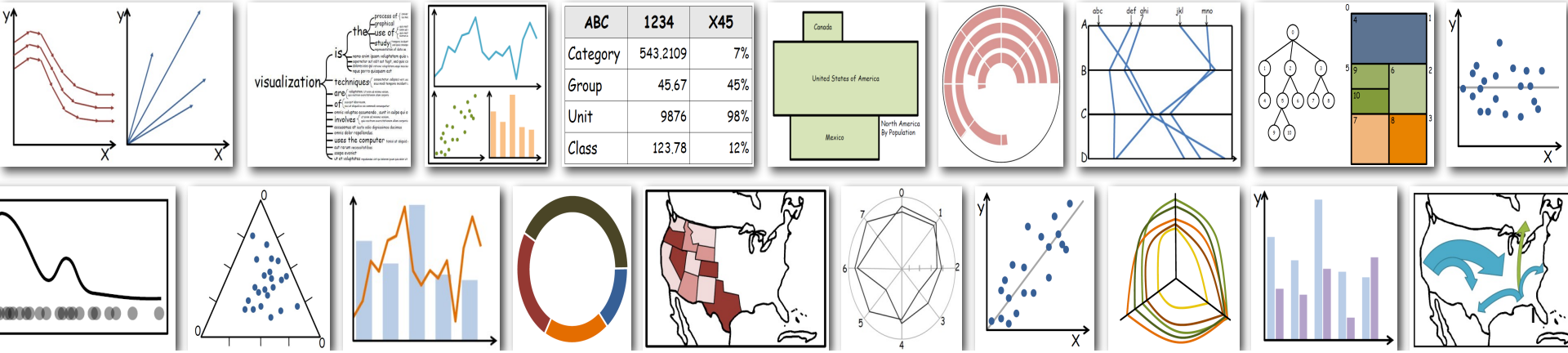
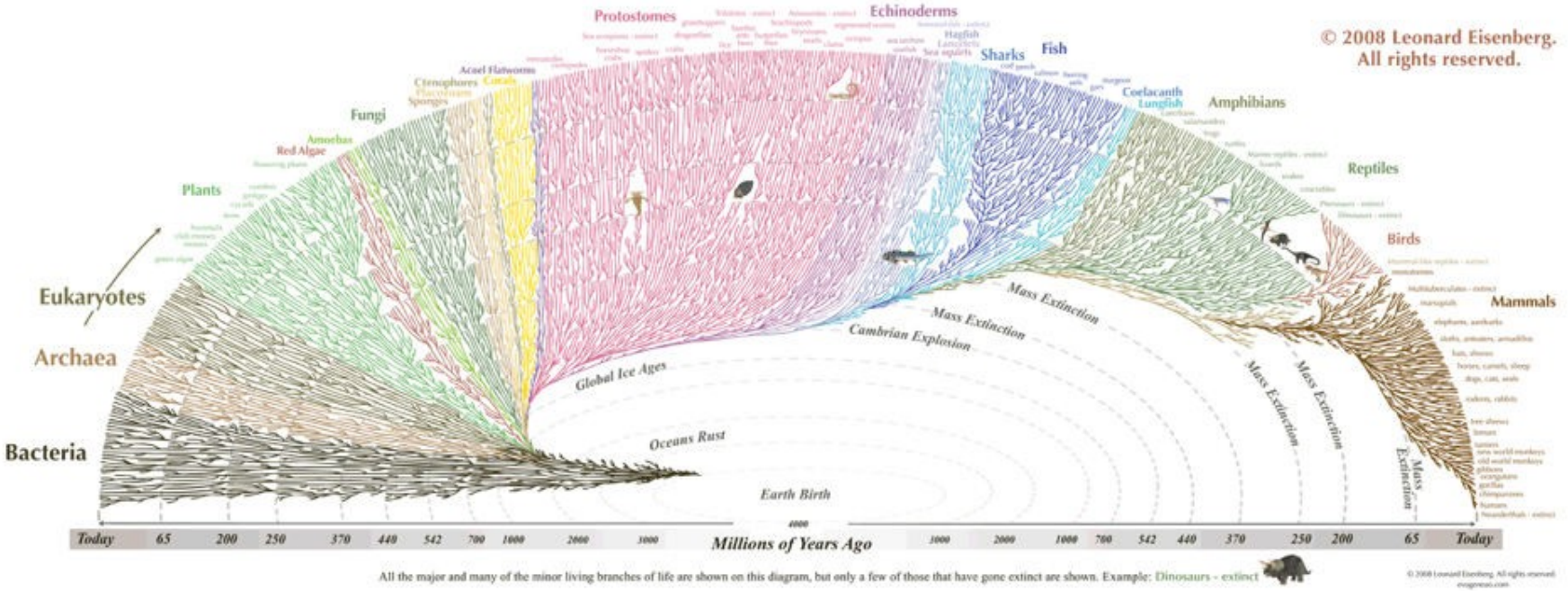


Lecture: Visualizing Trees

DATA ANALYSIS & VISUALIZATION
FALL 2021

Dr. Muhammad Faisal Cheema
FAST NU



- The “tree of life”
  - evolution of species creates branching mechanism and “ancestor-of” relationship

Tree Hierarchy

- Tree relation
  - if a is child of b and a is child of c, then:
    - b is child of c or c is child of b, but not both at the same time
- “Immediate boss is unique”

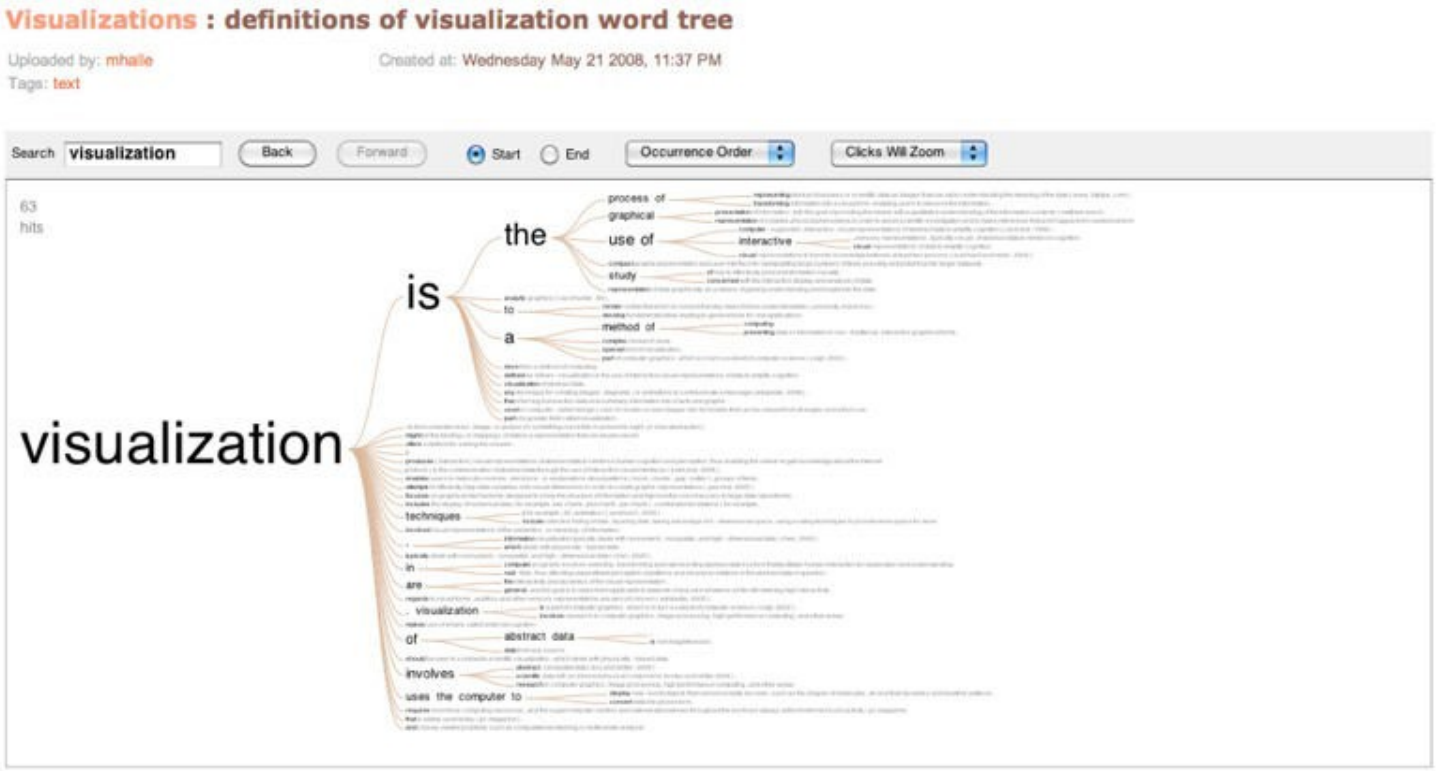
What do we want our drawings to show?

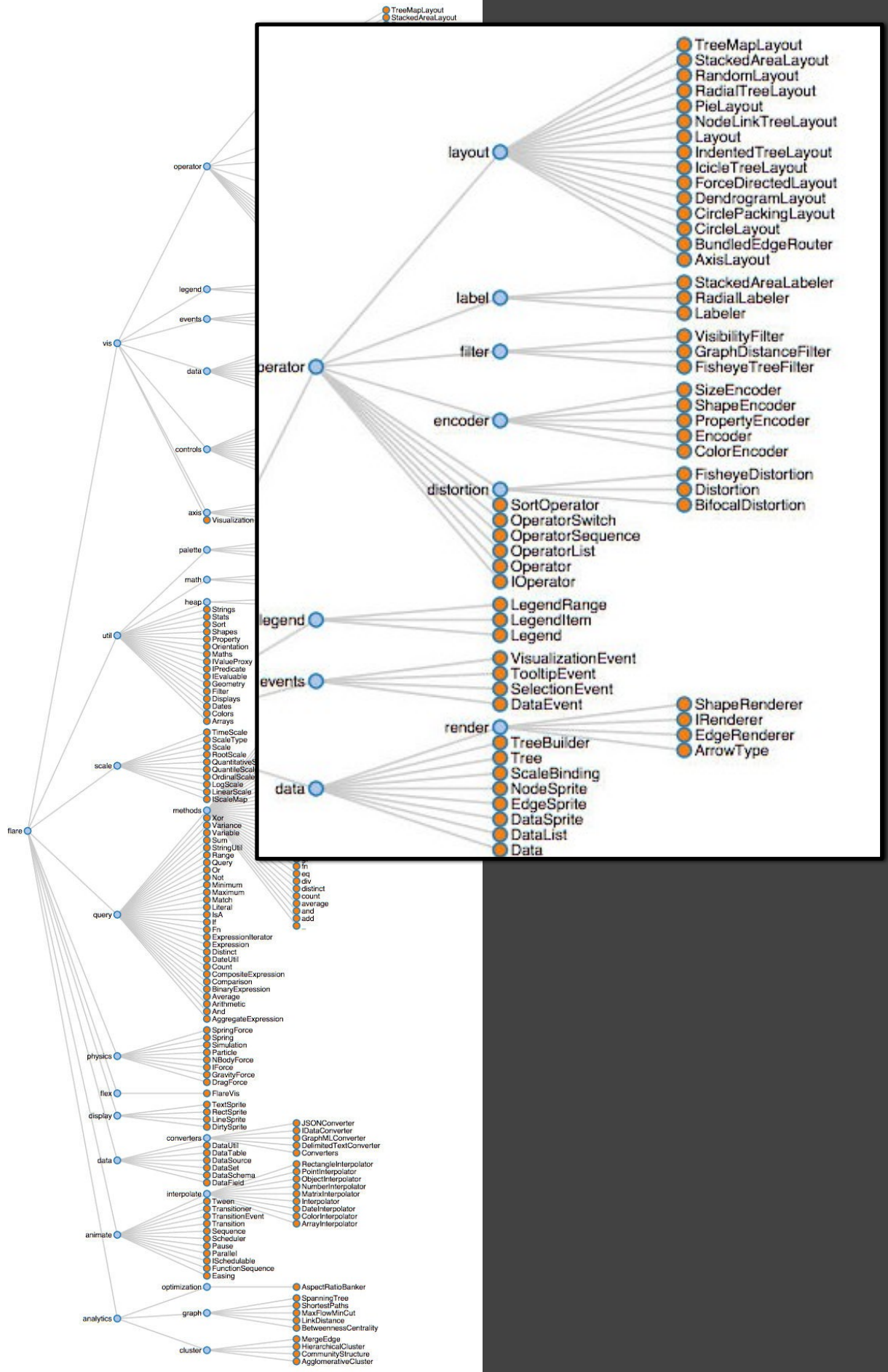
- Who reports to whom
  - … and **who doesn't**
- How big are "sub-organizations"
- …?

Many different ways to visualize trees

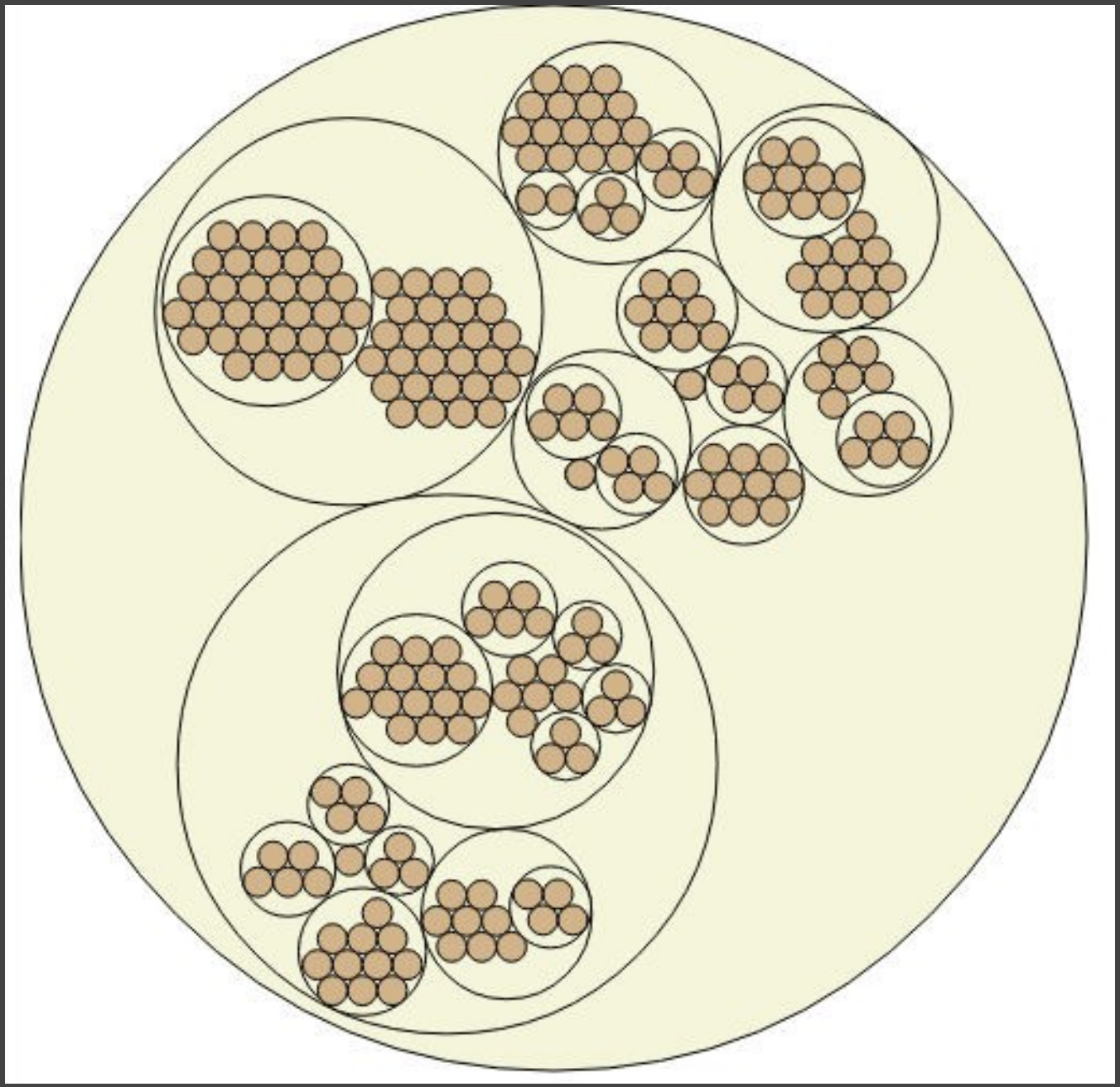
VISUALIZING TREES

WORD TREE

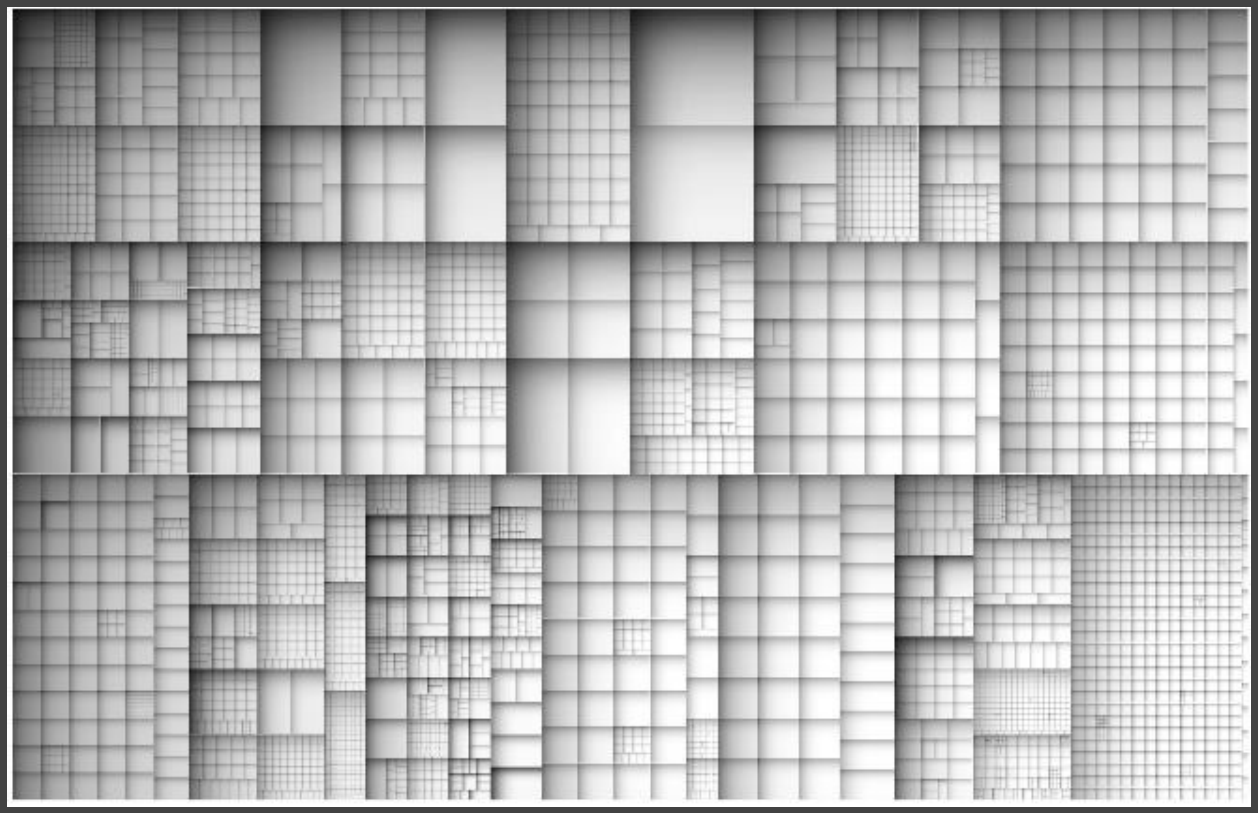

Visualizations : definitions of visualization word tree
Uploaded by: mhalle
Created at: Wednesday May 21 2008, 11:37 PM
Tags: text
Search visualization
Back
Forward
Start
End
Occurrence Order
Clicks Will Zoom
63
hits
visualization
is
the
process of
graphical
use of
interactive
study
to
a
method of
techniques
in
are
visualization
of
abstract data
involves
uses the computer to



