length (1D), depends on node position
  - unavailable: segment orientation, depends on node position
  - free: node size (1D or 2D), link width (1D)
  - free: symbol/shape, node orientation (if asymmetric symbol)
- tasks
  - understanding topology, following paths
- scalability
  - nodes: 1K with labels, 10K without labels



http://mbostock.github.com/d3/ex/tree.html

Idiom: **treemap**

- data
  - tree
  - 1 quant attrib at leaf nodes
- marks: poly containment marks, 2D shared boundaries
  - showing hierarchical structure of network topology
- channels
  - encode: 2D size (area), quant attrib
  - unavailable: horizontal & vertical position/order, used by algorithm
  - unavailable: symbol/shape & orientation, due to shared boundaries
  - free: color
- tasks
  - find extreme values, find distribution of attribute values
    - at leaf nodes or cumulative value higher up in subtree
  - ex: disk space usage within filesystem
- scalability
  - 1M leaf nodes



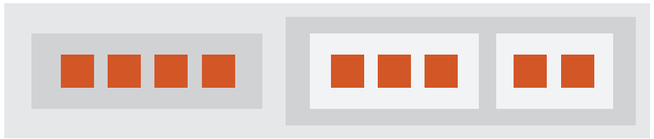
https://www.win.tue.nl/sequoiaview/

[Cushion Treemaps. van Wijk and van de Wetering. Proc. Symp. InfoVis 1999, 73-78.]

Enclosure

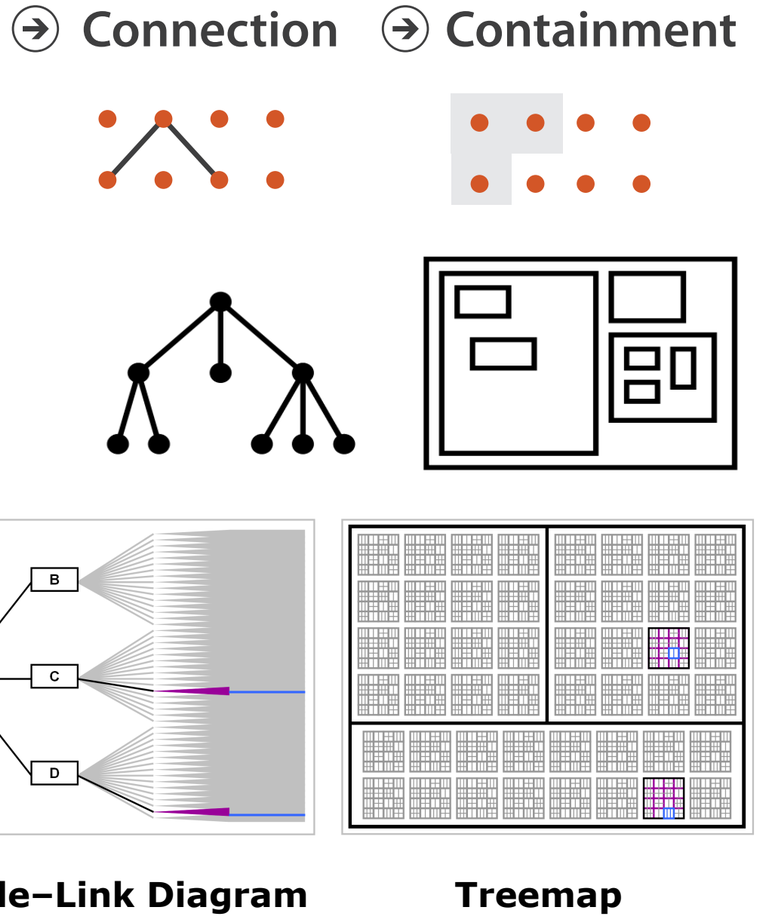
Containment Marks

✕ NETWORKS ✓ TREES



Link marks: Connection and containment

- marks as links (vs. nodes)
  - common case in network drawing
  - 1D case: connection
    - ex: all node-link diagrams
    - emphasizes topology, path tracing
    - networks and trees
  - 2D case: containment
    - ex: all treemap variants
    - emphasizes attribute values at leaves (size coding)
    - only trees



[Elastic Hierarchies: Combining Treemaps and Node-Link Diagrams. Dong, McGuffin, and Chignell. Proc. InfoVis 2005, p. 57-64.]

