# Visualisation of Trees

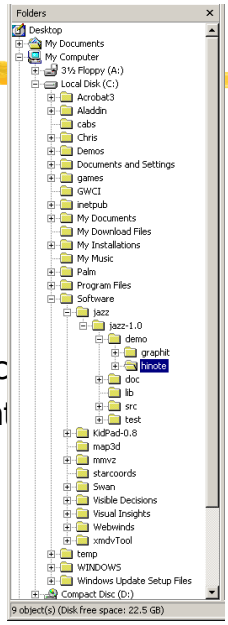
**Jean-Daniel Fekete**
**INRIA Saclay – Île-de-France**
**Projet AVIZ**

## Trees (Hierarchies)

- What is a tree?
  - DAG, one parent per node
  - Items + structure (nodes + links)
  - Table model: Add parent pointer attribute
- Examples
  - filesystem, family, classification/taxonomy, org charts, toc, data structures, menus
- Tasks
  - All previous tasks plus structure-based tasks:
  - Find descendants, ancestors, siblings, cousins
  - Overall structure, height, breadth, dense/sparse areas

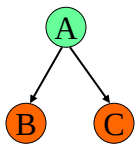
# Tree Visualization

- Example: Outliner
- Why is tree visualization hard?
  - Structure AND items
  - Structure harder, consumes more spac
  - Data size grows very quickly (exponent
    - #nodes = $b^{height}$

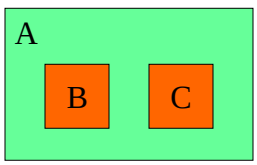



# 2 Approaches

- Connection (node & link) today
  - outliner




- Containment (node in node)
  - Venn diagram



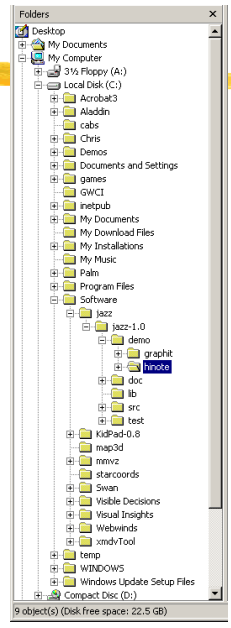
# Tree Properties

- Structure vs. attributes
  - Attributes only (multi-dimensional viz)
  - Structure only (1 attribute, e.g. name)
  - Structure + attributes
- Branching factor
- Fixed level, categorical

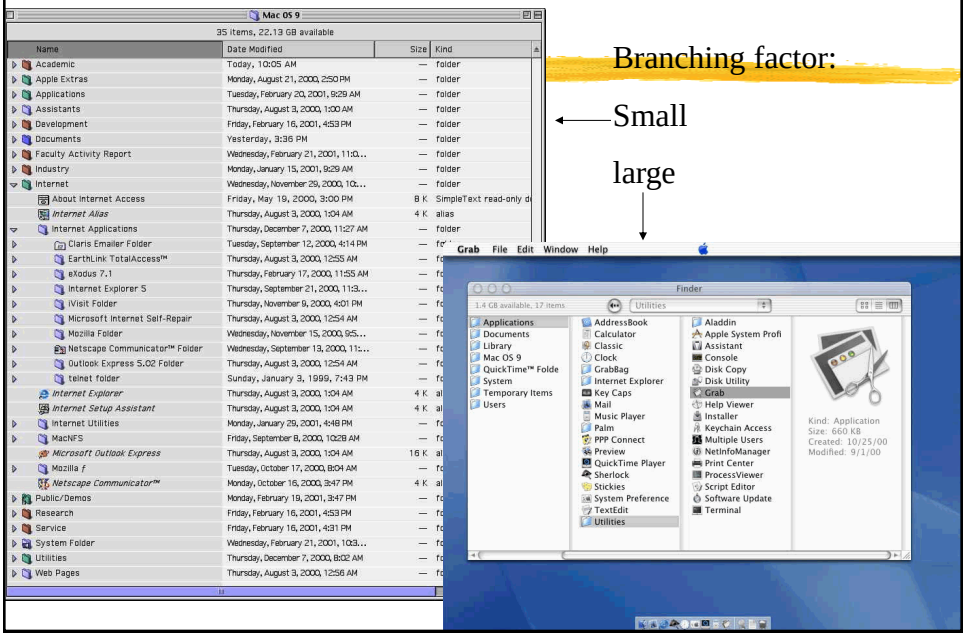
# Outliner

- Good for directed search tasks
- Not good for learning structure
- No attributes
- Apx 50 items visible
- Lose path to root for deep nodes
- Scroll bar!

cant see all the tree structure
•Scroll bar suck
•Structure only
•Lost screen space
•50 nodes
•Filtering open/close
•Search tasks ok
•Browsing not good
•Icons?



# Mac Finder



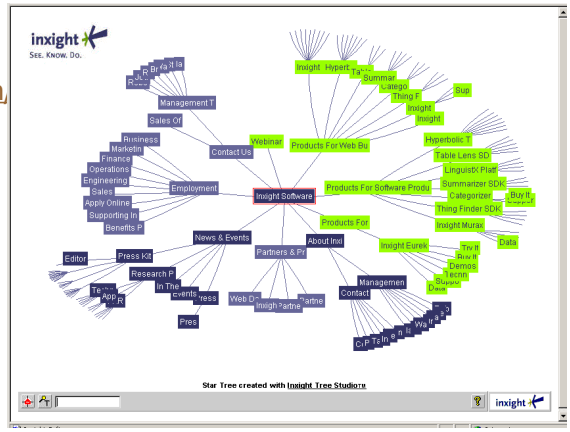

Branching factor:

Small

large

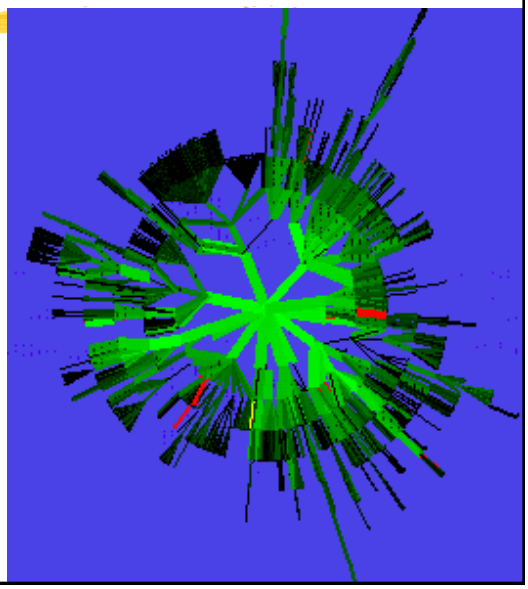
# Hyperbolic Trees

- Rao, "Hyperbolic Tree"
  - David, Harsha
- http://startree.inxight.com/
- Xerox PARC
- Inxight



