Part IV Arranging Space

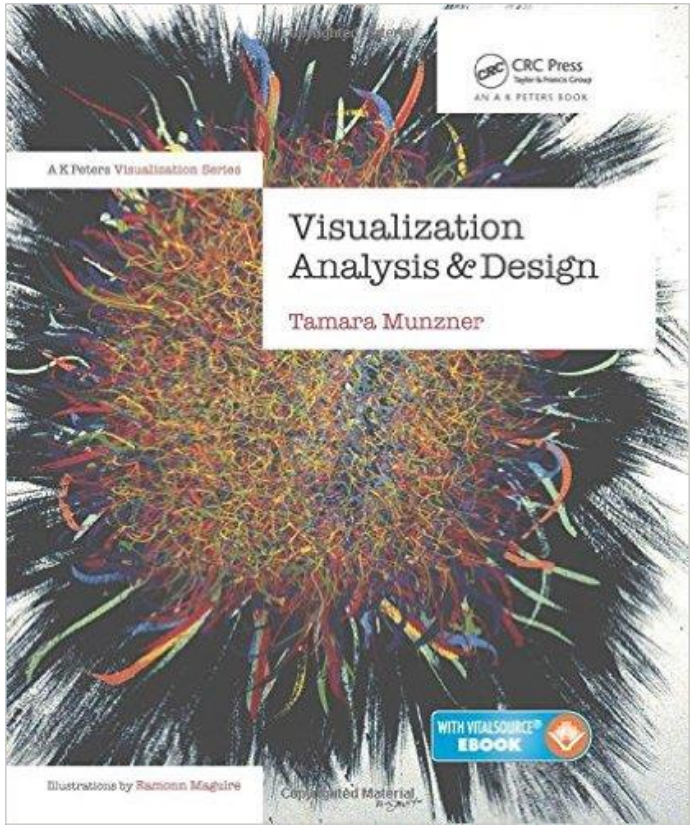
Networks and Trees

That's been one of my mantras - focus and simplicity.
Simple can be harder than complex:
You have to work hard to get your thinking clean to make it simple.
But it's worth it in the end because once you get there, you can move mountains.

Steve Jobs

Source of This Unit

- **Material of this unit is based on Chapter 9 of Tamara Munzner, *Visualization Analysis and Design*, AK Peters/CRC Press, 2014.**

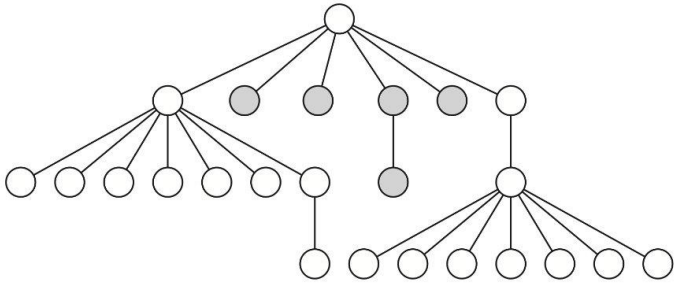


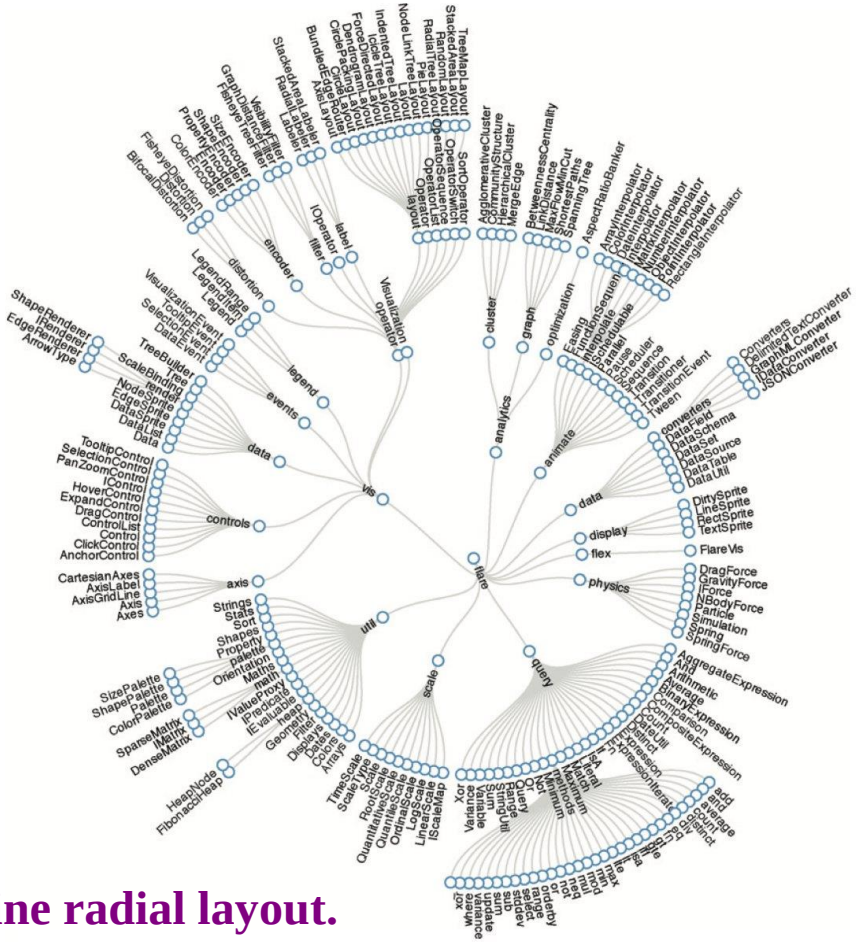
Some Initial Notes

- **The most common visual encoding idiom for tree and network data is node-link diagrams, where nodes are drawn as point marks and the links connecting them are drawn as line marks.**
- **The second major family of network encoding idioms are matrix views that directly show adjacency relationships.**
- **The tree structure can be shown with the containment channel, where enclosing link marks show hierarchical relationship through nesting.**

