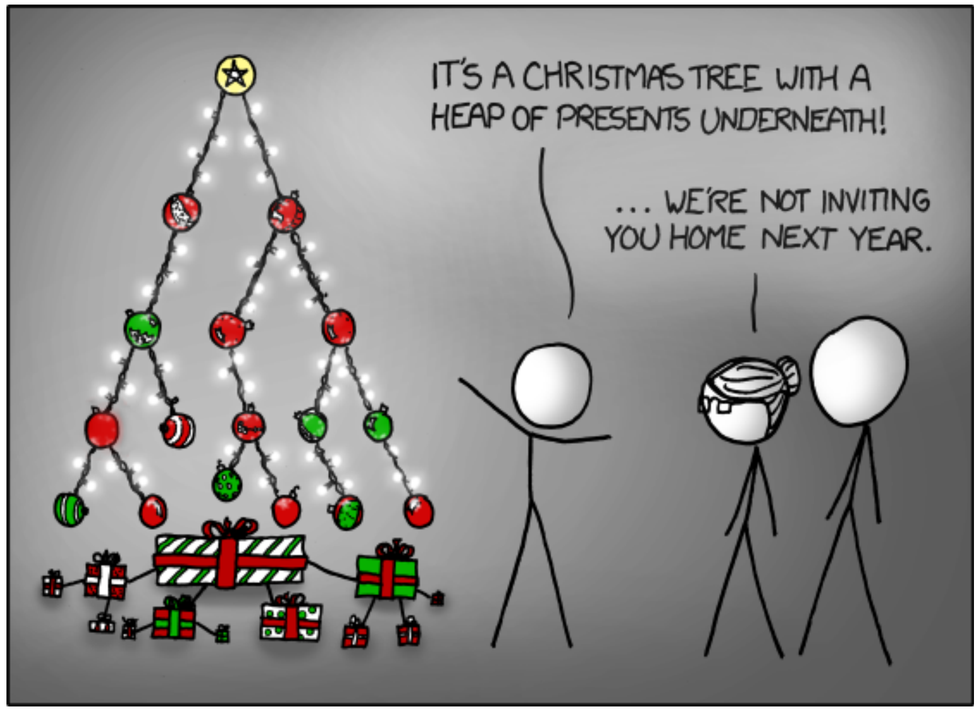
CS-5630 / CS-6630 Visualization
Graphs

Alexander Lex
alex@sci.utah.edu

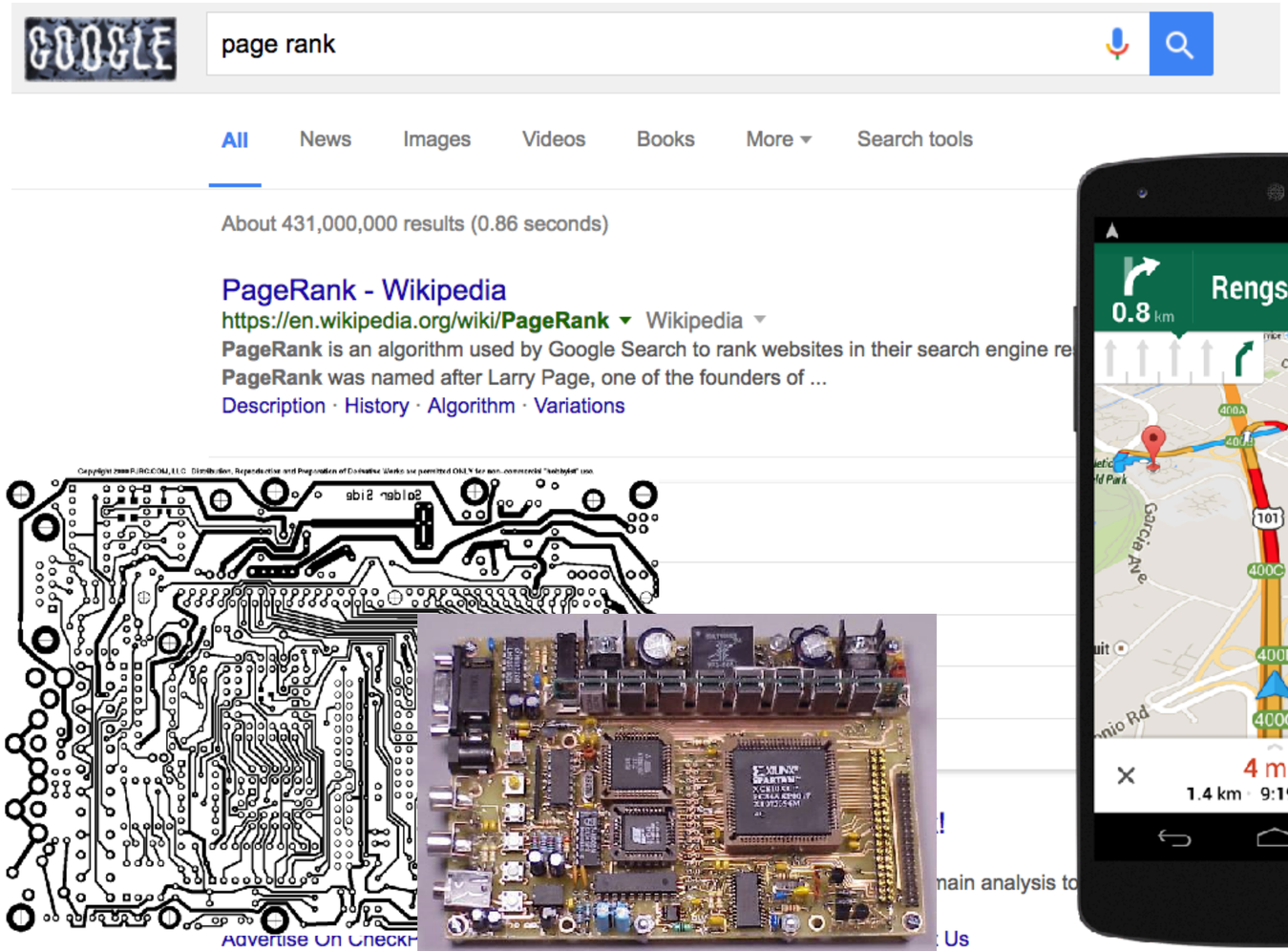
THE UNIVERSITY OF UTAH



[xkcd]

Applications of Graphs

Without graphs, there would be none of these:`




facebook
December 2010

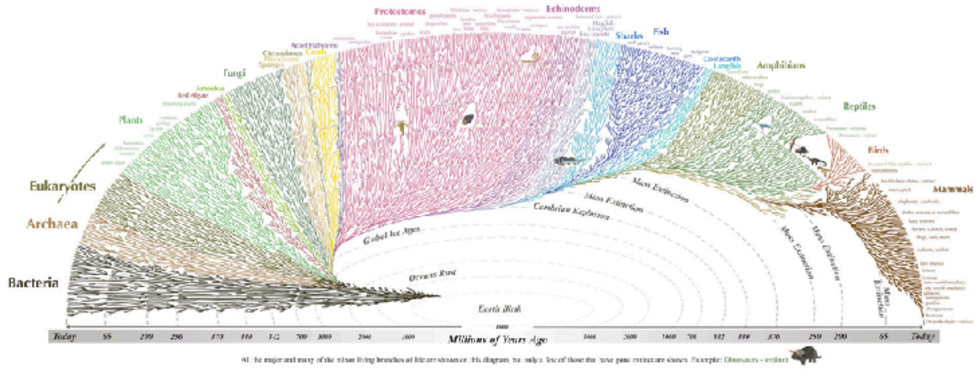
Biological Networks

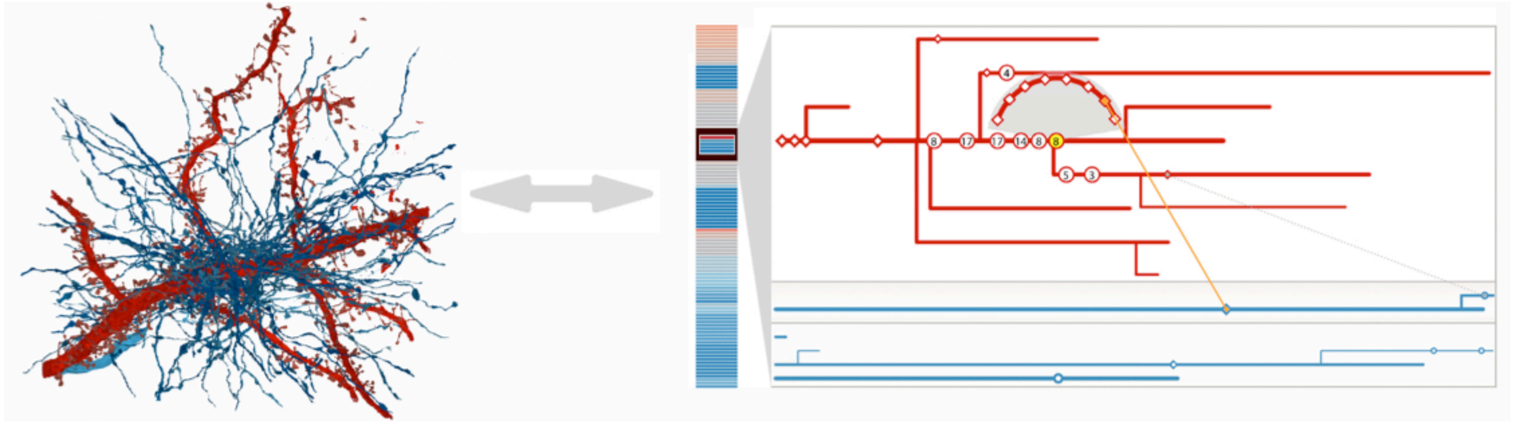
Interaction between genes, proteins and chemical products

The brain: connections between neurons

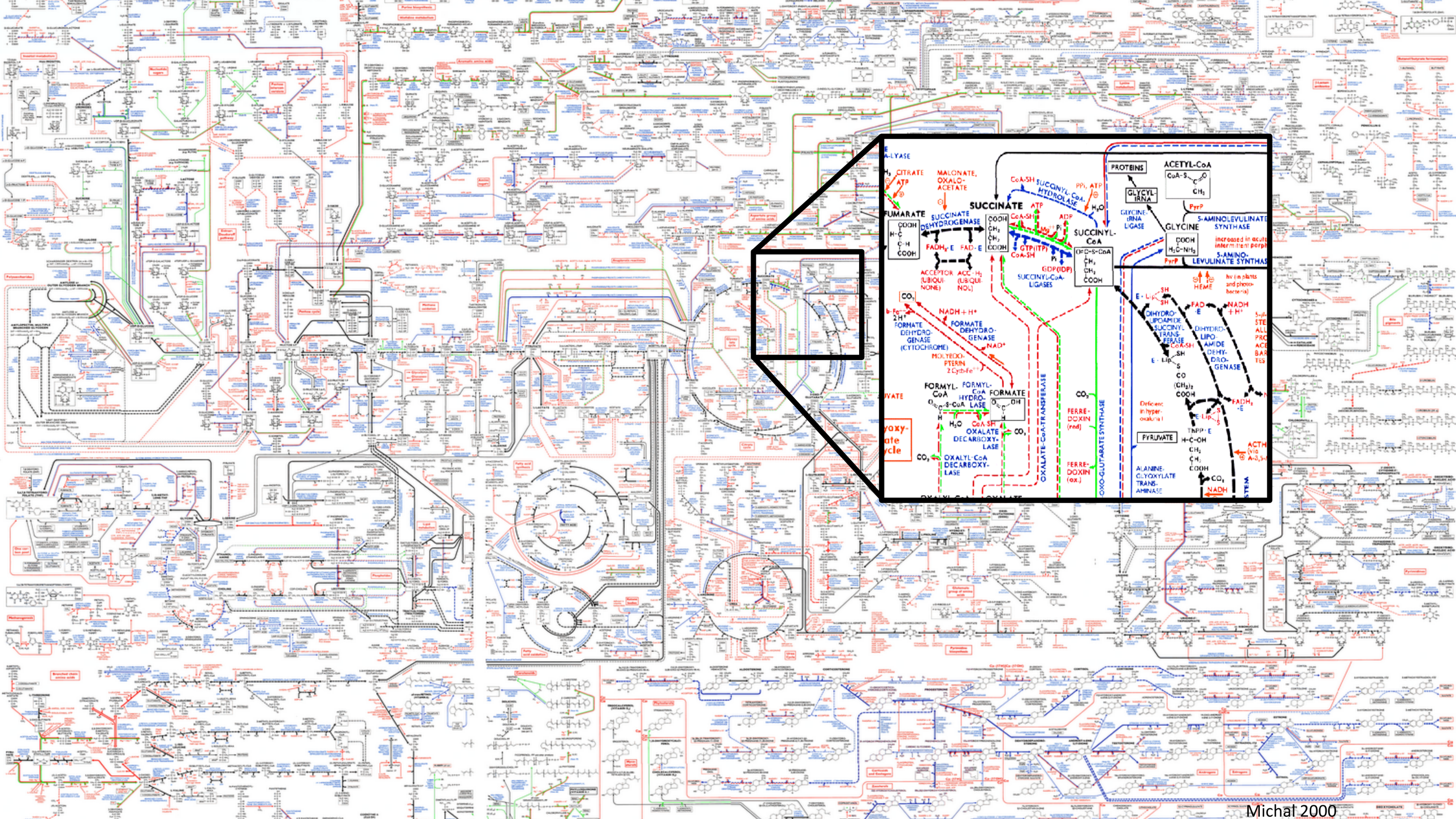
Your ancestry: the relations between you and your family