Idiom: implicit tree layouts (sunburst, icicle plot)

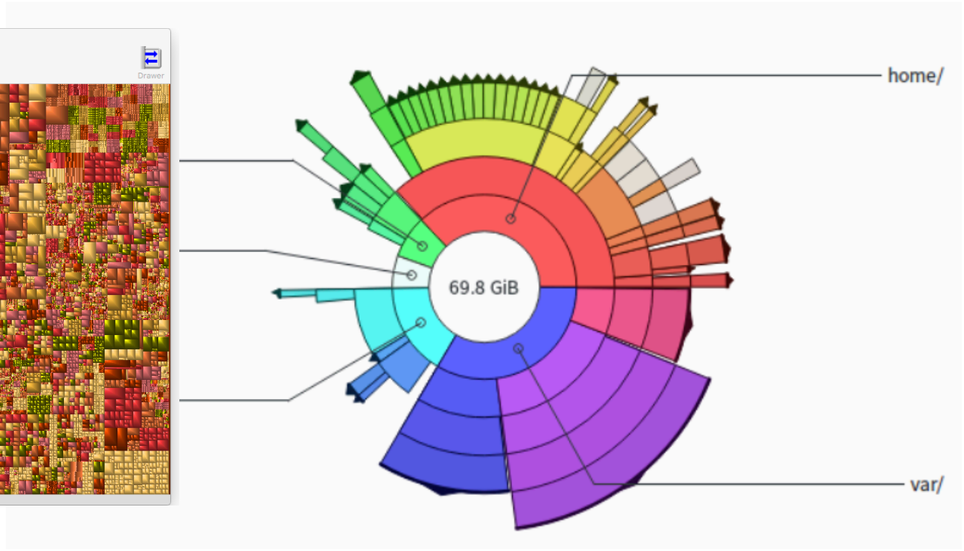
- alternative to connection and containment: position/order
  - show parent-child relationships only through order, show depth with position

Treemap

containment

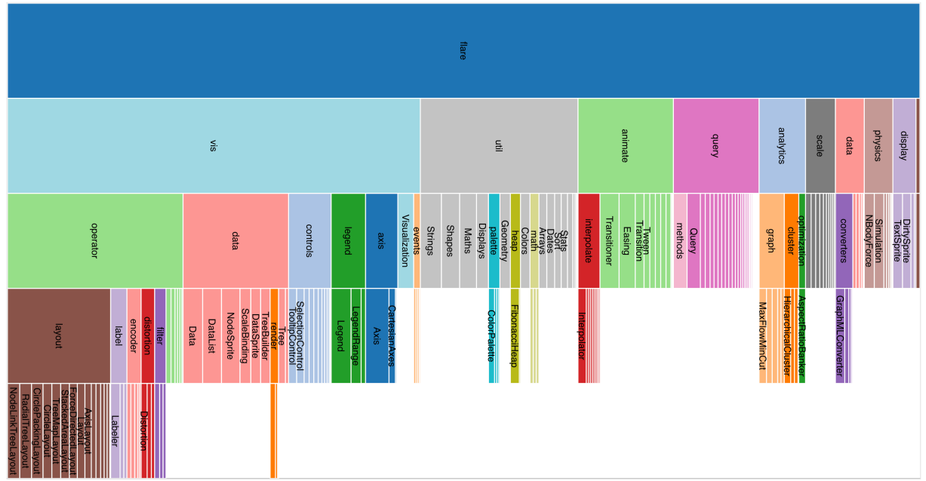
Sunburst

position (radial)



Icicle Plot

position (rectilinear)



Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position/order
  - show parent-child relationships only through order, show depth with position

