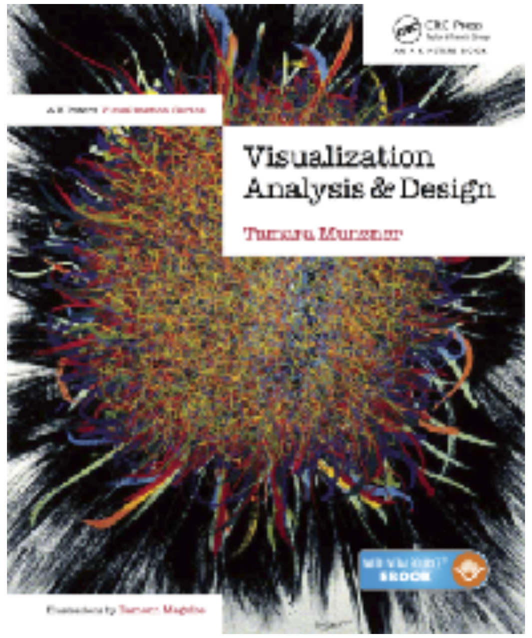
Visualization Analysis & Design

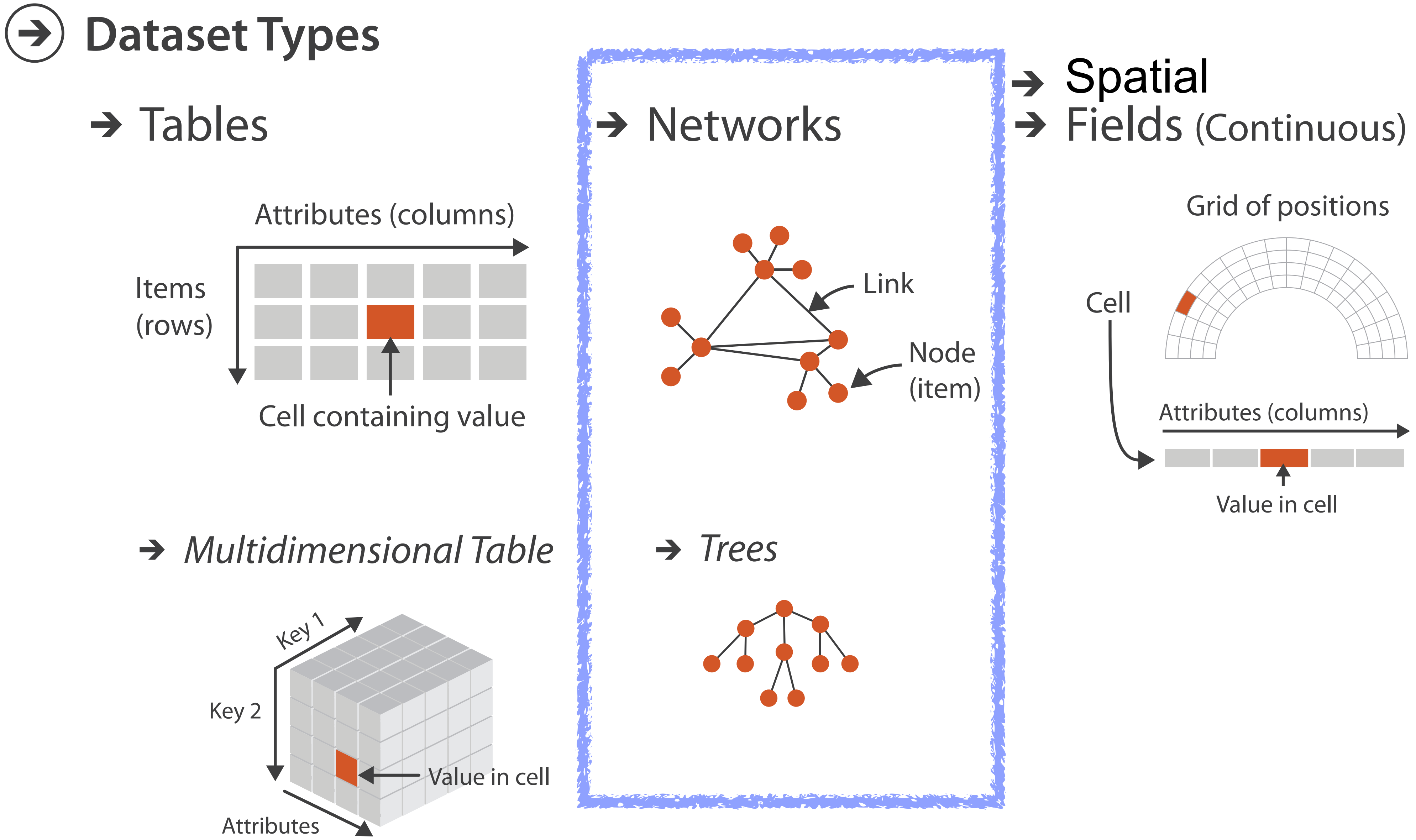
Network Data (Ch 9)



Tamara Munzner
Department of Computer Science
University of British Columbia
@tamaramunzner

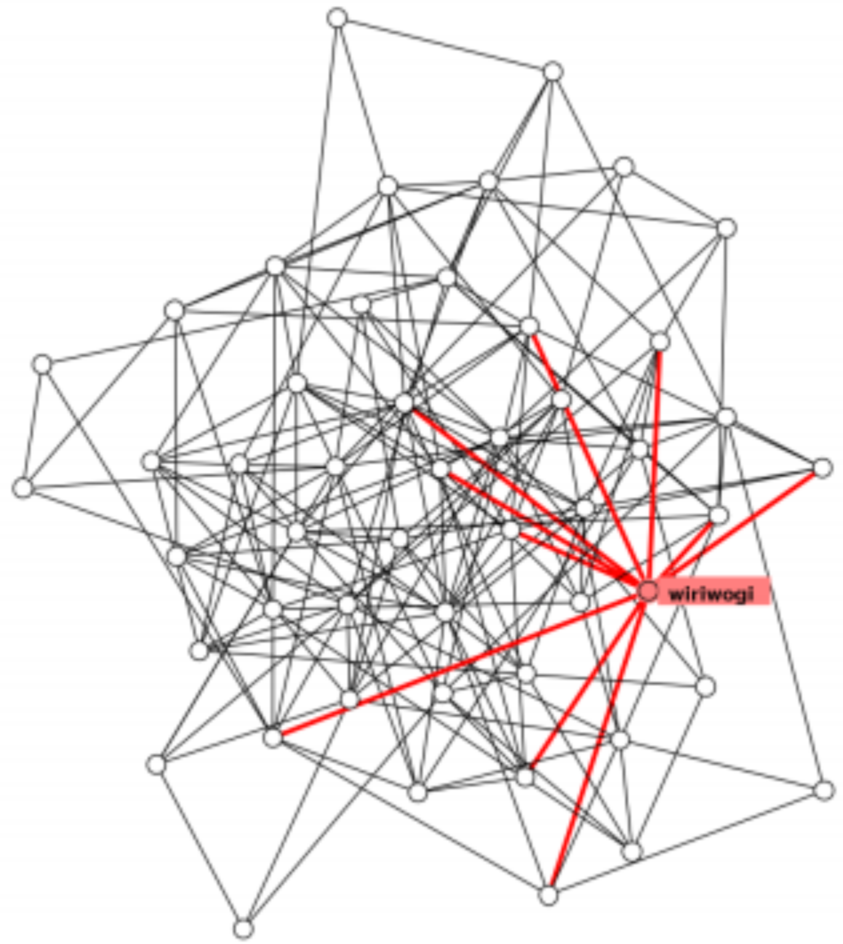
Network data

- 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



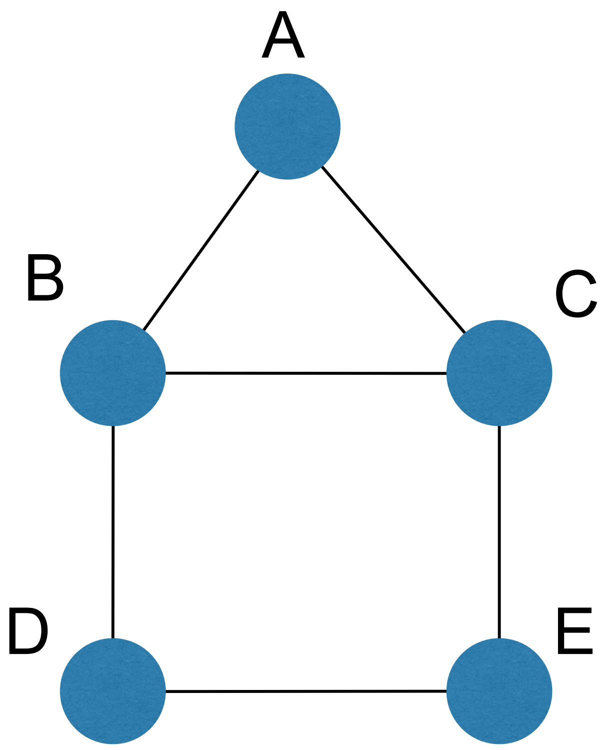
Network tasks: topology-based and attribute-based

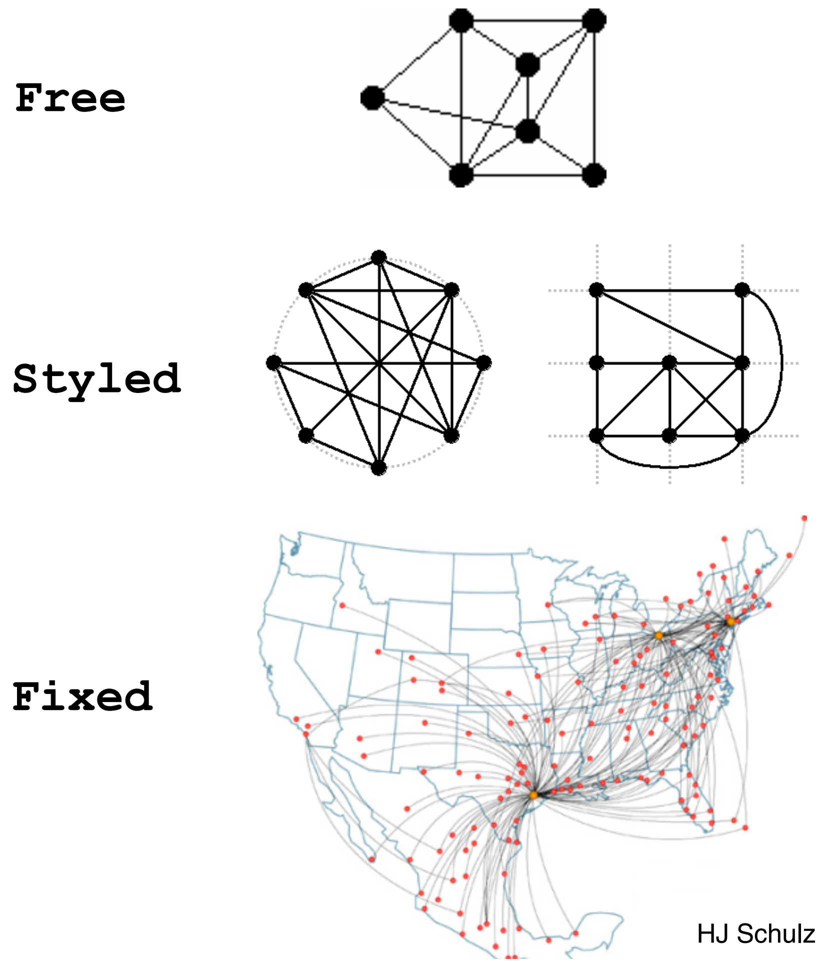
- topology based tasks
  - find paths
  - find (topological) neighbors
  - compare centrality/importance measures
  - identify clusters / communities
- attribute based tasks (similar to table data)
  - find distributions, ...
- combination tasks, incorporating both
  - example: find friends-of-friends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



Node-link diagrams

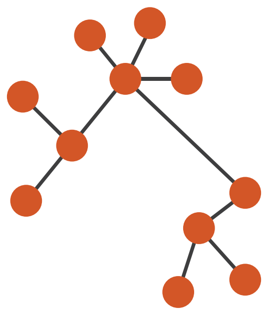
- nodes: point marks
- links: line marks
  - straight lines or arcs
  - connections between nodes
- intuitive & familiar
  - most common
  - many, many variants





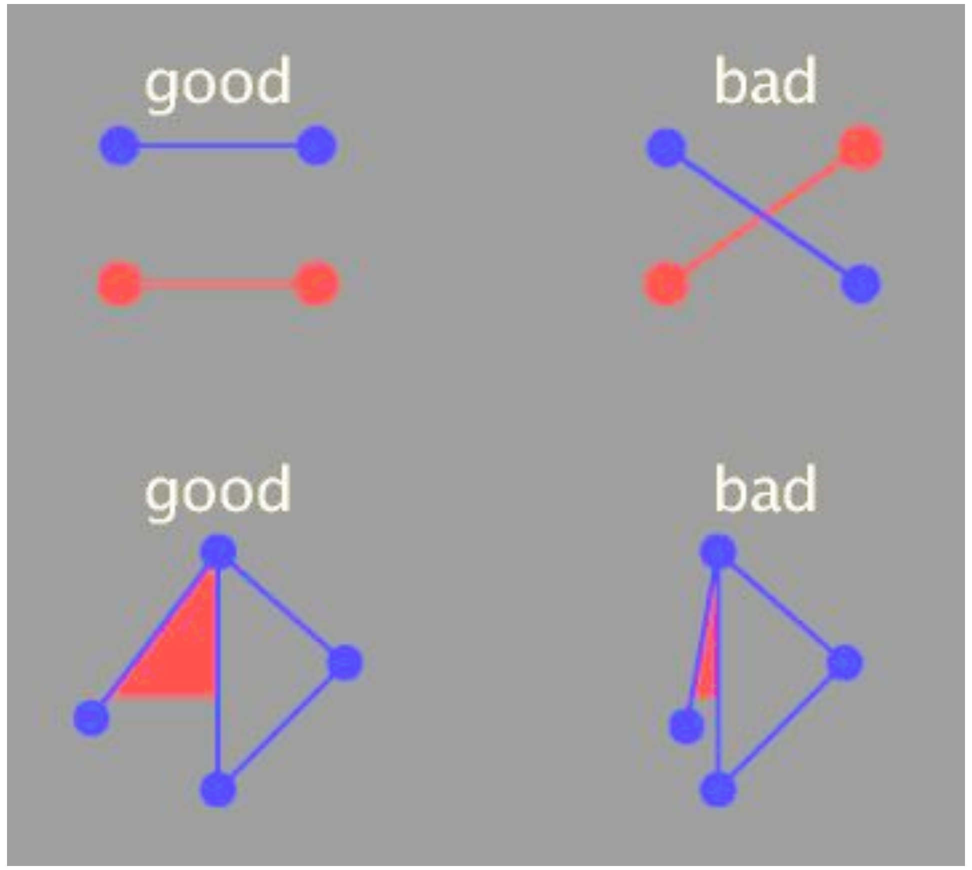
HJ Schulz 2006

➔ **Node–Link Diagrams**
Connection Marks
✔ NETWORKS ✔ TREES



Criteria for good node-link layouts

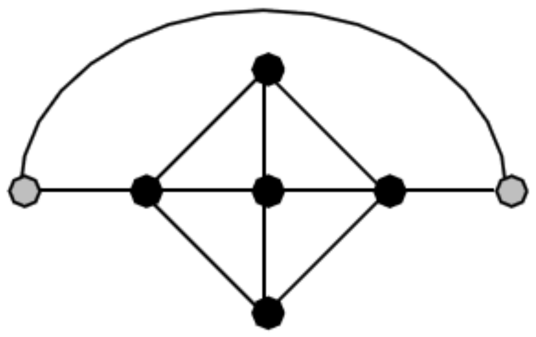
- minimize
  - edge crossings, node overlaps
  - distances between topological neighbor nodes
  - total drawing area
  - edge bends
- maximize
  - angular distance between different edges
  - aspect ratio disparities
- emphasize symmetry
  - similar graph structures should look similar in layout



Criteria conflict

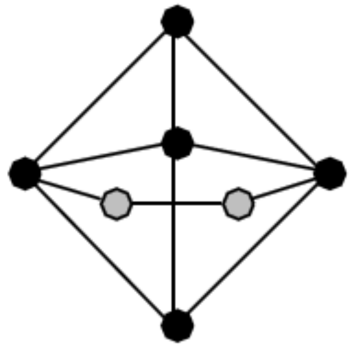
- most criteria NP-hard individually
- many criteria directly conflict with each other

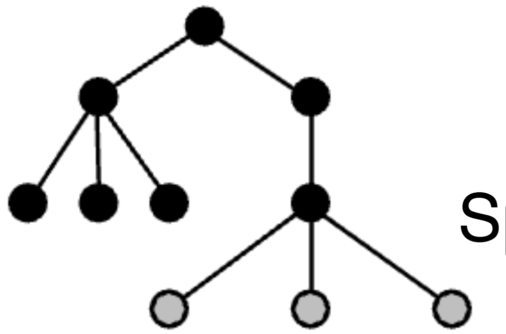
Minimum number of edge crossings



vs.

Uniform edge length

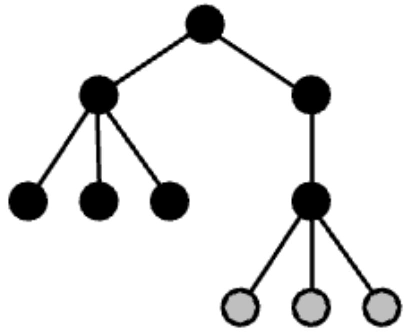




Space utilization

vs.