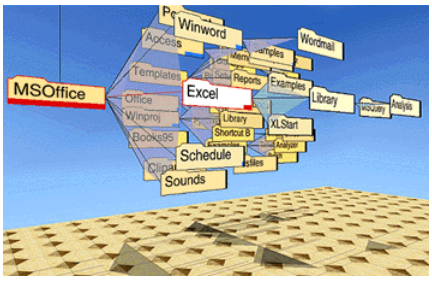
# Disk Tree

- Ed Chi, Xerox PARC



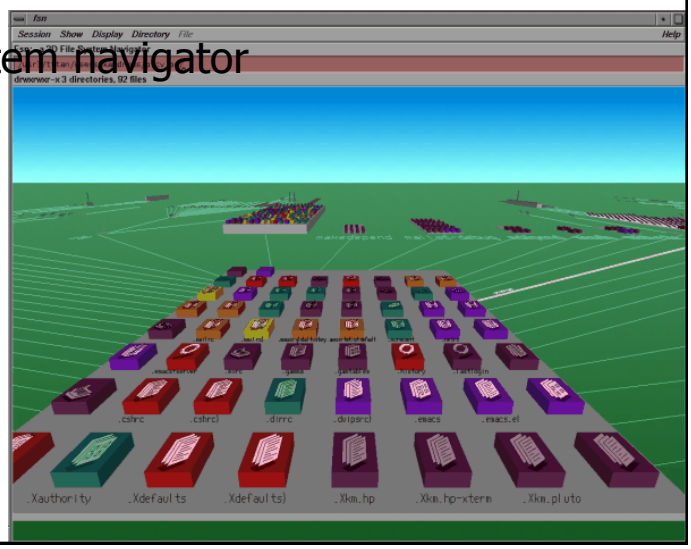
# Cone Trees

- Robertson, "ConeTrees"
  - Anuj, Atul
- Xerox PARC

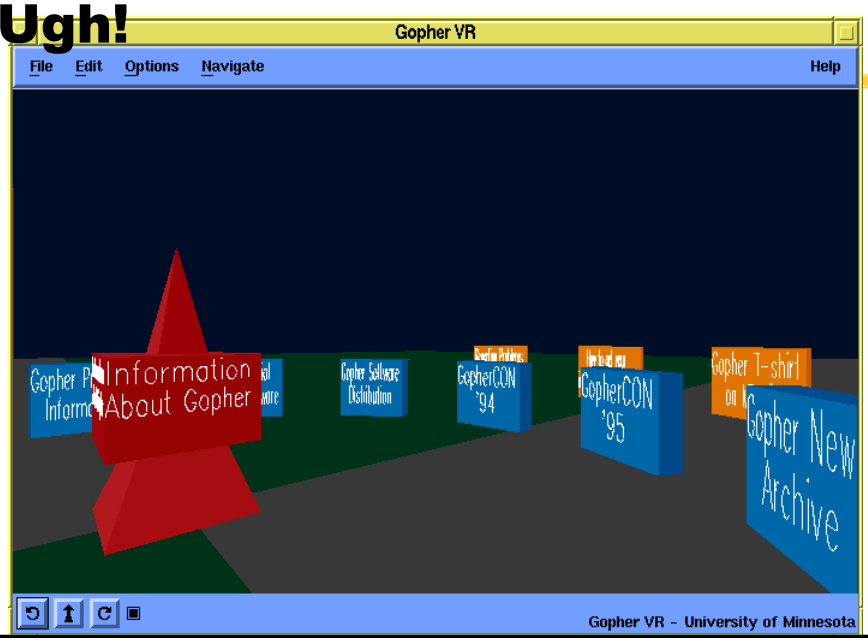


# FSN

- SGI file system navigator
- Jurassic Park

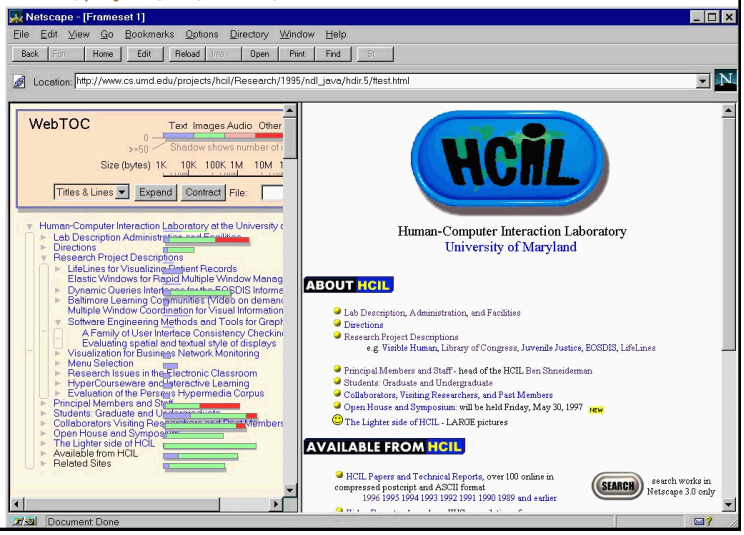



# Ugh!



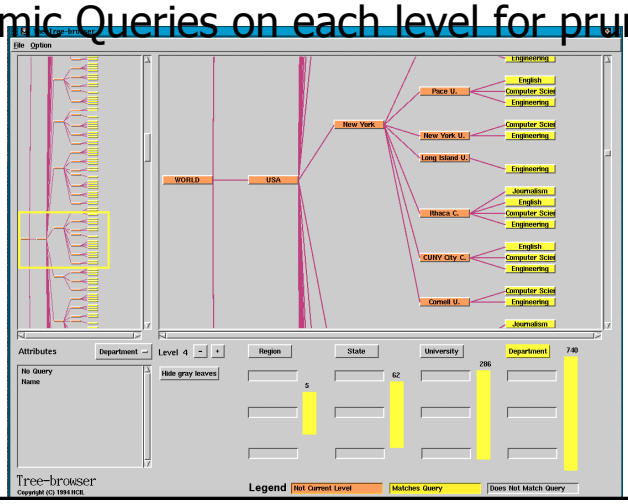
# WebTOC

- Website map: Outliner + size attributes
- http://www.cs.umd.edu/projects/hcil/webtoc/fhcil.html

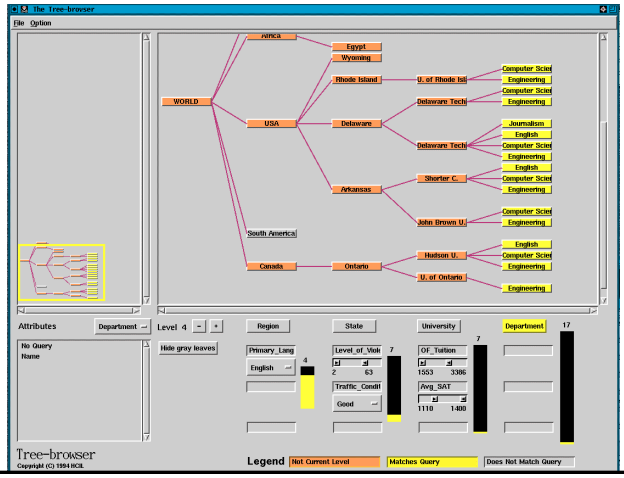



# PDQ Trees

- Overview+Detail of 2D tree layout
- Dynamic Queries on each level for pruning

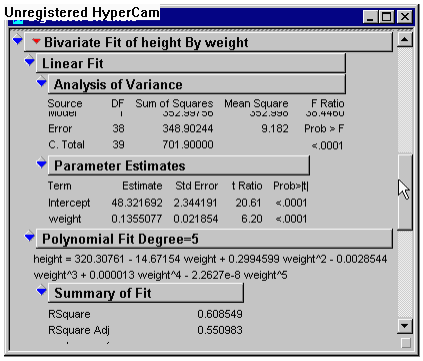


# PDQ Trees



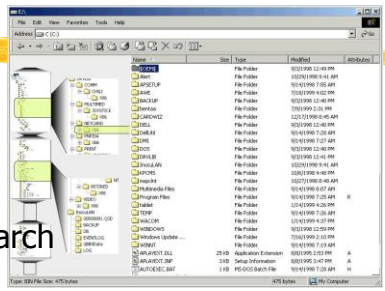

# Nifty App of the Day

- SAS JMP

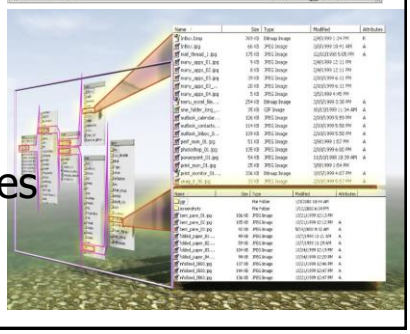


# Hard Problems

- Multiple foci
  - Robertson, Microsoft Research

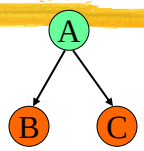


- Polyarchies:
  multiple inter-twined trees



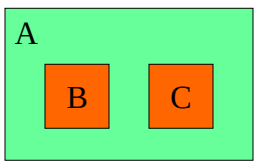
# 2 Approaches

- Connection (node & link)
  - Outliner, …

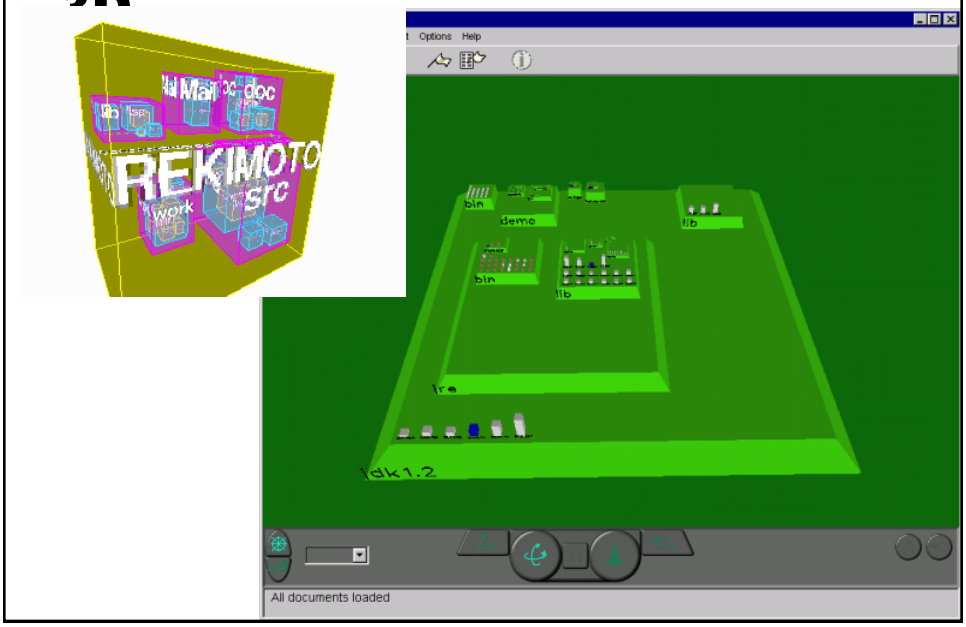



- Containment (node in node)
  - Venn diagram

today

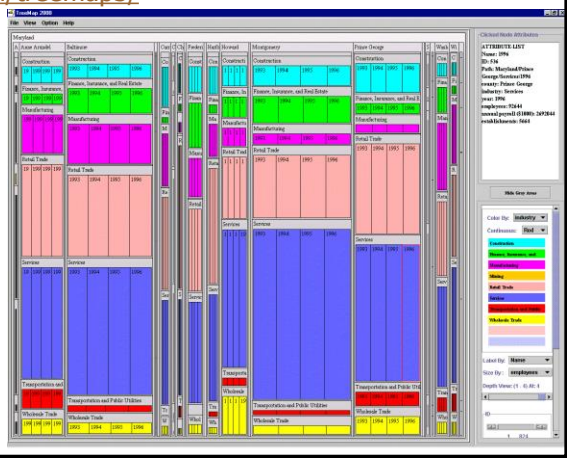



- Structure vs. attributes
  - Attributes only (multi-dimensional viz)
  - Structure only (1 attribute, e.g. name)
  - Structure + attributes



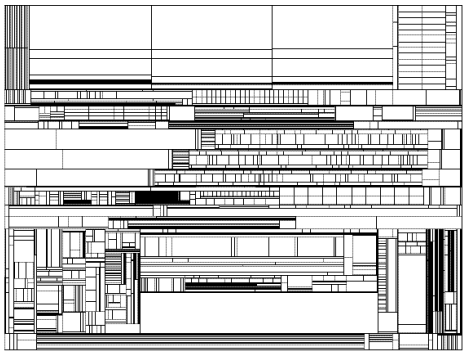

# Treemaps

- Shneiderman, "Treemaps"
  - Vishal, Jeevak
- http://www.cs.umd.edu/hcil/treemap3/
- Maryland