Connection: Link Marks 1/5

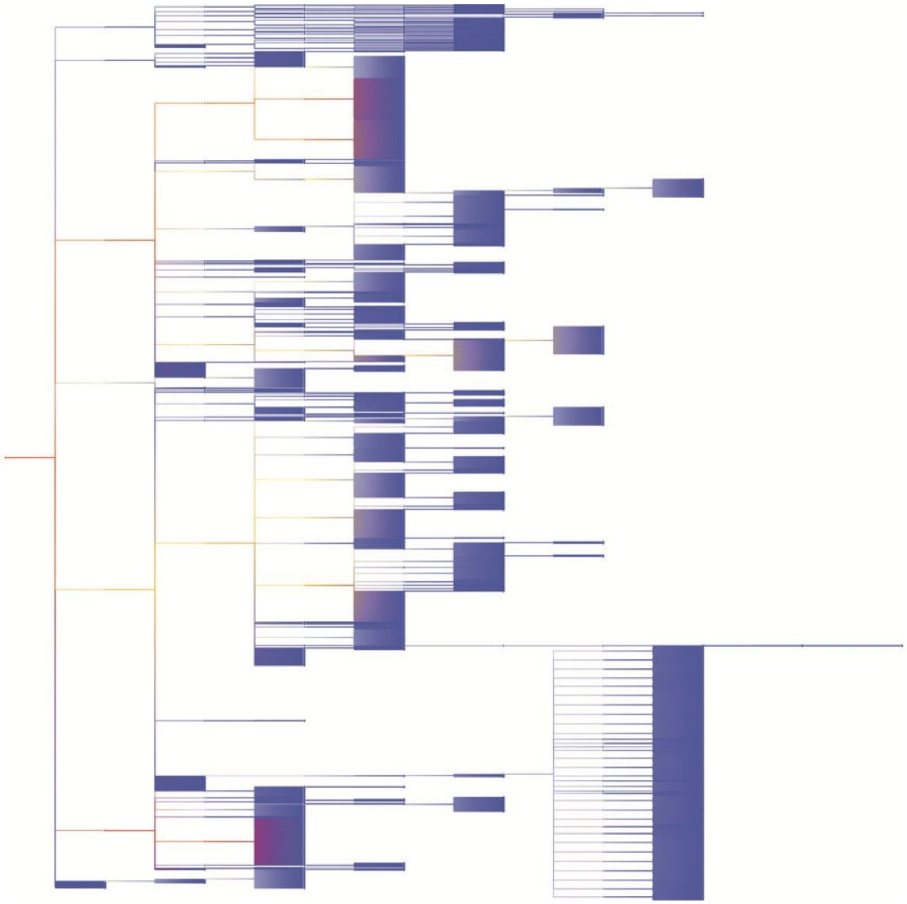
A tiny tree of 24 nodes with a triangular vertical node-link layout. It uses vertical spatial position channel to show the depth in the tree.



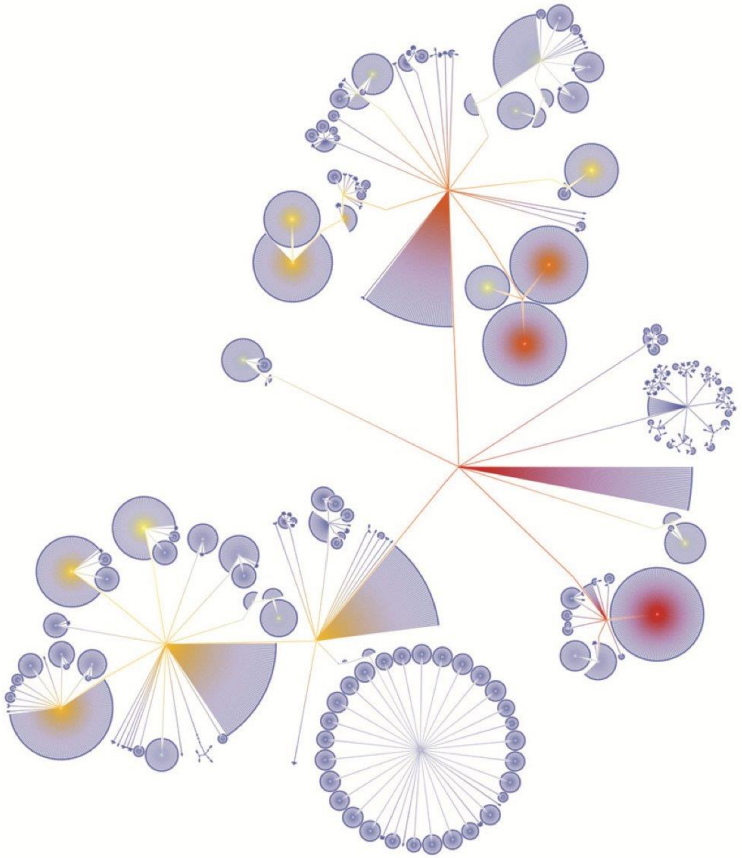


A tree of a few hundred nodes with a spline radial layout. The depth of the tree is encoded as distance away from the center of the circle. The links are drawn as smooth spline curves rather than lines.

Connection: Link Marks 2/5



A tree of 5161 nodes as a rectangular horizontal node-link diagram.
The edges are colored with a purple to orange.
Continuous colormap based on Strahler centrality metric.



Same tree with the BubbleTree algorithm.
BubbleTree is a radial layout where subtrees are laid out in full circle.

Connection: Link Marks 3/5

- **Networks are commonly represented as node-link diagrams, using connection.**
- **Nodes that are directly connected by a single link are perceived as having the tightest grouping, while nodes with a long path of multiple hops between them are less closely grouped.**
- **The number of hops is a network-oriented way to measure distances.**

Connection: Link Marks 4/5

- **Node-link diagrams in general are well suited for tasks that involve understanding the network topology:**
  - **A. finding all possible paths from one node to another**
  - **B. finding the shortest path between two nodes**
  - **C. finding all the adjacent nodes one hop way from a target node,**
  - **D. finding nodes that act as a bridge between two components of the network that would otherwise be disconnected**

Connection: Link Marks 5/5

- **Node-link diagrams are most often laid out within a two-dimensional planar region.**
- **Three-dimensional layout is possible, but due to occlusion and perceptual problems it is rarely an effective choice.**
- **Even though for a planar layout, if the node-link diagram is not planar, there can be many links crossing each other.**

Example: Force-Directed Placement 1/13

- **One of the most widely used idioms for node-link layout using connection marks is force-directed placement.**
- **Force-directed network layout idioms do not directly use spatial position to encode attributes.**
- **The algorithms try to minimize the number of distracting artifacts such as *edge crossings* and *node overlaps* so that the spatial location of the elements is a side effect of the computation rather than directly encoding attributes.**