Symmetry



Schulz 2004

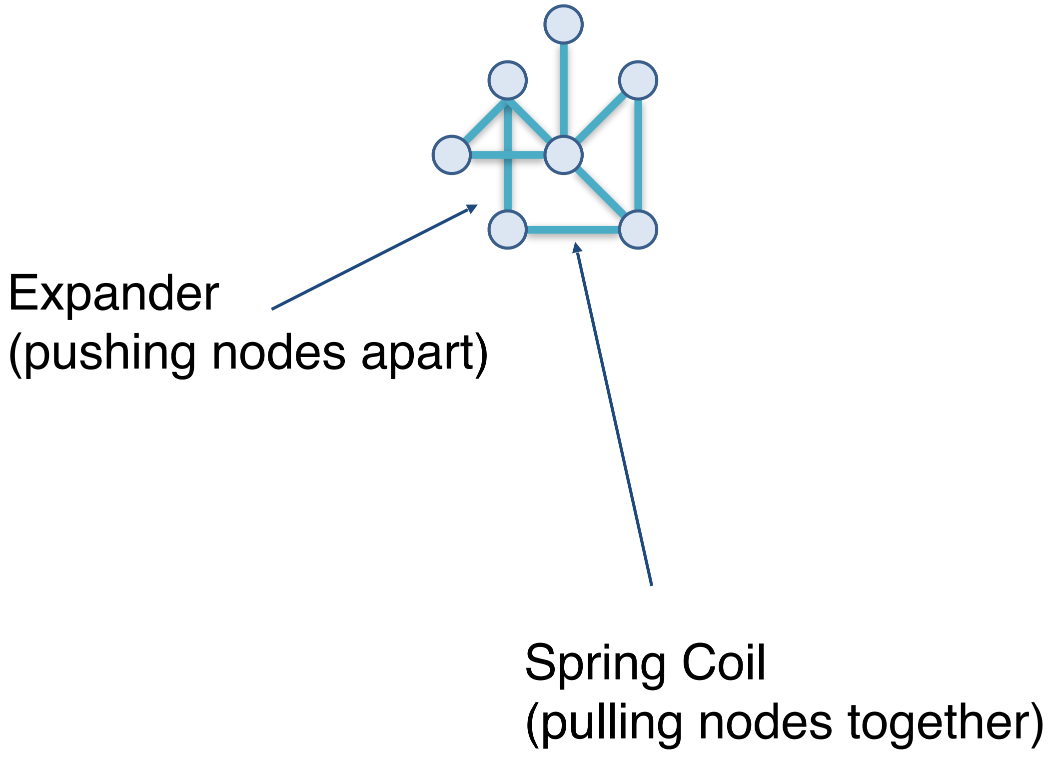
Optimization-based layouts

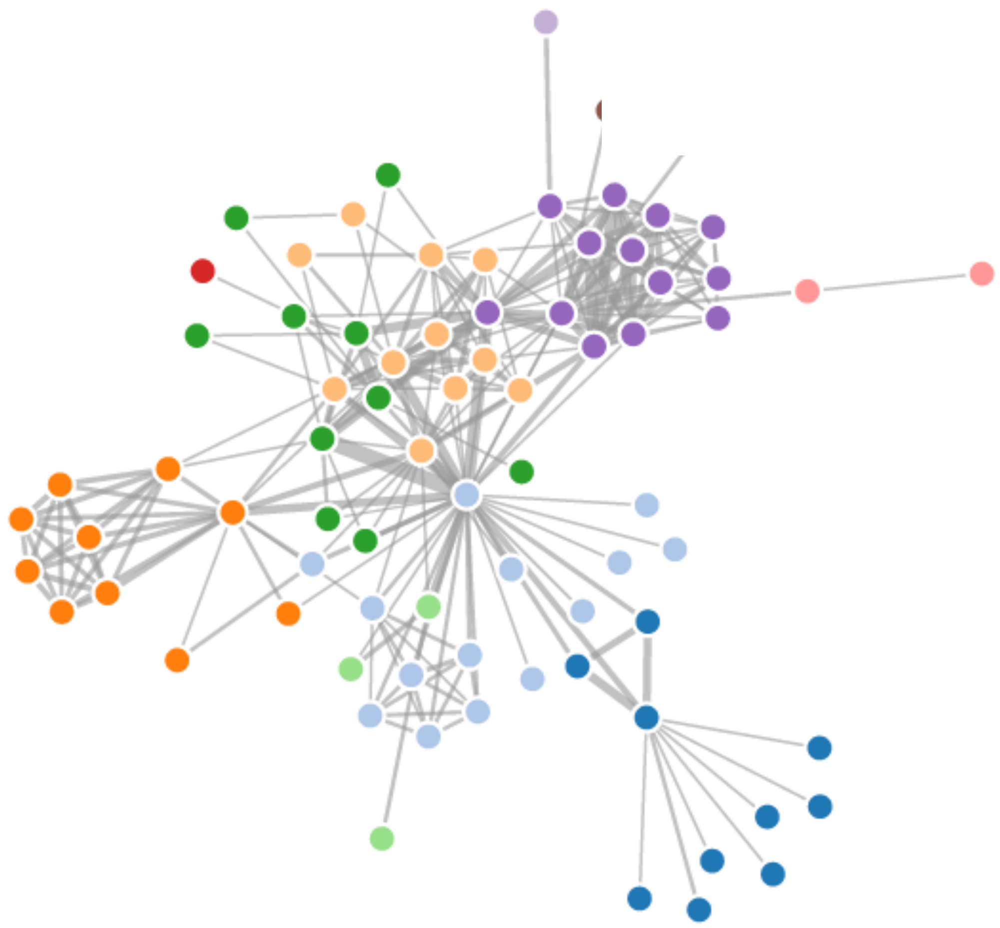
- formulate layout problem as optimization problem
- convert criteria into weighted cost function
  - F(layout) = a*[crossing counts] + b*[drawing space used]+...
- use known optimization techniques to find layout at minimal cost
  - energy-based physics models
  - force-directed placement
  - spring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

Force-directed placement

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

- 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

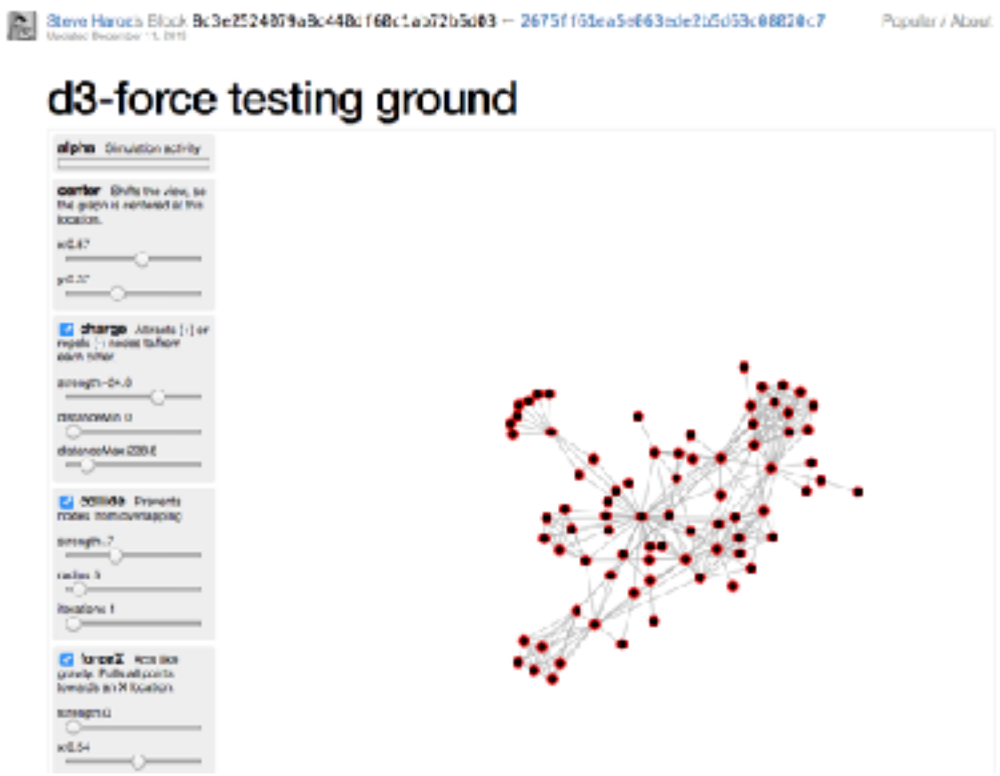




http://mbostock.github.com/d3/ex/force.html

Force-directed placement properties

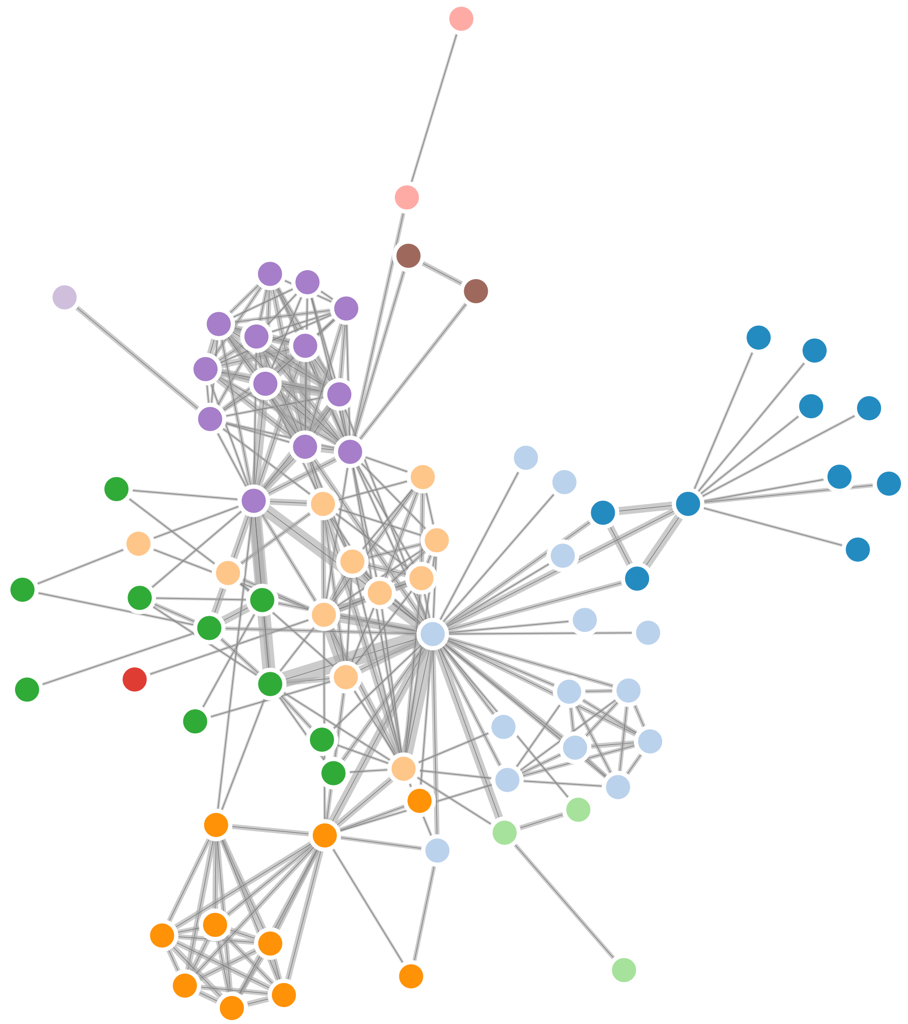
- strengths
  - reasonable layout for small, sparse graphs
  - clusters typically visible
  - edge length uniformity
- weaknesses
  - nondeterministic
  - computationally expensive: O(n^3) for n nodes
    - each step is n^2, takes ~n cycles to reach equilibrium
  - naive FD doesn't scale well beyond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



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

Idiom: **force-directed placement**

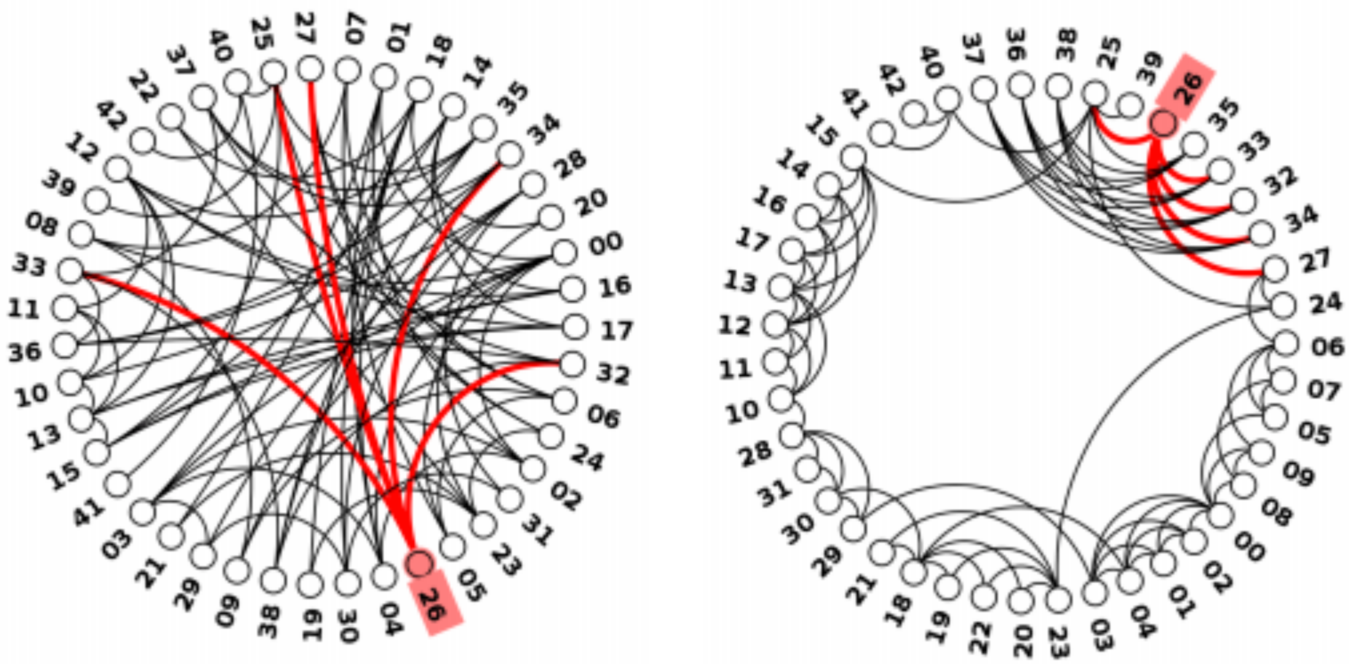
- visual encoding
  - link connection marks, node point marks
- considerations
  - spatial position: no meaning directly encoded
    - left free to minimize crossings
  - proximity semantics?
    - sometimes meaningful
    - sometimes arbitrary, artifact of layout algorithm
    - tension with length
      - long edges more visually salient than short
- tasks
  - explore topology; locate paths, clusters
- scalability
  - node/edge density E < 4N

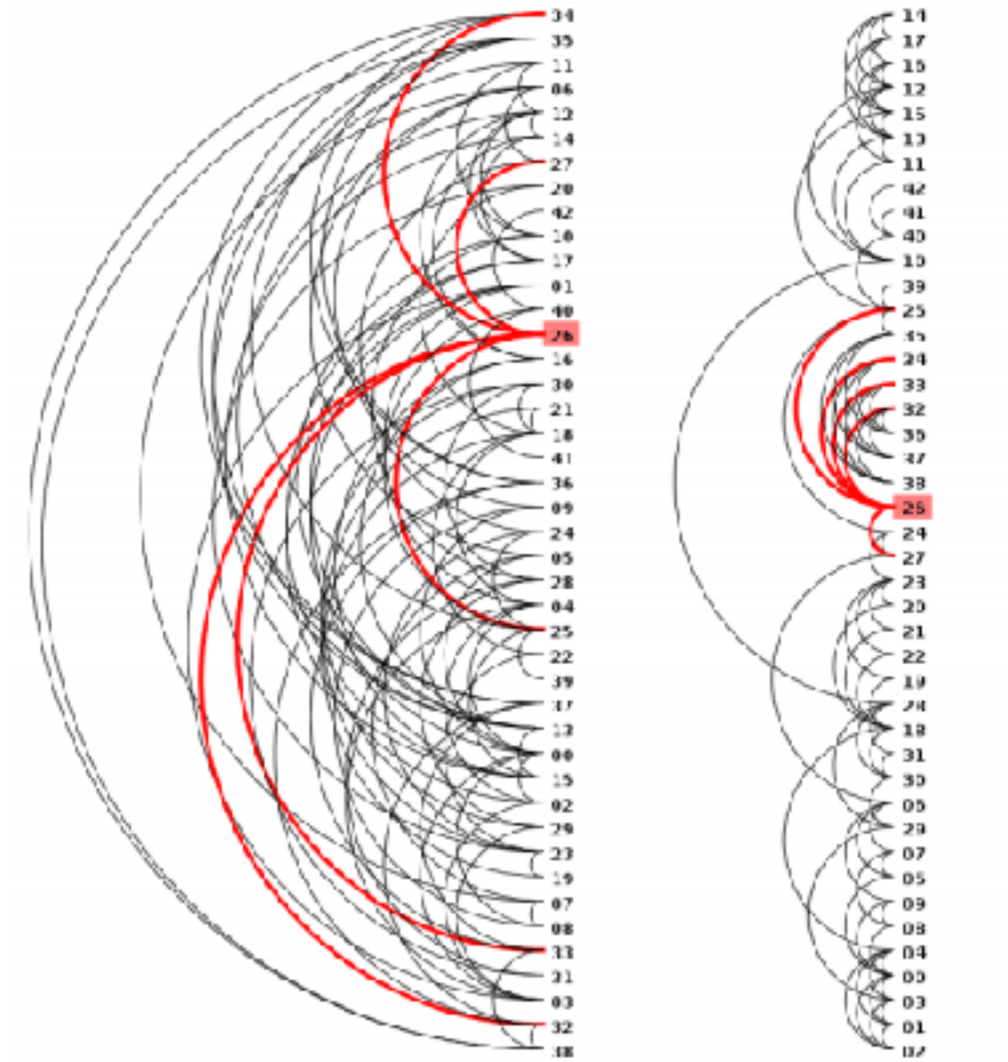


http://mbostock.github.com/d3/ex/force.html

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

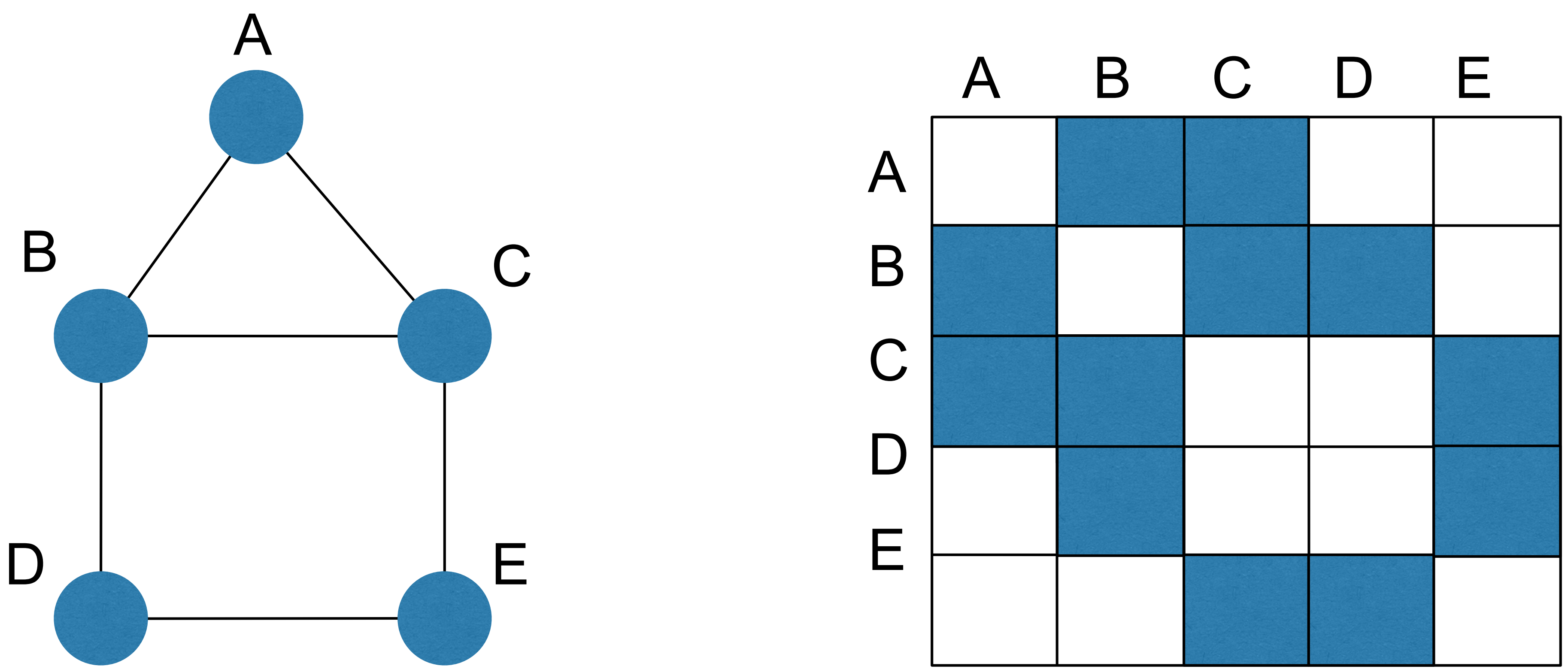
- restricted node-link layouts: lay out nodes around circle or along line
- data
  - original: network
  - derived: node ordering attribute (global computation)
- considerations: node ordering crucial to avoid excessive clutter from edge crossings
  - examples: before & after barycentric ordering