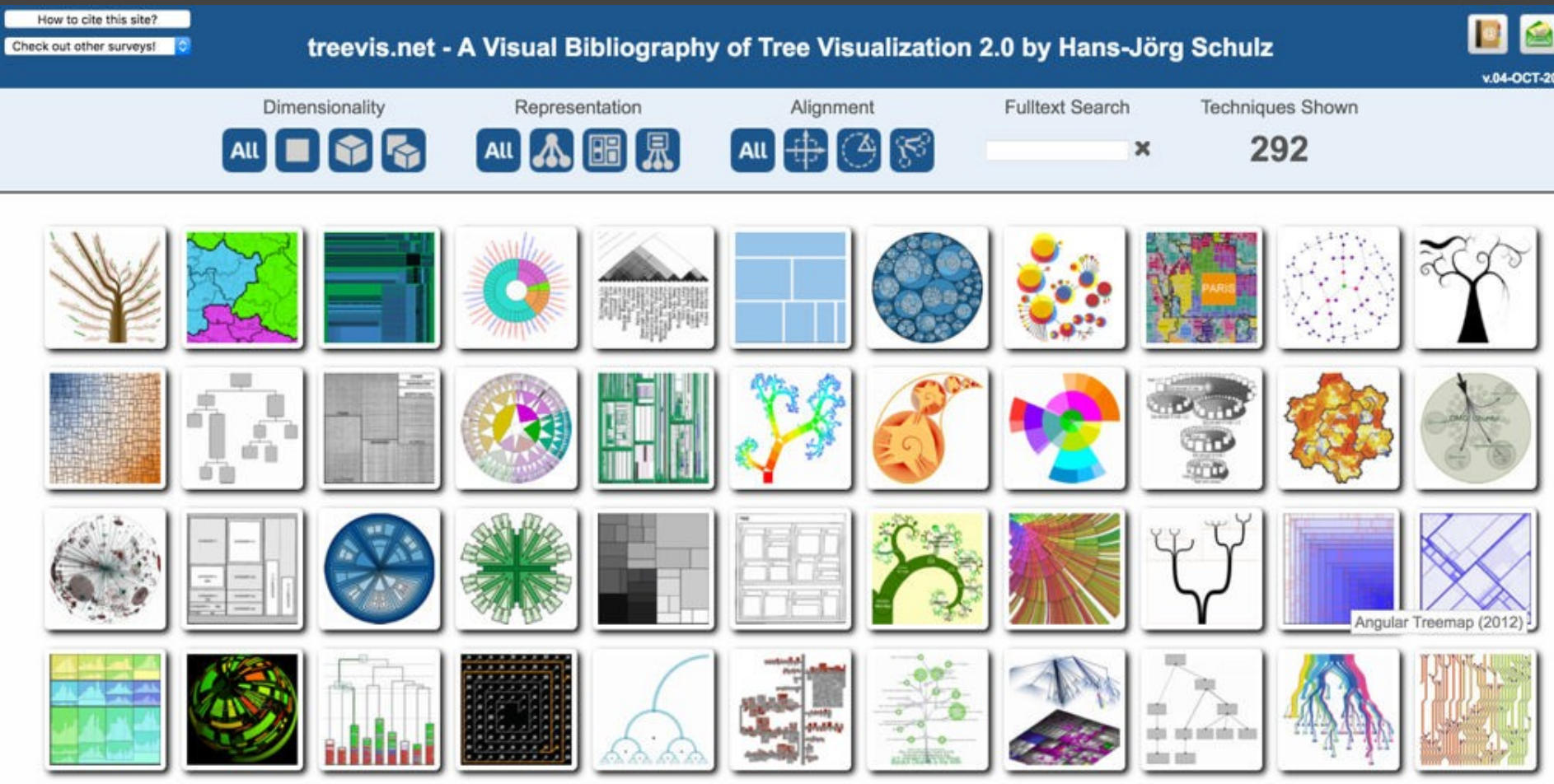
http://homes.cs.washington.edu/~jheer/files/zoo/ex/hierarchies/tree.html



http://jsfiddle.net/VividD/WDCpq/8/



http://www.cs.rug.nl/svcg/SoftVis/ViewFusion



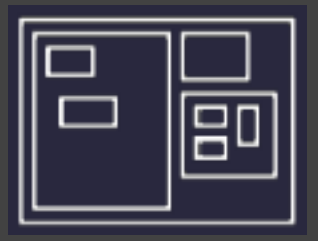
http://treevis.net

Tree Visualization

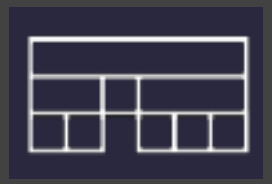
Indentation
Linear list, indentation encodes depth

Node-Link diagrams
Nodes connected by lines/curves

Enclosure diagrams
Represent hierarchy by enclosure



Layering
Relative position and alignment

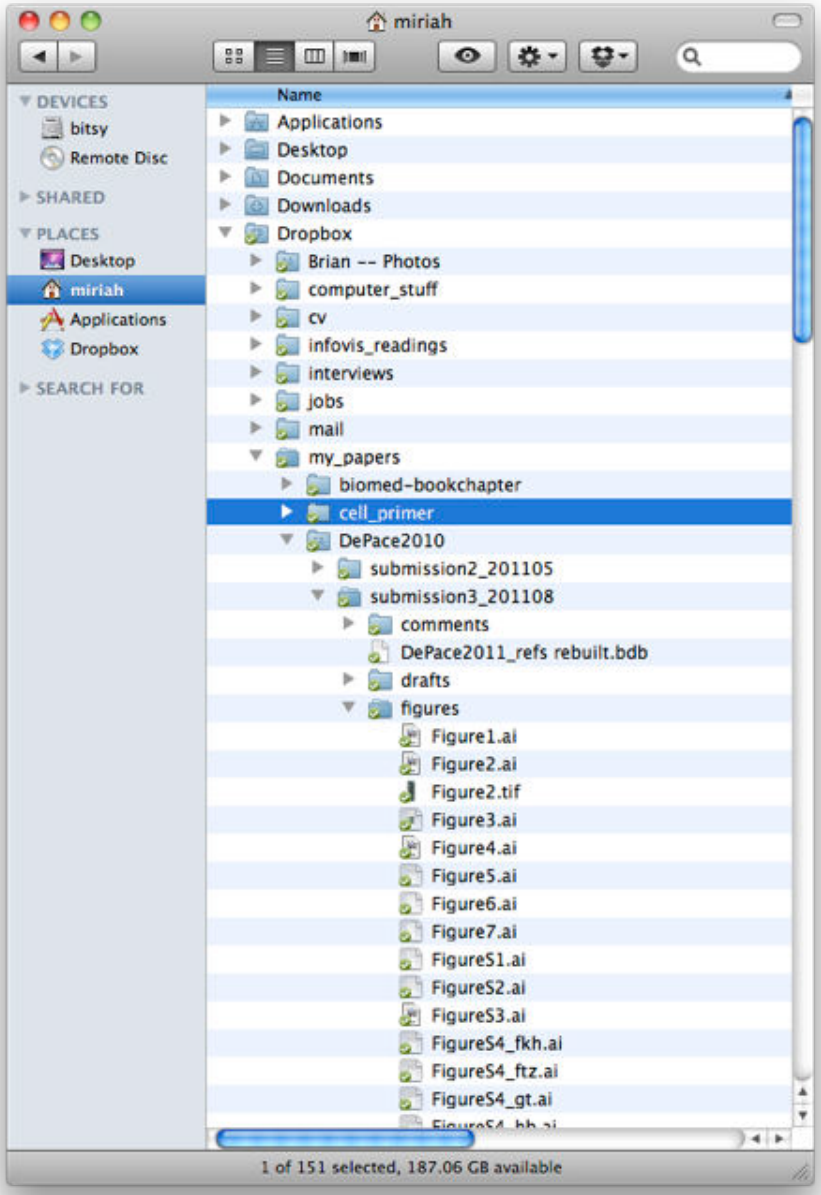


Typically fast: O(n) or O(n log n), interactive layout

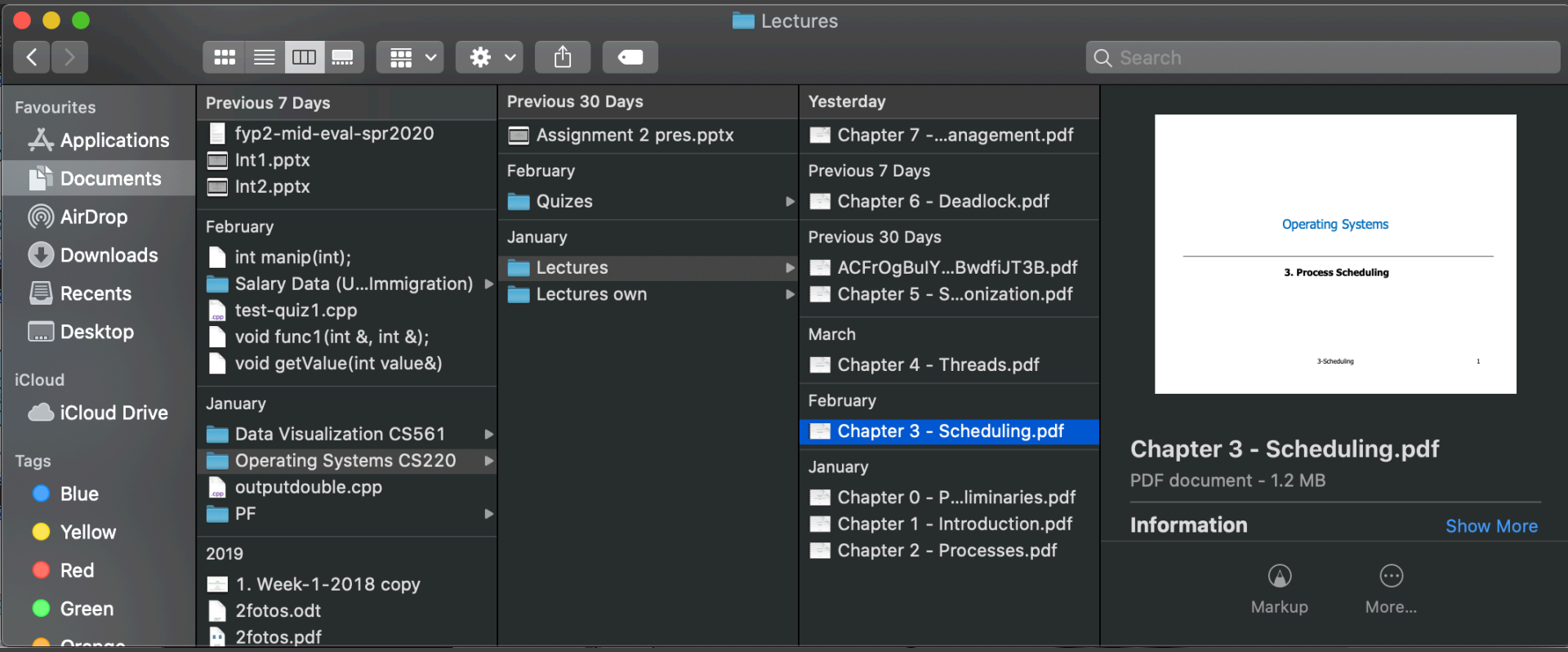
Indentation

INDENTATION

- **place all items along vertically spaced rows**
- **indentation used to show parent/child relationships**
- **commonly used as a component in an interface**
- **breadth and depth contend for space**
- **often requires a great deal of scrolling**



Single-Focus (Accordion) List



Separate breadth & depth along 2D.

Focus on a single path at a time.

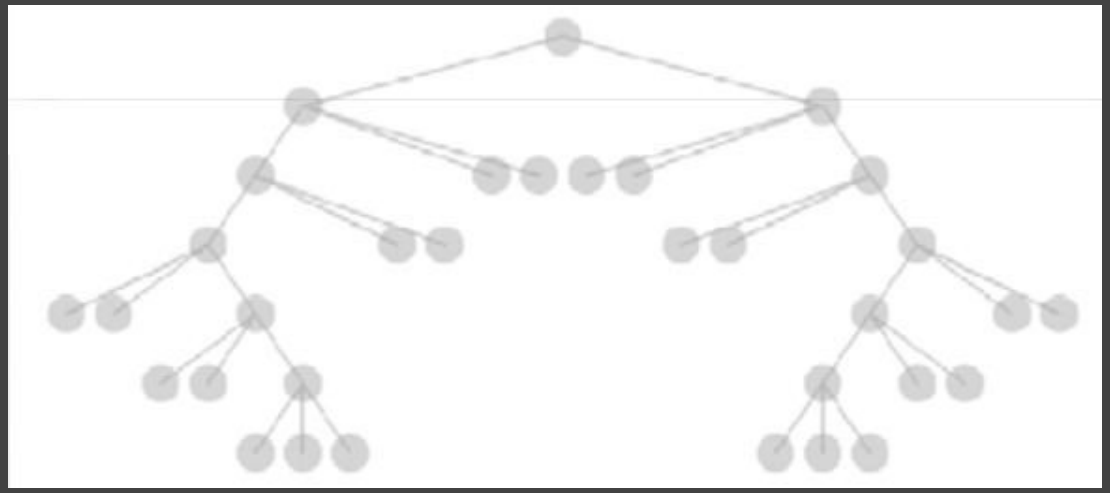
Node link Diagrams

NODE-LINK DIAGRAMS

- **nodes are distributed in space, connected by straight or curved lines**
- **typical approach is to use 2D space to break apart breadth and depth**
- **often space is used to communicate hierarchical orientation**

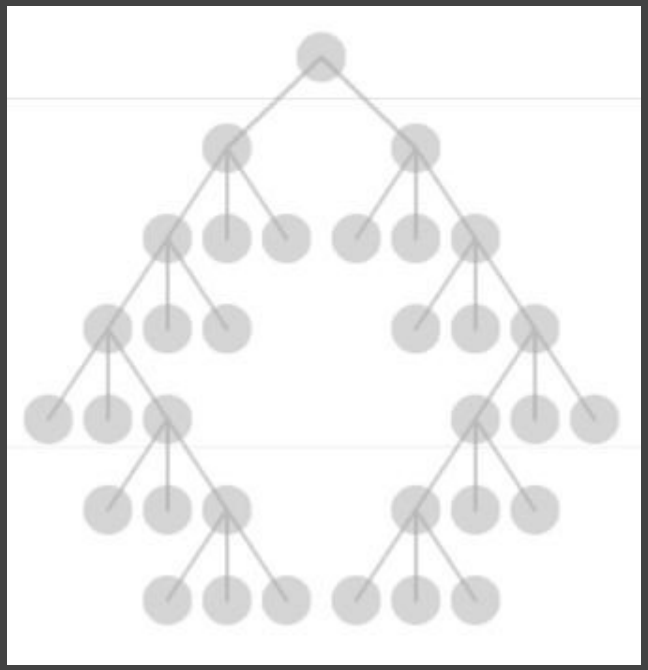
BASIC APPROACH (TOP DOWN LAYOUT)

- **repeatedly divide space for subtrees by leaf count**
  - breadth of tree along one dimension
  - depth along the other dimension
- **problem: exponential growth of breadth**

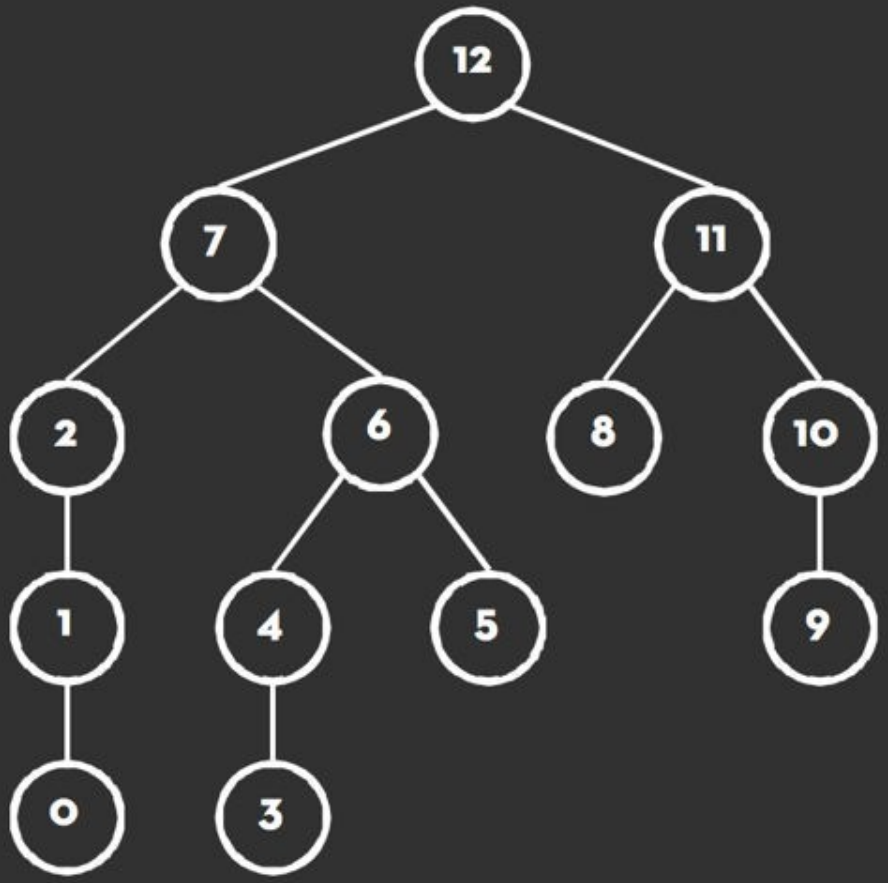


REINGOLD-TILFORD type layouts

- **goal**
  - make smarter use of space
  - maximize density and symmetry
  - Don't waste horizontal space
  - If tree is symmetric, so should be the drawing
- **design concerns**
  - clearly encode depth level
  - no edge crossings
  - isomorphic subtrees drawn identically
  - compact
- **approach**
  - bottom up recursive approach
  - for each parent make sure every subtree is drawn
  - pack subtrees as closely as possible
  - center parent over subtrees



Reingold-Tilford Algorithm



Reingold-Tilford Algorithm

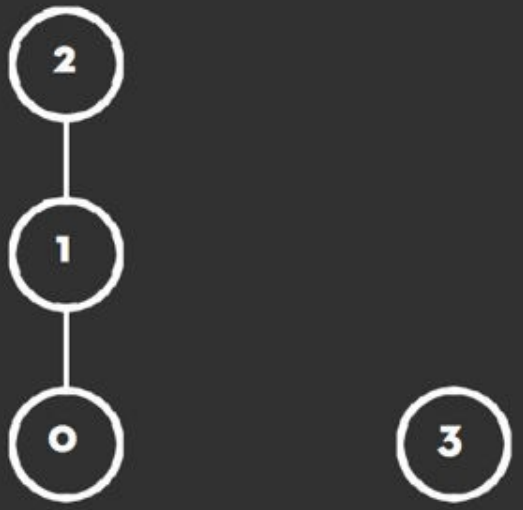
- Bottom-up tree traversal
- y-coord is the depth of the node, x-coords are “locally defined” (so first is arbitrary)
- merge trees
  - push right tree as close as possible to left tree (this is where the contour comes in)
  - position **shifts** saved at each node
  - parent nodes are centered above direct children
- Final top-down pass to convert shifts to positions