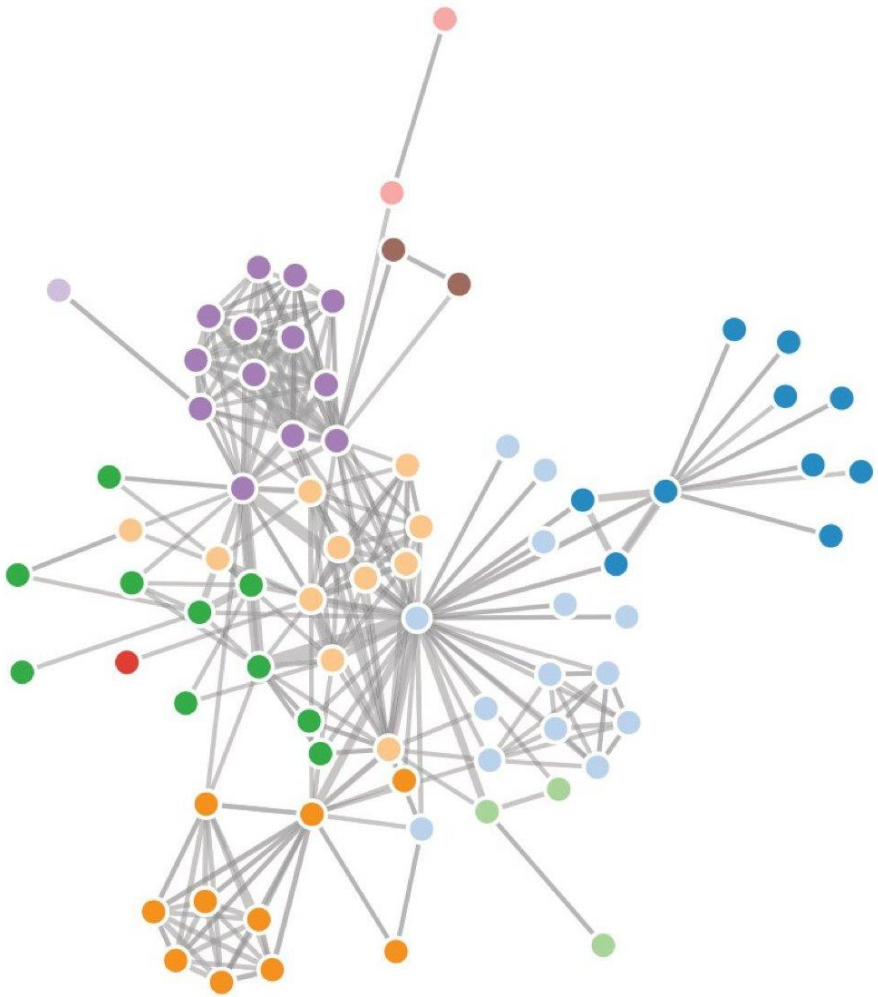
Example: Force-Directed Placement 2/13

- **Force-directed graph drawing algorithms assign forces among the set of edges and the set of nodes of a graph drawing.**
- **Spring-like *attractive* forces based on Hooke's law are used to attract pairs of endpoints of the graph's edges towards each other, while simultaneously *repulsive* forces based on Coulomb's law are used to separate all pairs of nodes.**

Example: Force-Directed Placement 3/13

- **In equilibrium states for this system of forces, the edges tend to have uniform length (because of the spring forces), and nodes that are not connected by an edge tend to be drawn further apart (because of the repulsion).**
- **Edge attraction and vertex repulsion forces may be defined using functions that are not based on the physical behavior of springs and particles. Some force-directed systems use springs whose attractive force is logarithmic rather than linear.**

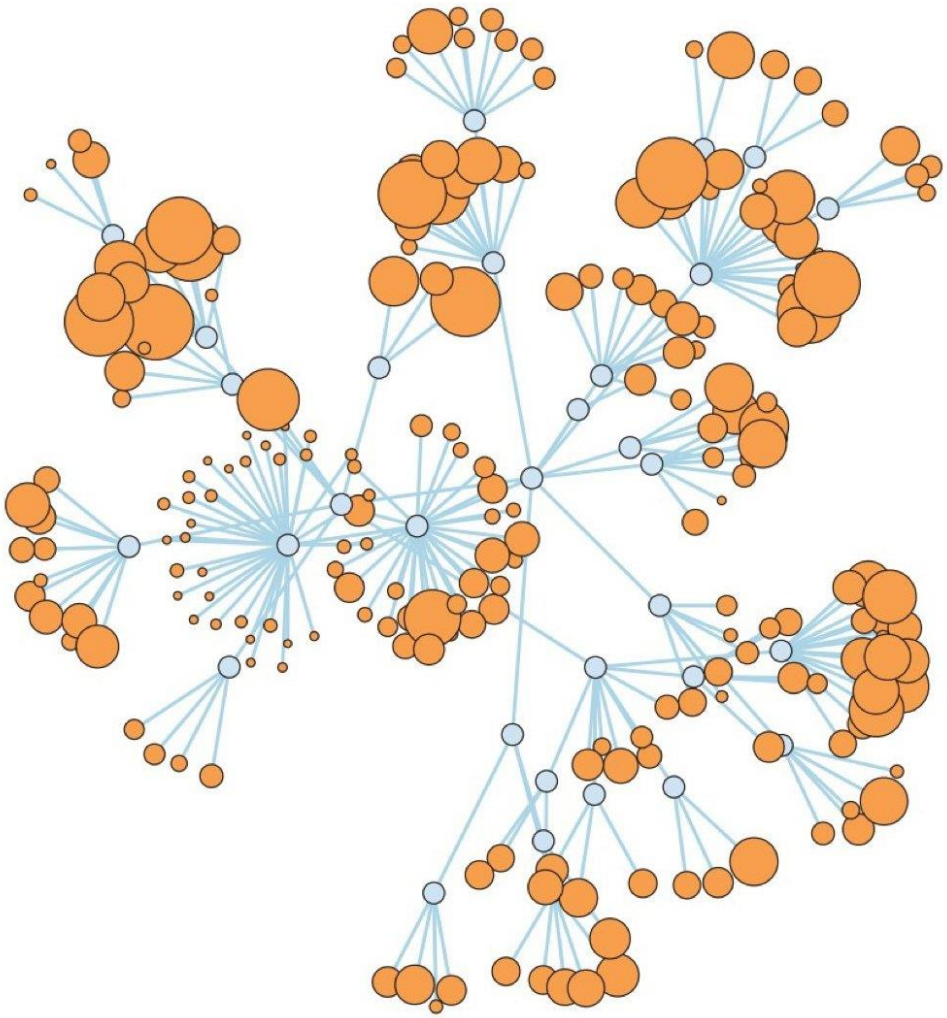
Example: Force-Directed Placement 4/13



This small network of 75 nodes is shown with the force-directed placement layout.

Size coding is done with link attributes.

Example: Force-Directed Placement 5/13



This large network is drawn with the force-directed placement layout.

However, size coding is done with node attributes.

Example: Force-Directed Placement 6/13

- **With the force-directed placement, spatial position does not directly encode any attributes of either nodes or links.**
- **In fact, the placement algorithm uses it indirectly.**
- **A tightly interconnect group nodes with many links between them will often tend to form a visual clump, so spatial proximity does indicate grouping through a strong perceptual cue.**
- **But, nodes may be pushed near each other because they were repelled from elsewhere, not because they are closely connected.**

Example: Force-Directed Placement 7/13

- **With the force-directed placement, proximity is sometimes meaningful but sometimes arbitrary.**
- **This ambiguity can mislead the user.**
- **This situation is a specific example of the general problem that occurs in all idioms where spatial position is *implicitly* chosen rather than *deliberately* used to encode information.**

Example: Force-Directed Placement 8/13

- **There are a few weaknesses of force-directed placement layout.**
  - **A. Nondeterministic layout**
  - **B. Scalability**
  - **C. Very brittle**

Example: Force-Directed Placement 9/13

- **Weakness #1 Non-Deterministic Layout**: **The layouts are often *nondeterministic*, meaning that they will look different each time the algorithm is run, rather than *deterministic* approaches that yield an identical layout each time for a specific dataset.**
- **The major problem with nondeterministic visual encoding is that spatial memory cannot be exploited across different runs of the algorithm.**

Example: Force-Directed Placement 10/13

- **Weakness #1 Non-Deterministic Layout**:
- **Region-based identification such as "the stuff is in the upper left corner" are not useful because the items placed in that region may change between runs.**
- **The randomness can lead to different proximity relationships each time, where the distances between the nodes reflect the randomly chosen initial positions rather than the intrinsic structure of the network.**

Example: Force-Directed Placement 11/13

- **Weakness #2 Scalability**:
- **The scalability issue can happen in terms of the visual complexity of the layout and the time required to compute it.**
- **With a few hundred or more nodes, the layout can quickly degenerate into a hairball of visual clutter, where the tasks of each path following or understanding overall structure relationships become very difficult.**
- **With thousands of nodes, it is virtually impossible.**

Example: Force-Directed Placement 12/13

- **Weakness #3 Being Brittle**:
- **Many force-directed placement algorithms are very brittle.**
- **These algorithms are based on optimization algorithms that minimize some functions. These algorithms may get stuck in *local* minimum rather than the *global* one.**
- **These algorithms usually involve the tweaking of many parameters in order to improve the layout of a particular dataset, but different settings are required for a different dataset.**

Example: Force-Directed Placement 13/13

Idiom	Force-Directed Placement
Data Type	Network.
Encoding	Point marks for nodes, connection marks for links.
Task	Explore topology, locate paths.
Scale	Nodes (N): dozens/hundreds Links (L): hundreds. Node/Link density: $L < 4*N$.