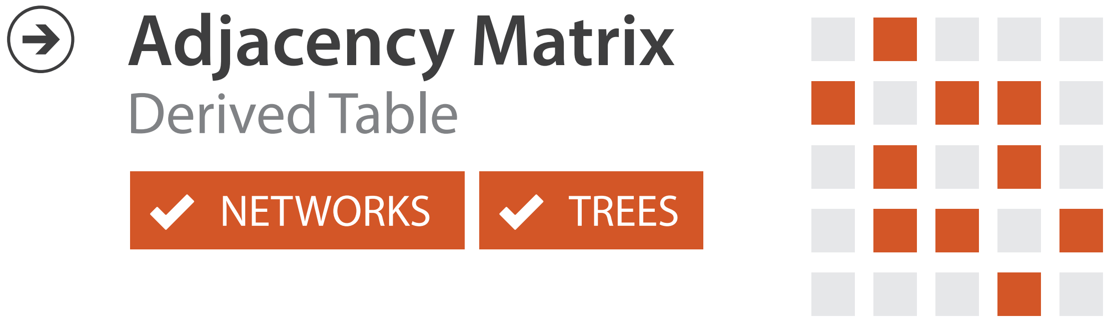




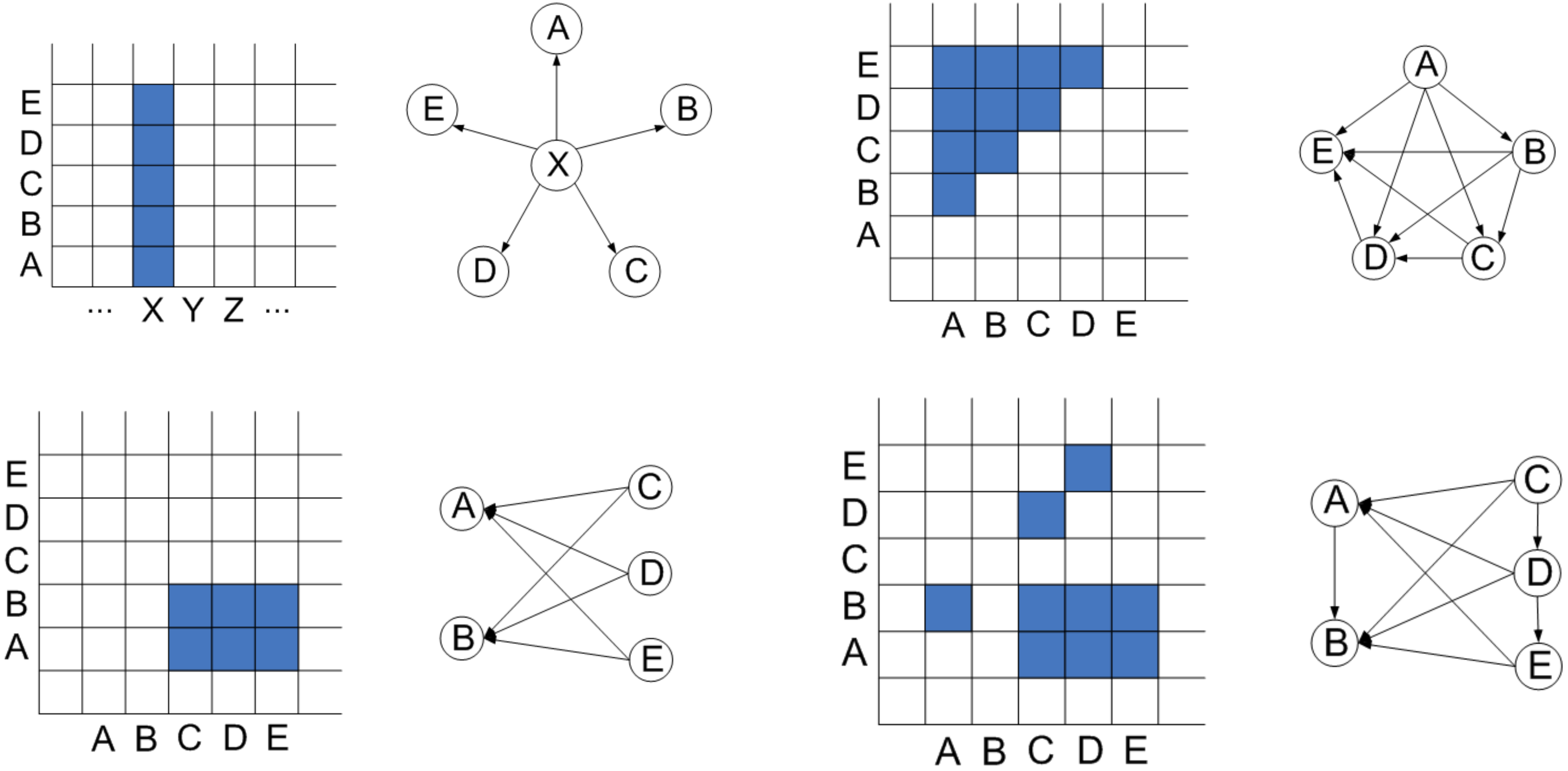
Adjacency matrix representations

- derive adjacency matrix from network



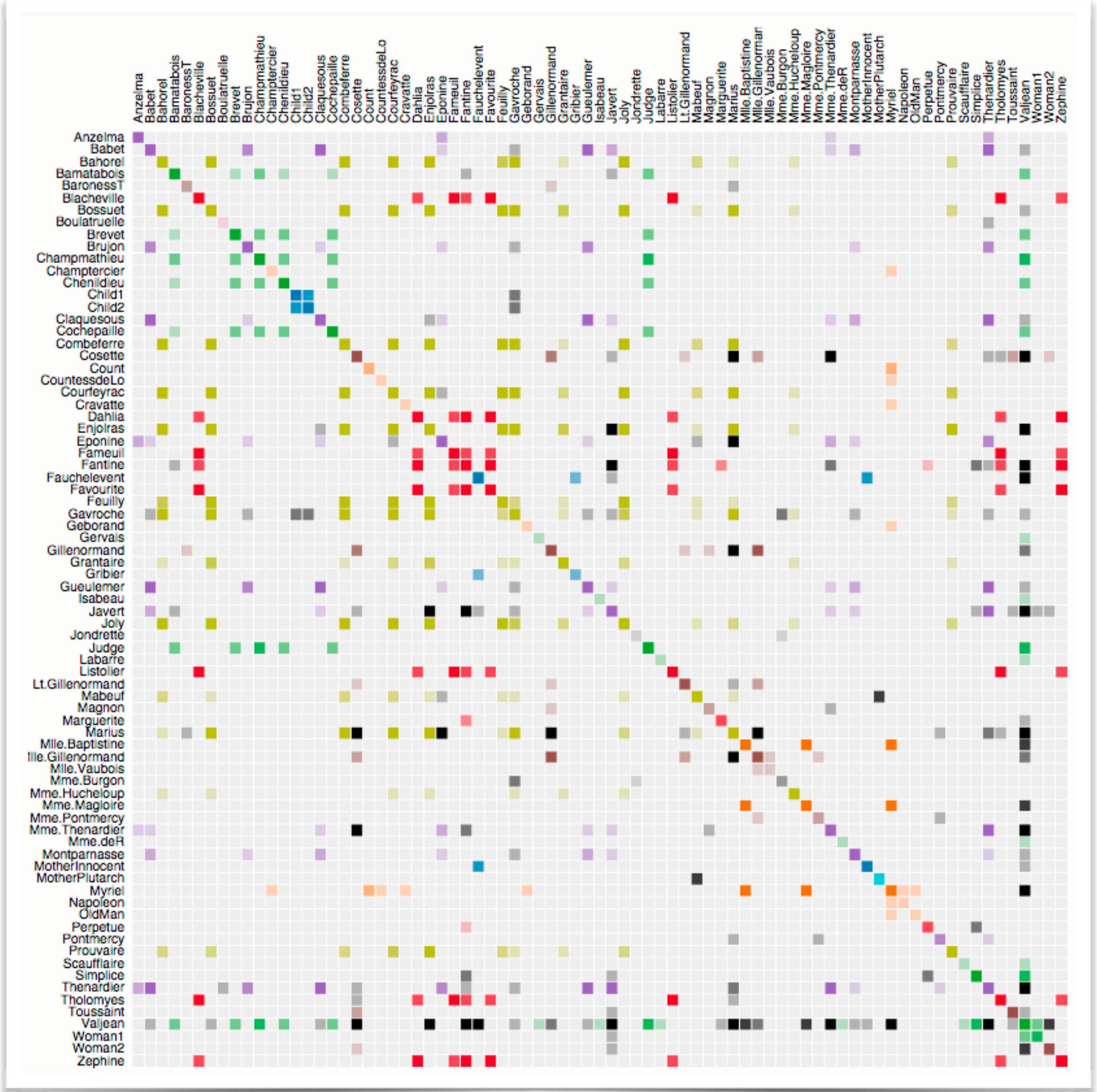


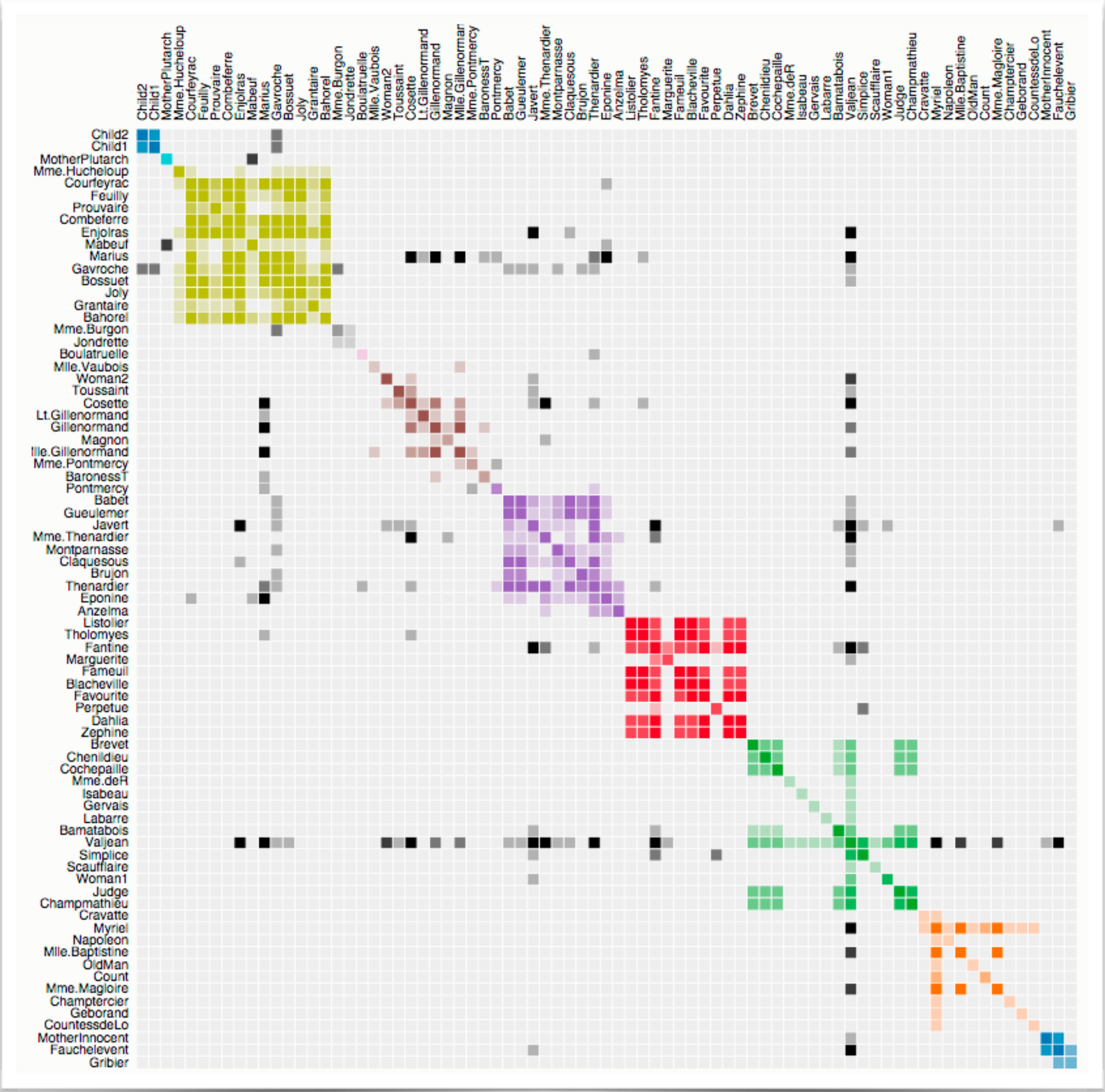
Adjacency matrix examples



HJ Schulz 2007

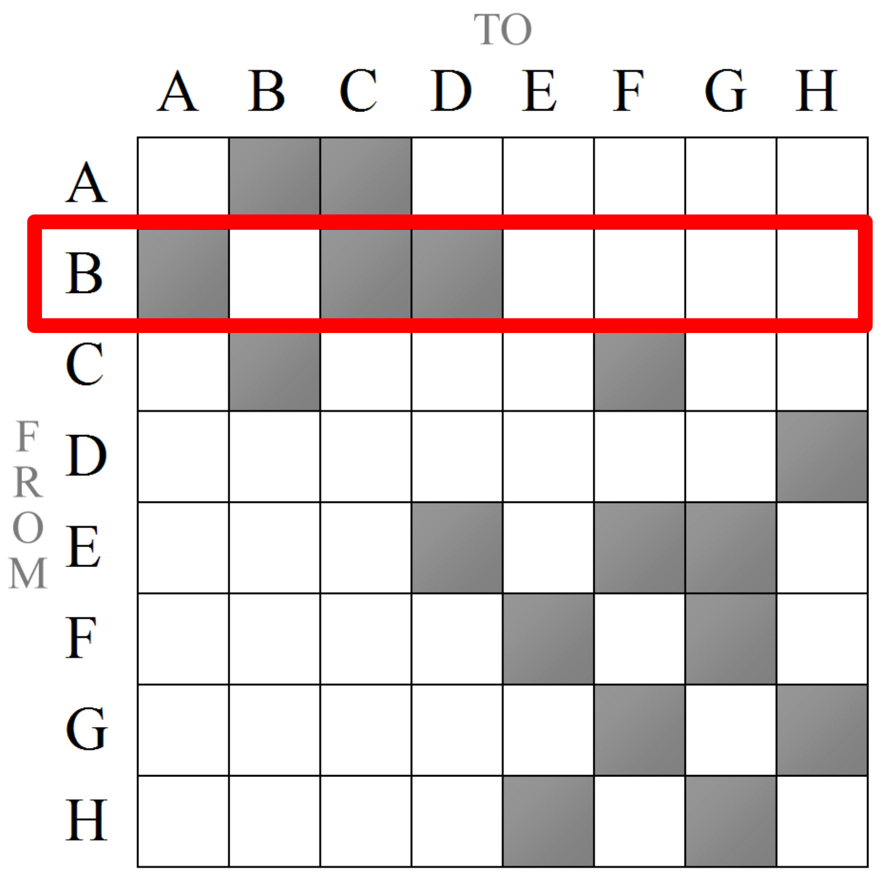
Node order is crucial: Reordering



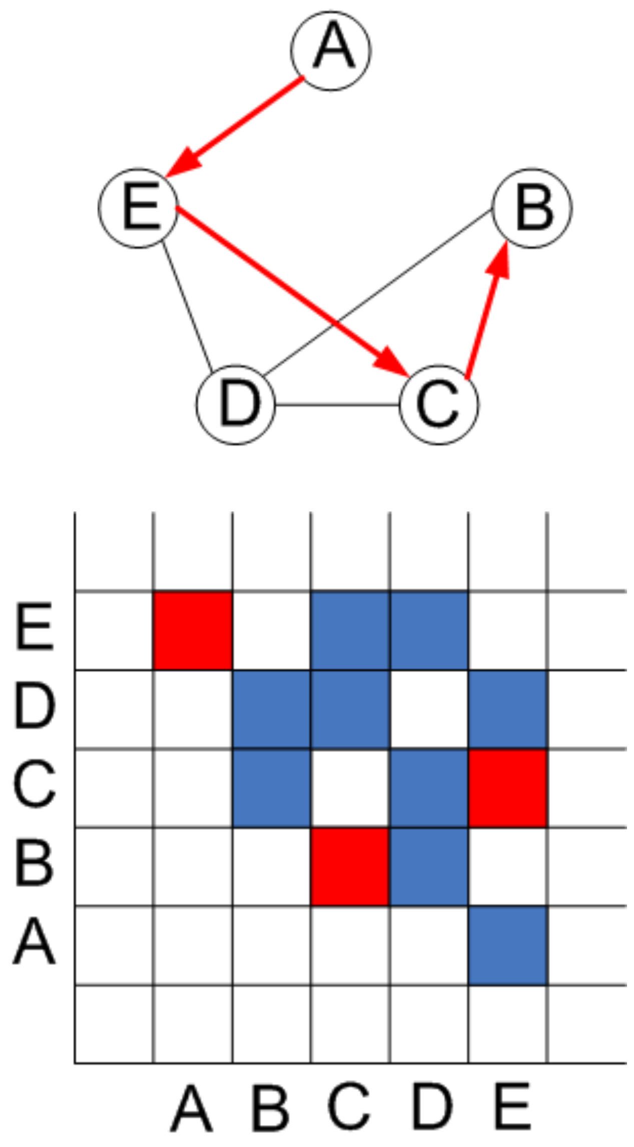


https://bost.ocks.org/mike/miserables/

Adjacency matrix

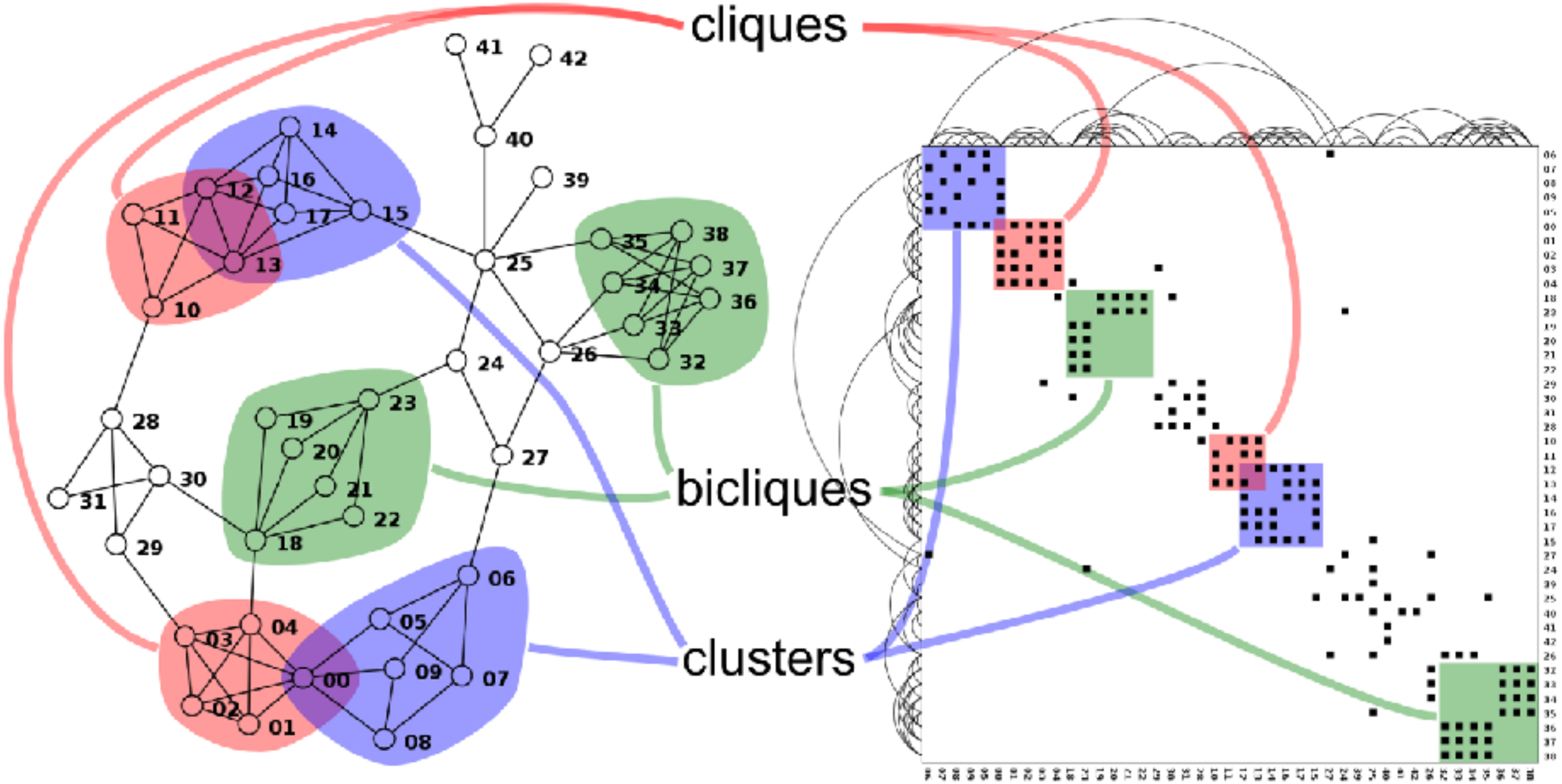


good for topology tasks related to neighborhoods (node 1-hop neighbors)



bad for topology tasks related to paths

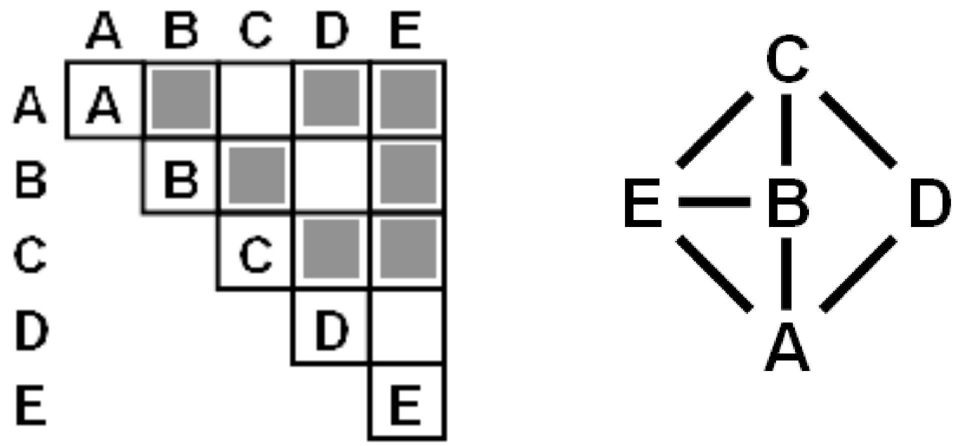
