Data Visualization

Fall 2016

Information Visualization

- Upon now, we dealt with scientific visualization (scivis)
  - Scivis includes visualization of physical simulations, engineering, medical imaging, Earth sciences, etc.
  - Typical datasets consist of samples of continuous quantities over compact domain
- Now, we will focus on more abstract data types
  - Typical datasets: generic graphs and trees, database tables, text, etc.
  - Information visualization (infovis) studies the visual representation of such data

Information Visualization

- Infovis is the fastest groving branch of the visualization
- Main goal is to assist users in understanding all the abstract data, i.e. visualize abstract quantities and relations in order to get insight in the data with no physical representation
- Differences:
  - Scivis – physical data with inherent spatial placement → mental and physical images overlap → considerably simplifies visualization
  - Infovis – information has no innate shape and color and its visualization has purely abstract character

Information Visualization

- Three main elements: representation, presentation, and interaction
- Infovis has potentially larger target audience with limited mathematical or engineering background than scivis
- Infovis covers areas such as:
  - Visual reasoning, visual data modeling, visual programming, visual information retrieval and browsing, visualization of program execution, visual languages, visual interface design, and spatial reasoning

Information Visualization

- General rules for design of infovis applications:
  - Follow the conventions accepted by that field
  - Integrate with other tools-of-the-trade of the field
- In some taxonomies (Spence), there also exists class of geovisualization (geovis) applications which address a field between the two

Information Visualization

- Data domain:
  - Datasets often do not contain spatial information (sample points)
  - No cells with interpolation function or cell notion serves a different purpose
  - Actual spatial layout is of little if any relevance for the content

Information Visualization

- Attribute data types in infovis:
    - Data attributes are of more types than numerical values and go beyond the semantic of numerical values
    - A different storage strategy (size of a single attribute is variable)

Information Visualization

Data type			Attribute domain	Operations	Examples
Nominal (categorical)	Qualitative (no addition and multiplication)	Categorical*	Unordered set	Comparison ($=, \neq$)	Text, references, syntax elements
Ordinal			Ordered set	Ordering ($=, \neq, <, >$)	Ratings (e.g., bad, average, good)
Discrete	Quantitative (allow interpolation)		Integers (Z, N)	Integer arithmetic	Lines of code
Continuous		-	Reals (R)	Real arithmetic	Code metrics

Notes:

* A data item belongs to a category rather than the value of quantity

Information Visualization

- Another classification of attribute data types:
  - Linear
  - Planar
  - Volumetric
  - Temporal
  - Multidimensional
  - Tree
  - Network
  - Workspace

