VISUALIZATION

Visualizing Graphs/Networks

Overview

- Graph Definitions
- Graph Visualization
- Graph Layouts
- Interaction

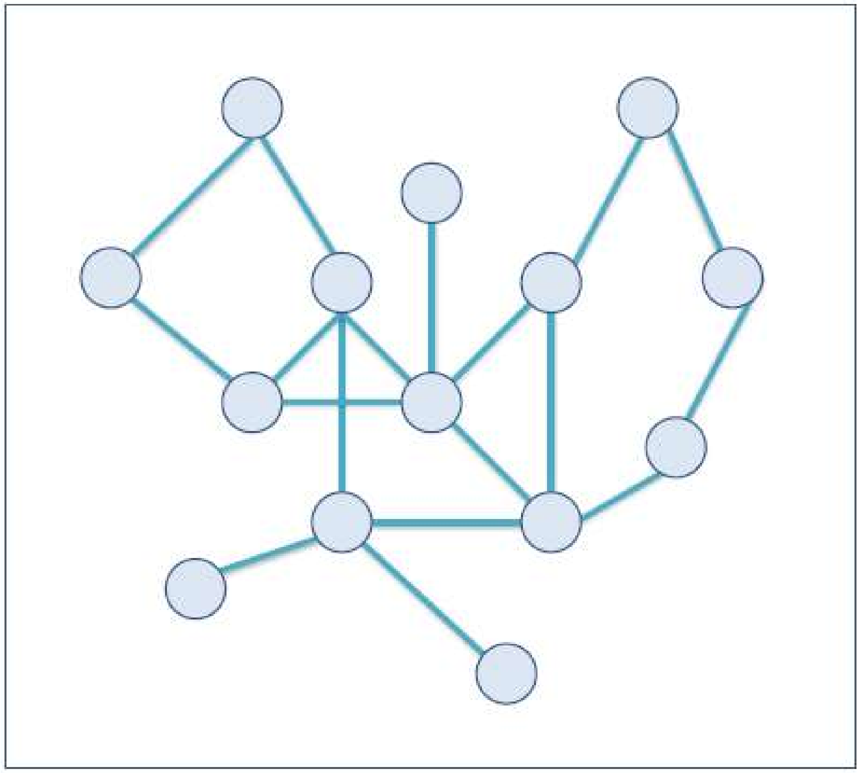
Overview

- Graph Definitions
- Graph Visualization
- Graph Layouts
- Interaction

What is a Graph?

A graph **G(V,E)** consists of a set of **vertices V** (also called nodes) and a

set of **edges E** connecting these vertices.

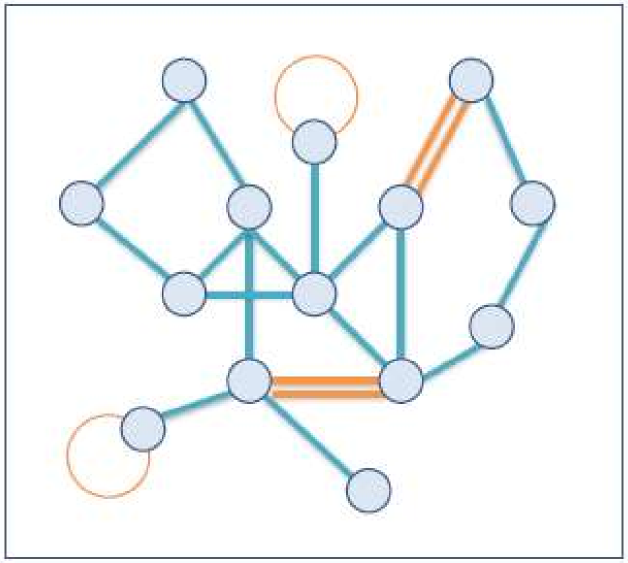


Examples

- Entity relationship diagrams
- Real-time systems (state transition diagrams)
- VLSI circuit design (circuit schematics rather than actual design)
- World-wide Web
- Social Network (Facebook, Twitter, etc)

Graph Terms

A simple graph G(V,E) is a graph which contains **no multi-edges** and **no loops**

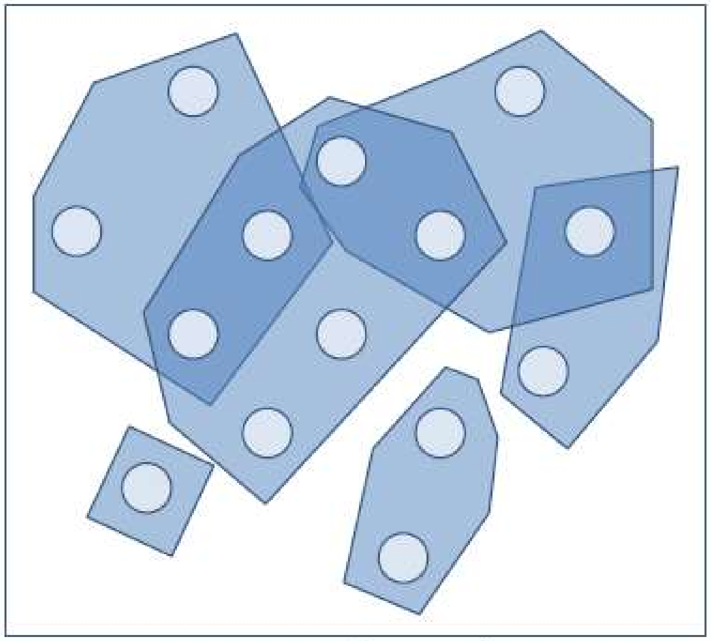


Not a simple graph!
→ A ***general graph***

Graph Terms

A directed graph (digraph) is a graph that discerns between the edges $A \rightarrow B$ and $A \leftarrow B$.

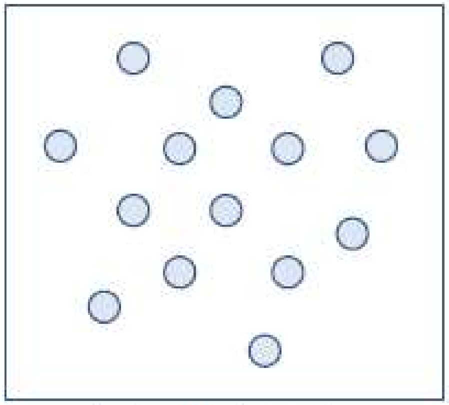
A hypergraph is a graph with edges connecting any number of vertices.

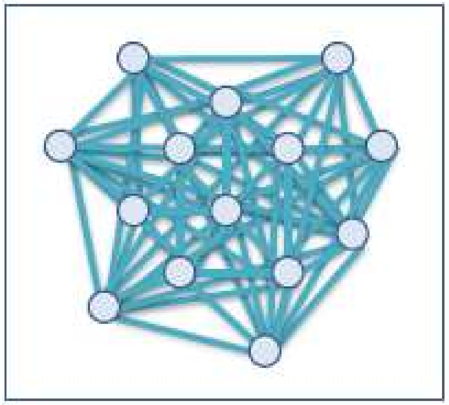

Hypergraph Example

Graph Terms

Independent Set
G contains no edges

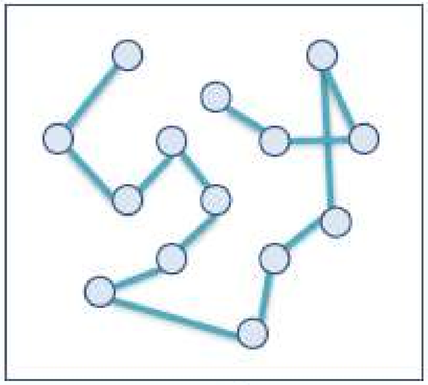
Clique
G contains all possible edges


Independent Set

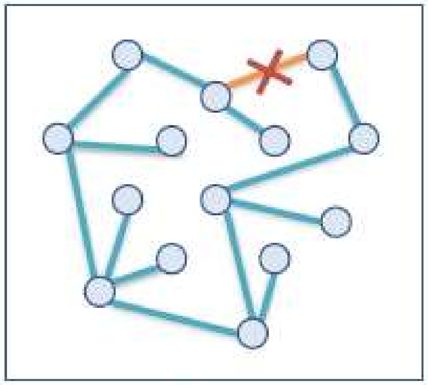

Clique

Graph Terms

Path
G contains only edges that can be consecutively traversed


Path

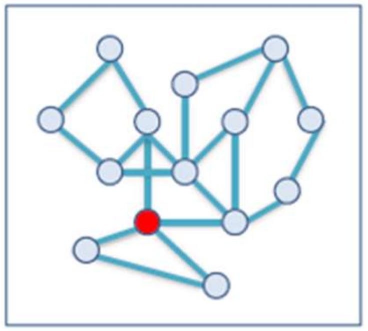
Tree
G contains no cycles


Tree

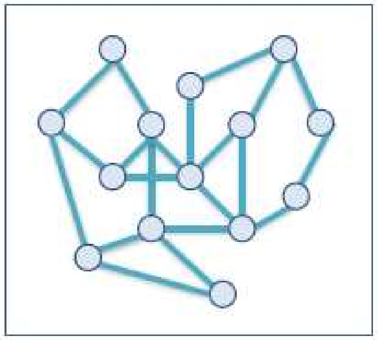
Network
G contains cycles

Graph Terms

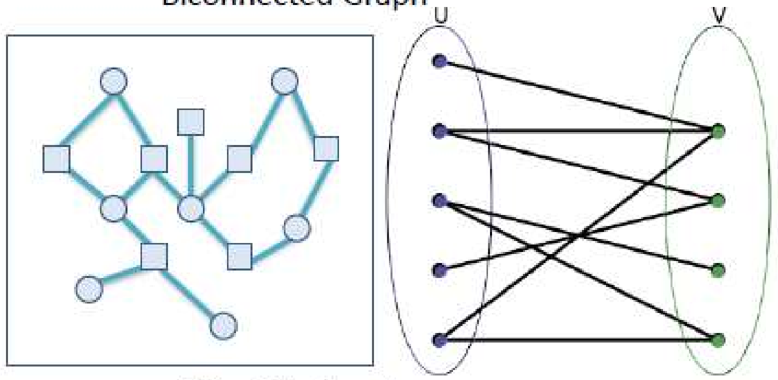
Articulation point
Vertices, which if deleted from the graph, would break up the graph in multiple sub-graphs.


Articulation Point (red)

Biconnected graph
A graph without articulation points.


Biconnected Graph

Bipartite graph
The vertices can be partitioned in two independent sets.



Bipartite Graph

Overview

- Graph Definitions
- Graph Visualization
- Graph Layouts
- Interaction

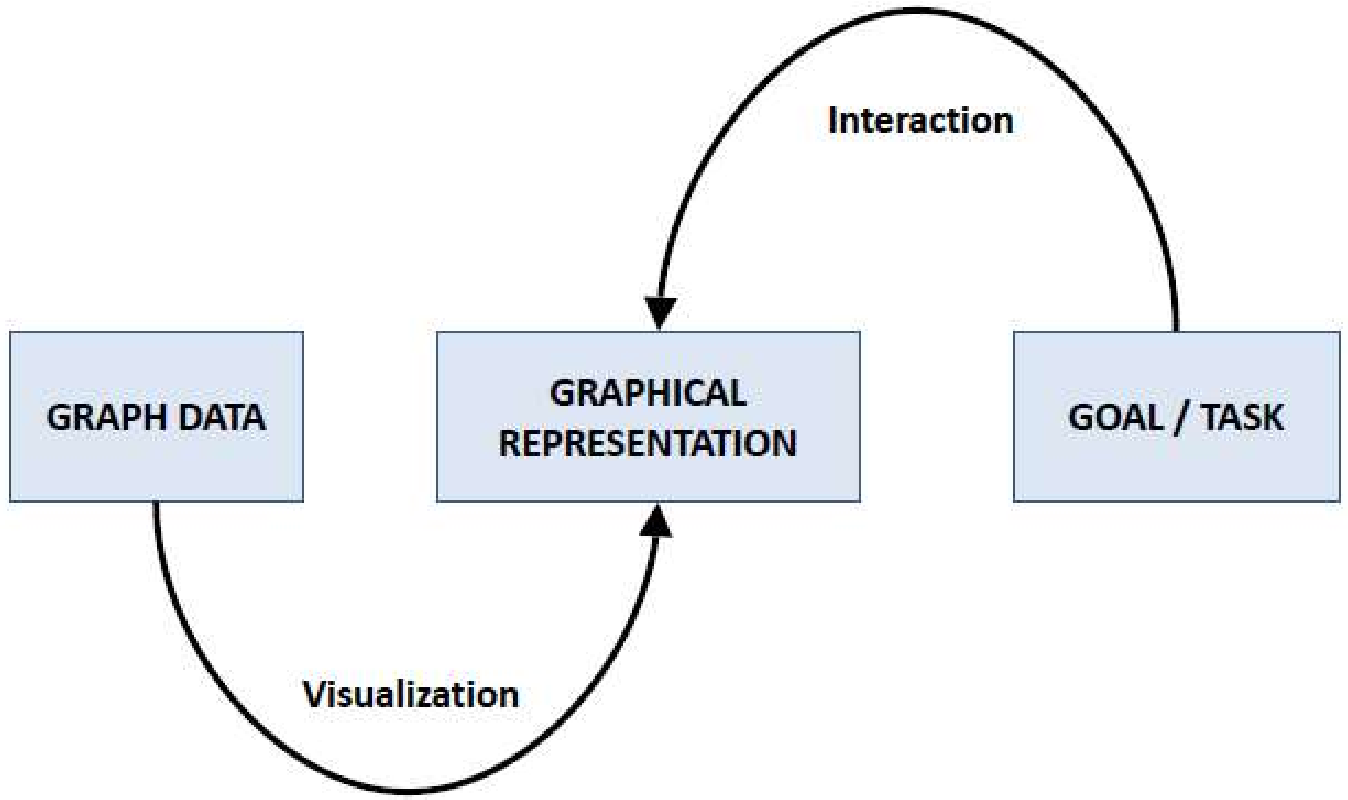
Why is graph visualization important?

- Visualization can be useful for Analysis and understanding of the graph
- Computing a layout of a graph is necessary to find insights
- It can allow users to see relationships, such as patterns and outliers, that would not be apparent through a metrics-based analysis alone

Graph Visualization vs. Graph Drawing

- Graph layout has evolved in two different directions:
  - Graph Drawing
    - Old topic, many books, etc.
    - Based on heavily algorithmic side drawing from mathematical graph theory
    - May have other goals than visualization (E.g. VLSI design)
  - Graph/Network Visualization
    - Size key issue
    - Can use node attributes for layout
    - Usability requires nodes to be discernable
    - Navigation considered

Graph Visualization



How to decide which **representation** to use for which **type of graph** in order to achieve which kind of **goal**?

Different Kinds of Tasks/Goals

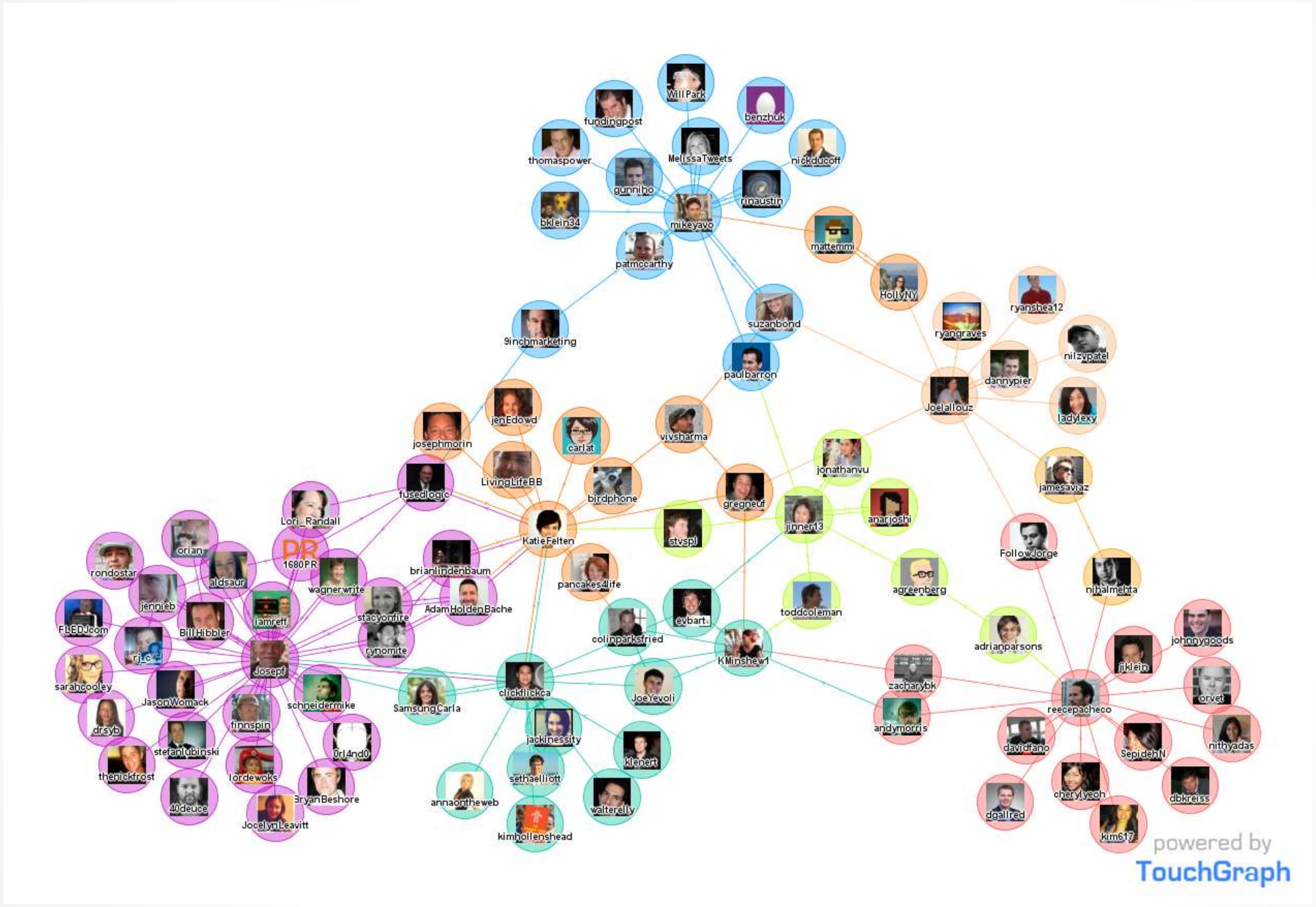
Two principal types of tasks: **attribute-based (ABT)** and **topology-based (TBT)**

- **Localize** – find a single or multiple nodes/edges that fulfill a given property
  - ABT: Find the edge(s) with the maximum edge weight.
  - TBT: Find all adjacent nodes of a given node.
- **Quantify** – count or estimate a numerical property of the graph
  - ABT: Give the number of all nodes.
  - TBT: Give the indegree (the number of incoming edges) of a node.
- **Sort/Order** – enumerate the nodes/edges according to a given criterion
  - ABT: Sort all edges according to their weight.
  - TBT: Traverse the graph starting from a given node.