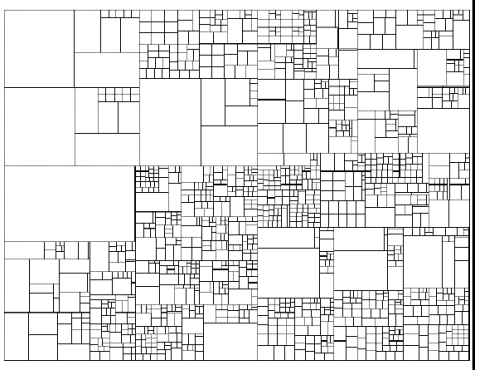


# Squared Treemaps

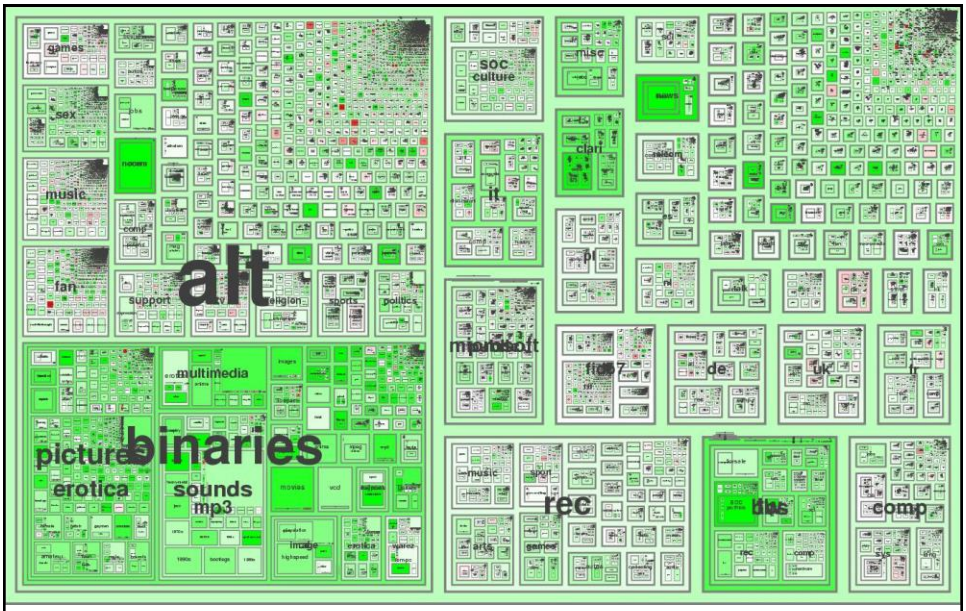
- Wattenberg
- Van Wijk



(a) File system



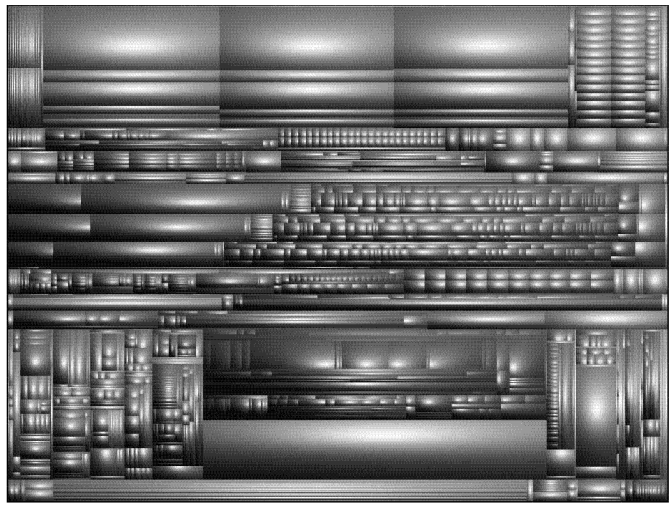
(a) File system




- http://www.research.microsoft.com/~masmith/all_map.jpg

# Cushion Treemaps

- Van Wijk
- http://www.win.tue.nl/sequoiaview/

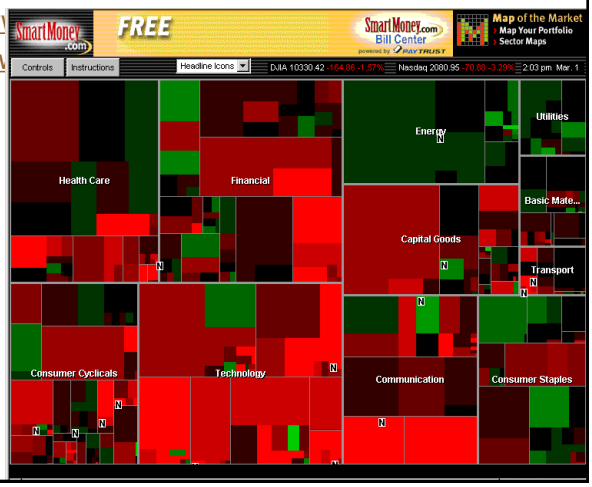


# Dynamic Queries

- http://www.cs

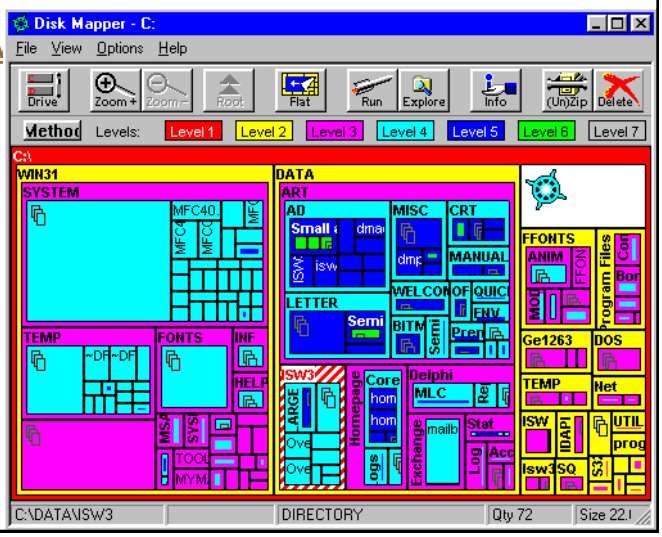
# Treemaps on the Web

- Map of the Market: http://www.smartmoney.com/marketmap/
- People Map: http://ww
- Coffee Map: http://ww



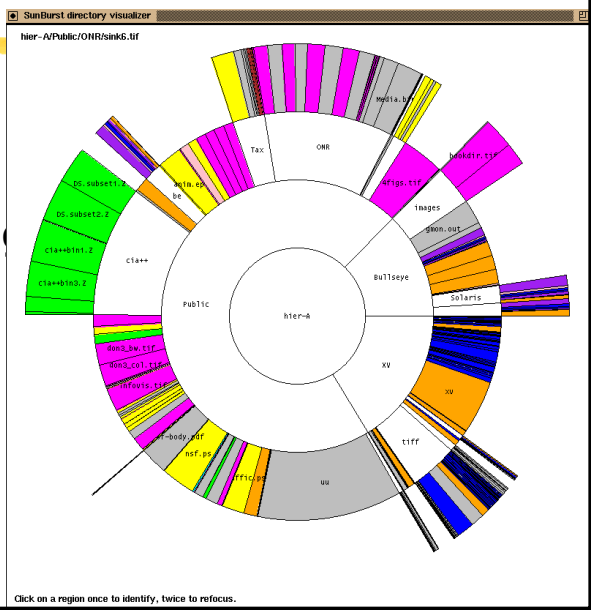

# DiskMapper

- http://www



# Sunburst

- Stasko, GaTech
- Radial layout
- Animated zoomin




# Sunburst vs. Treemap

- + Faster learning time: like pie chart
- + Details outward, instead of inward
- + Focus+context zooming

- - Not space filling
- - More space used by non-leaves
  - All leaves on 1-D space, perimeter
  - Treemap: 2-D space for leaves
- - Smaller scale?

# Network Visualization

**Jean-Daniel Fekete**
**INRIA Saclay – Île-de-France**

# Formal definition

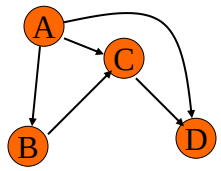
- A graph is:
  - A set of Vertices V=$\{v_i\}$
  - A set of edges E=$\{e_j\}$ with $e=(v_s, v_d) \in$ VxV
  - When the order of the couple in E is meaningful, the graph is **directed**, otherwise, it is **undirected**
  - A graph is a mapping of V into V
  - From there, we can define several measures or intrinsic properties on a graph

# Properties of a Graph

- From a graph structure, we can compute the following properties:
  - Degree (input/output) of a vertex
  - Distance between two vertices (min number of hops from one to the other)
  - Connected components (weak/strong)
  - Diameter
  - Centrality of a vertex
  - Density of a graph

# Two classical visual representations of a Graph

- Node Link Diagram
  - A node represents a vertex
  - A link represents an edge
- Adjacency Matrix
  - A line is a (source) vertex
  - A column is a (destination) vertex
  - A the intersection, the cell represents the edge




| ↗ | A | B | C | D |
|---|---|---|---|---|
| A | | X | X | X |
| B | | | X | |
| C | | | | X |
| D | | | | |

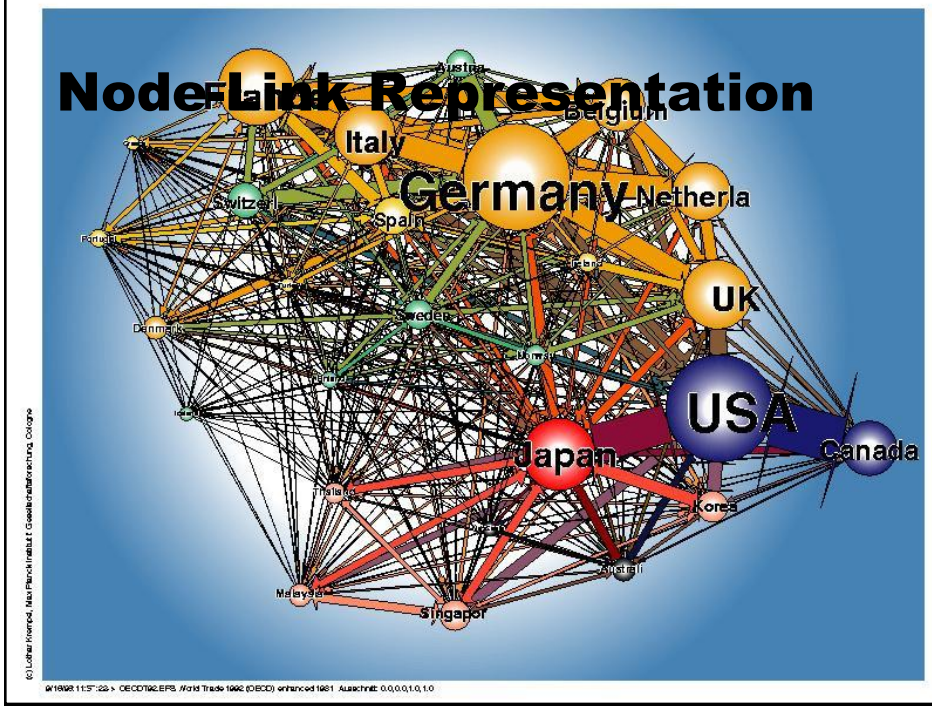
# Examples of Tasks on Graphs

- The main tasks related to graphs are:
  - **Vertices**: find their degree, if they are isolated (degree=0), source only (no incoming edge), sink (no outgoing edges), neighbors
  - **Paths**: shortest paths, all paths, cycles
  - **Sub-Graph:** connected components, etc.
- The list is unbounded, all the application domains add more useful tasks. However, there is a bounded list of very generic tasks.