Spatial aspect

Relational aspect

Information Visualization

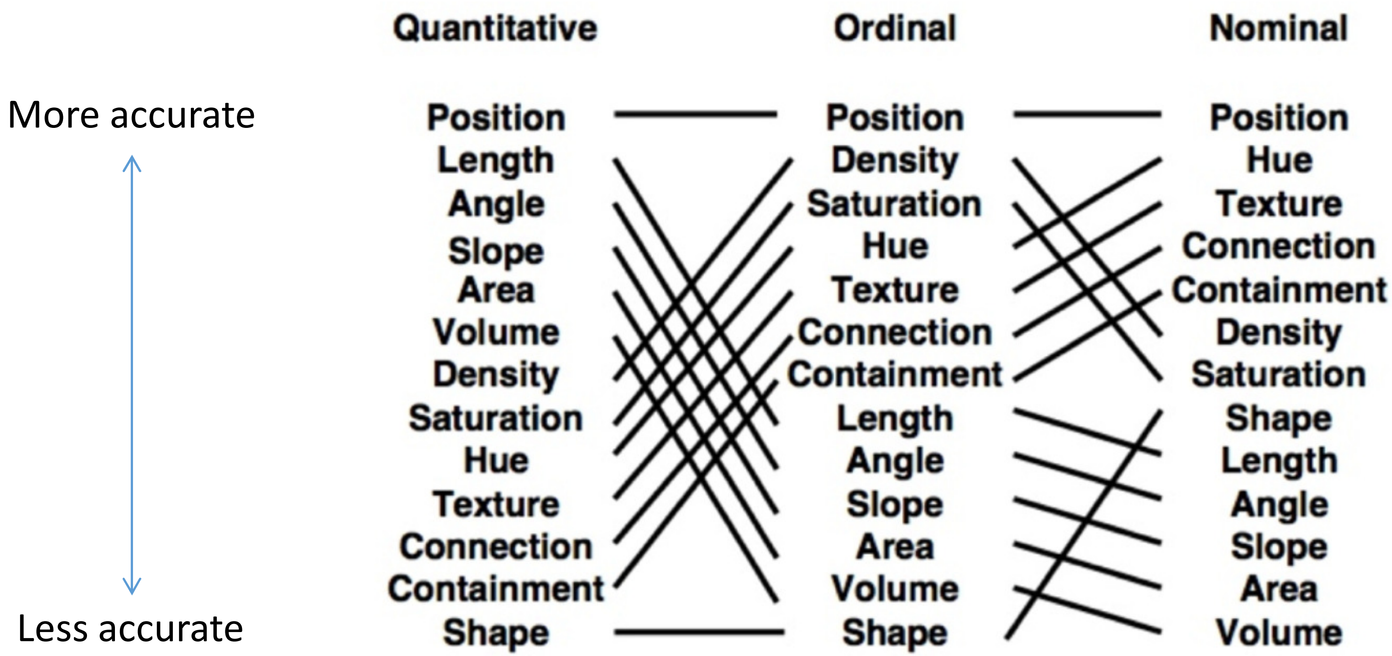
- Together with eight data types, seven interaction functions infovis application may provide:
  - Overview, zoom, filter, details on demand, relate, history, and extract
- These functions may be related to main steps of visualization pipeline:
  - Filtering, mapping, and rendering
- Data types and interaction types create a matrix of possibilities within which a infovis application may locate its functionality

Information Visualization

- Comparison of datasets notion in scivis and infovis

	Scivis	Infovis
Data domain	Spatial R^n	Abstract, nonspatial
Attribute types	Numeric R^m	Any data types
Data points	Samples of attributes over domain	Tuples of attributes without spatial location
Cells	Support interpolation	Describe relations
Interpolation	Piecewise continuous	Can be nonexistent

Information Visualization



Mackinlay, 1986

Information Visualization

- Infovis datasets are quite similar to the model used in relational databases or entity-relationship graphs
- Visualization methods:
  - Database tables, trees, graphs, and text

Table Visualization

- Table – simplest infovis data; two-dimensional array of rows (records) and columns (attributes)
- Improvements supporting readability:
    - Sorting
    - Filling background of cells using alternate colors
    - Bar graph as a cell background
    - Small glyphs or icons showing trends
    - Sparklines

Tasks completed by team members

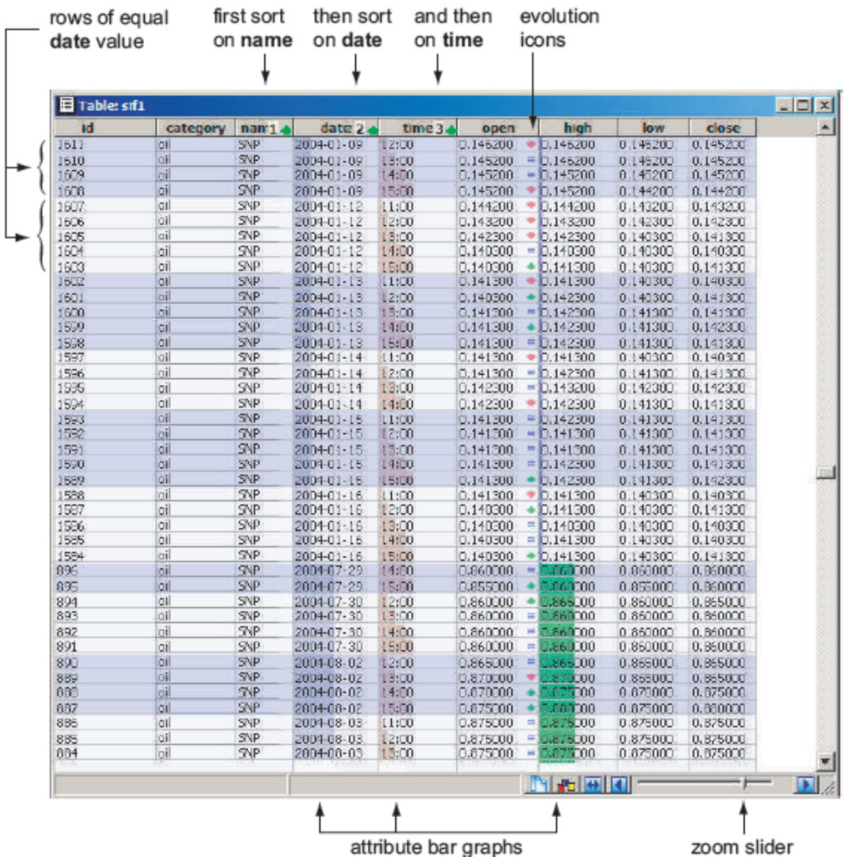
(last 26 weeks, YoY change shown in %s)

