Hierarchies and Trees 1
(Mostly Node-link)

CS 7450 - Information Visualization
November 7, 2016
John Stasko

Learning Objectives

- Define hierarchical data & related terms
  - root, node, link, leaf, depth, parent, child, sibling
- List example tasks for hierarchical data
- Be able to draw reasonable 2D tree
  - Understand basic approach/algorithm and method
- Describe particular drawing techniques and explain +/- of each
  - SpaceTree, Cone Tree, Hyperbolic tree, H3 tree, DOI tree, FlexTree, Space-optimized tree
- Explain general limitations of node-link approach
- Understand treemap algorithm
  - Be able to draw slice-and-dice treemap given a hierarchy
Hierarchies

Definition
- Data repository in which cases are related to subcases
- Can be thought of as imposing an ordering in which cases are parents or ancestors of other cases

Hierarchies in the World

Pervasive
- Family histories, ancestries
- File/directory systems on computers
- Organization charts
- Animal kingdom: Phylum,..., genus,...
- Object-oriented software classes
- ...
Analysis Tasks

- Example tasks?
  - Describe/understand structure
  - Find items
  - What are the parent/children/siblings of x?
  - Where is this subtree?
  - Where are nodes with particular values located?
  - What kind of attributes does this subtree have?
  - ...

"Quiz"

Draw a representation for the following hierarchy:
Node: Child1, Child2, ... (order means nothing)

A: J, H, U, F
J: E, P
H: D, R, L, W, B
F: S, M, N
E: T, K
P: V, C, O, I
S: Q, G
Trees

- Hierarchies often represented as trees
  - Directed, acyclic graph

- Two main representation schemes
  - Node-link
  - Space-filling

Node-Link Diagrams

- Root at top, leaves at bottom is very common
Sample Representation

From: Johnson & Shneiderman, '91


Election '16

Examples

Good for?
Search
Bad for?
Understanding structure

Why Put Root at Top?

Root can be at center with levels growing outward too

Can any node be the root?
Basic Algorithm

- Recursive algorithm
- Height on separate levels
- Width in unique columns
- Make room for subtrees upwards

Potential Problems

- For top-down, width of fan-out uses up horizontal real estate very quickly
  - At level $n$, there are $2^n$ nodes

- Tree might grow a lot along one particular branch
  - Hard to draw it well in view without knowing how it will branch
More Sophisticated

In what way?
• Regions compressed horizontally

Reingold-Tilford Algorithm

Compact layout
Uses symmetry
Depth on levels

Generalized from binary trees by Walker
Running time improved (linear) by Buchheim et al
InfoVis Solutions

- Techniques developed in Information Visualization largely try to assist the problems identified in the last slide
- Alternatively, Information Visualization techniques attempt to show more attributes of data cases in hierarchy or focus on particular applications of trees

Discuss

- How could we do better?
- What design changes could we make to help?
SpaceTree

• Uses conventional 2D layout techniques with some clever additions

Characteristics

• Vertical or horizontal
• Subtrees are triangles
  – Size indicates depth
  – Shading indicates number of nodes inside
• Navigate by clicking on nodes
  – Strongly restrict zooming

Grosjean, Plaisant, Bederson
InfoVis ’02
Design Features

- Make labels readable
- Maximize number of levels opened
- Decompose tree animation
- Use landmarks
- Use overview and dynamic filtering

3D Approaches

- Add a third dimension into which layout can go
- Compromise of top-down and centered techniques mentioned earlier
- Children of a node are laid out in a cylinder "below" the parent
  - Siblings live in one of the 2D planes
Cone Trees

Developed at Xerox PARC

3D views of hierarchies such as file systems

Robertson, Mackinlay, Card
CHI '91

Alternate Views
Cone Trees

• Pros & Cons?
  – Discuss

Cone Trees

• Pros
  – More effective area to lay out tree
  – Use of smooth animation to help person track updates
  – Aesthetically pleasing

• Cons
  – As in all 3D, occlusion obscures some nodes
  – Non-trivial to implement and requires some graphics horsepower
Alternative Solutions

- Change the geometry
- Apply a hyperbolic transformation to the space
- Root is at center, subordinates around
- Apply idea recursively, distance decreases between parent and child as you move farther from center, children go in wedge rather than circle

Hyperbolic Browser

- Focus + Context Technique
  - Detailed view blended with a global view
- First lay out the hierarchy on the hyperbolic plane
- Then map this plane to a disk
- Start with the tree’s root at the center
- Use animation to navigate along this representation of the plane

Lamping and Rao, JVLC '96
2D Hyperbolic Browser

- **Approach:** Lay out the hierarchy on the hyperbolic plane and map this plane onto a display region.

