Data Visualization (DSC 530/CIS 602-01)

Trees & Geospatial Data

Dr. David Koop
Arrange Tables

Express Values

Separate, Order, Align Regions

Separate
Order
Align

1 Key
2 Keys
3 Keys
Many Keys

List
Recursive Subdivision
Volume
Matrix

Rectilinear
Parallel
Radial

Dense
Space-Filling

[Munzner (ill. Maguire), 2014]
SPLOMs and Parallel Coordinates

Scatterplot Matrix

Parallel Coordinates

Math
Physics
Dance
Drama
Math Physics Dance Drama
Math Physics Dance Drama
100
90
80
70
60
50
40
30
20
10
0

[Scatterplot Matrix]

[Parallel Coordinates]

[SPLOMs and Parallel Coordinates]

[Munzner (ill. Maguire), 2014]
Overdraw in Parallel Coordinates

[Figure: Overdraw in Parallel Coordinates with two sets of data showing density and correlation issues]

[Fua et al., 1999]
Arrange Networks and Trees

Arrange Networks and Trees

Node–Link Diagrams
Connection Marks
✔ NETWORKS ✔ TREES

Adjacency Matrix
Derived Table
✔ NETWORKS ✔ TREES

Enclosure
Containment Marks
✔ NETWORKS ✔ TREES

[Munzner (ill. Maguire), 2014]
Web Sites as Graphs (amazon.com)

[M. Salathe, 2006]
Arc Diagram

[D. Eppstein, 2013]
Force-Directed Layout

- Nodes push away from each other but edges are springs that pull them together
- Weakness: nondeterminism, algorithm may produce different results each time it runs

[M. Bostock, 2012]
sfdp

[JGD_Homology@cis-n4c6-b14, 7220 nodes, 13800 edges.]

[Hu, 2005]
Adjacency Matrices
Node-Link or Adjacency Matrix?

- Empirical study: For most tasks, node-link is better for small graphs and adjacency better for large graphs.
- Multi-link paths are hard with adjacency matrices.
- Immediate connectivity or neighbors are ok, estimating size (nodes & edges also ok).
- People tend to be more familiar with node-link diagrams.
- Link density is a problem with node-link but not with adjacency matrices.
Assignment 2

- [www.cis.umassd.edu/~dkoop/dsc530/assignment2.html](http://www.cis.umassd.edu/~dkoop/dsc530/assignment2.html)
- Use D3!
- 2016 Campaign Finance Data
  - Changes from mid- to end-2015
  - Including SuperPAC receipts
  - Extra Credit: Transitions
- Cheating
- Questions?
Arrange Networks and Trees

Node–Link Diagrams
Connection Marks

Adjacency Matrix
Derived Table

Enclosure
Containment Marks

[Munzner (ill. Maguire), 2014]
Trees

- Trees are directed acyclic graphs
  - each edge has a direction: the origin is the parent, the destination is the child
  - cannot get back to a node after leaving it
- A tree has a root (every other node hangs off it)
- Can consider enclosure in trees using parent-child relationships
# Tree Visualizations

![Tree Visualizations](image)

**Abstract**

Quantifying the space-efficiency of various 2D graphical representations of tree structures is presented. As part of the evaluation, a novel metric is defined as a total area of 1 (possibly partitioned by weight), then we analyze the area of nodes in a tree representation, and that can be applied to a broad range of different representations of trees. Some others exist because they have a total area of 1, but also because they allow for an equal number of nodes. However, experience suggests that the representations within each of these pairs do not scale equally well with larger, deeper trees. This article shows that there are finer ways of distinguishing efficiency, i.e. the mean area exponent as well as the area they allocate to non-terminal nodes (i.e. the leaf nodes) in the representation, in addition to the relative sizes of nodes. The key ideas involved are (1) the use of a metric of space-efficiency used, and “optimal” space-efficiency is defined as a total area of 1 (possibly partitioned by weight), then we analyze the area of nodes in a tree representation, and that can be applied to a broad range of different representations of trees.

