BDVA 2016 Workshop
Visual Analytics for Relational Data

Dr. Quang Vinh Nguyen
Q.Nguyen@WesternSydney.edu.au
Relational Data

• Graphs (as a general term for relational data)
  – Software documentation, e.g. coding
  – WWW

• Networks (as a variation of graph)
  – Network of Computer
  – Social Network, e.g. Facebook

• Hierarchical structures / Trees (as a special case of graph)
  – Organisation chart
  – Family tree
Graph Visualisation

• Why Graph Visualisation?
  – One of the most important sections.
  – Nature of information
  – Aesthetical display data elements and relations
  – Quick and effective navigation/interaction
  – Wide range of applications:

• Current Issues
  – Size of the graph, i.e. large graphs
    • Screen size
    • Viewing limitations
    • Computational performance
  – Cognitive science
    • Need more usability studies
Graph

• Graphs are abstract structures that are used to model relational information

• Graph $G = (V, E)$
  – $V$: set of vertices (objects)
  – $E$: set of edges connecting vertices (relationships)

• Graph Visualisation: automatic construction of geometric representations of graphs in 2D/3D

• Graph edges and vertices may have values (weights) on them as well as other attributed property

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
Graph - Example

\[ G = (V, E) \]

\[ V = \{A, B, C, D, E\} \]

\[ E = \{E_1, E_2, E_3, E_4, E_5, E_6\} \]
Graph – Example (cont.)

\[ G = (V, E) \]
\[ V = \{A, B, C, D, E\} \]
\[ E = \{E_1, E_2, E_3, E_4, E_5, E_6\} \]

The output is a drawing of the graph that it is aesthetically nice → easy to understand and easy to perceive
Bad Graph Visualisations – Example 1
Bad Graph Visualisations – Example 2
Bad Graph Visualisations – Example 3
Good Graph Visualisations

Tom Sawyer Software

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
Good Graph Visualisations (cont.)

Tom Sawyer Software
Classification of Graph Visualisation

• Trees
• Acyclic graphs, DAGs
• General graphs
• Planar graphs
• Networks
TREE VISUALISATIONS
Trees

- Subcase of general graph
- No cycles
- Typically directed edges
- Special designated root vertex

Can You Give an Example of a Tree Structure?

- Object-oriented software classes
- Family histories, ancestries
- File/directory systems on computers
- Organization charts
- Animal kingdom: Phylum,…, genus,…
Connection Approach

• Presents graph/tree structures using node-link diagrams.
  – Traditional way to represent a structure
  – **Advantage:** directly see the relationships.
  – **Limitation:** not efficient in term of utilising display space.

• Sample techniques:
  – Classical hierarchical view [Reingold & Tilford 81]
  – Hyperbolic browser [Lamping & Rao 95]
  – Radial view [Eades 92]
  – Cone-tree [Robertson et al 91]
  – SpaceTree [Plaisant, Grosjean & Bederson 2002]
Enclosure Approach

- Uses enclosure to represent the hierarchical structures
  - Good for showing node attributes
  - **Advantage:** efficient in term of utilising display space.
  - **Limitation:** not show directly the relational structures of information.

- Sample techniques
  - Tree-maps [Johnson & Shneiderman 91],
  - Cushion tree-maps [Wijk & Wetering 99],
  - Squarified tree-maps [Bruls et al. 00],
  - Voroinoi Treemap
  - Sunburst [Stasko & Zhang 2000]
Enclosure+Connection Approach

• Uses polygonal area division approach to define the layout of sub-trees.
• Uses node-link diagrams to show the relationship among hierarchical data.
• Objectives:
  – Utilise the available space to display more information.
  – Retain the clarity of relationship of data structures.
• Typical techniques:
  – Space-Optimised Tree [Nguyen & Huang 2003]
  – EncCon Tree [Nguyen & Huang 2005]
Visualisation Design Principles

• Make labels readable
• Maximize number of levels opened
• Decompose tree animation
• Use landmarks
• Use overview and dynamic filtering
SpaceTree
[Plaisant, Grosjean, and Bederson 2002]