Reingold-Tilford Algorithm

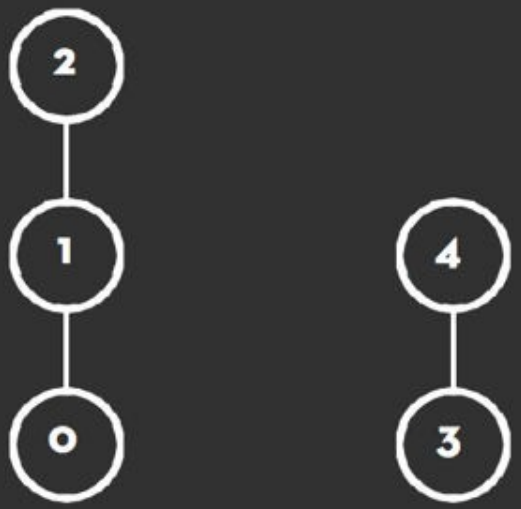
Reingold-Tilford Algorithm

Reingold-Tilford Algorithm

Reingold-Tilford Algorithm

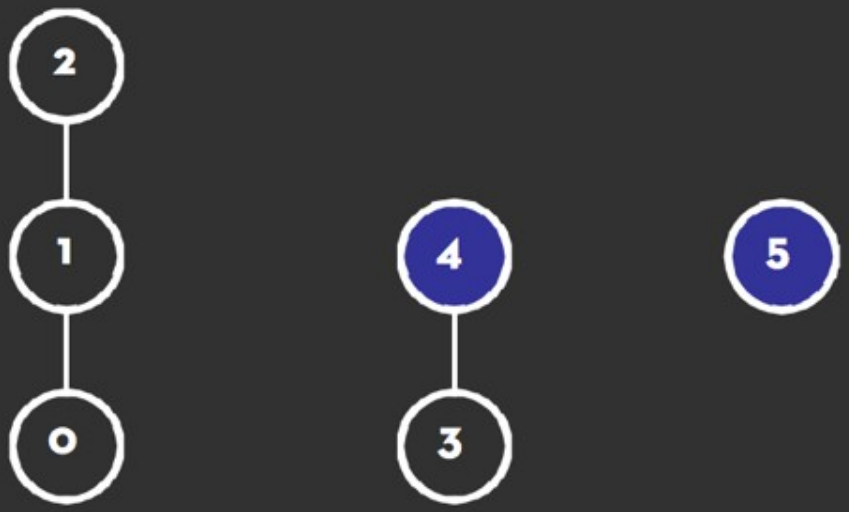


Reingold-Tilford Algorithm

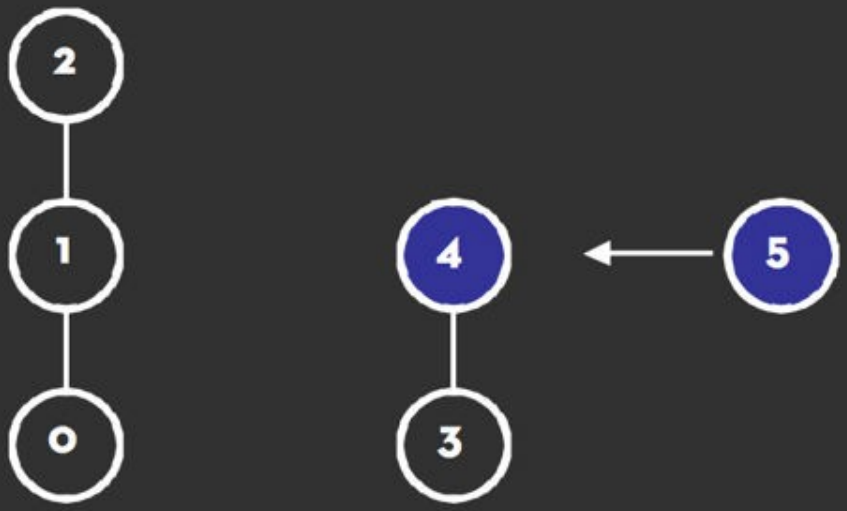


Reingold-Tilford Algorithm

Reingold-Tilford Algorithm

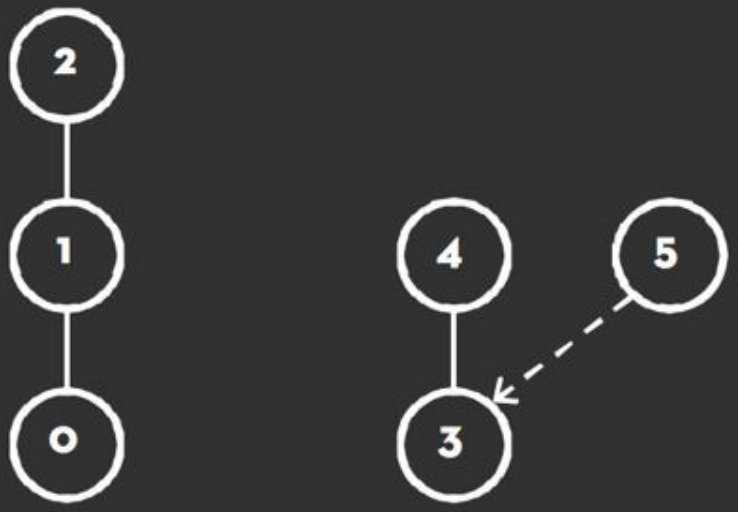


Reingold-Tilford Algorithm

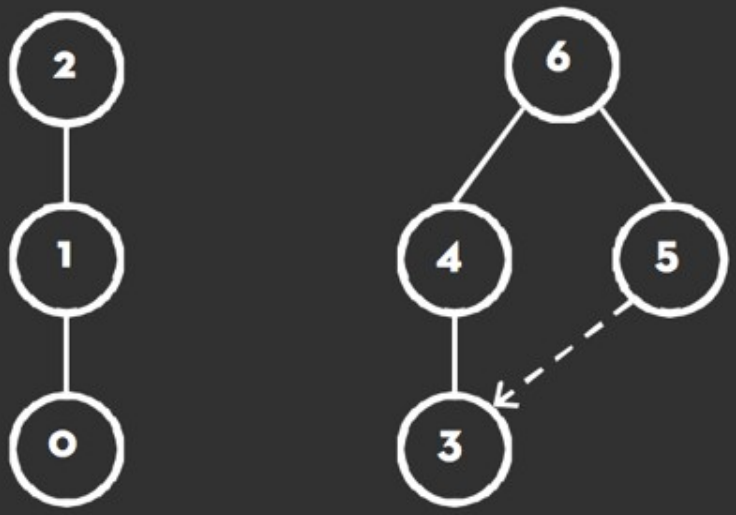


Reingold-Tilford Algorithm

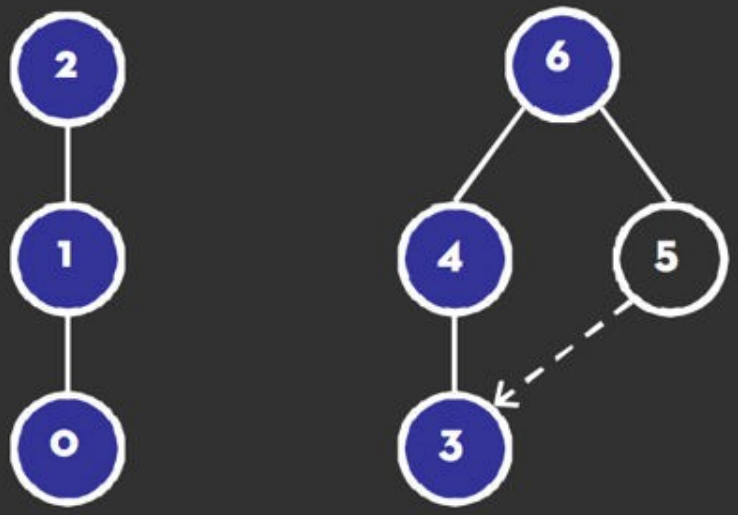
Reingold-Tilford Algorithm



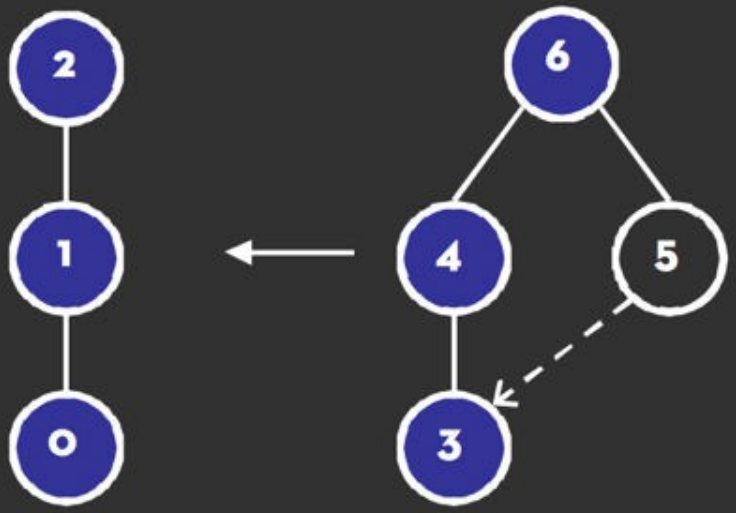
Reingold-Tilford Algorithm



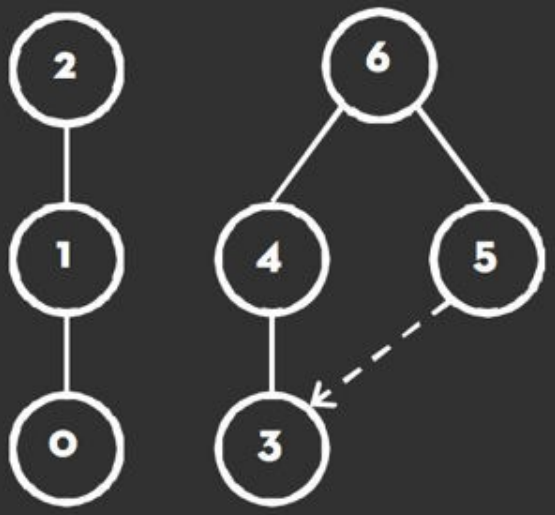
Reingold-Tilford Algorithm



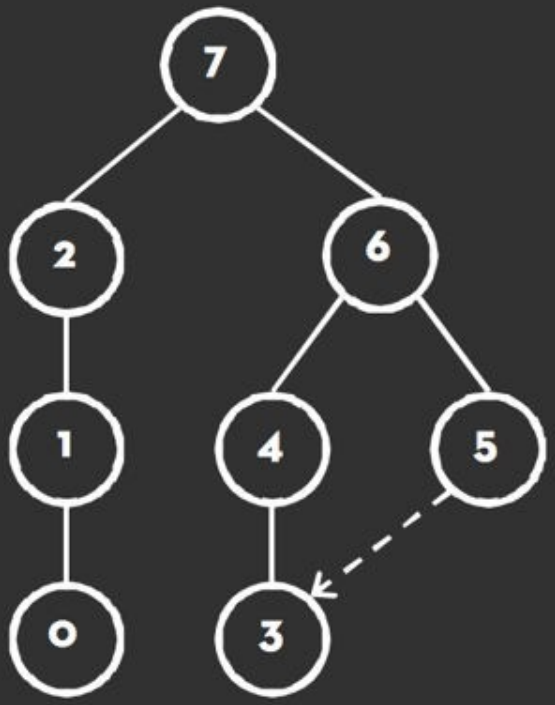
Reingold-Tilford Algorithm



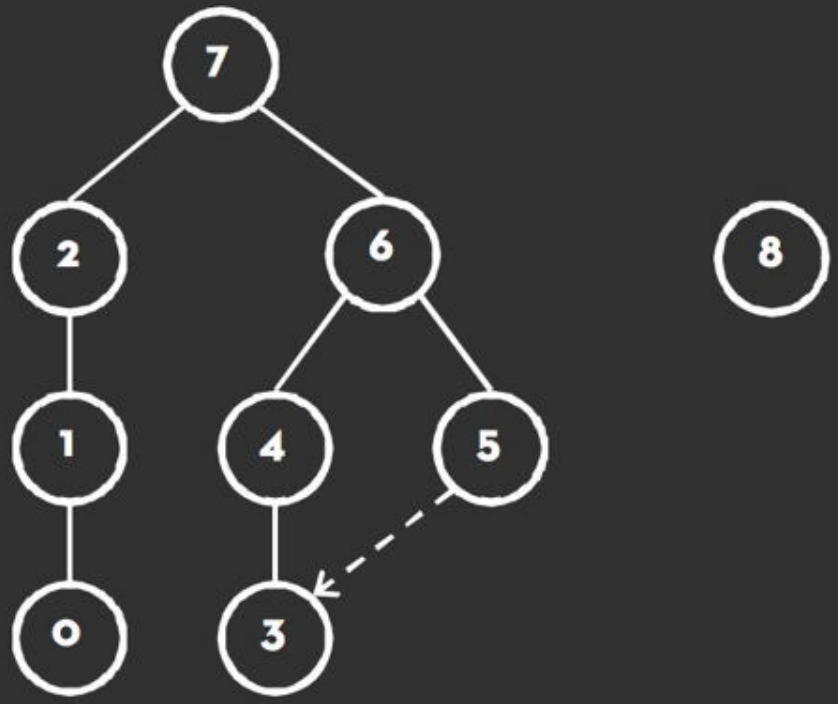
Reingold-Tilford Algorithm



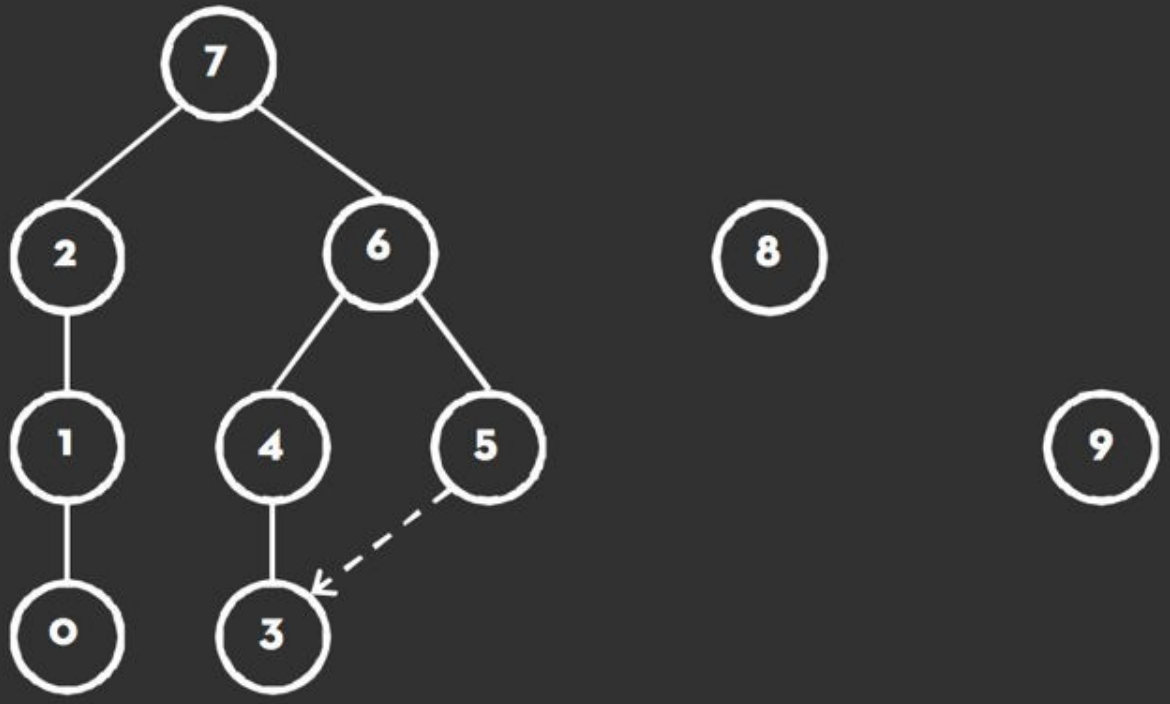
Reingold-Tilford Algorithm



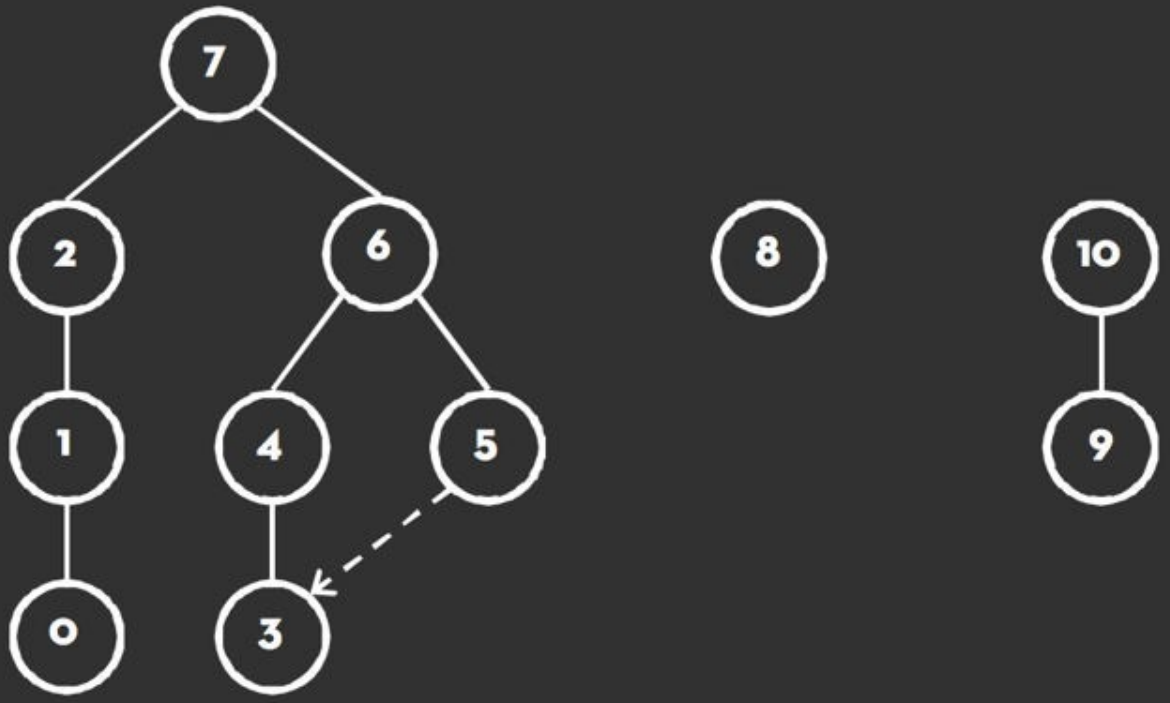
Reingold-Tilford Algorithm



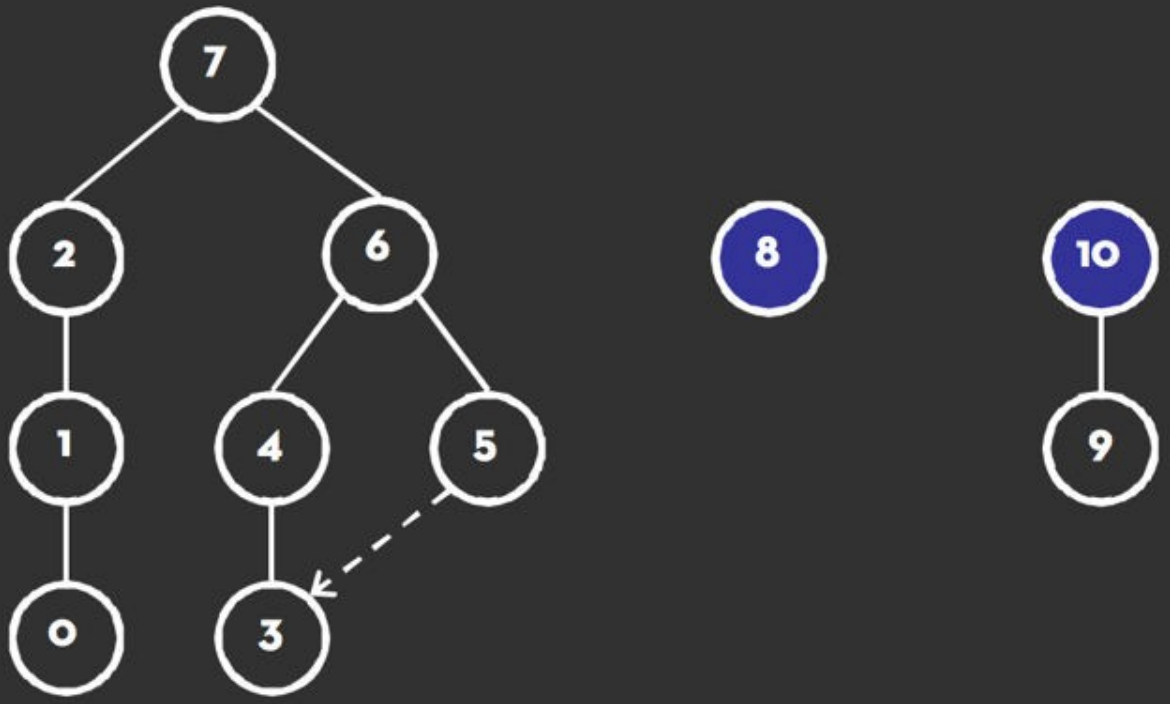
Reingold-Tilford Algorithm



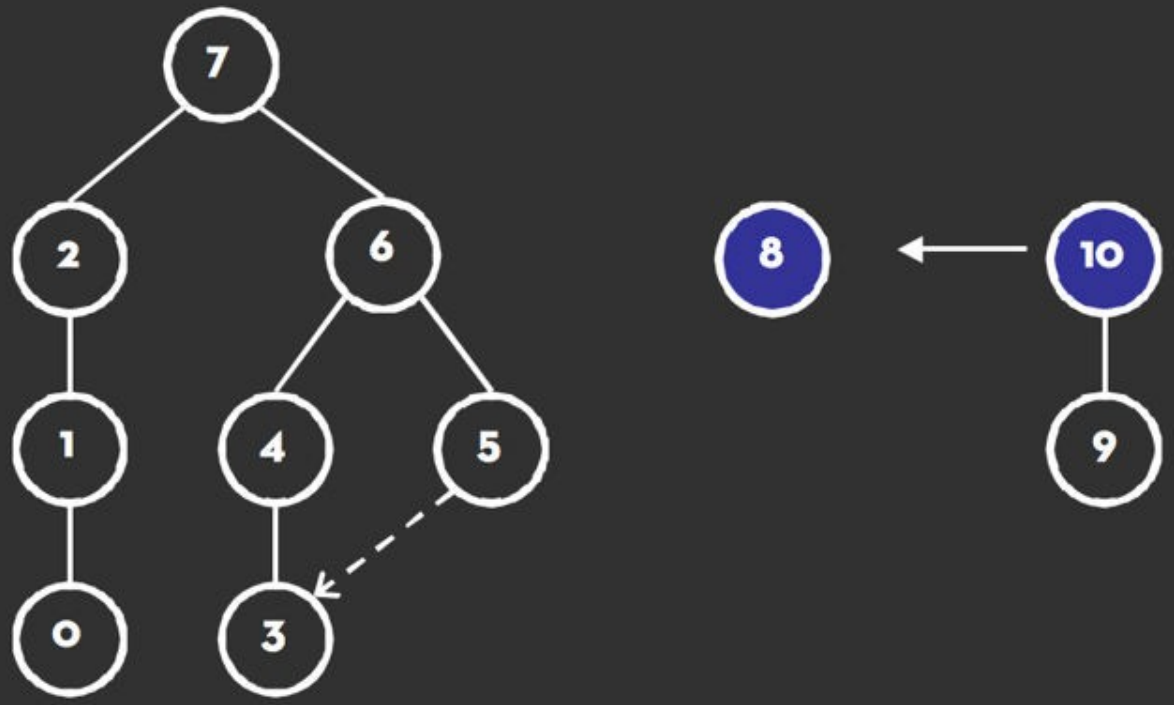
Reingold-Tilford Algorithm



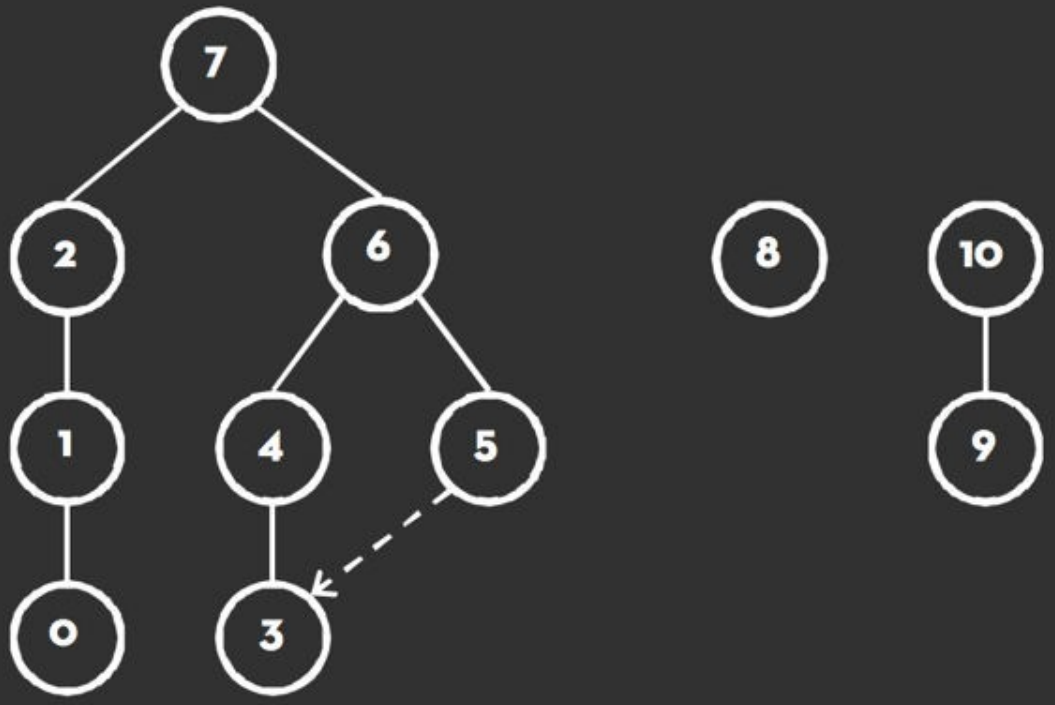
Reingold-Tilford Algorithm



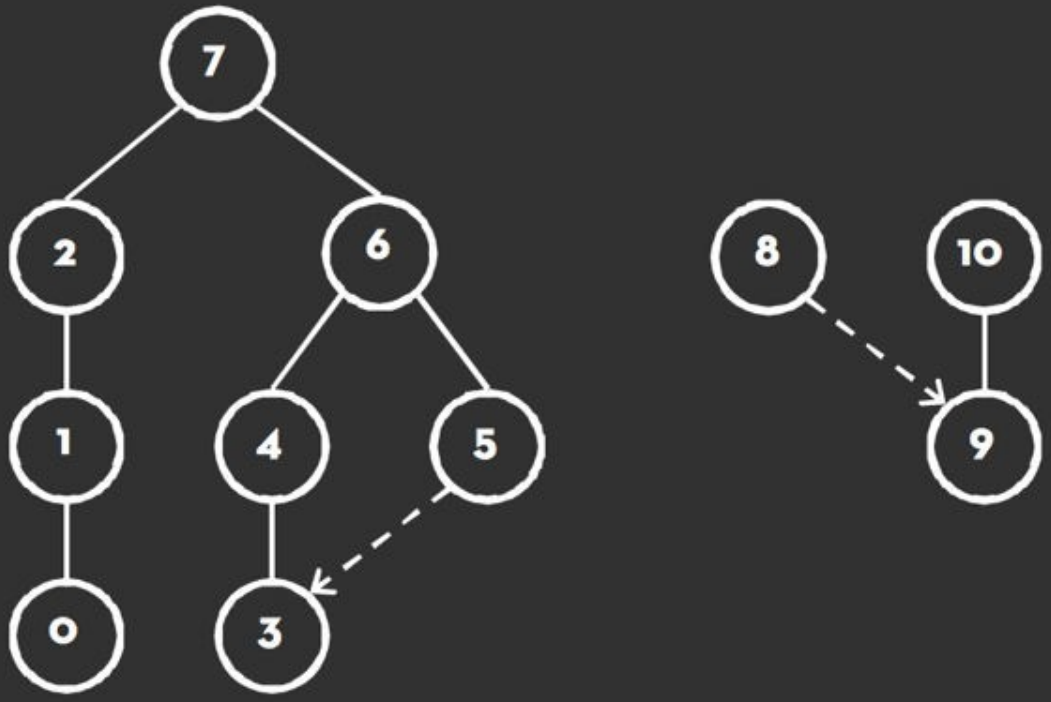
Reingold-Tilford Algorithm



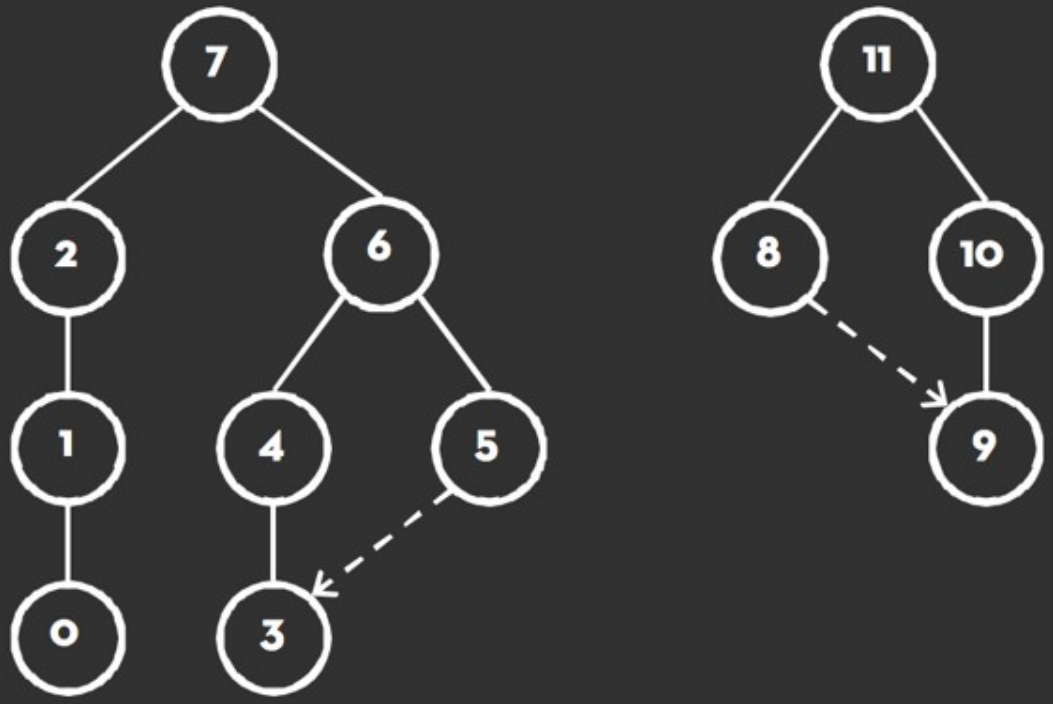
Reingold-Tilford Algorithm



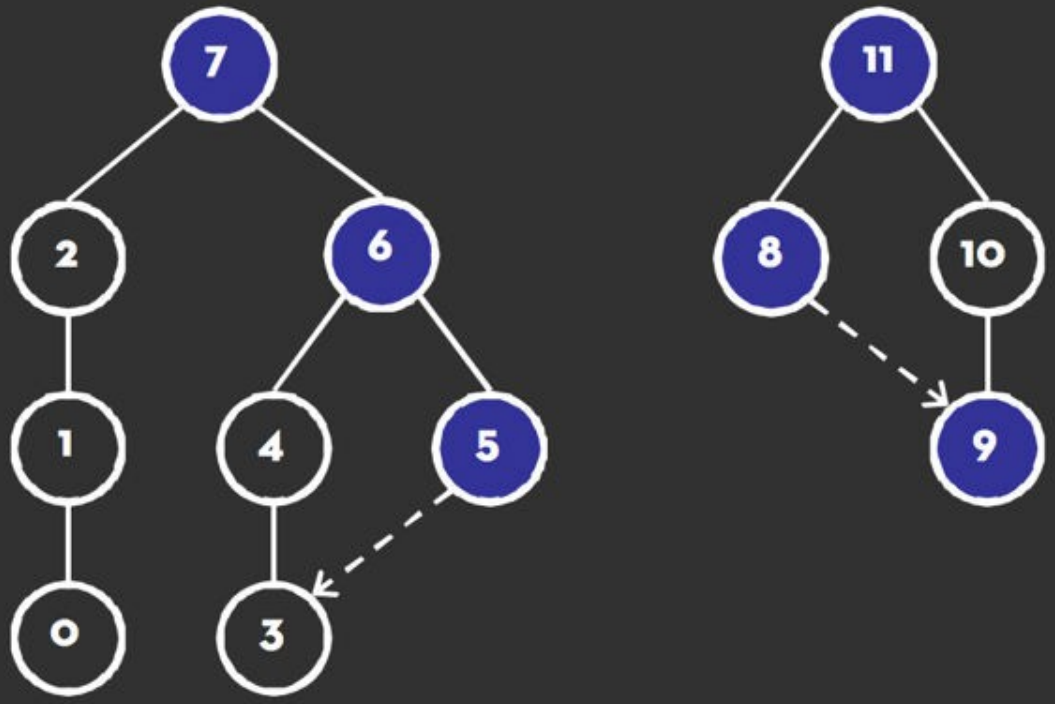