**Introduction**

Some treemaps [14, 26, 6, 30], concentric circles [2, 27, 31], and many other tree representations are analyzed and compared by calculating the mean area exponent across nodes in a tree representation, and that can be applied to a broad range of different representations of trees. Some others exist because they have a total area of 1, but also because they allow for an equal number of nodes. However, experience suggests that the representations within each of these pairs do not scale equally well with larger, deeper trees. This article shows that there are finer ways of distinguishing efficiency, i.e. the mean area exponent as well as the area they allocate to non-terminal nodes (i.e. the leaf nodes) in the representation, in addition to the relative sizes of nodes.

### Figure 1

**A**

- **B**: Various tree representations, here each node-link [23, 7].
- **B**: Tree representations, and it is unclear what approach would be general.
- **C**: Treemaps [14, 26, 6, 30], concentric circles [2, 27, 31], and many others.
- **D**: Various tree representations, and it is unclear what approach would be general.
- **E**: Various tree representations, and it is unclear what approach would be general.
- **F**: Various tree representations, and it is unclear what approach would be general.
- **G**: Various tree representations, and it is unclear what approach would be general.
- **H**: Various tree representations, and it is unclear what approach would be general.

[McGuffin and Robert, 2010]
Node-Link Diagram

- Trees are graphs
- ...but we have more structure
- Horizontal or vertical
- Idea 1: partition space for each node via recursion
- Idea 2: “Tidy” Drawing
  - Wetherell & Shannon: Don’t waste space (overlapping parent nodes is ok)
  - Reingold and Tilford: Keep symmetry, subtrees look similar

[WS Alg., Reingold and Tilford, 1981]
Reingold-Tilford Algorithm

- Recurse on left and right subtrees
- Shift subtree over as long as it doesn’t overlap
- Place parent centered above the subtrees
- Originally, only binary trees, extended by Walker

[Reingold and Tilford, 1981]
Icicle Plot

• Line marks
• Vertical position shows depth
• Horizontal position shows links and sibling order
• Scalability: 1 pixel leaves, but harder to label

[Bostock, 2011]
Radial Node-Link

- Use polar coordinates instead of rectilinear
- Same layout algorithms work (e.g. Reingold-Tilford)
- Benefit: space usage, labels

[Bostock, 2012]
Sunburst

- Icicle plot in a radial layout
- Reading labels?
- Intuitive navigation

[Heer et al., 2012]
Nested Circles

• Looks more like cluster diagram, but shows hierarchy
• Containment shown by the layering of semi-transparent circles
• Labeling becomes more difficult

[Bostock, 2012]
Indented Outline

- Like a filesystem tree
- Use horizontal position to show depth, vertical positions show sibling/order
Treemap

[Bostock, 2012]
Treemap

- Containment marks instead of connection marks
- At each step, orientation of division (horiz/vert) changes
- Encodes some attribute of the items as the size of the rectangles
- Not as easy to see the intermediate rectangles
- Scalability: millions of leaf nodes and links possible
- Canonical example: Disk inventory
Disk Inventory

[Disk Inventory X: http://www.derlien.com]
Improving Treemaps (Cushion)

• Leaves are ok, but it can be difficult to find the hierarchy
• Encode this as shading information
• More effective to understand hierarchy

[van Wijk and van de Wetering, 1999]
Improving Treemaps (Squarified)

• Switching from horizontal to vertical cuts may be ok for nicely-behaved trees, but can lead to bad aspect ratios
• Problem: harder to compare sizes, more difficult to select/mouse over the rectangles
• Solution: Choose divisions (x/y) based on the width/height of region in order to maintain good aspect ratios
  - use left and right side
  - process large rectangles first
• Ordering not preserved which may cause issues if the data is updated
These steps are repeated until all rectangles have been processed. Again, an optimal result cannot be guaranteed, and counterexamples can be set up. The order in which the rectangles are processed is important. We found that a decreasing order usually gives the best results. The initially large rectangle is then filled in first with the larger subrectangles.