- Hide unfocused sections, – show in simplified forms
- Among the best interaction techniques for Tree
- Elegant Design
- More suitable for large tree
Variations – DOITree

[Card and Nation 2002], [Heer and Card 2004], [Nguyen, Simoff, and Huang 2014]

- Representing trees in a classical and universally accepted way
- Providing multiple-foci views with focus+context interaction
- Utilizing display space
- Providing smooth fade-in and fade-out animations among transitions
- Integrate visual cues

Demo
Radial View

[Eades 1992]

• Based on an algorithm described in (Eades 1992).
• Recursively positions children of a sub-tree into circular wedges.
  – The angles of these wedges are proportional to the number of leaves of the sub-tree.
  – Avoid an angle assigned to a node being too large, radial view layout forces all wedges remain convex.
• The algorithm is simple, predictable and it behaves well in general
• Variation: MoireGraphs (Jankun-Kelly & Ma 2003)
Hyperbolic Browser / Star Tree

[Lamping and Rao 1995]

• Use Hyperbolic geometry to construct the visualisation.
  – 3D Hyperbolic Browser [Munzner 1997]

• Pros
  – Natural magnification (fisheye) in centre
  – Layout depends only on 2-3 generations from current node
  – Smooth animation and navigation for change in focus
  – Don’t draw objects when far enough from root (simplify rendering)

• Cons
  – Watching the view can be disorienting
  – When a node is moved, its children don’t keep their relative orientation to it as in Euclidean plane, they rotate
  – Not as symmetric and regular as Euclidean techniques, two important attributes in aesthetics

Demo
Tree-Maps
[Johnson & Shneiderman 1991]

• Represents hierarchical structures using enclosure for the partition of the display space to visualise the structural information.

• Standard Algorithm:
  – draw a node itself within its rectangular area with its display properties.
  – the bound of the root is the entirely rectangular display area.
  – then the node sets a new bound within its rectangular bound and drawing properties for all its children.

• Variations
  – Cushion tree-maps [Wijk & Wetering 1999]
  – Squarified tree-maps [Bruls et al. 2000]
  – Voronoi Treemaps [Balzer and Deussen 2005]
Devide & Conquer Treemaps

[Jie, Nguyen, Simoff and Huang 2013, 2015]

• Polygonal D&C Algorithm:
  – Use enclosure to partition the hierarchy, but in divide and conquer manner.
  – Divide into 2 similar groups
  – Apply the partition process for each group
    • Partition can be made from a vertex or a side
  – Repeat the division until reach to the “good” partition granularity
  – The polygonal algorithm produces mostly triangles

• Can be applied virtually to any shapes
Devide & Conquer Treemaps

[Jie, Nguyen, Simoff and Huang 2013, 2015]

• Angular D&C Algorithm:
  – Similar to D&C
  – Apply the partition process for each group but with pre-defined angles
    • Partition can be made from a side
    • If angle is 90° → normal Treemap
  – The polygonal algorithm produces mostly rectangles
• Can be applied virtually to any shapes
Voronoi Treemaps

[Balzer et al 2005]

• Generate polygonal Treemap layouts by utilising centroidal Voronoi tessellations

• **Pros**: generate polygons within areas of arbitrary shape

• **Cons**: high in computation

• https://www.jasondavies.com/voronoi-treemap/
Sunburst
[Stasko & Zhang 2000]

• Standard Algorithm:
  – Use enclosure to partition the hierarchy, but in a circular or radial manner
  – The layout algorithm is based on the other radial space-filling techniques from the circular visual designs [Chuah 1998] and information slices [Andrews & Heidegger 1998].
Space-Optimized Tree
[Nguyen & Huang 2002,2003]

• Put root node at centre, then draw children out radially
• Key: Smart positioning to optimize placement of branches (Voronoi diagram-like approach)
EncCon
[Nguyen & Huang 2004, 2005]

An example of the entire Java Doc. v.1.4.1 with approximately 9500 directories and files

An example of huge data set of approximately 50,000 nodes
Other Popular Tree Drawing Techniques

- Balloon View
- NicheWorks
- Disk-Tree
- Cone Tree
- 3D Hyperbolic
- Botanical Tree
Small Exercise

In your opinion,

How Many Tree Visualization Techniques are Available?
275 Published Techniques (and Variations) and more ...