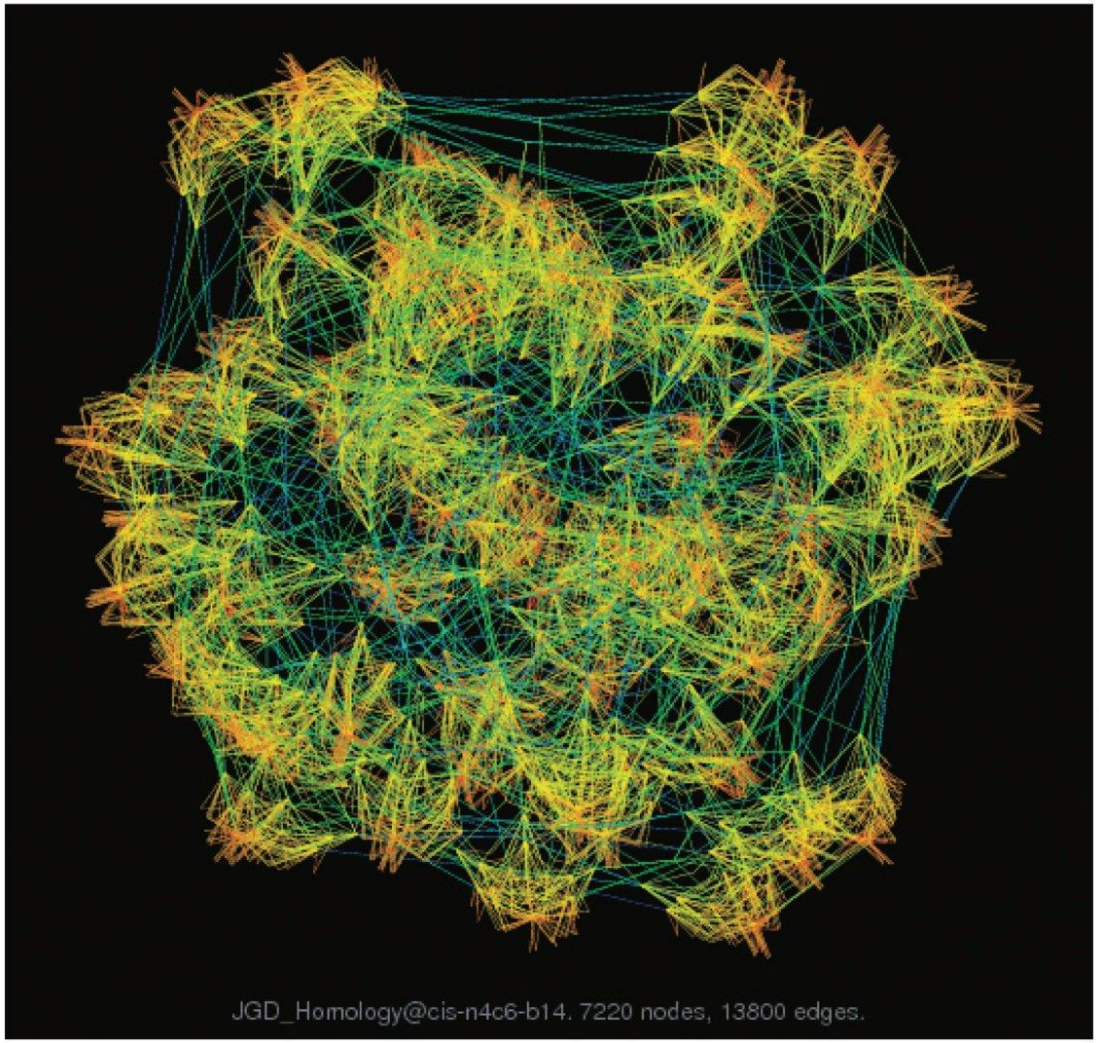
Multi-Level Network 1/2

- **Many recent approaches to scalable network drawing are multi-level network idioms, where the original network is augmented with a derived cluster hierarchy to form a compound network.**
- **The cluster hierarchy is computed by coarsening the original network into successively simpler networks that nevertheless attempt to capture the most essential aspects of the original structure.**

Multi-Level Network 2/2

- **The simplest version of the network is laid out first. Then, improve the layout with the more and more complex versions.**
- **In this way, both the speed and quality of the layout can be improved.**
- **These approaches do better at avoiding the local minimum problem.**