Treemap

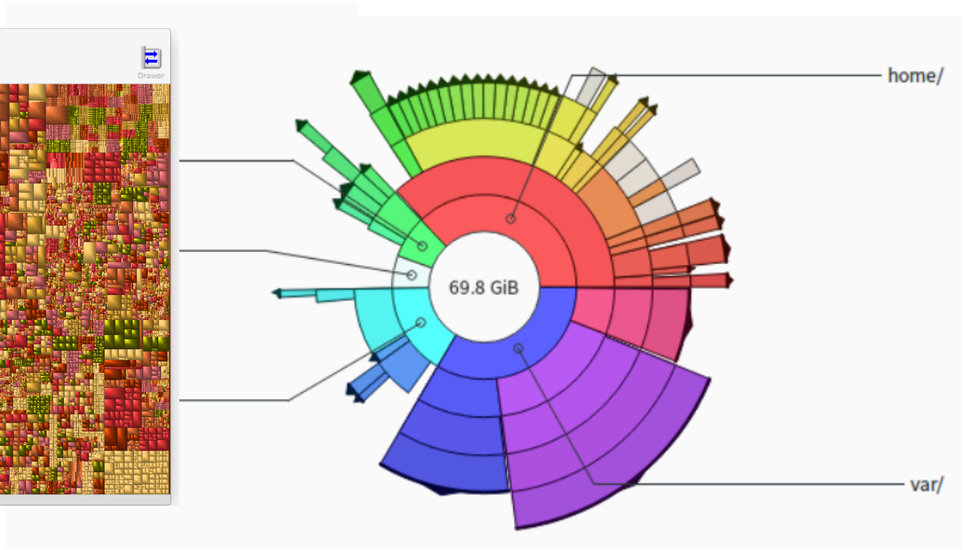
containment

only leaves visible

Sunburst

position (radial)

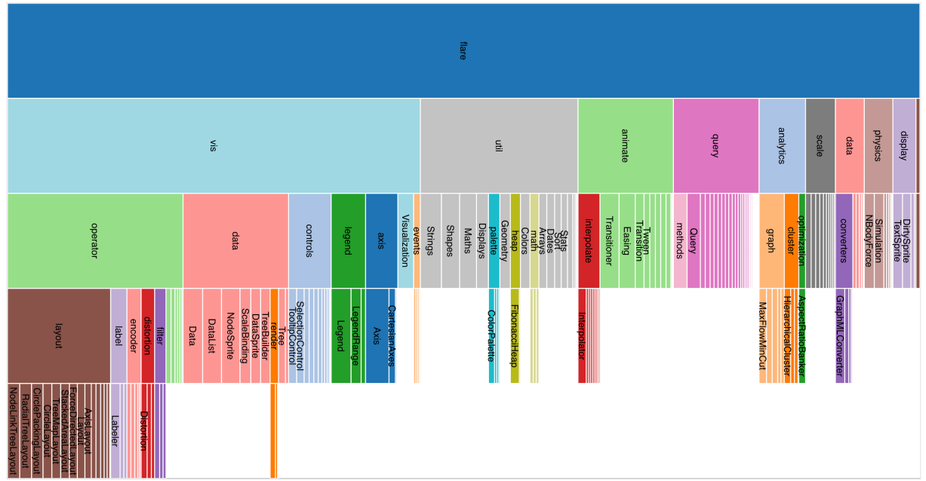
inner nodes & leaves visible



Icicle Plot

position (rectilinear)

inner nodes & leaves visible



Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position/order
  - show parent-child relationships only through order, show depth with position

Treemap

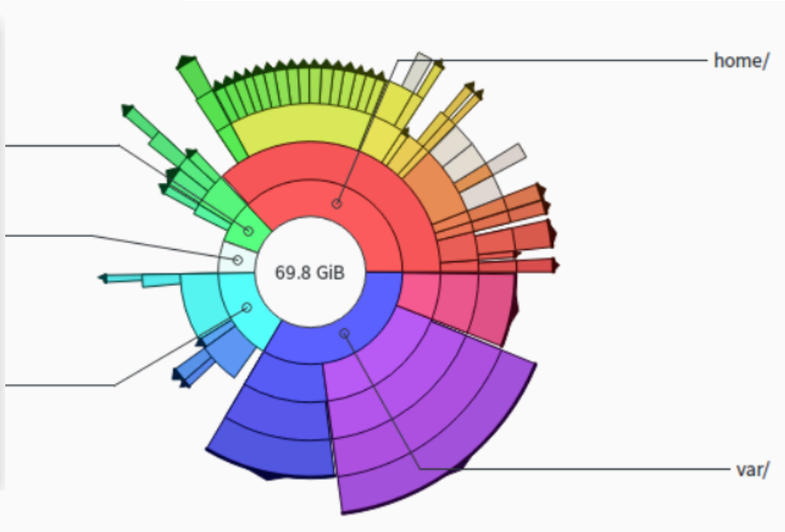
containment

only leaves visible

Sunburst

position (radial)