Structures visible in both



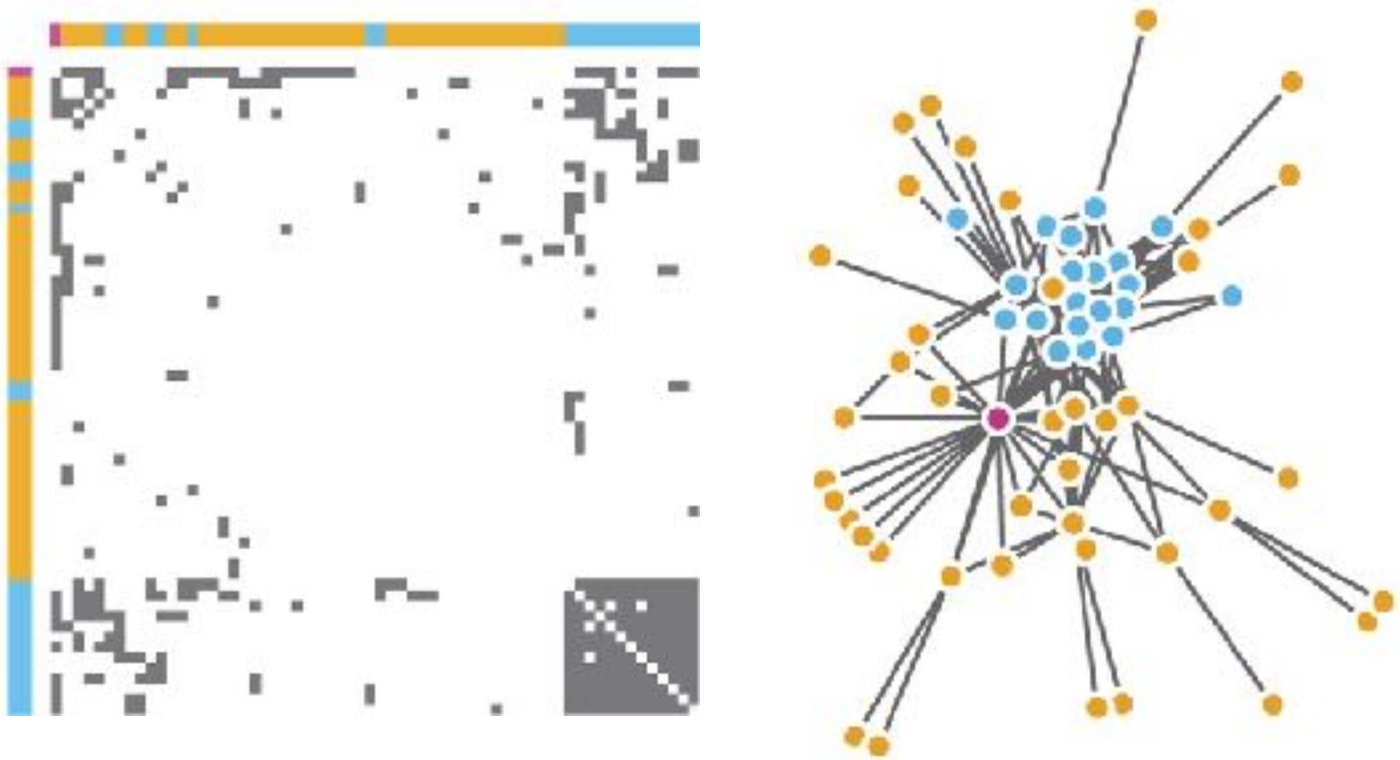
http://www.michaelmcguffin.com/courses/vis/patternsInAdjacencyMatrix.png

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
- visual encoding
  - cell shows presence/absence of edge
- 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.]

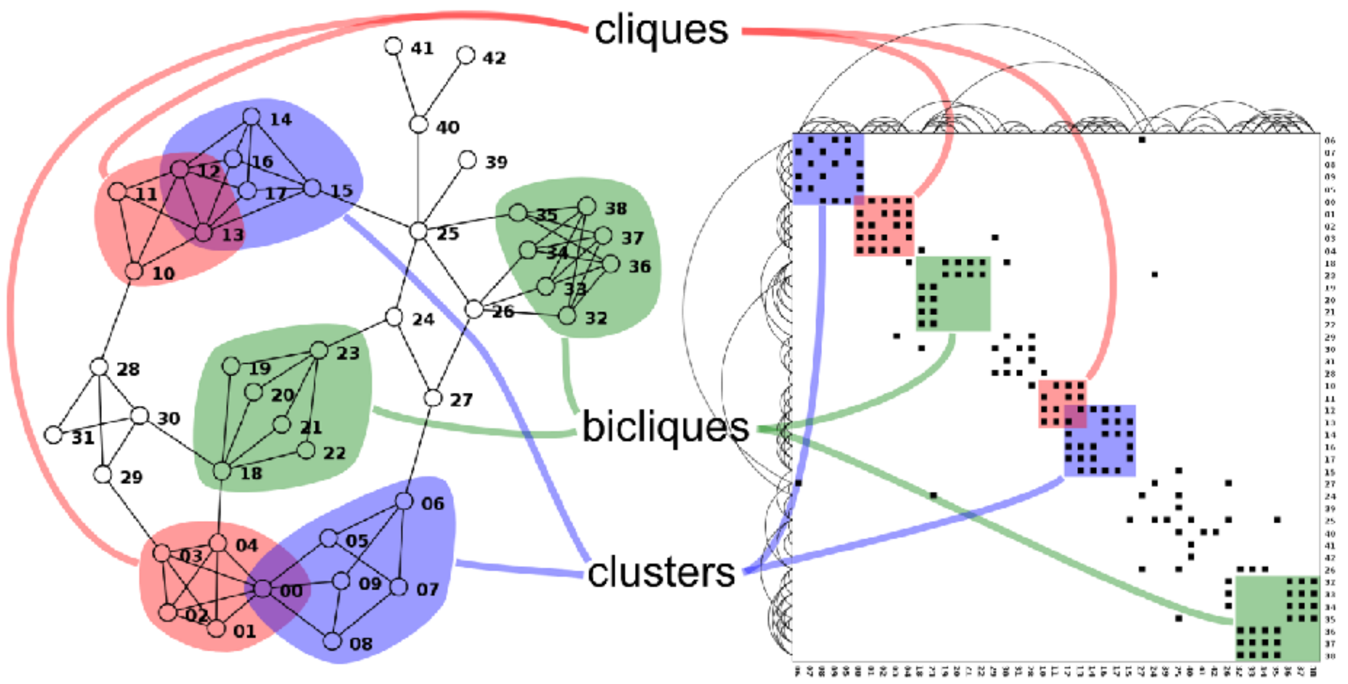


[Points of view: Networks. Gehlenborg and Wong. Nature Methods 9:115.]

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!

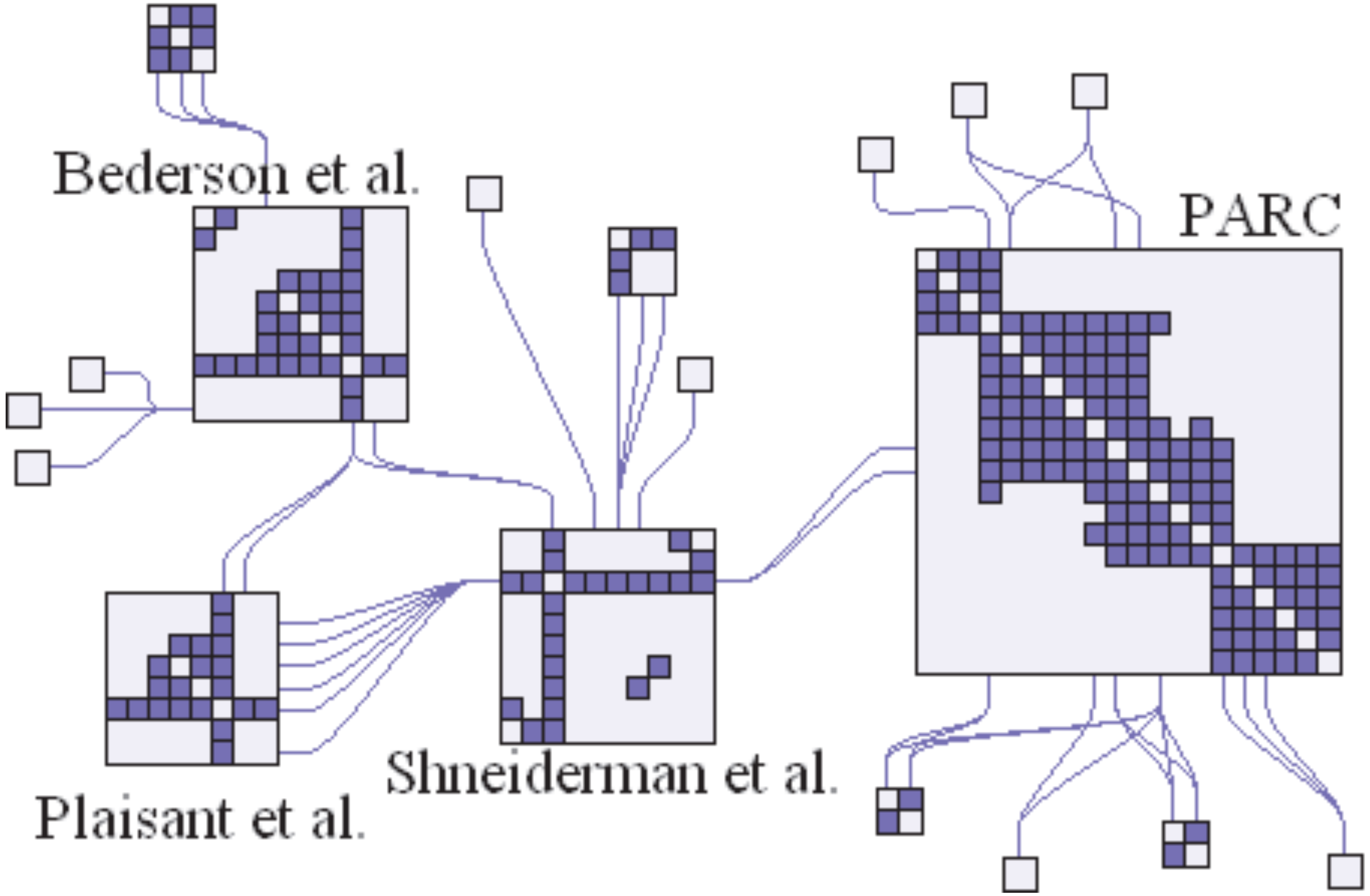
[On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. Ghoniem, Fekete, and Castagliola. Information Visualization 4:2 (2005), 114–135.]