Shneiderman’s Graph Visualization Nirvana

- Every node is visible
- For every node you can count its degree
- For every link you can follow it from source to destination
- Clusters and outliers are identifiable

Visualization of a Social Network

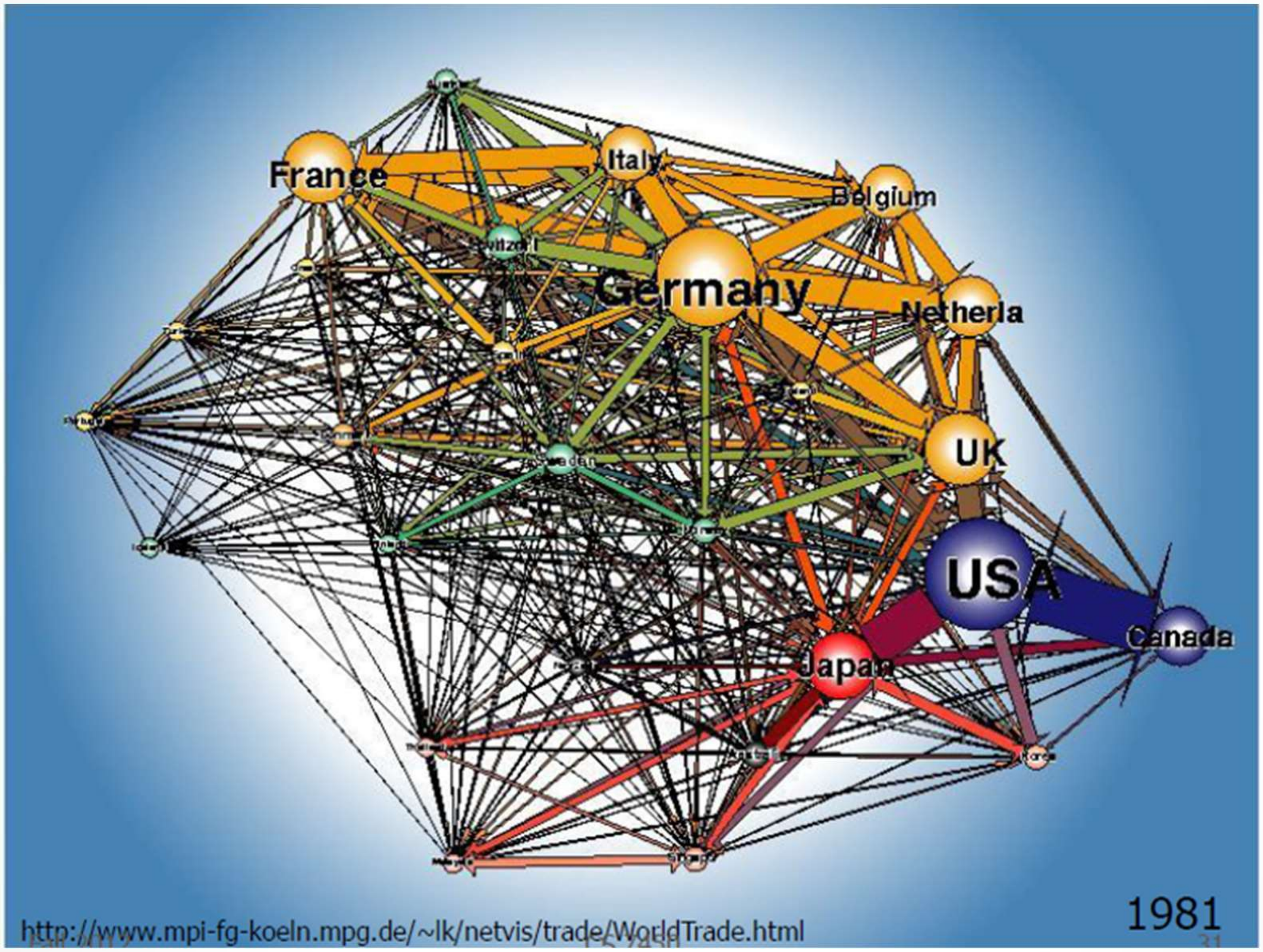


Visualization of Twitter activity data

Graph Visualization Challenges

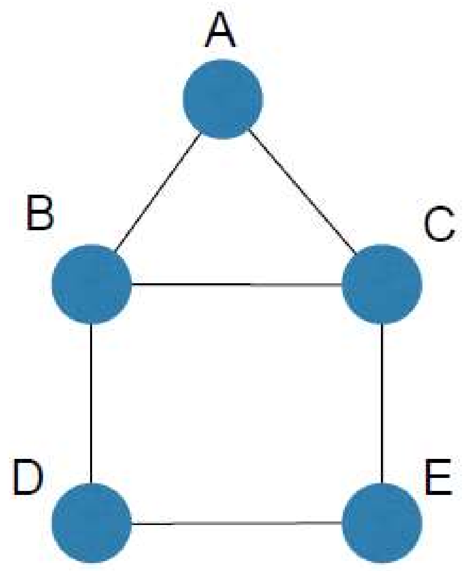
- Graph layout and positioning
  - Make a concrete rendering of abstract graph
- Navigation/Interaction
  - How to support user changing focus and moving around the graph
- Scale
  - Above two issues not too bad for small graphs, but large ones are much tougher

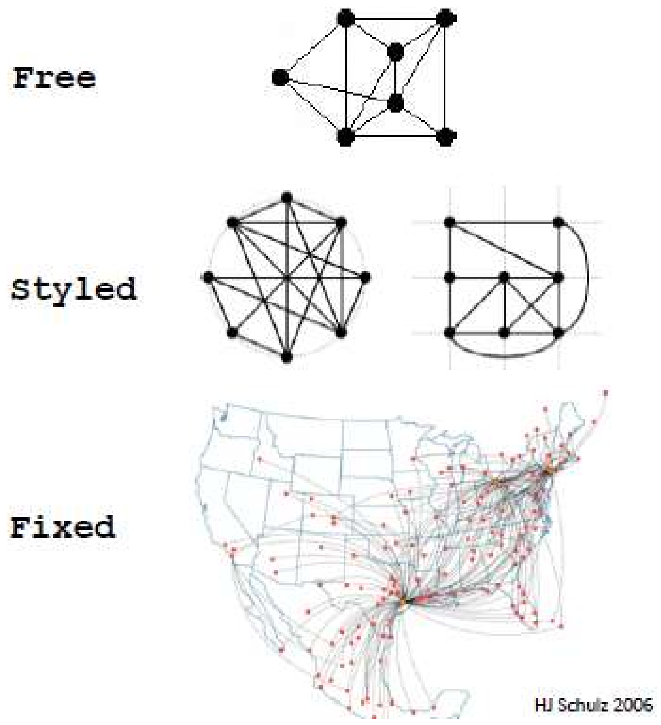
It is not easy to visualize large graphs!


France
Italy
Belgium
Germany
Netherla
UK
USA
Canada
Japan
1981
http://www.mpi-fg-koeln.mpg.de/~lk/netvis/trade/WorldTrade.html

Graph Representation

Node-link diagrams: vertex = point, edge = line/arc





Representation of Nodes and Links

- You can use various visual attributes to encode the information of nodes and links.
- Nodes
  - Shape
  - Color
  - Size
  - Label
- Links
  - Color
  - Thickness
  - Label

Criteria for Good Node-Link Layout

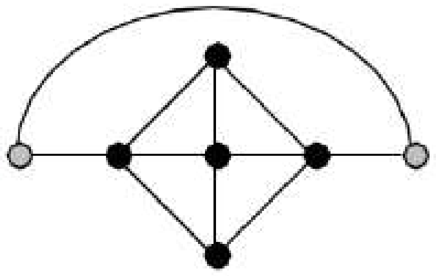
- Minimized **edge crossings**
- Minimized **distance** of neighboring nodes
- Minimized **drawing area**
- Uniform edge **length**
- Minimized edge **bends**
- Maximized **angular distance** between different edges
- Aspect ratio about 1 (not too long and not too wide)
- **Symmetry**: similar graph structures should look similar

Criteria and Justification

Criteria	Justification
Minimise Edge Crossings	Improves readability and aids the user in following paths. Edge crossings can also conceal important information and make a graph appear less approachable to the user
Symmetry	Symmetry aids in the understanding of the structure of a graph.
Uniform Edge Lengths	For a regular structure that prevents the graph becoming distorted.
Uniform Node Distribution	For a regular structure, visual appeal and to prevent the graph feeling cluttered
Separate Non-Adjacent Nodes	Proximity implies a relationship and so adjacent nodes should appear more related than non-adjacent nodes.
Node-Edge Overlap	Avoids visual elements appearing too clustered together and ambiguity as to where the edge ends

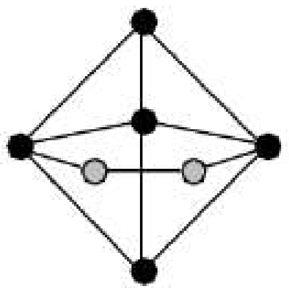
Conflicting Criteria

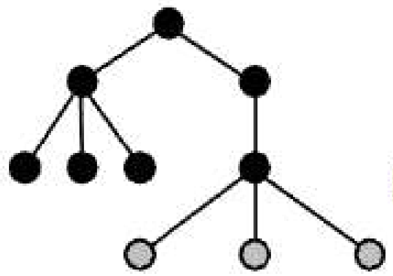
Minimum number of edge crossings



vs.

Uniform edge length

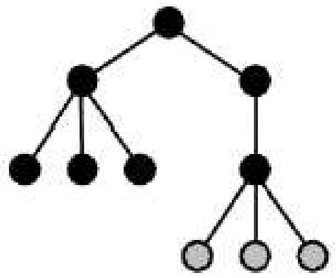




Space utilization

vs.

Symmetry



Other Criteria

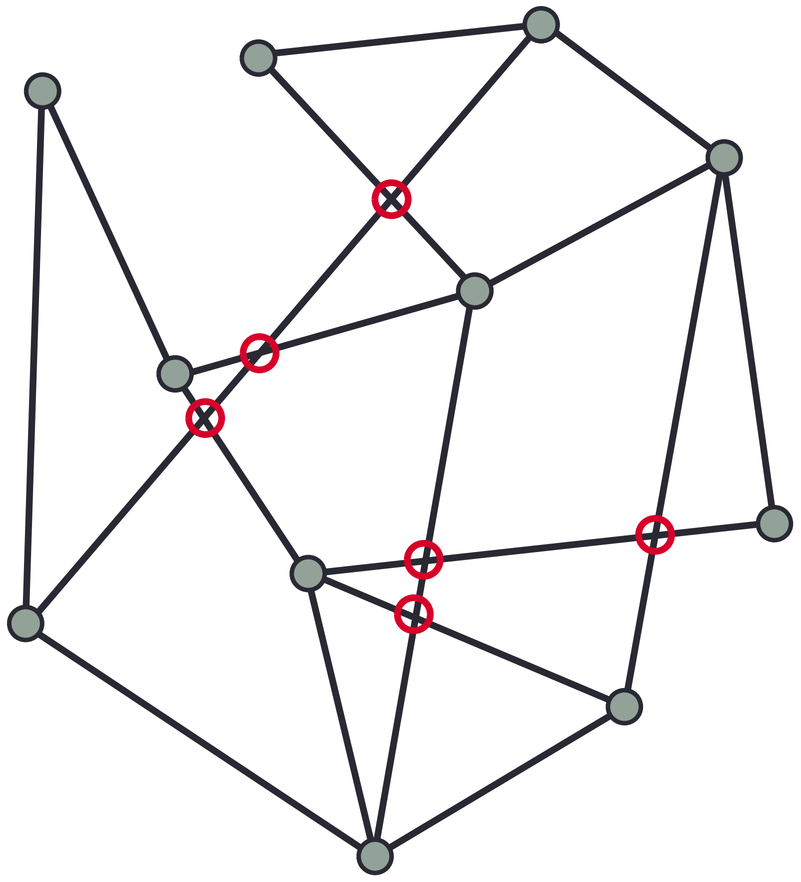
- Predictability
  - Two different runs on similar graphs should lead to similar layouts
- Time Complexity
  - Real time interaction

Exercise

- Try to layout the graph represented by the following adjacency matrix:

$$
\begin{array}{c|cccccccccc|}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\hline
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
2 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
4 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
5 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
6 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
8 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
9 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
\hline
\end{array}
$$

Edge Crossing

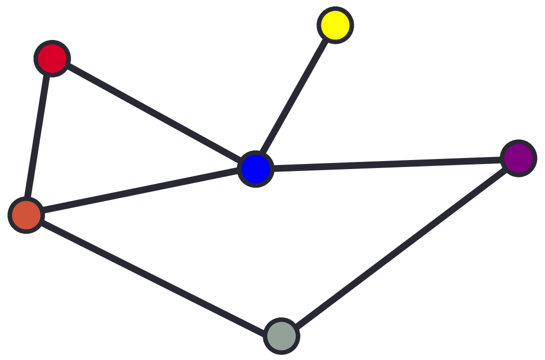


Minimizing edge crossing is NP-hard

Graph - Planarity

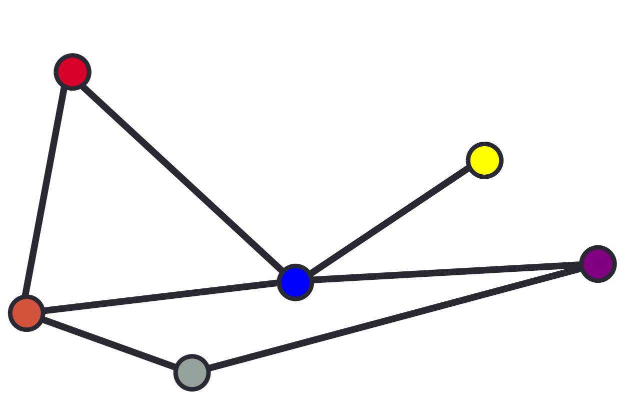
Planar Graph

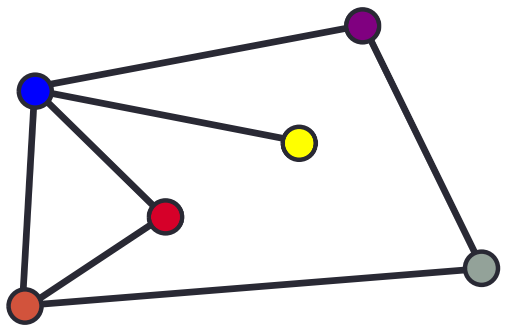
- Can be drawn in the plane without crossings

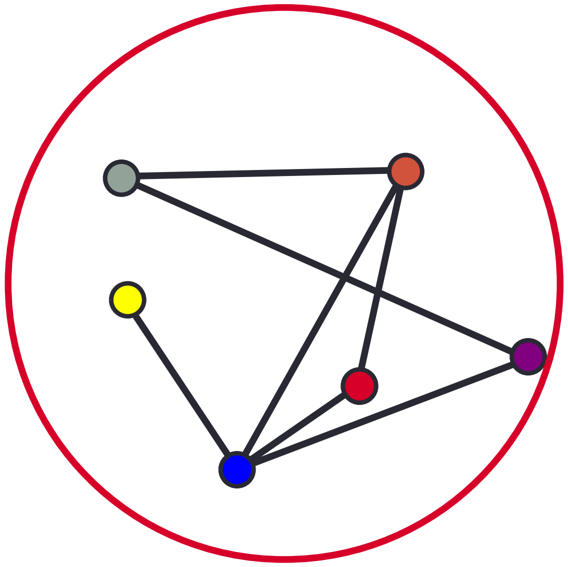


Plane Graph

- Planar graph with a fixed embedding

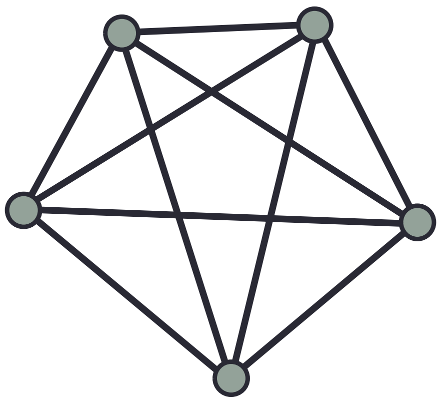




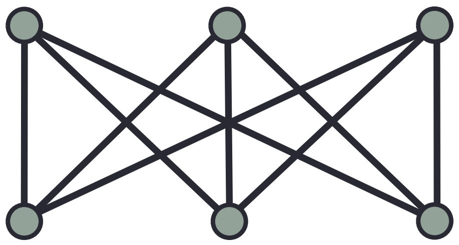


Smallest not-planar graphs

K_5



$K_{3,3}$



Planarity testing

Theorem [Kuratowski 1930 / Wagner 1937]

A graph G is planar if and only if it does not contain K_5 or $K_{3,3}$ as a minor.

Minor

A graph H is a minor of a graph G, if H can be obtained from G by a series of 0 or more deletions of edges, deletions of vertices, and contraction of edges.

- $G = (V, E)$, $|V| = n$, planarity testing in $O(n)$ time possible.