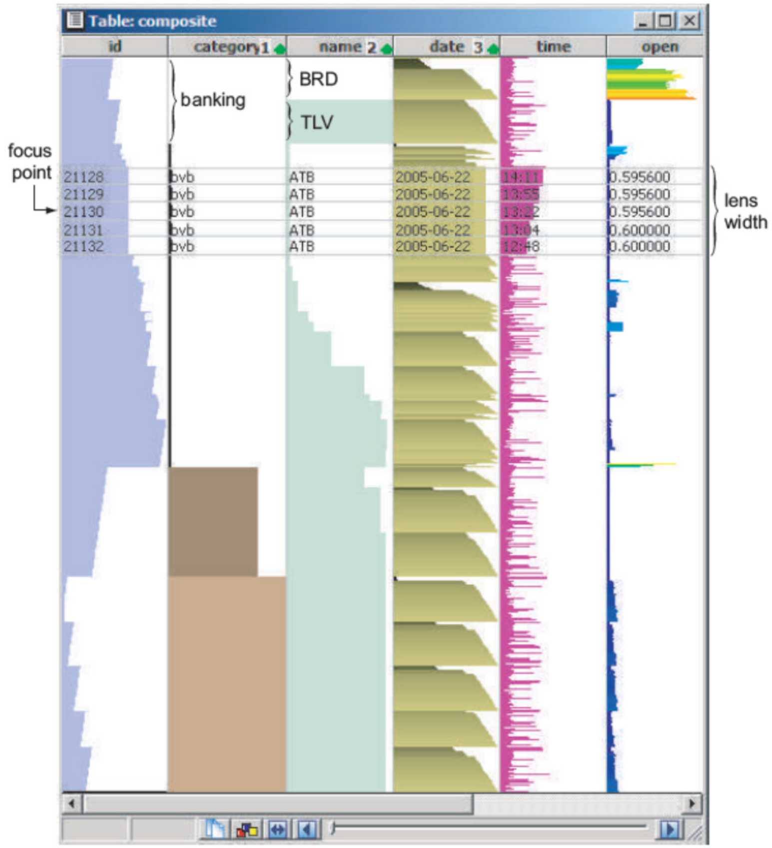
Team Member	Total Tasks Completed	w1	w2	w3	w25	w26
Julie	46%	13	15	19	11	19
John	45%	11	18	11	14	16
Jabba the hut	-20%	15	14	14	19	12
Johnson	6%	18	17	14	12	19
Jeremy	43%	14	20	10	12	20
Josh	-33%	15	12	19	11	10

Table Visualization

- Sampling issue
  - Text based visualization has fairly limited scalability
    - Zooming out the table visualization
    - We may drop displaying too small text and only show bar graphs
    - Use so called dense pixel displays or space filling displays

Table Visualization





Relation Visualization

- Frequenty encountered visualizations of relational datasets:
    - Trees, graphs, and Venn-Euler diagrams