Phylogeny: the evolutionary relationships of life



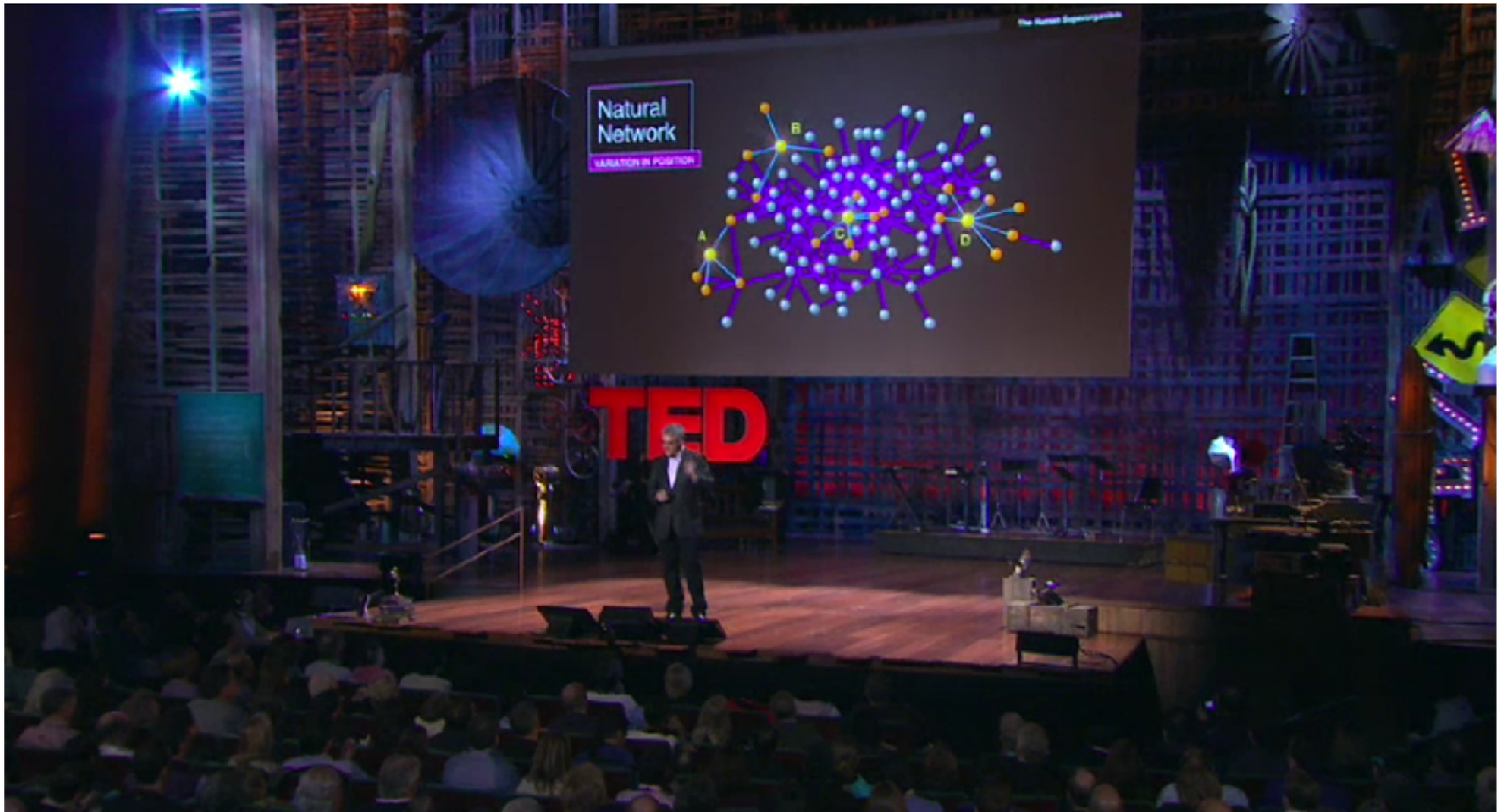


[Beyer 2014]


CITRATE
ATP
MALONATE, OXALO-ACETATE
SUCCINATE
SUCCINATE DEHYDROGENASE
FUMARATE
FADH2 E
FAD E
ACCEPTOR (UBIQUI-NONE)
ACC-H2 (UBIQUI-NOL)
CoA-SH
SUCCINYL-CoA HYDROLASE
PPi, ATP
H2O
ADP
Pi
GTP(ITP)
GDP(IDP)
SUCCINYL-CoA-LIGASES
SUCCINYL-CoA
PROTEINS
GLYCYL-tRNA
GLYCINE-tRNA LIGASE
GLYCINE
ACETYL-CoA
Pyr-P
5-AMINOLEVULINATE SYNTHASE
increased in acute intermittent porp
5-AMINO-LEVULINATE SYNTHAS
HEME
hv in plants and photo-bacteria)
CO2
NADH + H+
FORMATE DEHYDRO-GENASE (CYTOCHROME)
MOLYBDO-PTERIN
2 Cyt b Fe+++
FORMATE DEHYDRO-GENASE
NAD+
FORMYL-CoA
FORMYL-CoA HYDRO-LASE
FORMATE
H2O
CoA SH
OXALATE DECARBOXY-LASE
OXALYL-CoA DECARBOXY-LASE
OXALATE-CoA-TRANSFERASE
FERRE-DOXIN (red)
FERRE-DOXIN (ox.)
OXO-GLUTARATE SYNTHASE
DIHYDRO-LIPOAMIDE SUCCINYL-TRANS-FERASE
DIHYDRO-LIPO-AMIDE DEHY-DRO-GENASE
FAD E
NADH + H+
FADH2 E
Deficienc in hyper-oxaluri
PYRUVATE
ALANINE-GLYOXYLATE TRANS-AMINASE
ACTH (via 3',5'-cA
NADH

Michal 2000

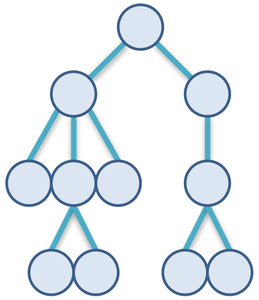
Graph Analysis Case Study



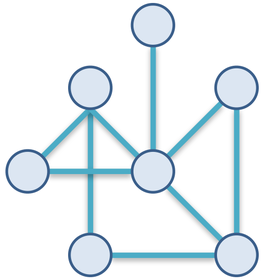
Graph Theory Fundamentals

See also “Network Science”, Barabasi
http://barabasi.com/networksciencebook/chapter/2

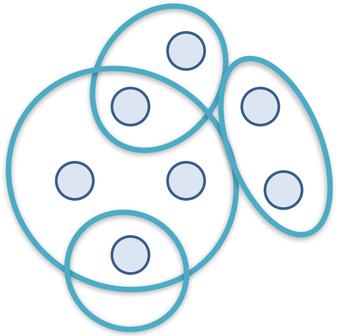
Tree



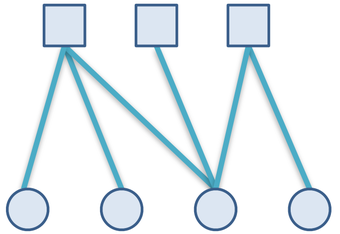
Network



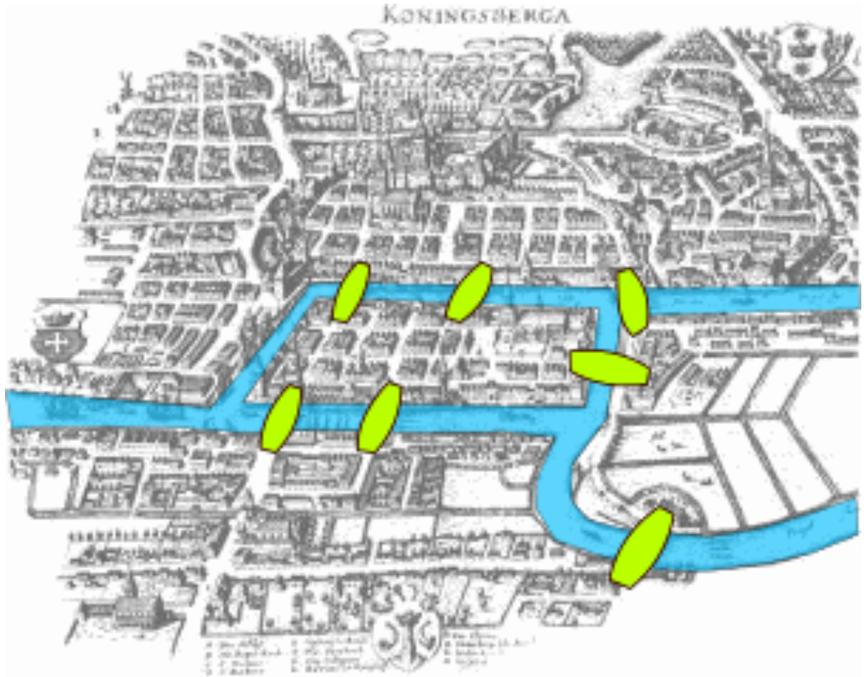
Hypergraph

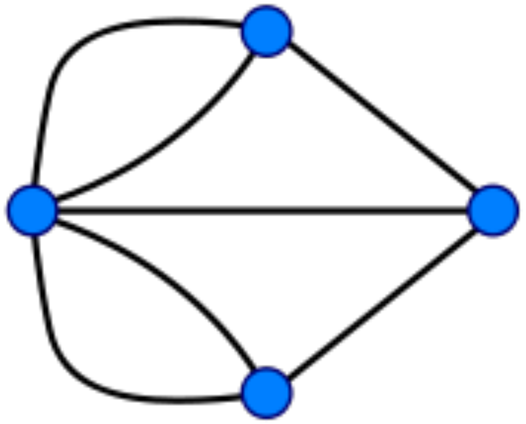


Bipartite Graph



Königsberg Bridge Problem (1736)





Only possible with a graph with at most two nodes with an odd number of links.
This graph has four nodes with odd number of links.

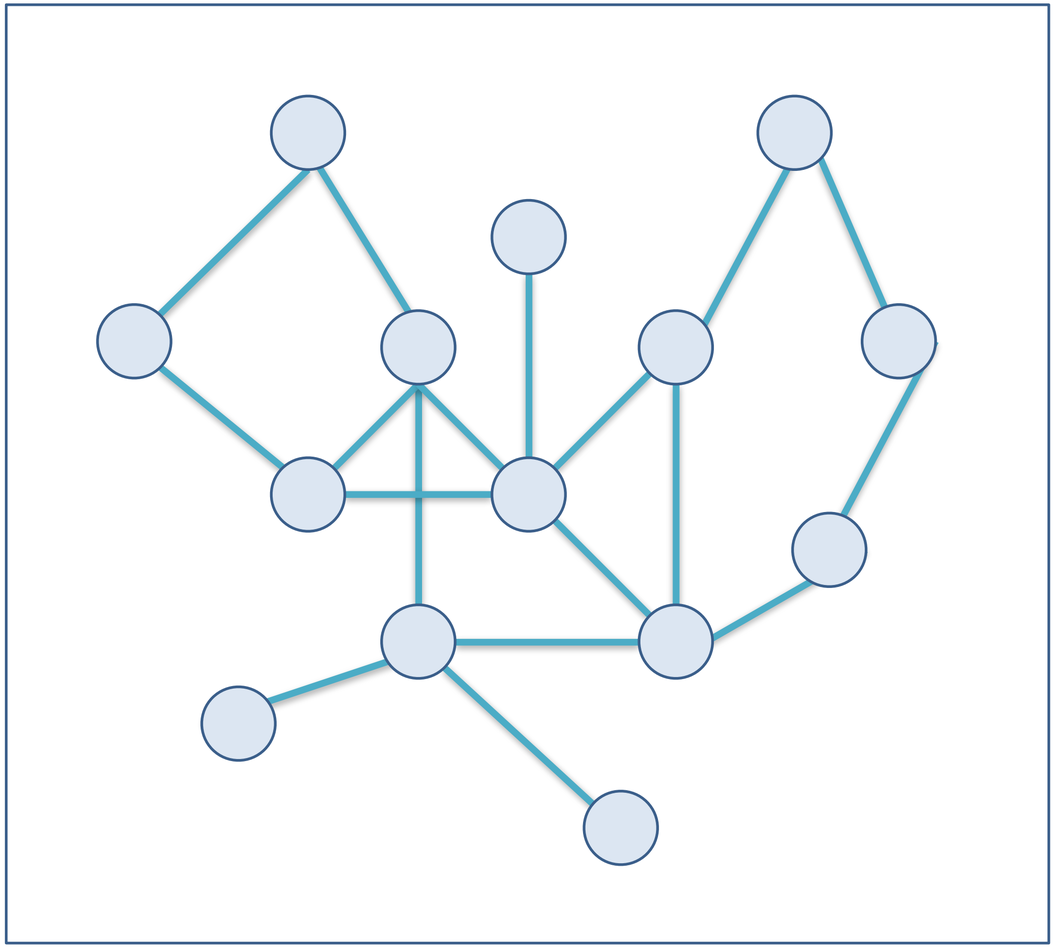
http://barabasi.com/networksciencebook/chapter/2#bridges

Graph Terms

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

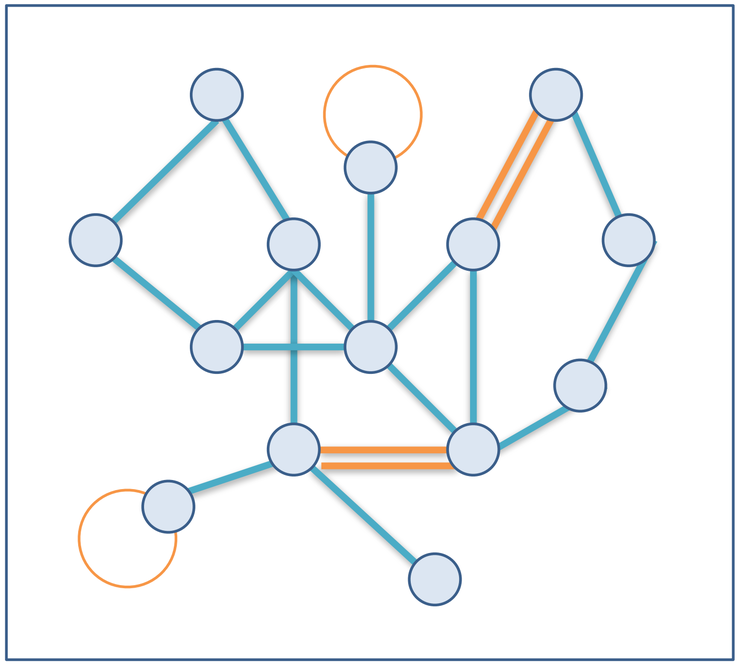
set of **edges E** (also called links) connecting these vertices.

Graph and **Network** are often used interchangeably



Graph Term: Simple Graph

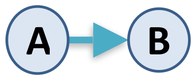
A simple graph G(V,E) is a graph which contains **no multi-edges** and **no loops**

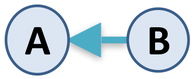


Not a simple graph!

→ A ***general graph***

Graph Term: Directed Graph

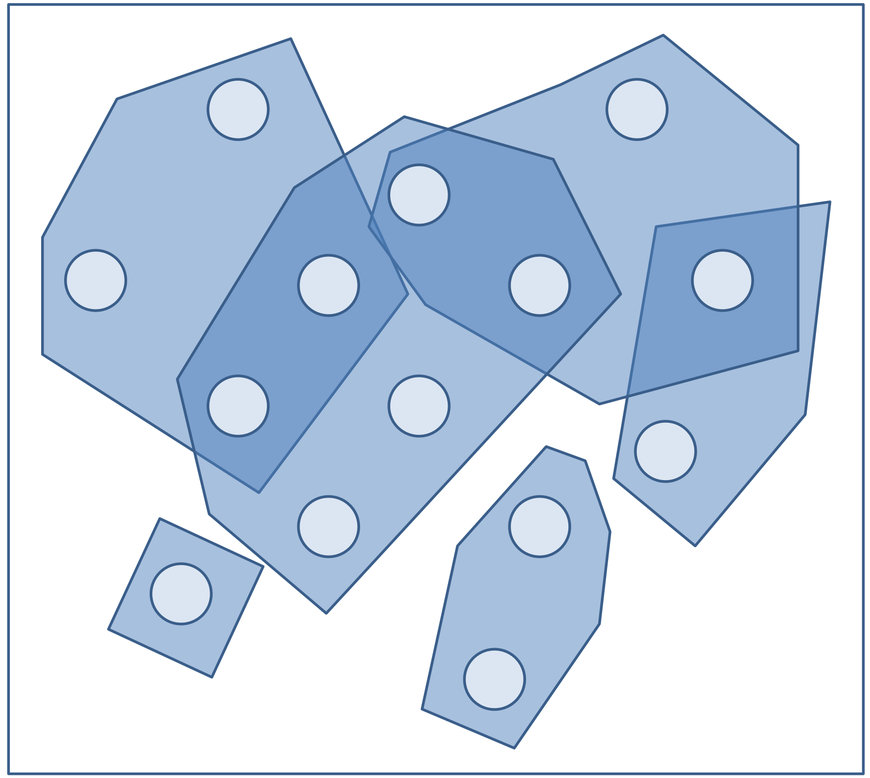
A directed graph (digraph) is a graph that discerns between the edges 

and 

.