3.2 Algorithm

Following the example, we present our algorithm for the layout of the children in one rectangle as a recursive procedure squarify. This procedure lays out the rectangle in horizontal and vertical rows. When a rectangle is processed, a decision is made between two alternatives. Either the rectangle is added to the current row, or the current row is fixed and a new row is started in the remaining subrectangle. This decision depends only on whether adding a rectangle to the row will improve the layout of the current row or not.

We assume a datatype Rectangle that contains the layout during the computation and is global to the procedure squarify. Its support function width() that gives the length of the shortest side of the remaining subrectangle in which the current row is placed and a function layoutrow() that adds a new row of children to the rectangle. To keep the description simple, we use some list notation: ++ is concatenation of lists, is the list containing element, and is the empty list. The input of squarify() is basically a list of real numbers, representing the areas of the children to be laid out. The list row contains the rectangles laid out so far.

Squarification Algorithm

[Brus et al., 1999]
Squarified Treemaps

(a) File system  
(b) Organization

Fig. 5. Squarified treemaps

(a) File system  
(b) Organization

Fig. 6. Squarified cushion treemaps

[Brus et al., 1999]
Compound Networks

- Add a hierarchy to the network (e.g. from clustering)
- GrouseFlocks: uses nested circles with colors

![Input Graph](a)

<table>
<thead>
<tr>
<th>Graph Hierarchy 1</th>
<th>Graph Hierarchy 2</th>
<th>Graph Hierarchy 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="b" alt="Graph Hierarchy 1" /></td>
<td><img src="b" alt="Graph Hierarchy 2" /></td>
<td><img src="b" alt="Graph Hierarchy 3" /></td>
</tr>
</tbody>
</table>

[Archambault et al., 2008]
Geometry

• Shape information that is not determined by an attribute
• Data is often derived from real-world positions
  - Medical scans
  - Earth boundaries
• We use the geometry because we are familiar with the existing layout

[Geometry (Spatial)]

[Munzner (Ill. Maguire), 2014]
Geographic Data

• Spatial data

• Cartography: the science of drawing maps
  - Lots of history and well-established procedures
  - May also have non-spatial attributes associated with items
  - Thematic cartography: integrate these non-spatial attributes (e.g. population, life expectancy, etc.)

• Goals:
  - Respect cartographic principles
  - Understand data with geographic references with the visualization principles
# Search Tasks

<table>
<thead>
<tr>
<th></th>
<th>Target known</th>
<th>Target unknown</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td></td>
<td></td>
</tr>
<tr>
<td>known</td>
<td>Lookup</td>
<td>Browse</td>
</tr>
<tr>
<td>unknown</td>
<td>Locate</td>
<td>Explore</td>
</tr>
</tbody>
</table>

[Source: Munzner (ill. Maguire), 2014]
Lookup

![Map of Massachusetts Dartmouth](image)
Rendering Effective Route Maps: Improving Usability Through Generalization

Maneesh Agrawala, Chris Stolte
Stanford University

Figure 1: Three route maps for the same route rendered by (left) a standard computer-mapping system, (middle) a person, and (right) LineDrive, our route map rendering system.

The standard computer-generated map is difficult to use because its large, constant scale factor causes the short roads to vanish and because it is cluttered with extraneous details such as city names, parks, and roads that are far away from the route. Both the handdrawn map and the LineDrive map exaggerate the lengths of the short roads to ensure their visibility while maintaining a simple, clean design that emphasizes the most essential information for following the route. Note that the handdrawn map was created without seeing either the standard computer-generated map or the LineDrive map.

(Handdrawn map courtesy of Mia Trachinger.)

Abstract
Route maps, which depict a path from one location to another, have emerged as one of the most popular applications on the Web. Current computer-generated route maps, however, are often very difficult to use. In this paper we present a set of cartographic generalization techniques specifically designed to improve the usability of route maps. Our generalization techniques are based both on cognitive psychology research studying how route maps are used and on an analysis of the generalizations commonly found in handdrawn route maps. We describe algorithmic implementations of these generalization techniques within LineDrive, a real-time system for automatically designing and rendering route maps. Feedback from over 2200 users indicates that almost all believe LineDrive maps are preferable to using standard computer-generated route maps alone.