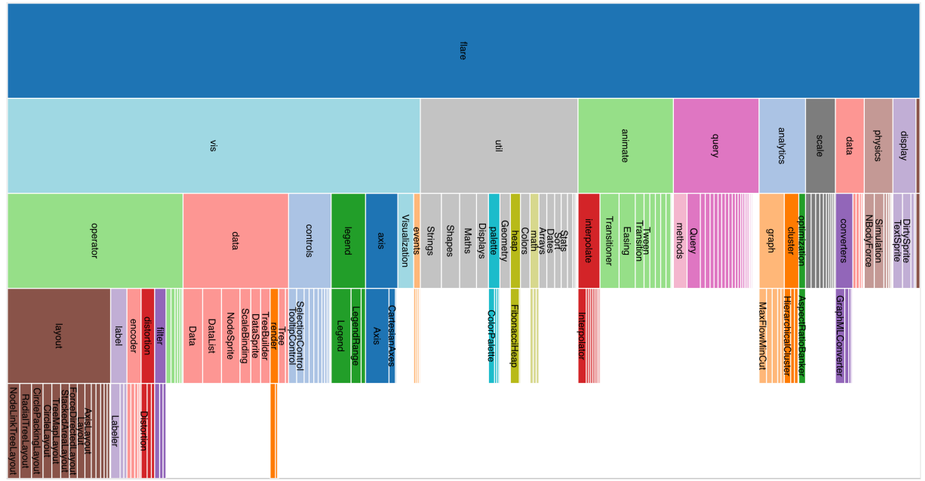
inner nodes & leaves visible



Icicle Plot

position (rectilinear)