Graph Terms: Hypergraph

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

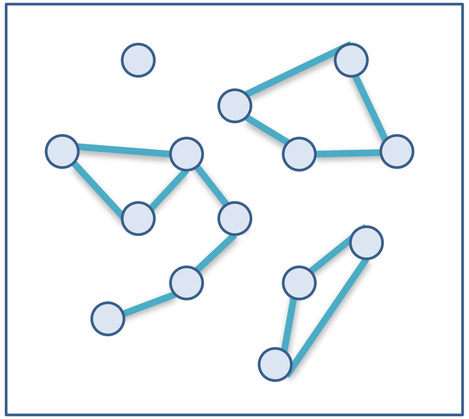


Hypergraph Example

Unconnected Graphs, Articulation Points

Unconnected graph

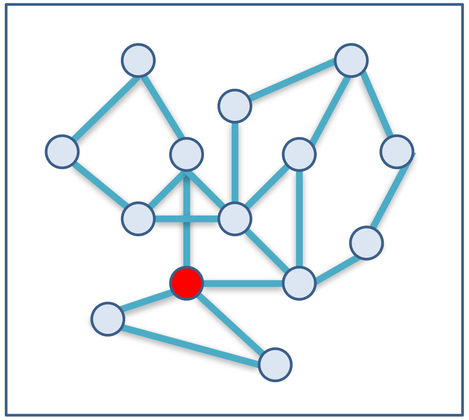
An edge traversal starting from a given vertex cannot reach any other vertex.



Unconnected Graph

Articulation point

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

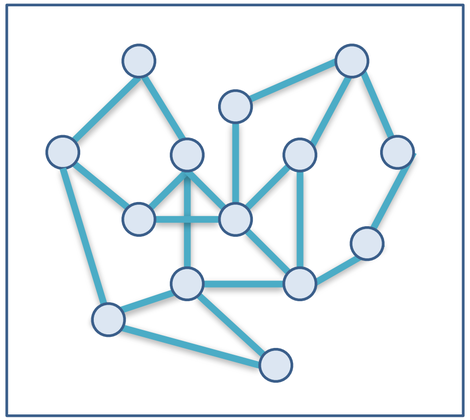


Articulation Point (red)

Biconnected, Bipartite Graphs

Biconnected graph

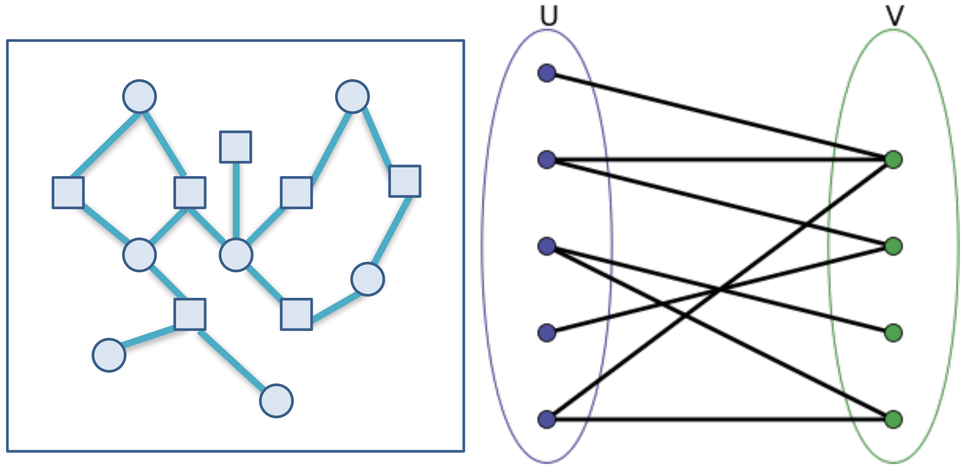
A graph without articulation points.



Biconnected Graph

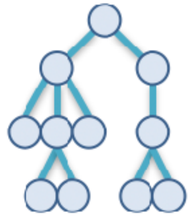
Bipartite graph

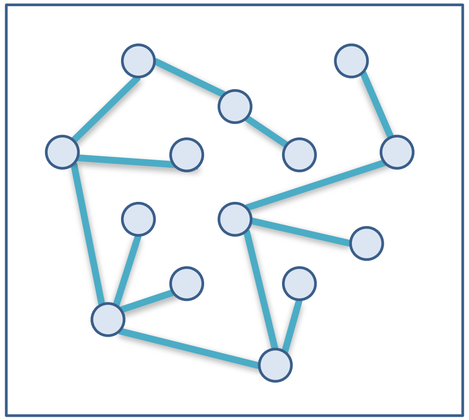
The vertices can be partitioned in two independent sets.



Bipartite Graph

Tree



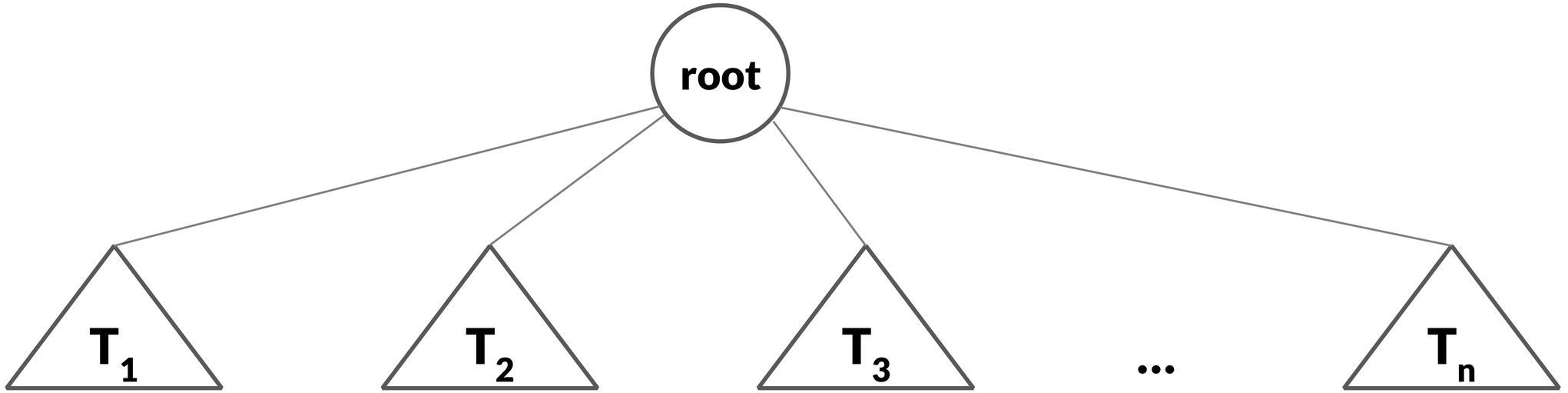


A graph with no cycles - or:

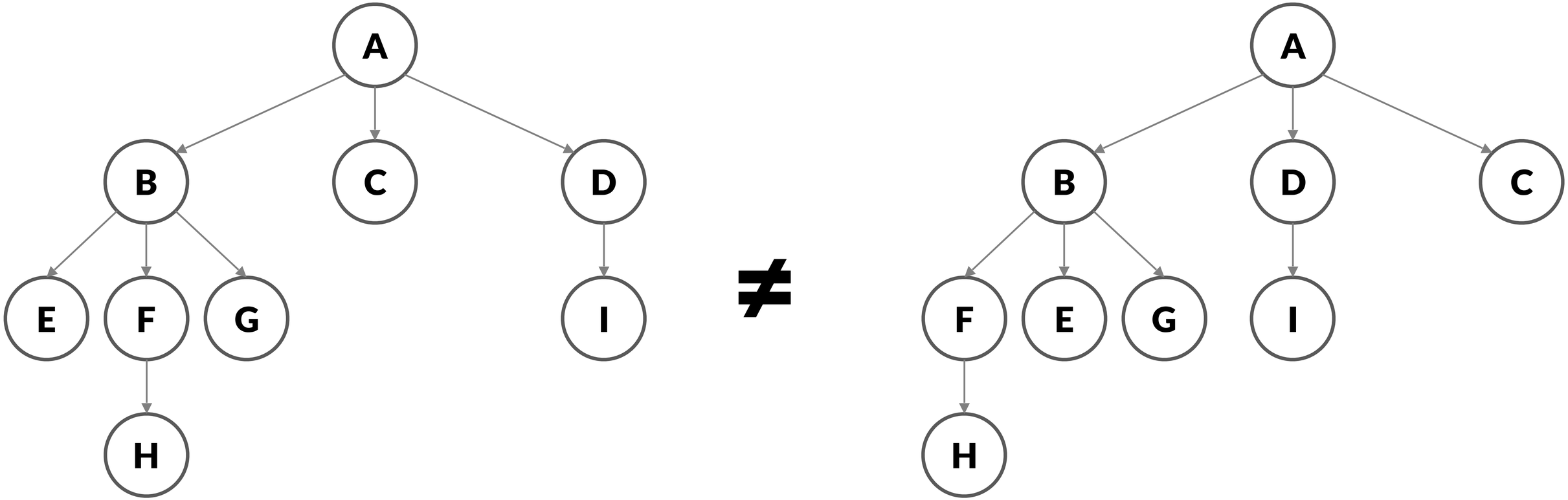
A collection of nodes

contains a root node and 0-n subtrees

subtrees are connected to root by an edge



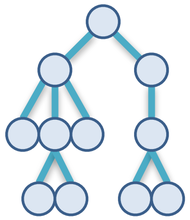
Ordered Tree



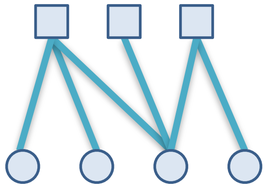
Different Kinds of Graphs

Over 1000 different graph classes

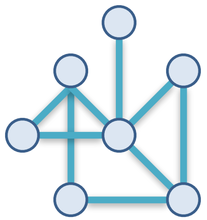
Tree



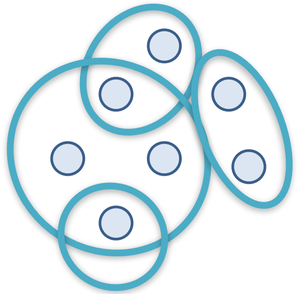
Bipartite Graph

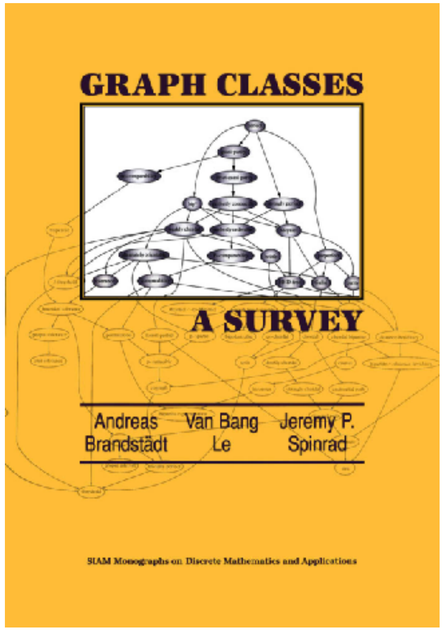


Network



Hypergraph





A. Brandstädt et al. 1999

Degree

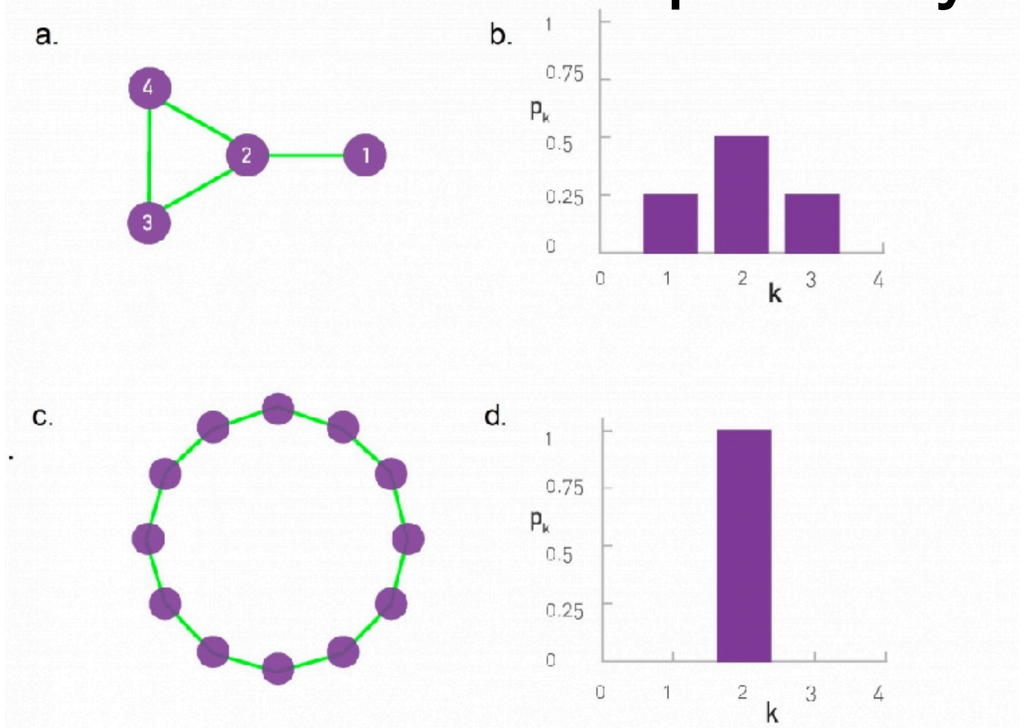
Node degree deg(x)

The number of edges being incident to this node. For directed graphs indeg/outdeg are considered separately.