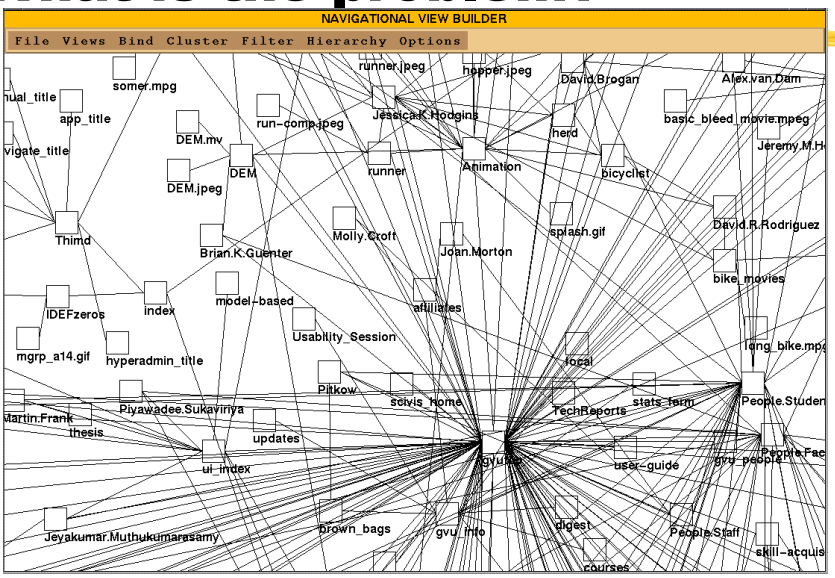
# Attributes of a Graph

- The Graph structure only defines a **topology**
- Attributes can also be associated to vertices and edges (I call the attributed graph a **Network)**
  - Examples: a label, weight, length, count…
- With attributes, new measures can be computed on a Network:
  - Shortest path according to edge length, cycles with constraints, etc.
- New tasks related to attributes are added
  - Shortest cycle going through the cities with the largest population

# Node-Link Representation



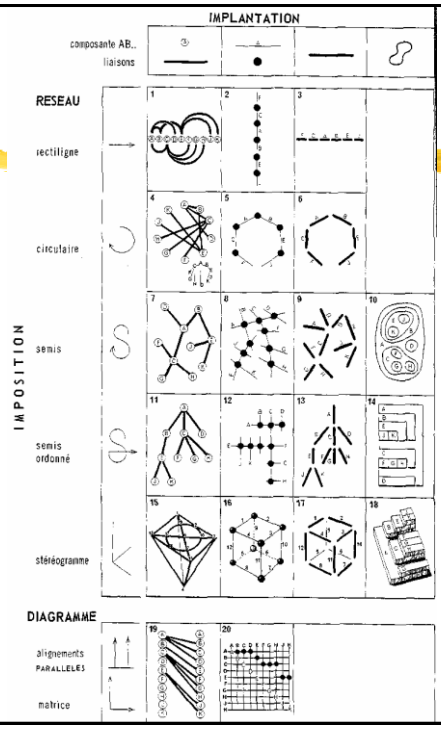

# What is the problem?



# Issues

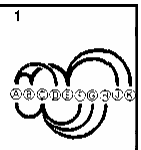
- Graph Layout
- Scalability
- Navigation

Beware of the hairball!

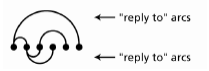


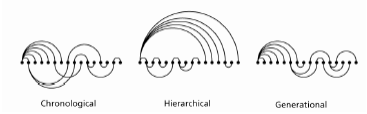

Bertin 67

# Linear layout



- **THREAD ARCS: An Email Thread Visualization** (Bernard Kerr, IBM)




- Arc Diagrams: The Shape of Song (Martin Wattenberg, IBM)
- http://www.bewitched.com/

# Linear Layout Theory

- Order vertices so that a structure appears
  - Layout φ
- Some well-known objective functions (Diaz et al. 02):
  - Bandwidth
  - MinLA
  - Cut Width
  - Min Cut
  - Sum Cut

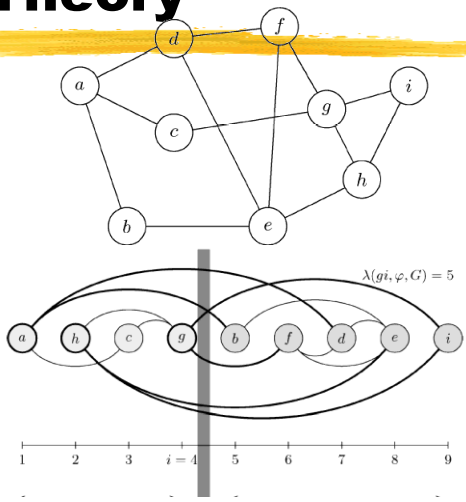
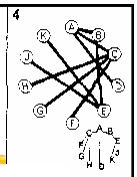



**Fig. 1**. A graph $G$ together with some layout measures and a graphical representation of the layout $\varphi = \{(a, 1), (b, 5), (c, 3), (d, 7), (e, 8), (f, 6), (g, 4), (j, 9), (h, 2)\}$.

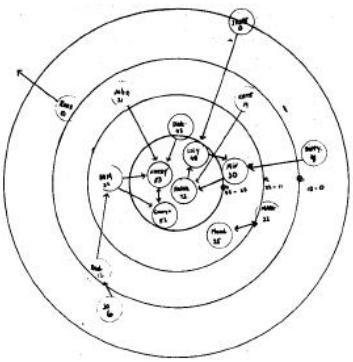
# Circular Layout

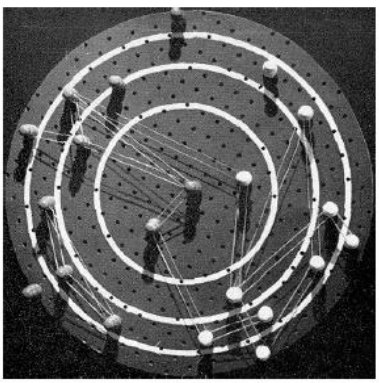


- Egocentric Social Networks
- Can be ordered too

## centrality visualization

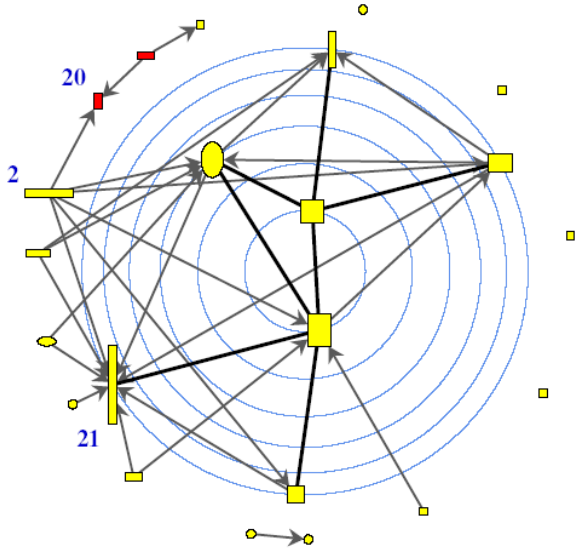
Northway (1940)





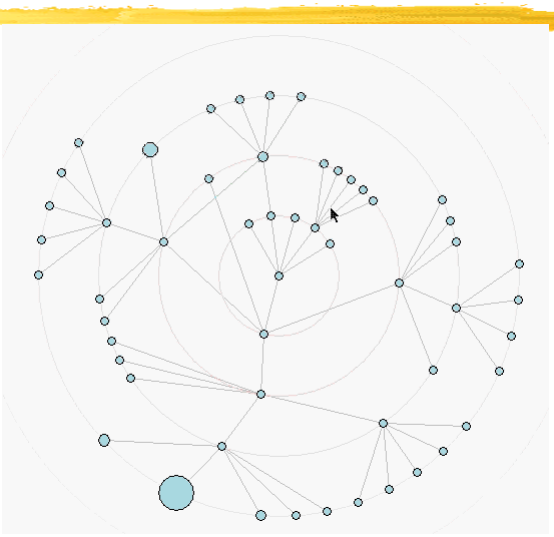
## centrality visualization

network of organizations involved in policy making



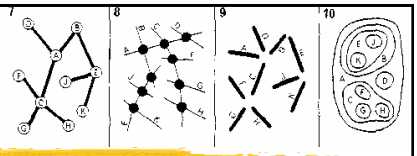

## network

**[Animated Exploration of Graphs with Radial Layout, InfoVis'01]**



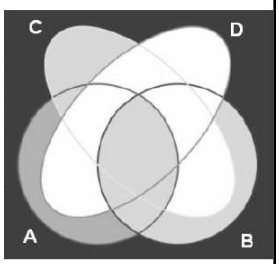
http://www.youtube.com/watch?v=OPX5iGro_lA

# Free Layout



- Order is not meaningful, only proximity is
- Force-directed methods
- Treemaps and Venn Diagrams

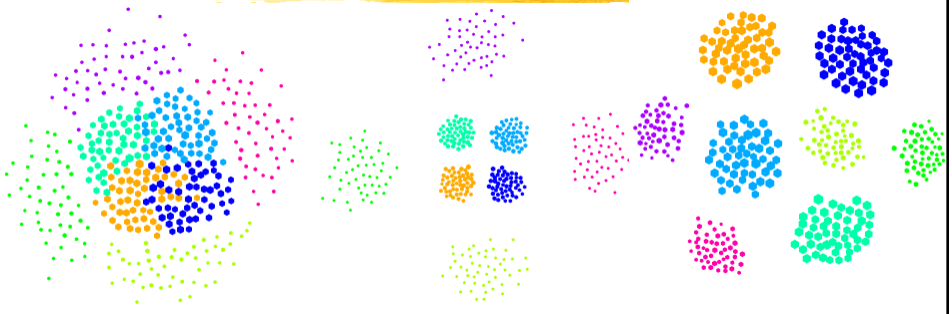
**for** $t \leftarrow 1$ **to** *ITERATIONS* **do**
  **for** $v \in V$ **do**
    $D \leftarrow \sum_{u:\{u,v\}\notin E} f_{\text{rep}}(p_u, p_v) + \sum_{u:\{u,v\}\in E} f_{\text{spring}}(p_u, p_v)$;
    $p_v \leftarrow p_v + \delta \cdot D$




# Force-directed Layout

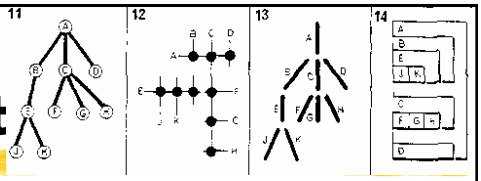
- Spring Model: $f_{rep}$ linear, $f_{att}$ linear
  - Good space filling when the network is sparse and homogeneous
- Kamada&Kawai : $f_{rep}$ linar, $f_{att}$ function of the Graph-theoretical distrance
  - Shows the shortest paths as quasi straight lines
- Fruchterman&Reingold : $f_{att}$ quadratic
  - Fills the space better for varying degrees
- Davidson&Harel : $f_{rep}$ linear, $f_{att}$ linear*degree
  - Better space filling for heterogenous graphs
- Hall/Spectral
  - Very fast but works best with grids
- LinLog : $f_{rep}$ linear, $f_{att}$ linear+log
  - Show clusters

# Example

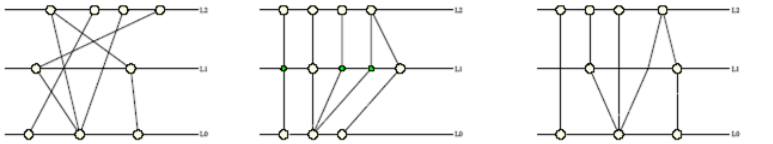


Fruchterman-Reingold model  Node-repulsion LinLog  Edge-repulsion LinLog model

# Ordered layout



- One dimension is orderable (e.g. genealogical trees)
  - The other can be partially ordrer
- Sugiyama algorithm
- Improved by C. North [A Technique for Drawing Directed Graphs.]

