CIS 4930/6930-902
Scientific Visualization

Trees & Graphs

Paul Rosen
Assistant Professor
University of South Florida

slides credits Alex Bigelow (U of Utah), Miriah Meyer (U of Utah), Hanspeter Pfister (Harvard), Jeff Heer (Stanford)
ADMINISTRATIVE

project 3 due on Tuesday
project 4 handed out by Tuesday

Read: Elzen and van Wijk, Multivariate Network Exploration and Presentation: From Detail to Overview via Selections and Aggregations (video)
dataset types

<table>
<thead>
<tr>
<th>Tables</th>
<th>Networks &amp; Trees</th>
<th>Fields</th>
<th>Geometry</th>
<th>Clusters, Sets, Lists</th>
</tr>
</thead>
<tbody>
<tr>
<td>Items</td>
<td>Items (nodes)</td>
<td>Grids</td>
<td>Items</td>
<td>Items</td>
</tr>
<tr>
<td>Attributes</td>
<td>Links</td>
<td>Positions</td>
<td>Positions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Attributes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Multidimensional Table**: A table with rows and columns, where items are in rows and attributes are in columns.
- **Trees**: A network of nodes (items) and links (connections between items).
- **Grids**: A grid of positions, where each cell represents an item with attributes.
- **Clusters, Sets, Lists**: A collection of items with positions.

- **Attributes (columns)**: Values associated with items in tables or nodes in trees.
- **Value in cell**: The attribute value stored in each cell of a grid.
- **Cell containing value**: The specific cell in a grid or table where an attribute value is stored.
DEFINITIONS
A graph $G$ consists of:

- $V$ - a collection of vertices (or nodes)
- $E$ - a set of edges consisting of vertex pairs

An edge $e_{xy} = (x, y)$ connects two vertices $x$ and $y$

**Example**

$V = \{1, 2, 3, 4\}$

$E = \{(1, 2), (1, 3), (2, 3), (3, 4), (4, 1)\}$
### Nodes

<table>
<thead>
<tr>
<th>ID</th>
<th>Attribute 1</th>
<th>Attribute 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.4</td>
<td>Good</td>
</tr>
<tr>
<td>2</td>
<td>5.8</td>
<td>Bad</td>
</tr>
<tr>
<td>3</td>
<td>1.1</td>
<td>Ugly</td>
</tr>
<tr>
<td>4</td>
<td>-3.5</td>
<td>Really Ugly</td>
</tr>
</tbody>
</table>

### Edges

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th>Attribute 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>200</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>150</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>250</td>
</tr>
</tbody>
</table>
GRAPHS & TREES

graphs
model relationships about data
nodes and edges

trees
graphs with hierarchical structure
nodes as parents and children
A BUNCH OF DEFINITIONS

- A directed graph
- An undirected graph
- Weighted
- Unconnected
- Node degrees
- A cycle
- An acyclic graph
- A connected acyclic graph, a.k.a. a tree
- A rooted tree or hierarchy
- Node depths
VISUALIZING TREES
ROOTED TREES

recursion makes it elegant and fast to draw trees

approaches:

- node link
- layered
- indentation
- enclosure
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.
REINGOLD-TILFORD

REPEATEDLY DIVIDE SPACE FOR
SUBTREES BY LEAF COUNT

breadth of tree along one dimension
depth along the other dimension
REINGOLD-TILFORD

**goal**

make smarter use of space
maximize density and symmetry
REINGOLD-TILFORD

design concerns
  clearly encode depth level
  no edge crossings
  isomorphic subtrees drawn identically
  compact
REINGOLD-TILFORD approach
bottom up recursive approach
for each parent make sure every subtree is drawn
pack subtrees as closely as possible
center parent over subtrees
LAYERED DIAGRAMS

recursive subdivision of space
structure encoded using:
layering
adjacency
alignment
<table>
<thead>
<tr>
<th>Flare</th>
<th>vgs</th>
<th>operator</th>
<th>Layout</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>data</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>controls</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>legend</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>axis</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>util</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Strings</td>
<td></td>
</tr>
<tr>
<td></td>
<td>animate</td>
<td></td>
<td>Interpolate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Transitioner</td>
</tr>
<tr>
<td></td>
<td>query</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>analytics</td>
<td></td>
<td>graph</td>
</tr>
<tr>
<td></td>
<td>scale</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>data</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>physics</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>display</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

