Large Scale Information Visualization

Jing Yang
Fall 2008

Review: Graph Visualization
When?

- Ask the question:

  - Is there an inherent relation among the data elements to be visualized?

- If yes -> data: nodes
  - relations: edges

Graph Usage

- In information visualization, many data sets can be modeled as a graph
  - US telephone system
  - World Wide Web
  - Distribution network for on-line retailer
  - Call graph of a large software system
  - Semantic map in an AI algorithm
  - Set of connected friends
Example: Social Network Visualization

- Vizster: Visualizing Online Social Networks [Heer Infovis 05]
- Online social networks – millions of members publicly articulate mutual “friendship” relations
  - Friendser.com, Tribe.net, and orkut.com
- Vizster
  - Playful end-user exploration and navigation of large-scale online social networks
  - Explore connectivity, support visual search and analysis, and automatically identifying and visualizing community structures
  - Video

Terminology and Concepts of Graph Theory
Graph-Theoretic Data Structures

- List structures
- Matrix structures

Incidence Matrix

- Incidence matrix – nodes: rows, edges: columns, 1: related, 0: unrelated

figure from http://mathworld.wolfram.com/IncidenceMatrix.html
Adjacency Matrix

- Adjacency matrix - N by N matrix, where N is the number of vertices in the graph. If there is an edge from some vertex x to some vertex y, then the element $M_{x,y}$ is 1, otherwise it is 0.

<table>
<thead>
<tr>
<th>Labeled graph</th>
<th>Adjacency matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="figure from wikipedia" /></td>
<td><img src="matrix" alt="adjacency matrix" /></td>
</tr>
</tbody>
</table>

Graph, Simple Graph, Degree, Density

- A graph G consists of two sets of information:
  - a set of nodes $N = \{n_1, n_2, ..., n_g\}$
  - a set of lines $L = \{l_1, l_2, ..., l_L\}$
    - Each line is an unordered pair of distinct nodes, $l_k = (n_i, n_j)$
- Simple graph: a graph that has no loops and includes no more than one line between a pair of nodes (default)
- Nodal degree: the degree of a node $d(n_i)$ is the number of lines that are incident with it.
- Density of graph: the proportion of possible lines that are actually present in the graph $\frac{L}{((g(g-1))/2)}$
Directed Graph (Digraph)

- A directed graph $G$ consists of two sets of information:
  - a set of nodes $N = \{n_1, n_2, \ldots, n_g\}$
  - a set of arcs $L = \{l_1, l_2, \ldots, l_L\}$
    Each line is an ordered pair of distinct nodes, $l_k = \langle n_i, n_j \rangle$
- Indegree: the number of arcs terminating at the node
- Outdegree: the number of arcs originating with the node

Subgraph

- A graph $G_s$ is a subgraph of $G$ if the set of nodes of $G_s$ is a subset of the set of nodes of $G$, and the set of lines in $G_s$ is a subset of the lines in the graph $G$. 
Walks, Trails, and Paths

- Walks: a sequence of nodes and lines, starting and ending with nodes, in which each node is incident with the lines following and preceding it in the sequence
- Trails: a walk in which all of the lines are distinct, though some nodes may be included more than once
- Path: a walk in which all nodes and all lines are distinct

Connected Graphs and Components

- Connected graph: a graph is connected if there is a path between every pair of nodes in the graph
-Disconnected graph: a graph that is not connected
- Component: a maximal connected subgraph of a graph
-Maximal entity: one that cannot be made larger and still retain its property
Geodesics, Distance, Diameter

- Geodesic: a shortest path between two nodes
- Geodesic distance (distance): the length of a geodesic between two nodes
- Eccentricity (association number): the largest geodesic distance between that node and any other node
- Diameter: the largest geodesic distance between any pair of nodes in a graph
- Small world graph: a graph has a small diameter compared to the number of nodes and exhibits a local cluster structure

Structural and Locational Properties

- Prominent node in a social network: the ties of the actor (node) makes the actor particularly visible to the other actors in the network
- Graph - centrality:
  - Degree centrality (many ties)
  - Closeness centrality (quickly interact with all others)
  - Betweenness centrality (actors in the middle, control the communication)
Graph Visualization Techniques

Techniques:
- Node-link diagrams,
- Adjacency matrices.