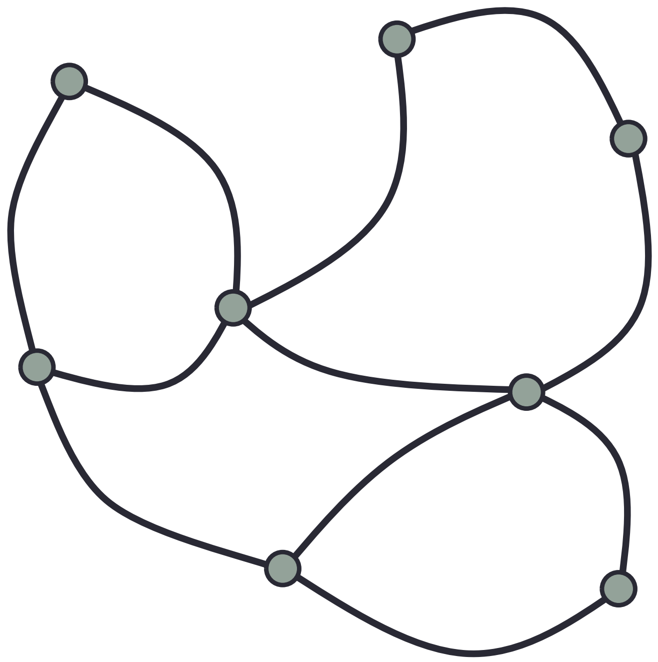
Overview

- Graph Definitions
- Graph Visualization
- Graph Layouts
- Interaction

Planar drawings

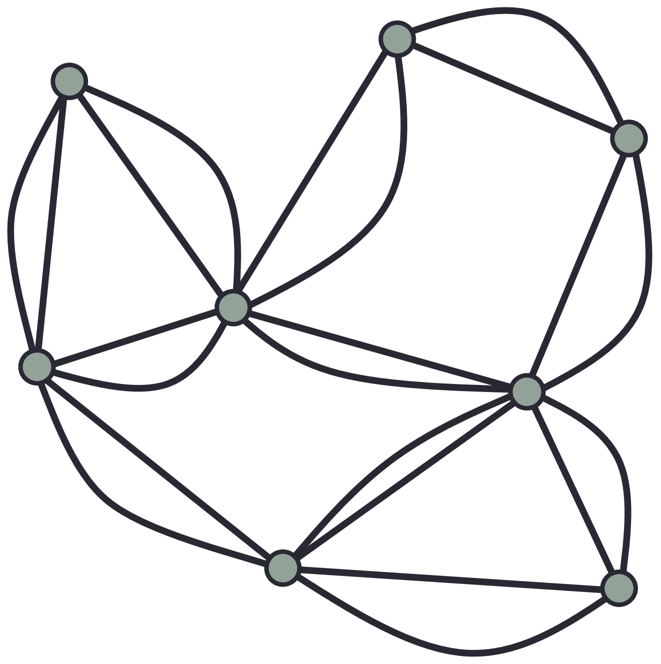
Vertices	points in the plane
Edges	curves

- No edge crossings



Straightline drawings

Vertices	points in the plane
Edges	straight lines



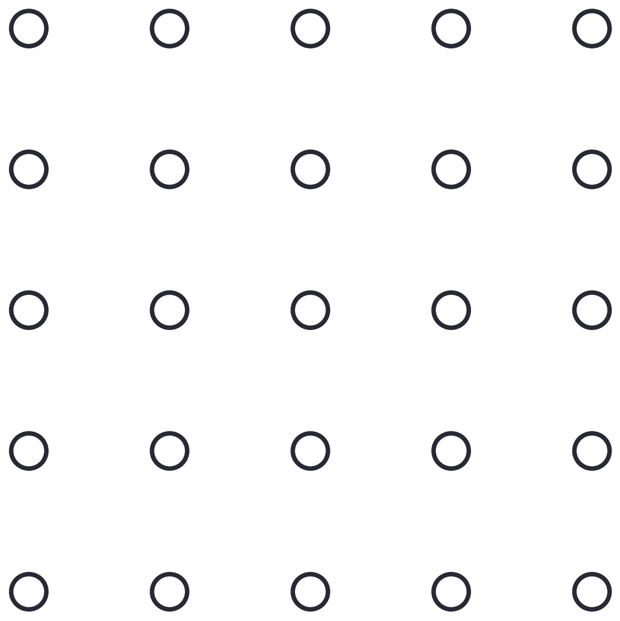
Theorem

Every planar graph has a plane embedding where each edge is a straight line.

[Wagner 1936, Fáry 1948, Stein 1951]

Grid Layout

Vertices	points in the plane on a grid
Edges	polylines, all vertices on the grid

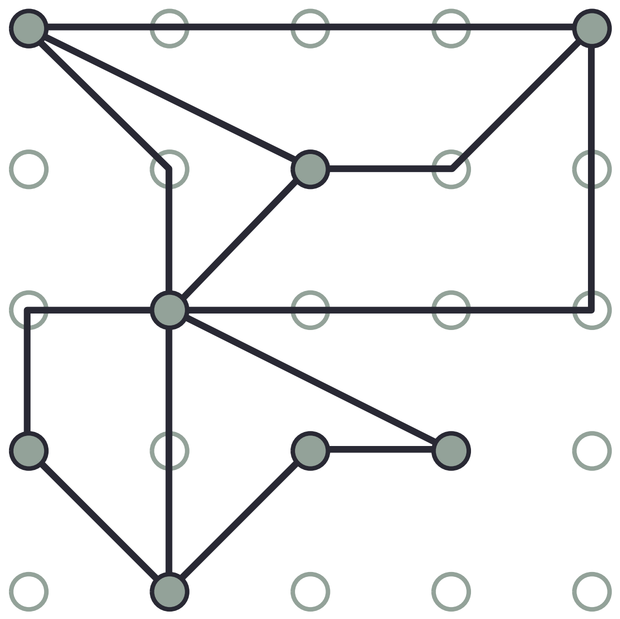


Grid Layout

Vertices	points in the plane on a grid
Edges	polylines, all vertices on the grid

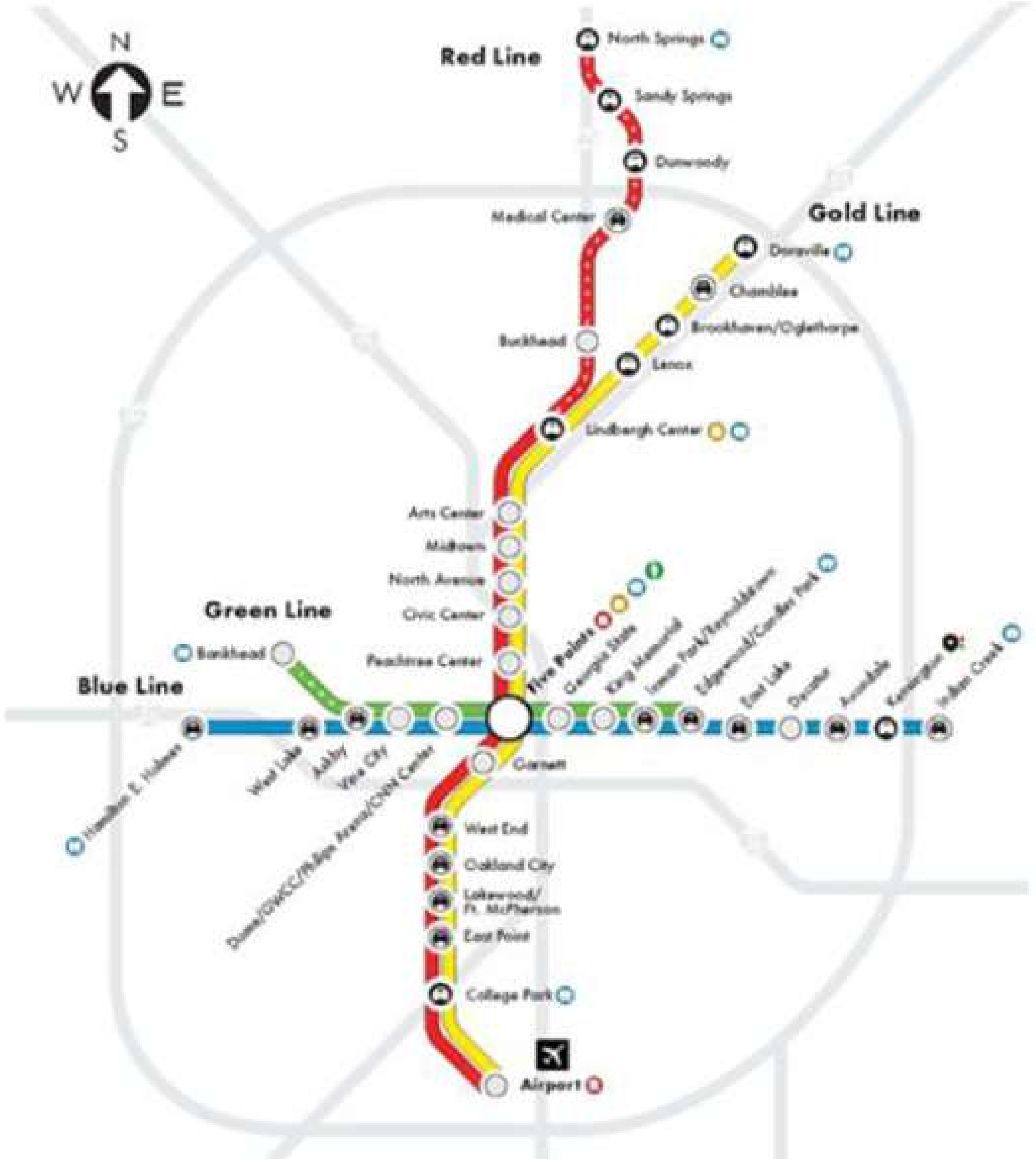
Objective

minimize grid size



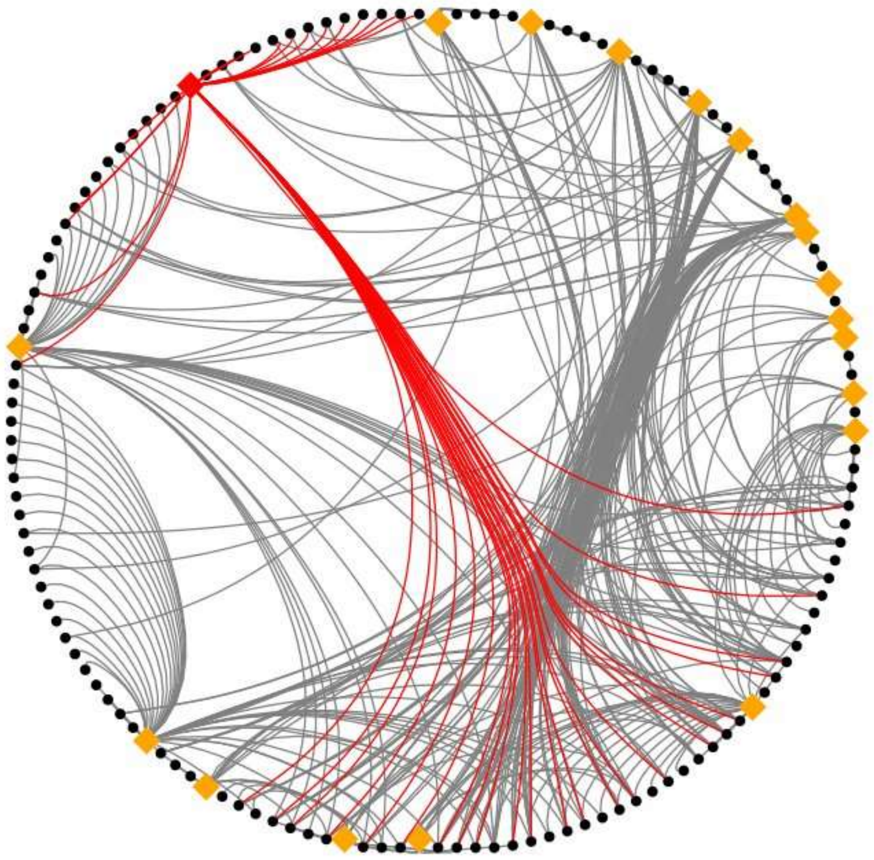
Geography-based Layout

Atlanta
MARTA



Circular Layout

- **Circular layout** is a style of drawing that places the vertices of a graph on a circle, often evenly spaced so that they form the vertices of a regular polygon.
- Draw edges to connect vertices



Hierarchical Layout

Often called Sugiyama layout

- Try to impose hierarchy on graph
  - Reverse edges if needed to remove cycles
  - Introduce dummy nodes
- Put nodes into layers or levels
- Order l->r to minimize crossings

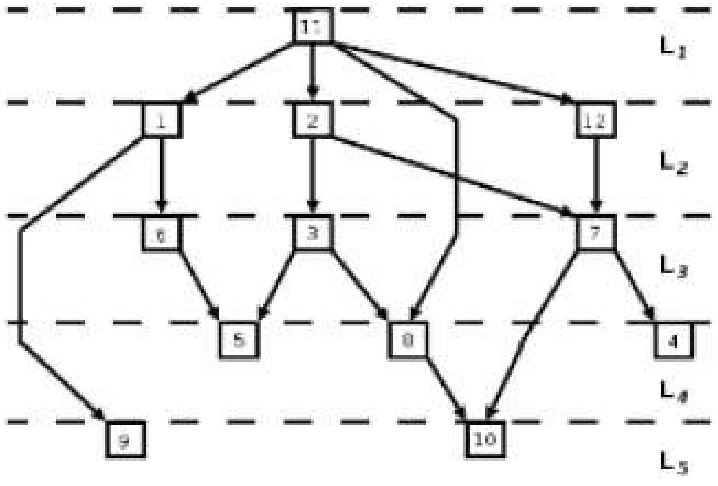


Figure: A graph showing a layered layout, created with the Sugiyama heuristic, with the layers shown. The bends in the edges correspond to dummy nodes.

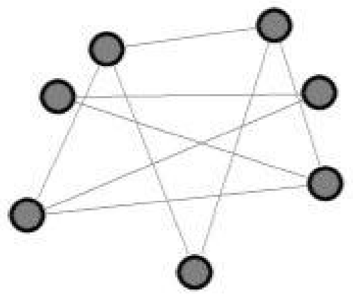
Force directed Layouts

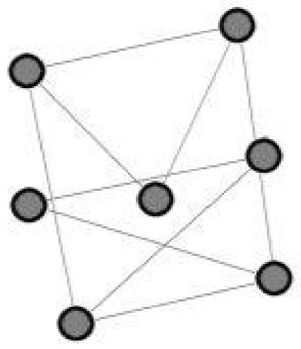
- Most common algorithmic graph drawing is based on the force-directed laypyt
- Modelling a graph as a physical system where nodes are attracted and repelled according to some force.

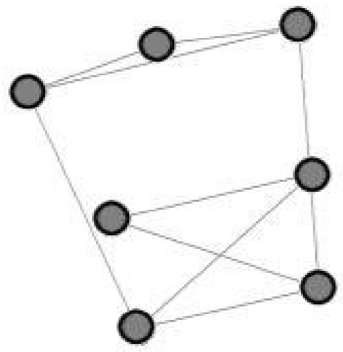
Spring-embedded Force-directed Layout

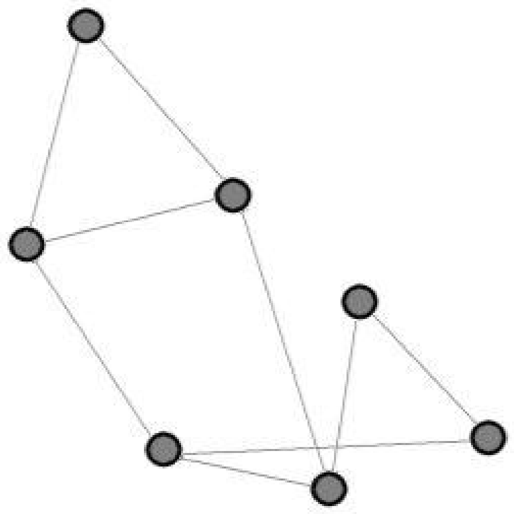
- Eades (A heuristic for graph drawing. *Congressus Numerantium* 1984) is the basis for almost all force-directed techniques.
- Nodes are modelled as steel rings and edges as springs; the system is put into some random initial conguration and released, leaving the system to reach a stable state where the force on each node is zero.
- Connected nodes are attracted to one another whilst all other nodes, modelled as electrical charges, repel
- Equations for forces
  - Attractive force = c1 * log(d/c2)
  - Repulsive force = c3/d^2
  - d is the length of the spring
  - c1, c2, c3 are constants
- From experimentation, Eades made the decision to use springs of logarithmic strength claiming that linear strength springs were too strong
- The resulting layout should have edges of uniform length and symmetry.

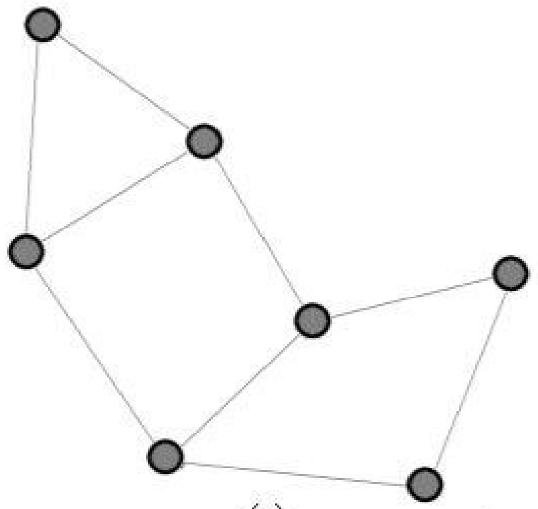
Example

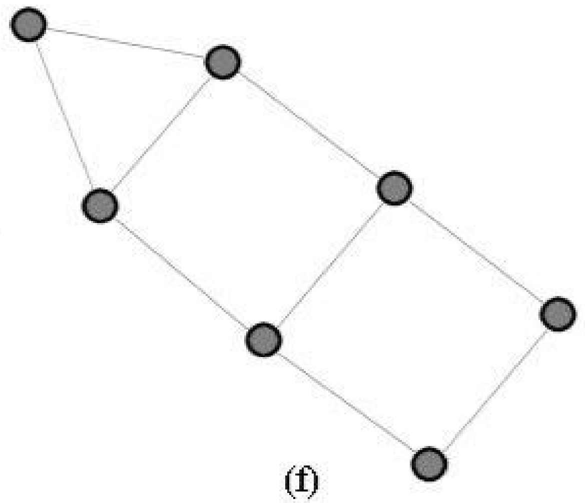

(a)


(b)

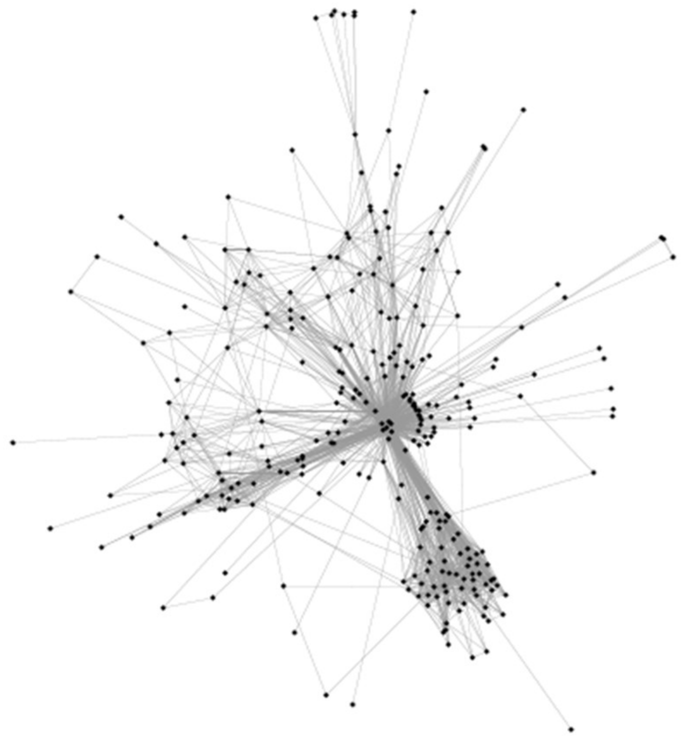

(c)


(d)


(e)


(f)

Example



The spring embedder for a protein-protein interaction network with 283 nodes and 1749 edges

Fruchterman Reingold Algorithm (1/2)