Tree Visualization

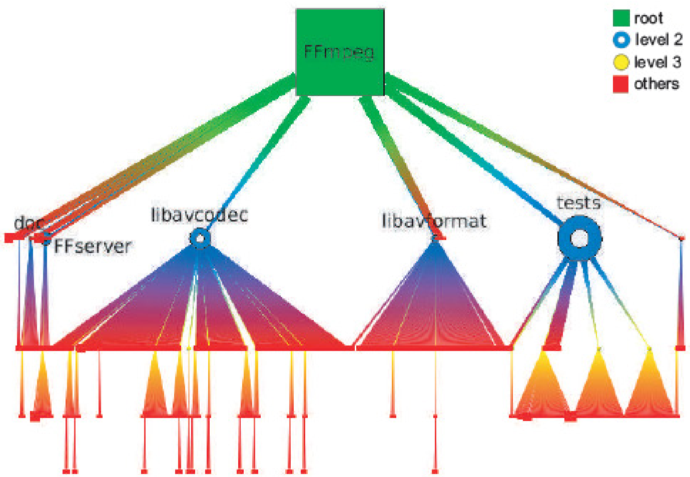
- Trees are a particular type of relational data
- $T = (N, E)$, where N = $\{n_i\}$ is set of nodes (vertices) connected by edges from set of edges $E = \{e_i\}$ where each edge e_i is represented as a pair (n_j(parent), n_k(child)) of nodes
- Properties of a tree:
  - There is a unique path between any two nodes in the tree
  - Subsequently, there are no loops
  - Parent may have any number of children; child can have only one parent; leaves have no children
  - Root – single node with no parents
  - Depth – longest path in the tree

Tree Visualization

- **Node-link visualization** (ball and stick) with two degrees of freedom:
  - Position of the glyphs (layout)
  - The appearance of the glyph
- Layout requirements:
  - No or minimal overlapping of nodes and edges
  - Aspect ratio not far from unity
  - Avoid long or unnecessarily bent edges

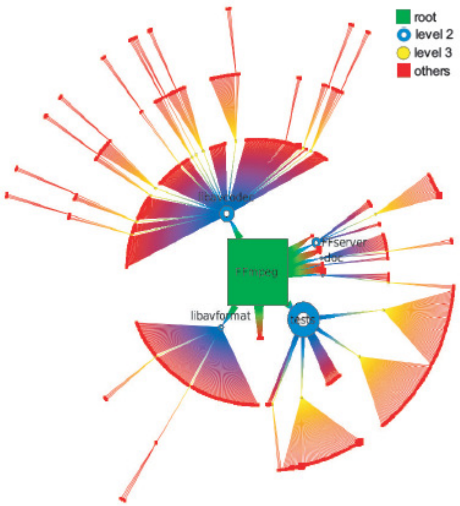
Tree Visualization

- **Rooted tree layout**:
  - All children nodes of the same parent have the same y-coordinate
  - X-axis is used to reflect certain ordering



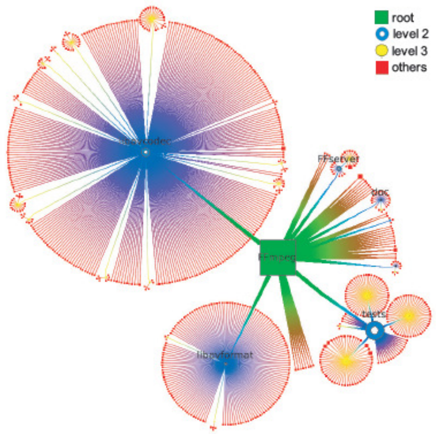
Tree Visualization

- **Radial tree layout**:
  - Use polar coordinate system
  - Always has 1:1 aspect ratio but problems with space allocation



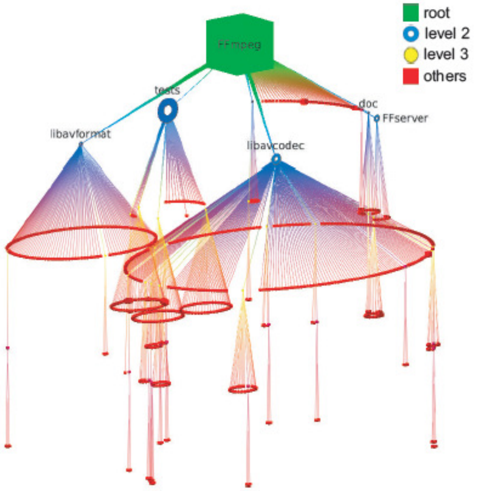
Tree Visualization

- **Bubble tree layout**:
  - Edges have now considerably different lenghts
  - This makes the visual size of the subrees reflect their number of children