Key issues:
- The size of the graph to view
  - performance
  - viewability, usability
  - Comprehension and detailed analysis

Node-Link Diagrams
Challenges

- Graph layout and positioning
  - Make a concrete rendering of abstract graph
- Navigation/Interaction
  - How to support user changing focus and moving around the graph

Graph Layout Algorithms

- Entire research community’s focus
- Good references:
  - Tutorial (talk slides)
Aesthetic Considerations

- **Crossings** -- minimize towards planar
  - A graph is planar if it has an intersection free 2D drawing
- **Total Edge Length** -- minimize towards proper scale
- **Area** -- minimize towards efficiency
- **Maximum Edge Length** -- minimize longest edge
- **Uniform Edge Lengths** -- minimize variances
- **Total Bends** -- minimize orthogonal towards straight-line

Vertex Issues

- Shape
- Color
- Size
- Location
- Label
Edge Issues

- Color
- Size
- Label
- Form
  - Polyline, straight line, tube, orthogonal, grid, curved, planar, upward/downward, ...

General GD Information

- Good web links
  - www.cs.brown.edu/people/rt/gd.html
  - www.research.att.com/sw/tools/graphviz/
  - rw4.cs.unisb.de/users/sander/html/gstools.html
Existing frameworks

- Tulip (University of Bordeaux – France),
- Pajek (University of Ljubljani – Slovenia),
- GraphViz (AT&T),
- JUNG (University of California, Irvine).
Forth Directed Graph Drawing

- Force-directed layout schemes are usually selected for undirected graphs, this being ideal for simulating physical and chemical models.
  - **Spring forces**
    A spring embedder is simulated. The nodes of a graph are regarded as electrically charged particles that repel one another, the edges being regarded as springs connecting the particles. Particles that are far away from one another attract each other by spring forces, particles that are too close repel one another.

Forth Directed Graph Drawing

- **Magnetic forces**
  In directed graphs all edges should have a uniform direction to point in. Here the edges are interpreted as magnetic needles that align themselves according to a magnetic field.
- **Gravitational forces**
  In unconnected graphs simulating a spring embedder makes unconnected nodes move away from one another as there are only repulsive forces but no attractive forces. That is why gravitational forces are introduced. All nodes are attracted to the bary center of all the other nodes.

Matrix-based Graph Visualization

A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations N. Henry et al. Infovis 2004

Review: Hierarchy and Tree Visualization
Hierarchies

Definition

- An ordering of groups in which larger groups encompass sets of smaller groups.

- Data repository in which cases are related to subcases

Hierarchies in the World

- Family histories, ancestries
- File/directory systems on computers
- Organization charts
- Object-oriented software classes
Good Hierarchy Visualization

- Allow adequate space within nodes to display information
- Allow users to understand relationship between a node and its context
- Allow to find elements quickly
- Fit into a bounded region
- Much more

Trees

- Hierarchies are often represented as trees
  - Directed, acyclic graph
- Two major categories of tree visualization techniques:
  - Node-link diagram
    - Visible graphical edge from parents to their children
  - Space-filling
Node-Link Diagrams

Put Root at Top or Left
The Challenges

- **Scalability**
  - # of nodes increases exponentially
  - Available space increases polynomially (circular case)
- Showing more attributes of data cases in hierarchy or focusing on particular applications of trees
- Interactive exploration
3D Approach 1 - 3D Tree