HTTP://MBOSTOCK.GITHUB.IO/D3/TALK/20111018/PARTITION.HTML
SCALE PROBLEM

tree breadth often grows exponentially—quickly run out of space!

solutions
scrolling or panning
filtering or zooming
hyperbolic layout
INDENTATION

indentation used to show parent/child relationships

breadth and depth contend for space

problem: often requires a great deal of scrolling
ENCLOSURE DIAGRAMS

encode structure using spatial enclosure
often referred to as treemaps
TREEMAPS

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
VISUALIZING GRAPHS
ADMINISTRATIVE

project 4 questions?

project 3 presentations
visual complexity

Search the VC database:

Latest Projects:

Filter by:
- Art (62)
- Biology (52)
- Business Networks (29)
- Computer Systems (33)
- Food Webs (8)
- Internet (30)
- Knowledge Networks (118)
- Multi-Domain Representation (62)
- Music (35)
- Others (53)
- Pattern Recognition (28)
- Political Networks (22)
- Semantic Networks (30)
- Social Networks (101)
- Transportation Networks (49)
- World Wide Web (94)

See All (772)
VISUALIZING GRAPHS

node link layouts
  Reingold-Tilford (discussed previously)
  Sugiyama (directed acyclic graphs)
  Force directed
  Attribute-based

adjacency matrices

aggregate views
  Motif Glyphs
  PivotGraph
SPATIAL LAYOUT

primary concern of graph drawing is the spatial layout of nodes and edges

often (but not always) the goal is to effectively depict the graph structure

connectivity, path-following
network distance
clustering
ordering (e.g., hierarchy level)
What is the depth of V7M?
SUGIYAMA

+ nice, readable top down flow
+ relatively fast (depending on heuristic used for crossing minimization)

- not really suitable for graphs that don’t have an intrinsic top down structure
- hard to implement

use free graphviz lib instead: http://www.graphviz.org
FORCE-DIRECTED

no intrinsic layering, now what?

physically-based model

edges = springs
nodes = repulsive particles
FORCE MODEL

many variations, but usually physical analogy:

**repulsion:** $f_R(d) = C_R \times m_1 \times m_2 / d^2$
- $m_1, m_2$ are node masses
- $d$ is distance between nodes

**attraction:** $f_A(d) = C_A \times (d - L)$
- $L$ is the rest length of the spring
- i.e. Hooke’s Law

total force on a node $x$ with position $x'$
- $\sum_{\text{neighbors}(x)}: f_A(||x' - y'||) \times (x' - y') + -f_R(||x' - y'||) \times (x' - y')$
ALGORITHM

start from random layout

(global) loop:
  for every node pair compute repulsive force
  for every edge compute attractive force
  accumulate forces per node
  update each node position in direction of accumulated force

stop when layout is ‘good enough’
FORCE DIRECTED

+ very flexible, aesthetic layouts on many types of graphs
  + can add custom forces
  + relatively easy to implement

- repulsion loop is $O(n^2)$ per iteration
  can speed up to $O(n \log n)$ using quadtree or k-d tree
  - prone to local minima
    can use simulated annealing
OTHER LAYOUTS

- **orthogonal**
  - great for UML diagrams
  - algorithmically complex

- **circular layouts**
  - emphasizes ring topologies
  - used in social network diagrams

- **nested layouts**
  - recursively apply layout algorithms
  - great for graphs with hierarchical structure
The Open Graph Viz Platform

Gephi is an interactive visualization and exploration platform for all kinds of networks and complex systems, dynamic and hierarchical graphs.

Runs on Windows, Linux and Mac OS X. Gephi is open-source and free.