inner nodes & leaves visible

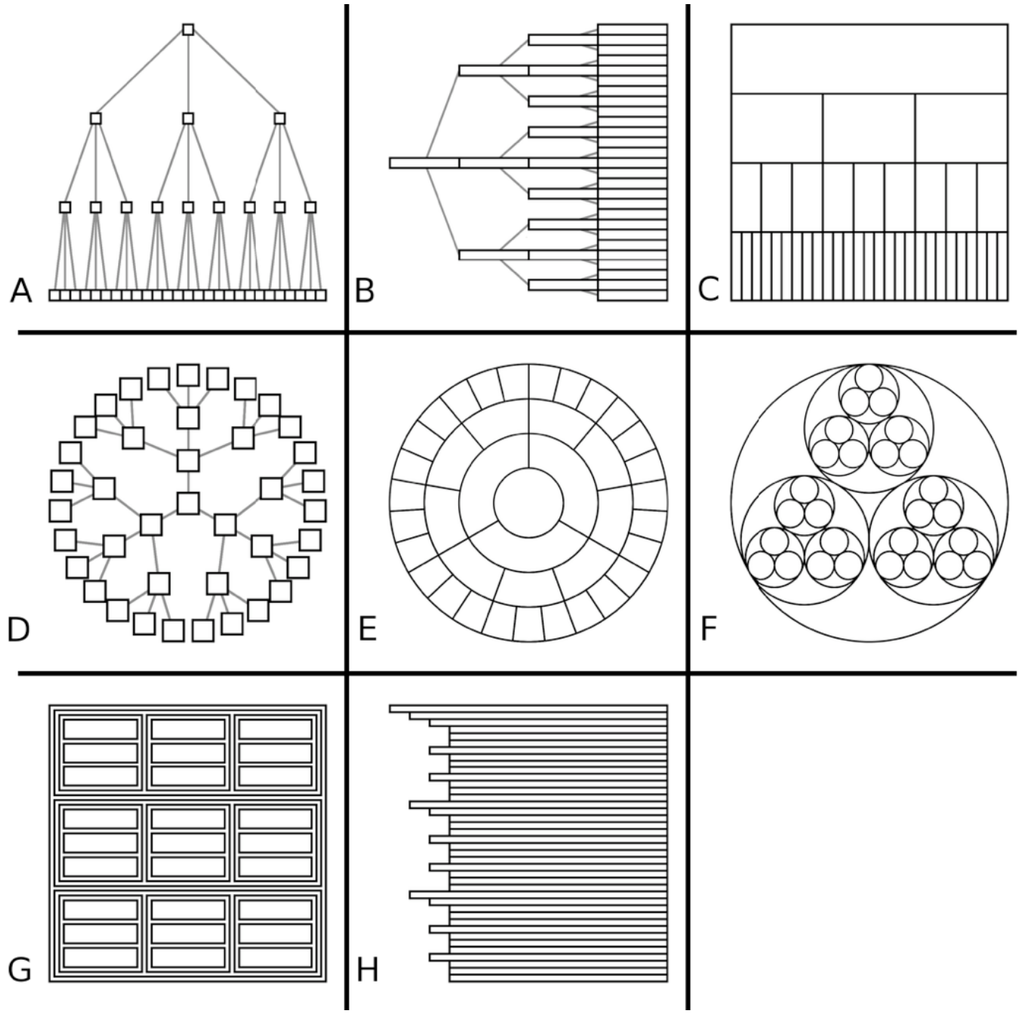


Implicit

Spatial Position

✕ NETWORKS ✓ TREES

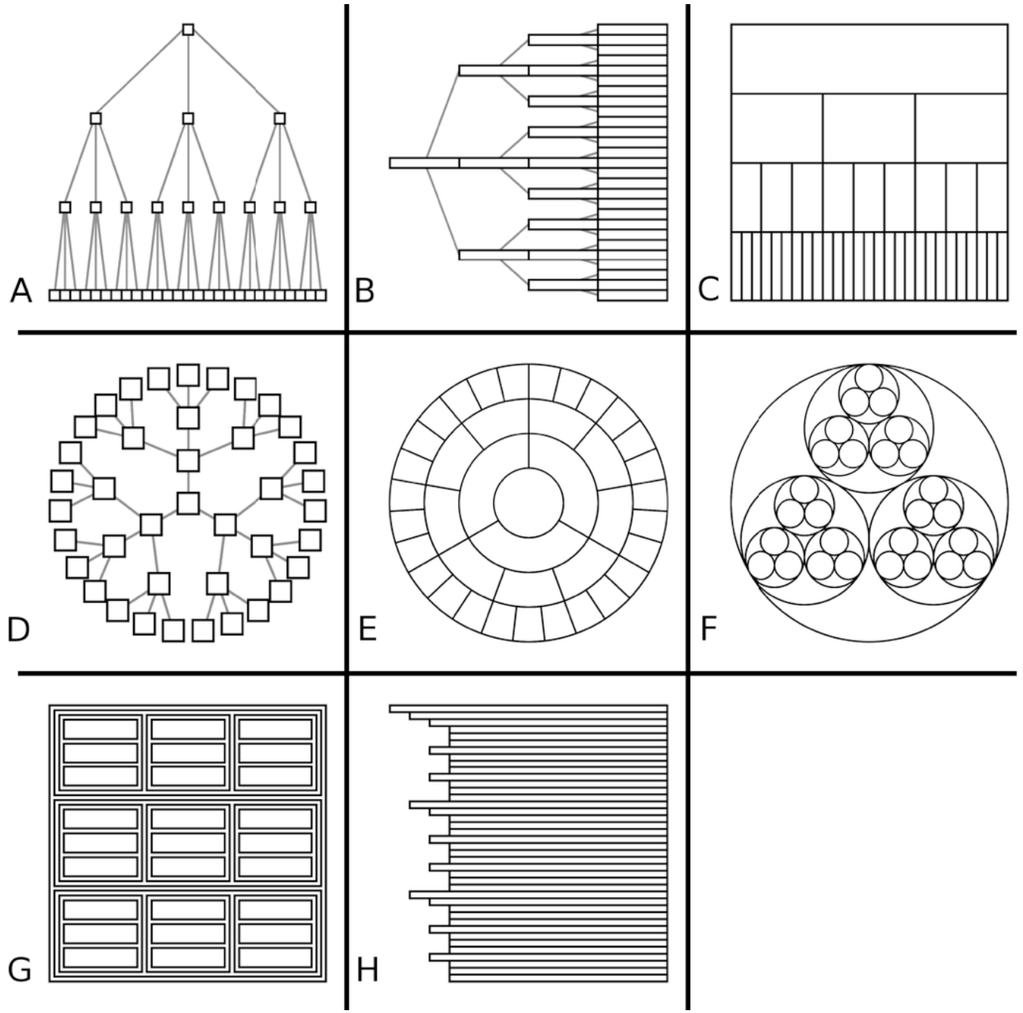
Comparison: tree drawing idioms



[Quantifying the Space-Efficiency of 2D Graphical Representations of Trees. McGuffin and Robert. Information Visualization 9:2 (2010), 115–140.]