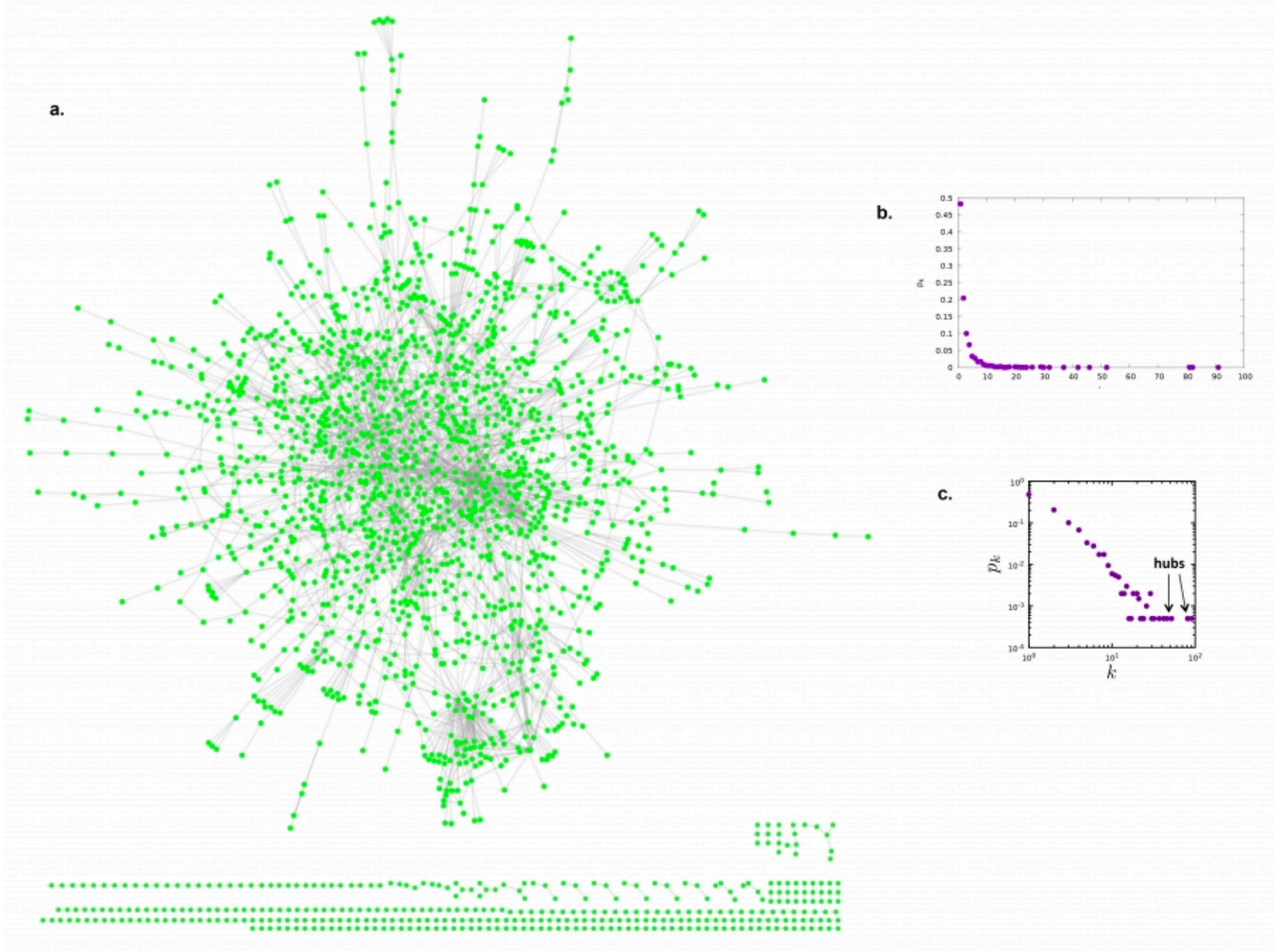
Average degree

$$\langle k \rangle = \frac{1}{N} \sum_{i=1}^{N} k_i = \frac{2L}{N}$$

Degree distribution



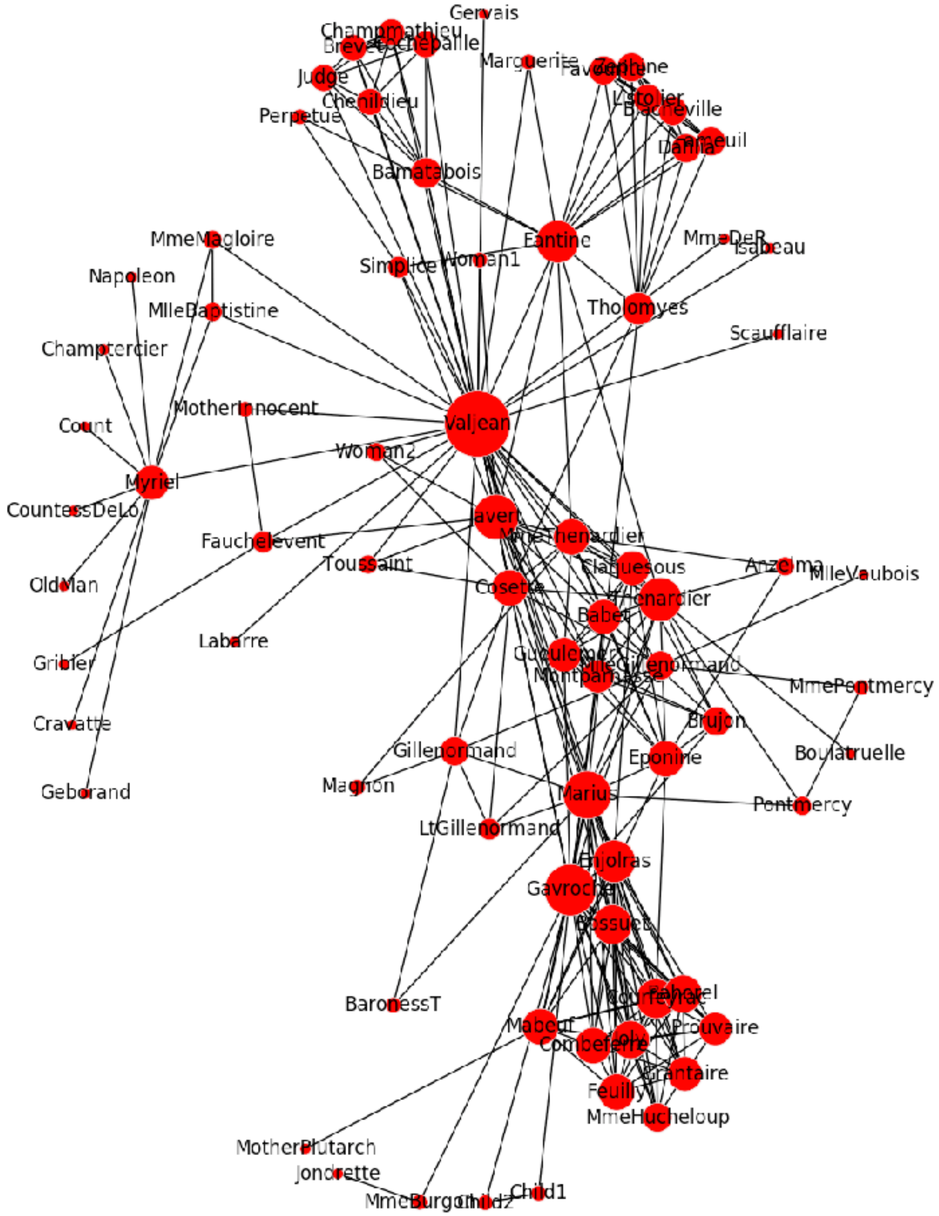
Degree Distribution of a real Network



Protein Interaction Network

Degrees

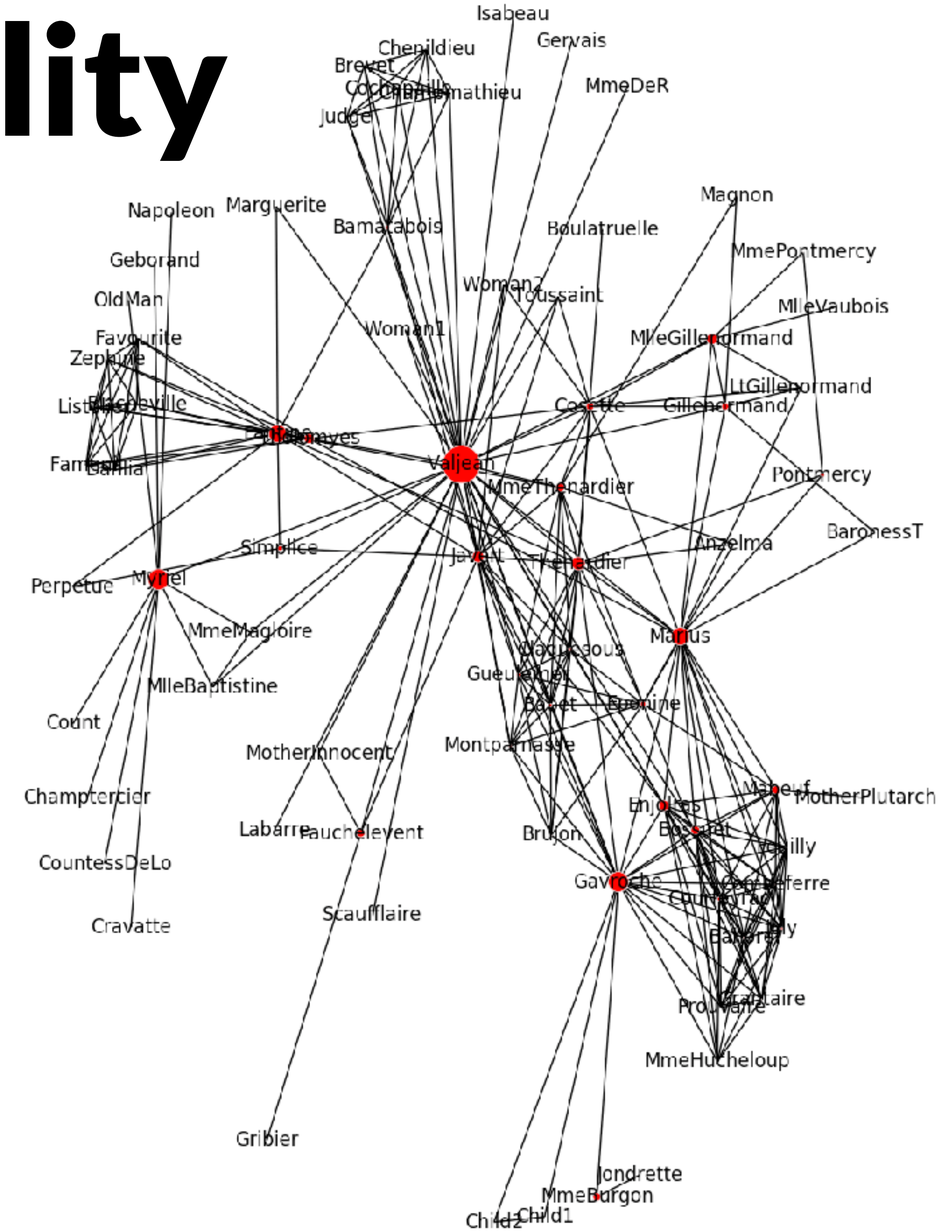
Degree is a measure of local importance



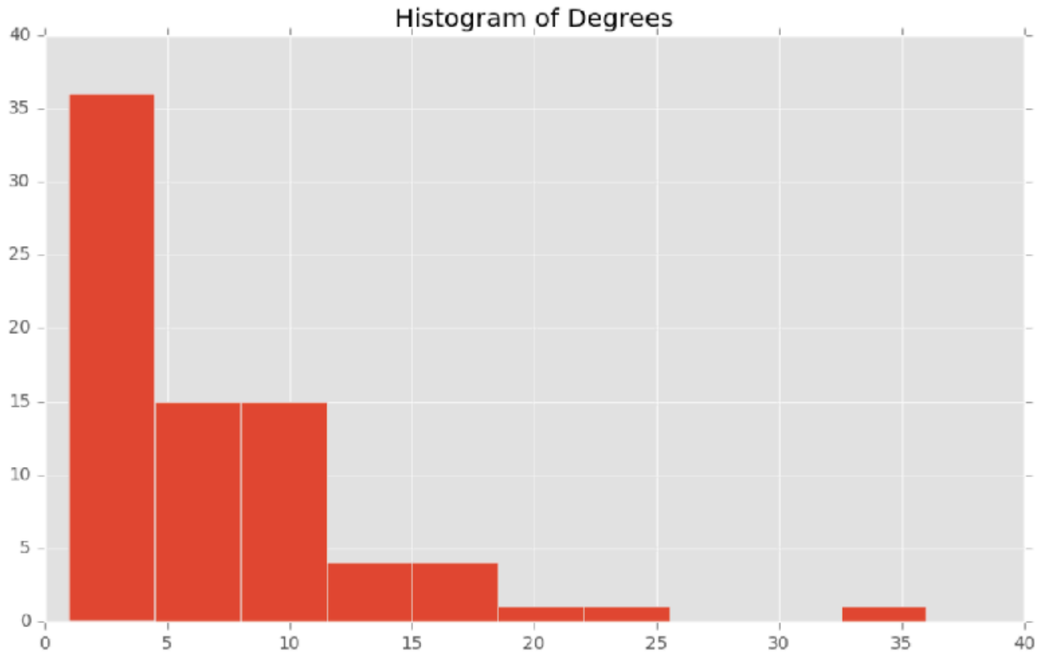
Betweenness Centrality

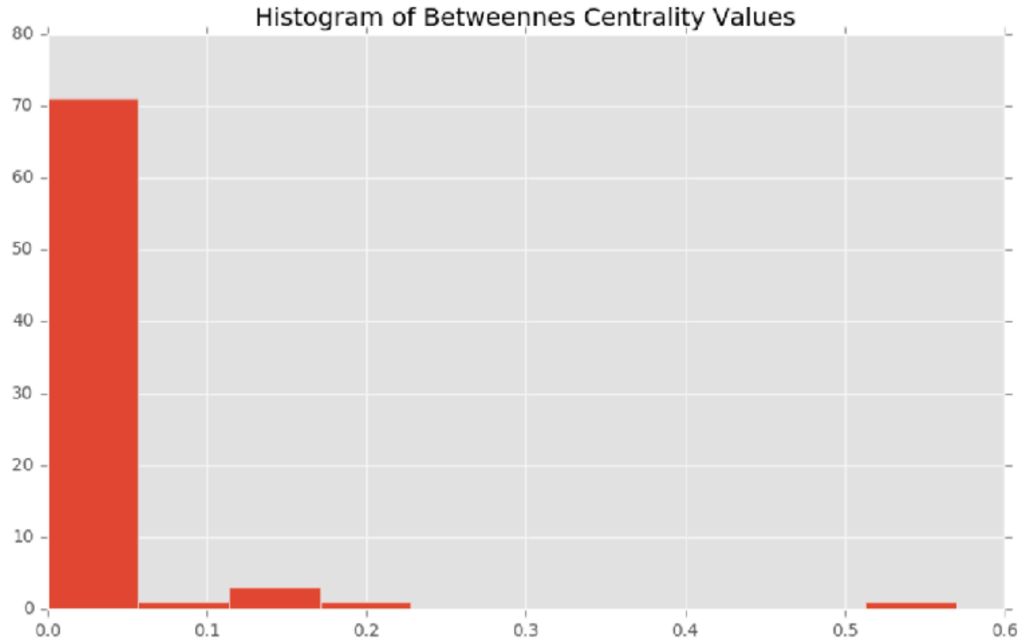
a measure of how many shortest paths pass through a node

good measure for the overall relevance of a node in a graph



Degree vs BC


Histogram of Degrees
40
35
30
25
20
15
10
5
0
0
5
10
15
20
25
30
35
40


Histogram of Betweennes Centrality Values
80
70
60
50
40
30
20
10
0
0.0
0.1
0.2
0.3
0.4
0.5
0.6