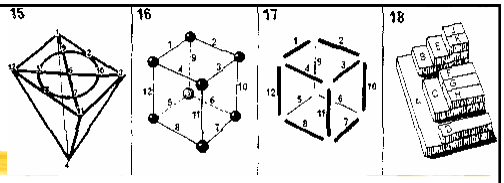


vertical ordering
layer assignment

horizontal ordering
crossing reduction

coordinate assignment
spacing and straightness

# 3D Networks

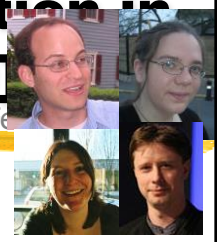


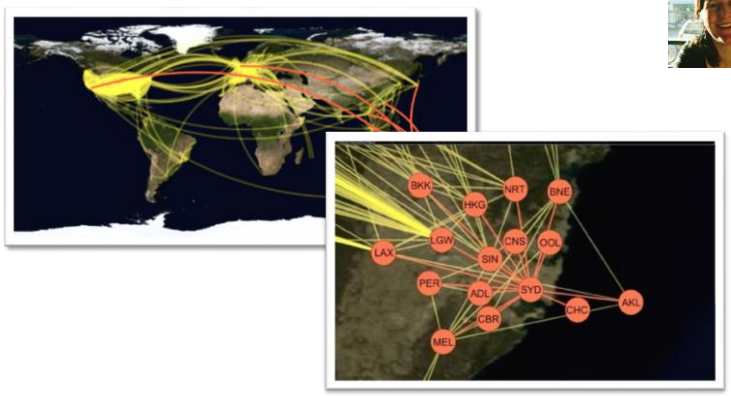
- 3D methods usually extend to 3D naturally
- Navigation is then required

# Interaction

# Topology-Aware Navigation in Large Networks [CHI 09]

T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, J.-D. Fe

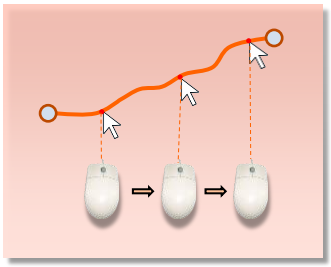




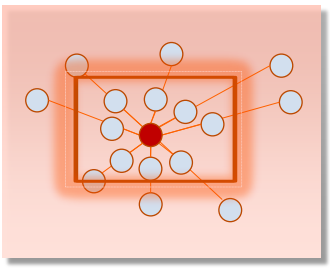

Aviz Visual Analytics Project in|situ| INRIA MICROSOFT RESEARCH INRIA JOINT CENTRE

# Further exploiting topological information

Link sliding



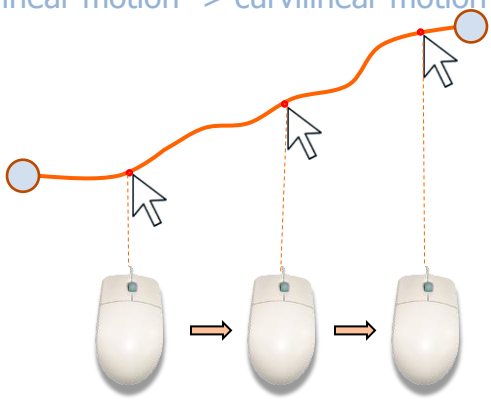
Bring'n Go



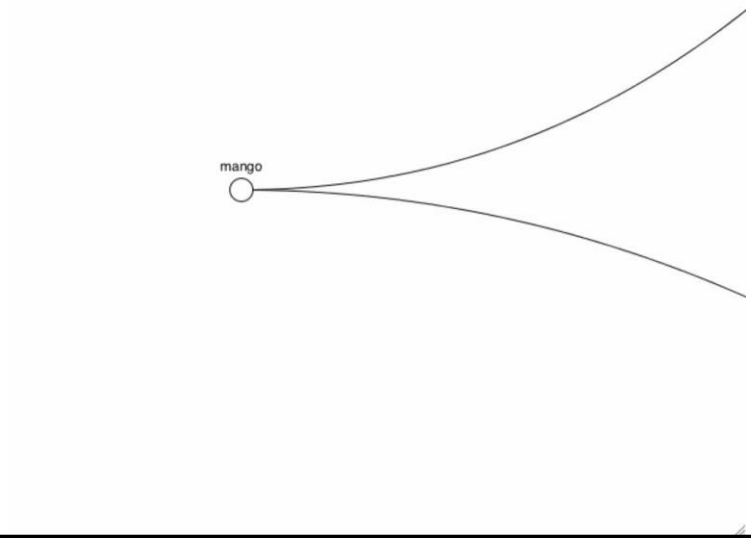
# Navigating in large networks: Link Sliding

- Motion constrained on the links

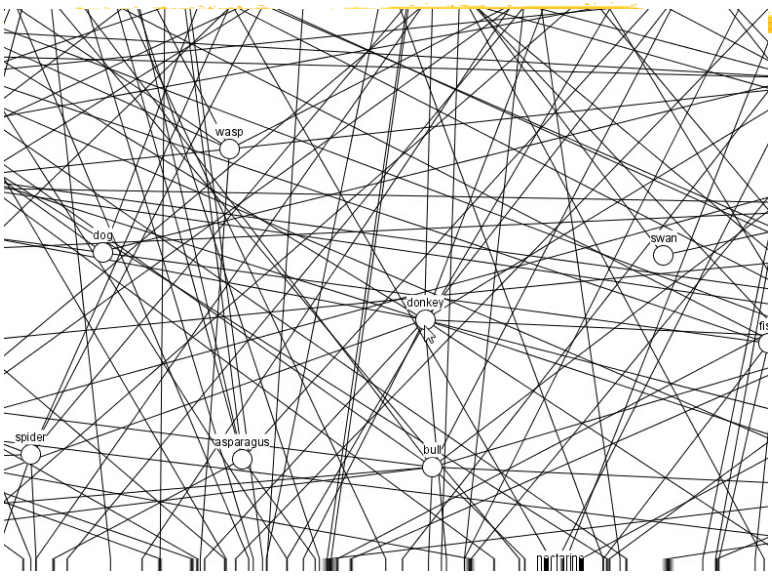
linear motion -> curvilinear motion



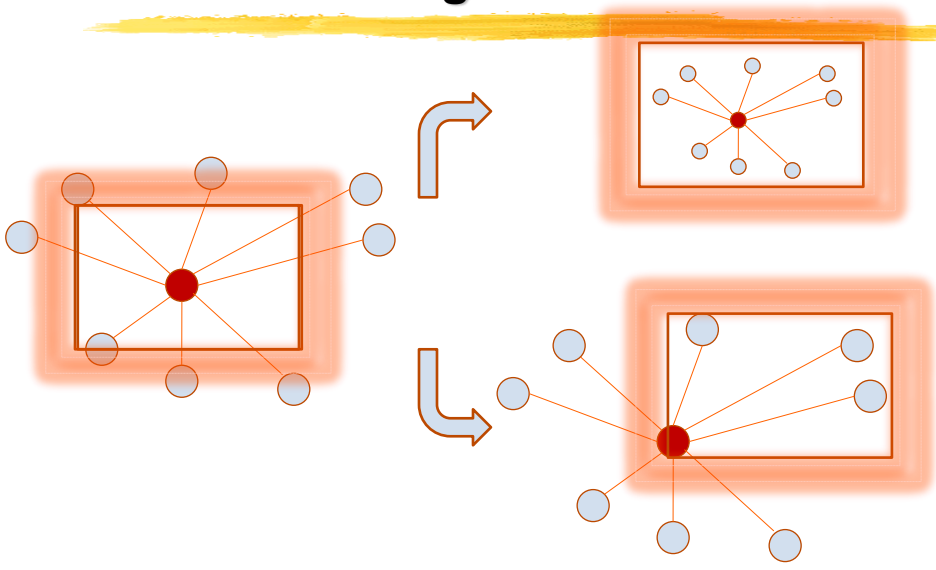
# Navigating in large networks: Link Sliding



# Navigating in large networks: Link Sliding

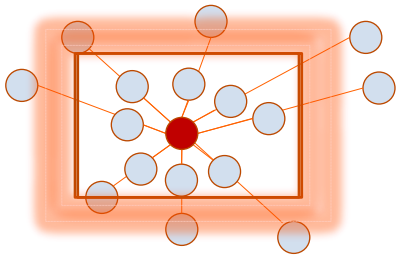



# Navigating in large networks: Bring'n Go

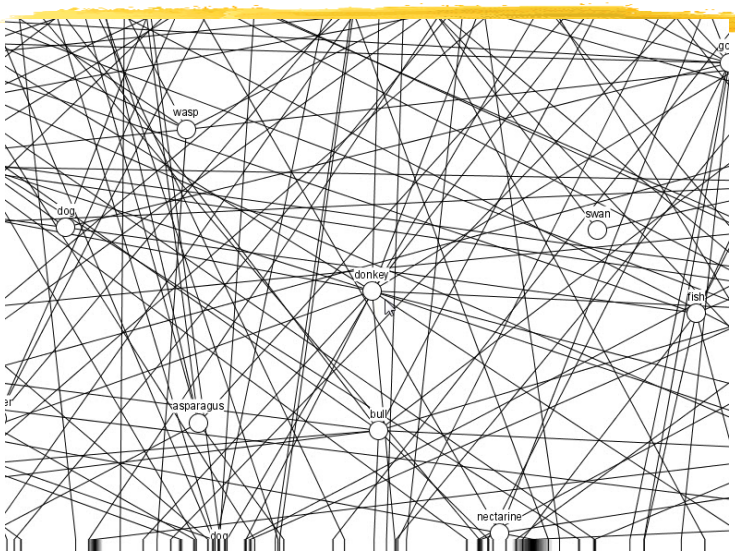


## networks
### Bring'n Go

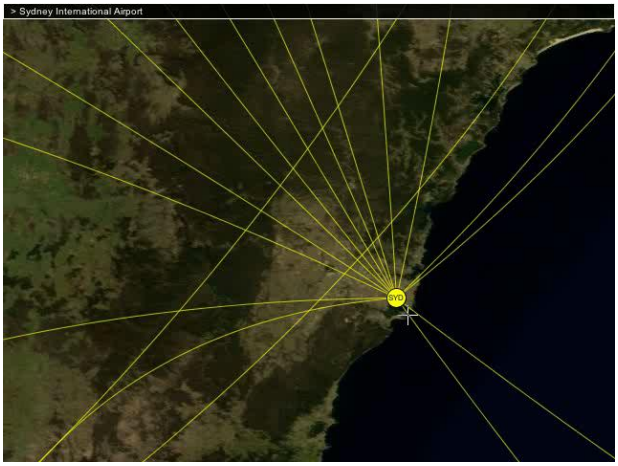
- Bring the neighbors into the view



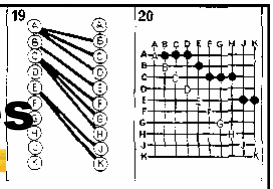
## Navigating in large networks: Bring'n Go