- Principle:
  - Vertices connected by an edge should be drawn near each other (but not too close)
  - Layout should have even node distribution, few edge crossings, uniform edge length, symmetry and fitting the drawingto the frame.
- Forces
  - Attractive force = d^2 /k
  - Repulsive force = $-k^2/d$
- k: Optimum distance between vertices.
  - where area is the space available and C an experimentally determined constant

$$k = C\sqrt{\left(\frac{\text{area}}{\text{number of vertices}}\right)}$$

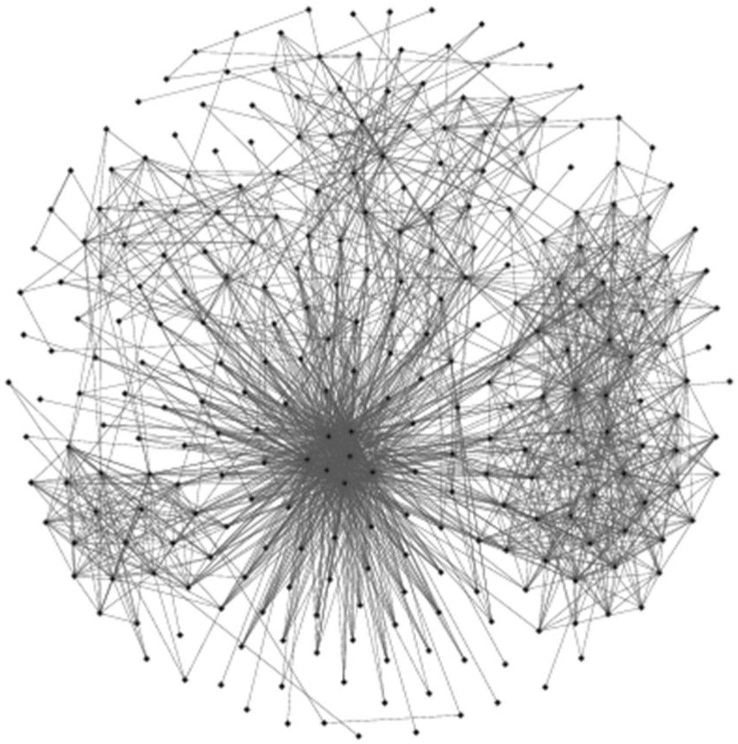
Fruchterman Reingold Algorithm (2/2)

- Adds global temperature
  - If hot, nodes move further each step
  - If cool, smaller movements
  - Generally cools over time
  - The idea is that the displacement of a vertex is limited to some maximum value
  - This maximum value decreases over time
  - So, as the layout becomes better, the amount of adjustment becomes finer and finer.
- The algorithm is not guaranteed to converge
  - Iterations of the algorithm is sufficient for an optimum layout

Fruchterman Reingold Algorithm - Pseudocode

```
area:= $W * L$; {$W$ and $L$ are the width and length of the frame}
$G := (V, E)$; {the vertices are assigned random initial positions}
$k := \sqrt{area/|V|}$;
**function** $f_a(x)$ := **begin return** $x^2/k$ **end**;
**function** $f_r(x)$ := **begin return** $k^2/x$ **end**;
**for** $i := 1$ **to** *iterations* **do begin**
    {calculate repulsive forces}
    **for** $v$ in $V$ **do begin**
        {each vertex has two vectors: .*pos* and .*disp*
        $v.disp := 0$;
        **for** $u$ in $V$ **do**
            **if** $(u \neq v)$ **then begin**
            {$\delta$ is the difference vector between the positions of the two vertices}
                $\delta := v.pos - u.pos$;
                $v.disp := v.disp + (\delta/|\delta|) * f_r(|\delta|)$
            **end**
    **end**
    {calculate attractive forces}
    **for** $e$ in $E$ **do begin**
        {each edges is an ordered pair of vertices .$vand.u$}
        $\delta := e.v.pos - e.u.pos$;
        $e.v.disp := e.v.disp - (\delta/|\delta|) * f_a(|\delta|)$;
        $e.u.disp := e.u.disp + (\delta/|\delta|) * f_a(|\delta|)$
    **end**
    {limit max displacement to temperature $t$ and prevent from displacement
outside frame}
    **for** $v$ in $V$ **do begin**
        $v.pos := v.pos + (v.disp/|v.disp|) * \min(v.disp, t)$;
        $v.pos.x := \min(W/2, \max(-W/2, v.pos.x))$;
        $v.pos.y := \min(L/2, \max(-L/2, v.pos.y))$
    **end**
    {reduce the temperature as the layout approaches a better configuration}
    $t := cool(t)$
**end**
```

Example

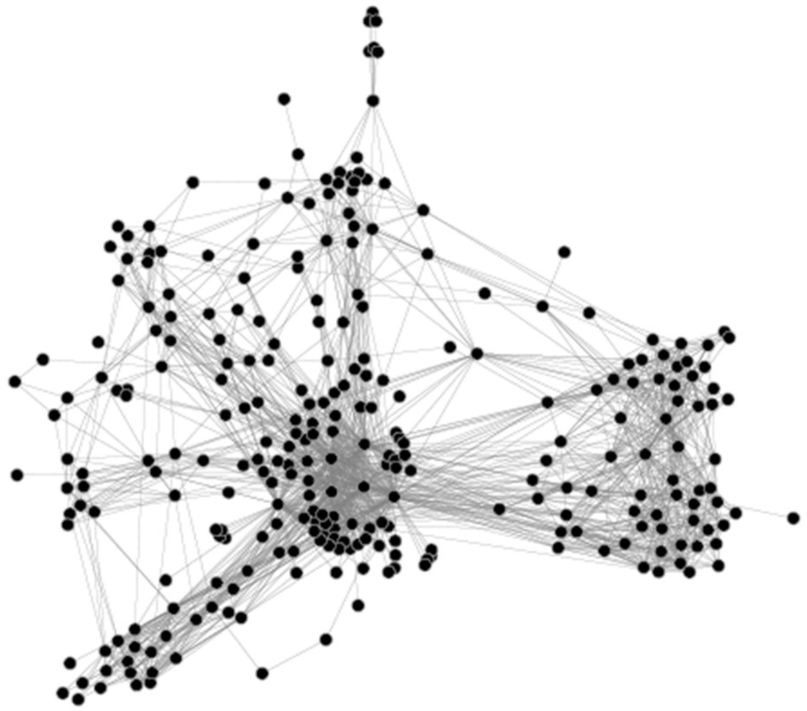


The protein-protein interaction network laid out using Fruchterman and Reingold's algorithm

Energy-based Approaches

- Energy-based approaches consider layout to be the minima of an optimisation problem in which an energy function encodes the desired properties of the graph.
- Example: Force Atlas 2014 https://doi.org/10.1371/journal.pone.0098679
- Main aim is to highlight clustering in a graph
- Uses linear attractive and repulsive forces.
  - Attractive forces are again distances between nodes
  - Repulsive forces are based on distances and node degree plus one.
  - This ensures all nodes have at least some repulsive force and so poorly connected nodes are brought closer to well connected ones reducing visual clutter.
  - Nodes with high degree are less likely to be clumped in the center of the graph
  - Layout is much more clearly able to show any underlying clustered structure in the graph
- The algorithm maximises speed until it is clear that some nodes are unstable at which point it slows and emphasises precision.
- The layout is interactive for the user
  - There are many options that can be configured even while the layout is running to produce a layout that is most suitable for the user

Example



The protein-protein interaction network laid out using Force Atlas algorithm

Force-directed Layouts – Evaluation

- Pros
  - Strong theoretical foundations
  - Simplicity: Few lines of code
  - Good results (till graphs of about 1000 nodes)
  - Works well with structures such as grids, trees and sparse graphs rather than those with a dense structure.
- Cons
  - Computationally expensive
    - Often produces poor layouts for large graphs
    - Time complexity $O(|V|^2 + |E|)$
    - For interactive visualization limit is about 1000 nodes
  - Not predictable

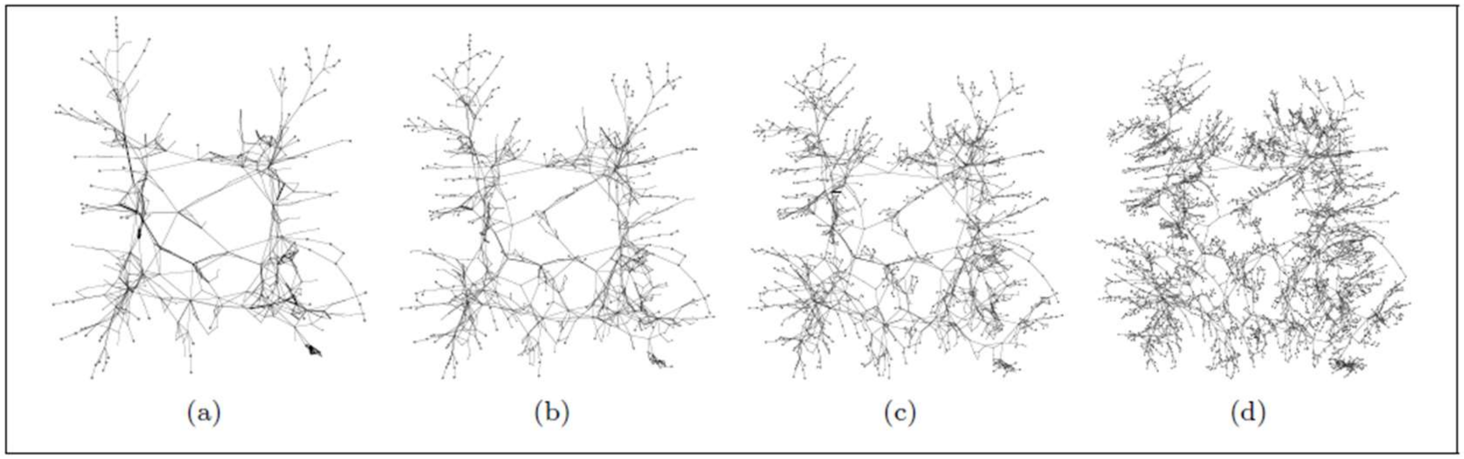
Computational Improvements - Multi-level Techniques

- Find a sequence of coarser representations of a graph
  - The coarser representations are created by collapsing connected nodes whose edges becomes the union of the edges of all the nodes it comprises
- Optimise the drawing in the coarsest representation, and propagate that layout back through to the original graph
- Example: Hu Y [2005]

Yifan Hu Algorithm

- Follow a simple edge collapsing coarsening scheme
  - If more than 50% of the nodes remain after the edge collapsing then the algorithm uses *maximum independent vertex sets (MIVS)* for their coarsening procedures
  - An IVS is formed if there are no two nodes connected by an edge in the subset
  - It is said to be maximal if the addition of any edge to the subset would break this property
- Uilises the Barnes-Hut approximation
  - A square is placed over the initial layout of the graph and is divided into four.
  - If thereis more than one node inside a sub-square then it is divided into four.
  - This procedure continues recursively until each square contains at most one node.
  - Nodes are then clustered based on their position in the quad-tree
  - Those clusters considered to be far from the node of interest form a super-node whose forces can be considered as one reducing the complexity of the force calculation.
- Adds an adaptive cooling scheme to a general force-directed model
  - Step length remains constant until there are consecutive energy reductions, in which case the step size is increased, or
  - if there is an energy increase then step length is decreased

Example



Each image from left to right shows the layout propagated back up through each less coarse version of the graph,
i.e., (a) is the most coarse version of the graph while (d) is the final layout.

Dimension Reduction for Layout

- Dimension reduction is the process of taking data expressed in high-dimensional space and projecting it onto a lower-dimensional space.
- The challenge is to try to retain the information that is in the high-dimensional space and capture it in the lower-dimensional representation.
- Most dimension reduction techniques currently used for graph layout use the graph-theoretic distance between a pair of nodes as the information that is to be preserved.

High-dimensional Embedding (HDE)

- The aim of this layout was to follow the convention of placing adjacent nodes close together and non-adjacent nodes further apart.
- The graph is first embedded in a high-dimensional space by choosing 50 nodes as pivots and associating each pivot with a dimension
- Nodes are then expressed in high-dimensional space as graph-theoretic distances from pivots.
- Then the graph is linearly projected onto two dimensions using PCA

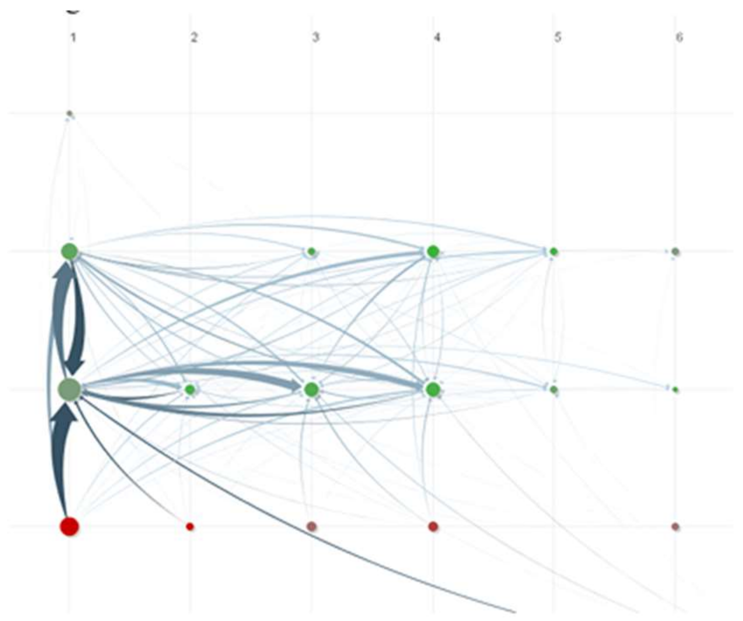
- Harel D and Koren Y, Graph drawing by high-dimensional embedding. Journal of Graph Algorithms and Applications 2004; 8: 207-219.

Using Node Attributes for Layout

- Attributes of nodes influence geometric positioning
  - Example: Scatter plot based on node values
- Not just some arbitrary layout - Utilize graph statistical analysis
- Dynamic queries and/or brushing can be used to explore connectivity
- There are three main ways attributes can be used in graph layout:
  - use them to impose a set of restrictions on the placement of nodes
  - use membership of a group or cluster to position the nodes
  - directly map an attribute (or attributes) to a coordinate in the layout space (e.g., x and y in a Cartesian system).

Pivot Graph

- Cluster on common node attributes
  - Put all A's together, all B's together, …
- Position nodes into a grid based on attributes
- Draw edge from A to B depending on how many edges from some A to some B
- Problems:
  - Only 2 variables
  - Doesn't support continuous variables

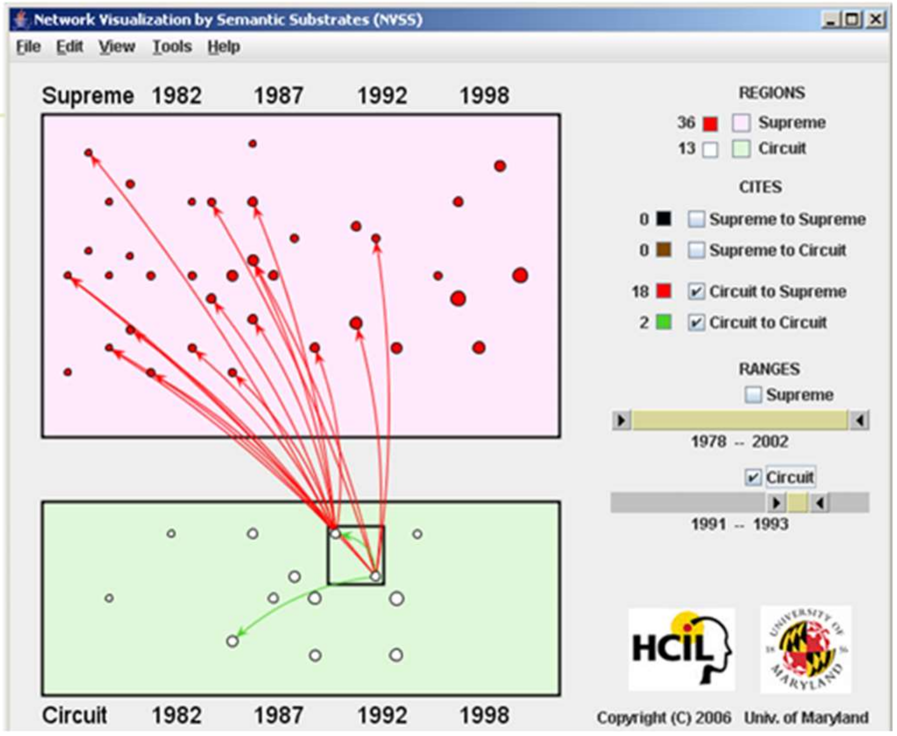


Communication network of people in a large company. X-axis is division, y-axis is office geography. The division in the leftmost column has far more cross-location communication than the others.

Wattenberg CHI '06

Network Visualisation by Semantic Substrates (NVSS)

- Group nodes into regions according to an attribute
  - Categorical, ordinal, or binned numerical
- In each region: Position nodes according to some other attribute(s)
- Give users control of link visibility

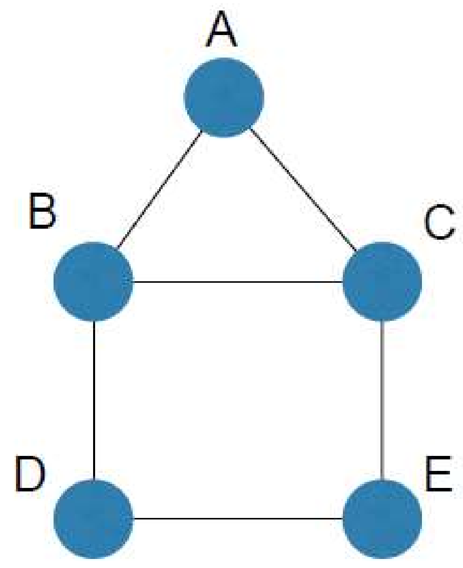

Network Visualization by Semantic Substrates (NVSS)
File Edit View Tools Help
Supreme 1982 1987 1992 1998
Circuit 1982 1987 1992 1998
REGIONS
36 Supreme
13 Circuit
CITES
0 Supreme to Supreme
0 Supreme to Circuit
18 Circuit to Supreme
2 Circuit to Circuit
RANGES
Supreme
1978 -- 2002
Circuit
1991 -- 1993
HCIL
Copyright (C) 2006 Univ. of Maryland

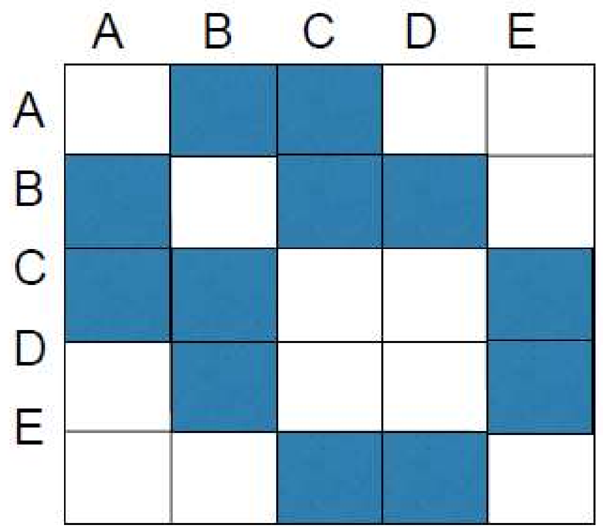
- Shneiderman & Aris TVCG (InfoVis) 2006

Cases are first grouped into Supreme Court and Circuit Court. Positioned based on year. Selecting 2 Circuit Court cases in 1991-1993 generates a comprehensible display of the 18 red Supreme Court and the 2 green Circuit Court citations.