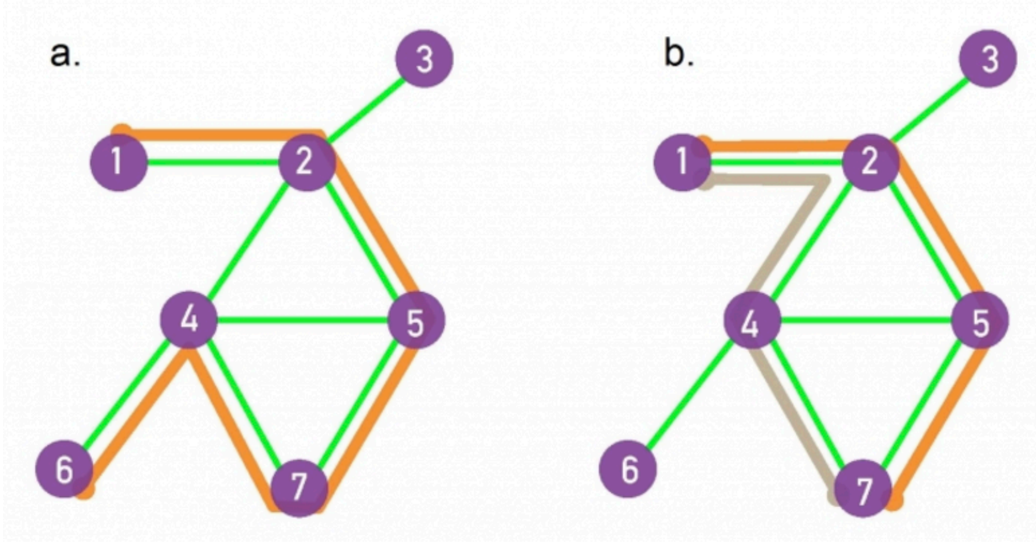
Paths & Distances

Path is **route along links**

Path length is the **number of links** contained

Shortest paths connects nodes i and j with the smallest number of links

Diameter of graph G
The longest shortest path within G.

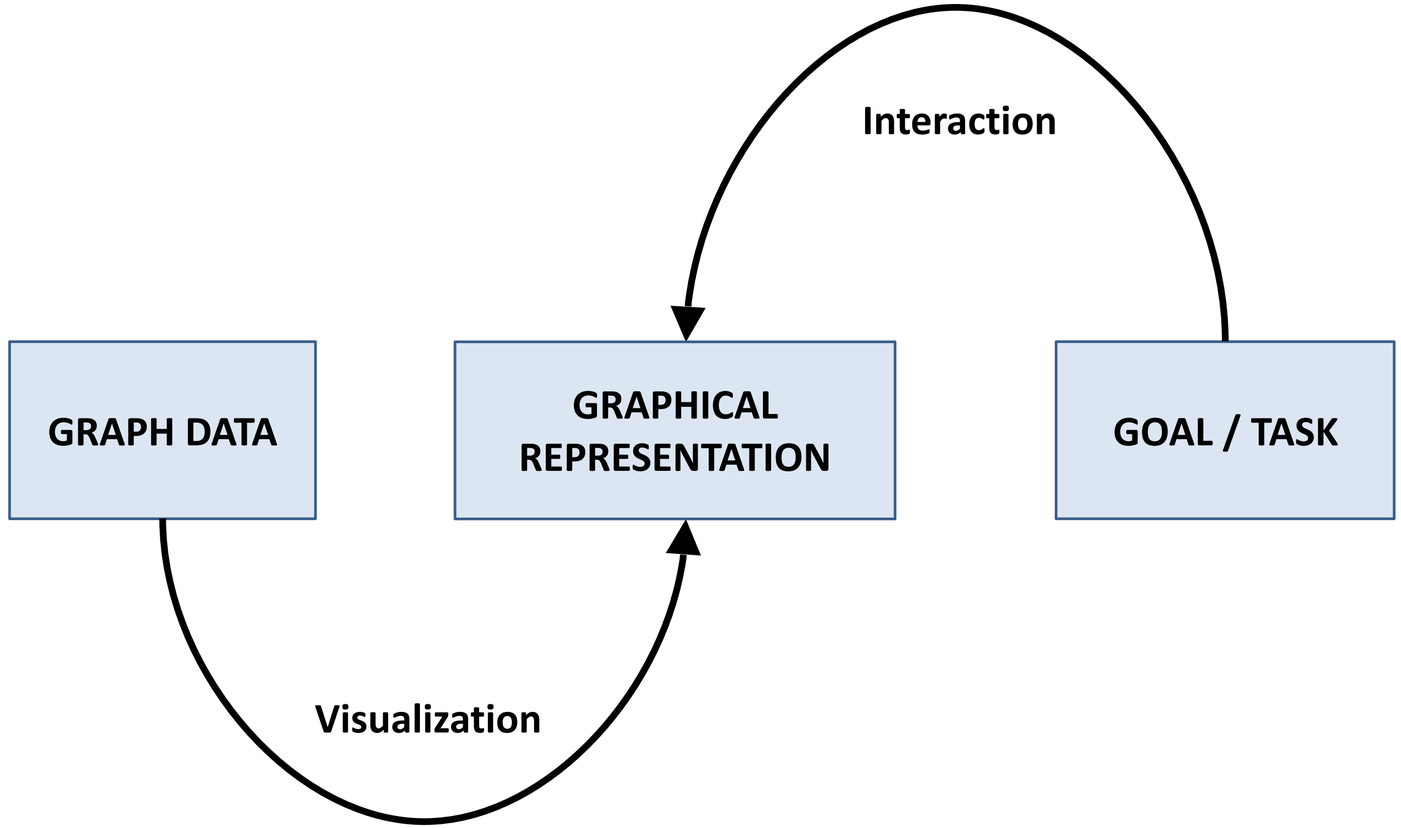


A path from 1 to 6

Shortest paths (two) from 1 to 7.

Graph and Tree Visualization

Setting the Stage



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

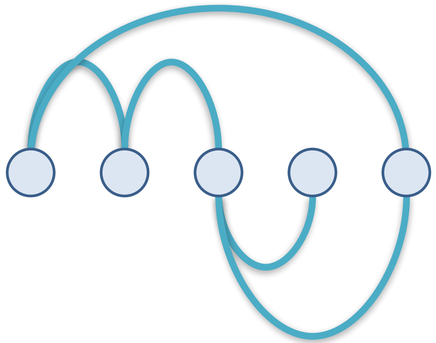
- ABT: Give the number of all nodes.
- TBT: Give the indegree (the number of incoming edges) of a node.

Sort/Order – enumerate the nodes/edges according to a given criterion

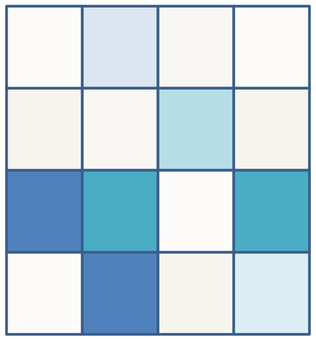
- ABT: Sort all edges according to their weight.
- TBT: Traverse the graph starting from a given node.

list adapted from Schulz 2010

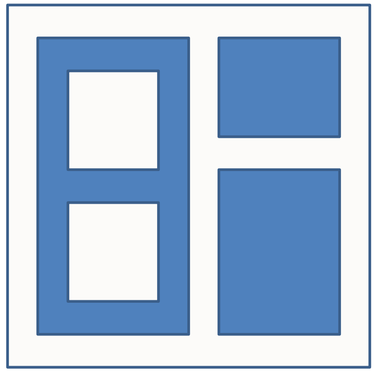
Three Types of Graph Representations



Explicit
(Node-Link)



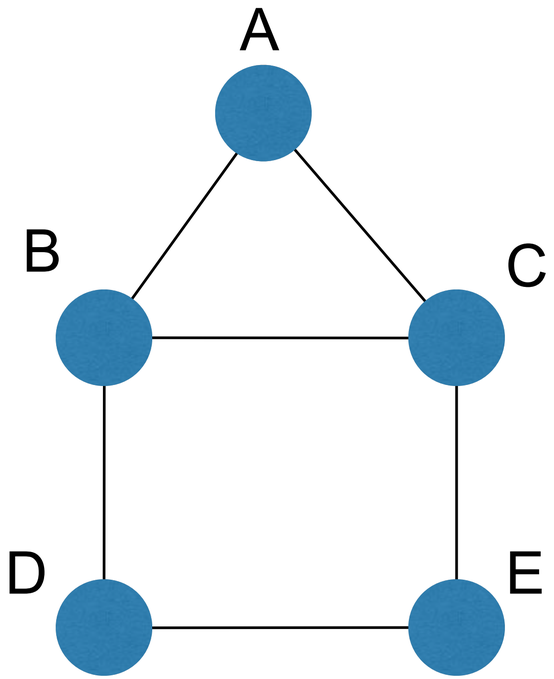
Matrix



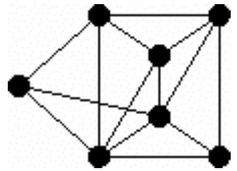
Implicit

Explicit Graph Representations

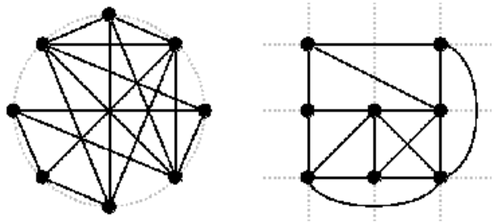
Node-link diagrams: vertex = point, edge = line/arc



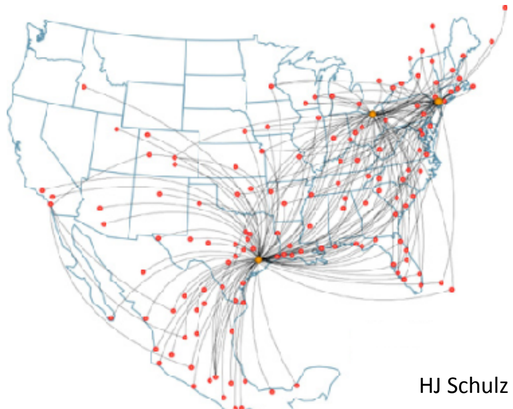
Free



Styled



Fixed



HJ Schulz 2006

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

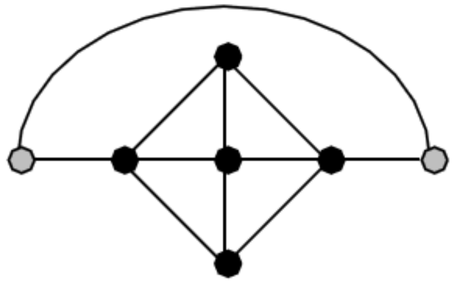
Aspect ratio about 1 (not too long and not too wide)

Symmetry: similar graph structures should look similar

list adapted from Battista et al. 1999

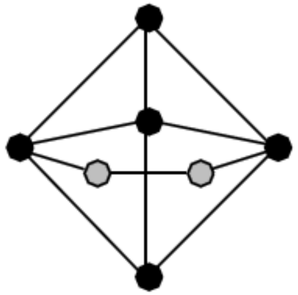
Conflicting Criteria

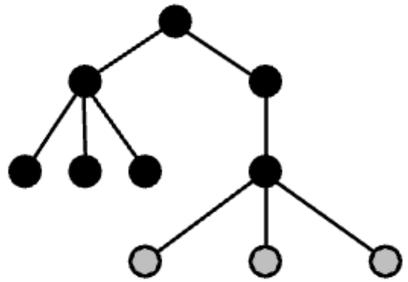
Minimum number of edge crossings



vs.

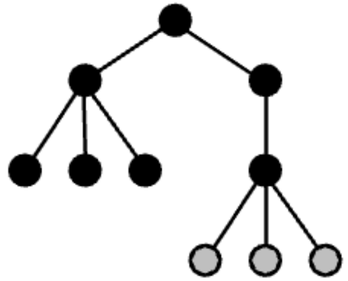
Uniform edge length





Space utilization

vs.



Symmetry

Schulz 2004

Force Directed Layouts

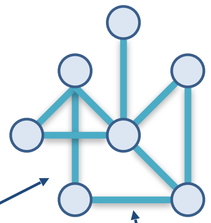
Physics model:
edges = springs,
vertices = repulsive magnets

in practice: damping,
center of gravity

Computationally

expensive: $O(n^3)$

Limit (interactive): ~1000 nodes



Expander
(pushing nodes apart)

Spring Coil
(pulling nodes together)

http://bl.ocks.org/steveharoz/8c3e2524079a8c440df60c1ab72b5d03

Giant Hairball

Explicit Representations

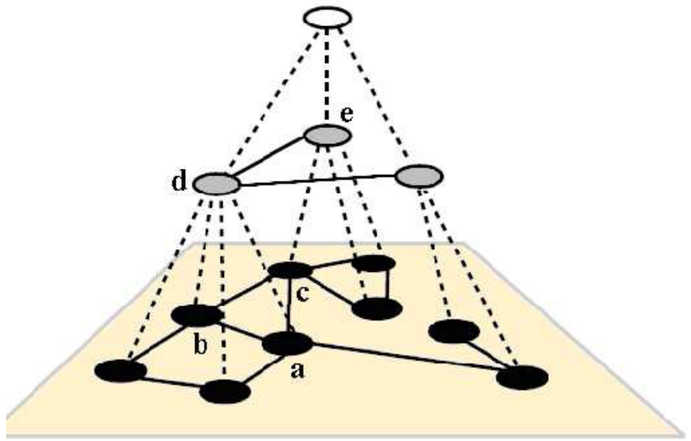
Problem #1: computing an optimal layout lies in NP