# Navigating in large networks: Bring'n Go




# Diagrams and Matrices




Directed network

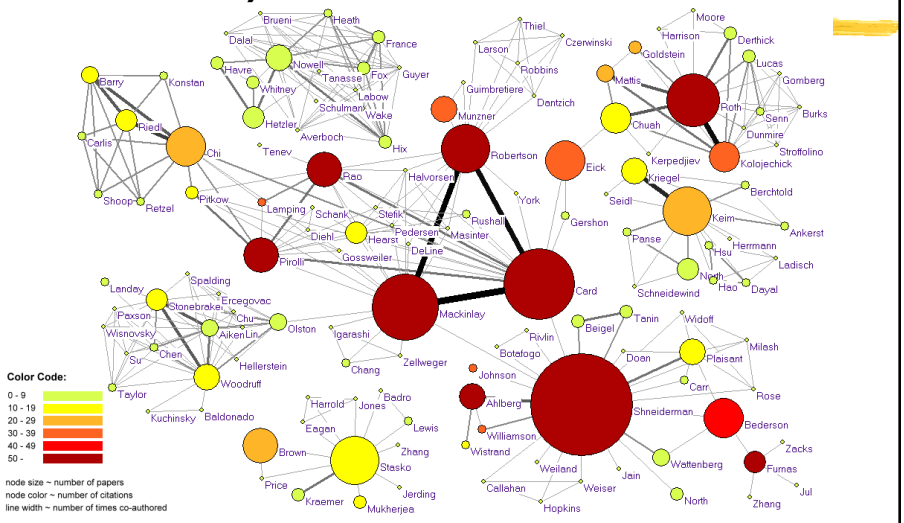

Adjacency matrix

| | A | B | C | D |
|---|---|---|---|---|
| A | 0 | 1 | 0 | 0 |
| B | 0 | 0 | 1 | 0 |
| C | 0 | 0 | 0 | 1 |
| D | 0 | 0 | 0 | 0 |