Source: http://vcg.informatik.uni-rostock.de/~hs162/treeposter/poster.html
275 Published Techniques and more ...

Source: http://vcg.informatik.uni-rostock.de/~hs162/treeposter/poster.html
275 Published Techniques and more ...

Source: http://vcg.informatik.uni-rostock.de/~hs162/treeposter/poster.html
275 Published Techniques and more ...
275 Published Techniques and more ...

Source: http://vcg.informatik.uni-rostock.de/~hs162/treeposter/poster.html
275 Published Techniques and more ...

Source: http://vcg.informatik.uni-rostock.de/~hs162/treeposter/poster.html
275 Published Techniques and more ...

Source: http://vcg.informatik.uni-rostock.de/~hs162/treeposter/poster.html
GRAPH VISUALISATION
Hierarchical Drawings, Sugiyama (cont)
Hierarchical Drawings, Sugiyama

- directed acyclic graphs, DAGs
  

  (1) break cycles
  (2) compute layering, the Y-coordinates
      and insert dummy nodes for long-span edges
  (3) crossing reduction
      repeat
          down phase: sort next layer
          placement on lower layer
          up phase: sort previous layer
          placement on upper layer
      until DONE
  (4) routing of the edges
Pros and Cons

• Pros:
  – Most visual edges flow in the same direction
  – Number of their intersections are minimized
  – Show the flow and hierarchy very clearly

• Cons:
  – Not well suit for general graphs
  – Not scale well on large graphs

• Often chosen for the arrangement of flow diagrams, process charts and workflows
Force Directed Methods

Idea: a Spring model (Originated by Peter Eades, USYD)

select optimal edge length (node distance) \( k \)
repeat
    for each node \( v \) do
        for each pair of nodes \( (u, v) \)
            compute repulsive force \( f_r(u,v) = -c \cdot \frac{k^2}{d(u, v)} \)
        for each edge \( e = (u,v) \)
            compute attractive force \( f_a(u,v) = c \cdot \frac{d(u,v)^2}{k} \)
        sum all force vectors \( F(v) = \sum f_r(u,v) + \sum f_a(u,v) \)
        move node \( v \) according to \( F(v) \)
until DONE
Forces Directed Methods

• Attractive forces
  – along each edge
  – proportional to shortest paths

• Repulsive forces
  – between each pair of nodes \( O(n^2) \) pairs, costly!
  – only between closely related nodes (hash grid)

• Other forces
  – center of gravity (attractive)
  – underlying magnetic fields (concentric, radial, horizontal)
  – angular forces (between adjacent edges at nodes \( v \))
  – from the boundary (repulsive; bounce back)
Forces Directed Methods

Demo


Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
Pros and Cons

• Pros:
  – Well suite for general graphs with different variations
  – Good layout
  – Easy to implement

• Cons:
  – Layout is not predictable
  – Slow in computational cost $O(N^3)$ or $O(N^2 \log(N))$
  – Too much twisting to achieve optimal layout
  – Not scale well on large graphs of more than thousands of nodes

• Widely use in practice
Big Graphs

• Difficult to keep all in memory
• Not effective to show all nodes
• Often visualized as “hairballs”
• Solutions
  – Structural clustering or
  – Abstraction and aggregation
  – Show a high-level overview of topology
Address Computational Scalability – Multilevel Approaches

[Schulz 2004]
Address Computational Scalability — Abstraction/Aggregation

Cytoscape.org

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
Address Computational Scalability —
Collapsible Force Layout

• Supernodes: aggregate of nodes
• Manual or algorithmic clustering

http://bl.ocks.org/mbostock/1062288
Aesthetics

• What is a nice drawing?
• What makes drawings understandable or readable?
• How can we measure quality?
• Can we formalise aesthetics?