Example: Multi-Level Scalable Force-Directed Placement 1/3

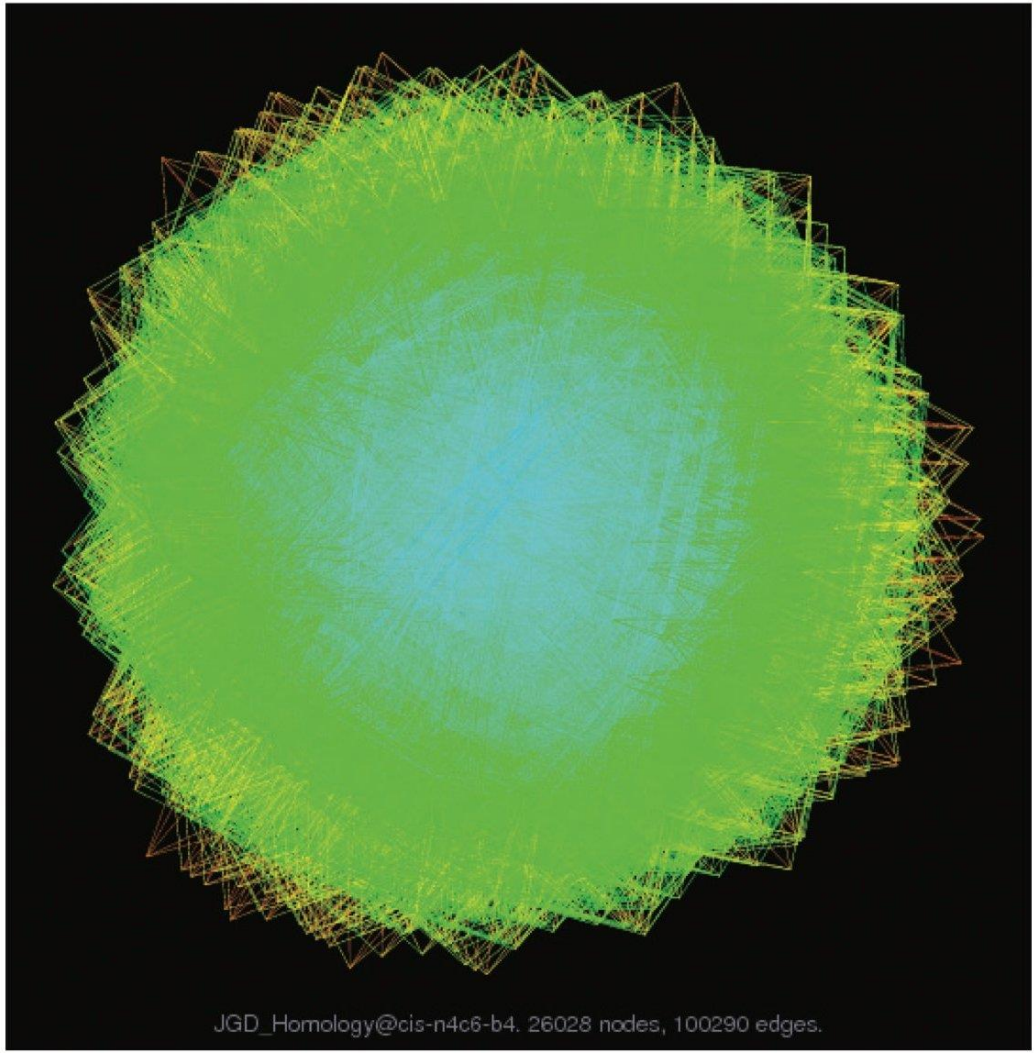


This is a network of 7220 nodes and 13800 edges using a multi-level scalable force-directed placement algorithm.

The edges are colored by length.

Significant clutter structure is visible in the layout, where the dense clusters with short orange and yellow edges can be distinguished from the long blue and green edges.

Example: Multi-Level Scalable Force-Directed Placement 2/3



This multi-level scalable force-directed placement algorithm still has its limit.

This is a network of 26,028 nodes and 100,290 edges, the same algorithm fails to show any discernible structures.

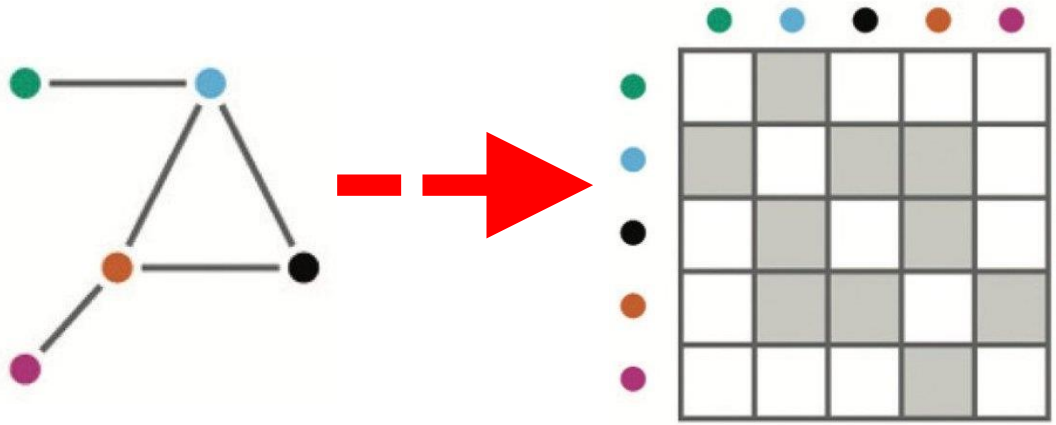
The large number of edge-crossings leads to overwhelming visual clutter caused by occlusion.

Example: Multi-Level Scalable Force-Directed Placement 3/3

Idiom	Multi-Level Force-Directed Placement
Data Type	Network.
Derived Data Type	Cluster hierarchy atop original network.
Encoding	Point marks for nodes, connection marks for links.
Task	Explore topology, locate paths and clusters.
Scale	Nodes (N): 1000-10,000. Links (L): 1000-10,000. Node/Link density: $L < 4*N$.

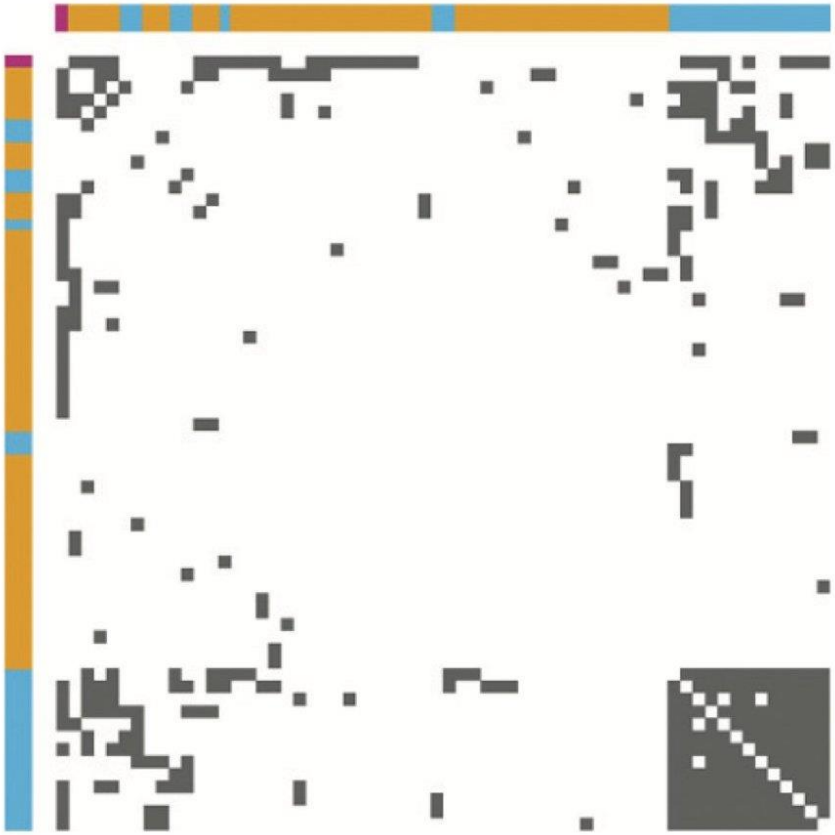
Matrix Views

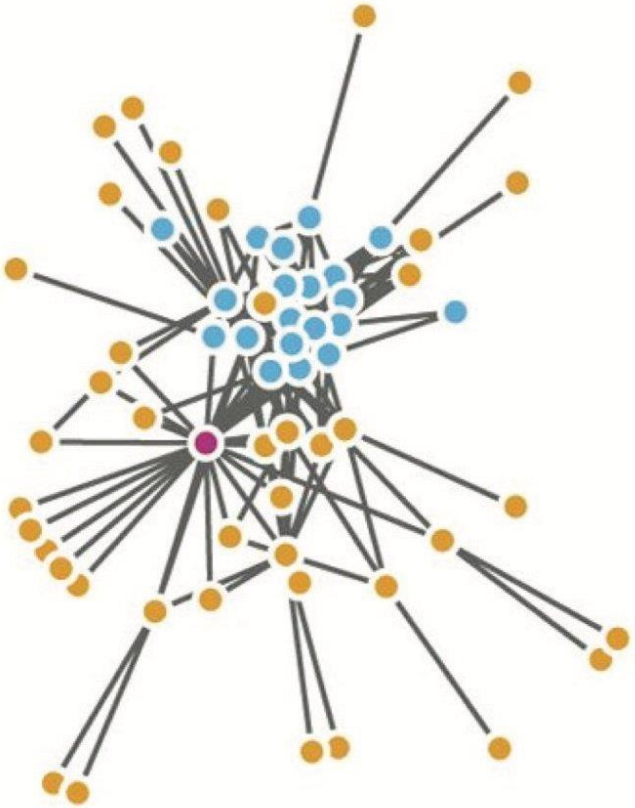
- Network data can also be encoded with a matrix view by deriving a table from the original network data.
- One of the most commonly seen table is the **adjacency matrix**. Each node has a row and a column, and if two nodes has a link, set the corresponding entry to 1.