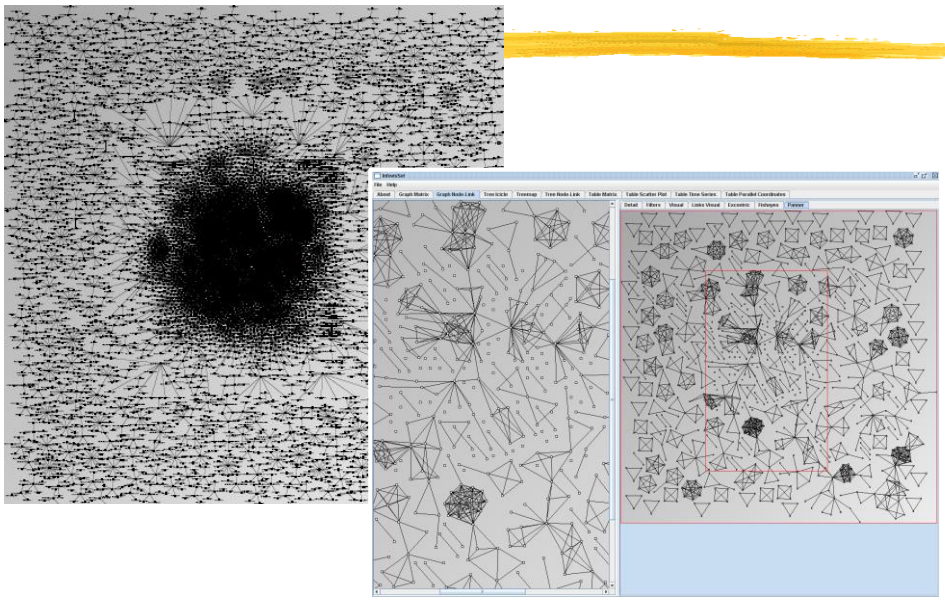
*Destination/target*

*Source*

# InfoVis Co-authoring (K. Börner et al.)

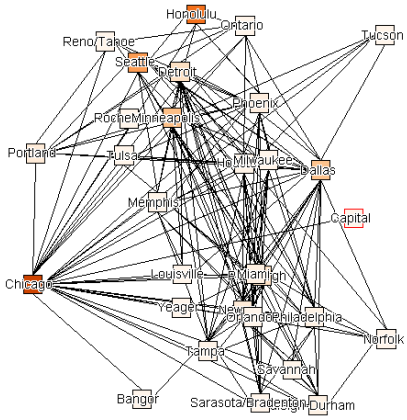


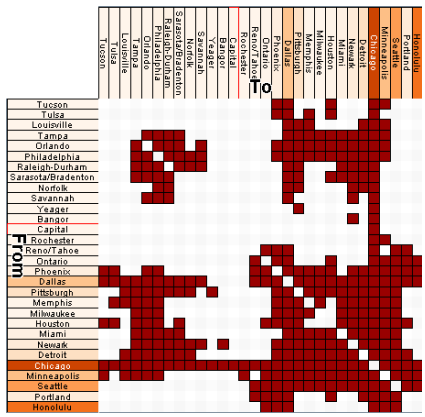

# Generally, after loading…

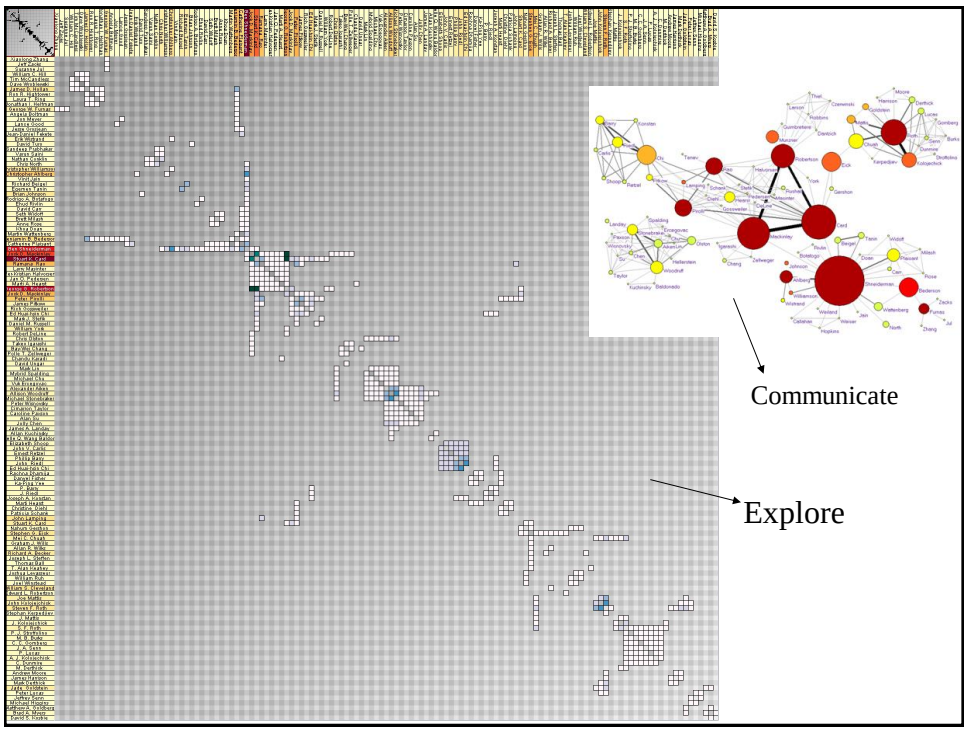


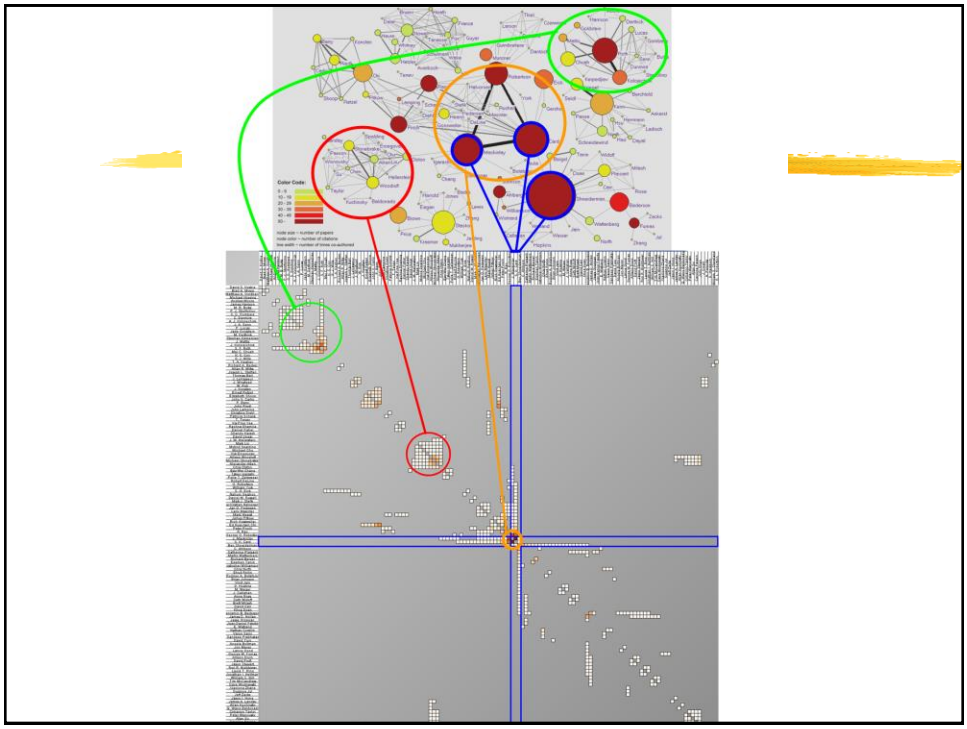
# Two classical representations

- Node-Link
- Adjacency Matrix

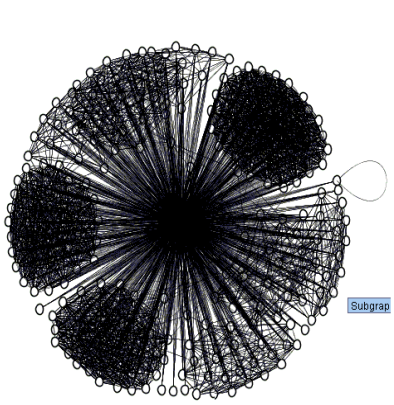


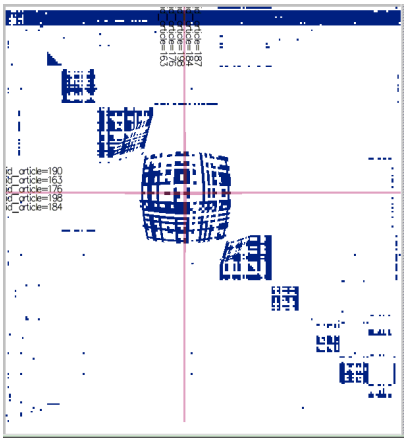



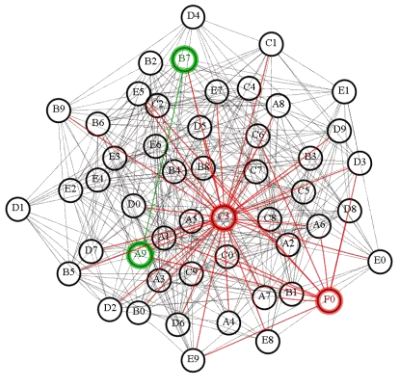


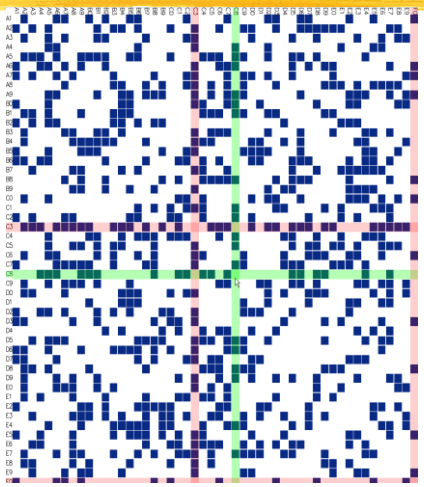
# Web Site Example



# Readability Experiment



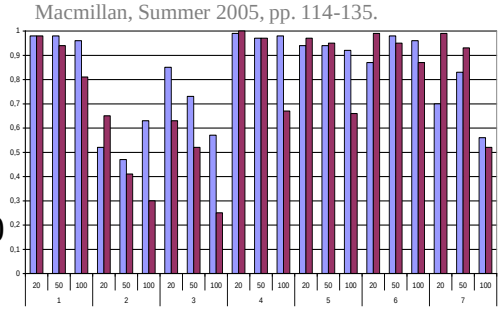


# Controlled Experiment: Node Link Diagrams vs. Adjacency Matrices

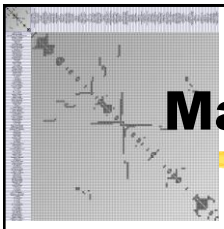
- **The Tasks:**
- Tasks related to the overview
  - Number of vertices
  - Number of arcs
- Tasks related to graph elements
  - Finding an element (a vertex, a link)
  - Finding the most connected vertex (a central actor, a pivot, a hub)
  - Finding a common neighbor
  - Finding a path
- Random graphs (3 sizes et 3 densities)
- 2 representations: Node-Link + Matrix
- **Results:**
- Node-link diagrams are preferable for small sparse graphs (20 vertices)

Matrices are more readable wrt dense graphs and medium/large graphs ( > 20 vertices) wrt the selected tasks, except path finding

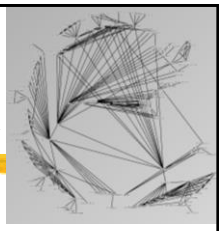
**References:**
Mohammad Ghoniem, Jean-Daniel Fekete and Philippe Castagliola *Readability of Graphs Using Node-Link and Matrix-Based Representations: Controlled Experiment and Statistical Analysis*, Information Visualization Journal, 4(2), Palgrave Macmillan, Summer 2005, pp. 114-135.




Completion time for the 7 tasks, 3 densities and 2 representations (Node-Link in blue, Matrix in red)



# Matrix vs. NodeLink



\+

- Usable without reordering
- No node overlapping
  No edge crossing
  ➔ Readable for dense graphs
- Fast navigation
- Fast manipulation
  ➔ Usable interactively
- More readable for some tasks
- Familiar
- Compact
- More readable for path following
- More effective for small graphs
- More effective for sparse graphs

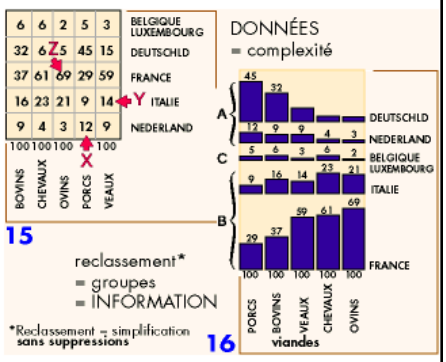
\-

- Less familiar
- Use more space
- Weak for path following tasks
- Useless without layout
- Node overlapping
  Edge crossing
  ➔ Not readable for dense graphs
- Manipulation requires layout computation

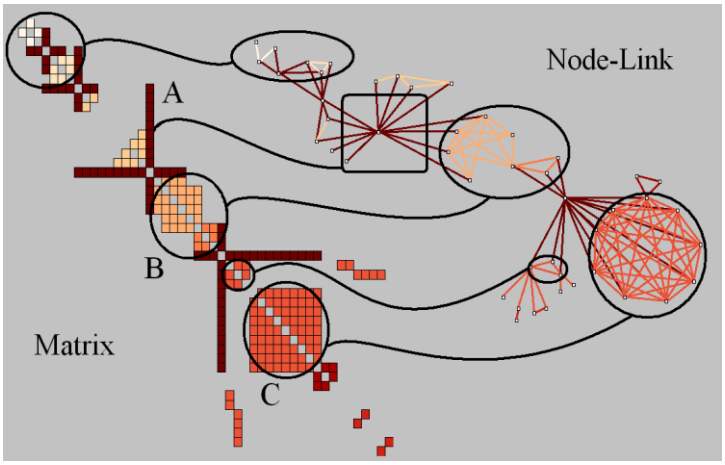
# The Reorderable Matrix

- Introduced by Bertin 67 as a representation for relational data
- Table or Network
- The value table provides details
- The reordered table provides details AND overall structure in the same representation

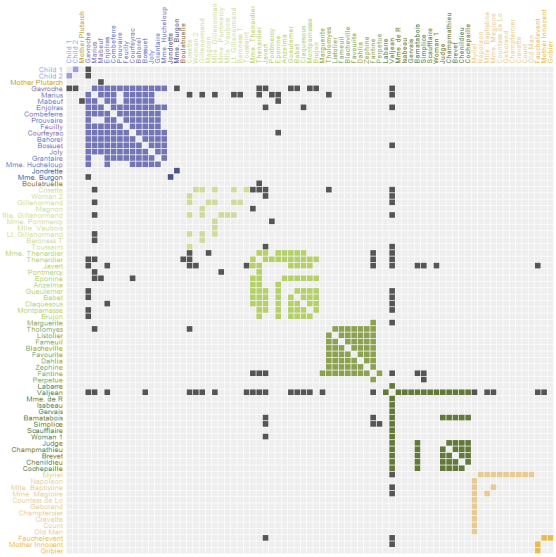
- Problems:
- how to compute a good ordering?
  - Row and column permutations
- how to assess its quality?



# Visual Patterns with Ordered Matrices



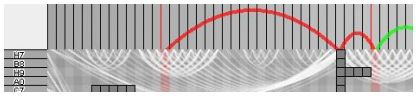

# Characters in “Les Misérables”



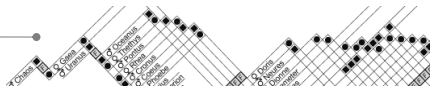
# Breakthrough in Social Network Visualization: Improving Matrices

Several representations:

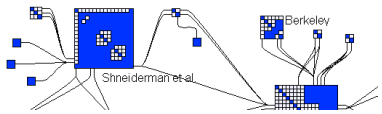
1. **Combined**
   - MatrixExplorer (Henry&Fekete InfoVis'06)
2. **Augmented**
   - MatLink (Henry&Fekete Interact'07, **Best Paper**)



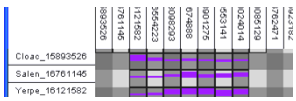
   - GeneaQuilts (Bezerianos et al. InfoVis'10)



3. **Hybrid**
   - NodeTrix (Henry et al. InfoVis'07)
   - CoCoNutTrix (Isenberg et al. CG&A'09)

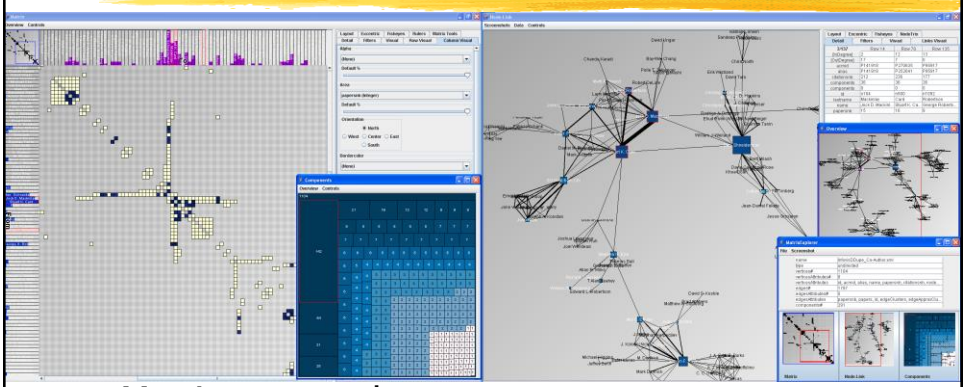



4. **Multiscale**
   - ZAME (Elmqvist et al. PacificVis'08)