- **Comparison**
  - A standard 2D browser: 100 nodes (w/3 character text strings)
  - Hyperbolic browser: 1000 nodes, about 50 nearest the focus can show from 3 to dozens of characters

Clicking on the blue node brings it into focus at the center
Watch it Work

- Video
- Demo from prefuse system

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)
Problems

• What might be problems with this approach?

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
How about 3D?

- Can same hyperbolic transformation be applied, but now use 3D space?
- Sure can
- Have fun with the math!

H3Viewer

Munzner, IEEE CG&A '98

Video
Old School

- After all the interest in 3D and hyperbolic techniques in the ‘90’s, recently, there has been renewed interest in the old 2D methods (just done better)
  - SpaceTree presented earlier
  - Next 3 papers...

Degree-of-Interest Trees

- Problem: Trees quickly degrade into line

- Approach: Use fisheye-like focus & context ideas to control how a tree is drawn
Approach

- Combine multiple ideas:
  - Expanded DOI computation
  - Logical filtering to elide nodes
  - Geometric scaling
  - Semantic scaling
  - Clustered representation of large unexpended branches
  - Animated transition

Example Operations

1. Display of a uniform tree of 4 levels
2. Same table with focus on Node 3
3. Same tree expanded down to a leaf node
Compression

- For nodes: compress to fit (compress in X or in Y)

| Free layout  
| (~70%)       |
| Compression zone |
| Aggregation zone |

- Within-node compression
  - Data deletion
  - Word abbreviation
  - Node rotation

Better View of Org Chart

Organization chart with over 400 nodes accessible over WWW through Web browser
**FlexTree**

- Horizontally-drawn tree with compression along vertical dimension
- One focus is on showing decision trees well
- Contextual multi-foci view
- Basic idea: Push all nodes down as far as you can

**Example**

![Diagram of FlexTree](image)

*Figure 3: Concept diagram of FlexTree - space between nodes is compressed to achieve a compact view.*
Bar Chart and Partial Views

Figure 4  Bar chart view of FlexTree - nodes within the same level stack closely to each other in a space-filling manner.

Figure 5  Partial tree view of FlexTree - the structure of the tree is partially revealed.

Full Tree View

Figure 6  Full tree view of FlexTree - the structure of the tree is fully revealed.
Node Details

Space-Optimized Tree

- Put root node at center, then draw children out radially
- Key: Smart positioning to optimize placement of branches (Voronoi diagram-like approach)

Nguyen & Huang

*Information Visualization '03*
Space-optimized tree

- Connections + Enclosures
  - Goal: Show relationships and optimize space

- Layout
  - Vertex
  - Subtree
  - Wedge
  - Polygon

Viewing and Navigation

- Modified Semantic Zooming
  - Reduce density of tree
  - Selected Node to Root
  - History Path
Viewing and Navigation

- Focus + Context
  - Browsing (a)
  - Distortion (b)

Compare & Critique

- Which of the techniques do you find most appealing?
- Why?
Food for Thought

- Which of these techniques are useful for what purpose?
- How well do they scale?
- What if we want to portray more variables of each case?

Node-link Shortcoming?

- Difficult to encode more variables of data cases (nodes)
  - Shape
  - Color
  - Size
  - ...but all quickly clash with basic node-link structure
**Space-Filling Representation**

Each item occupies an area

Children are “contained” under parent

One example: “Icicle plot”

---

**Treemap**

- Space-filling representation developed by Shneiderman and Johnson, Vis ’91
- Children are drawn inside their parent
- Alternate horizontal and vertical slicing at each successive level
- Use area to encode other variable of data items
Example

Example
Example

Treemap

- Example
Treemap Example

Old one I built for our Sun workstations

Treemap Algorithm

Draw()
{
    Change orientation from parent (horiz/vert)
    Read all files and directories at this level
    Make rectangle for each, scaled to size
    Draw rectangles using appropriate size and color
    For each directory
        Make recursive call using its rectangle as focus
}
Nested vs. Non-nested

- Non-nested Tree-Map
- Nested Tree-Map

Learning Objectives

- Define hierarchical data & related terms
  - root, node, link, leaf, depth, parent, child, sibling
- List example tasks for hierarchical data
- Be able to draw reasonable 2D tree
  - Understand basic approach/algorithm and method
- Describe particular drawing techniques and explain +/- of each
  - SpaceTree, Cone Tree, Hyperbolic tree, H3 tree, DOI tree, FlexTree, Space-optimized tree
- Explain general limitations of node-link approach
- Understand treemap algorithm
  - Be able to draw slice-and-dice treemap given a hierarchy
Project Updates

- Progress
- Meetings
  - Sign up sheet: At least 24 hours in advance

Upcoming

- Hierarchies 2 – More on Space-filling reps
- Graphs & Networks 1