• A proverb
  "A picture is worth a thousand words"
• R. Feynman (Nobel prize in Physics)
  "It’s all visual"
• R.A. Earnshaw (a pioneer in computer graphics, 1973)
  "visualization uses interactive compute graphics to help provide insight on complicated problems, models or systems".
  "Scientific visualization is exploring data and information graphically, gaining understanding and insights into the data"
• R. Hamming (1973)
  "the purpose of computing is insight not numbers"
Aesthetic Criteria in Graph Drawing

• **Edge Crossings**: minimise towards planar
• **Total Edge Length**: minimise towards proper scale
• **Area**: minimise towards efficiency
• **Reducing maximum Edge Length**: minimise longest edge
• **Uniform Edge Lengths**: minimise variances
• **Total Bends**: minimise orthogonal towards straight-line
Drawing Styles

• polyline drawings
  reduce bends, no sharp angles, polish by with Bezier splines

• straight-line or curve
  uniform (short) edge length

• orthogonal drawings
  minimize bends

• planar drawings
  minimize crossings and bends

• grid embeddings
  grid coordinates for nodes and bend-points

• visibility
  horizontal bar nodes and vertical visibility
Node Attributes

- Colouring
- Positions
- Size
- Shapes
- Multiple Views
- Etc.
Edge Attributes

- Colouring
- Positions
- Size
- Shapes
- Thickness
- Drawing style
  - Straight line
  - Curve (edge bundling)
  - Orthogonal

Matrix Representations

• Use adjacency matrix to show the relations
Matrix Representations - Exercise

• Show the adjacency matrix for the following graph
NETWORK VISUALISATION
What’s common in Network Visualisation?

- A Network is a type Graph
- Usually refer to real applications with large size and high complexity
  - Millions of items and connections
- Structural behaviour is usually unknown
- Relationships are usually non-directional
- Aesthetic Criteria is not usually considered as the most important criteria for the design
- Usually accompany with an interaction technique for the exploration
Visualisation of the Internet

How many web pages that the Internet has? ≈ 50 billion Web Pages

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au

http://www.worldwidewebsize.com/
Facebook
1.5+ Billion Users
And Many More Networks

- Citation Network: 250+ million Articles
- Twitter: 500+ million Articles
- Amazon: 120+ million Users
- AT&T: mobile phone network 100+ million users
- Protein-protein interactions: 200 million possible interactions in human genome
Small-World Networks

• A small-world network is a type of mathematical graph in which most nodes are not neighbours of one another, but most nodes can be reached from every other by a small number of hops or steps.

• A small world network, where nodes represent people and edges connect people that know each other, captures the small world phenomenon of strangers being linked by a mutual acquaintance.
Small-World Networks (cont.)

• It needs **6-steps** to reach an unknown item
• Many empirical graphs are well modeled by small-world networks.
  – Social networks
  – The connectivity of the Internet
  – Gene networks
World Trade in 1992

http://www.mpi-fg-koeln.mpg.de/~lk/netvis/trade/WorldTrade.html

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
A Social Network Visualisation

www.pi-werk.de/projects-interactive.html

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
Visualisation of a very large Network

Problem?
How to deal with large Networks or Graphs?

• Partition a graph into a hierarchy of sub-graphs
  – Simplify the complex structures of large graphs through the global abstraction for easy interpretation, perception and navigation of large information spaces.

• Improve the layout algorithms to utilise display space allowing more nodes and information to be displayed.
  – More information can be displayed on very high-resolution and large screen, but it does not necessarily provide very much more information into the brain [Ware 2004]
  – Investigation of space-efficient techniques for visualising large datasets could be more effective and economical than the use of expensive large display devices.