Reingold-Tilford Algorithm



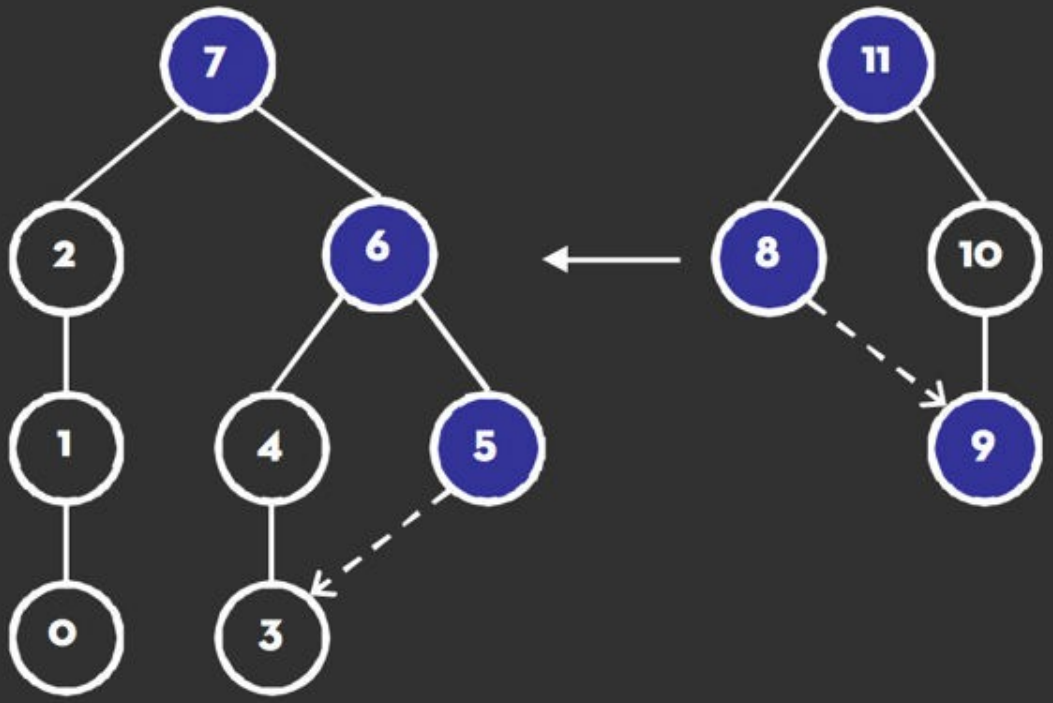
Reingold-Tilford Algorithm



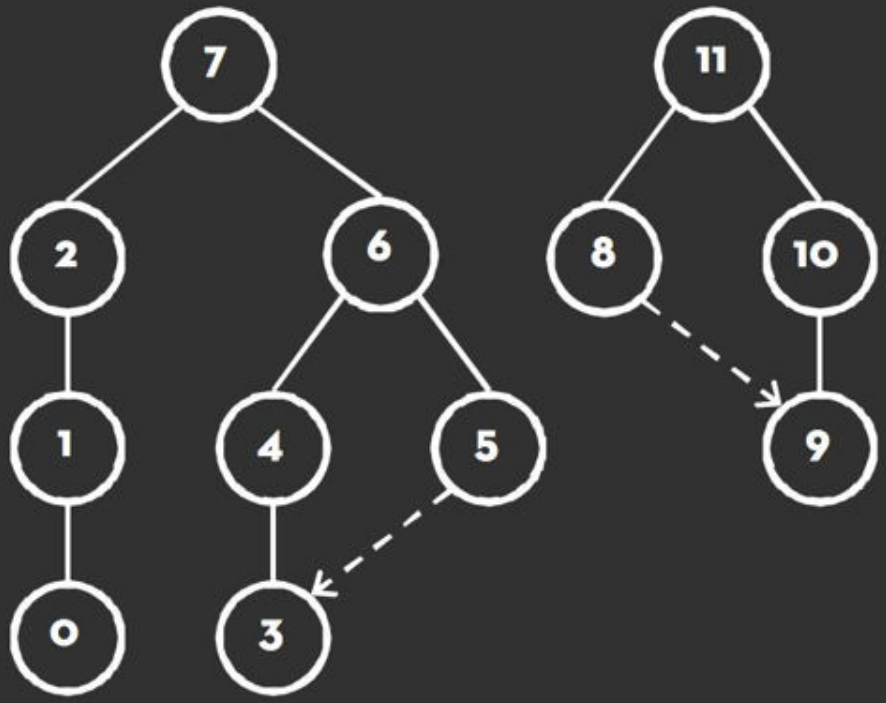
Reingold-Tilford Algorithm



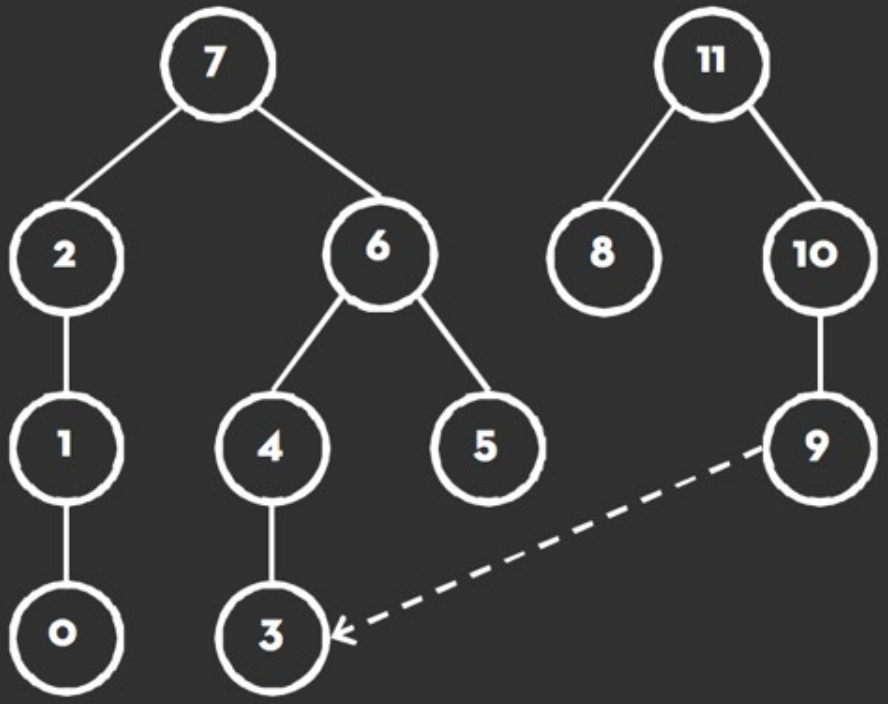
Reingold-Tilford Algorithm



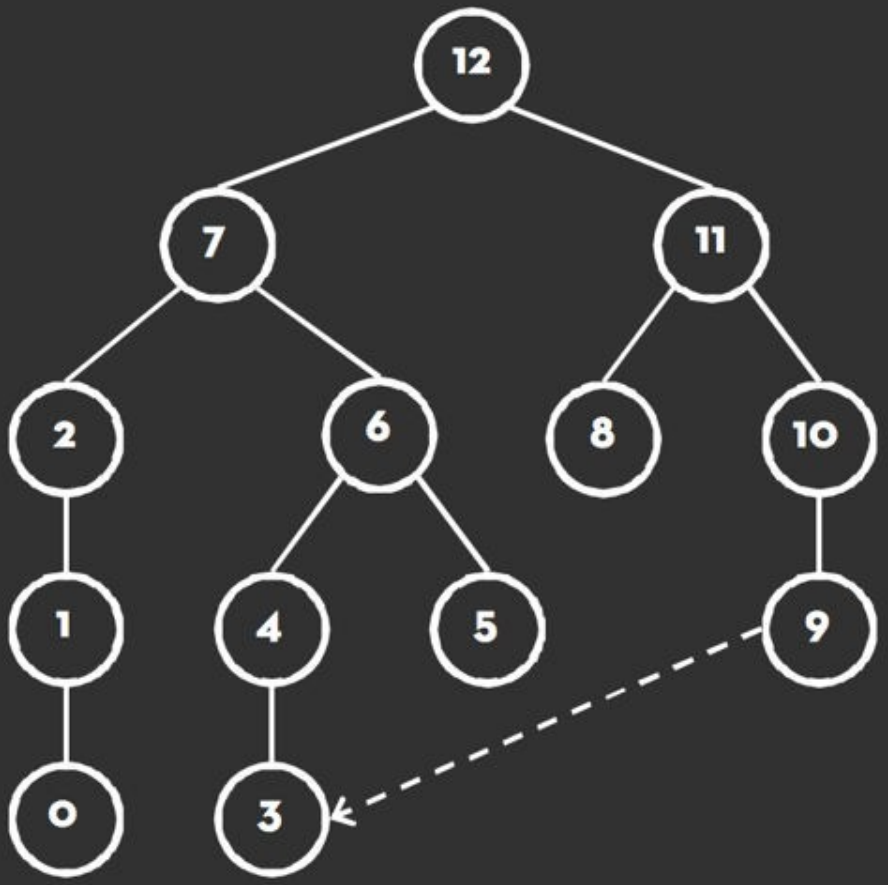
Reingold-Tilford Algorithm



Reingold-Tilford Algorithm



Reingold-Tilford Algorithm



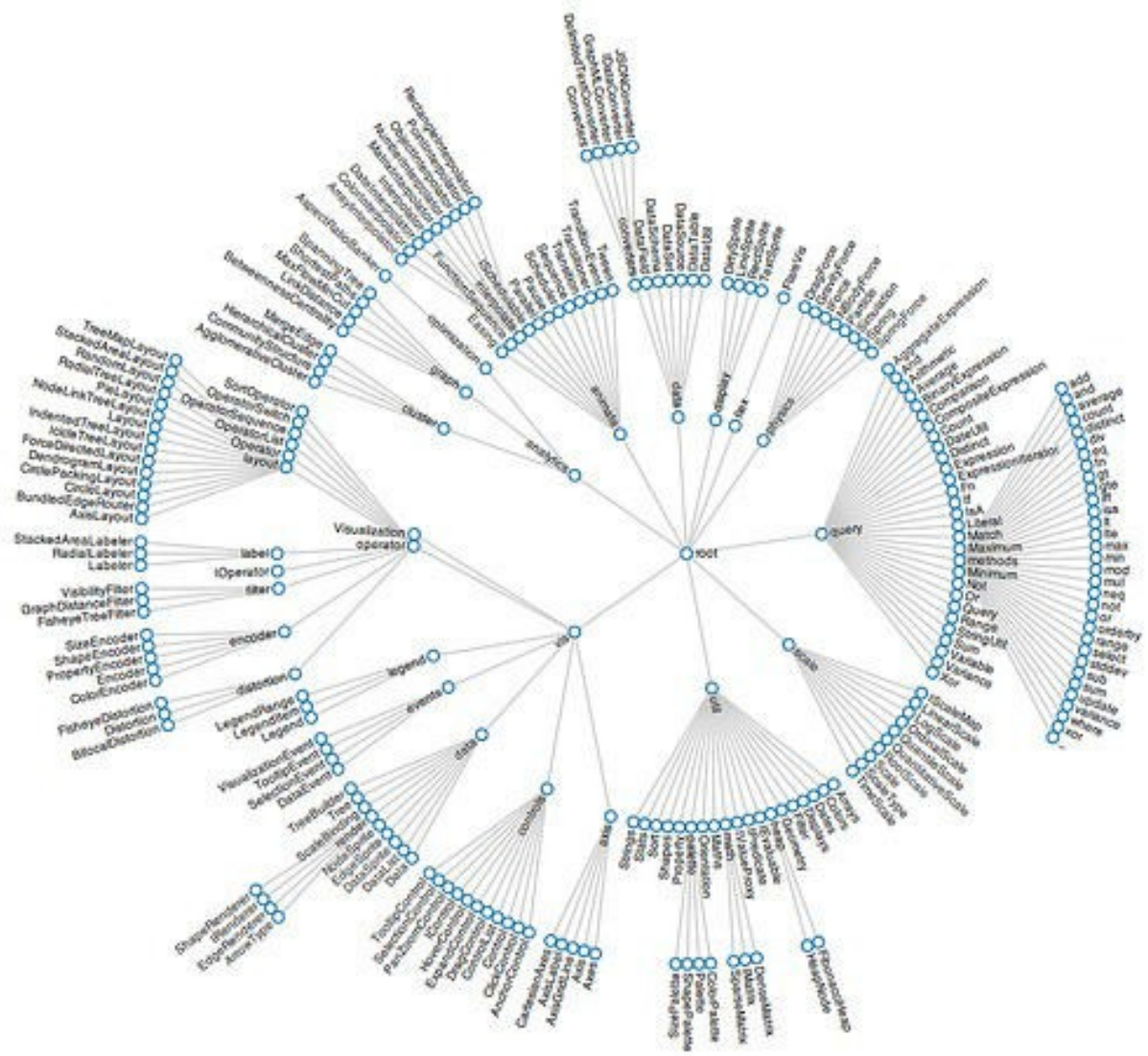
Reingold-Tilford Algorithm

- Bottom-up tree traversal
- y-coord is the depth of the node, x-coords are “locally defined” (so first is arbitrary)
- merge trees
  - push right tree as close as possible to left tree (this is where the contour comes in)
  - position **shifts** saved at each node
  - parent nodes are centered above direct children
- Final top-down pass to convert shifts to positions

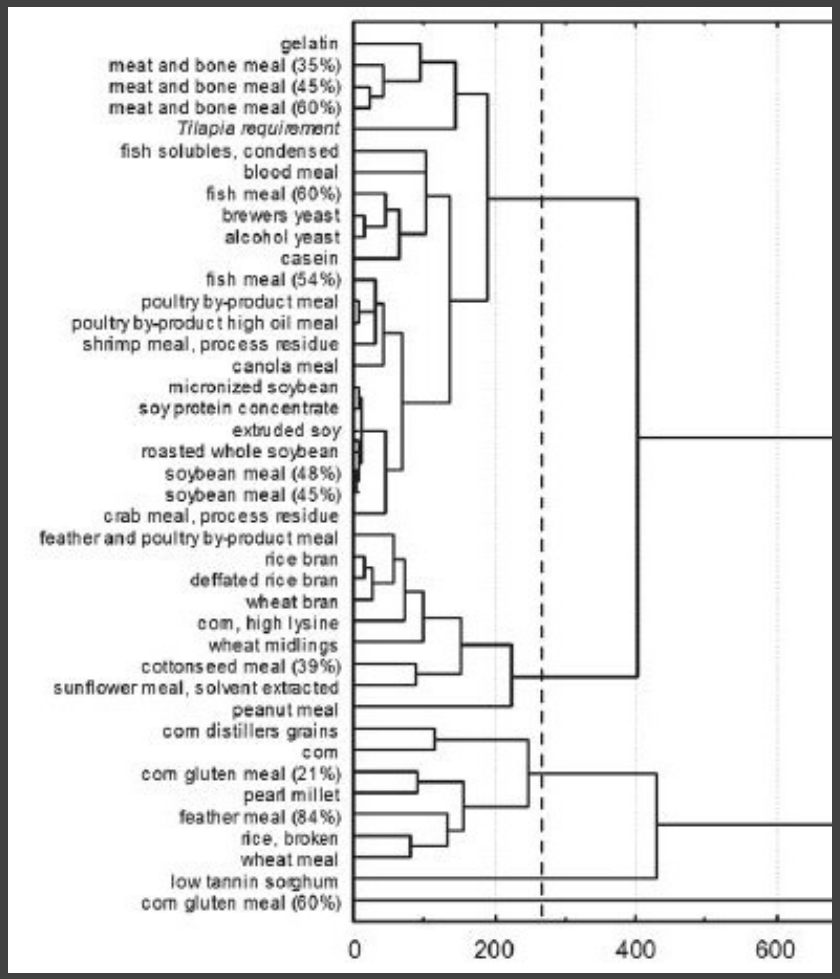
RADIAL LAYOUT

- node-link diagram in polar coordinates
- Use the available screen space more efficiently by
  - placing the root node at the center of the screen space and
  - placing all other nodes at concentric circles around the root node, where the distance represents the depth of the node. angular sectors assigned to subtrees
- Reingold-Tilford can be applied

RADIAL LAYOUT



Cluster Dendrograms



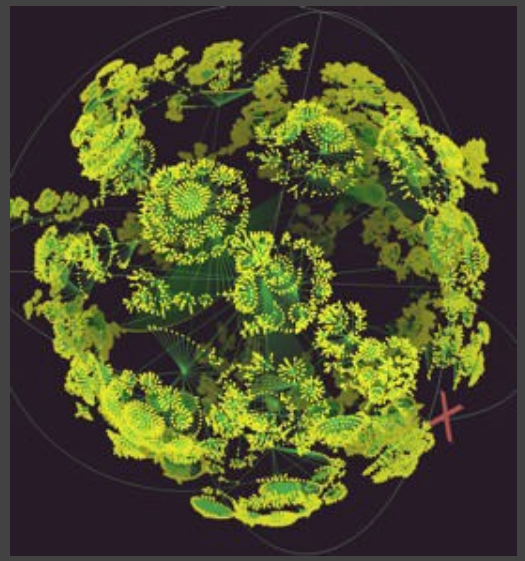
Depicts cluster trees produced by hierarchical clustering algorithms.