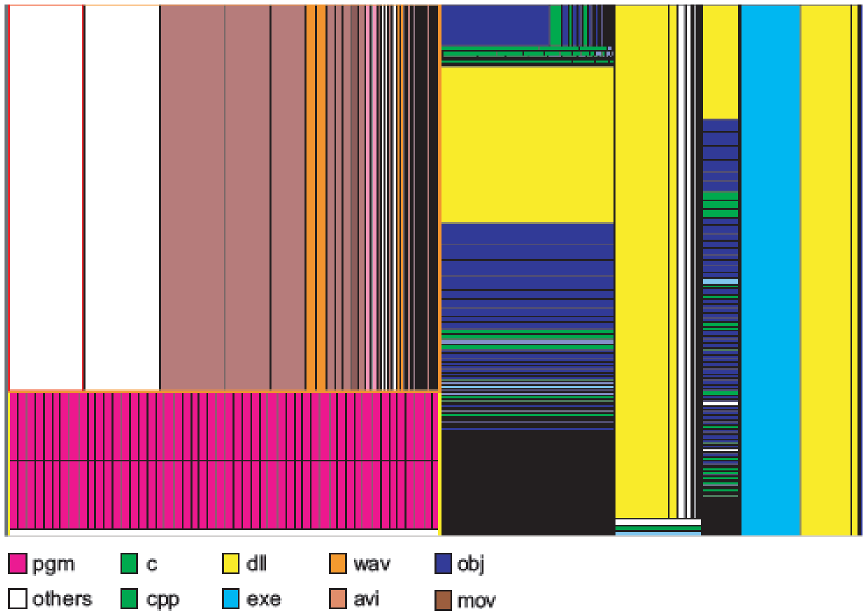
Tree Visualization

- **Cone tree layout**:
  - Arranged in 3D, may be more compact than other layouts
  - Problems: occlusions, chance of “getting lost“ in 3D space