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

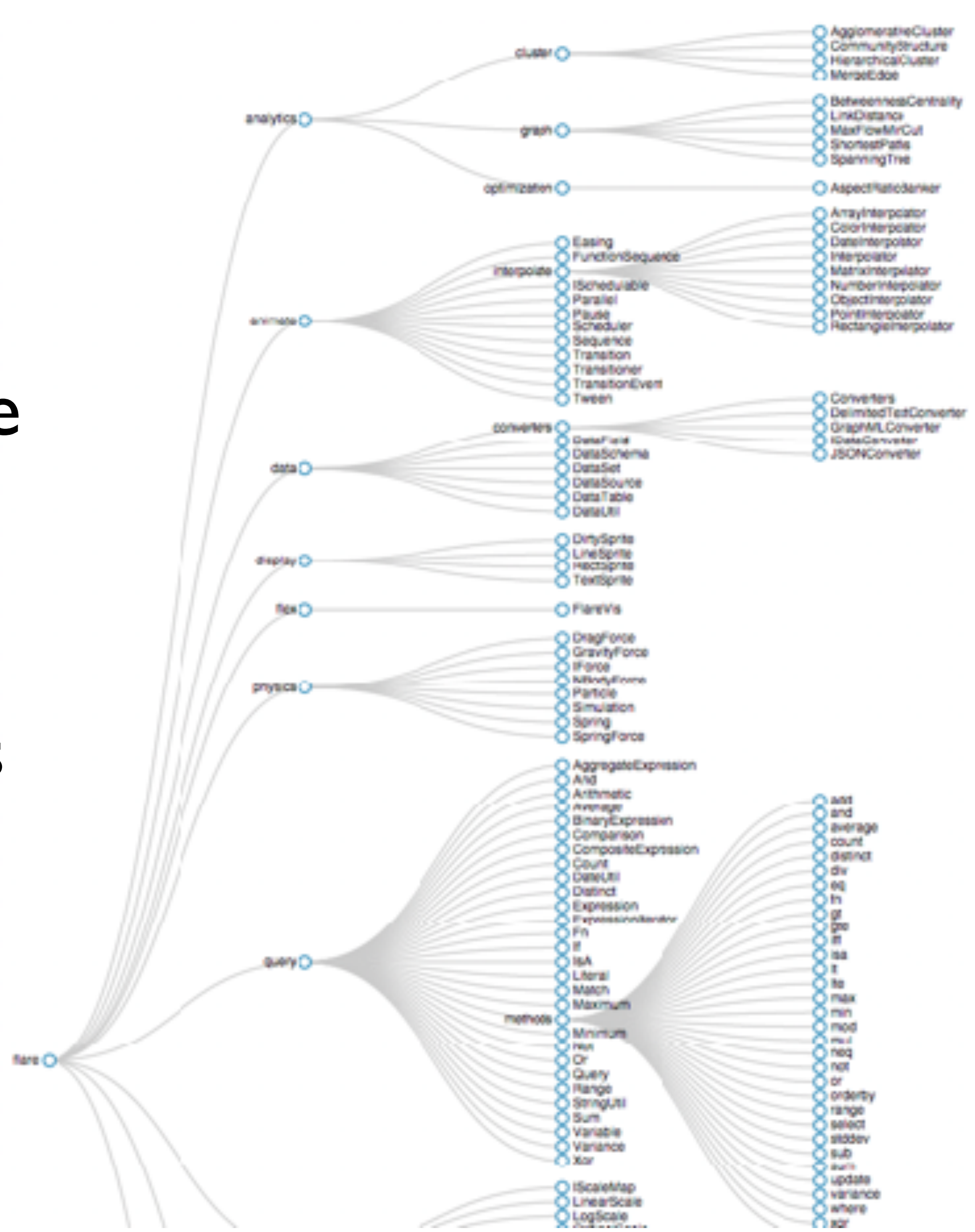
Node-link trees

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

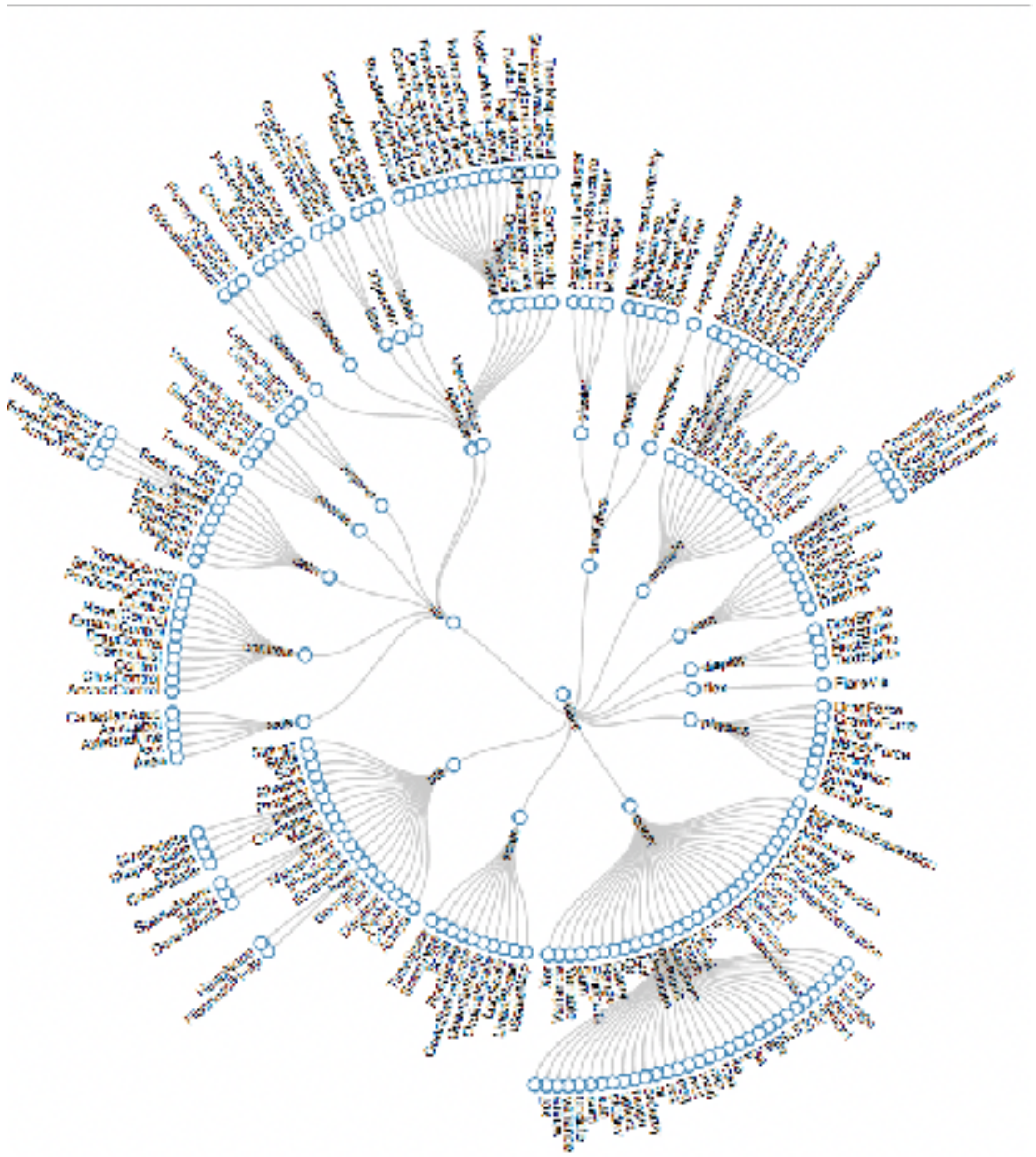
  [Tidier drawing of trees. Reingold and Tilford. IEEE Trans. Software Eng., SE-7(2):223–228, 1981.]

  - nice algorithm writeup

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



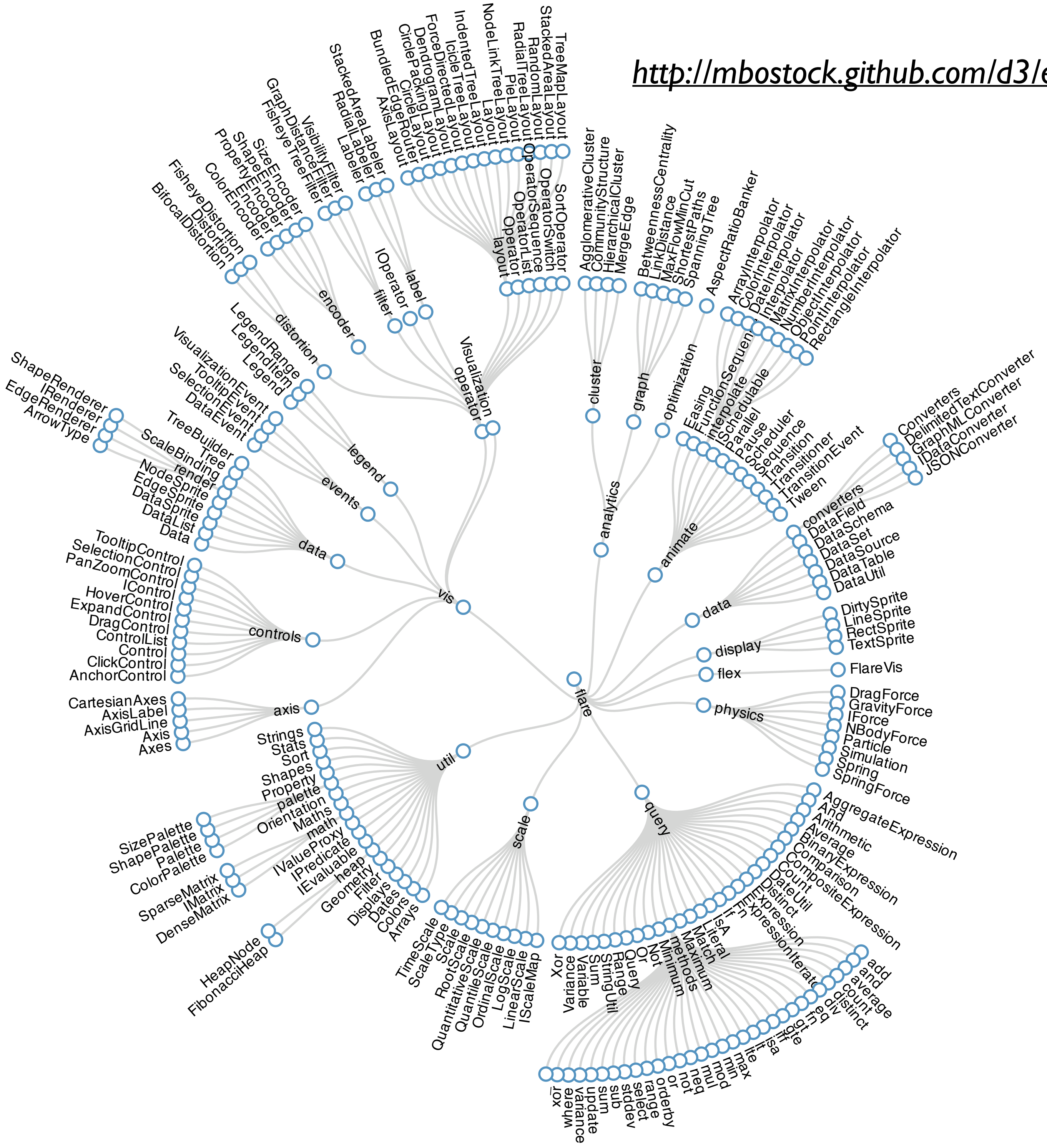
http://bl.ocks.org/mbostock/4339184



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

Idiom: **radial node-link tree**

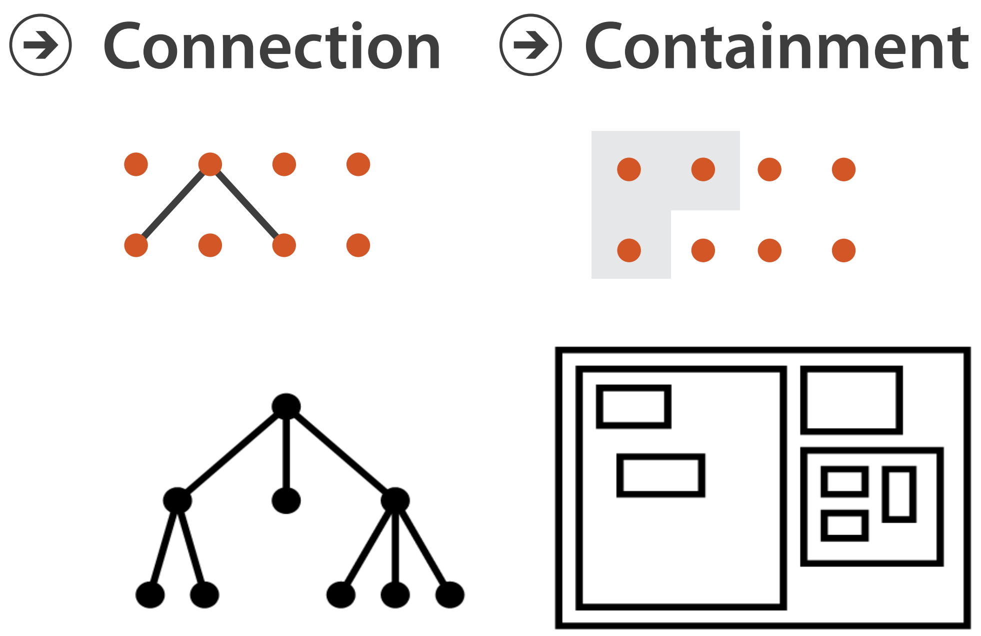
- data
  - tree
- encoding
  - link connection marks
  - point node marks
  - radial axis orientation
    - angular proximity: siblings
    - distance from center: depth in tree
- tasks
  - understanding topology, following paths
- scalability
  - 1K - 10K nodes (with/without labels)

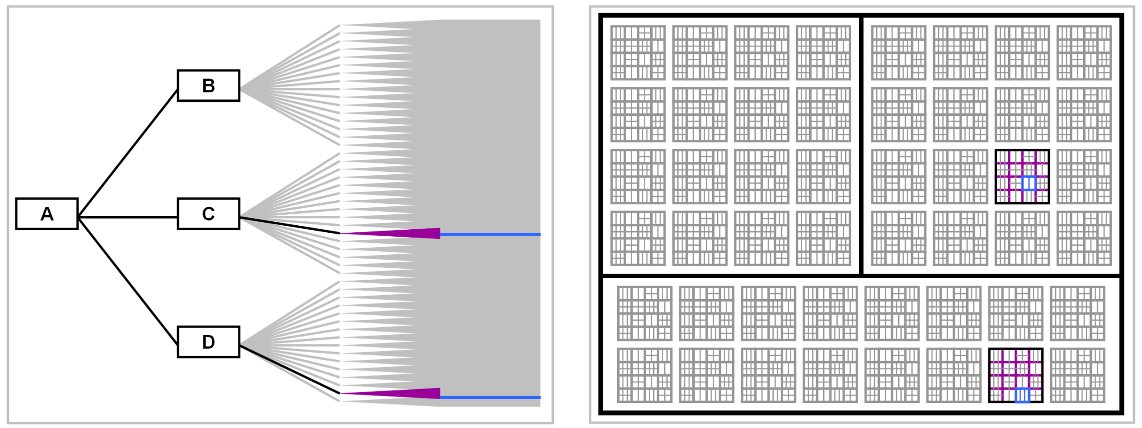


http://mbostock.github.com/d3/ex/tree.html

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



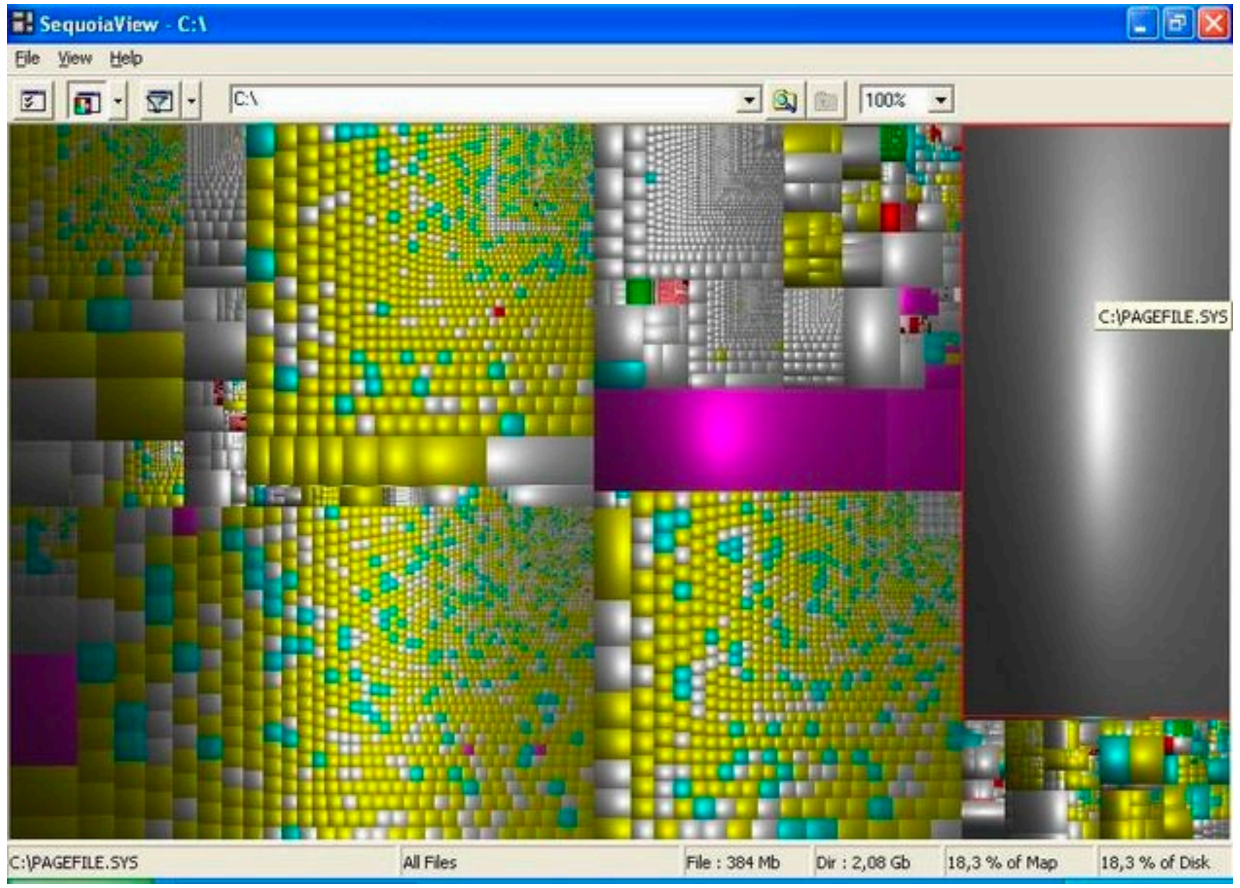


Node–Link Diagram   Treemap

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

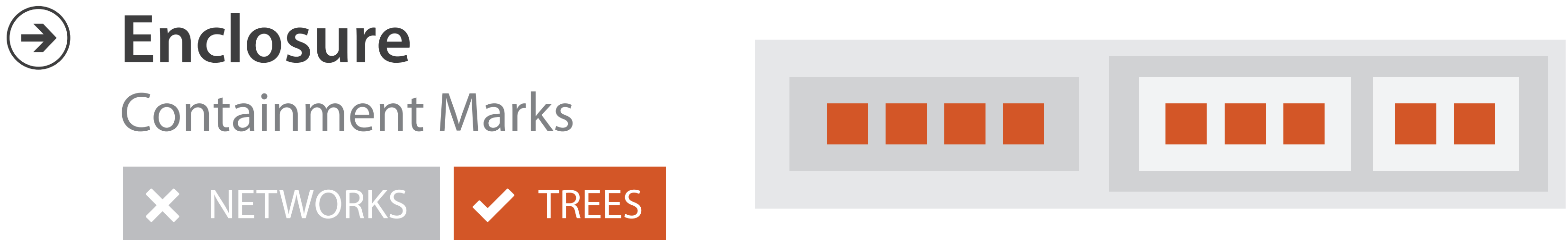
Idiom: **treemap**

- data
  - tree
  - 1 quant attrib at leaf nodes
- encoding
  - area containment marks for hierarchical structure
  - rectilinear orientation
  - size encodes quant attrib
- tasks
  - query attribute at leaf nodes
  - 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.]



Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position
  - show parent-child relationships only through relative positions

Treemap

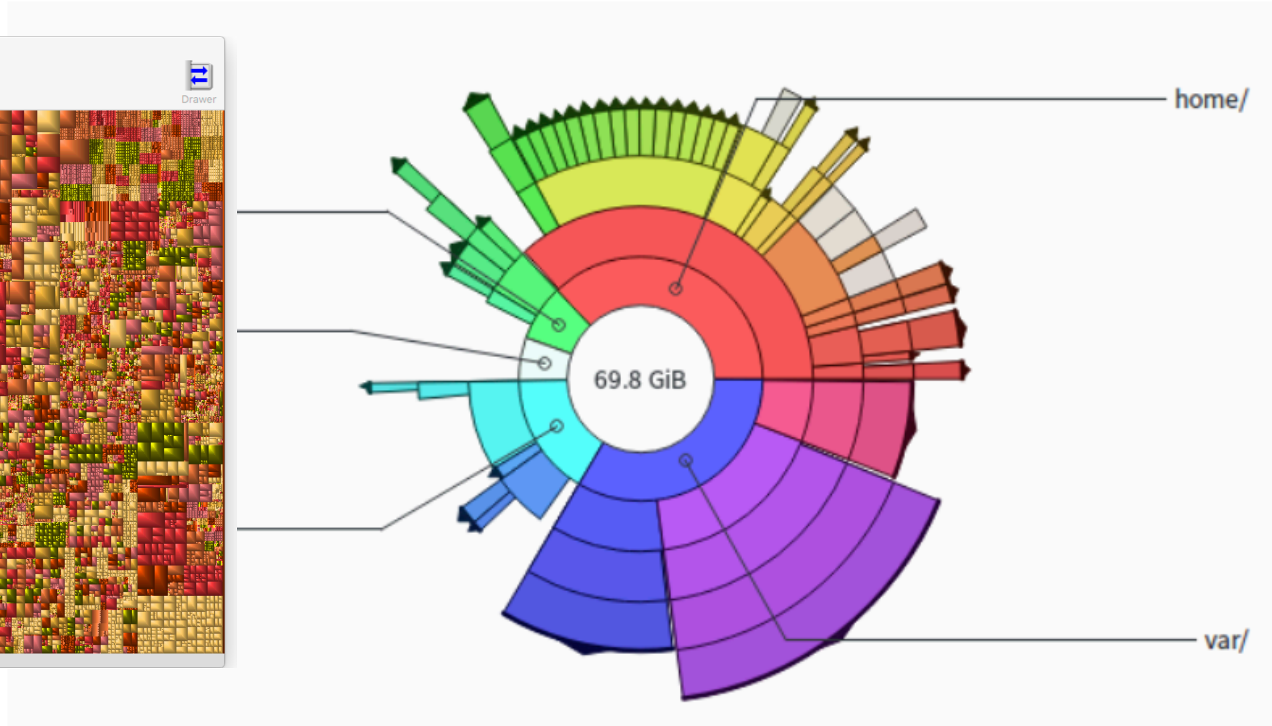
containment

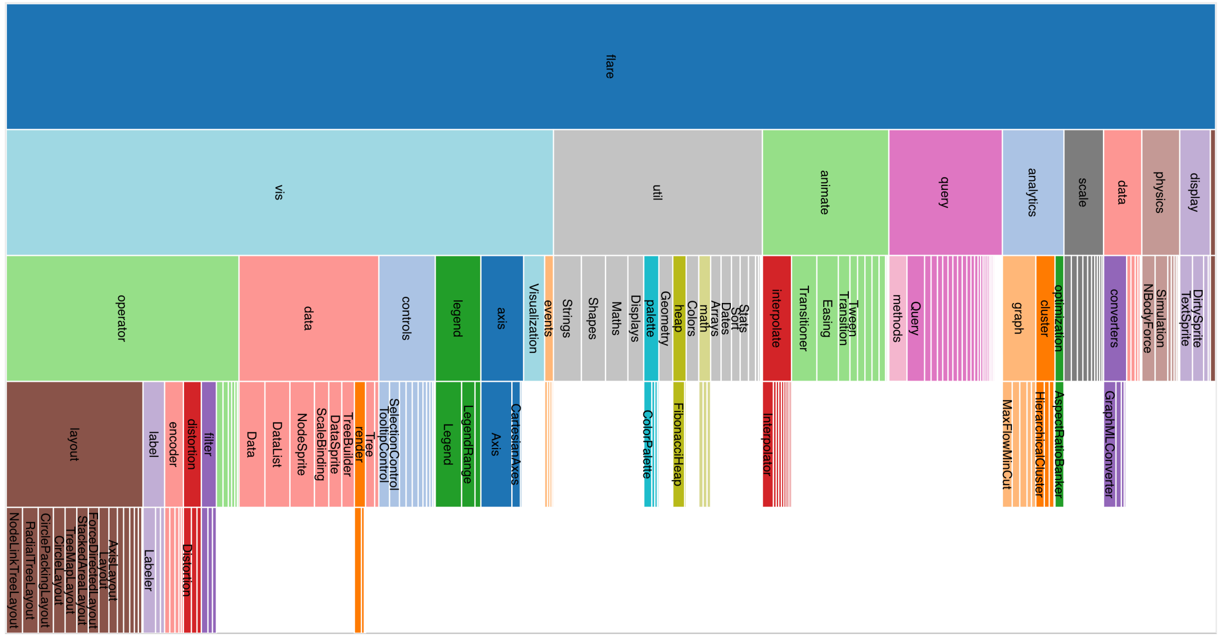
Sunburst

position (radial)