Leaf nodes arranged in a line, internal node depth indicates order/value at which clusters merge.

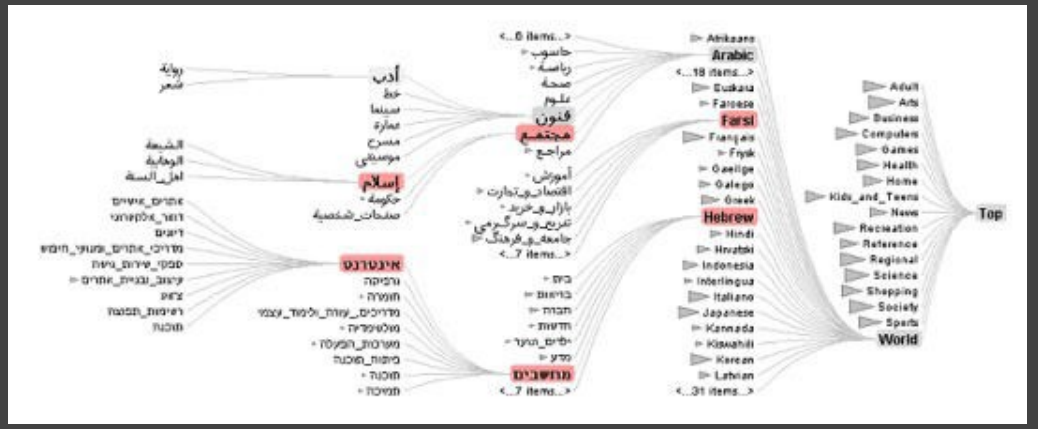
Naïve recursive layout with orthogonal two-segment edges.

NODE-LINK PROBLEMS

- **scale**
  - tree breadth often grows exponentially
  - quickly run out of space!
- **solutions**
  - Radial layout
  - hyperbolic layout
  - filtering
  - scrolling or panning
  - zooming

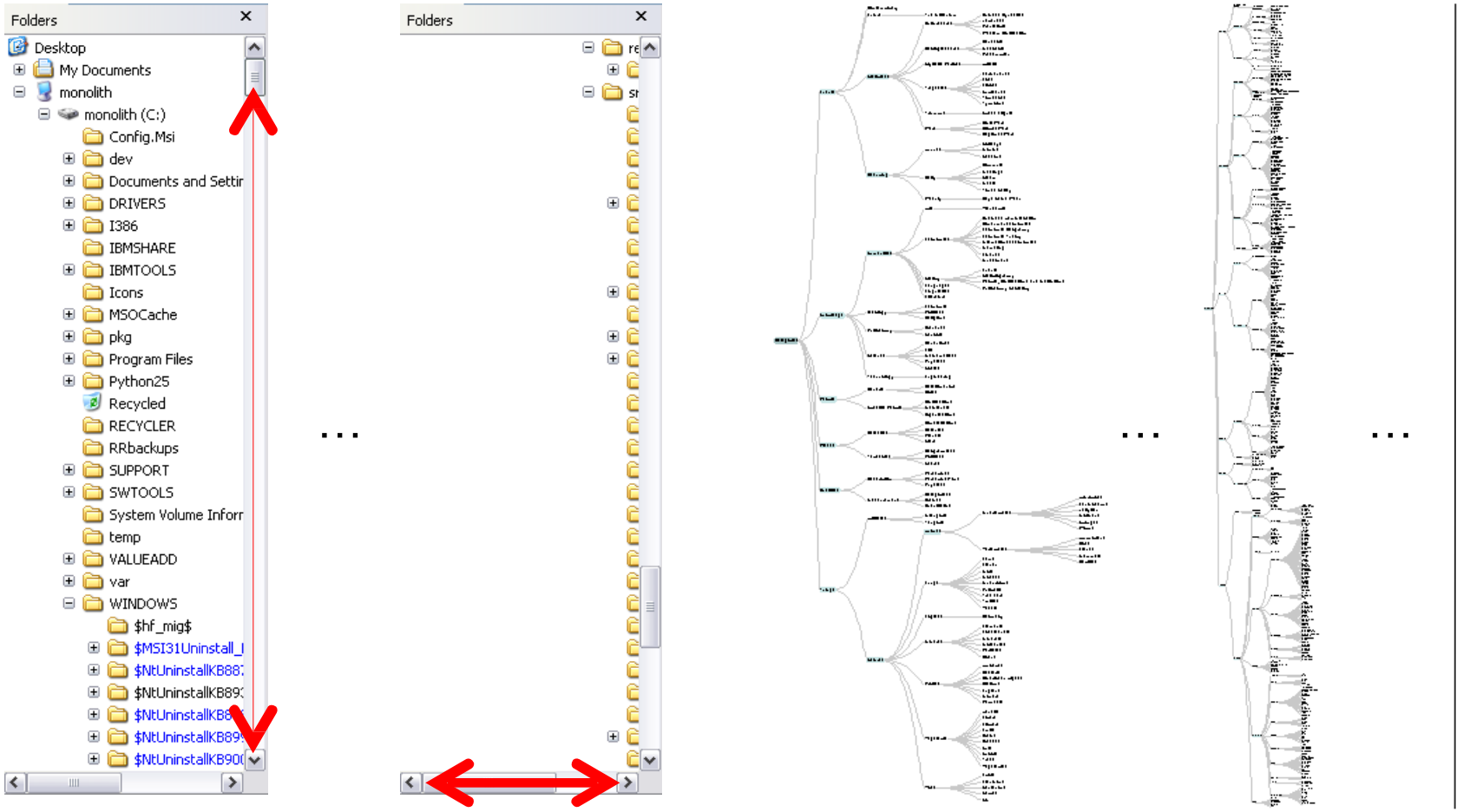


http://www.caida.org/tools/visualization/walrus/



http://vis.stanford.edu/papers/doitrees-revisited

Visualizing Large Hierarchies



Indented Layout

Reingold-Tilford Layout

More Nodes, More Problems…

Scale

Tree breadth often grows exponentially

Even with tidy layout, quickly run out of space

Possible Solutions

Filtering

Focus+Context

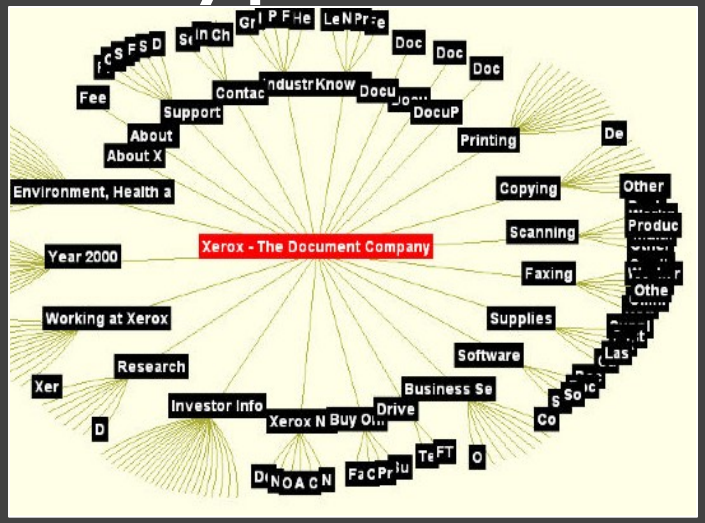
Scrolling or Panning

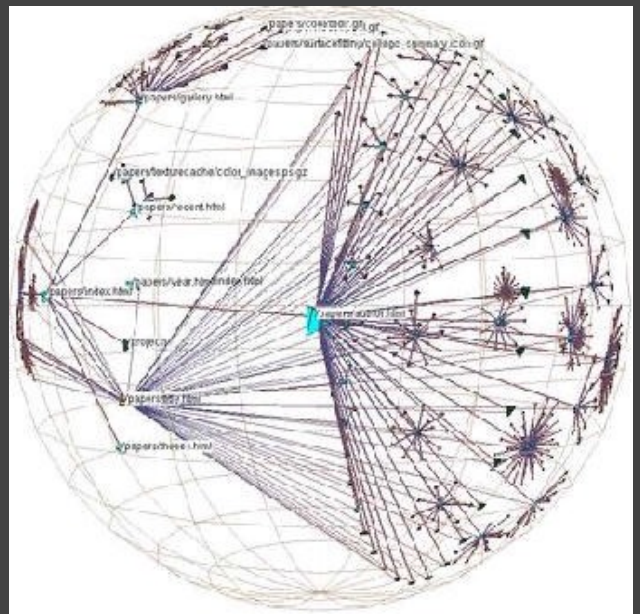
Zooming

Aggregation

MC Escher, *Circle Limit IV*

Hyperbolic Layout





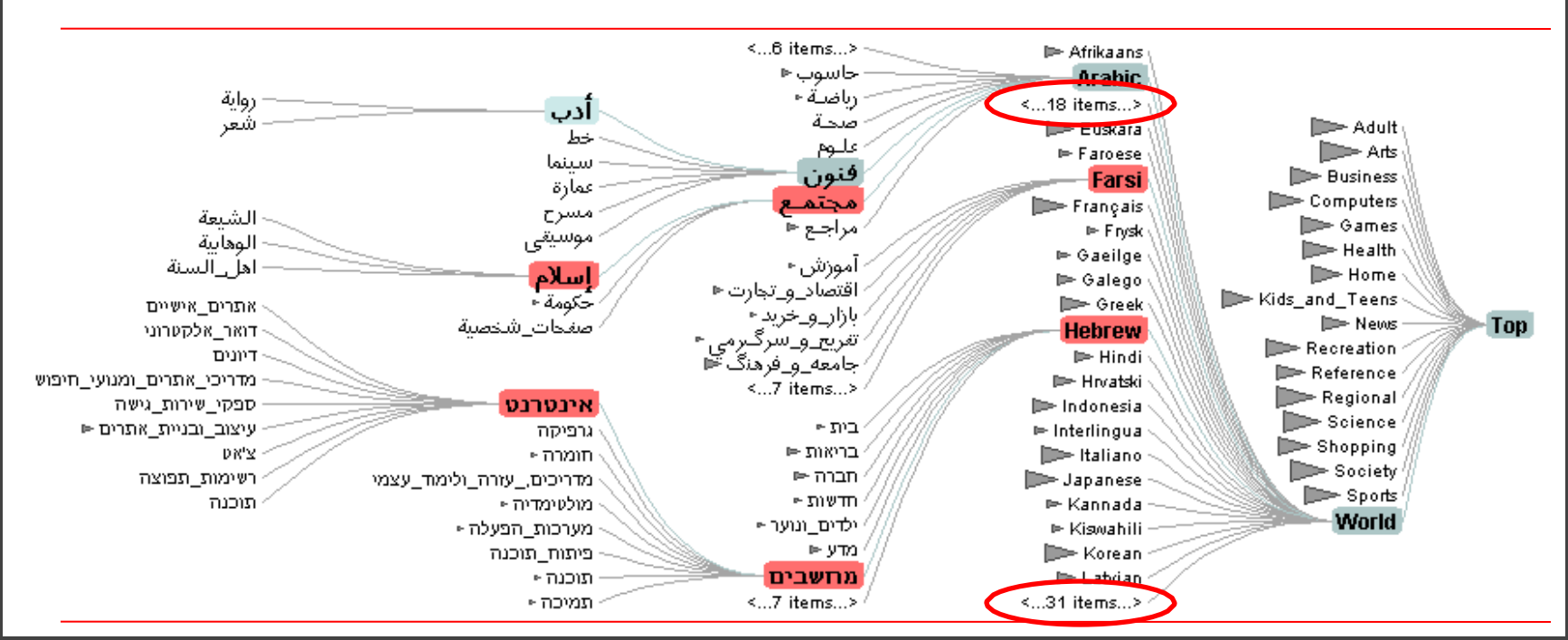
Perform tree layout in hyperbolic geometry, project the result on to the Euclidean plane.

Why? Like tree breadth, the hyperbolic plane expands exponentially!

Also computable in 3D, projected into a sphere.

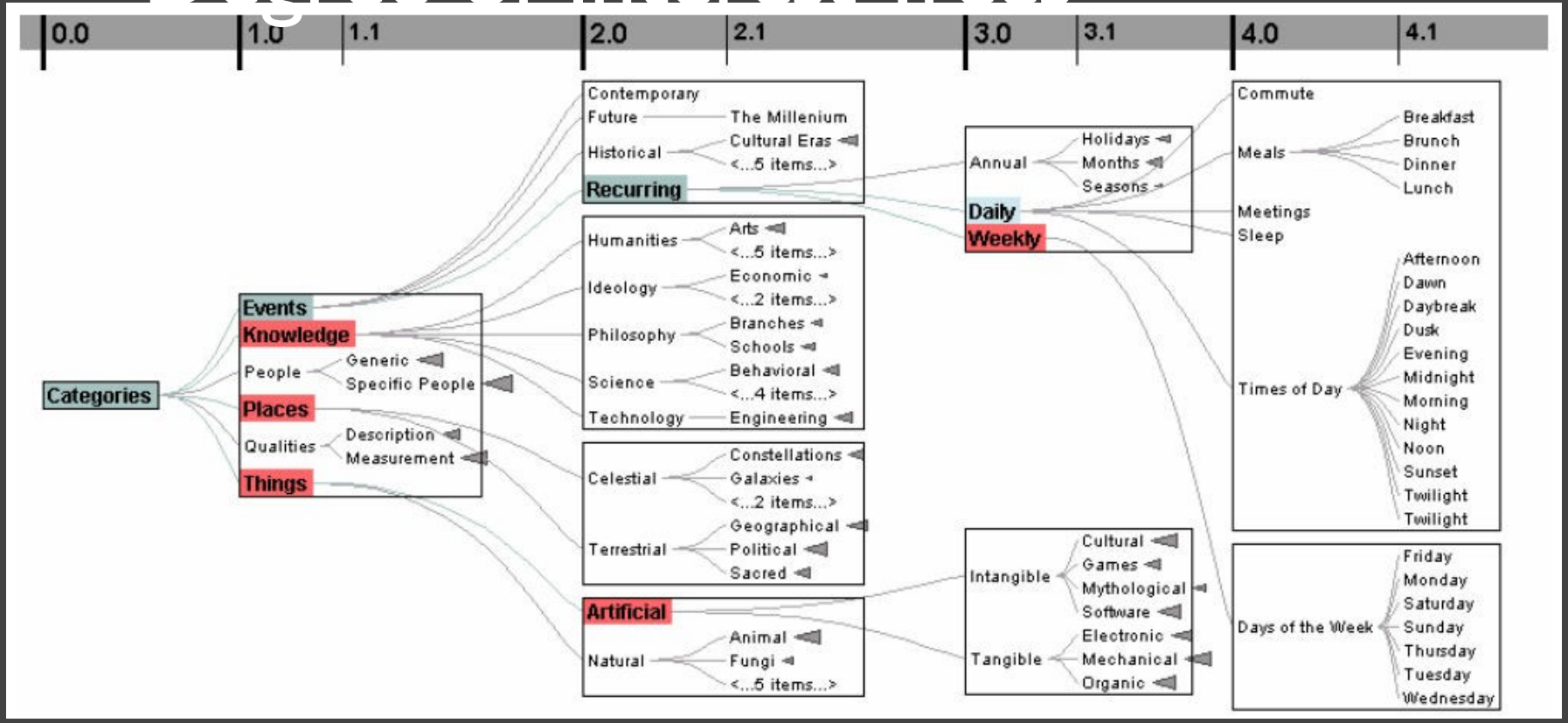
Focus + Context

Degree-of-Interest Trees



Space-constrained, multi-focal tree layout

Degree-of-Interest Trees



Remove “low interest” nodes at a given depth level until all blocks on a level fit within bounds.

Attempt to center child blocks beneath parents.

Enclosure

Enclosure Diagrams

Encode structure using **spatial enclosure**
Popularly known as **treemaps**

Benefits
Provides a single view of an entire tree
Easier to spot large/small nodes

Problems
Difficult to accurately read structure / depth

Circle Packing Layout

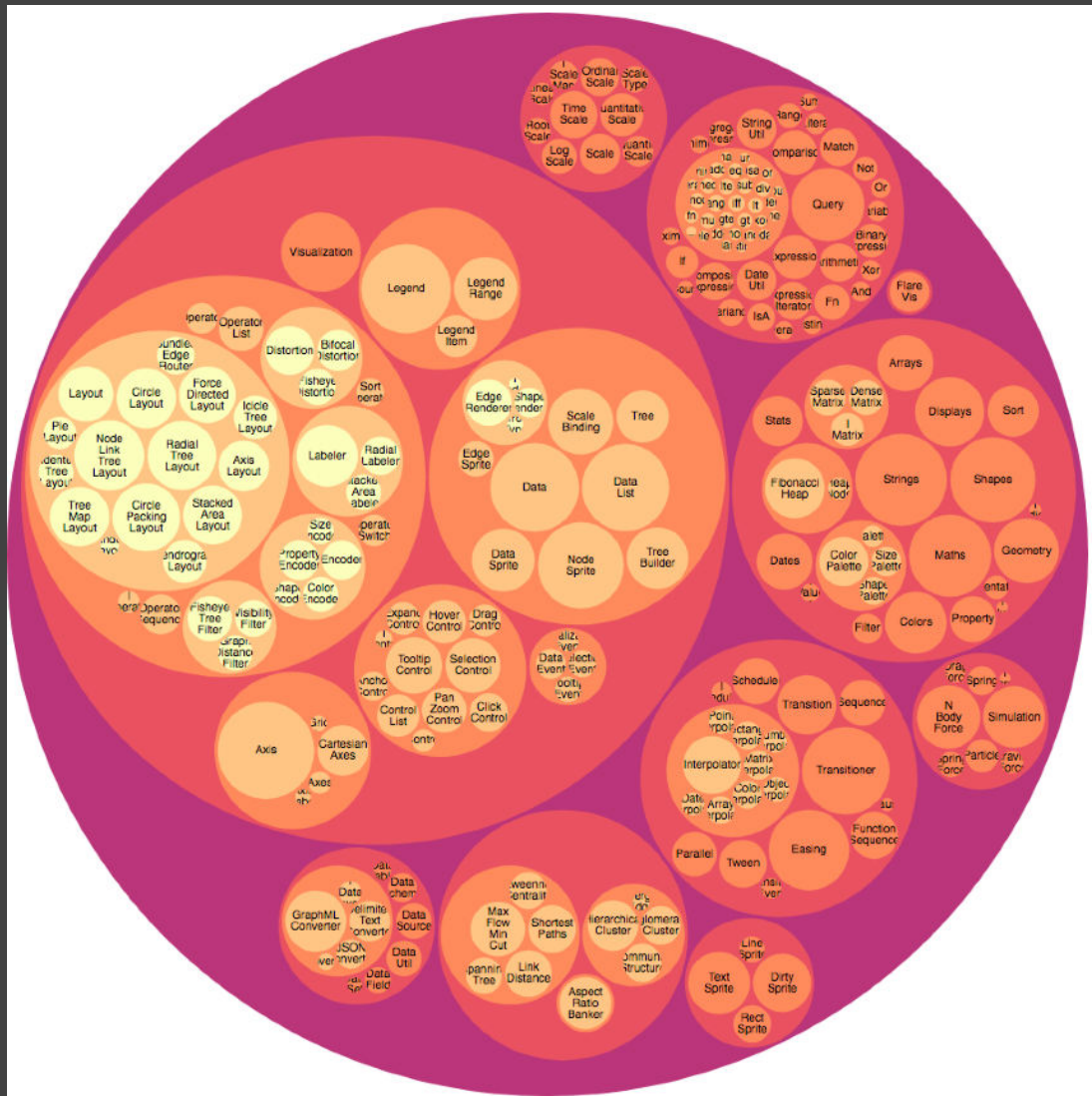
Nodes are represented as sized circles.

Nesting shows parent-child relationships.

Issues?

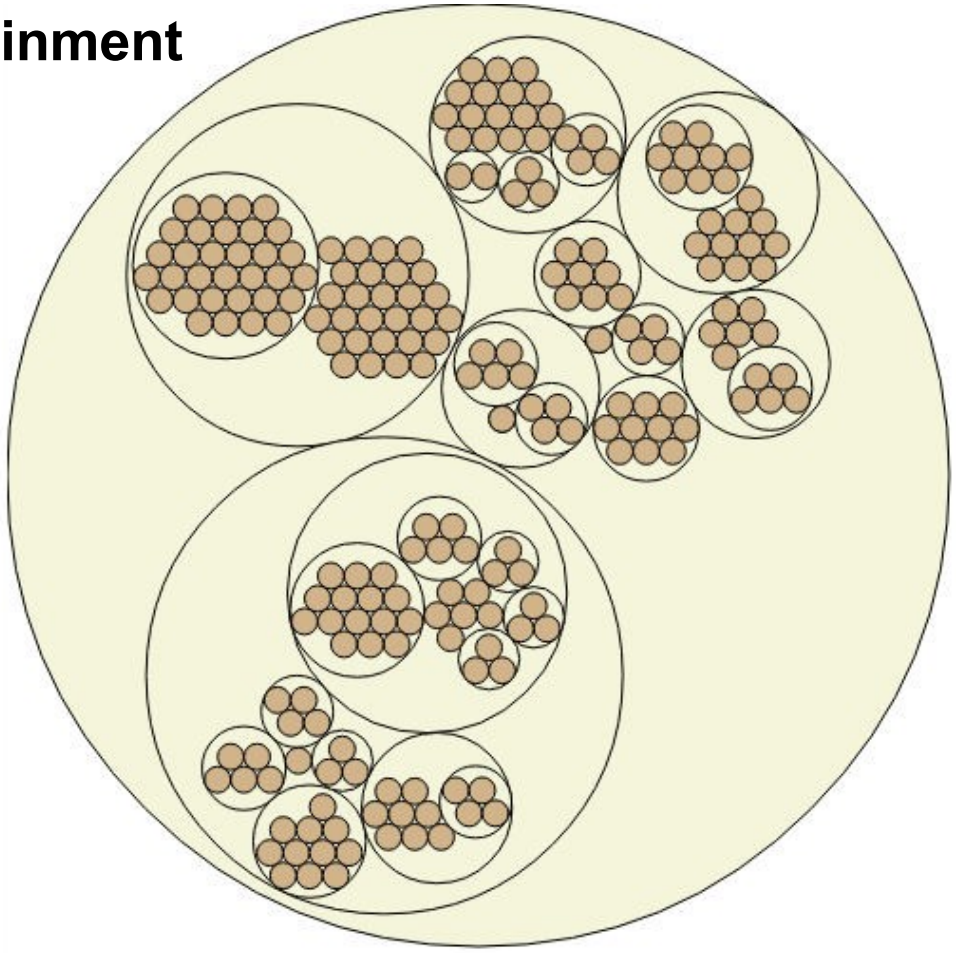
Inefficient use of space.