Example: Adjacency Matrix View 1/4

- The following shows a network and its corresponding adjacency matrix view:





Example: Adjacency Matrix View 2/4

- **Additional information about another attribute is often encoded by coloring matrix cells.**
- **The possibility of *size* coding matrix cells is limited by the number of available pixels per cell: only a few levels would be distinguishable between the largest and smallest cell size.**
- **Network matrix views can also show *weighted* networks, where each link has an associated quantitative value attribute, by encoding with an *order* channel such as *luminance* or *size*.**

Example: Adjacency Matrix View 3/4

- **For undirected networks where links are symmetric, the corresponding adjacency matrix is also symmetric and only half of the matrix needs to be shown, above or below the diagonal.**
- **Matrix views of networks can achieve very high information density, up to a limit of one thousand nodes and one million edges,.**
- **This is similar to cluster heatmaps.**

Example: Adjacency Matrix View 4/4

Idiom	Adjacency Matrix View
Data Type	Network.
Derived Data Type	Table: network nodes as keys, link between two nodes as values.
Encoding	Area marks in 2D matrix alignment.
Scale	Nodes: 1000. Links: one million.

Connection vs. Matrix: 1/7

- **The great strength of node-link layouts is that for sufficiently small networks they are extremely intuitive for supporting many of the abstract tasks that pertain to network data.**
- **They are particularly useful for tasks that rely on understanding the topological structure of the network, such as path tracing and searching local topological neighborhoods a small number of hops from a target node, and also be very effective for tasks such as general overview or finding similar substructures.**

Connection vs. Matrix: 2/7

- **On the other hand, their weakness is that passing a certain limit of network size and link density, they become impossible to read because of occlusion from edge crossing each other and crossing underneath nodes.**
- **The link density of a network is *the number of links compared with the number of nodes*.**
- **Trees have a link density of 1.**
- **The upper limit for node-link diagram effectiveness is a link density around 3 or 4.**

Connection vs. Matrix: 3/7

- **A major strength of matrix views is perceptual scalability for both large and dense networks.**
- **Matrix views completely eliminate the occlusion of node-link views, and are effective even at very high information densities.**
- **Another strength is predictability (i.e., predictable visual space), stability (i.e., adding a new item only causing a small visual change), and support for reordering.**

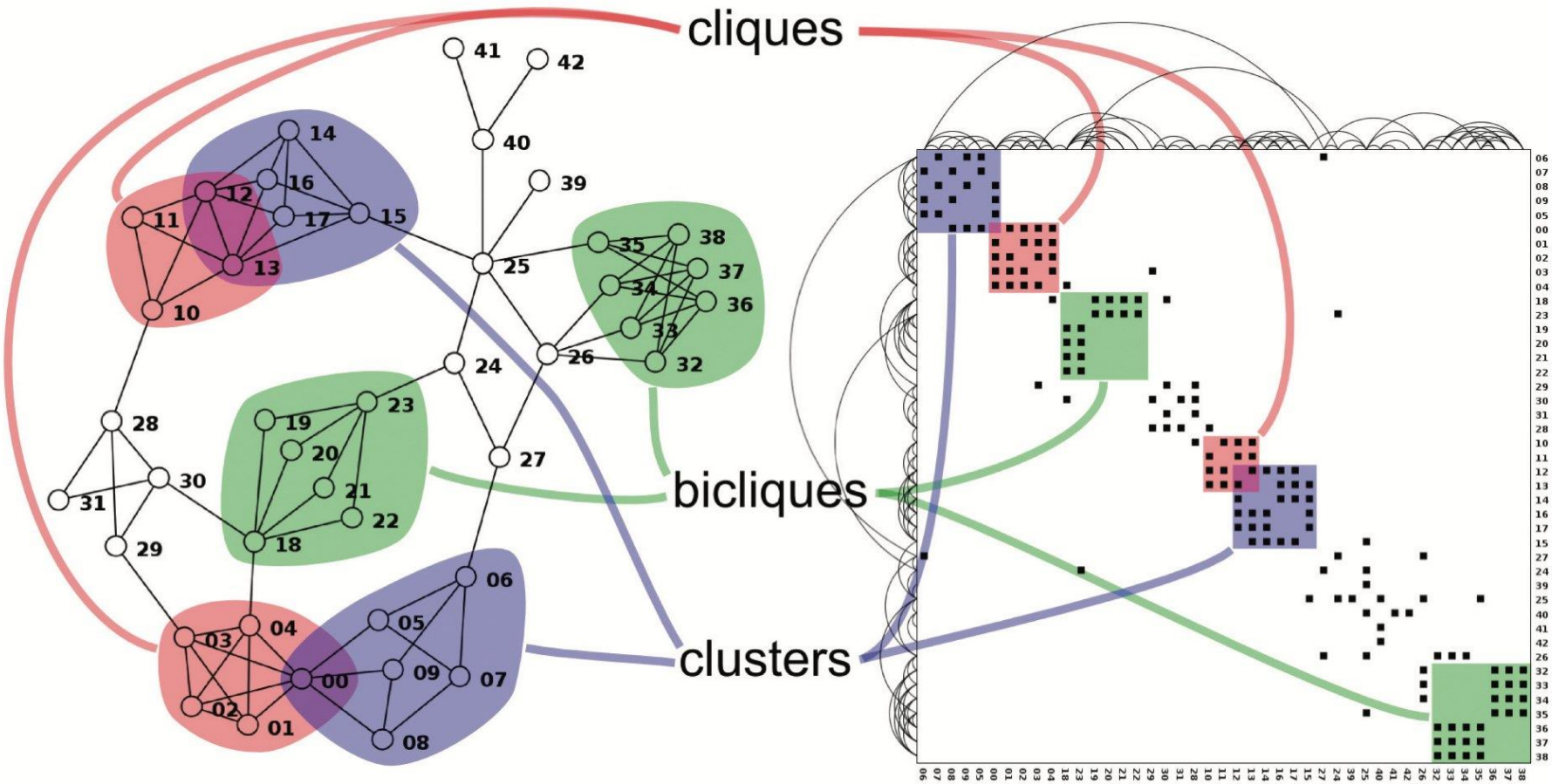
Connection vs. Matrix: 4/7

- **One major weakness of matrix views is unfamiliarity: most users are able to easily interpret node-link views of small networks with the need of training, but they typically need training to interpret matrix views.**
- **With sufficient training, many aspects of matrix views can become salient. For example, finding specific types of nodes or node groups are supported by both views.**

Connection vs. Matrix: 5/7

- **Let us look at an example:**
  - **A. A clique of a network is a subset of nodes such that every pair of nodes in this subset is connected with a link.**
  - **B. A biclique is asubset in which links connect each node in one subset with one in another.**
- **Identifying cliques and bicliques are easy with the node-link layout.**
- **We can find cliques and bicliques by searching for patterns in matrix views.**