Icicle Plot

position (rectilinear)





Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position
  - show parent-child relationships only through relative positions

Treemap

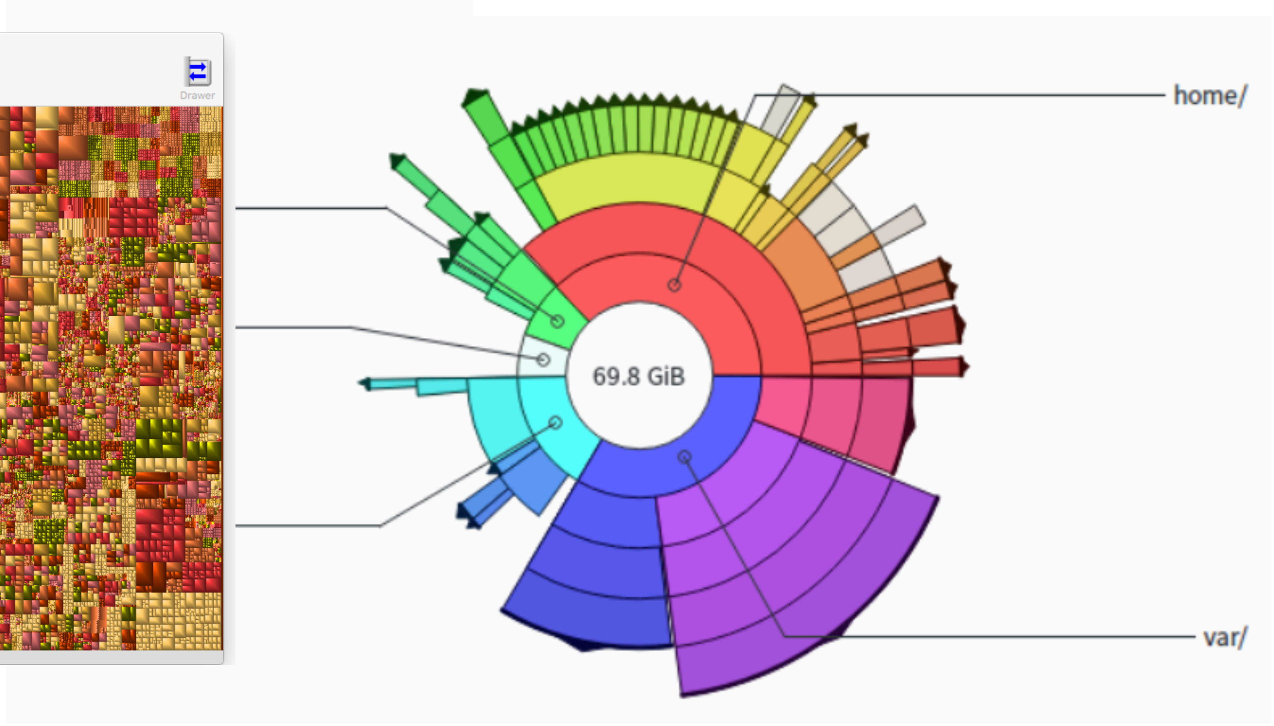
containment

only leaves visible

Sunburst

position (radial)

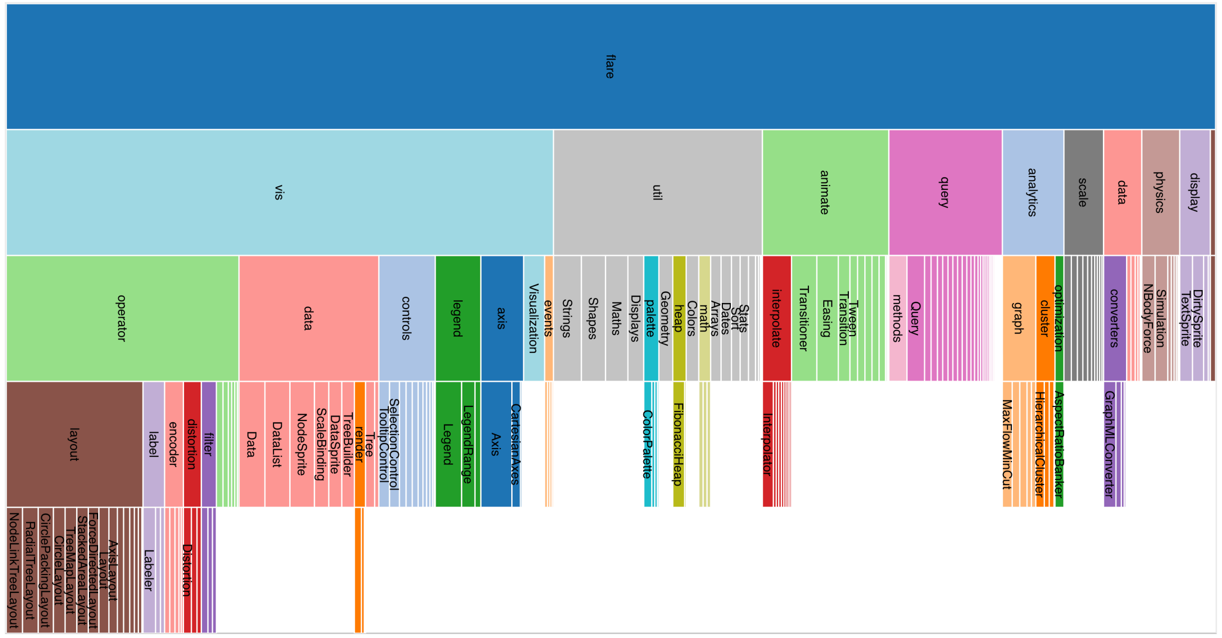
inner nodes & leaves visible



Icicle Plot

position (rectilinear)

inner nodes & leaves visible



Idiom: implicit tree layouts (sunburst, icicle plot)

- alternative to connection and containment: position
  - show parent-child relationships only through relative positions

Treemap

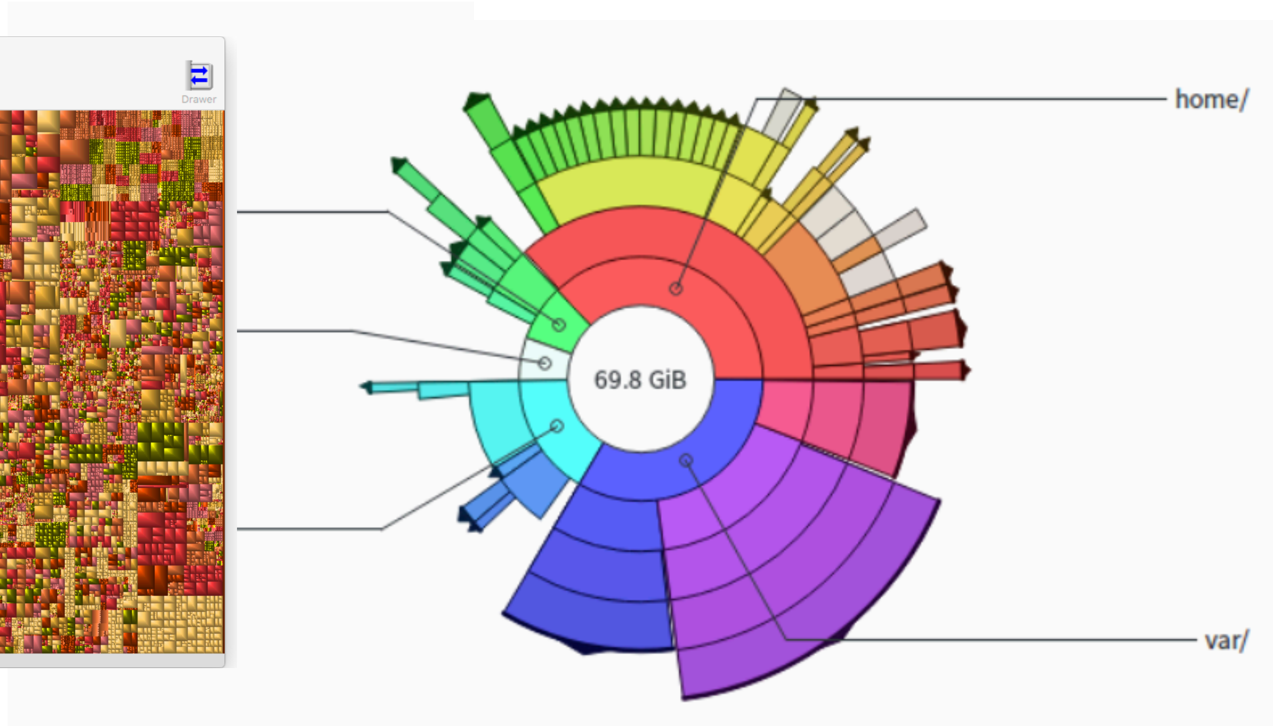
containment

only leaves visible

Sunburst

position (radial)

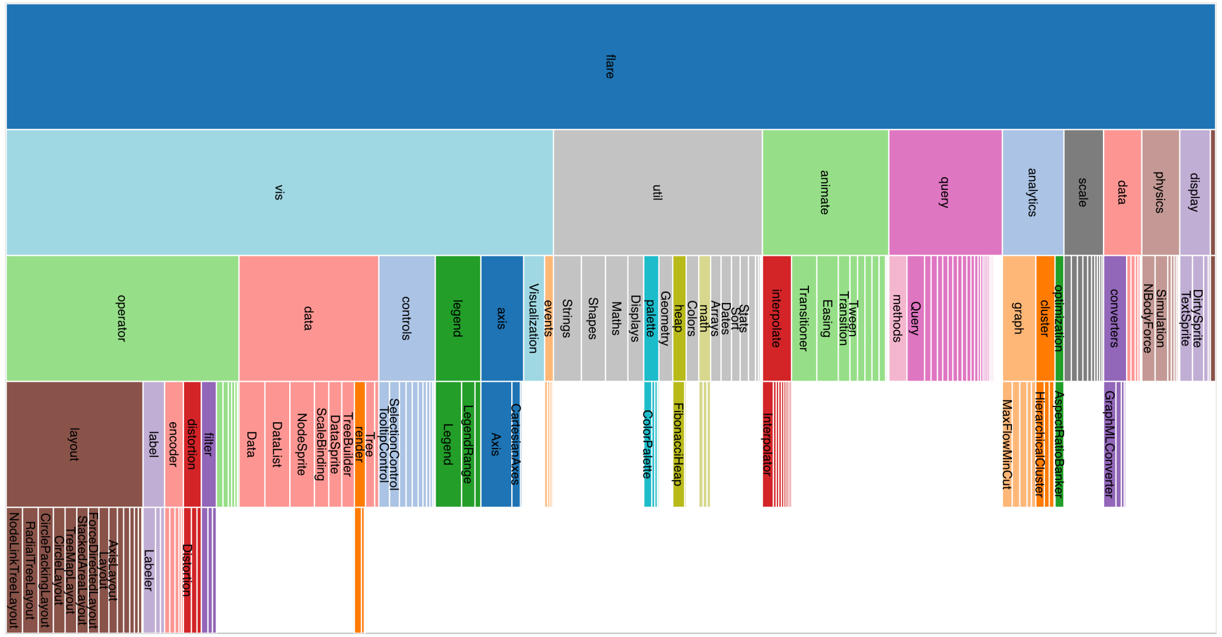
inner nodes & leaves visible



Icicle Plot

position (rectilinear)