Parent size misleading?



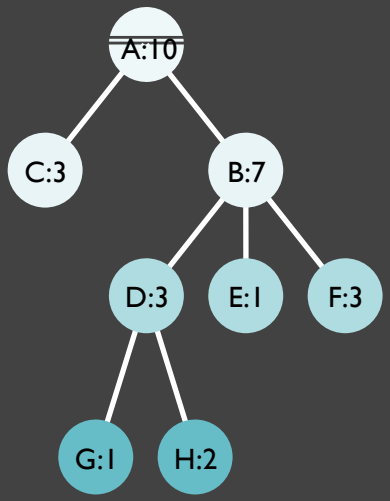
Bubble Charts

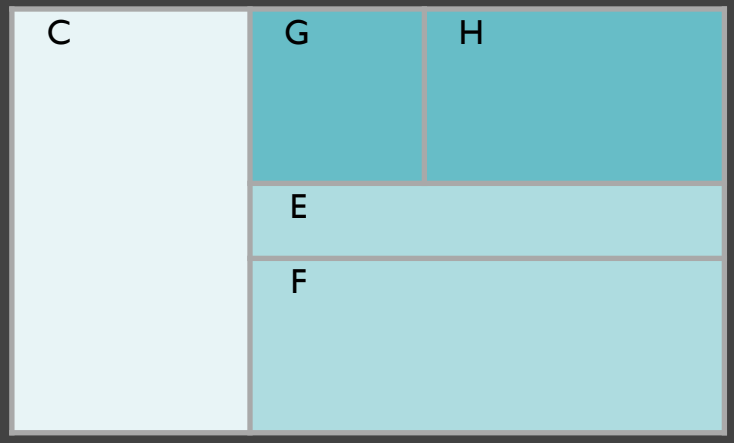
- Represent **hierarchy** by **containment**
- Let's work out a simple algo!



TREEMAPS

- **a 2-D space-filling layout**
- **recursively fill space based on a size metric for nodes**
- **enclosure indicates hierarchy**
- **additional measures can control aspect ratio of cells**
- **most often use rectangles, but other shapes possible**
  - square, circle, voronoi tessellation





Treemaps

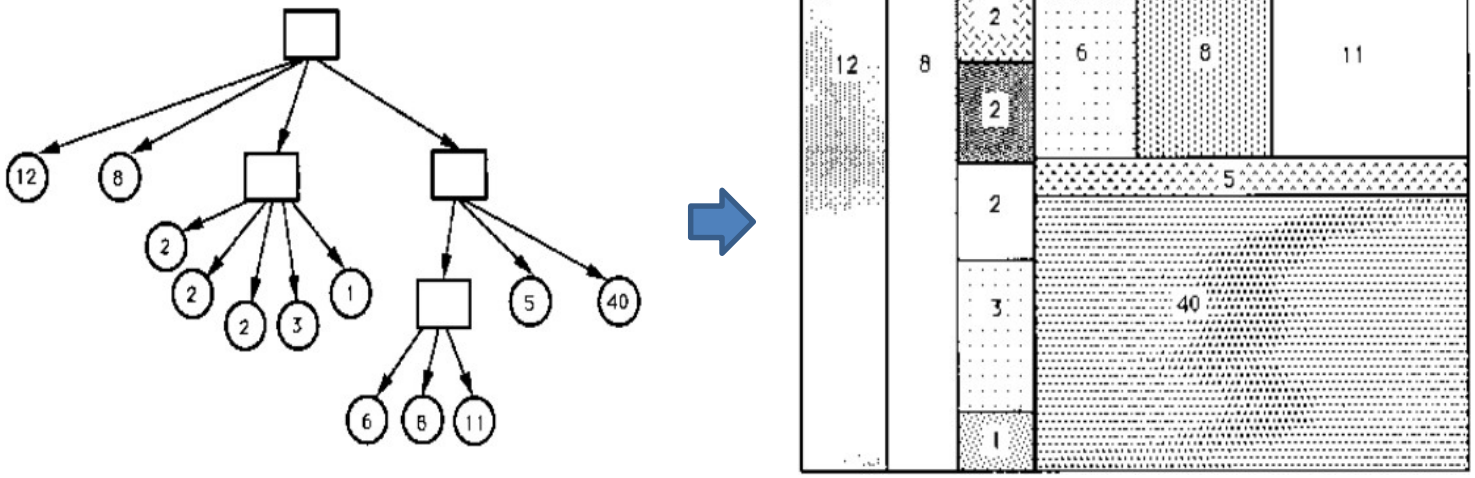
Hierarchy visualization that emphasizes values of nodes via area encoding.

Partition 2D space such that leaf nodes have sizes proportional to data values.

First layout algorithms proposed by Shneiderman et al. in 1990, with focus on showing file sizes on a hard drive.

A CLASSIC CONTAINMENT LAYOUT

- example tree to rebuild with treemap algorithm
- size of each node as numbers in leaves



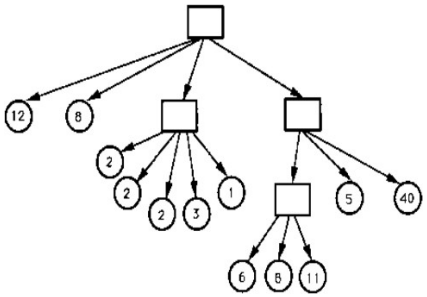
TREEMAP ALGORITHM

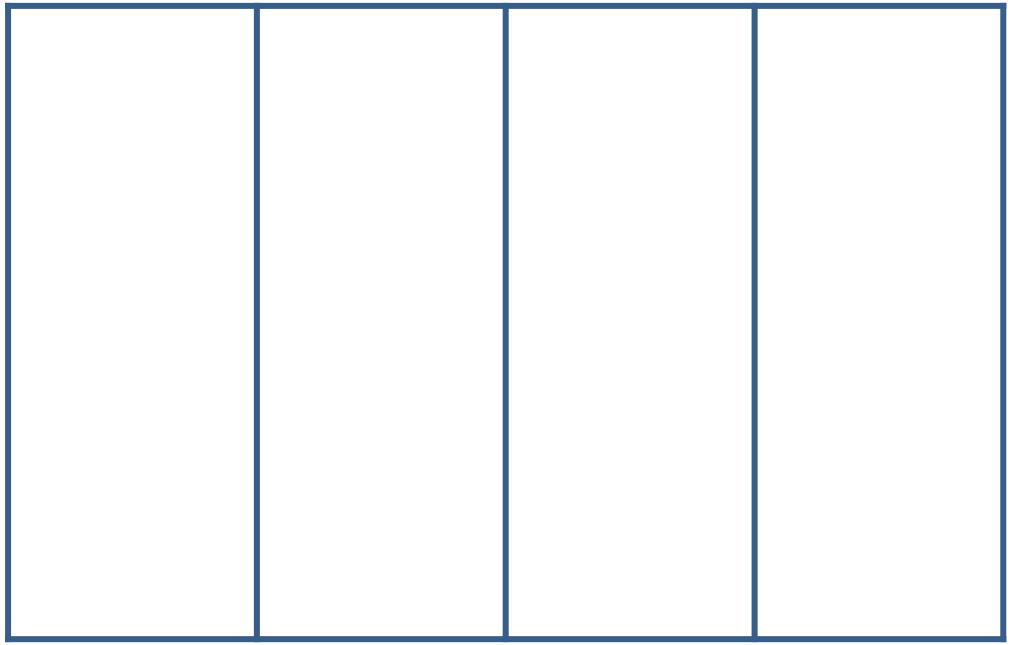
- Take a rectangular display area P1(x1,y1), Q1(x2,y1)
- This area represents the root of the tree

TREEMAP ALGORITHM

recursive algorithm

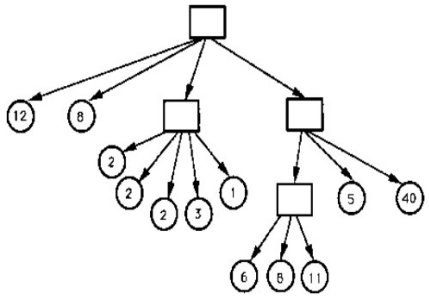
– the number of children of the current node define the number of partitions of the current node

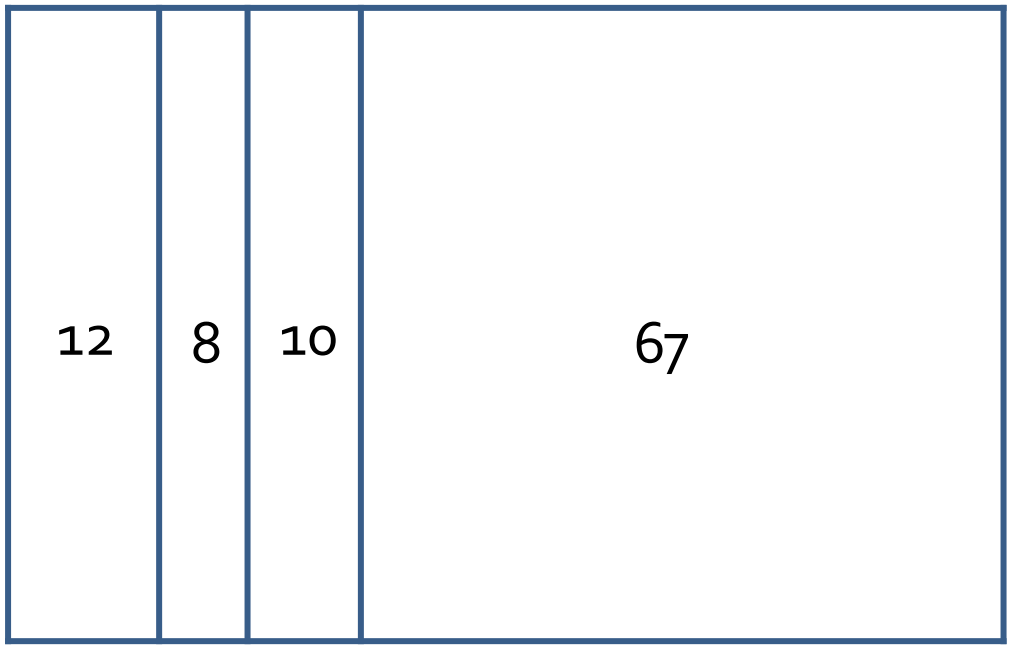




TREEMAP ALGORITHM

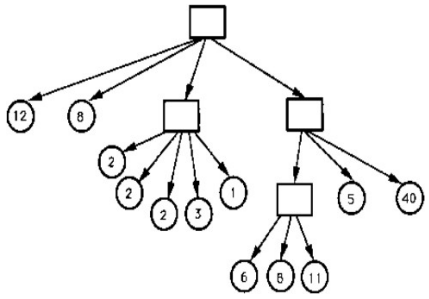
the weight of each node determines the size of each partition

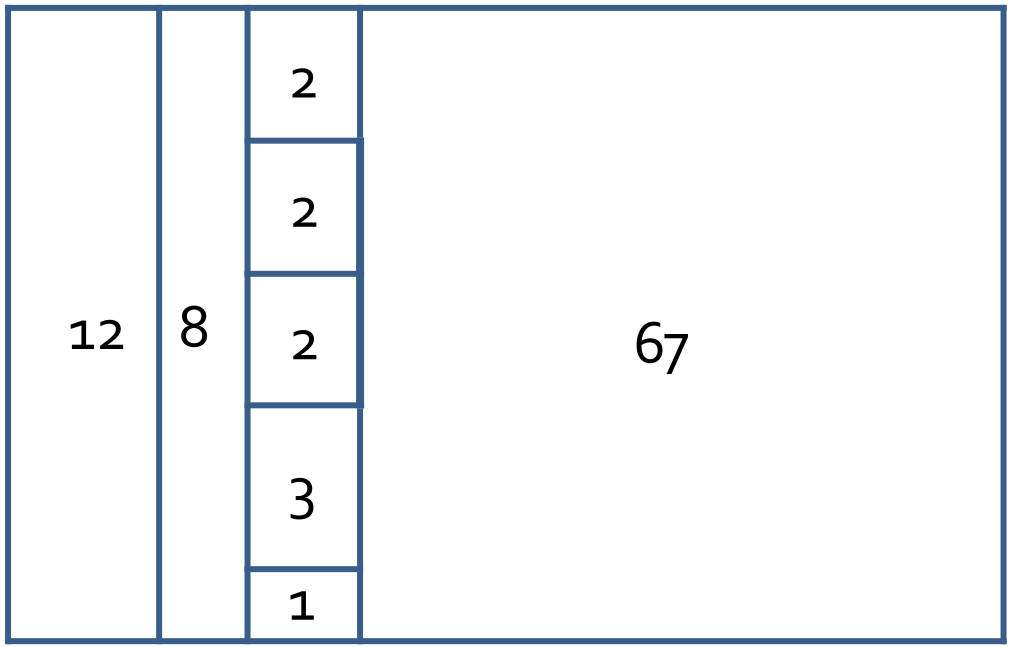




TREEMAP ALGORITHM

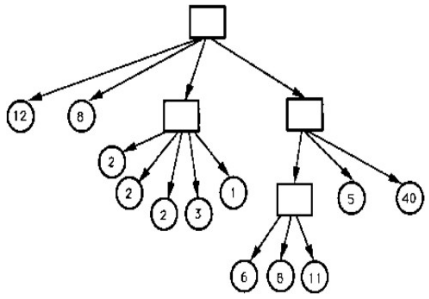
at each change of level, rotate orientation of split by 90 degrees

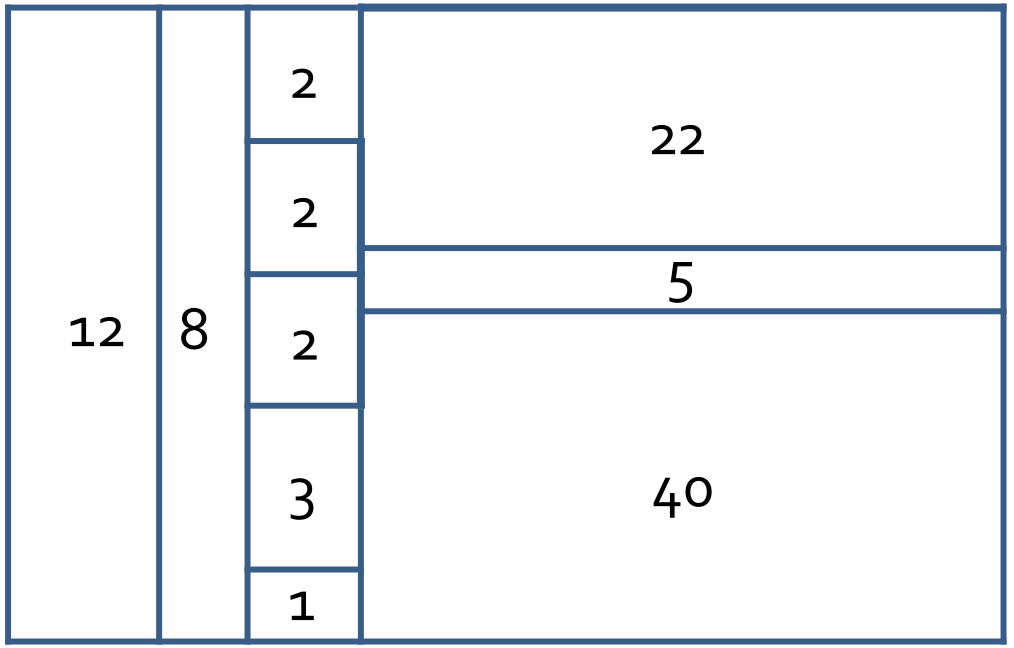




TREEMAP ALGORITHM

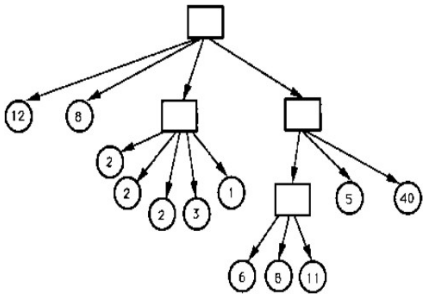
at each change of level, rotate orientation of split by 90 degrees

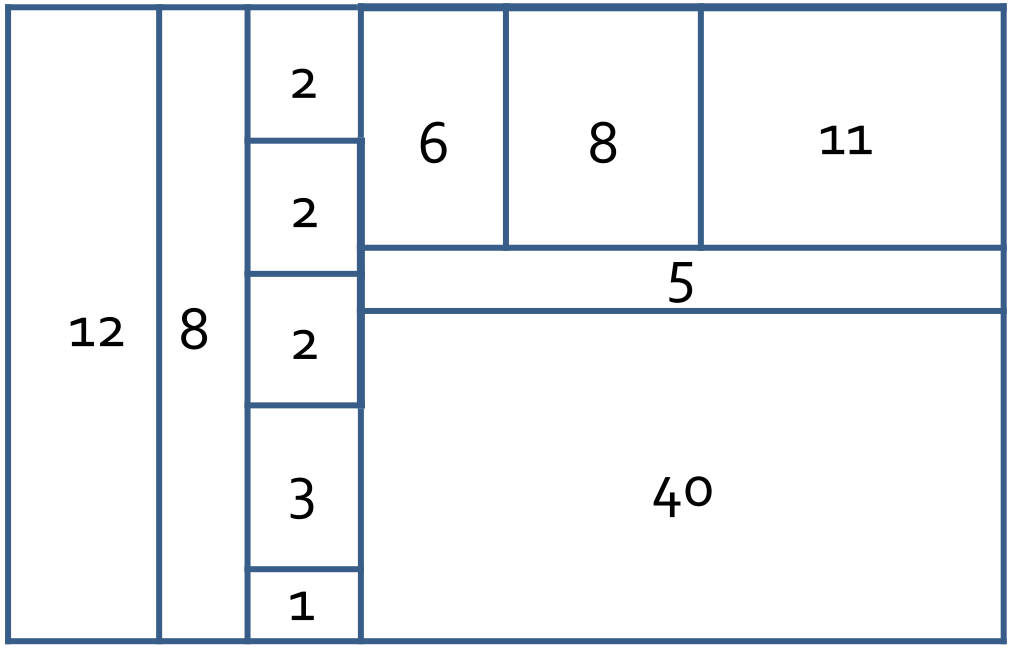




TREEMAP ALGORITHM

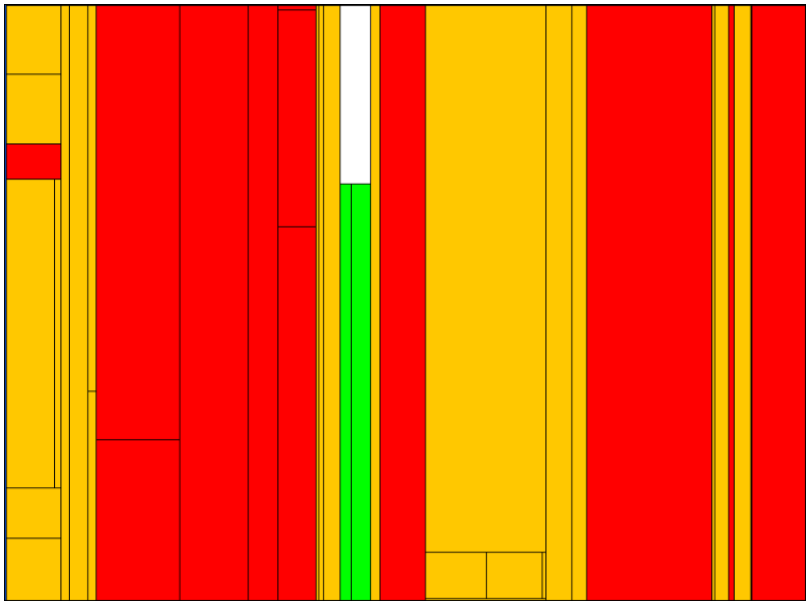
at each change of level, rotate orientation of split by 90 degrees





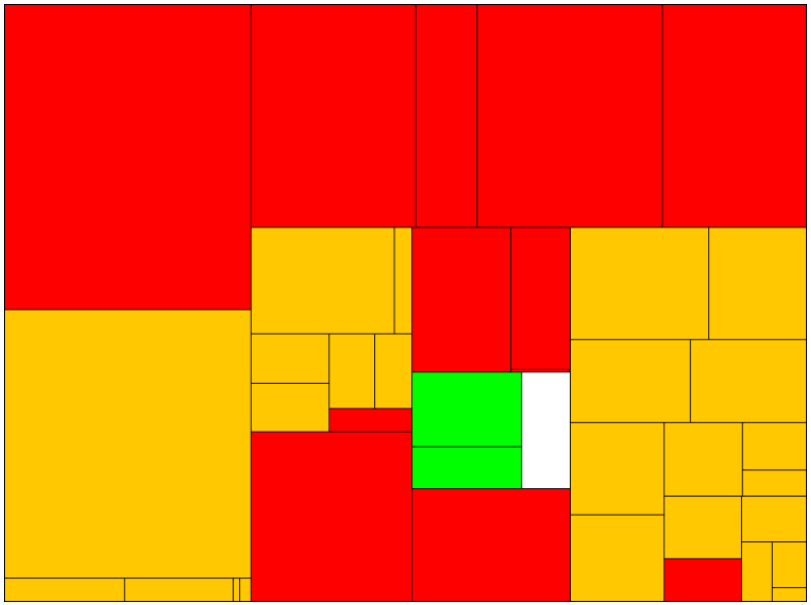
TREEMAP VARIATIONS

- problem with original treemap: lots of long stripes
- for long stripes the areas are difficult to compare