Matrix Representation

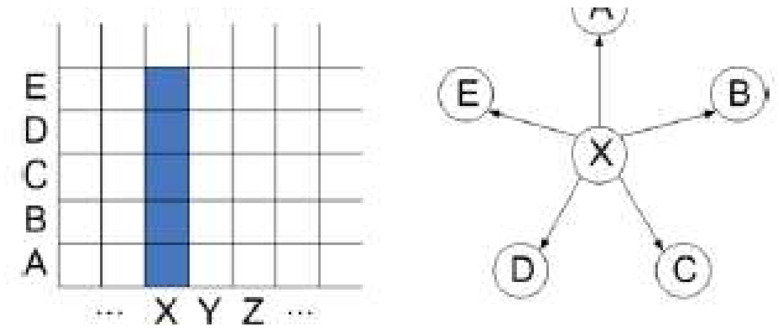
Instead of node link diagram, use adjacency matrix

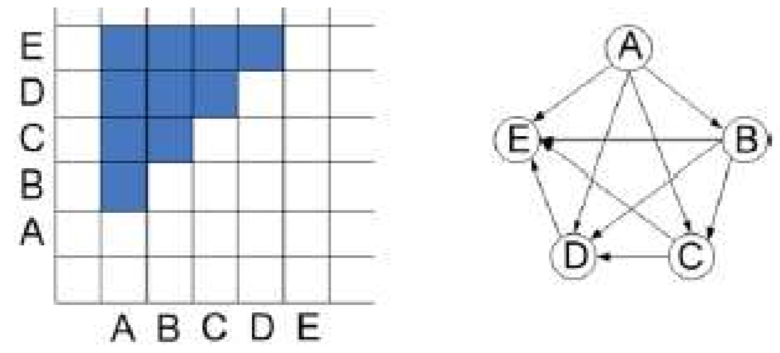


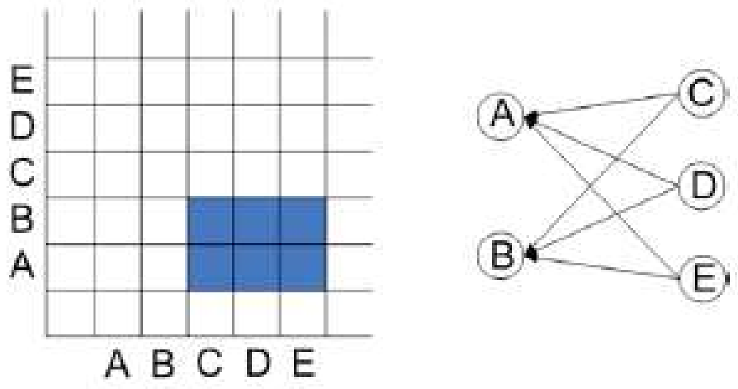


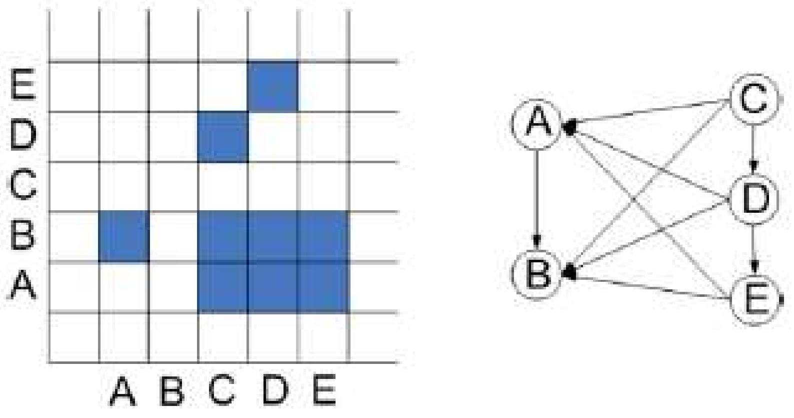
	A	B	C	D	E
A		🟦	🟦		
B	🟦		🟦	🟦	
C	🟦	🟦			🟦
D		🟦			🟦
E			🟦	🟦	

Examples

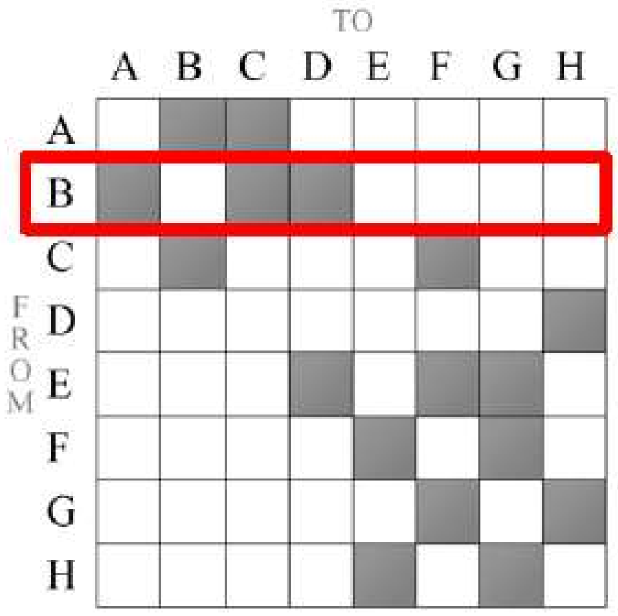

E
D
C
B
A
··· X Y Z ···
A
E
B
X
D
C


E
D
C
B
A
A B C D E
A
E
B
D
C

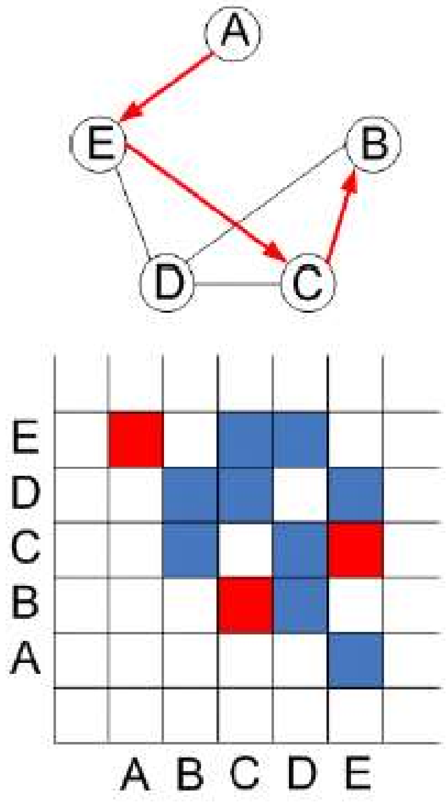

E
D
C
B
A
A B C D E
A
C
D
B
E


E
D
C
B
A
A B C D E
A
C
D
B
E

Pros and Cons



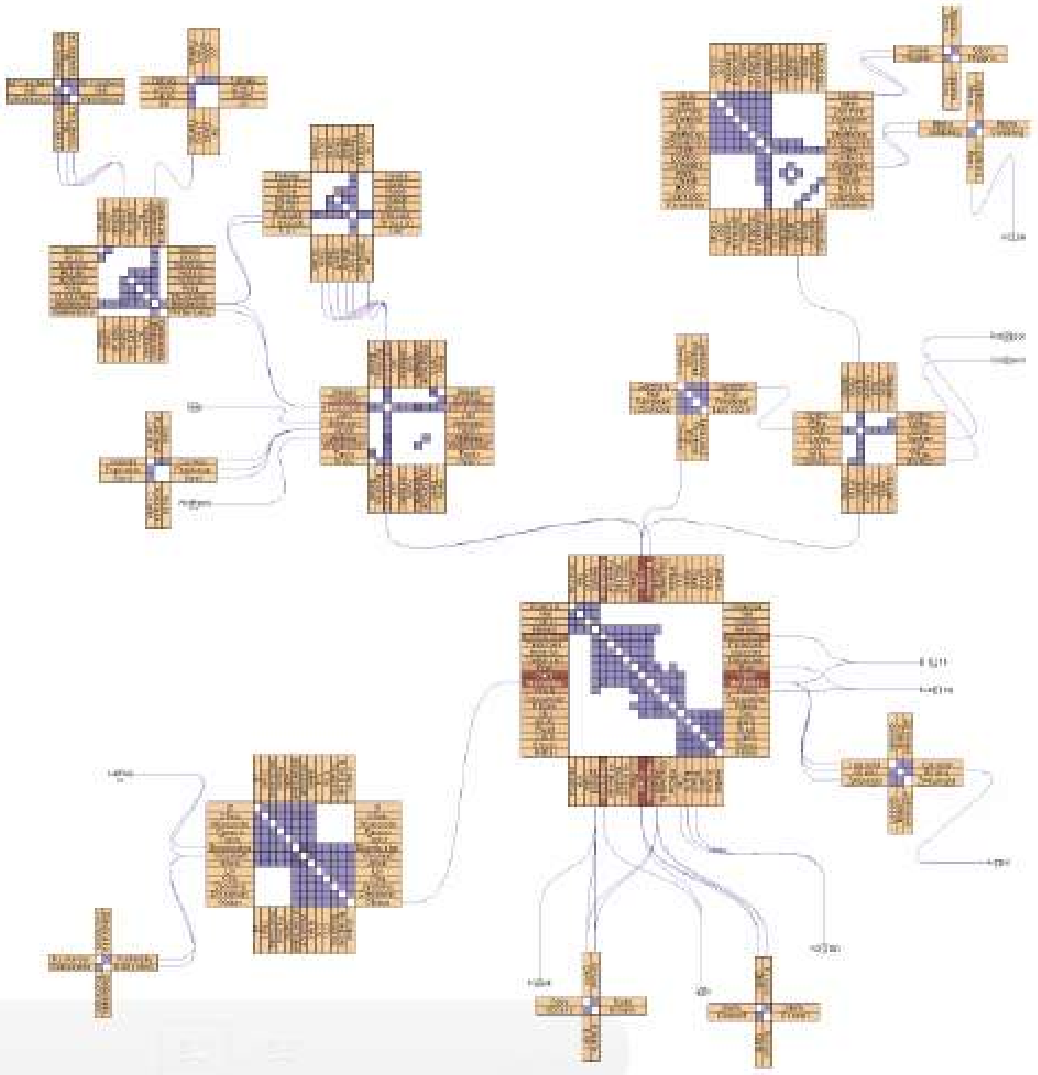
Well suited for neighborhood-related TBTs



Not suited for path-related TBTs

NodeTrix

Hybrid of matrix
and node-link



Henry & Fekete
TVCG (InfoVis) '07

Reading

1. H. Gibson, J. Faith, and P. Vickers, “A survey of two-dimensional graph layout techniques for information visualisation,” Information Visualization, vol. 12, no. 3-4, pp. 324–357, 2013. https://journals.sagepub.com/doi/pdf/10.1177/1473871612455749

2. Fruchterman and Reingold. **Graph Drawing by Force-directed Placement.** *Software - Practice and Experience, 1991* Available at: *http://reingold.co/force-directed.pdf*

3. Hu Y, Efficient, high-quality force-directed graph drawing. Mathematica Journal 2005; 10:37-71 *https://www.researchgate.net/publication/235633159_Efficient_and_High_Quality_Force-Directed_Graph_Drawing*

4. M. Wattenberg, **Visual exploration of multivariate graphs**, *Proceedings of ACM CHI 2006*, Available at: http://hint.fm/papers/pivotgraph.pdf

5. Henry and Fekete: **NodeTrix: A Hybrid Visualization of Social Networks.** *IEEE Transactions on Visualization and Computer Graphics*, 2007. Available at: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/12/Henry_infovis07.pdf

Overview

- Graph Definitions
- Graph Visualization
- Graph Layouts
- Interaction

Interaction

- One of the key ways we move beyond graph layout to graph visualization (InfoVis) is interaction with the graph
- Interaction is essential for understanding large graphs
- We can utilize various techniques

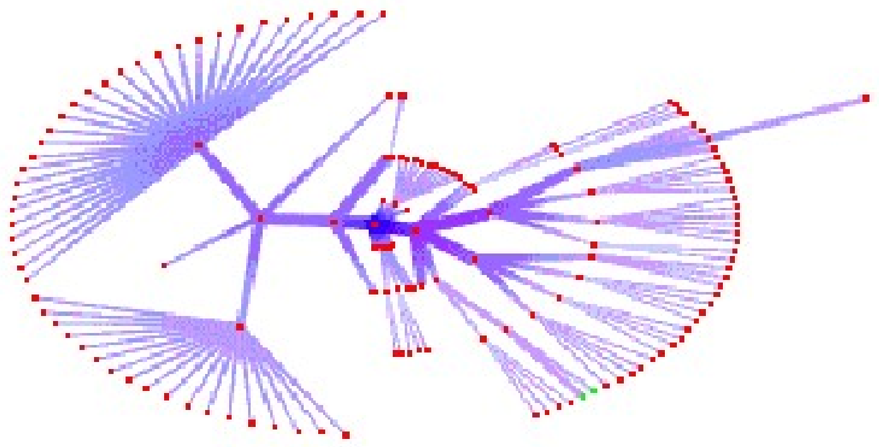
Filtering

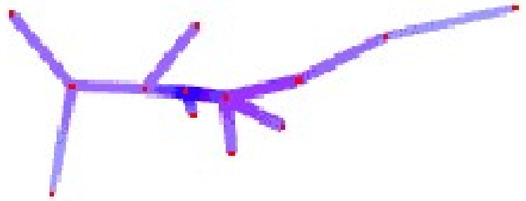
- “Show me something conditionally”
- Change the set of data items being presented based on some specific conditions.

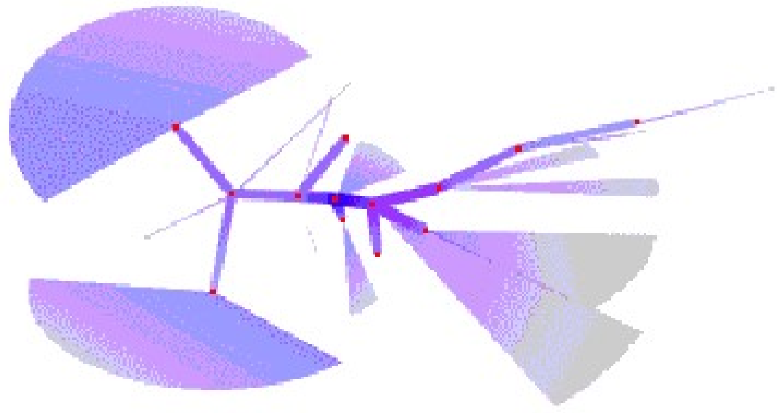
- Example
  - Dynamic query
  - Clustering

Methods of representing unselected nodes

- Ghosting
  - De-emphasizing or relegating nodes to background
- Hiding
  - Not displaying at all
- Grouping
  - Grouping under super -node representation







Clustering

- Structure-based Clustering
  - Most common in graph visualization
  - Often retain structure of graph

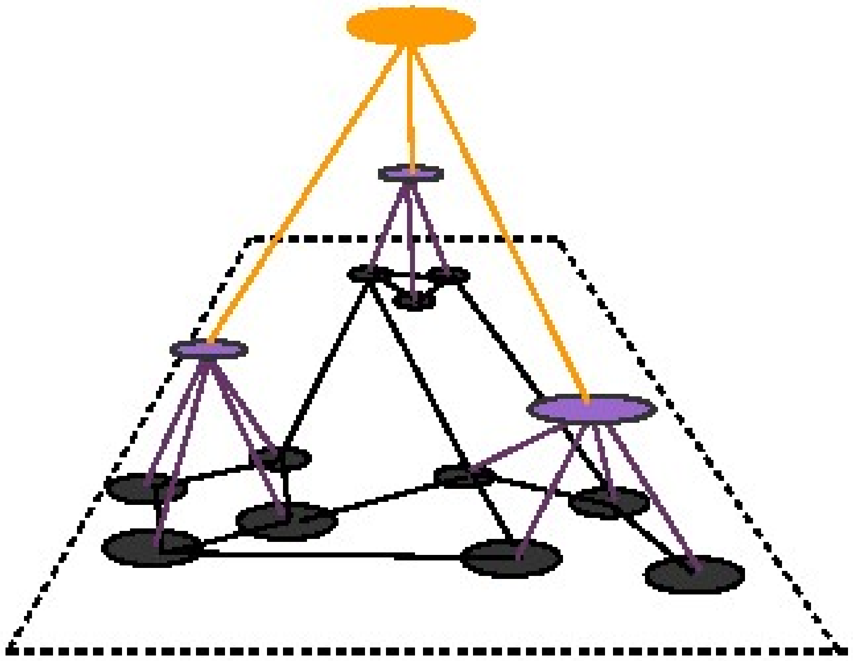
  ⇒Useful for user orientation
- Content-based Clustering
  - Application specific
  - Can be used for
    - Filtering: de-emphasis or removal of elements from view
    - Search: emphasis of an element or group of elements

Clustering – Goal & Technique

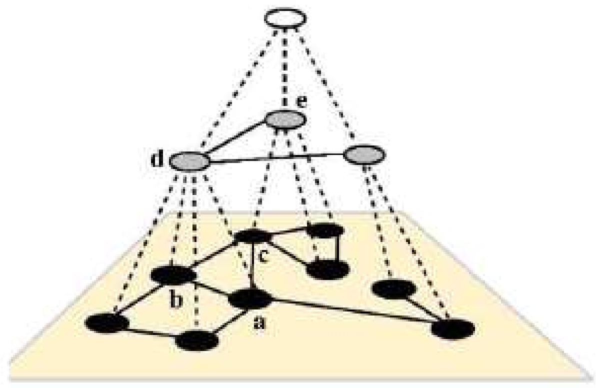
- Common goal
  - Finding disjoint clusters
  - Finding overlapping clusters
- Common technique
  - Least number of edges between neighbors (Ratio Cut technique in VLSI design)