• Apply Interaction and Navigation

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
Visualisation of a very large Network
Combined Clustering & Space-Efficient
Visualisation of a very large Network
Combined Clustering & Space-Efficient
## Handling Large Graphs / Networks

<table>
<thead>
<tr>
<th>Graph</th>
<th>Nodes</th>
<th>Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>YahooWeb</td>
<td>1.4 Billion</td>
<td>6 Billion</td>
</tr>
<tr>
<td>Symantec Machine-File Graph</td>
<td>1 Billion</td>
<td>37 Billion</td>
</tr>
<tr>
<td>Twitter</td>
<td>104 Million</td>
<td>3.7 Billion</td>
</tr>
<tr>
<td>Phone call network</td>
<td>30 Million</td>
<td>260 Million</td>
</tr>
</tbody>
</table>
Step 1: Storing Large Graphs/Networks

• On your laptop computer
  – SQLite
  – Neo4j (GPL license)
• On a server
  – MySQL, PostgreSQL, etc.
• With a cluster
  – Hadoop (generic framework)
  – HBase, inspired by Google’s BigTable
  – Hama, inspired by Google’s Pregel
  – FlockDB, by Twitter
• Comparison of “graph databases”
  – http://nosql.mypopescu.com/post/40759505554/a-comparison-of-7-graph-databases
Storing Large Graph on a PC - SQLite

• Light weight
• Easily handle up to gigabytes"
• Roughly tens of millions of nodes/edges (perhaps up to billions?). Very good! For today’s standard.
• Very easy to maintain: one cross-platform file
• Has programming wrappers in numerous languages
  - C++, Java (Andriod), Python, Objective C (iOS),...
• Queries are so easy!
  - e.g., find all nodes’ degrees = 1 SQL statement"
• Bonus: SQLite even supports full-text search
Step 2: Data Mining or Graph Mining

• Analyse it first by data mining or graph mining.
• If the dataset is small, we can visualise it first
• Does it follow any expected patterns? Or does it NOT follow some patterns (outliers)?
  – Why does this matter?"
  – If we know the patterns (models), we can do prediction, recommendation, etc.
    • e.g., is Alice going to “friend” Bob on Facebook?
  – People often buy beer and diapers together.
  – Outliers often give us new insights
    • e.g., telemarketer’s friends don’t know each other
Step 3: Visualising the Results

- Visualise data mining results
- Visualise the graph using mining attributes
  - E.g. filtering, attributes, importance
- Hierarchical visualisation based on clusters from clustering process
  - Coarse to granularity
- Aims: find or verify patterns or (ab)normality
GRAPH TOOLS & APPLICATIONS
Prefuse Visualization Toolkit

- Rich set of features for data modelling, visualization, and interaction
- Written in Java, using the Java 2D graphics library → can use for web applets
Jgraph: Java Graph Visualization and Layout

- Powerful, easy-to-use, feature-rich and standards-compliant open source graph visualization for Java.
- [http://www.jgraph.com/jgraph.html](http://www.jgraph.com/jgraph.html)
Jung - Java Universal Network/Graph Framework

- Provides a common and extendible language for the modeling, analysis, and visualization of data that can be represented as a graph or network.
- Written in Java, which allows JUNG-based applications to make use of the extensive built-in capabilities of the Java API, as well as those of other existing third-party Java libraries.
Gephi
https://gephi.org/

The Open Graph Viz Platform

Gephi is an interactive visualization and exploration platform for all kinds of networks and complex systems, dynamic and hierarchical graphs.

Runs on Windows, Mac OS X and Linux. Gephi is open-source and free.

Learn More on Gephi Platform »
Cytoscape
http://www.cytoscape.org/
Cytoscape Web
http://cytoscapeweb.cytoscape.org/
NetworkX
https://networkx.github.io/
Proceus - Network-based Visual Analysis of Tabular Data

http://www.cc.gatech.edu/gvu/ii/ploceus/

Dr. Quang Vinh Nguyen. Email: q.nguyen@westernsydney.edu.au
NEXT WORKSHOP
VISUAL ANALYTICS FOR MULTIDIMENSIONAL DATA
VANTED