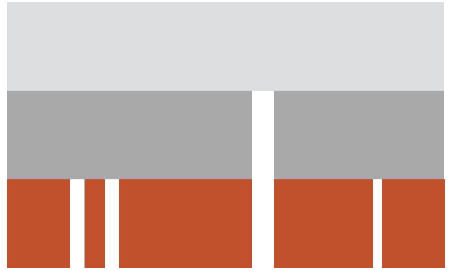
inner nodes & leaves visible



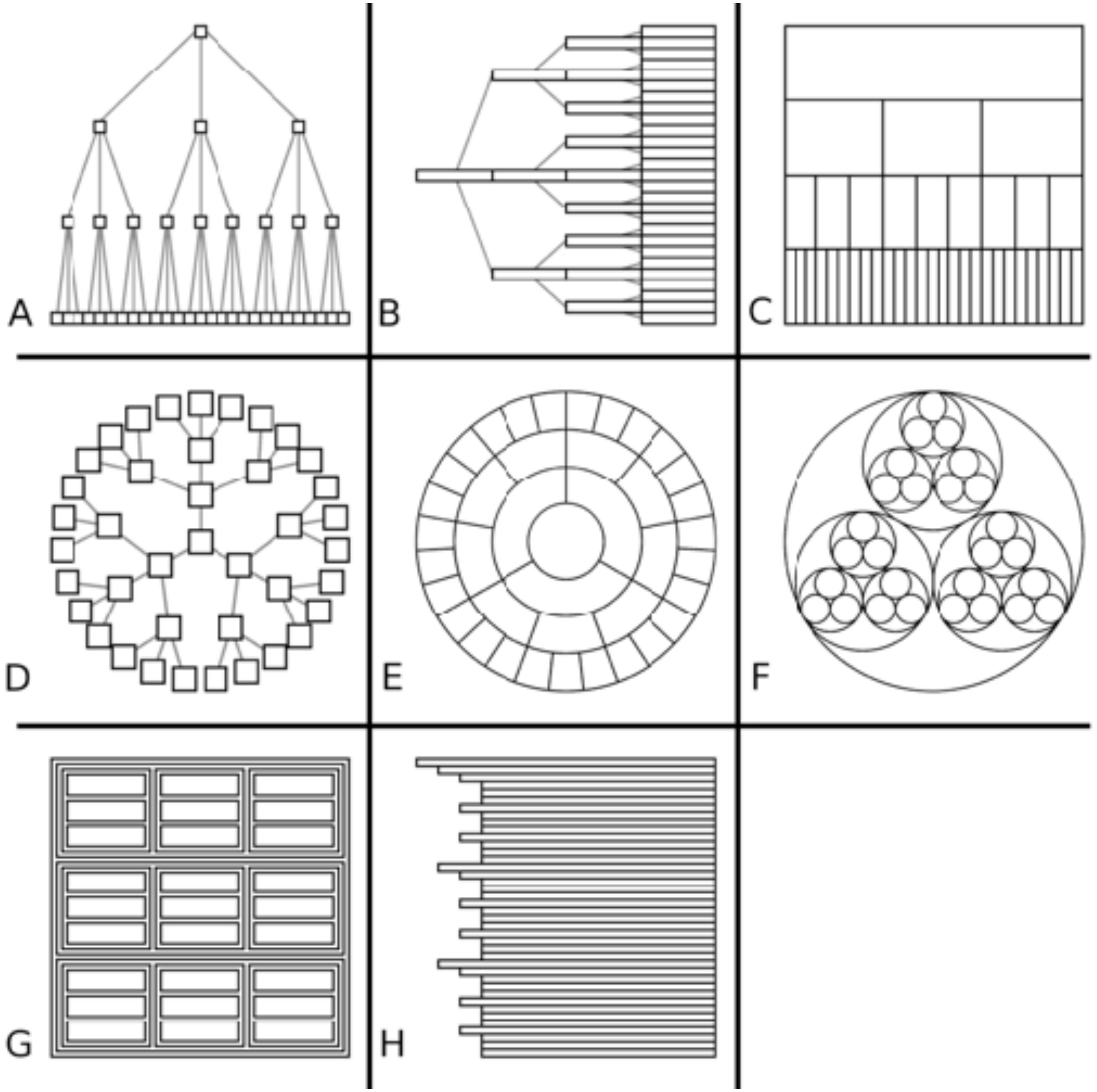
➔ Implicit

Spatial Position

✕ NETWORKS ✔ TREES



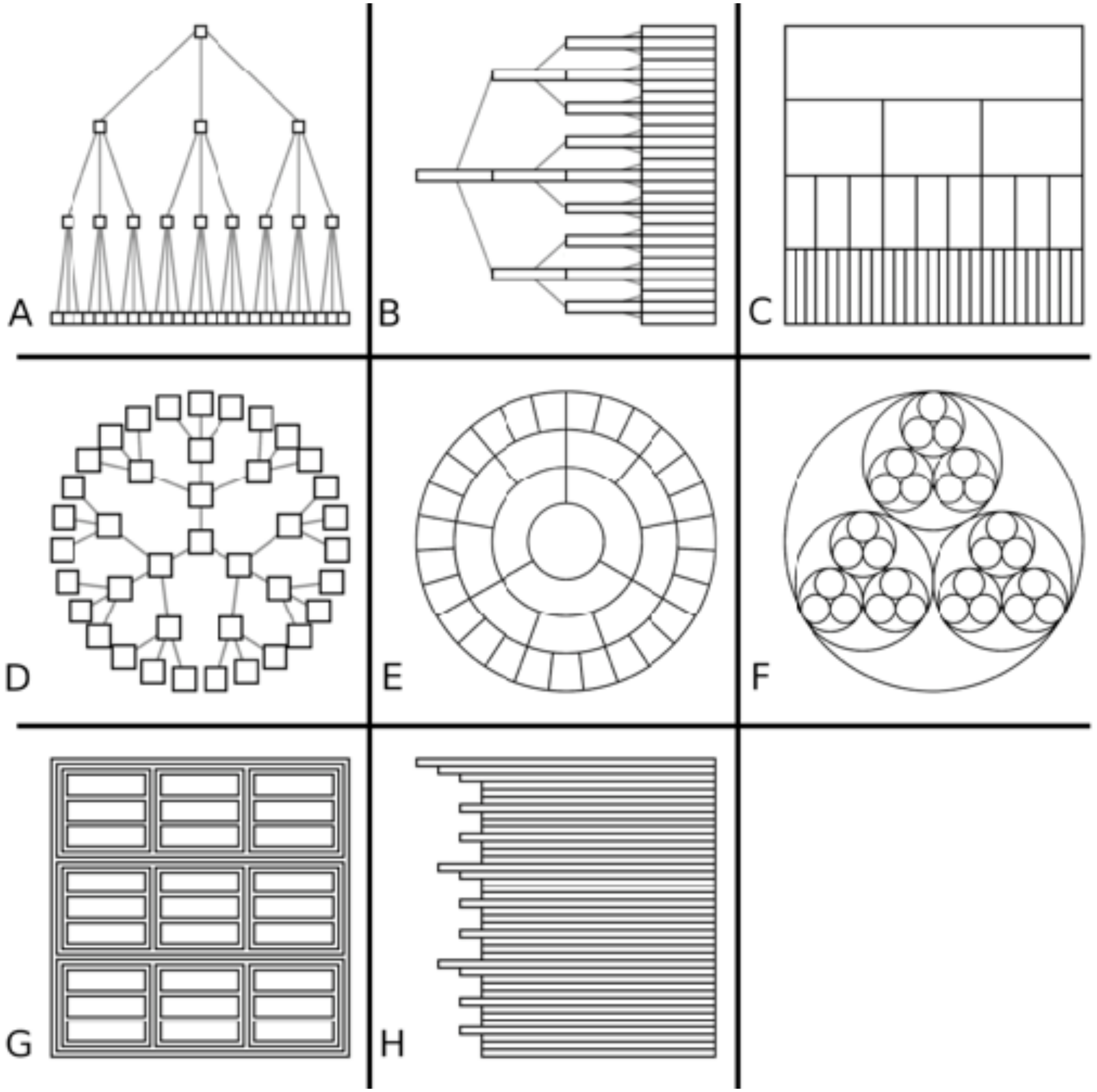
Tree drawing idioms comparison



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

Comparison: tree drawing idioms

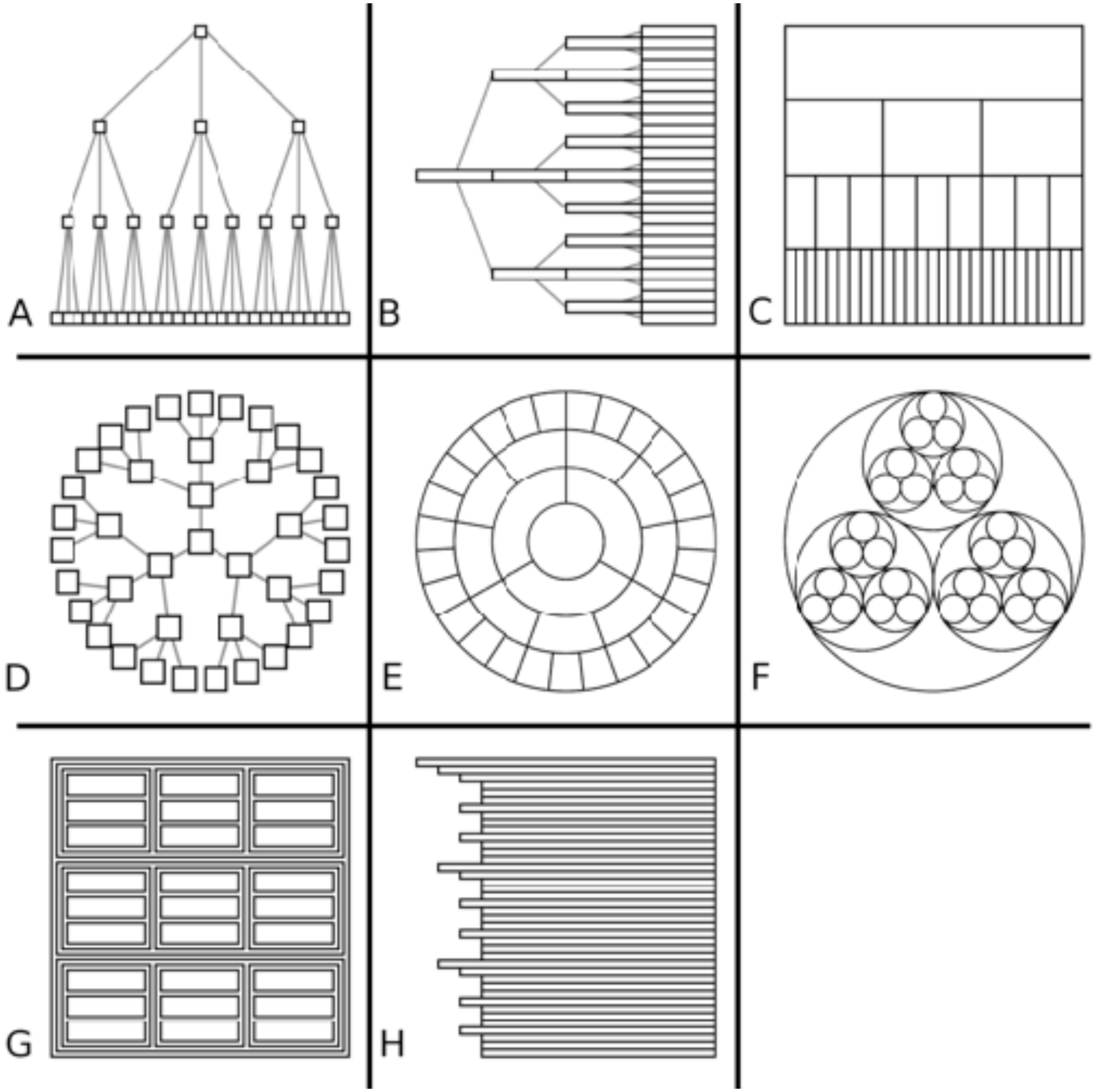
- data shown
  - link relationships
  - tree depth
  - sibling order



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

Comparison: tree drawing idioms

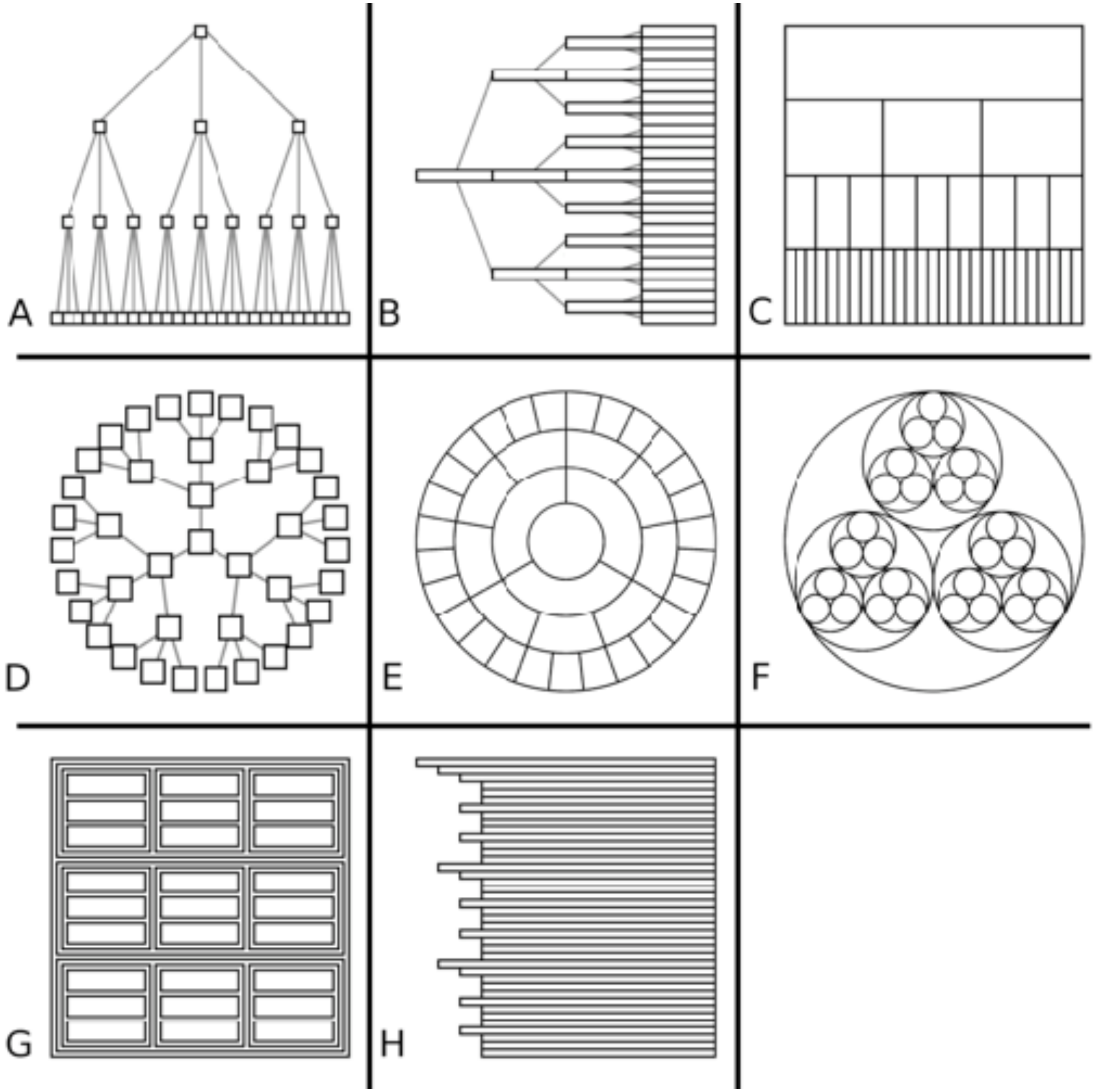
- data shown
  - link relationships
  - tree depth
  - sibling order
- design choices
  - connection vs containment link marks
  - rectilinear vs radial layout
  - spatial position channels



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

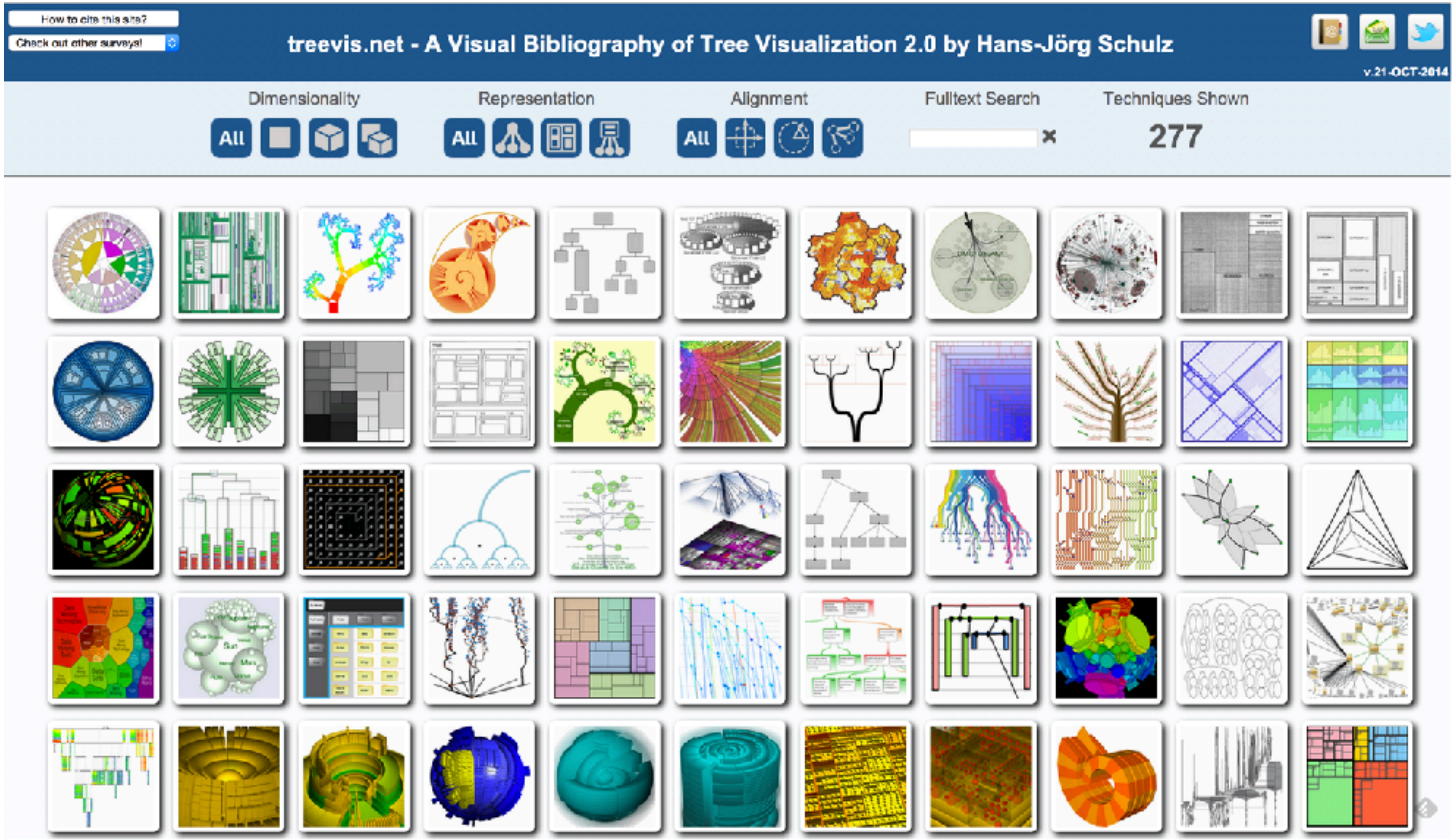
Comparison: tree drawing idioms

- data shown
  - link relationships
  - tree depth
  - sibling order
- design choices
  - connection vs containment link marks
  - rectilinear vs radial layout
  - spatial position 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!