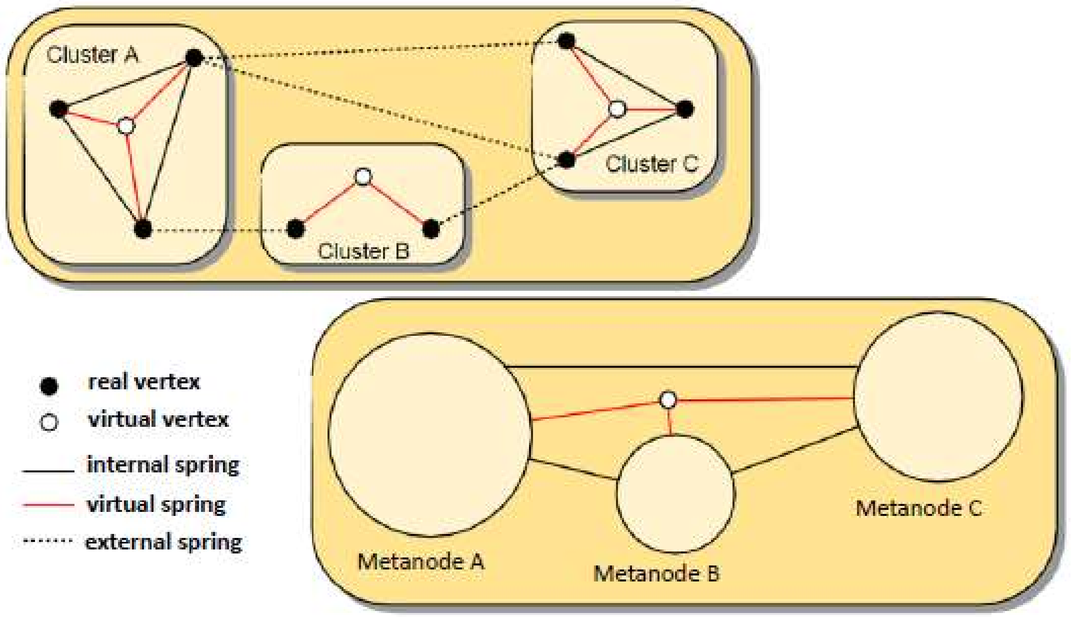
Hierarchical Clustering

- From successive application of clustering process
- Can be navigated as tree



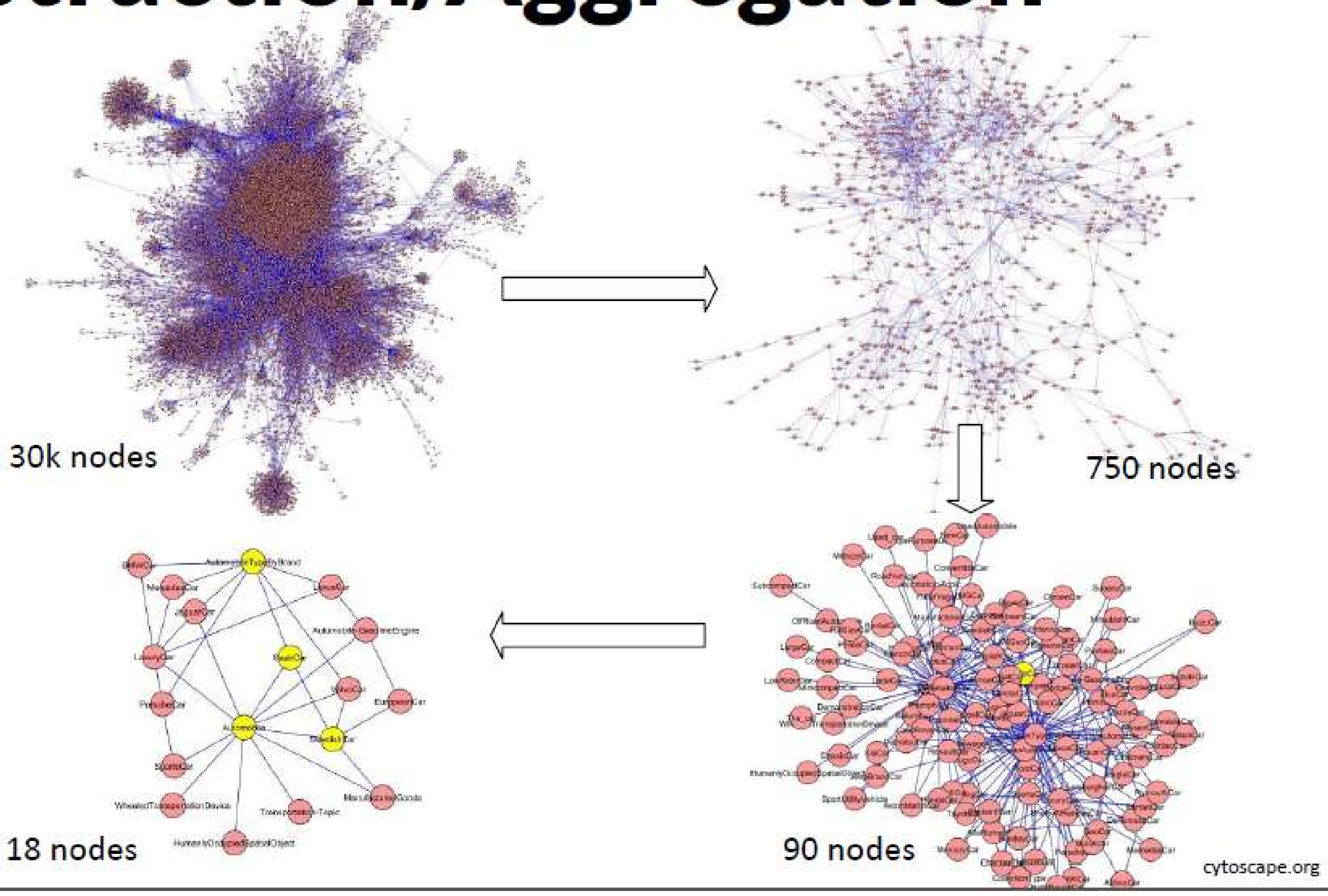
Addressing Computation Scalability



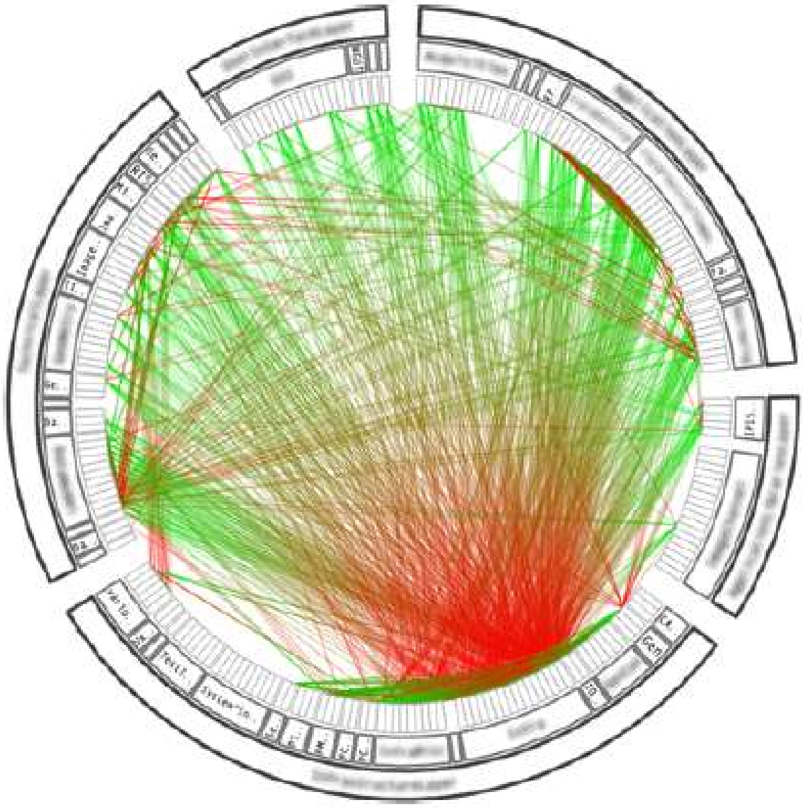


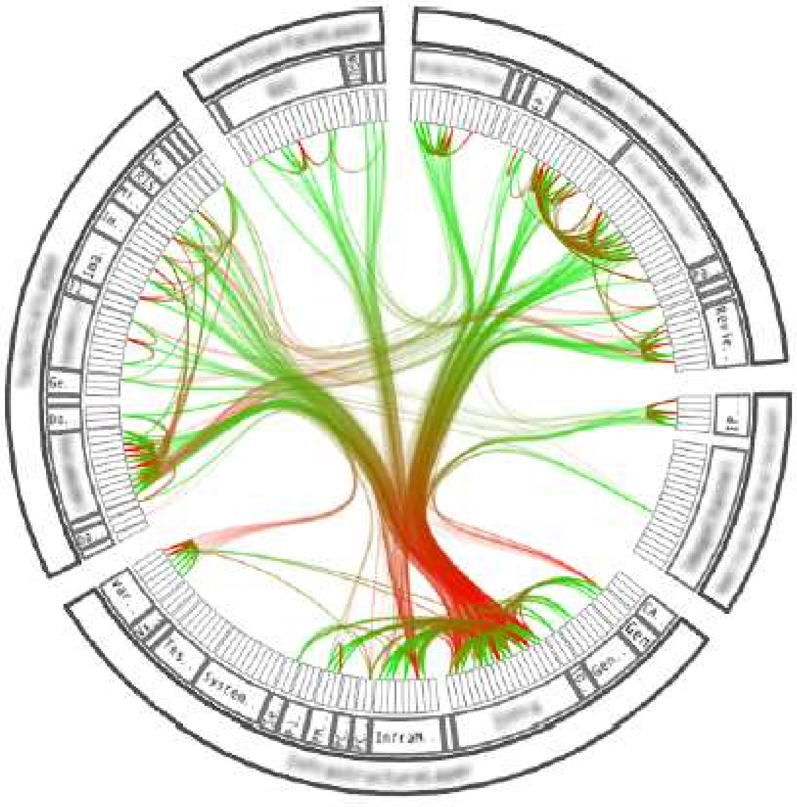
[Schulz 2004]

Abstraction/Aggregation



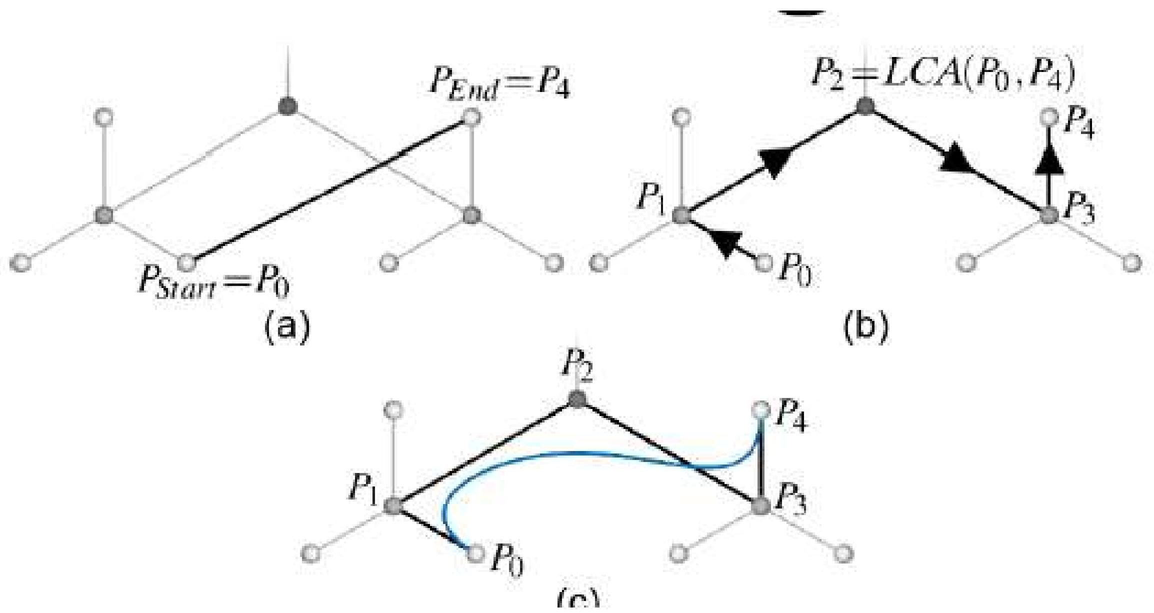
Reduce Clutter: Edge Bundling

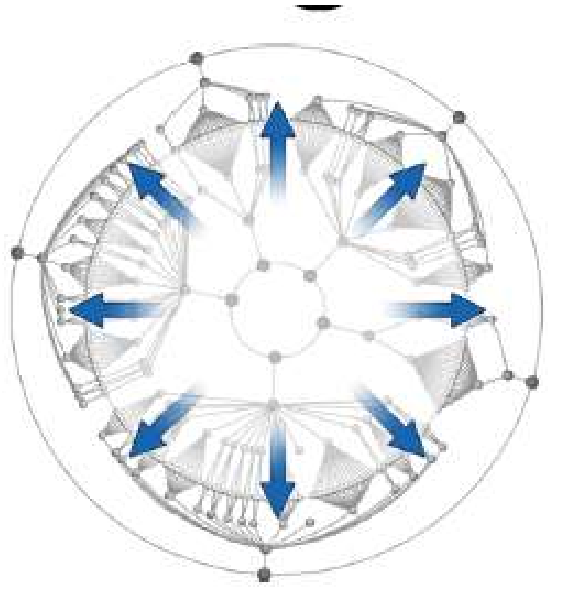




Hierarchical Edge Bundling

- Bundle edges that go from/to similar nodes together
  - Holten TVCG (InfoVis) ‘06





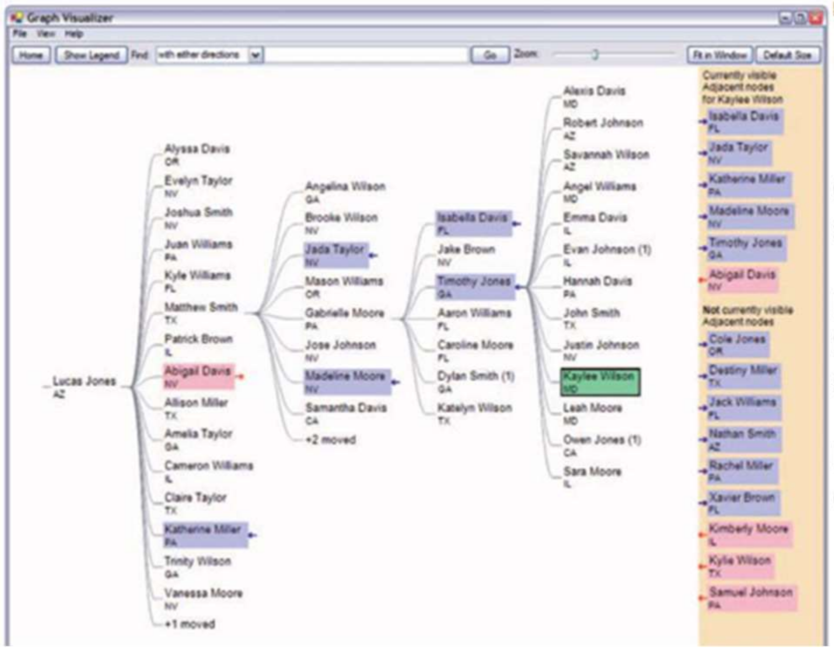
(a) Straight line connection between P0 and P4
(b) Path along the hierarchy between P0 and P4
(c) Spline curve depicting the connection between P0 and P4 by using the path from (b) as the control polygon.

Spanning Trees

- Graph can be visualized through minimum spanning trees
  - Additional edges added later
  - Very common technique
  - Helps with predictability
  - Visualization depends on starting point

Tree Plus

- Don’t draw entire graph
- Have a focus vertex, then incrementally expand and show connections (min span tree) from there.
- Interaction:
  - Singleclick: Show connections via highlight
  - Doubleclick: New focus vertex
- Smooth animated change in focus
- “Plant a seed and watch it grow”

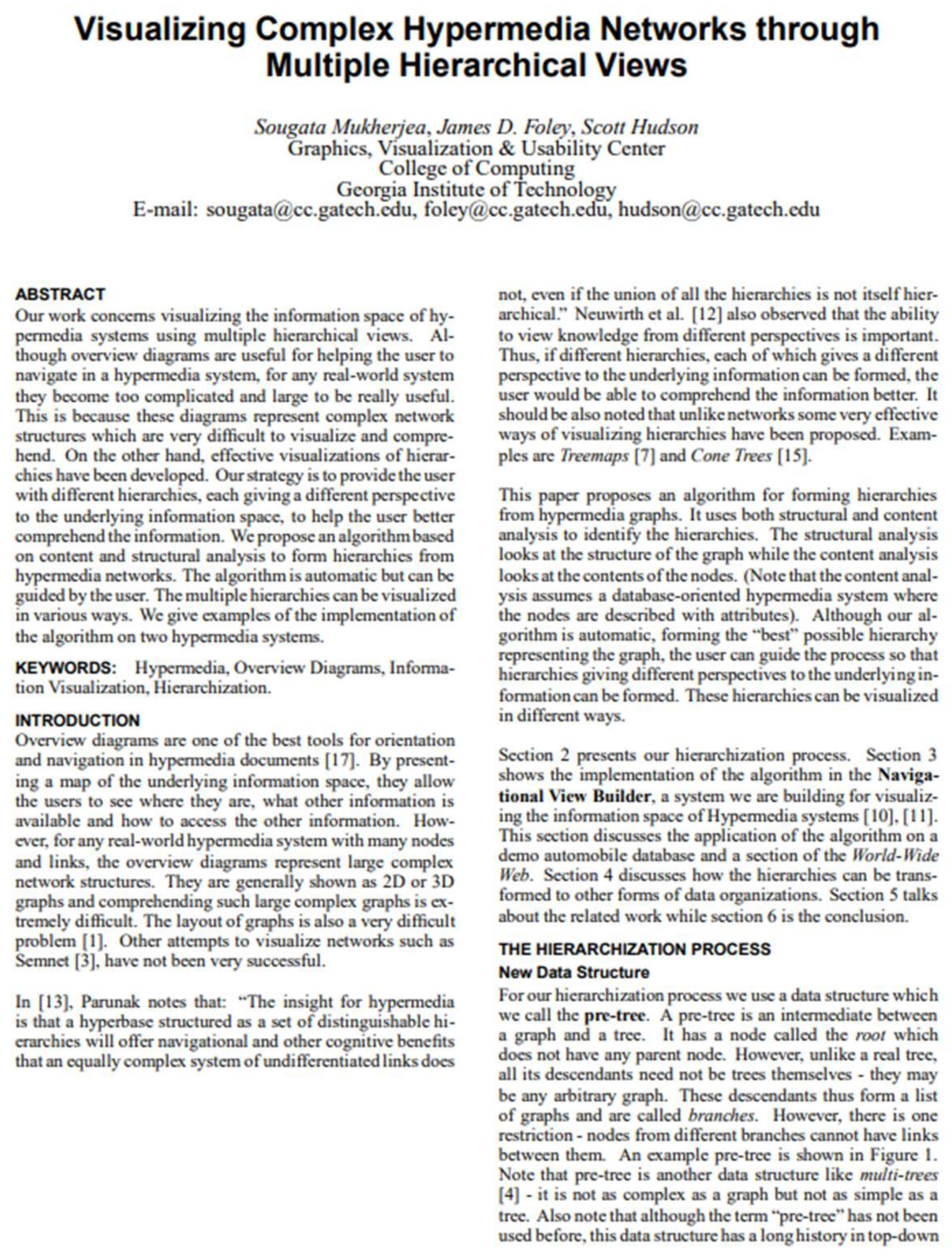


Green – current selection

Blue – vertices from current selection

Red – vertices to current selection

Lee et al TVCG ‘06



Visualizing Complex Hypermedia Networks through Multiple Hierarchical Views

Sougata Mukherjea, James D. Foley, Scott Hudson
Graphics, Visualization & Usability Center
College of Computing
Georgia Institute of Technology
E-mail: sougata@cc.gatech.edu, foley@cc.gatech.edu, hudson@cc.gatech.edu

ABSTRACT
Our work concerns visualizing the information space of hypermedia systems using multiple hierarchical views. Although overview diagrams are useful for helping the user to navigate in a hypermedia system, for any real-world system they become too complicated and large to be really useful. This is because these diagrams represent complex network structures which are very difficult to visualize and comprehend. On the other hand, effective visualizations of hierarchies have been developed. Our strategy is to provide the user with different hierarchies, each giving a different perspective to the underlying information space, to help the user better comprehend the information. We propose an algorithm based on content and structural analysis to form hierarchies from hypermedia networks. The algorithm is automatic but can be guided by the user. The multiple hierarchies can be visualized in various ways. We give examples of the implementation of the algorithm on two hypermedia systems.