Tavanti and Lind, InfoVis 01

3D Approach 2 - Cone Tree

Robertson, Mackinlay, Card CHI ’91
Advantages vs. Limitations

- **Positive**
  - More effective area to lay out tree
  - Use of smooth animation to help person track updates
  - Aesthetically pleasing

- **Negative**
  - As in all 3D, occlusion obscures some nodes
  - Non-trivial to implement and requires some graphics horsepower

---

Hyperbolic Browser

- **Key idea:**
  - Find a space (hyperbolic space) that increases exponentially, lay the tree on it
  - Transform from the hyperbolic space to 2D Euclidean space

---

J. Stasko’s InfoVis class slides

Hyperbolic space background

geometry with exponential "amount of room"
- good match for exponential node count of trees

2D hyperbolic plane

hemisphere area

hyperbolic: exponential
\[ 2\pi \sinh^2(r) \]

euclidean: polynomial
\[ 2\pi r^2 \]

[Thurston and Weeks 84]

http://graphics.stanford.edu/~munzner/talks/calgary02

1D hyperbolic space

hyperbola projects to line

image plane
eye point
Hyperbolic Browser

R. Spence. Information Visualization
Change Focus

Key Attributes

- Natural magnification (fisheye) in center
- Layout depends only on 2-3 generations from current node
- Smooth animation for change in focus
- Don’t draw objects when far enough from root (simplify rendering)
H3 Browser

- Use hyperbolic transformation in 3D space

Tamara Munzner: H3: laying out large directed graphs in 3D hyperbolic space.
INFOVIS 1997: 2-10

Scalability - Model Selection (1)

Projective model: keeps lines straight but distorts angles.
Conformal model: preserves angles but maps straight lines to circular arcs.

From Tamara Munzner’s Ph.D. dissertation
Scalability - Model Selection (1)

Projective vs. Conformal Model

- Projective model
  - Less aesthetically pleasing 😞
  - Transformation: 4X4 matrices 😊
  - Straight lines 😊
- Conformal model
  - More aesthetically pleasing 😊
  - Transformation: 2X2 complex matrices 😊
  - Curves 😞

Scalability - Layout (1)

- Find a spanning tree from an input graph
  - Use domain-specific knowledge
- Layout algorithm
  - Nodes are laid out on the surface of hemispheres
  - A bottom-up pass to estimate the radius needed for each hemisphere
  - A top-down pass to place each child node on its parental hemisphere's surface
Scalability - Layout (2)

- Lays out child nodes on the surface of a hemisphere like sprinkles on an ice cream cone.

Left: cone tree approach      Middle and right: H3 approach

Scalability - Layout (3)

- An approximate layout is fine, whereas a perfect but slow iterative solution would be inappropriate.
Scalability - Adaptive Drawing (1)

- Maintain a target frame rate (movie 3)
  - Draw only as much of the neighborhood around a center point as is possible in the allotted time
    - Unterminated links
  - Fill in scene fringe using several bounded idle frames when the user is idle

Scalability - Adaptive Drawing (2)

Active mode - when the user is active dragging the mouse, or during animated transitions.

- Initialization: Initialize ActionQueue with a single node with the largest projected screen area in the previous frame (it is close to the ball's center)

- Draw Loop
  - Current node is popped off the ActionQueue.
  - Handle the current node.
  - Handle the nodes one hop away from the current one in the spanning tree: the parent node and the children nodes.
  - If the projected screen area of any of these neighboring nodes is at least one pixel, insert it into the ActionQueue, maintaining sorted order.

- Terminate if: No more time left, or ActionQueue is empty
Scalability - Adaptive Drawing (3)

Idle mode - when the user stops dragging the mouse, or an animated transition ends

- **Initialization**: ActionQueue from the previous frame is left untouched.
- **Draw Loop Termination**: Terminate if either: No more time left, or ActionQueue is empty.
- Several idle frames can be drawn back to back if no input is found
  - Why?