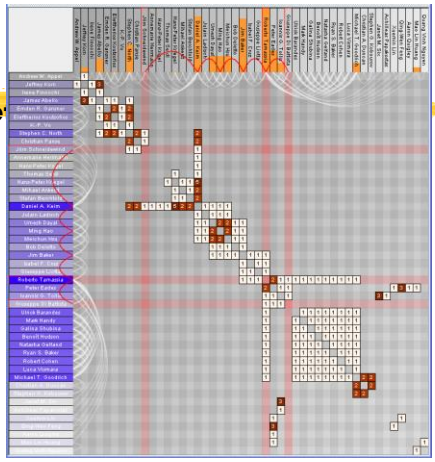

# MatrixExplorer [Henry&Fekete06]

## Combined representation



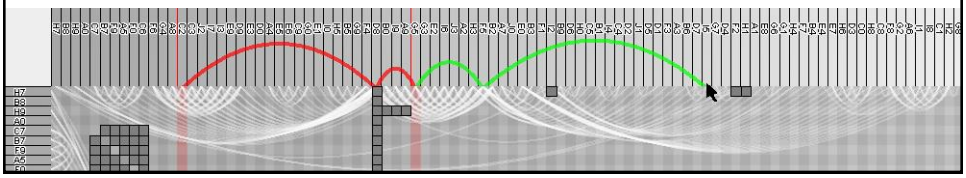
- Matrices to explore
- Node-Link diagrams to present findings

# MatLink[Henry&Feke



Augmented representation

- Augmenting matrices with interactive links
- Solving the path-related tasks problem for matrices



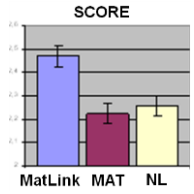
# MatLink significantly improves matrices

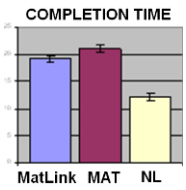
- Controlled experiment
  - 3 vis. x 6 datasets x 5 tasks

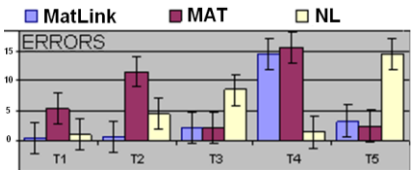
  Matrix , Node-Link, MatLink

  **Data**: From almost-trees
  To complete-graphs
  Including small-world networks

  **Tasks**:
  1. CommonNeighbour,
  2. ShortestPath,
  3. MostConnected,
  4. ArticulationPoint,
  5. LargestClique

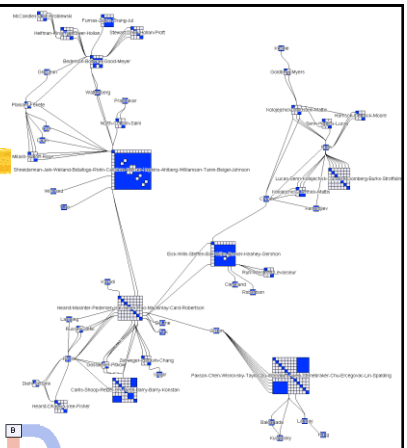


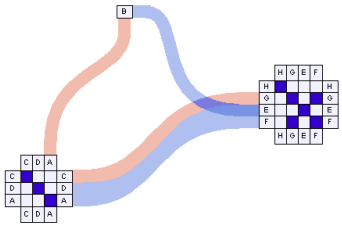



# NodeTrix[Henry et al.07]

**Hybrid representation**

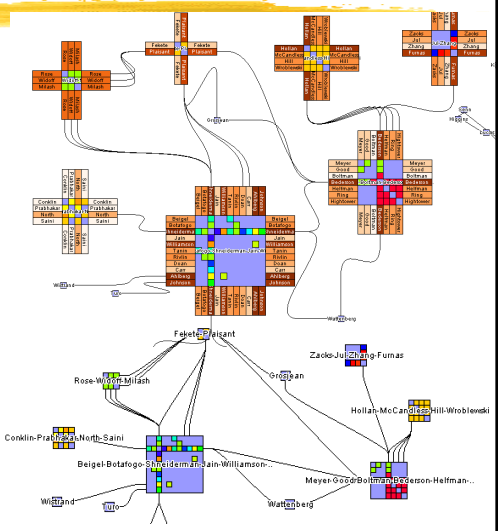


- Designed for small-world networks
  - Globally sparse
  - Locally dense
- Visualizing dense sub-graphs as matrices
- Interact to create, edit and remove the matrices



# NodeTrix: the NetVis Nirvana?

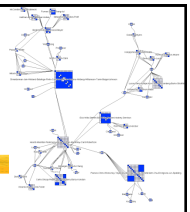
✓Can you see every node?
✓Can you count each node's degree?
✓Can follow every link from its source to its destination?
✓Can you identify clusters and outliers?

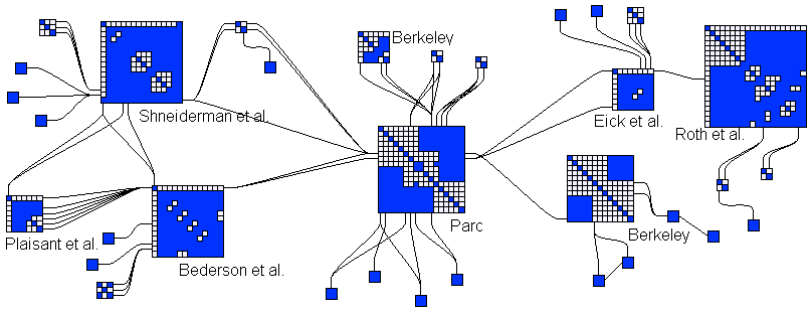



- Node Labels
- Link Labels (excentric labels?!)
- … even cluster labels
- Node Attributes
- Link Attributes
- … even clusters attributes
- Directed Graph (links width?!)

… But… beware the graphics overload!

# Visual Patterns





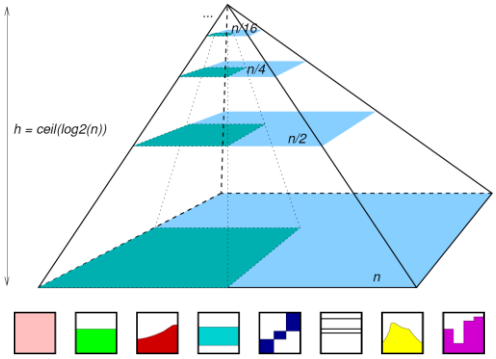

Infovis Coauthorship (133 actors)

# ZAME: Interactive Large-Scale Graph Visualization (Elmqvist et al. 08)

Visualize very large networks:

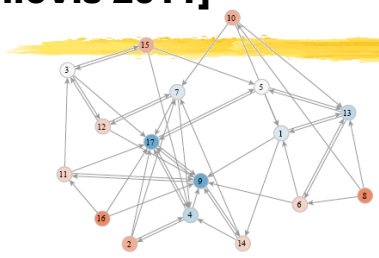
- Larger than $10^7$ vertices and edges

- Reorder
- Create a pyramid
- Aggregate attributes
- Visualize using enhanced glyphs

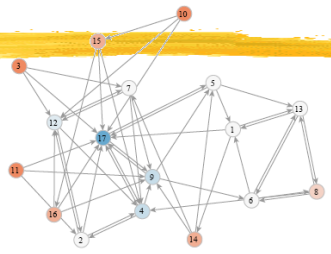


# Longitudinal Social Networks

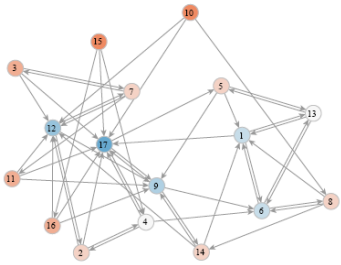
**Ulrik Brandes and Bobo Nick, Univ. Konstanz**
**[InfoVis 2011]**



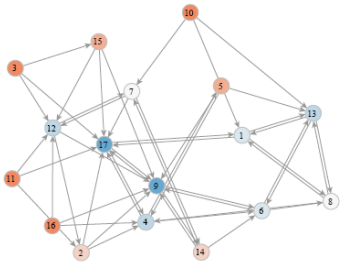
(a) wave 2



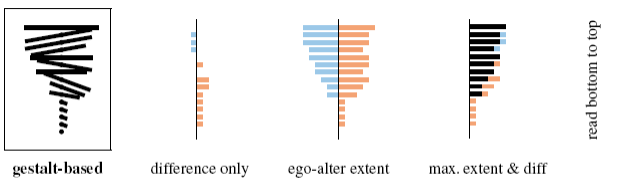
(b) wave 6



(c) wave 10



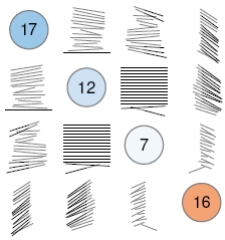
(d) wave 14




data: (1,1),(1,2),(1,2),(1,2),(1,2),(3,4),(3,5),(4,6),(5,5),(5,6),(6,6),(6,5),(7,6),(7,6),(7,7)

While 2 befriends popular 17 only late, 16 17 seems to stand no chance at that. In fact, 16 is unsuccessful in entering the tightly knit group of 17, 12, and 7.

The relationships between 7 14, 3 15, or 10 15 are rather more complicated. The latter is no surprise as 10 is fairly unpopular with many.




Gestaltmatrix of Newcomb Fraternity Data, showing the evolution using all 4080 data points (rankings) from 15 waves. Labels on the diagonal are numbered according to Nordlie [42] and colored according to the standardized overall popularity of the corresponding actor. The sorting of actors is according to a spectral approach.

# Conclusion

- Visualization of Networks has been greatly improved in the last years
- Novel representations are denser and more expressive
  - Though they require training
- Huge and dense networks can be visualized
  - Relations between clients, suppliers, employees
  - Aggregated over long periods of time
- Needs more research to understand how reordering leads to better understanding

## Challenges

- Exploring Very Large / Very Dense Networks
- Linking Exploration and Modeling
- Reordering Methods for Matrices
  - What is a good order and why?
  - Orderings for directed graphs
  - Multi-Scale Ordering (top-down methods)

## Free Software

- Java
  - Prefuse, Cytoscape, JUNG, Guess, Gephi
- C/C++/Python
  - GraphViz, igraph, NetworkX
  - Tulip
- Web
  - Flare (flash)
  - **D3,** Protovis, Javascript InfoVis Toolkit (JIT)