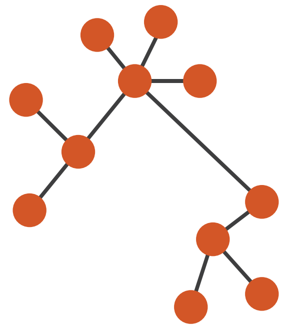


https://treevis.net/

Arrange networks and trees

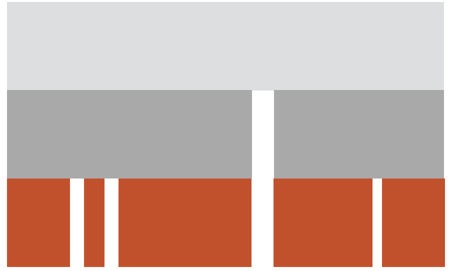
Node–Link Diagrams
Connection Marks

✔ NETWORKS ✔ TREES



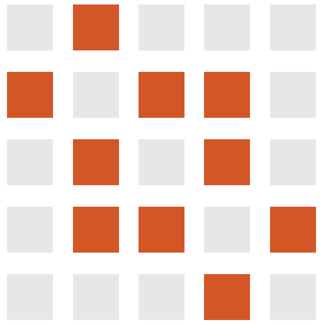
Implicit
Spatial Position

✘ NETWORKS ✔ TREES



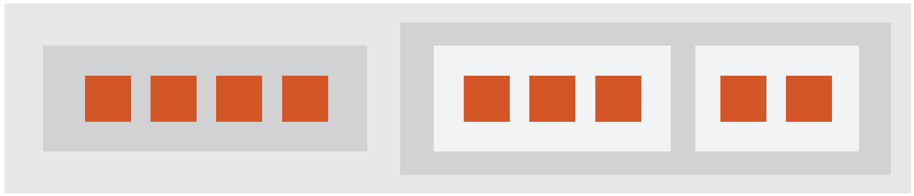
Adjacency Matrix
Derived Table

✔ NETWORKS ✔ TREES



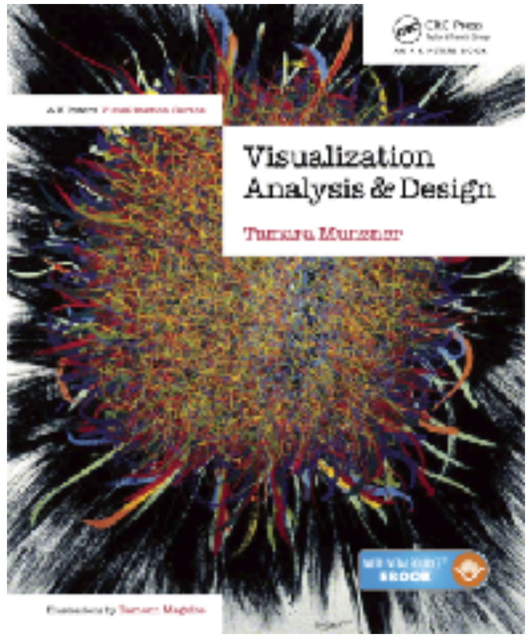
Enclosure
Containment Marks

✘ NETWORKS ✔ TREES



Visualization Analysis & Design

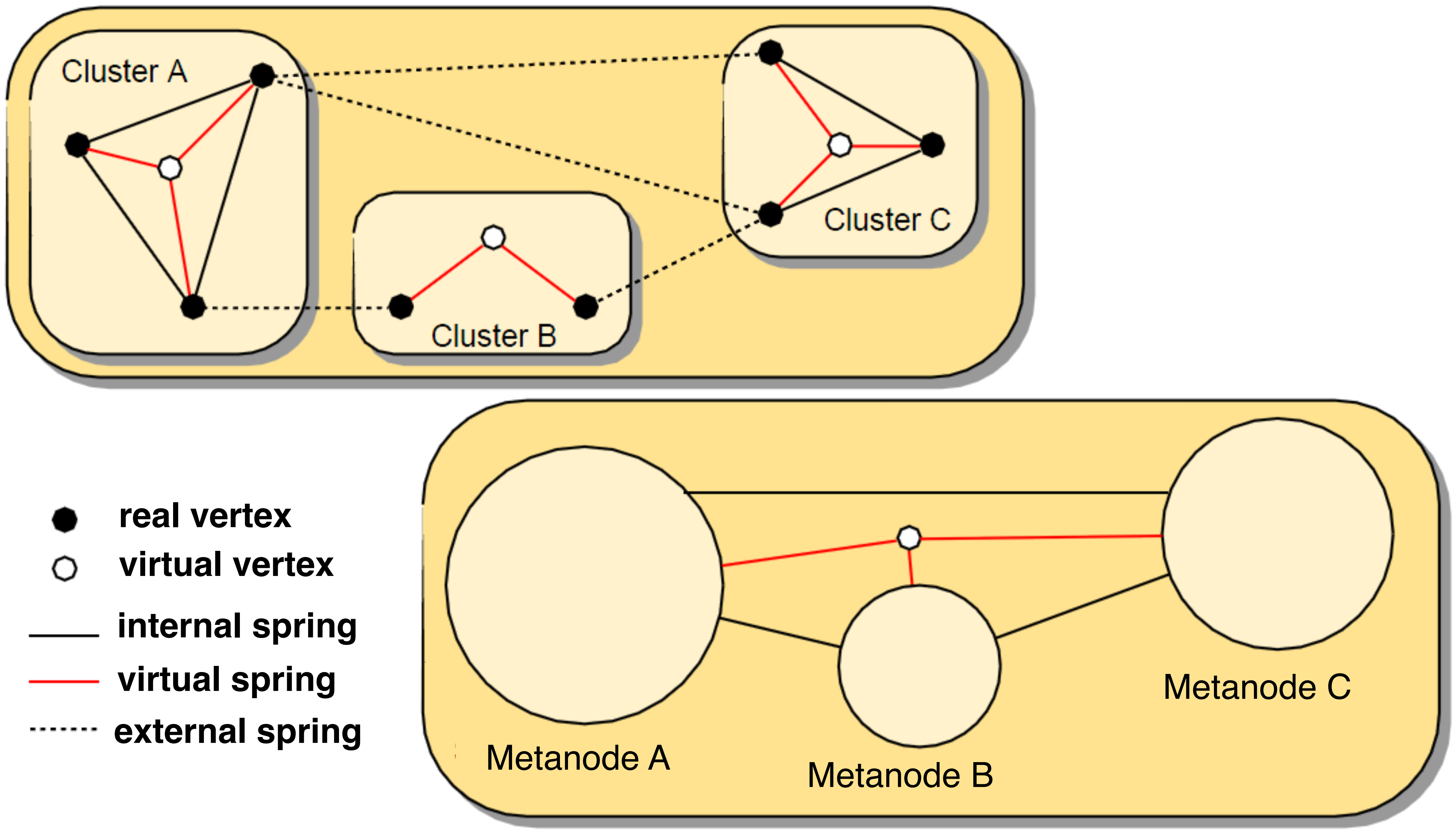
Network Data (Ch 9) II

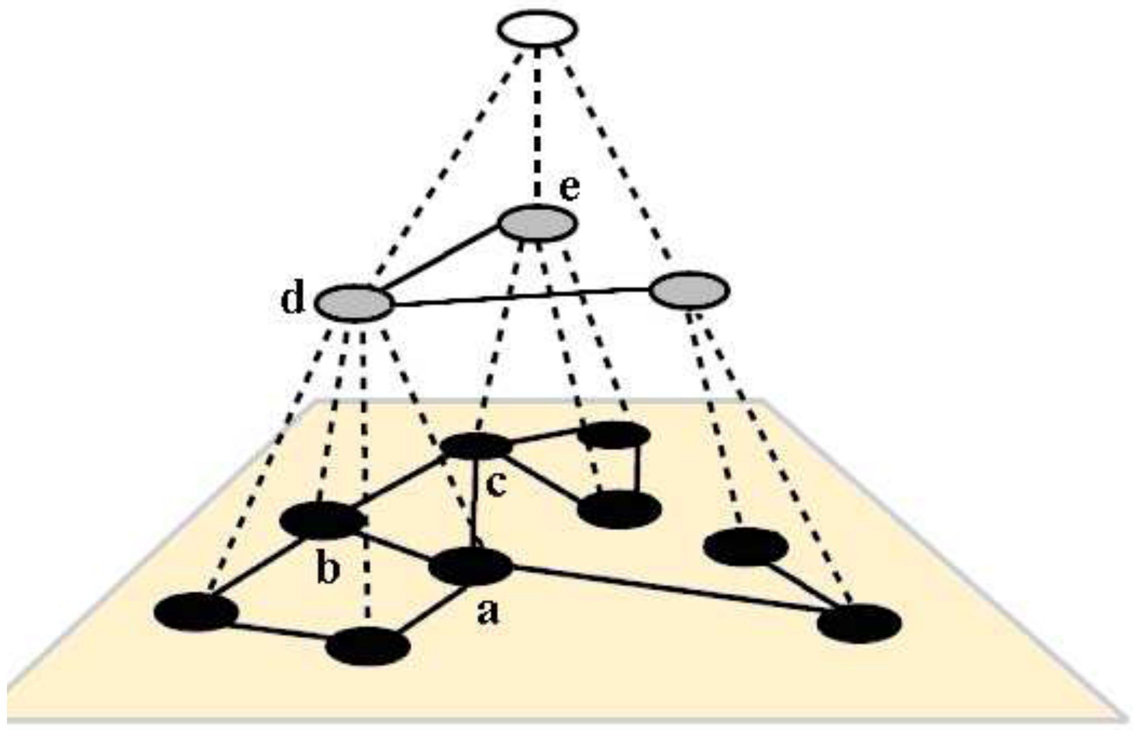


Tamara Munzner
Department of Computer Science
University of British Columbia
@tamaramunzner

Multilevel networks

- derive cluster hierarchy of metanodes on top of original graph nodes

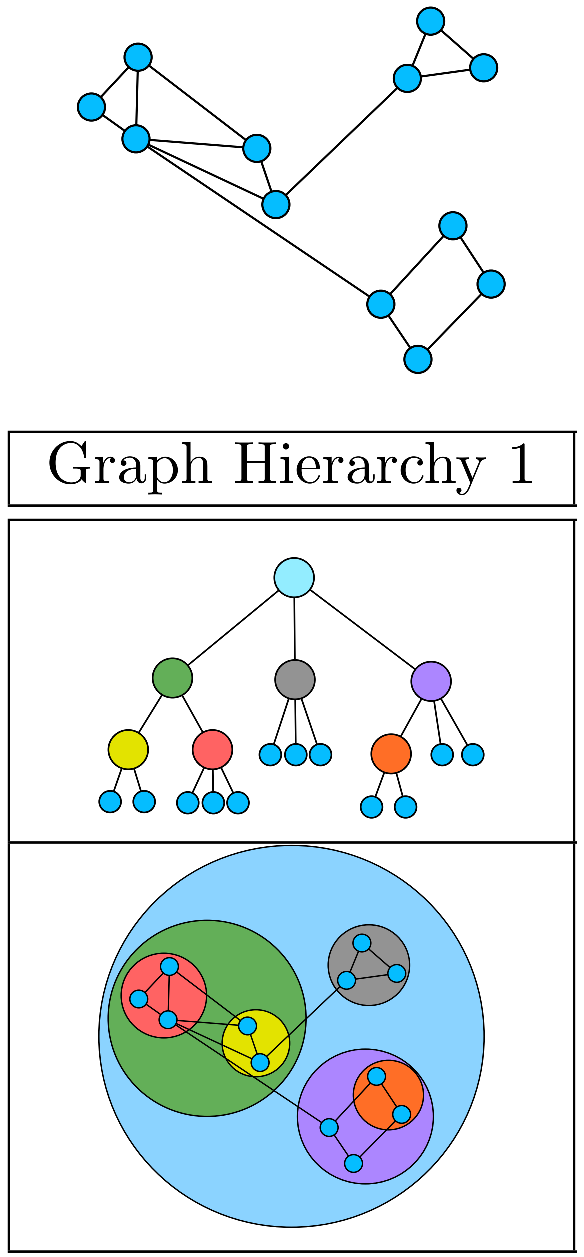




[Schulz 2004]

Idiom: **GrouseFlocks**

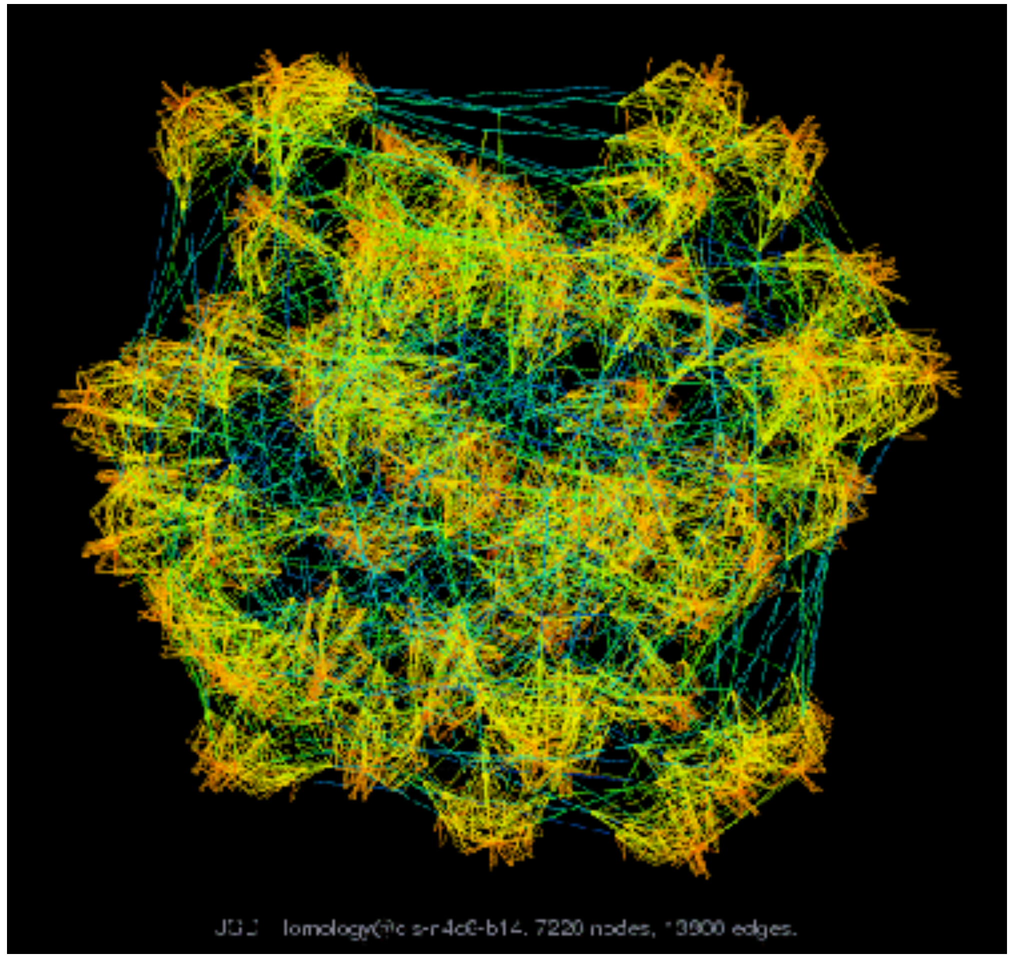
- data: compound network
  - network
  - cluster hierarchy atop it
    - derived or interactively chosen
- visual encoding
  - connection marks for network links
  - containment marks for hierarchy
  - point marks for nodes
- dynamic interaction
  - select individual metanodes in hierarchy to expand/ contract



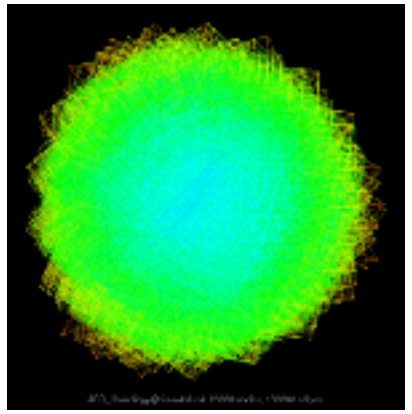
[GrouseFlocks: Steerable Exploration of Graph Hierarchy Space. Archambault, Munzner, and Auber. IEEE TVCG 14(4):900-913, 2008.]

Idiom: **sfdp** (multi-level force-directed placement)

- data: compound graph
  - original: network
  - derived: cluster hierarchy atop it
- considerations
  - better algorithm for same encoding technique
    - same: fundamental use of space
    - hierarchy used for algorithm speed/quality but not shown explicitly
- scalability
  - nodes, edges: 1K-10K
  - hairball problem eventually hits



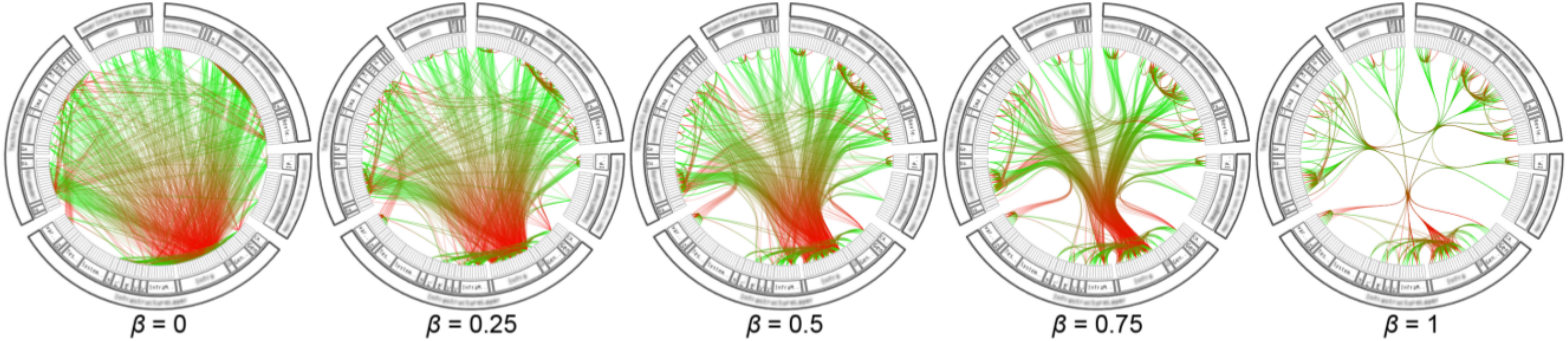
[Efficient and high quality force-directed graph drawing. Hu. The Mathematica Journal 10:37–71, 2005.]



http://www.research.att.com/yifanhu/GALLERY/GRAPHS/index1.html

Idiom: **hierarchical edge bundling**

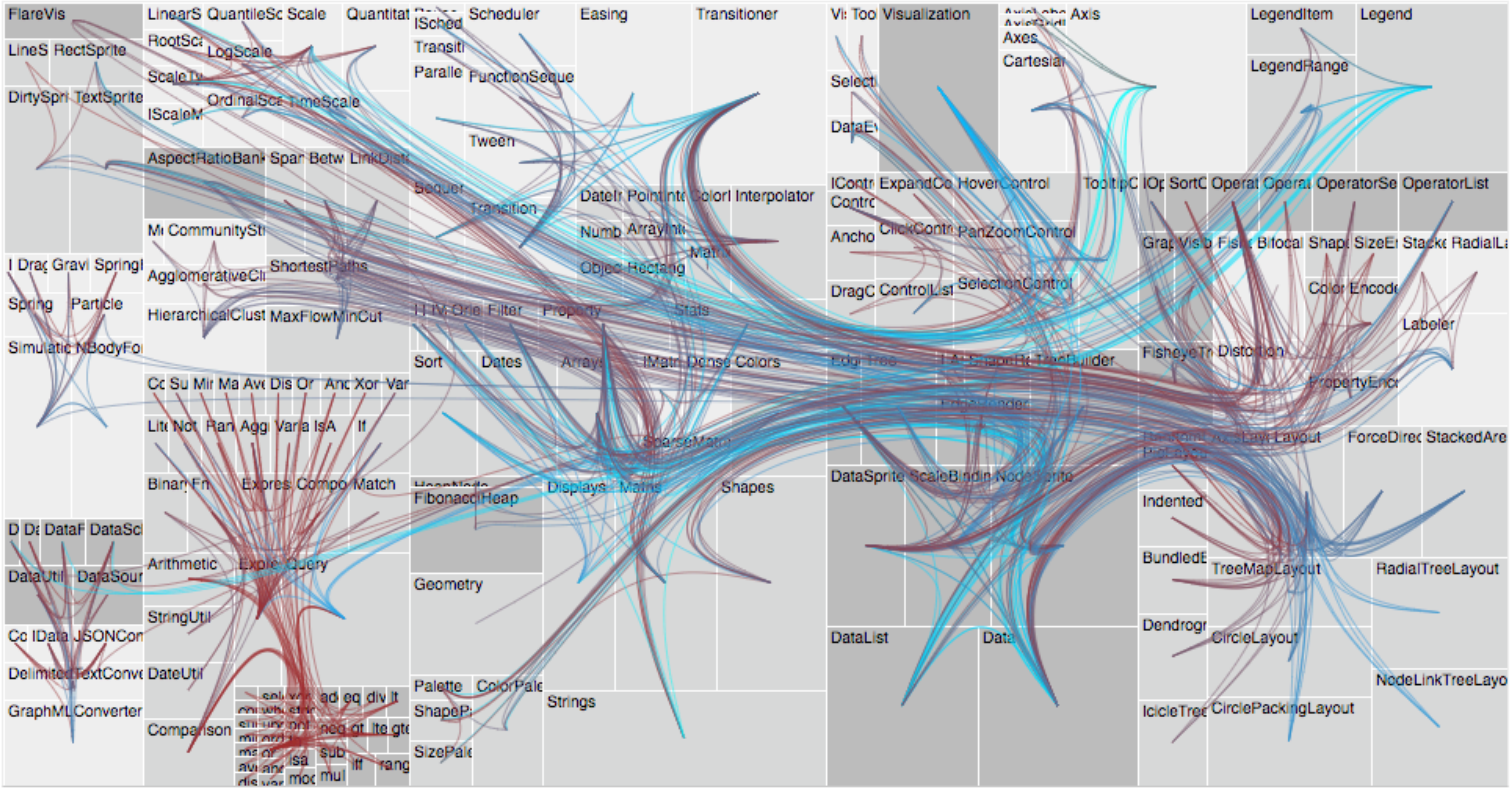
- data
  - any layout of compound network
    - network: software classes (nodes), import/export between classes (links)
    - cluster hierarchy: class package structure
  - derived: bundles of edges with same source/destination (multi-level)
- idiom: curve edge routes according to bundles
- task: edge clutter reduction

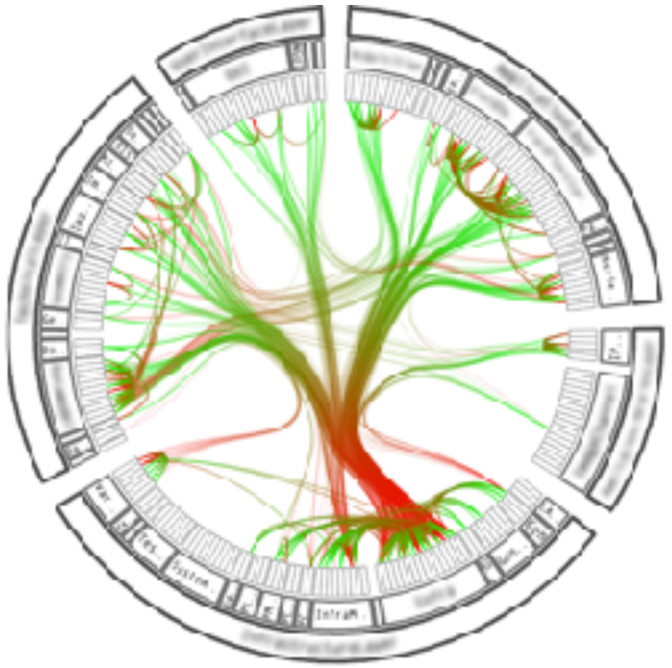


[Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. Danny Holten. TVCG 12(5):741-748 2006]

Hierarchical edge bundling

- works for any layout: treemap vs radial





[Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. Danny Holten. TVCG 12(5):741-748 2006]