KEYWORDS: Hypermedia, Overview Diagrams, Information Visualization, Hierarchization.

INTRODUCTION

Overview diagrams are one of the best tools for orientation and navigation in hypermedia documents [17]. By presenting a map of the underlying information space, they allow the users to see where they are, what other information is available and how to access the other information. However, for any real-world hypermedia system with many nodes and links, the overview diagrams represent large complex network structures. They are generally shown as 2D or 3D graphs and comprehending such large complex graphs is extremely difficult. The layout of graphs is also a very difficult problem [1]. Other attempts to visualize networks such as Semnet [3], have not been very successful.

In [13], Parunak notes that: "The insight for hypermedia is that a hyperbase structured as a set of distinguishable hierarchies will offer navigational and other cognitive benefits that an equally complex system of undifferentiated links does not, even if the union of all the hierarchies is not itself hierarchical." Neuwirth et al. [12] also observed that the ability to view knowledge from different perspectives is important. Thus, if different hierarchies, each of which gives a different perspective to the underlying information can be formed, the user would be able to comprehend the information better. It should be also noted that unlike networks some very effective ways of visualizing hierarchies have been proposed. Examples are *Treemaps* [7] and *Cone Trees* [15].

This paper proposes an algorithm for forming hierarchies from hypermedia graphs. It uses both structural and content analysis to identify the hierarchies. The structural analysis looks at the structure of the graph while the content analysis looks at the contents of the nodes. (Note that the content analysis assumes a database-oriented hypermedia system where the nodes are described with attributes). Although our algorithm is automatic, forming the "best" possible hierarchy representing the graph, the user can guide the process so that hierarchies giving different perspectives to the underlying information can be formed. These hierarchies can be visualized in different ways.

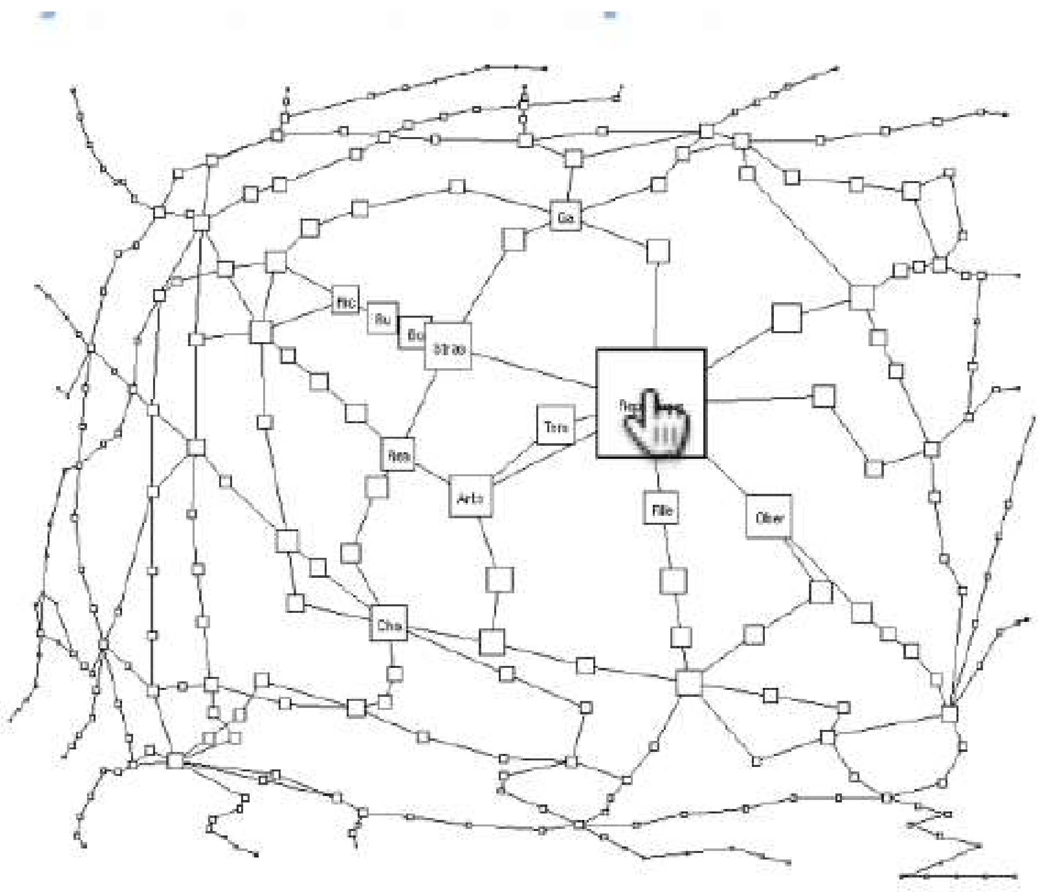
Section 2 presents our hierarchization process. Section 3 shows the implementation of the algorithm in the **Navigational View Builder**, a system we are building for visualizing the information space of Hypermedia systems [10], [11]. This section discusses the application of the algorithm on a demo automobile database and a section of the *World-Wide Web*. Section 4 discusses how the hierarchies can be transformed to other forms of data organizations. Section 5 talks about the related work while section 6 is the conclusion.

THE HIERARCHIZATION PROCESS

New Data Structure

For our hierarchization process we use a data structure which we call the **pre-tree**. A pre-tree is an intermediate between a graph and a tree. It has a node called the *root* which does not have any parent node. However, unlike a real tree, all its descendants need not be trees themselves - they may be any arbitrary graph. These descendants thus form a list of graphs and are called *branches*. However, there is one restriction - nodes from different branches cannot have links between them. An example pre-tree is shown in Figure 1. Note that pre-tree is another data structure like *multi-trees* [4] - it is not as complex as a graph but not as simple as a tree. Also note that although the term "pre-tree" has not been used before, this data structure has a long history in top-down

Focus + Context – Fisheye View of Graphs

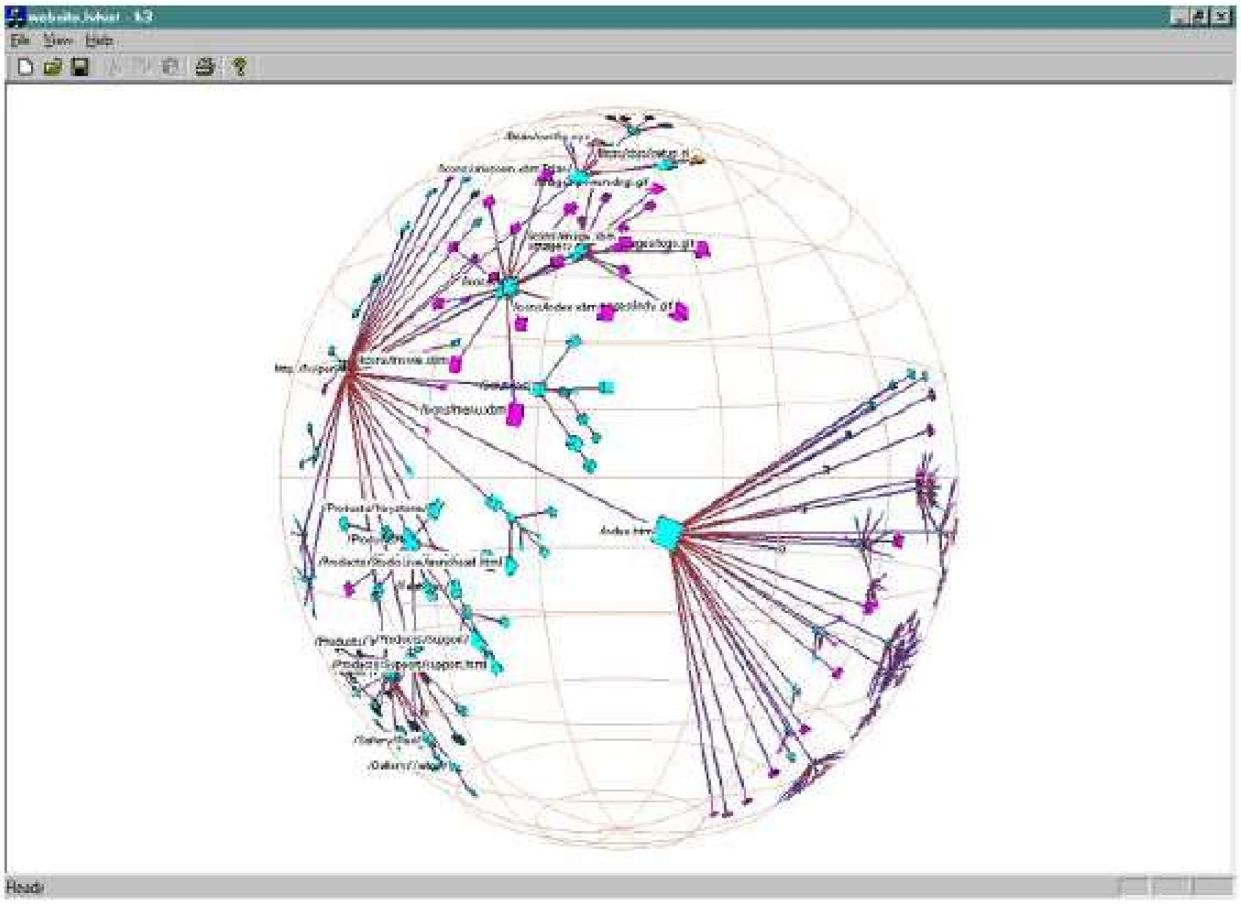


Sarkar and Brown, 1992

Problem with Fisheye

- Distortion should also be applied to links
  - Prohibitively slow (polyline)
- Alternative:
  - Continue using lines
  - Can result in unintended line crossings
- Other Alternative
  - Combine layout with focus + context
  - ⇒ Hyperbolic viewer

H3Viewer – Viewing Graphs in 3D Hyperbolic Space

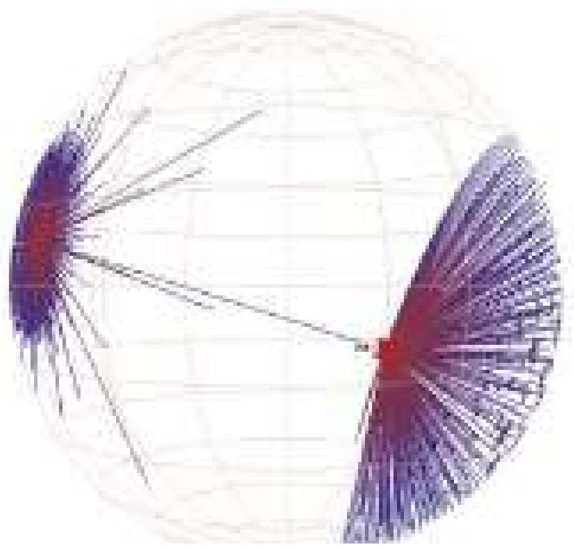
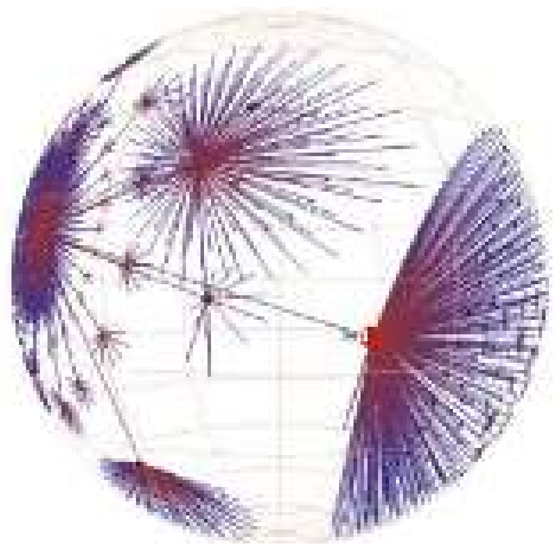


H3 Viewer - Technique

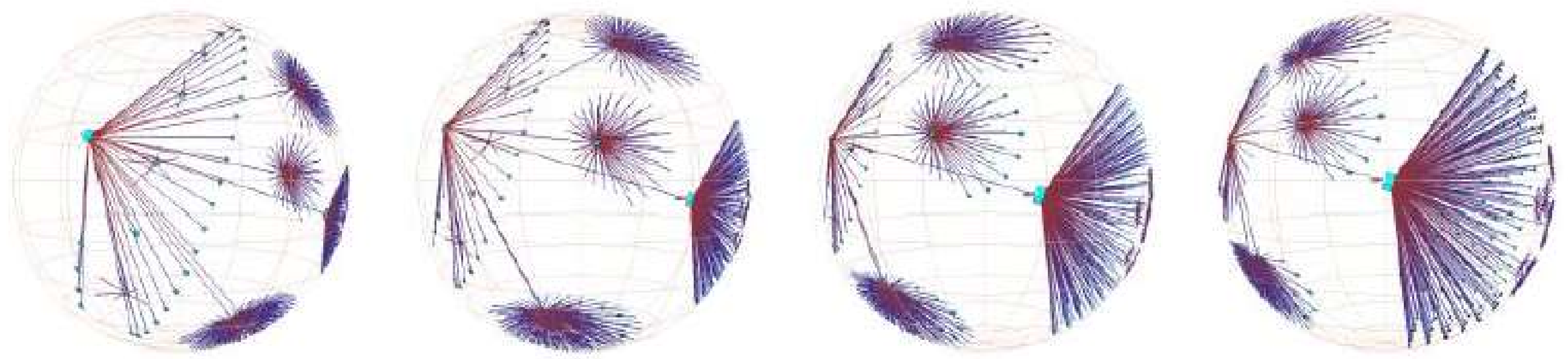
- Find a spanning tree from an input graph
- Layout algorithm
  - Nodes are laid out on the surface of a hemisphere
  - A bottom up pass to estimate the radius needed for each hemisphere
  - A top down pass to place each child node on its parental hemisphere’s surface

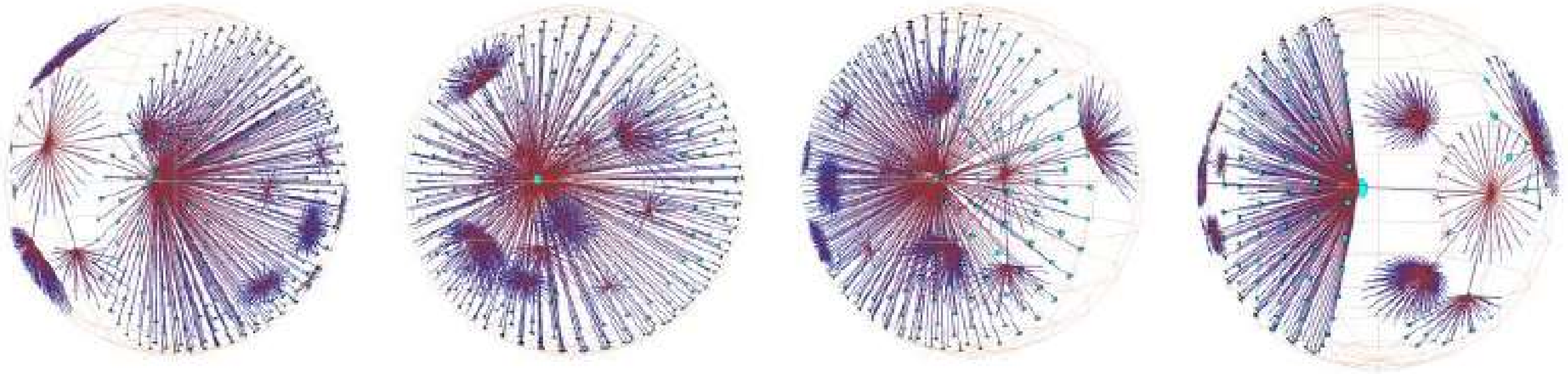
H3 Viewer - Drawing

- Showing less of the context surrounding the node of interest during interactive browsing
- Fill in more of the surrounding scene when the user is idle

H3 Viewer - Navigation


Translation of a node to the center


Rotation around the same node

Multiple Views

MatrixExplorer

- Provides matrix view in combination with node-link and various operations for gaining different perspectives