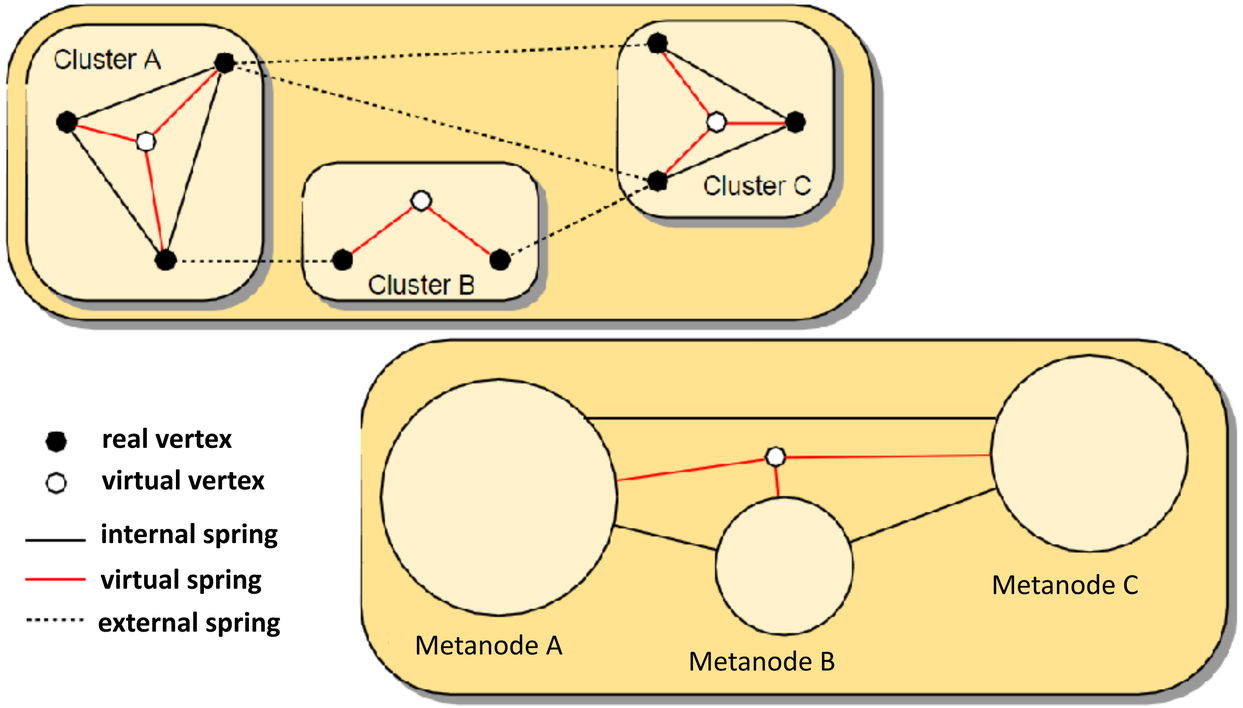
Solution approach: formulate the layout problem as an optimization problem

BUT: naïve runtime complexity is still $O(n^2)$!

in each optimization step, all vertices have to be checked against all other vertices

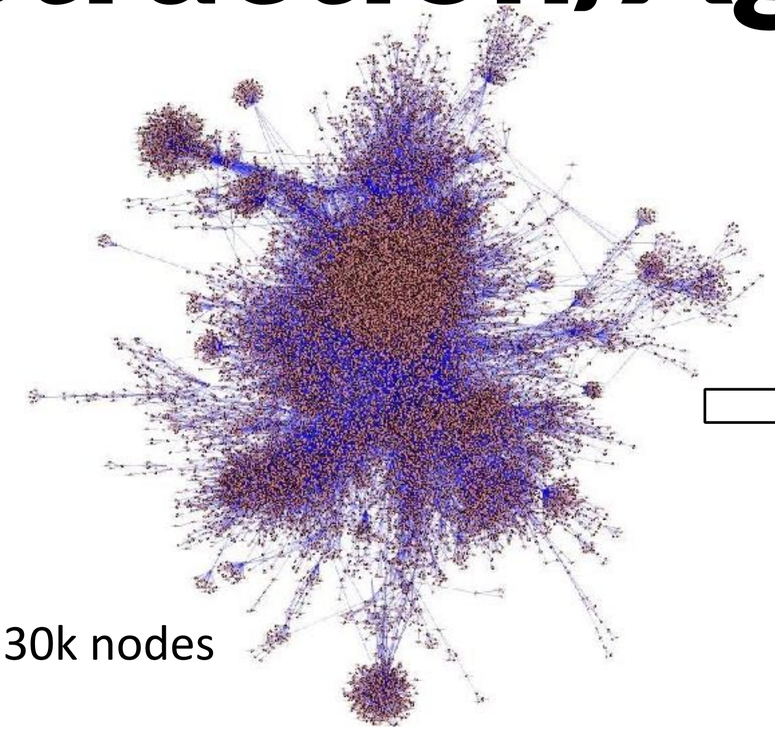
Adress Computational Scalability: Multilevel Approaches



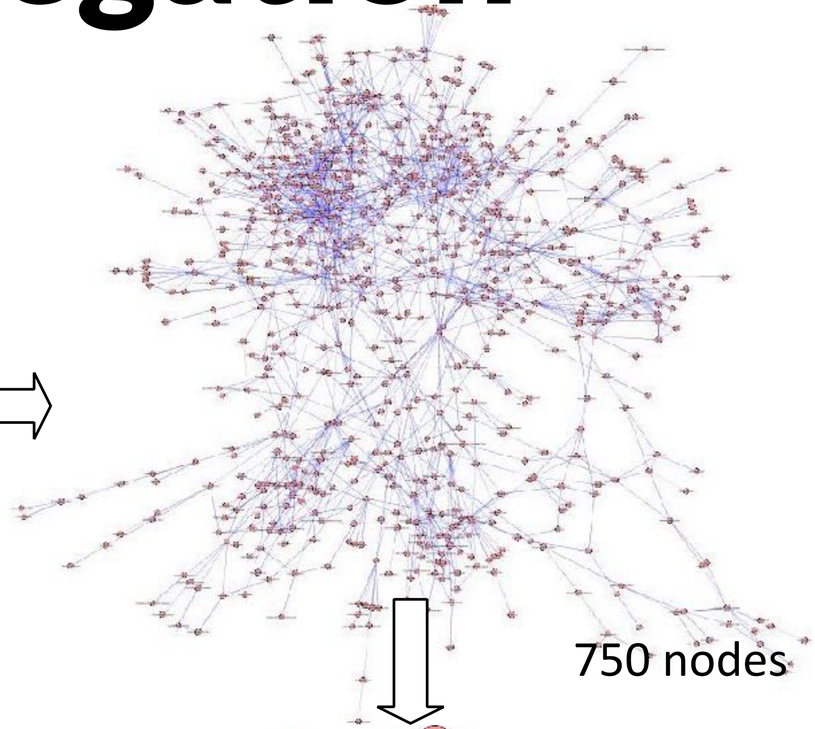


[Schulz 2004]