Scalability - Other Tricks

- Only draw a local neighborhood of nodes
  - Nodes sufficiently far from the center will project to less than a single pixel – terminate drawing when features project to subpixel areas
- Use front buffer for highlighting
Navigation

Translation of a node to the center

Rotation around the same node (Movie 0)

From Tamara Munzner’s Ph.D. dissertation

Non-Tree Links

Drawing all the non-tree links

Drawing the outgoing non-tree links for the entire subtree beneath the highlighted yellow node

From Tamara Munzner’s Ph.D. dissertation
Problems

- Orientation
  - Watching the view can be disorienting
  - When a node is moved, its children don’t keep their relative orientation to it as in Euclidean plane. They rotate
- Not as symmetric and regular as Euclidean techniques, two important attributes in aesthetics

J. Stasko’s InfoVis class slides

Botanical Tree [E. Kleiberg et. al. InfoVis 2001]

- Botanical tree: The same directory with different settings
Collapsible Cylindrical Tree [Dachselt & Ebert Infovis 01]

- Basic idea: use a set of nested cylinders according to the telescope metaphor
- Limitation: one path is visible in once
- Interactions: rotation, go down/up

Space-Filling Techniques
Space-Filling Techniques

- Each item occupies an area
- Children are “contained” within parent

Visualization of Large Hierarchical Data by Circle Packing W. Wang et al. CHI 2006

Key ideas:
- Tree visualization using nested circles
- Brother nodes represented by externally tangent circles
- Nodes at different levels displayed by using 2D nested circles or 3D nested cylinders
Visualization of Large Hierarchical Data by Circle Packing

W. Wang et al. CHI 2006

Figure 4. Packing 1000 circles with random radii

 Visualization of Large Hierarchical Data by Circle Packing

W. Wang et al. CHI 2006

Figure 6. The visualization of a file system
Visualization of Large Hierarchical Data by Circle Packing  
W. Wang et al. CHI 2006

(a) User Interface and the overview of “D:\MyInfot”

Treemap

- Children are drawn inside their parents
- Alternative horizontal and vertical slicing at each successive level
- Use area to encode other variables of data items

Treemap

Example

Directories

J. Stasko's InfoVis class slides
Treemap Affordances

- It is rectangular! It makes better use of space
- Good representation of two attributes beyond node-link: color and area
- Not as good at representing structure
  - Can get long-thin aspect ratios
  - What happens if it's a perfectly balanced tree of items all the same size?

Aspect ratios

These kinds of rectangles are visually unappealing

Which has bigger area?

J. Stasko's InfoVis class slides
Treemap Variation

Make rectangles more square

Slice-and-dice  Cluster  Squarified

Pivot-by-middle  Pivot-by-size  Strip

Showing Structure

A tree with 698 node (from Balzer:infovis2005)

How about a perfectly balanced binary tree?
Showing Structure

- Borderless treemap: hard to discern structure of hierarchy
  - What happens if it's a perfectly balanced tree of items all the same size?
- Variations:
  - Use border
  - Change rectangles to other forms

Nested vs. Non-nested
Nested Treemap

- Borders help on small trees, but take up too much area on large, deep ones


Cushion Treemap

- Add shading and texture (Van Wijk and Van de Wetering InfoVis'99)
Voronoi Treemaps [balzer:infovis05]

- Enable subdivisions of and in polygons
- Fit into areas of arbitrary shape

Basic Voronoi Tessellations

- Enable partitioning of m-dimensional space without holes or overlappings
- Planar VT in 2D:
  - \( P = \{p_1, \ldots, p_n\} \) a set of \( n \) distinct points –generators
  - Divide 2D space into \( n \) Voronoi regions \( V(P_i) \):
    - Any point \( q \) lies in the region \( V(P_i) \) if and only if
    - \( \text{distance}(p_i, q) < \text{distance}(p_j, q) \) for any \( j \neq i \)
Weighted Voronoi Tessellations