Comparison: tree drawing idioms

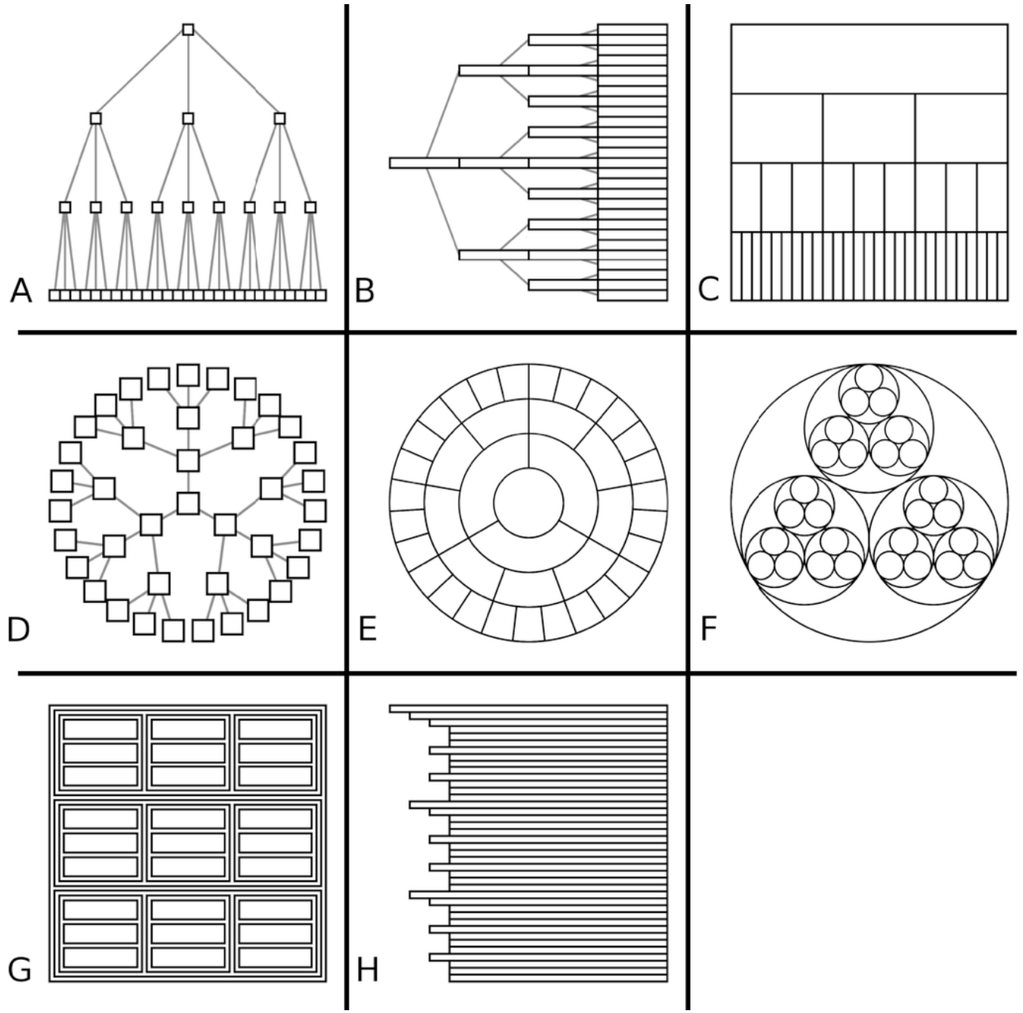
- data shown
  - link relationships
  - tree depth
  - sibling order



[Quantifying the Space-Efficiency of 2D Graphical Representations of Trees. McGuffin and Robert. Information Visualization 9:2 (2010), 115–140.]