Connection vs. Matrix: 6/7



Connection vs. Matrix: 7/7

- **The most crucial *weakness* of matrix views is their lack of support for investigating topological structures because they show links in a more indirect way.**
- **This weakness is a direct trade-off for their strength in avoiding clutter.**
- **Hybrid multiple-view systems that use both node-link and matrix views are a nice way to combine their complementary strengths.**

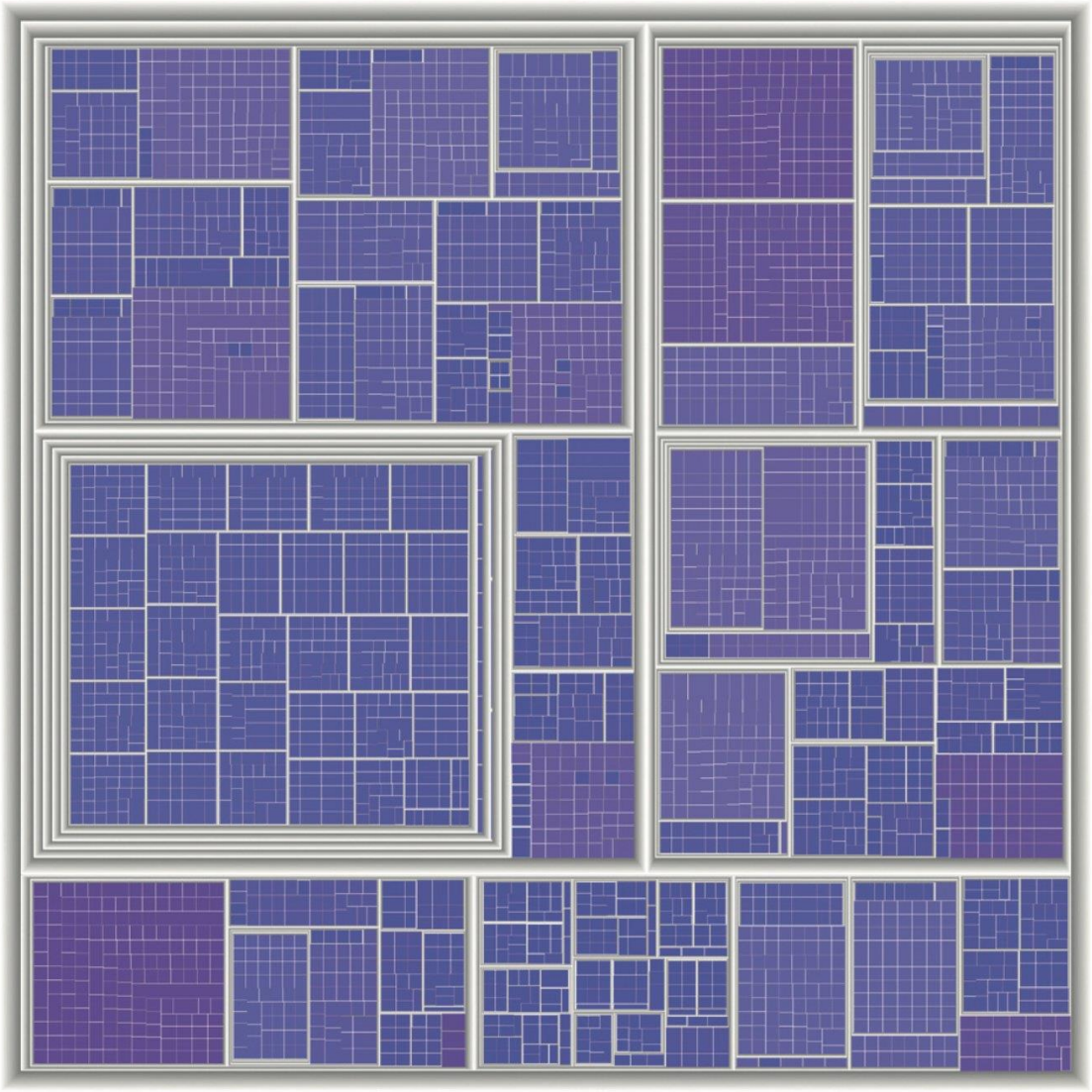
Containment: Hierarchy Marks

- **Containment marks are very effective for showing complete information about hierarchical structure.**
- **Connection marks only show pairwise relationships between two items at once.**

Example: Treemaps 1/4

- The idiom of **treemaps** is an alternative to node-link tree drawing, where the hierarchical relationships are shown with containment rather than connection.
- All of the children of a tree node are enclosed within the area allocated to that node, creating a nested layout.
- The size of the nodes is mapped to some attribute of the node.

Example: Treemaps 2/4



This is a treemap of a 5161 nodes computer file system.

Node size encodes file size.

Example: Treemaps 3/4

- **Containment marks are not as effective as the pairwise connection marks for tasks focused on topological structure (e.g., tracing paths through the tree).**
- **They are good for tasks for understanding attribute values at the leaves of the tree, and are very effective for spotting the outliers of very large attribute values (e.g., large files).**
- **Treemaps are commonly used when hierarchies are shallow rather than deep.**

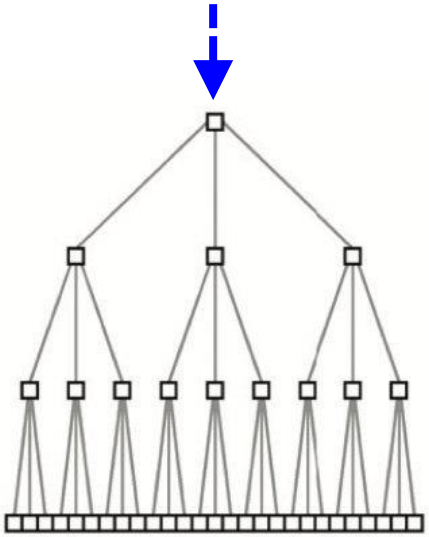
Example: Treemaps 4/4

Idiom	Treemaps
Data Type	Tree.
Encoding	Area marks and containment, with rectilinear layout.
Task	Query attributes at leaf nodes
Scale	Leaf Nodes: one million. Links: one million.