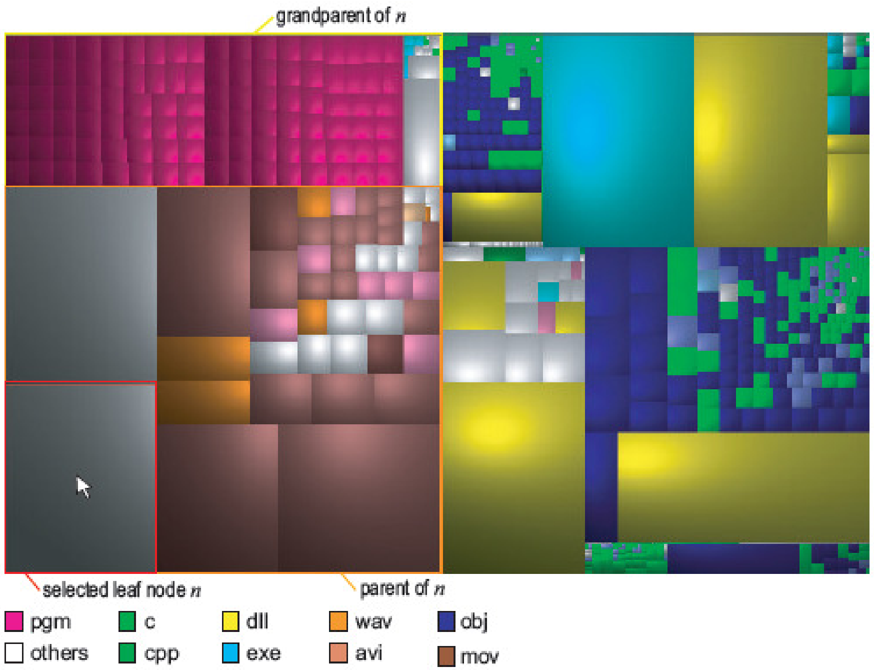
Tree Visualization

- **Tree Maps**
  - Slice and dice layout



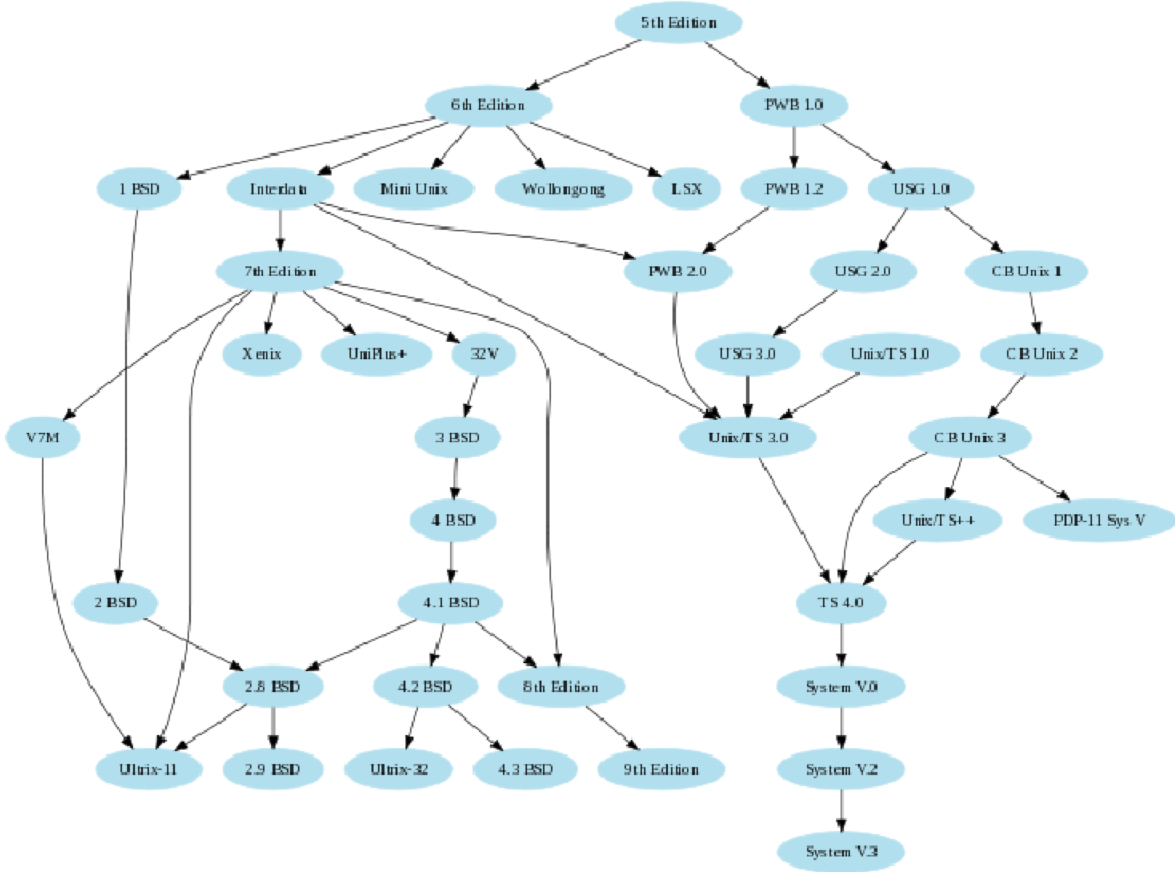
Tree Visualization

- **Tree Maps**
  - Squarified layout



Tree Visualization

- **Tree Maps**
  - Hierarchical layout



- **Force-directed layout**

$$\begin{aligned}\mathbf{F}_a(n_i,n_j) &= \frac{\|p_i-p_j\|}{k}(p_j-p_i),\\ \mathbf{F}_r(n_i,n_j) &= -\frac{k^2}{\|p_i-p_j\|^2}(p_j-p_i)\end{aligned}$$