- **Basic VT:** 
  \[ \text{distance}_c(p_i, q) := \|p_i - q\| = \sqrt{(x_i - x)^2 + (y_i - y)^2} \]

- **Additively weighted Voronoi (AW VT):** 
  \[ \text{distance}_{\text{aw}}(p_i, w_i, q) := \|p_i - q\| - w_i \]

- **Additively weighted power voronoi (PW VT):** 
  \[ \text{distance}_{\text{pw}}(p_i, w_i, q) := \|p_i - q\|^2 - w_i \]

---

Centroidal Voronoi Tessellations (CVT)

- **Property of CVT:** Each generator is itself center of mass (centroid) of corresponding voronoi region

---

Figure 6: Voronoi tessellation of 20 random points and an associated CVT—traces illustrate the movements of the points during the computation of the CVT
Centroidal Voronoi Tessellations (CVT)

- CVT minimize the energy function:
  \[ \mathcal{E}(P, \mathcal{V}(P)) = \sum_{i} \int_{V(p_i)} \|x - p_i\|^2 dx \]

- The energy of the CVT is equivalent to the overall aspect ratio of the subareas of the treemap layout

Voronoi Treemap Algorithm

- Size of each Voronoi region should reflect size of the tree node
- Area size is not observed in CVT computation
- Extension:
  - Use iteration
  - In each iteration, adjust the area of regions by their weights
  - Weights are adjusted according to the size of the node
  - Iterate until the relative size error is under a threshold
- Video
Treemap Applications

- Software visualization
- Multimedia visualization
- Tennis matches
- File/directory structures
- Basketball statistics
- Stocks and portfolios

Marketmap

http://www.smartmoney.com/marketmap/
Software Visualization

- SeeSys (Baker & Eick, AT&T Bell Labs)

![Diagram of SeeSys](image)

New code in this release

Figure 2: NCSL and new development by subsystem in a recent release.

Internet News Groups

- Netscan (Fiore & Smith Microsoft)

![Diagram of Netscan](image)
SequoiaView

- File visualizer  www.win.tue.nl/sequoiaview/

Photomesa

- Image browser (quantum and bubble treemap)  http://www.cs.umd.edu/hcil/photomesa/
Space-Filling Techniques

- Each item occupies an area
- Children are “contained” within (under) parent

One Example

Icicle Plot

- Icicle plot (similar to Kleiner and Hartigan’s concept of castles)
  - Node size is proportional to node width

Barlow and Neville InfoVis 2001
Radial Space Filing Techniques

- InterRing [Yang02]

Node Link + Space Filling Techniques
Elastic Hierarchies: Combining Treemaps and Node-Link Diagrams [zhao:infovis 05]

- A hybrid approach
- Dynamic
- Video

Space-Optimized Tree - Motivation

Q. Nguyen and M. Huang Infovis 02
Space-Optimized Tree [Q. Nguyen and M. Huang Infovis 02]

Key idea:
- Partition display space into a collection of geometrical areas for all nodes
- Use node-link diagrams to show relational structure

Algorithm for dividing a region:
1. weight calculation for each direct child
2. wedge calculation for each direct child
3. vertex position calculation for each direct child
Weight Calculation

\[ w(v_i) = 1 + C \sum_{j=0}^{k-1} w(v_{i+j}) \]

- \( V_i \): the direct child
- \( V_{i} - V_{i+k} \): Direct children of \( V_i \)
- Constant \( C \): decide difference between vertexes with more descendants and vertexes with fewer descendants.

Wedge Calculation

\[ \alpha(v_{i+m}) = \frac{A \cdot w(v_{i+m})}{\sum_{j=0}^{k} w(v_{i+j})} \]

Example of dividing the local region of one node
Vertex Position Calculation

Area ABCP = Area AEDP
Vertex is the midpoint of line AP

Space-Optimized Tree

Example: Tree with approximately 55000 nodes