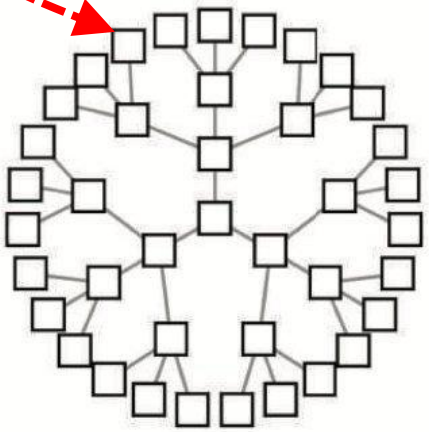
Hierarchy Marks (Cont.): 1/2

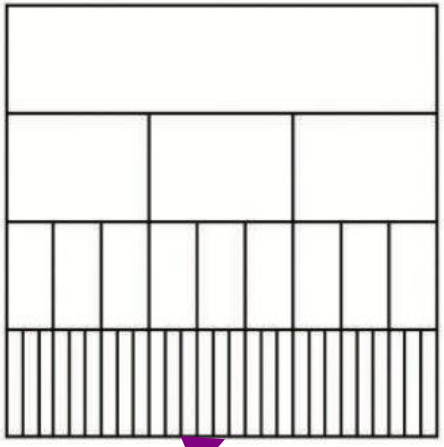
- Several visual encoding idioms for tree are available as follows:

Rectilinear vertical node-link

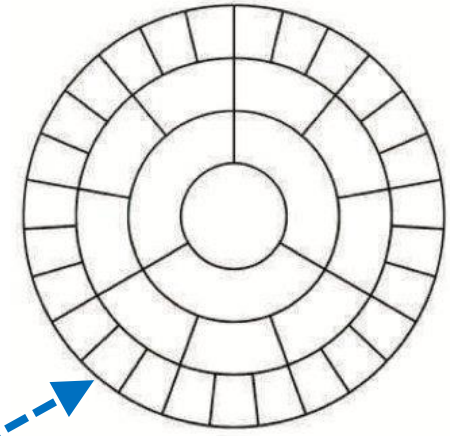


Radial node-link, using connection to show link relationship, with radial depth spatial position showing tree depth and radial angular position showing sibling order





Icicle with vertical spatial position and size showing tree depth and horizontal spatial position showing sibling order



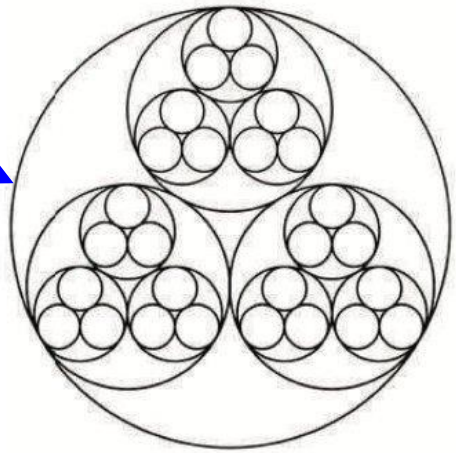
Concentric circles, with radial depth spatial position and size showing tree depth and radial angular spatial position showing link relationship and sibling

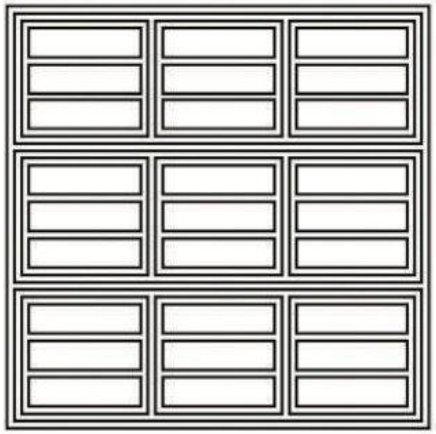
Hierarchy Marks (Cont.): 2/2

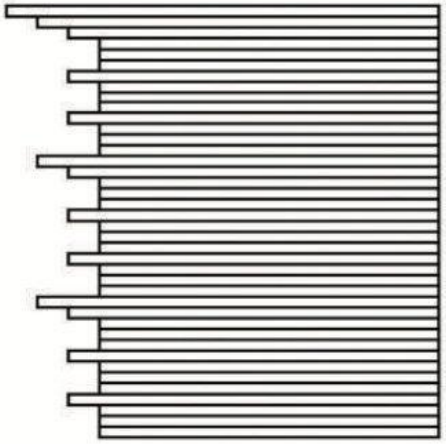
- **Several visual encoding idioms for tree are available as follows:**

Nested circles, using radial containment, with nesting level and size showing tree depth

Indented outline, with horizontal spatial position showing tree depth and link relationships and vertical spatial position showing sibling order







Treemap, using rectilinear containment, with nesting level and size showing tree depth

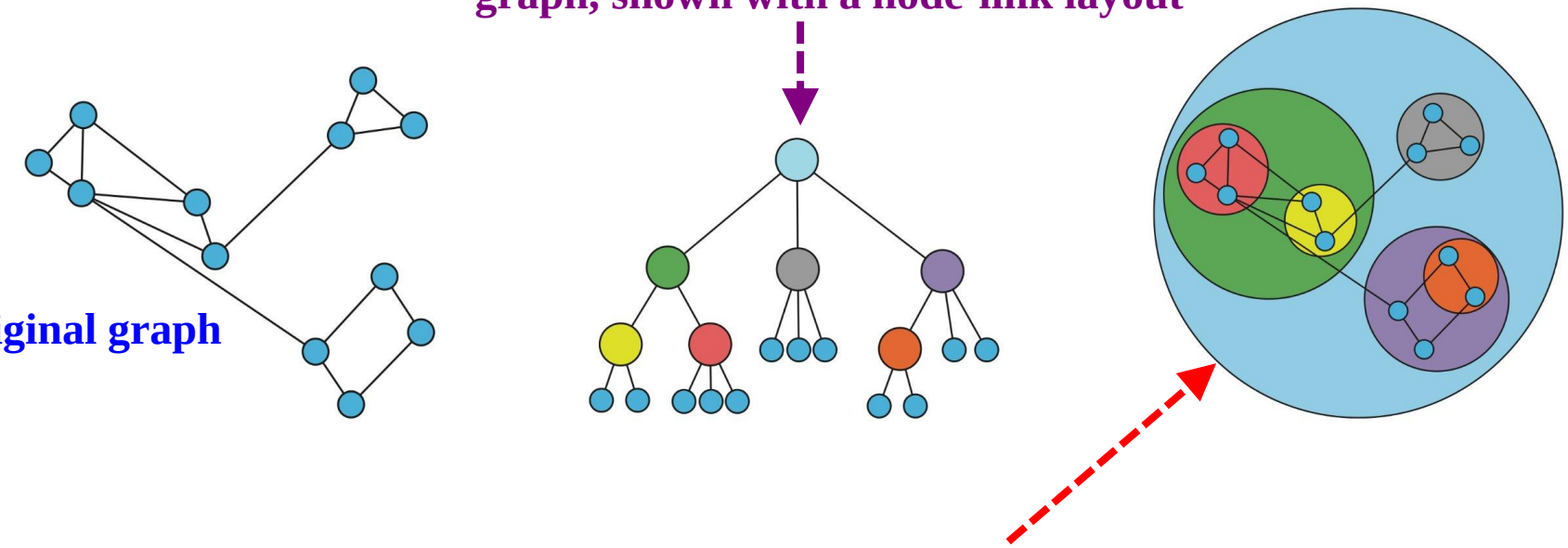
Example: GrouseFlocks 1/3

- **The containment design is usually only used when there is a hierarchical structure (i.e., trees).**
- **It is possible that we have a compound network, which is the combination of a network and a tree.**
- **In other words, in addition to the base network, there is also a cluster hierarchy that groups the nodes hierarchically.**

Example: GrouseFlocks 2/3

- **In the GrouseFlocks system, users can investigate multiple possible hierarchies that are shown explicitly.**

Cluster hierarchy built on top of the graph, shown with a node-link layout



Original graph

Network encoded using connection, with hierarchy encoded using containment

Example: GrouseFlocks 3/3

Idiom	GrouseFlocks
Data Type	Networks.
Derived Data Type	Cluster hierarchy on top of the original network.
Encoding	Connection marks for original work, containment marks for cluster hierarchy

The End