Comparison: tree drawing idioms

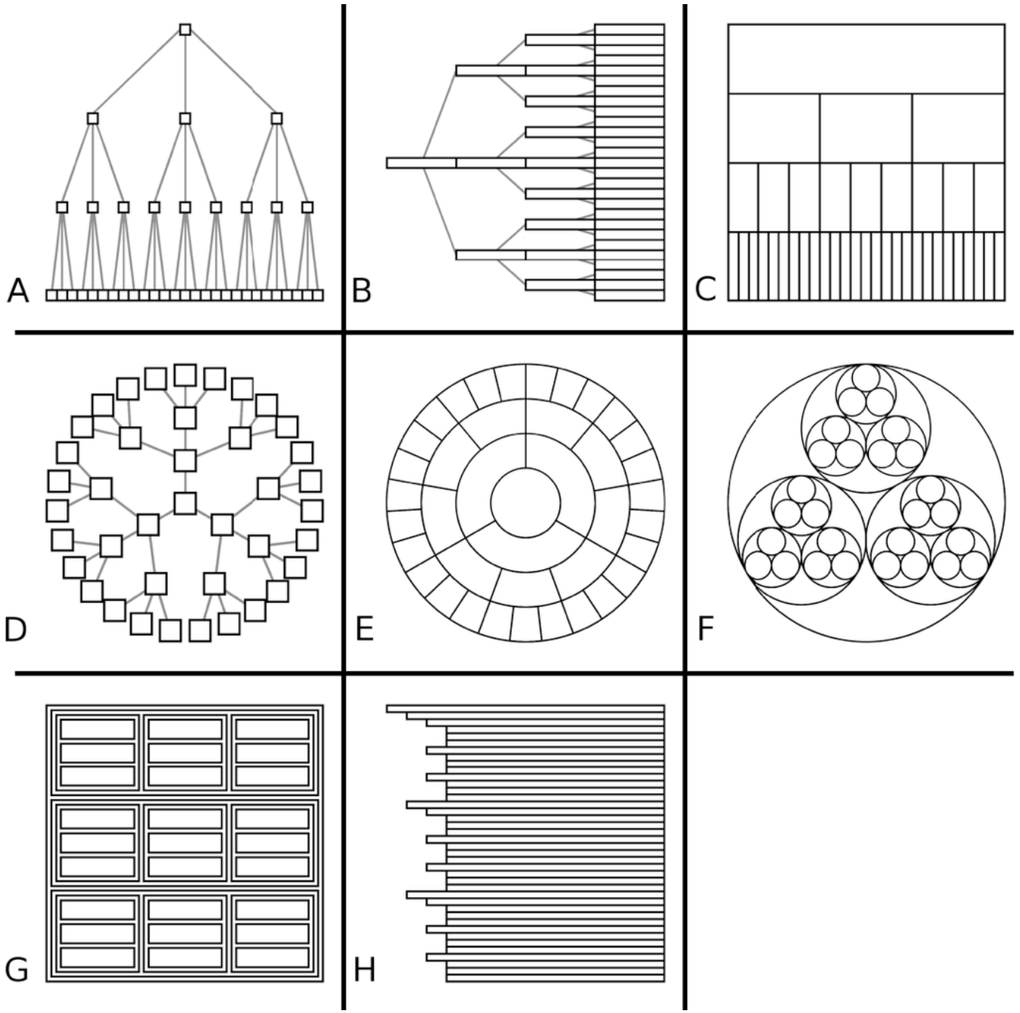
- data shown
  - link relationships
  - tree depth
  - sibling order
- design choices
  - connection vs containment vs implicit for links
  - rectilinear vs radial layout
  - spatial position/order channels



[Quantifying the Space-Efficiency of 2D Graphical Representations of Trees. McGuffin and Robert. Information Visualization 9:2 (2010), 115–140.]