Gephi 0.8 beta has been released! Discover a new Preview and dynamic features, start building commercial applications with the new open source license.

APPLICATIONS

- Exploratory Data Analysis: intuition-oriented analysis by networks manipulations in real-time.
- Link Analysis: revealing the underlying structures of associations between objects, in particular in scale-free networks.
- Social Network Analysis: easy creation of social data connectors to map community organizations and small-world networks.
- Biological Network analysis: representing patterns of biological data.
- Poster creation: scientific work promotion with high-quality printable maps.

"Like Photoshop™ for graphs."

— the Community

PAPERS

LATEST NEWS

- Weekly news
- Annual report 2011
- Gephi Nesaj presentation at FOSDEM
- Gephi meet-up #4 in Berlin
- Introducing the Gephi Plugins Bookcamp

Download FREE Gephi 0.8 beta

Release Notes | System Requirements

Features | Quick start | Screenshots | Videos
ATTRIBUTE-DRIVEN LAYOUT

large node-link diagrams get messy!
are there additional structures to exploit?

idea: use data attributes to perform layout
  e.g., scatterplot based on node values
dynamic queries and/or brushing can be used
to enhance perception of connectivity
CEREBRAL
NODE LINK

+ understandable visual mapping
  + can show overall structure, clusters, paths
  + flexible, many variations

- all but the most trivial algorithms are $> O(n^2)$
- not good for dense graphs
  hairball problem!
ALTERNATIVE: ADJACENCY MATRIX

instead of node link diagram, use adjacency matrix representation
SPOTTING PATTERNS IN MATRICES
LES MISÉRABLES

character co-occurrence

HTTP://BOST.OCKS.ORG/MIKE/MISERABLES/
**Adjacency Diagram**

- great for dense graphs
- visually scalable
- can spot clusters

- row order affects what you can see
- abstract visualization
- hard to follow paths
AGGREGATE VIEWS
MOTIF GLYPHS

Connector

Clique

Fan
MOTIF GLYPHS
CRITIQUE

When *can* you use this technique?
When *should* you use this technique?
Design and Update of a Classification System: The UCSD Map of Science

Katy Börner,1,2∗, Richard Klavans3, Michael Patek3, Angela M. Zoss1, Joseph R. Biberstine1, Robert P. Light1, Vincent Larivière1,2, Kevin W. Boyack2

1) Laboratory for Network Science and Innovation, School of Library and Information Science, Indiana University, Bloomington, Indiana, United States of America; 2) Royal Netherlands Academy of Arts and Sciences (KNAW), Amsterdam, The Netherlands; 3) SciTech Strategies Inc., Bensalem, Pennsylvania, United States of America; 4) École de Bibliothéconomie et des Sciences de l'Information, Université de Montréal, Montréal, Canada; 5) Direction des Sciences et des Technologies (DS3), Centre Interuniversitaire de Recherche sur la Science et la Technologie (CIRST), Université du Québec à Montréal, Montréal, Canada; 6) SciTech Strategies Inc., Albuquerque, New Mexico, United States of America

Abstract

Global maps of science can be used as a reference system to chart career trajectories, the location of emerging research frontiers, or the expertise profiles of institutes or nations. This paper details data preparation, analysis, and layout performed when designing and subsequently updating the UCSD map of science and classification system. The original classification...
CIRCULAR PARAMETERIZATION

topological analysis tool that finds circles in high-dimensional data and visualizes results
DISCUSSION

Elzen and van Wijk,

Multivariate Network Exploration and Presentation: From Detail to Overview via Selections and Aggregations

(video)
RECAP
TREES

indentation
simple, effective for small trees

node link and layered
looks good but needs exponential space

enclosure (treemaps)
great for size related tasks but suffer in structure related tasks
GRAPHS

node link
familiar, but problematic for large or dense graphs

adjacency matrices
abstract, hard to follow paths

aggregation can help
not always possible, not always appropriate

extracting structure can help
unclear how crosscutting it will be
TAKE HOME MESSAGE: NO BEST SOLUTION