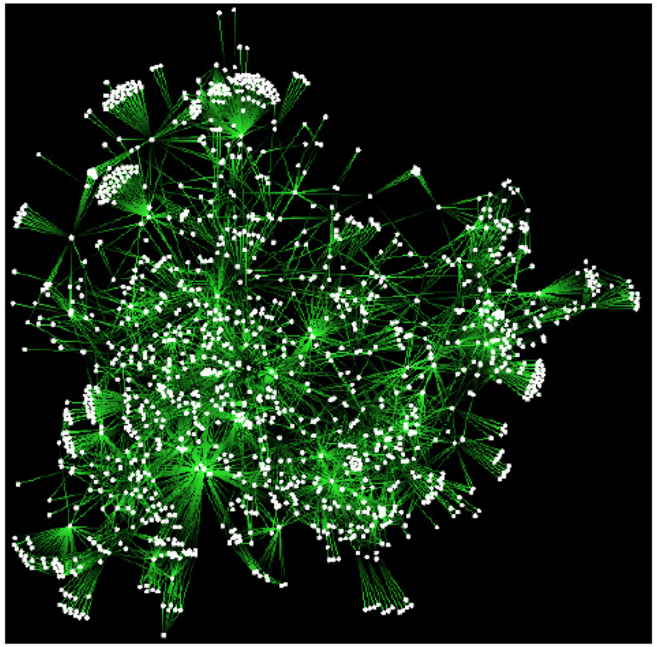
$$\begin{aligned}\mathbf{F}_a(n_i,n_j) &= k\,\log\|p_i-p_j\|\frac{p_j-p_i}{\|p_j-p_i\|}\\ \mathbf{F}_r(n_i,n_j) &= -\frac{k}{\|p_i-p_j\|^3}(p_j-p_i).\end{aligned}$$

The energy function os not monotonic
Can get stuck in local minima
No clear ordering – where to start reading the plot

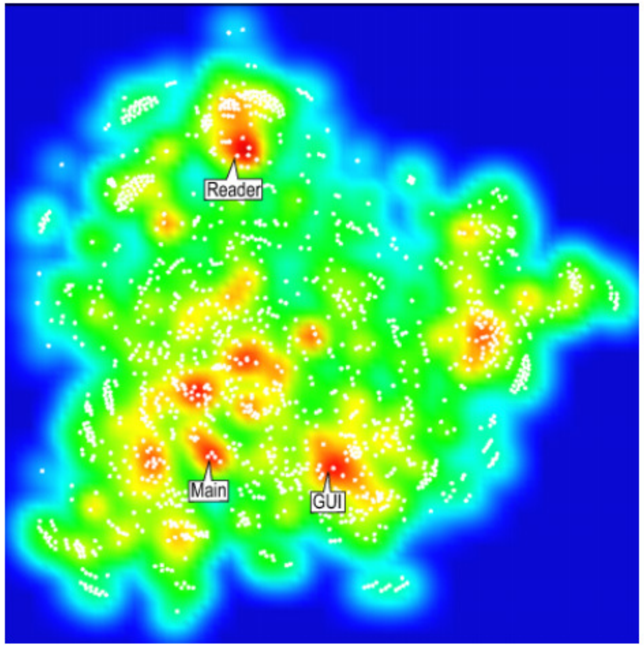
```
for(int i=0;i<N;i++)                          //Initialize layout
  p_i = random position;
float t = t_0;                                //Initial maximal allowed move

for (int i=1;i<ITER;i++)                      //Do the layout
{
  for(int i=0;i<N;i++)                        //Compute repulsive forces F_r
  {
    f_i = 0;
    for (int j=0;j<N;j++)
      if (j!=i)
        f_i +=F_r(i,j);
  }

  for (int edge=0;edge<E;edge++)              //Compute attractive forces F_a
  {
    int i = edge.first;                       //Get first node of edge
    int j = edge.second;                      //Get second node of edge
    f_f -= F_a(i,j);
    f_e += F_a(i,j);
  }

  for(int i=0;i<N;i++)                        //Move the nodes by applying forces
  {
    p_i+ = f_i/||f_i|| min (δ, t||f_i||);
  }

  t -= tΔt;                                   //Reduce maximal allowed move t
}
```

Tree Visualization

- **Graph Splatting**
  - Convolve nodes (optionaly edges) with Gaussian filter



Discrete dataset



Continuous dataset

Matrix Visualization

- **(Directed/undirected) Adjacency Matrix**
  - Order of rows and columns highly impact the visualization (spotting clusters etc.)