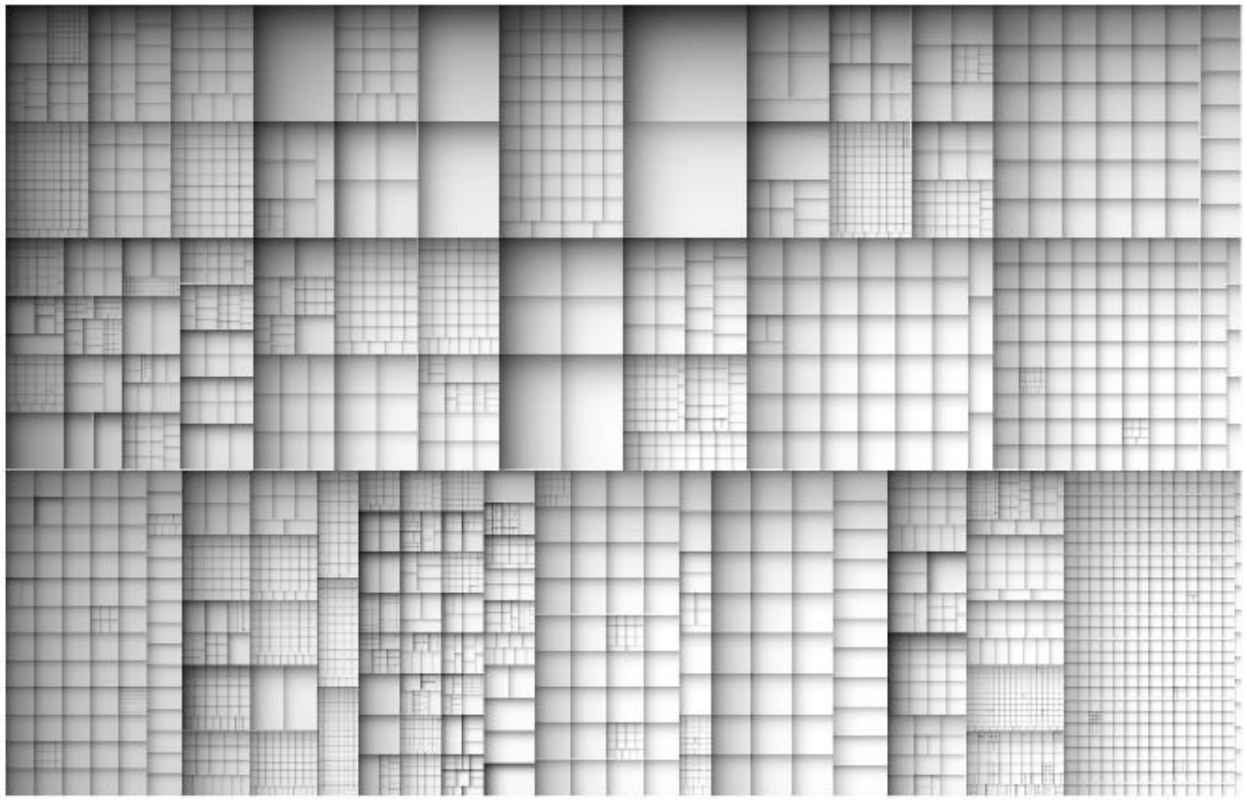
SQUARIFIED TREEMAP

- calculates more squared regions
- problem: order not as easily read, not very stable with dynamically changing data



Squarified Treemaps

- A little harder, tries to make square shapes



Why Squares? [Bruls et al. ’00]

Posited Benefits of 1:1 Aspect Ratios

1. Minimize perimeter, reducing border ink.
   Mathematically true!

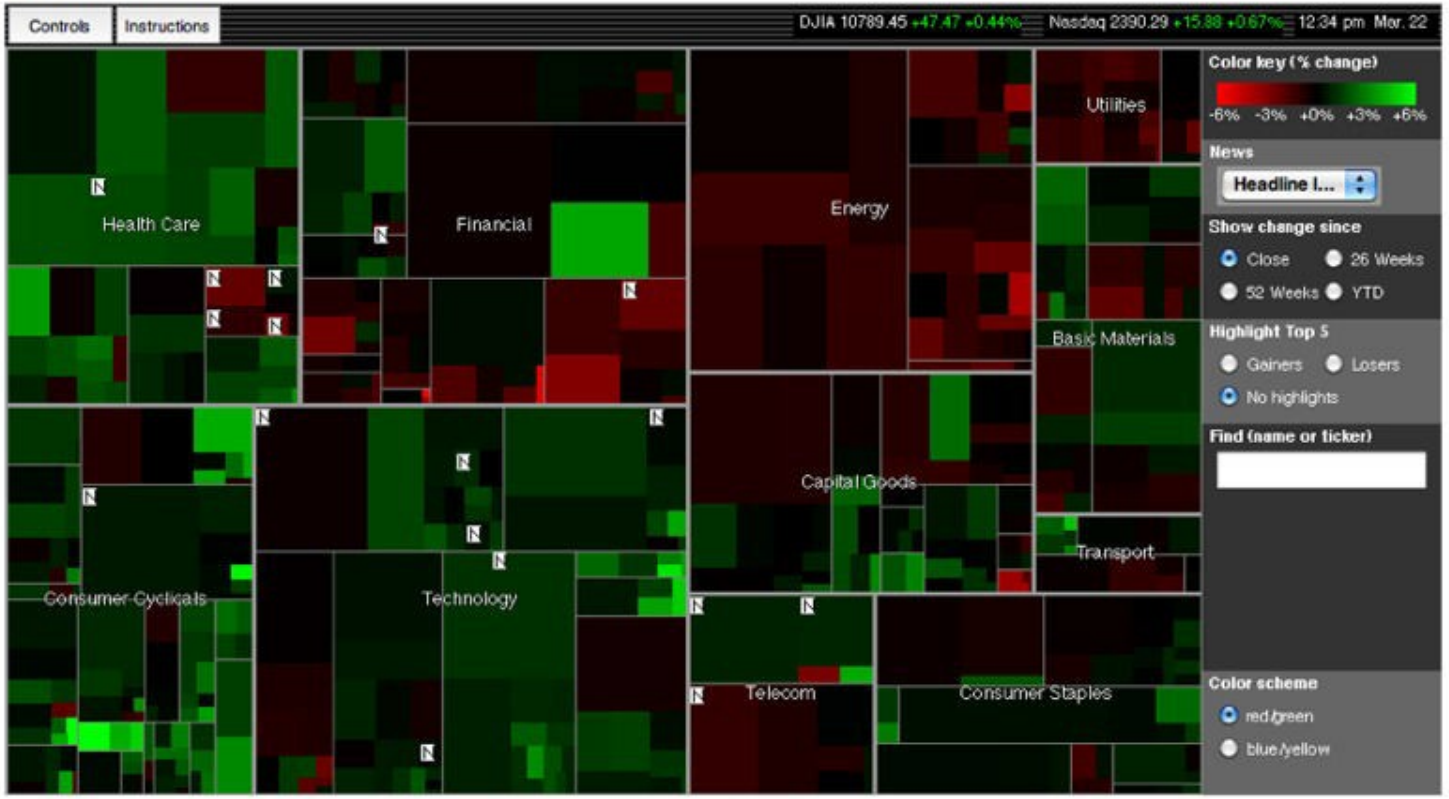
2. Easier to select with a mouse cursor.
   Validated by empirical research & Fitt’s Law!

3. Similar aspect ratios are easier to compare.
   Seems intuitive, but is this true?

Map of the Market

Launch Map in Separate Window

SmartMoneySelect Upgrade here to access the Market Map 1000 and search 1,000 companies with enhanced screening capabilities.



MARKET NEWS

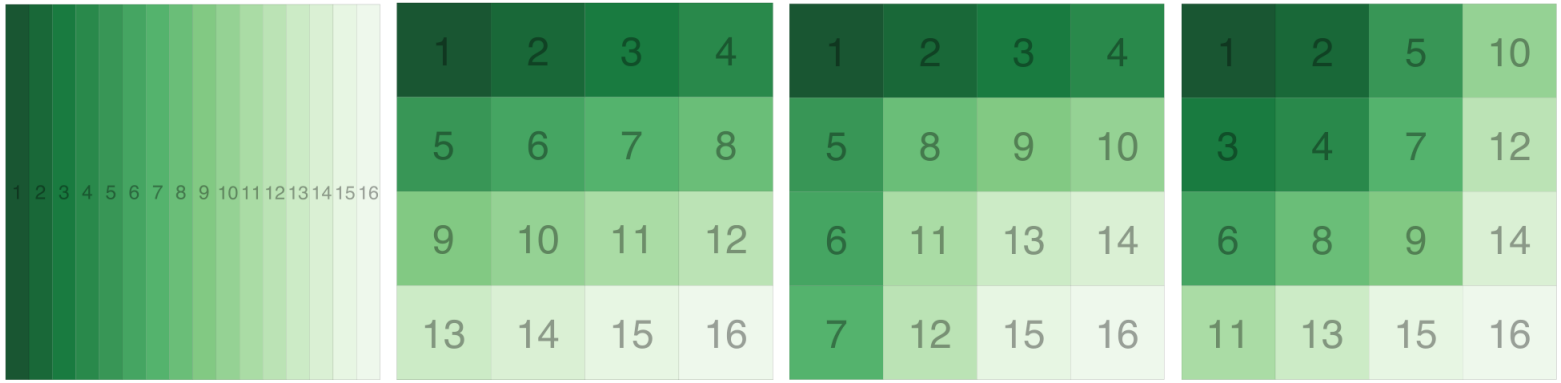
- President Obama Hails Passage of Health Care Bill
- Health Bill Taxes Drug, Device Makers and the Rich
- Stock Screen: 3 Stocks With Big Dividends and Buybacks

Patent No.: US 6,583,794 B1

Click Here to License the Map Applet

ORDERED TREEMAP

several algorithms in comparison



slice and dice

B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. ACM Transactions on Graphics, 11:92–99, 1992.

strip

B. B. Bederson, B. Shneiderman, and M. Wattenberg. Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. ACM Transactions on Graphics, 21:833–854, 2002.

squarified

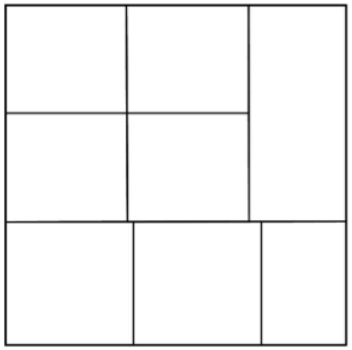
M. Bruls, K. Huizing, and J. van Wijk. Squarified treemaps. EuroVis, pages 33–42, 2000.

ordered squarified

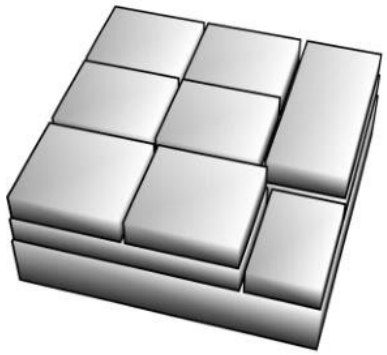
B. Shneiderman and M. Wattenberg. Ordered treemap layouts. In Infovis01, pages 73–78, 2001.

Readability scores: 1.0, 0.625, 0.375, 0.125 (1 = no angular change, 0=every jump between sequential nodes requires an abrupt angular change

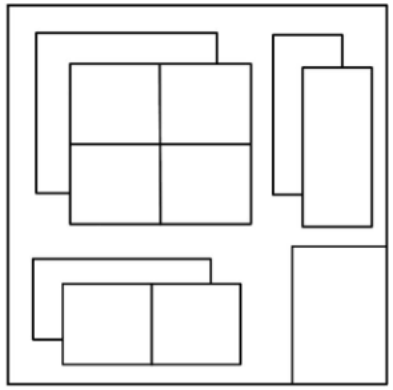
OTHER



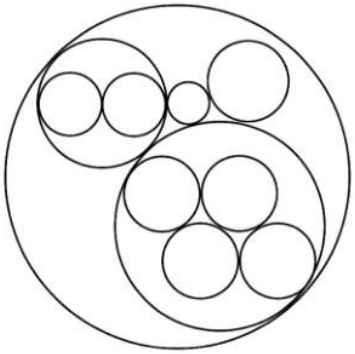
original squarified:
emphasizes leafs and their attributes



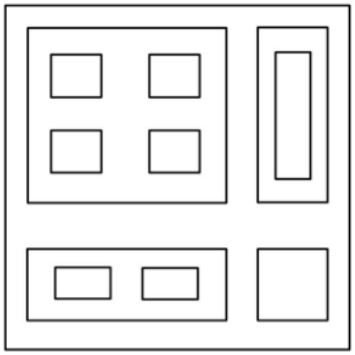
steptree:
emphasizes structure with extrusion



cascaded layout:
emphasizes structure with overlap



circular treemap:
emphasizes structure with
non-space-filling primitive

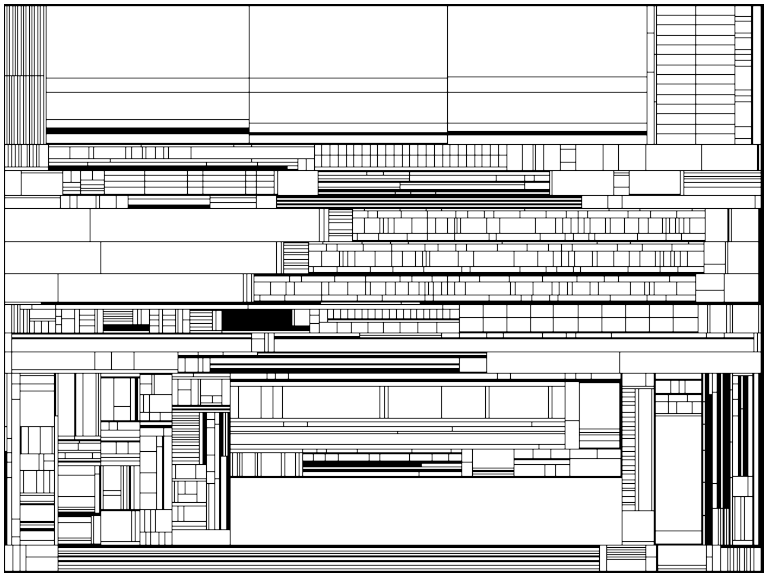


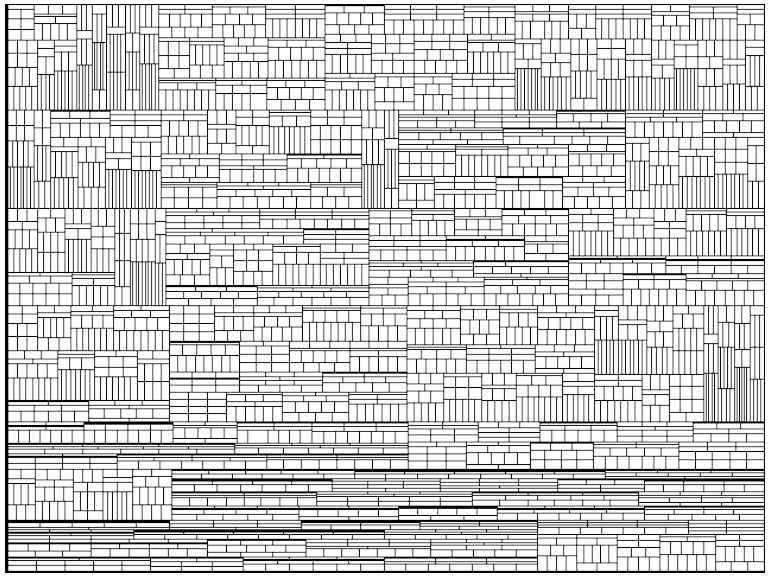
nested layout:
emphasizes structure with whitespace

TVCG, 17(4), The Design Space of Implicit Hierarchy Visualization: A Survey

Treemap results

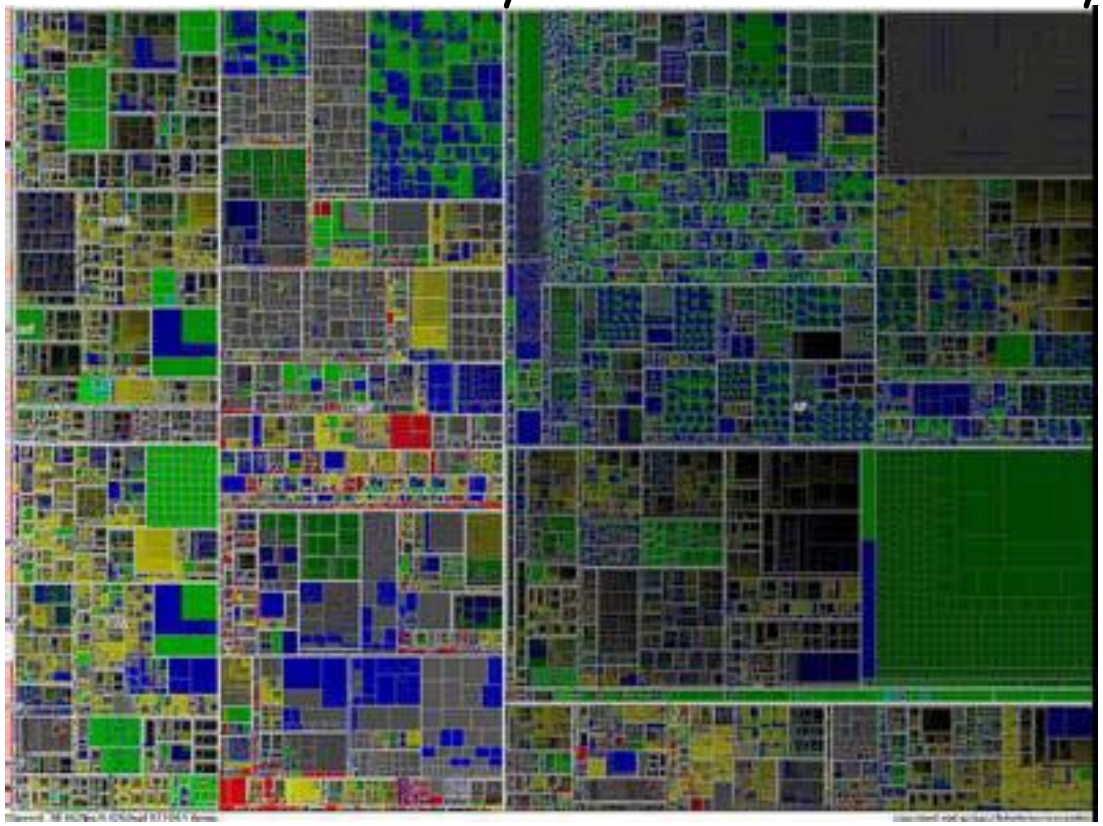
- Treemaps are great at representing sizes and using screen space, but hierarchical structure is less obvious.
- Examples:
  - Treemaps of file system & organizational structure.