Keywords: Information Visualization, Non-Realistic Rendering, WWW Applications, Human Factors

1 Introduction
Route maps, which depict a path from one location to another, are one of the most common forms of graphic communication. Although creating a route map may seem to be a straightforward task, the underlying design of most route maps is quite complex. Mapmakers use a variety of cartographic generalization techniques including distortion, simplification, and abstraction to improve the clarity of the map and to emphasize the most important information [16, 21]. This type of generalization, performed either consciously or sub-consciously, is prevalent both in quickly sketched maps and in professionally designed route maps that appear in print advertisements, invitations, and subway schedules [25, 13].

Recently, route maps in the form of driving directions have become widely available through the Web. In contrast to handdesigned route maps, these computer-generated route maps are often more precise and contain more information. Yet these maps are more difficult to use. The main shortcoming of current systems for automatically generating route maps is that they do not distinguish between essential and extraneous information, and as a result, cannot apply the generalizations used in hand-designed maps to emphasize the information needed to follow the route.

Figure 1 shows several problems arising from the lack of differentiation between necessary and unnecessary information. The primary problem is that current computer-mapping systems maintain a constant scale factor for the entire map. For many routes, the lengths of roads can vary over several orders of magnitude, from tens of feet within a neighborhood to hundreds of miles along a highway. When a constant scale factor is used for these routes, it forces the shorter roads to shrink to a point and essentially vanish. This can be particularly problematic near the origin and destination of the route where many quick turns are often required to enter or exit a neighborhood. Even though precisely scaled roads might help navigators judge how far they must travel along a road, it is far more important that all roads and turning points are visible. Handdrawn maps make this distinction and exaggerate the lengths of shorter roads to ensure they are visible.

Another problem with computer-generated maps is that they are often cluttered with information irrelevant to navigation. This extraneous information, such as the names and locations of cities, parks, and roads far away from the route, often hides or masks information that is essential for following the route. The clutter makes the maps very difficult to read, especially while driving. Handdrawn maps usually include only the most essential information and are very simple and clean. This can be seen in figure 1 (middle) where even the shape of the roads has been distorted and simplified to improve the readability of the map. Furthermore, distorting
Locate

![Map with search for fish location](image-url)

- **David's Fish Market**
- **Flint Fish Market**
- **Kozy Nook Restaurant**
- **Bubba Fish & Seafood**
- **Whaling City Seafood Display Auction**
- **Fathoms Bar & Grill**
- **Northern Palegic Group Llc**
- **M F Foley Co Inc**
- **Horta's Restaurant**
- **Kozy Nook Restaurant**
- **Bittersweet Farm Restaurant**
- **Western Produce Market**
- **SE Mass Bioreserve**

Map data ©2015 Google, Terms, Privacy, Report a problem, 2 mi.
Adding Data

• Discrete: a value is associated with a specific position
  - Size
  - Color Hue
  - Charts

• Continuous: each spatial position has a value (fields)
  - Heatmap
  - Isolines
Discrete Categorical Attribute: Shape
Discrete Categorical Attribute: Shape

[Acadia NP, National Park Service]
Discrete Quantitative Attribute: Color Saturation
Discrete Quantitative Attribute: Size
Discrete Quantitative Attributes: Bar Chart

Railway Network Development and Bar Chart of Province Population in Turkey

[http://mis4gis.com/hgistr.org/]
Continuous Quantitative Attribute: Color Hue

[http://tampaseo.com/2012/02/websites-heat-mapping-users/]
Time as the attribute

Isolines

[USGS via Wikipedia]
Isolines

• Scalar fields:
  - value at each location
  - sampled on grids
• Isolines use *derived data* from the scalar field
  - Interpret field as representing continuous values
  - Derived data is *geometry*: new lines that represent the same attribute value
• Scalability: dozens of levels
• Other encodings?
Choropleths