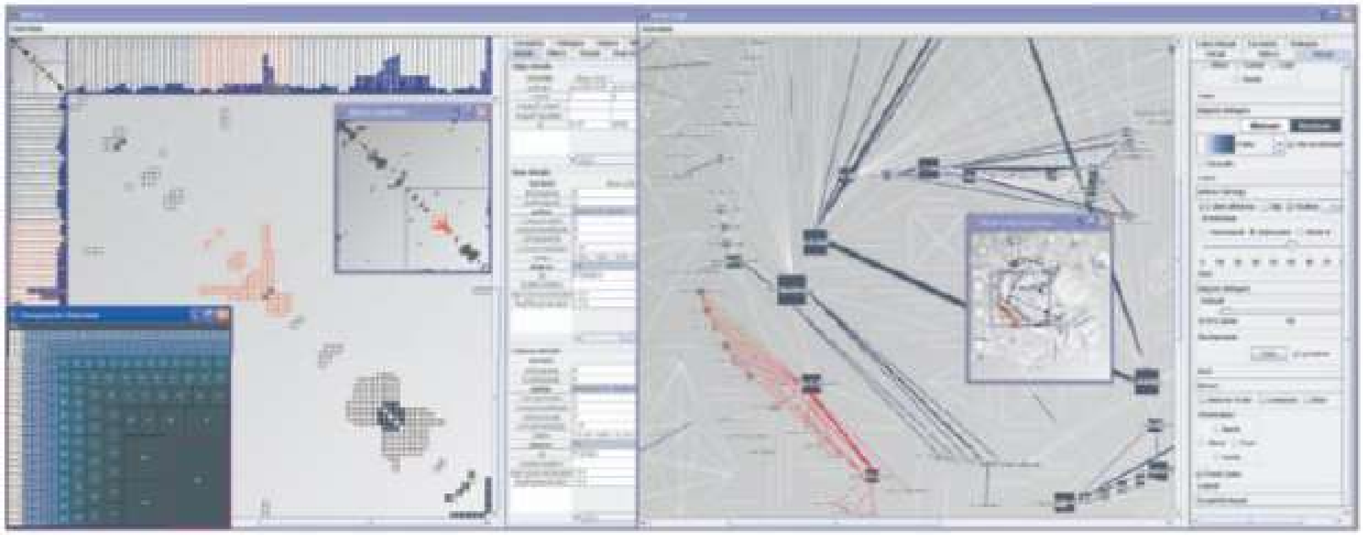
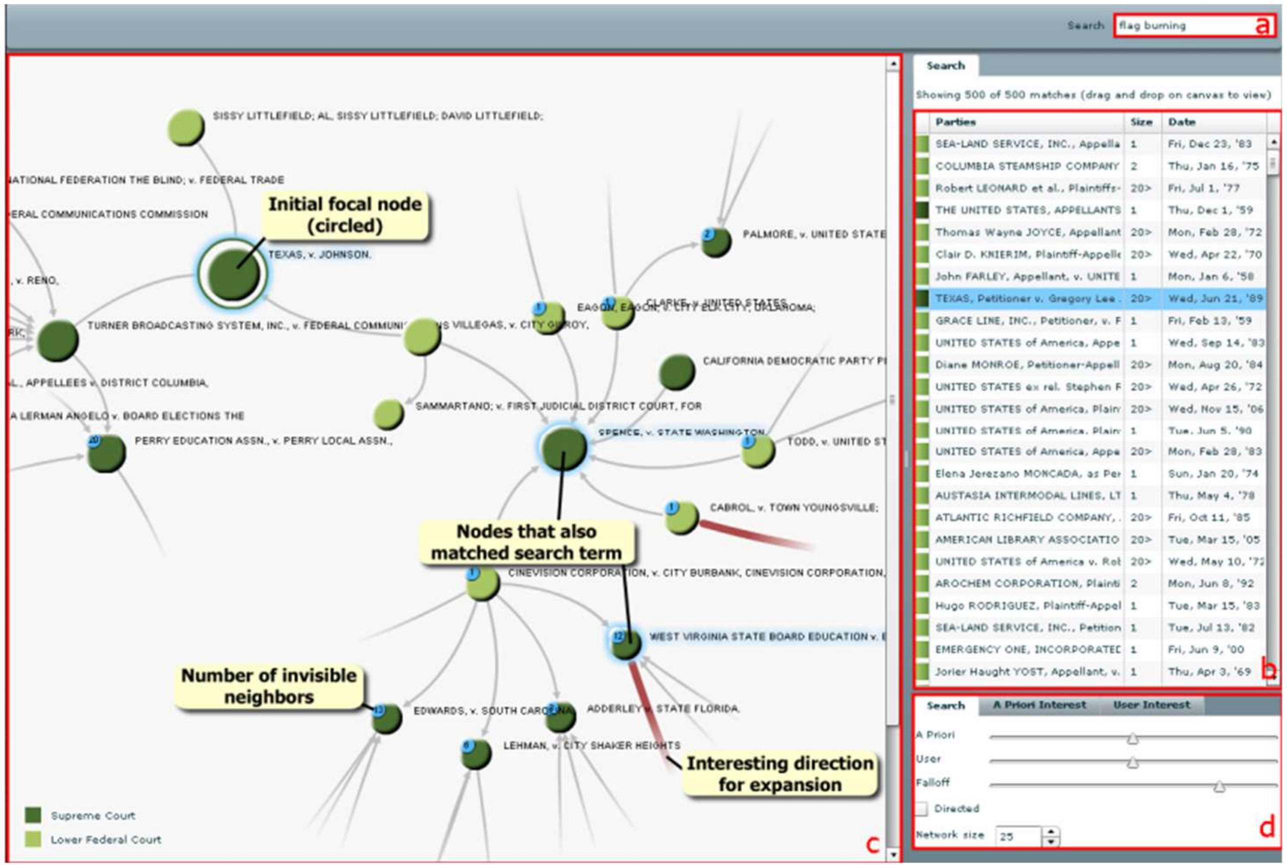


Fig. 1. MatrixExplorer showing two synchronized representations of the same network: matrix on the left and node-link on the right.

Multiple Views – Search & Visualization

- "Search, show context, expand on demand"
- Show some of the details, rather than high level structure
- Allow users to focus on particular nodes
- Adapt DOI algorithm from trees to graphs
- Rely heavily on interaction


Search
flag burning
a
SISSY LITTLEFIELD; AL. SISSY LITTLEFIELD; DAVID LITTLEFIELD;
NATIONAL FEDERATION THE BLIND; v. FEDERAL TRADE
ERAL COMMUNICATIONS COMMISSION
Initial focal node (circled)
TEXAS, v. JOHNSON.
v. RENO,
PALMORE, v. UNITED STATE
TURNER BROADCASTING SYSTEM, INC., v. FEDERAL COMMUNI
S VILLEGAS, v. CITY OF GILROY,
CALIFORNIA DEMOCRATIC PARTY PI
L., APPELLEES v. DISTRICT COLUMBIA,
A LERMAN ANGELO v. BOARD ELECTIONS THE
SAMMARTANO; v. FIRST JUDICIAL DISTRICT COURT, FOR
PERRY EDUCATION ASSN., v. PERRY LOCAL ASSN.,
SPENCE, v. STATE WASHINGTON
TODD, v. UNITED ST
CABROL, v. TOWN YOUNGSVILLE;
Nodes that also matched search term
CINEVISION CORPORATION, v. CITY BURBANK, CINEVISION CORPORATION,
WEST VIRGINIA STATE BOARD EDUCATION v.
Number of invisible neighbors
EDWARDS, v. SOUTH CAROLINA
ADDERLEY v. STATE FLORIDA.
LEHMAN, v. CITY SHAKER HEIGHTS
Interesting direction for expansion
Supreme Court
Lower Federal Court
c
Search
b
Search
A Priori Interest
User Interest
A Priori
User
Falloff
Directed
Network size
25
d

Showing 500 of 500 matches (drag and drop on canvas to view)

	Parties	Size	Date
🟩	SEA-LAND SERVICE, INC., Appella	1	Fri, Dec 23, '83
🟩	COLUMBIA STEAMSHIP COMPANY	2	Thu, Jan 16, '75
🟩	Robert LEONARD et al., Plaintiffs-	20>	Fri, Jul 1, '77
🟩	THE UNITED STATES, APPELLANTS	1	Thu, Dec 1, '59
🟩	Thomas Wayne JOYCE, Appellant	20>	Mon, Feb 28, '72
🟩	Clair D. KNIERIM, Plaintiff-Appelle	20>	Wed, Apr 22, '70
🟩	John FARLEY, Appellant, v. UNITE	1	Mon, Jan 6, '58
🟩	TEXAS, Petitioner v. Gregory Lee .	20>	Wed, Jun 21, '89
🟩	GRACE LINE, INC., Petitioner, v. F	1	Fri, Feb 13, '59
🟩	UNITED STATES of America, Appe	1	Wed, Sep 14, '83
🟩	Diane MONROE, Petitioner-Appell	20>	Mon, Aug 20, '84
🟩	UNITED STATES ex rel. Stephen F	20>	Wed, Apr 26, '72
🟩	UNITED STATES of America, Plain	20>	Wed, Nov 15, '06
🟩	UNITED STATES of America, Plain	1	Tue, Jun 5, '90
🟩	UNITED STATES of America, Appe	20>	Mon, Feb 28, '83
🟩	Elena Jerezano MONCADA, as Per	1	Sun, Jan 20, '74
🟩	AUSTASIA INTERMODAL LINES, LT	1	Thu, May 4, '78
🟩	ATLANTIC RICHFIELD COMPANY, .	20>	Fri, Oct 11, '85
🟩	AMERICAN LIBRARY ASSOCIATIO	20>	Tue, Mar 15, '05
🟩	UNITED STATES of America v. Rol	20>	Wed, May 10, '72
🟩	AROCHEM CORPORATION, Plainti	2	Mon, Jun 8, '92
🟩	Hugo RODRIGUEZ, Plaintiff-Appel	1	Tue, Mar 15, '83
🟩	SEA-LAND SERVICE, INC., Petition	1	Tue, Jul 13, '82
🟩	EMERGENCY ONE, INCORPORATEL	1	Fri, Jun 9, '00
🟩	Jorier Haught YOST, Appellant, v.	1	Thu, Apr 3, '69

- Basic user interface layout. A user types a query in the searchbox which yields a number of hits presented in tabular form
- One of these hits can then be dragged to the main screen which shows the subgraph centered on that node. Other nodes that matched the user's search are highlighted in blue.
- Users can adapt the balance between different components of the DOI function and the size of the subgraph in a separate panel.

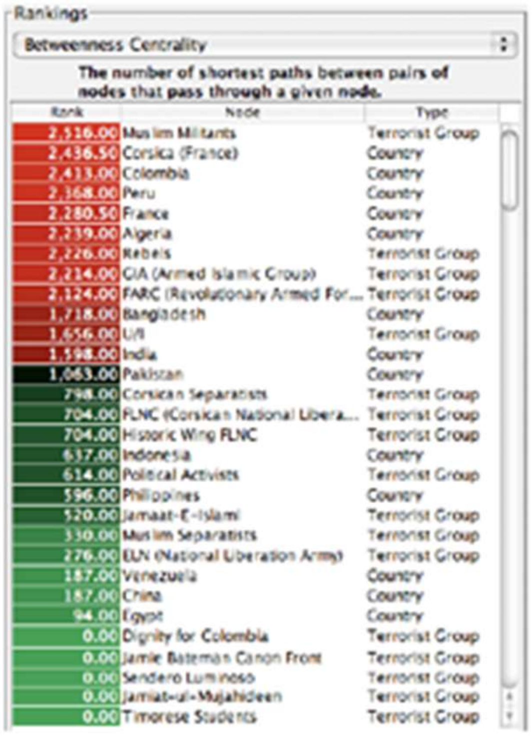
van Ham & Perer
TVCG (InfoVis) 2009

Social Action

- Combines graph structural analysis (ranking) with interactive visual exploration
- Multiple coordinated views
  - Lists by ranking for analysis data
  - Basic force-directed layout for graph vis

Perer & Shneiderman TVCG (InfoVis) ‘06

Multiple Coordinated Views



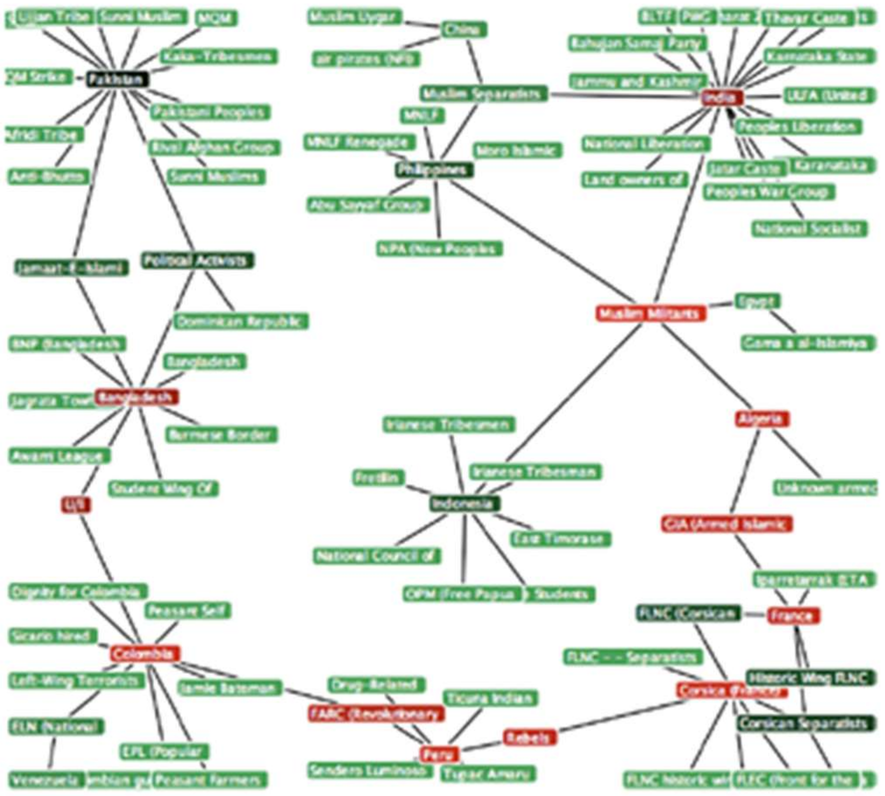
Rankings

Betweenness Centrality

The number of shortest paths between pairs of nodes that pass through a given node.

Rank	Node	Type
2,516.00	Muslim Militants	Terrorist Group
2,436.50	Corsica (France)	Country
2,413.00	Colombia	Country
2,368.00	Peru	Country
2,280.50	France	Country
2,239.00	Algeria	Country
2,226.00	Rebels	Terrorist Group
2,214.00	GIA (Armed Islamic Group)	Terrorist Group
2,124.00	FARC (Revolutionary Armed For...	Terrorist Group
1,718.00	Bangladesh	Country
1,656.00	U/I	Terrorist Group
1,598.00	India	Country
1,063.00	Pakistan	Country
798.00	Corsican Separatists	Terrorist Group
704.00	FLNC (Corsican National Libera...	Terrorist Group
704.00	Historic Wing FLNC	Terrorist Group
637.00	Indonesia	Country
614.00	Political Activists	Terrorist Group
596.00	Philippines	Country
520.00	Jamaat-E-Islami	Terrorist Group
330.00	Muslim Separatists	Terrorist Group
276.00	ELN (National Liberation Army)	Terrorist Group
187.00	Venezuela	Country
187.00	China	Country
94.00	Egypt	Country
0.00	Dignity for Colombia	Terrorist Group
0.00	Jamie Bateman Canon Front	Terrorist Group
0.00	Sendero Luminoso	Terrorist Group
0.00	Jamiat-ul-Mujahideen	Terrorist Group
0.00	Timorese Students	Terrorist Group

(a) Ordered list of 97 nodes in the largest connected component of the terrorism network in 1996. The nodes are ranked according to their betweenness centrality.



(b) Network visualization of the same 97 nodes, colored according to their ranking. The nodes with highest betweenness rankings, sometimes referred to as "gatekeepers", are painted red.

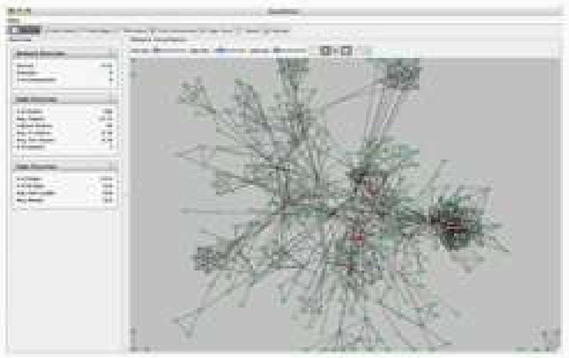
Social Network Attributes

- **Bary center** – total shortest path of a node to all other nodes
- **Betweenness centrality** – how often a node appears on the shortest path between all other nodes
- **Closeness centrality** – how close a node is compared to all other nodes
- **Cut-points** – the subgraph becomes disconnected if the node is removed
- **Degree** – number of connections for node
- **Power centrality** – how linked a node is to rest of network
- **HITs** – “hubs and authorities” measure
- **Page Rank** – number and quality of in-links

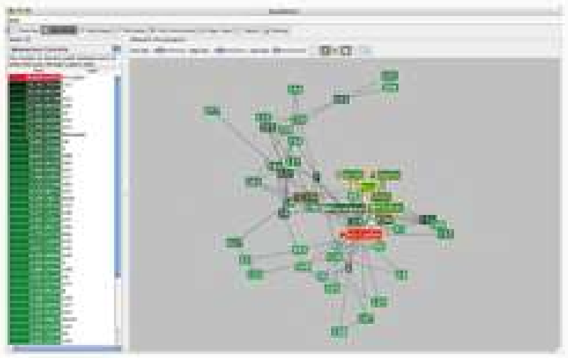
Attribute Ranking

- Run these measures on all nodes and rank them
- Sort the rankings and show in lists and scatterplots
- Allow user to filter based on rankings
- Can aggregate rankings for cohesive subgroups of nodes

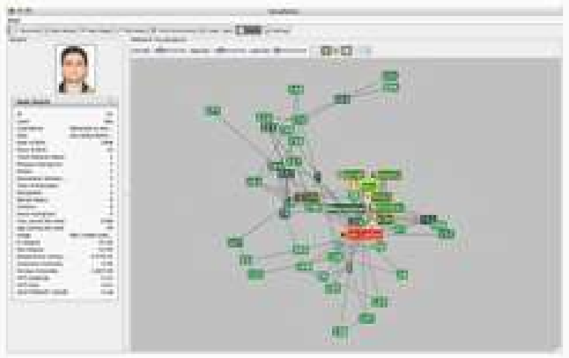
Visual Analysis



Users begin with an overview of the entire social network. On the left side, overview statistics that describe the overall structure are presented. On the right, the network is visualized using a force directed algorithm.



The gatekeepers are found using a statistical algorithm. Users filter out the unimportant nodes using a dynamic slider which simplifies the visualization while maintaining the node positions and structure of the network.



Labels are always given priority so users can understand what the data represents. When user selects a node, neighbors are highlighted and details appear on the left. In order to protect sensitive information, node labels have been anonymized except for those individuals publicly identified in the Zacarias Moussaoui trial.

Reading

1. D. Holten,, **Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data**. *IEEE Trans. on Visualization and Computer Graphics*, 2006. Available at: https://www.researchgate.net/publication/6715561

2. T.Munzner **H3: Laying Out Large Directed Graphs in 3D Hyperbolic Space.** *Proceedings of the 1997 IEEE Symposium on Information Visualization, 1997.* Available at: https://graphics.stanford.edu/papers/h3/

3. A. Perer, B. Shneiderman, **Balancing Systematic and Flexible Exploration of Social Networks**, *IEEE Trans. on Visualization and Computer Graphics*, 2006. Available at: https://www.researchgate.net/publication/6715555