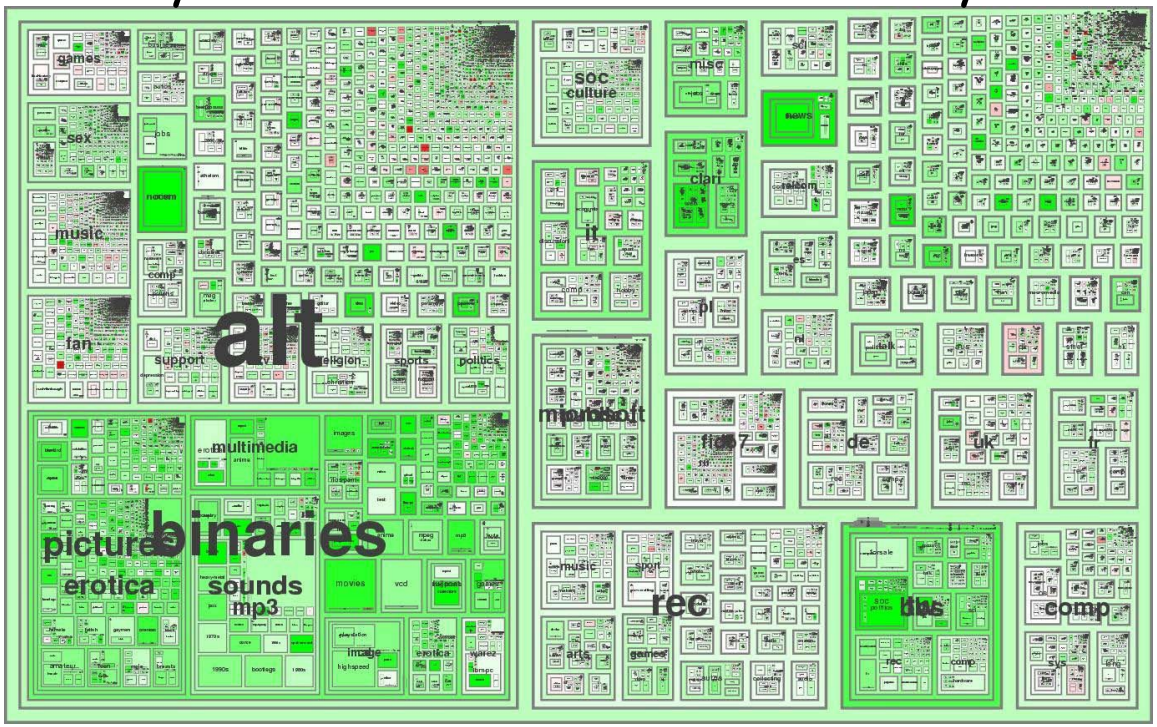
Border width

- The borders' widths may indicate the hierarchy level:



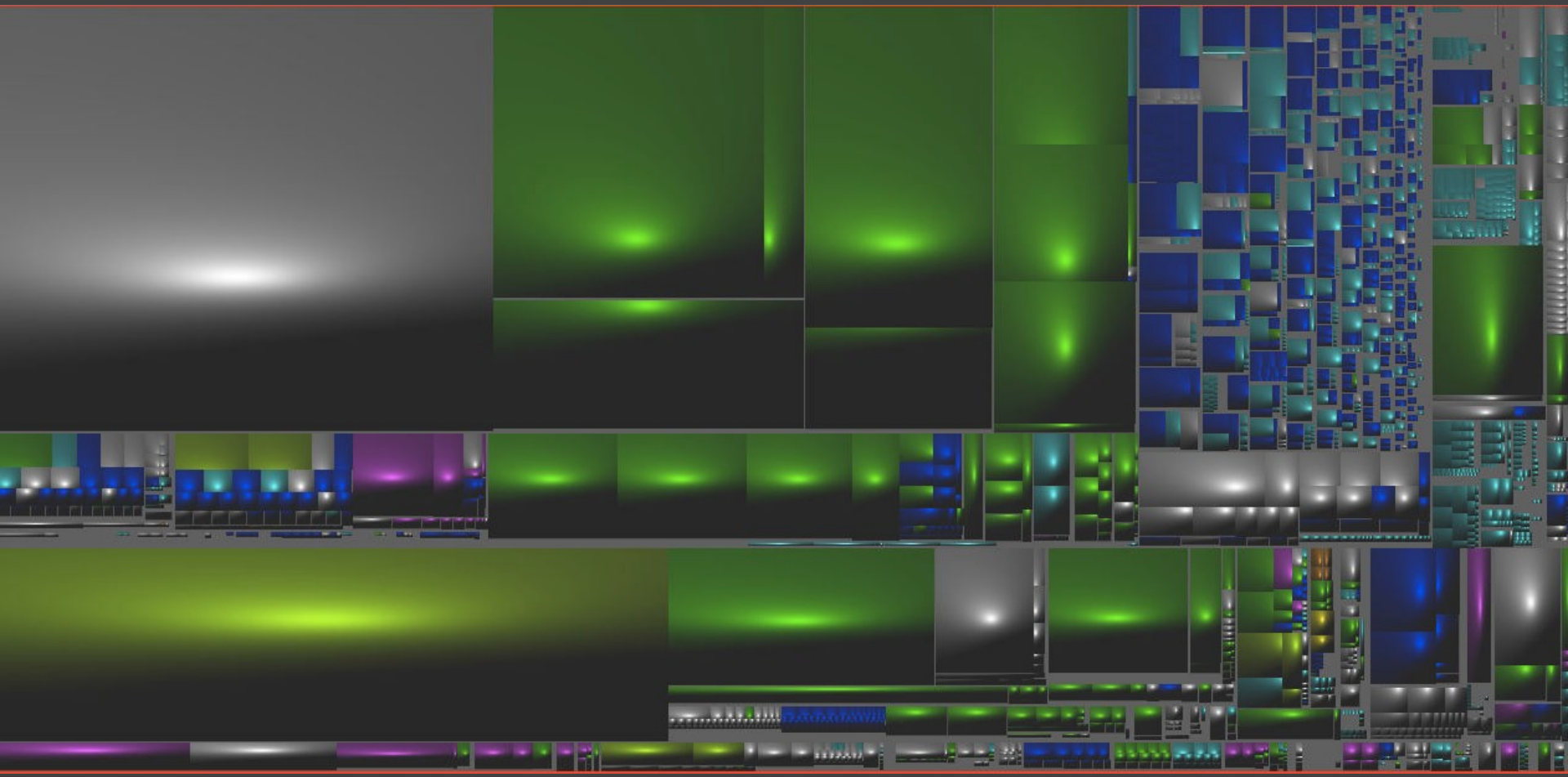
Frames

- Frames may be used to make the hierarchy more explicit:



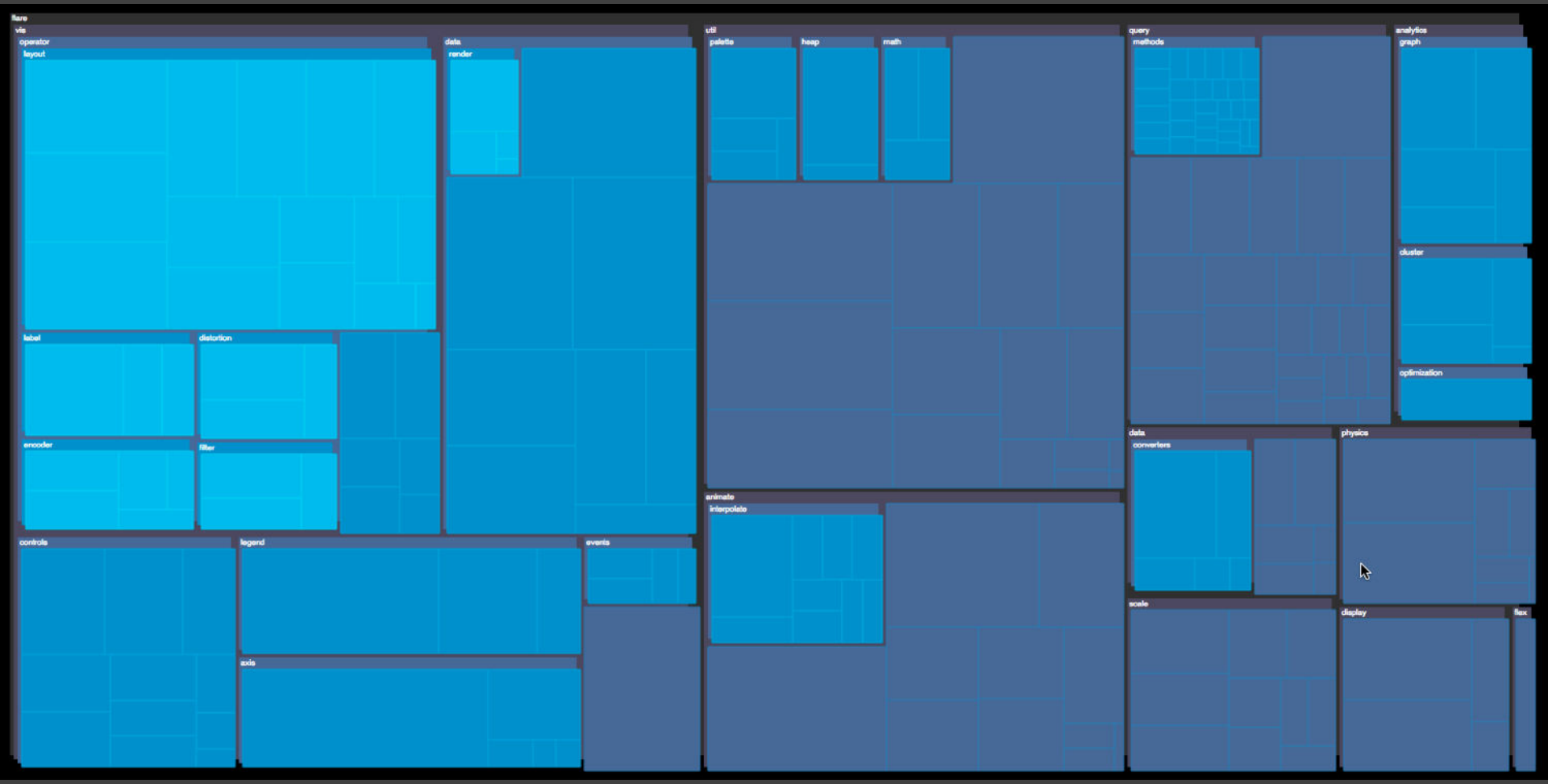
- Frames take away screen space (especially for many levels).

Cushion Treemaps [van Wijk & Wetering ’99]



Uses shading to emphasize hierarchal structure.

Cascaded Treemaps [Lü & Fogarty '08]

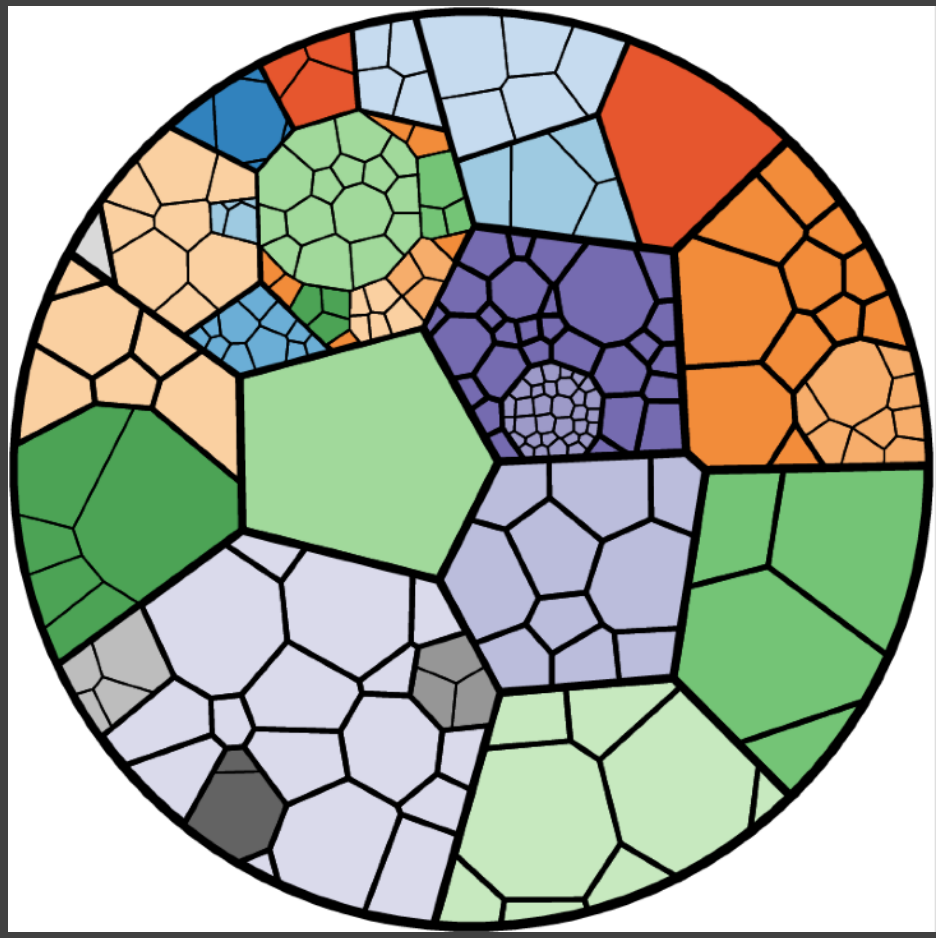


Uses 2.5D effect to emphasize hierarchy relations.

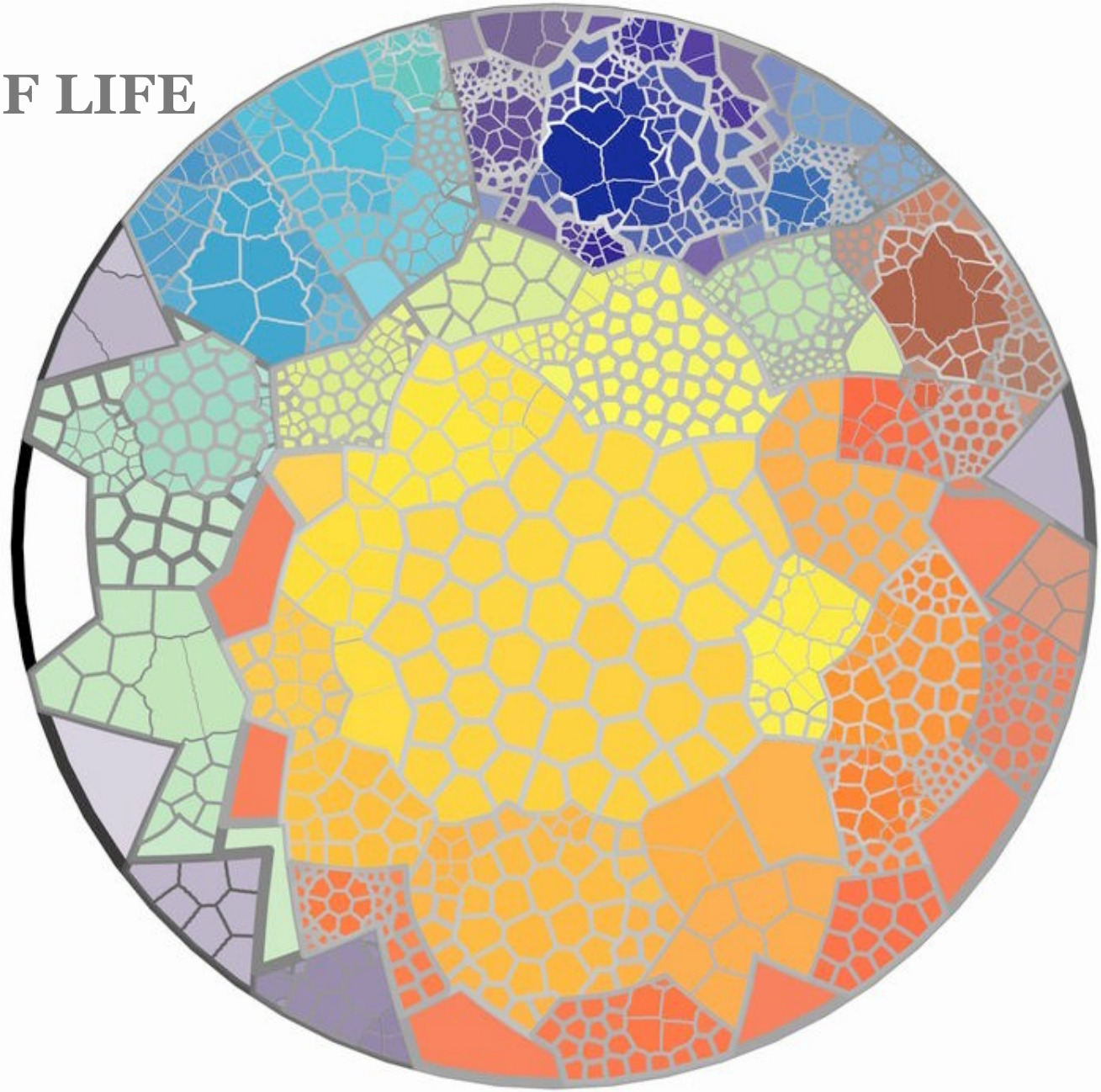
Voronoi Treemaps [Balzer et al. ’05]

Instead of rectangles, create treemaps with arbitrary polygonal shapes and boundary.

Use iterative, weighted Voronoi tessellations to achieve cells with value-proportional areas.



TREE OF LIFE
mammals



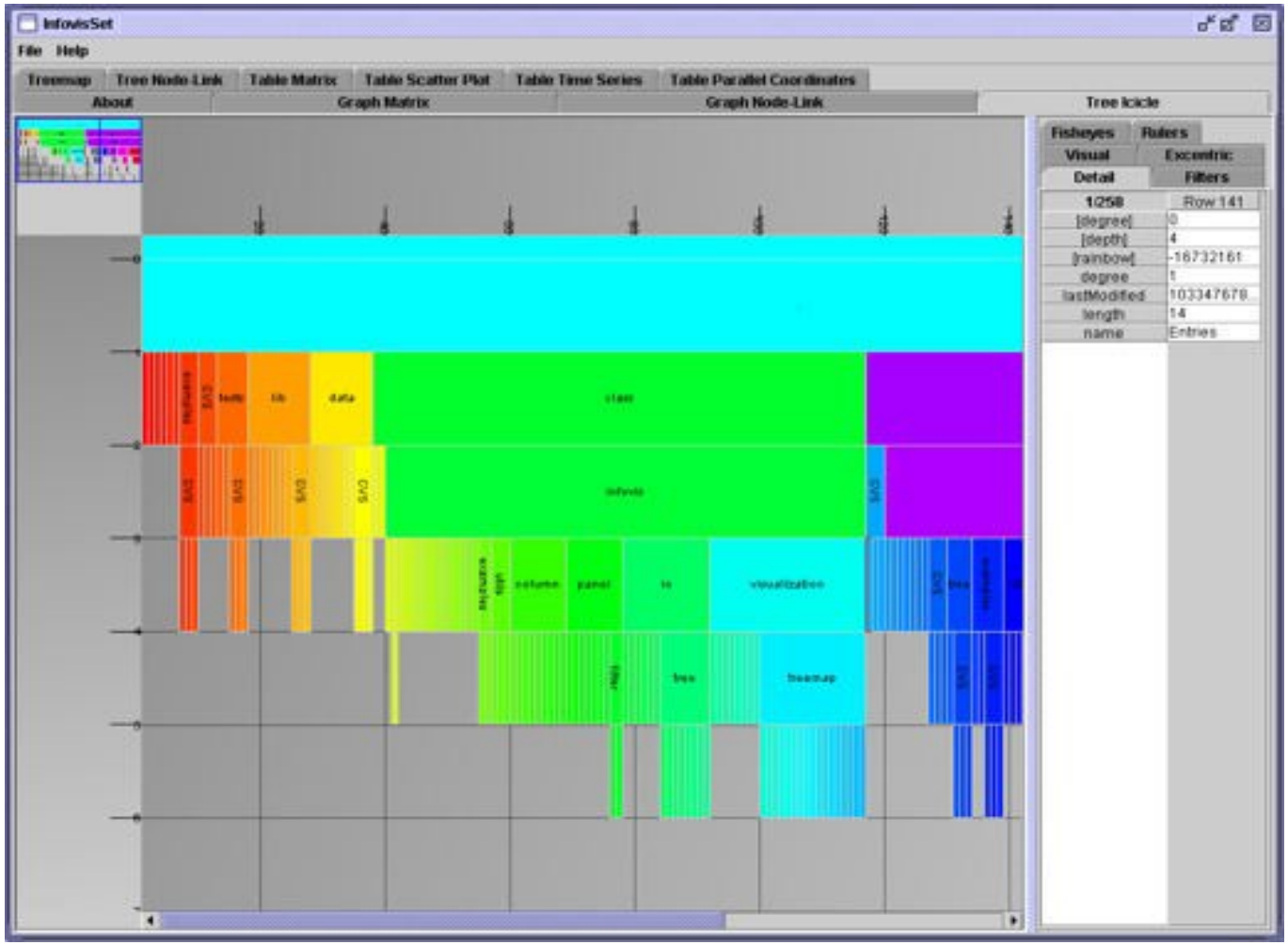
http://www.flickr.com/photos/arenamontanus/2037614308/sizes/o/in/photostream/

Layering

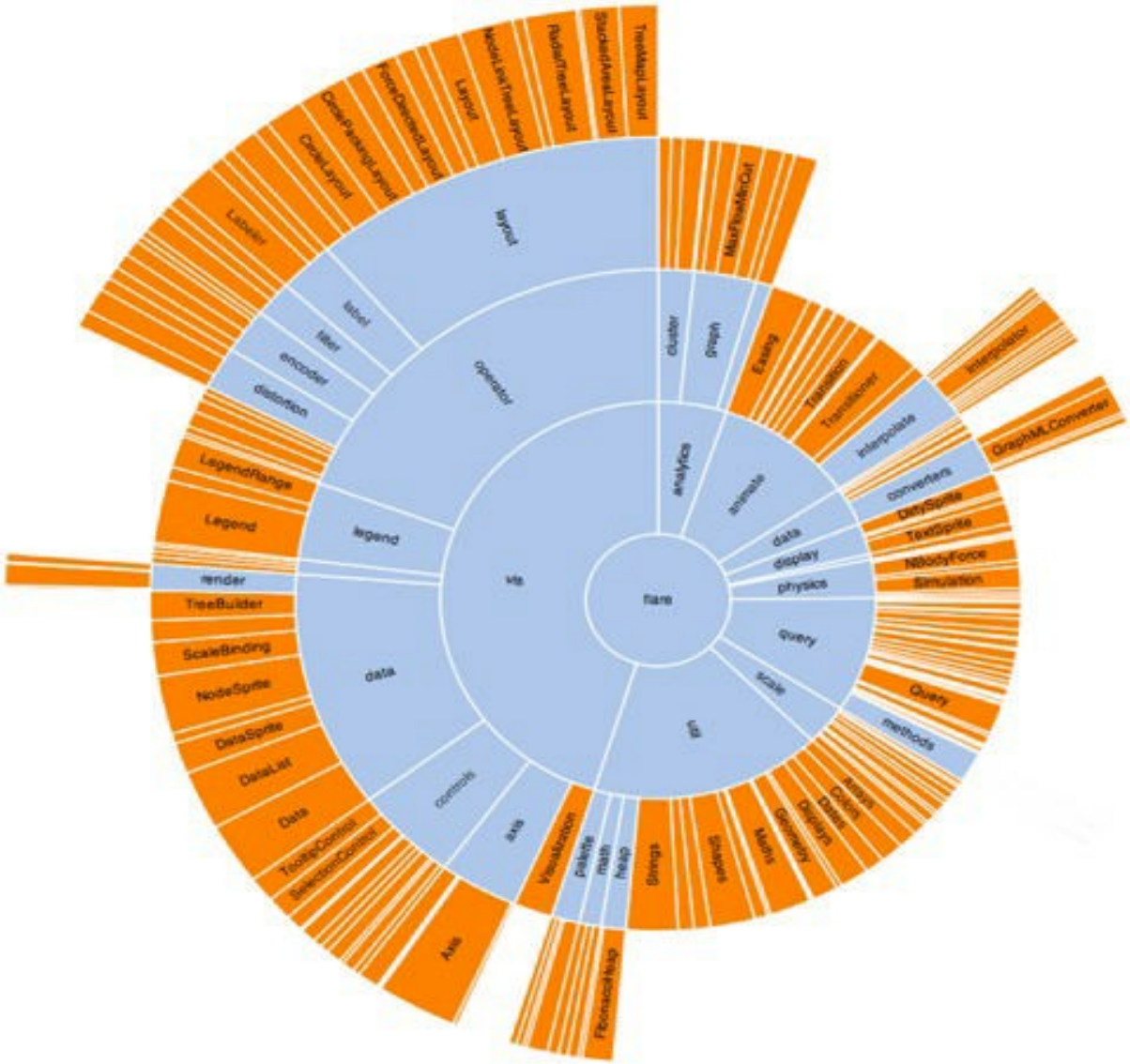
LAYERED DIAGRAMS

- **similar to node-link layouts without edges**
  - structured encoded using:
    - *layering*
    - *adjacency*
    - *alignment*
- **recursive subdivision of space**
- **apply same set of approaches as in node-link layout**

ICICLE TREES



SUNBURST TREES



Summary: NodeLink vs SpaceFilling

- Node-link diagrams or space-filling techniques?
- It depends on the properties of the data
  - Node-link typically better at exposing structure of information structure
  - Space-filling good for focusing on one or two additional variables of cases