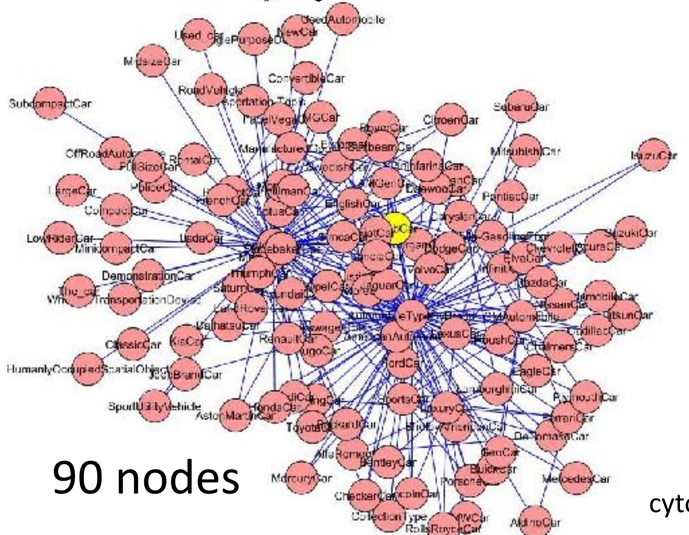
Abstraction/Aggregation



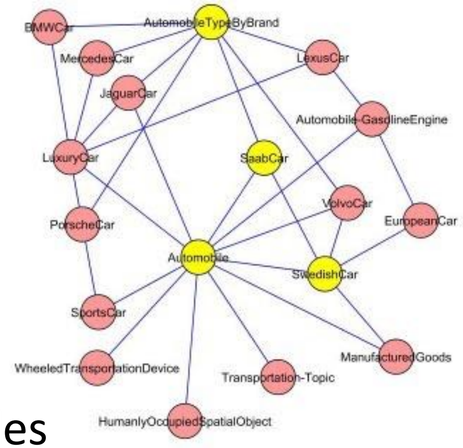
30k nodes



750 nodes



90 nodes



18 nodes

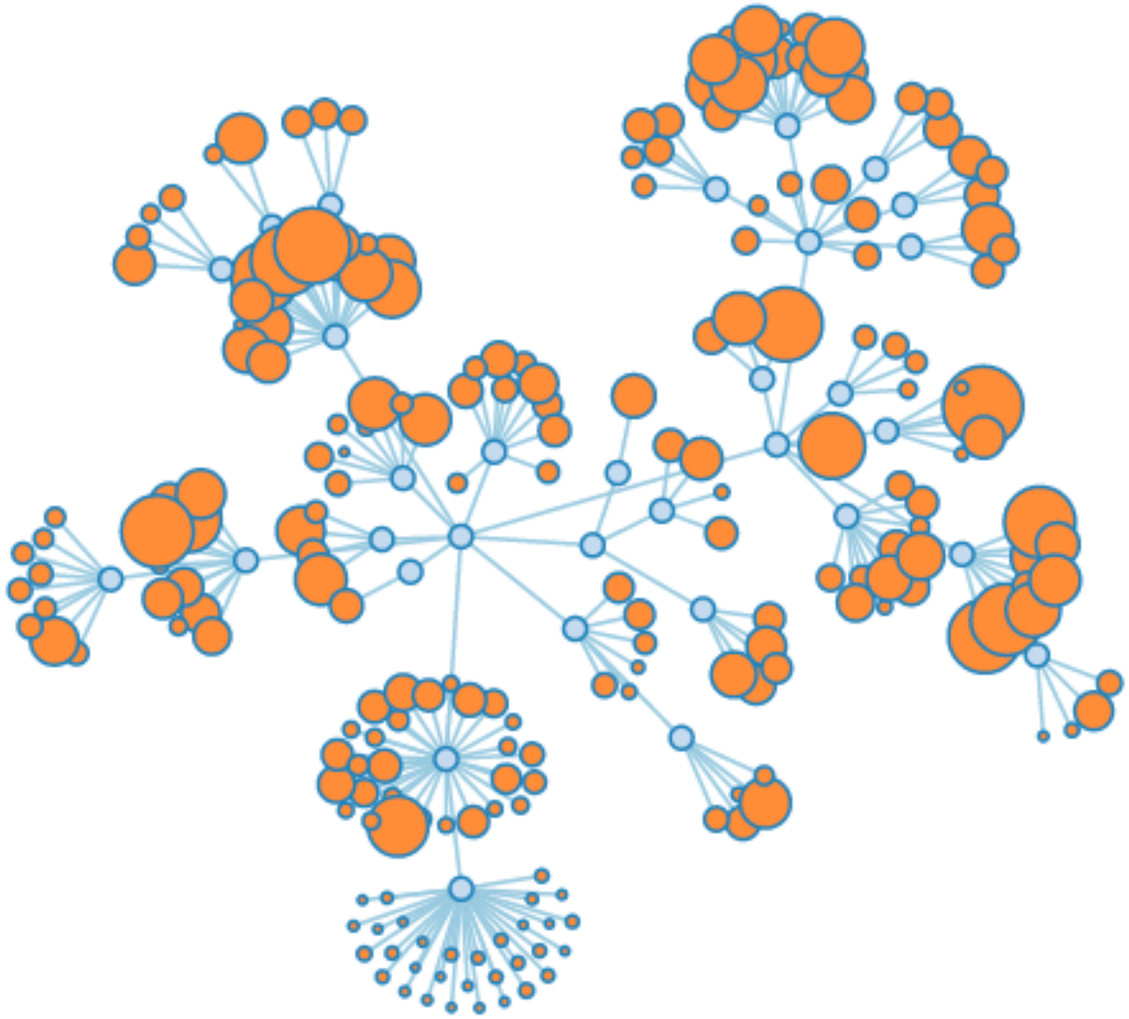
cytoscape.org

Collapsible Force Layout

Supernodes: aggregate of nodes

manual or algorithmic

- clustering

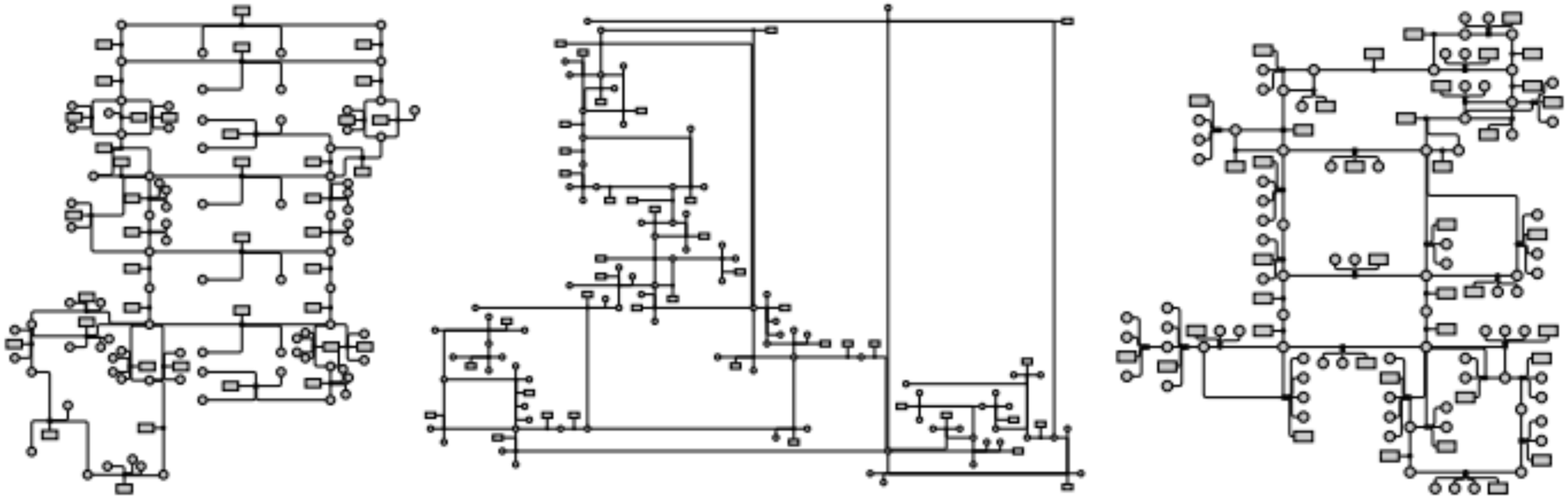


HOLA: Human-like Orthogonal Layout

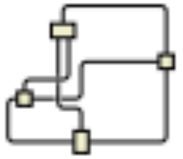
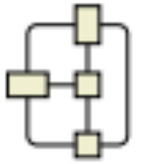
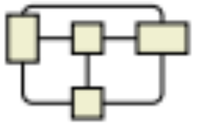
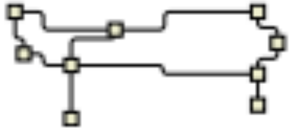
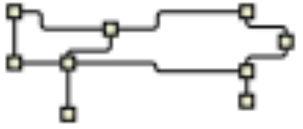
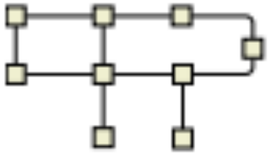
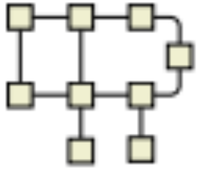
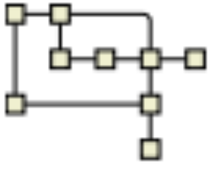
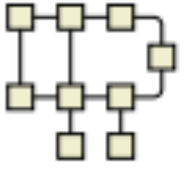
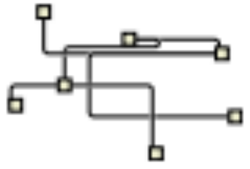
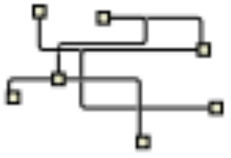
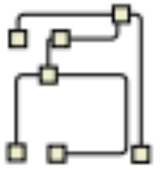
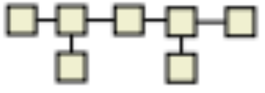
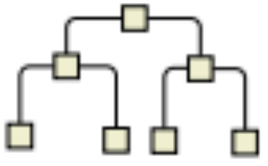
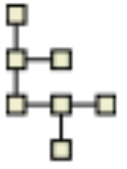
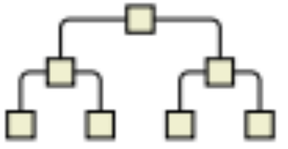
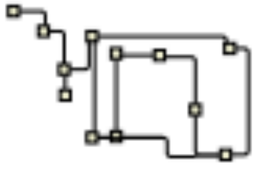
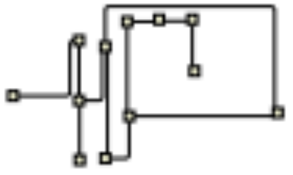
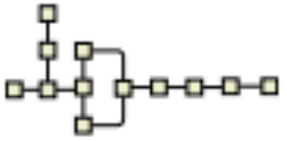
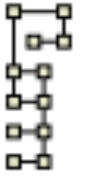
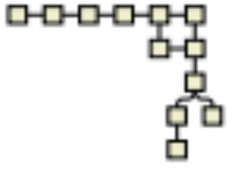
Study how humans lay-out a graph

Try to emulate layout

Left: human, middle: conventional algo, right new algo



[Kieffer et al, InfoVis 2015]

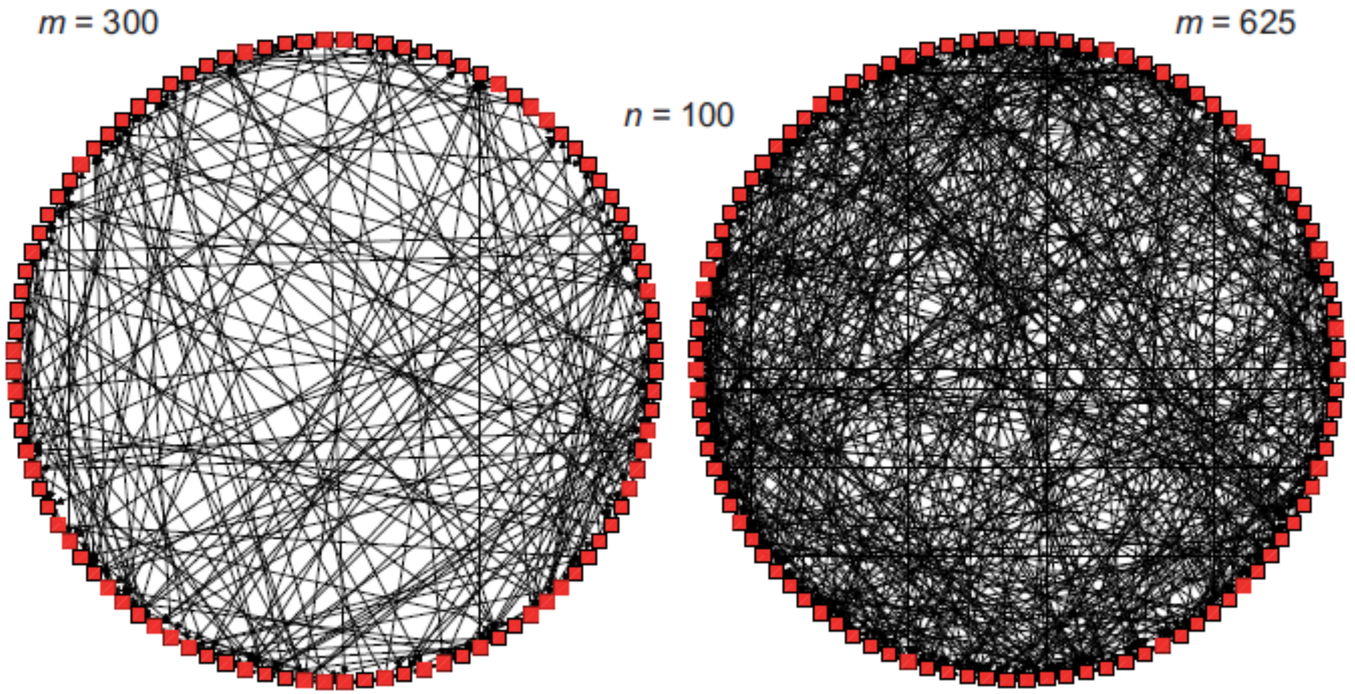
	Initial	Human 17th	Human 16th	Human 2nd	Human 1st	yFiles	HOLA
Graph 1							
	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.51$	$\bar{\mu}_1 = 0.53$, $\bar{\mu}_2 = 0.61$	$\bar{\mu}_1 = 0.51, \bar{\mu}_2 = 0.41$	$\bar{\mu}_2 = 0.48$
Graph 2							
	$\bar{\mu}_1 = 0.02$	$\bar{\mu}_1 = 0.02$	$\bar{\mu}_1 = 0.09$	$\bar{\mu}_1 = 0.57$	$\bar{\mu}_1 = 0.58$, $\bar{\mu}_2 = 0.81$	$\bar{\mu}_1 = 0.25, \bar{\mu}_2 = 0.21$	$\bar{\mu}_2 = 0.49$
Graph 3							
	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.59$	$\bar{\mu}_1 = 0.69$, $\bar{\mu}_2 = 0.52$	$\bar{\mu}_1 = 0.33, \bar{\mu}_2 = 0.10$	$\bar{\mu}_2 = 0.88$
Graph 4							
	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.00$	$\bar{\mu}_1 = 0.58$	$\bar{\mu}_1 = 0.59$, $\bar{\mu}_2 = 0.59$	$\bar{\mu}_1 = 0.21, \bar{\mu}_2 = 0.11$	$\bar{\mu}_2 = 0.80$
Graph 5							