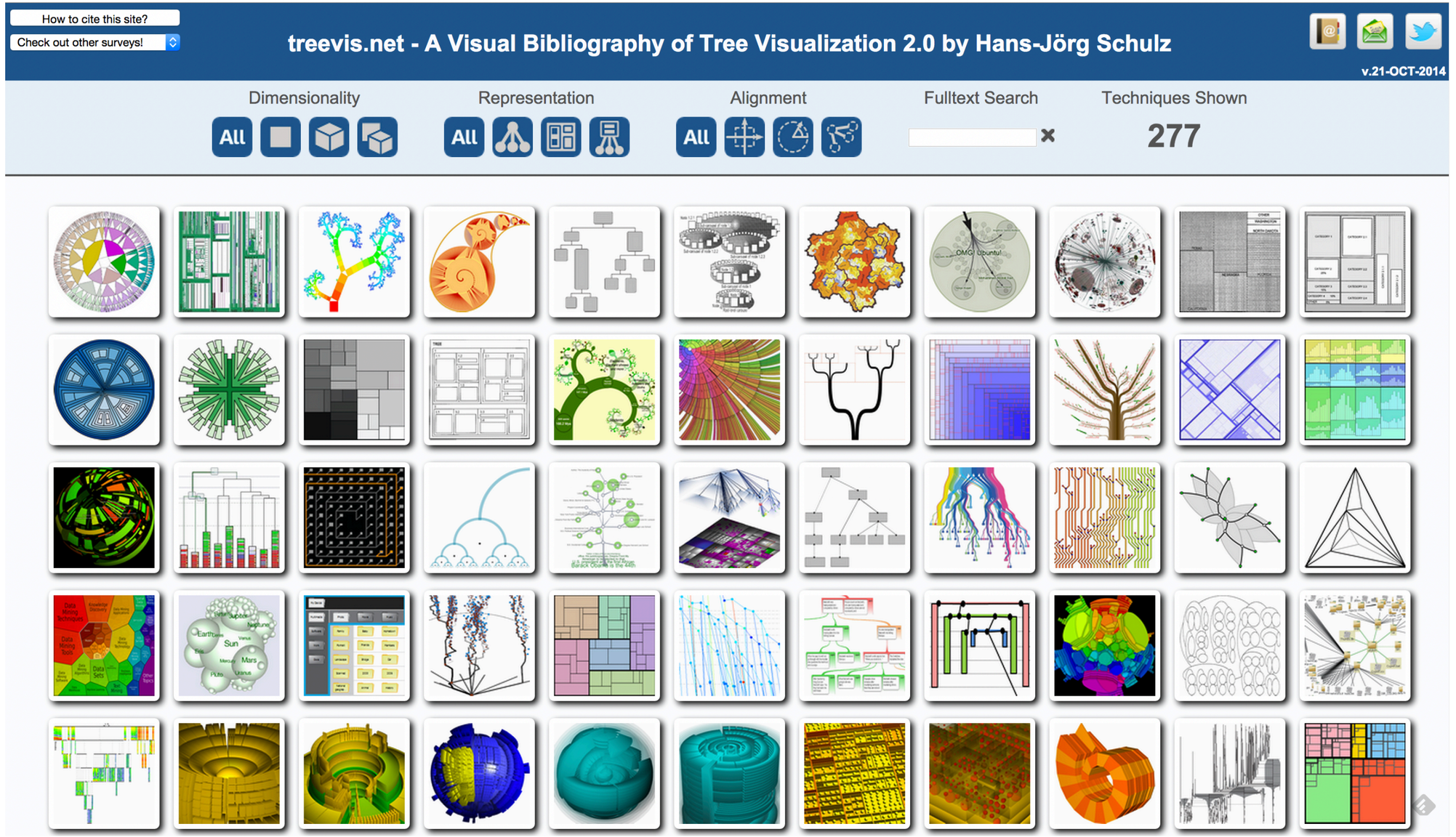
Comparison: tree drawing idioms

- data shown
  - link relationships
  - tree depth
  - sibling order
- design choices
  - connection vs containment vs implicit for links
  - rectilinear vs radial layout
  - spatial position/order channels
- considerations
  - redundant? arbitrary?
  - information density?
    - avoid wasting space
    - consider where to fit labels!



[Quantifying the Space-Efficiency of 2D Graphical Representations of Trees. McGuffin and Robert. Information Visualization 9:2 (2010), 115–140.]

treevis.net: Many, many options!



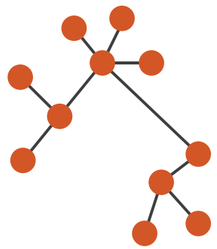
https://treevis.net/

Arrange networks and trees

Node–Link Diagrams

Connection Marks

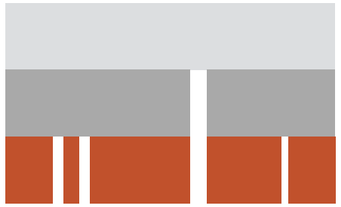
✔ NETWORKS ✔ TREES



Implicit

Spatial Position

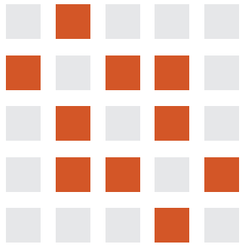
✘ NETWORKS ✔ TREES



Adjacency Matrix

Derived Table

✔ NETWORKS ✔ TREES



Enclosure

Containment Marks

✘ NETWORKS ✔ TREES