Styled / Restricted Layouts

Circular Layout

Node ordering

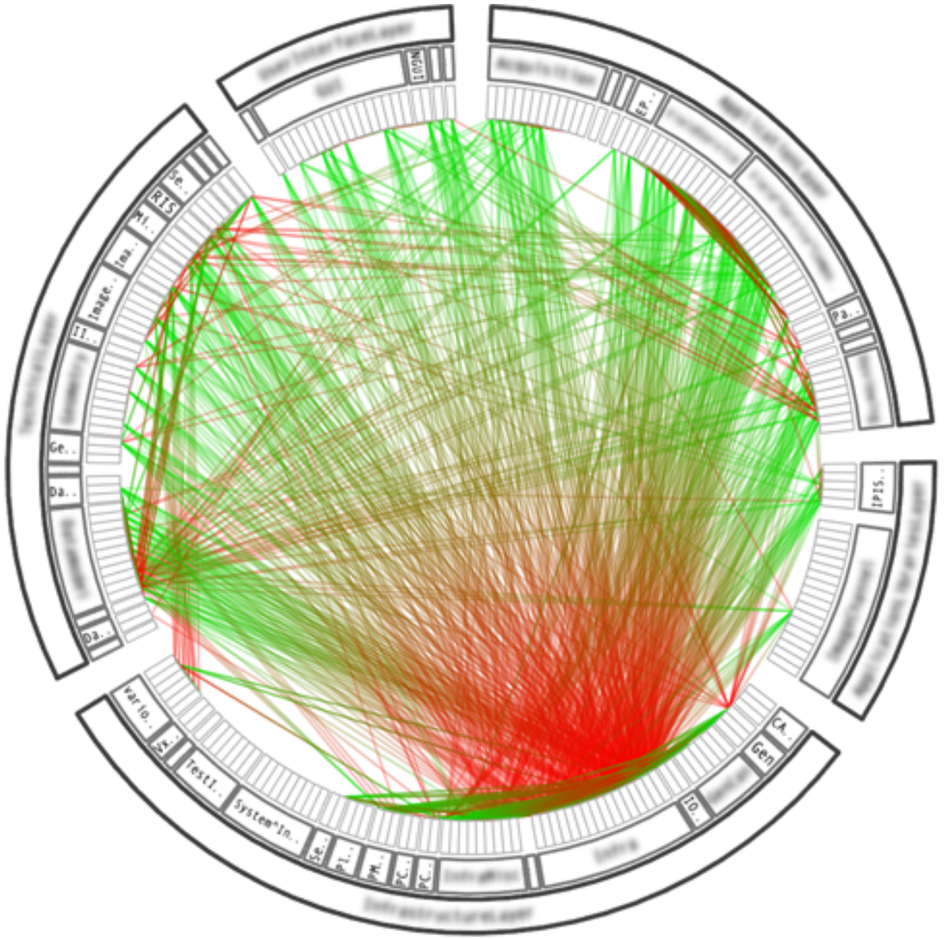
Edge Clutter

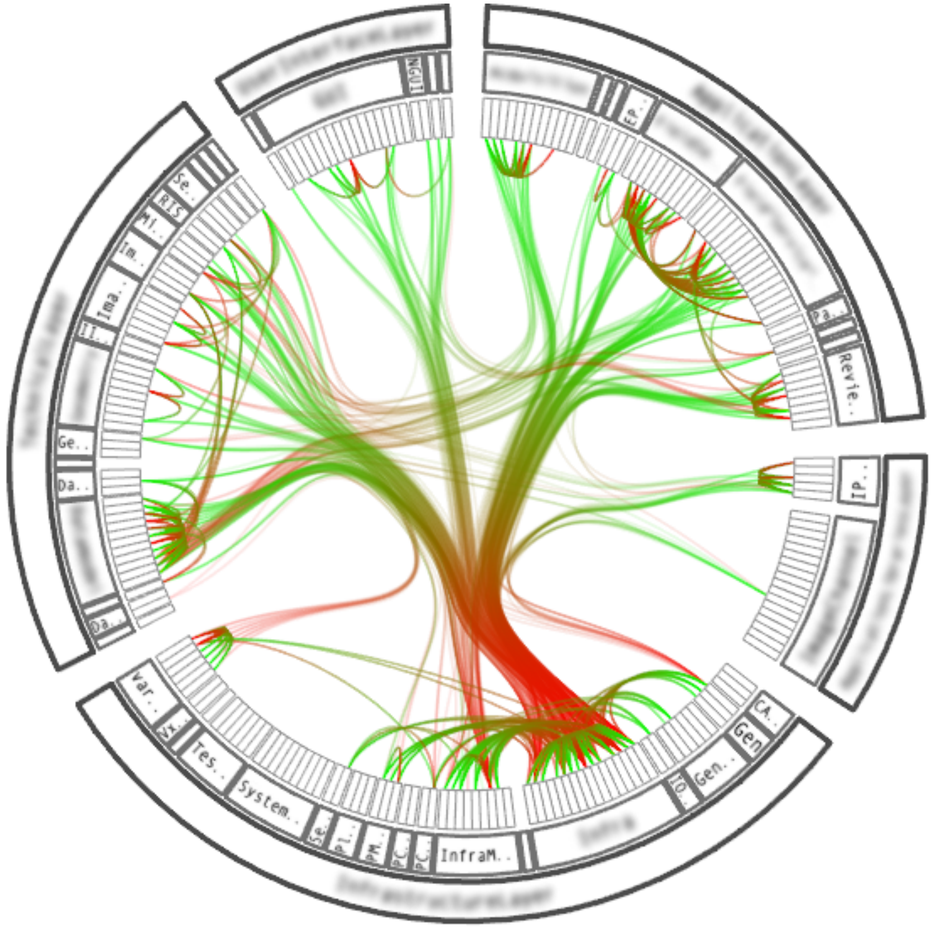


ca. 3% of all possible edges

ca. 6,3% of all possible edges

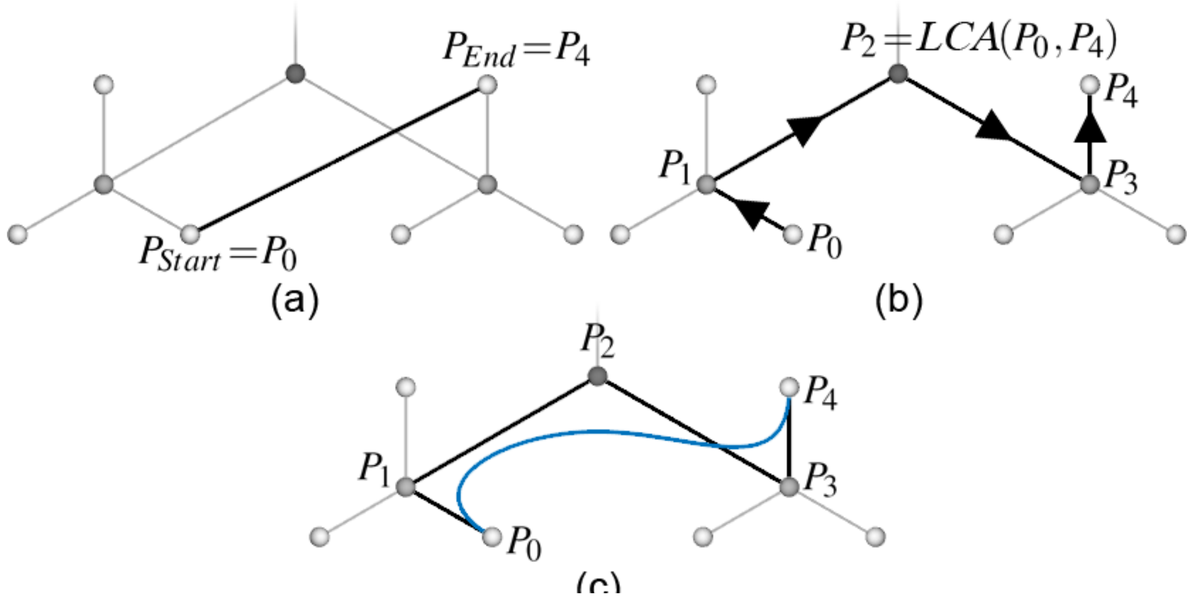
Reduce Clutter: Edge Bundling

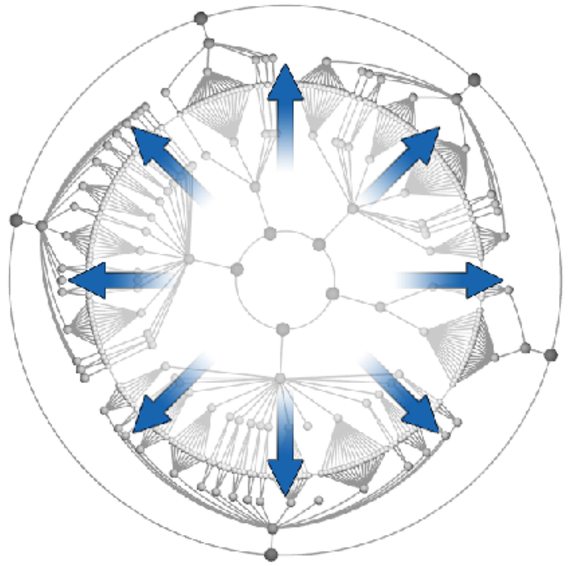


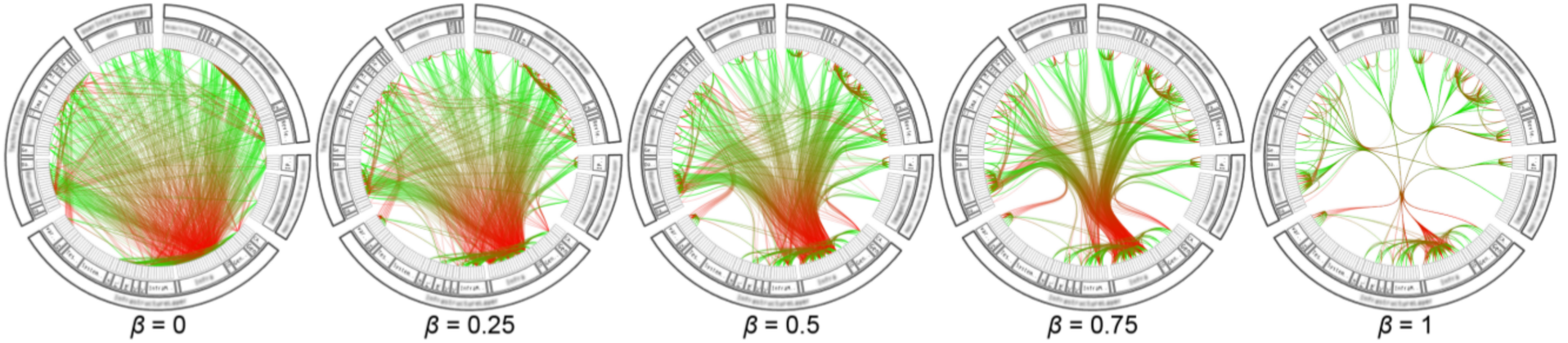


Holten et al. 2006

Hierarchical Edge Bundling



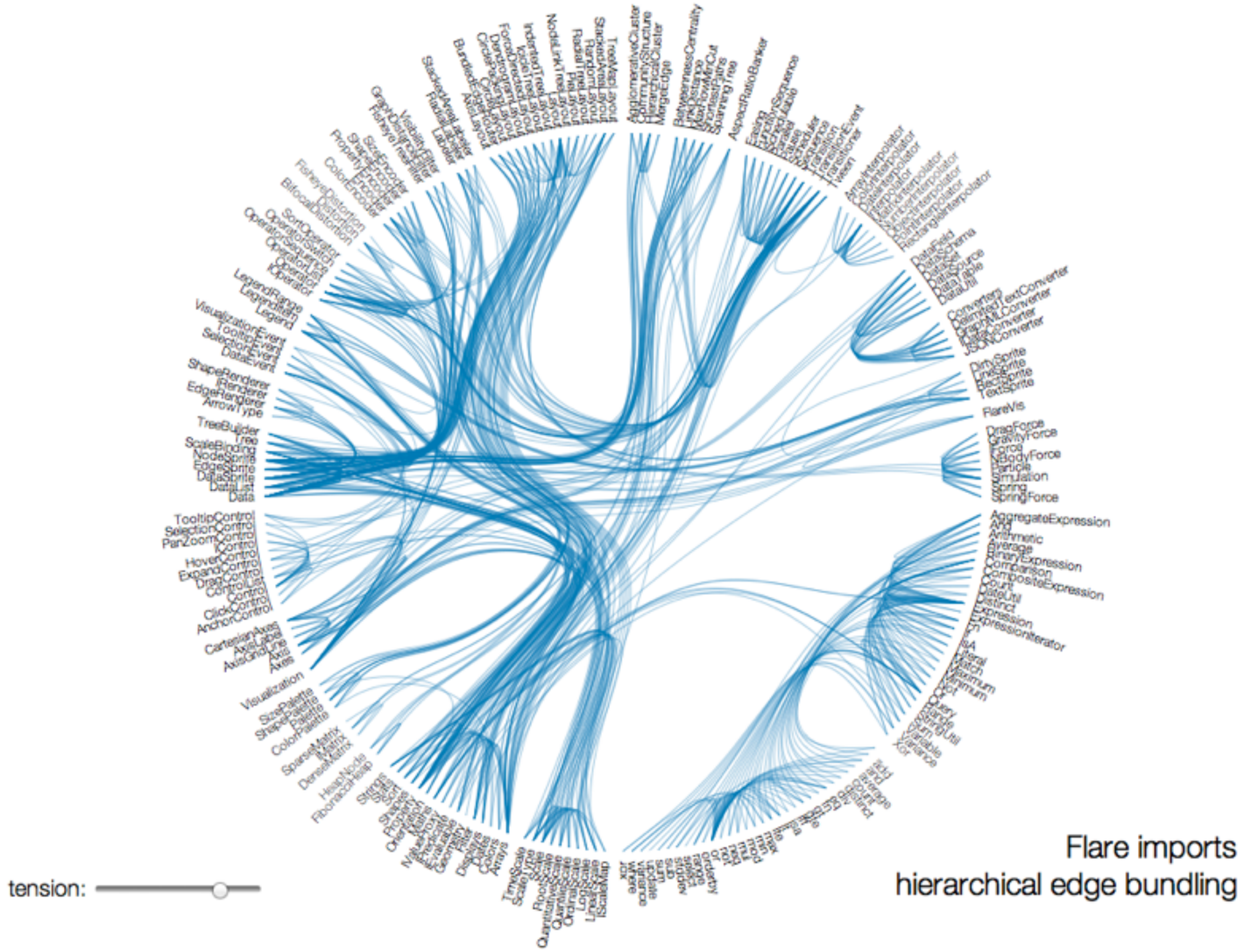




Bundling Strength

Holten et al. 2006

Bundling Strength



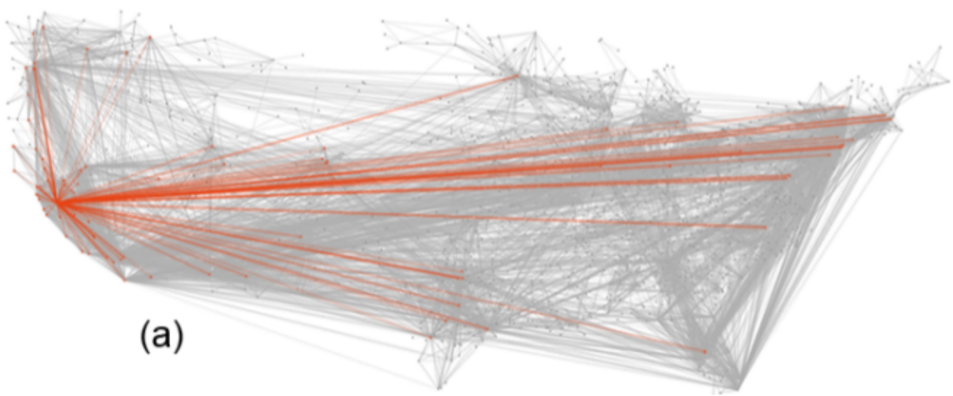
mbostock.github.com/d3/talk/20111116/bundle.html

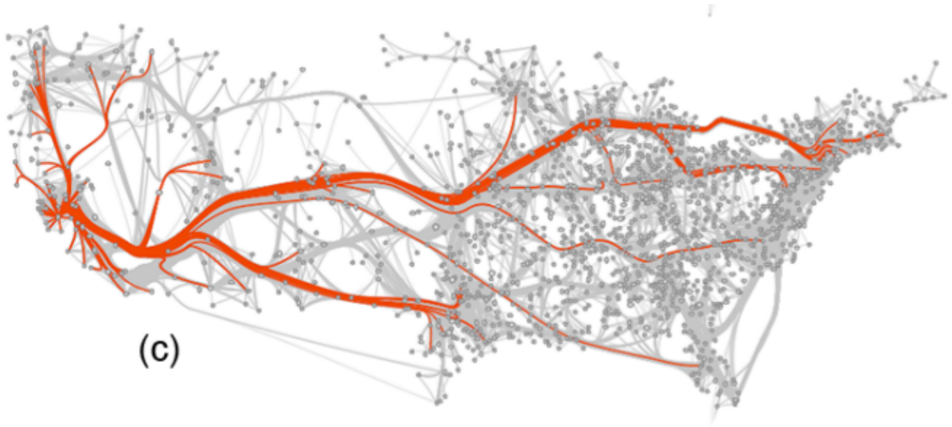
Michael Bostock

Fixed Layouts

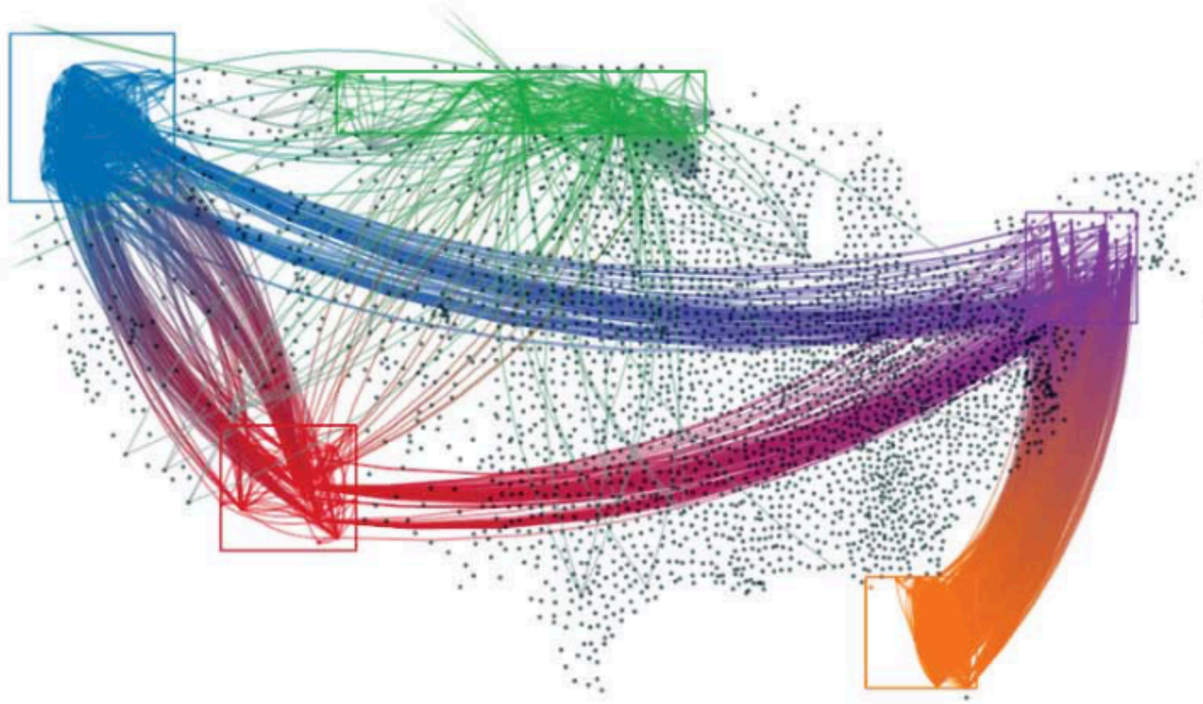
Can’t vary position of nodes

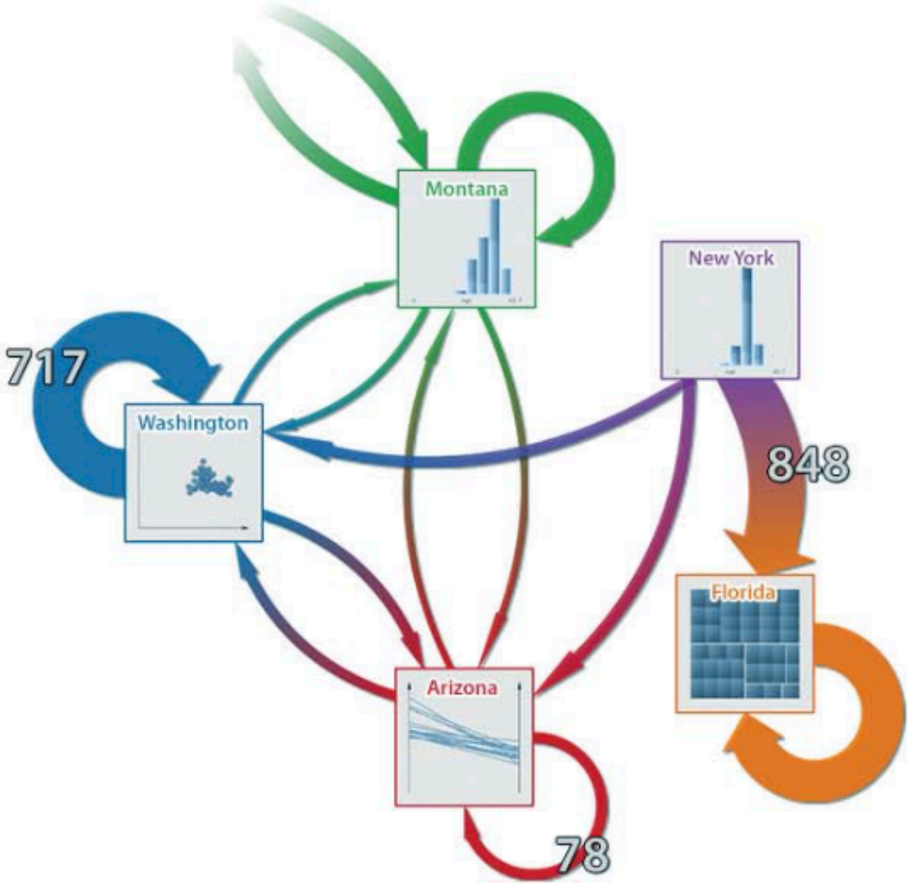
Edge routing important





Aggregation



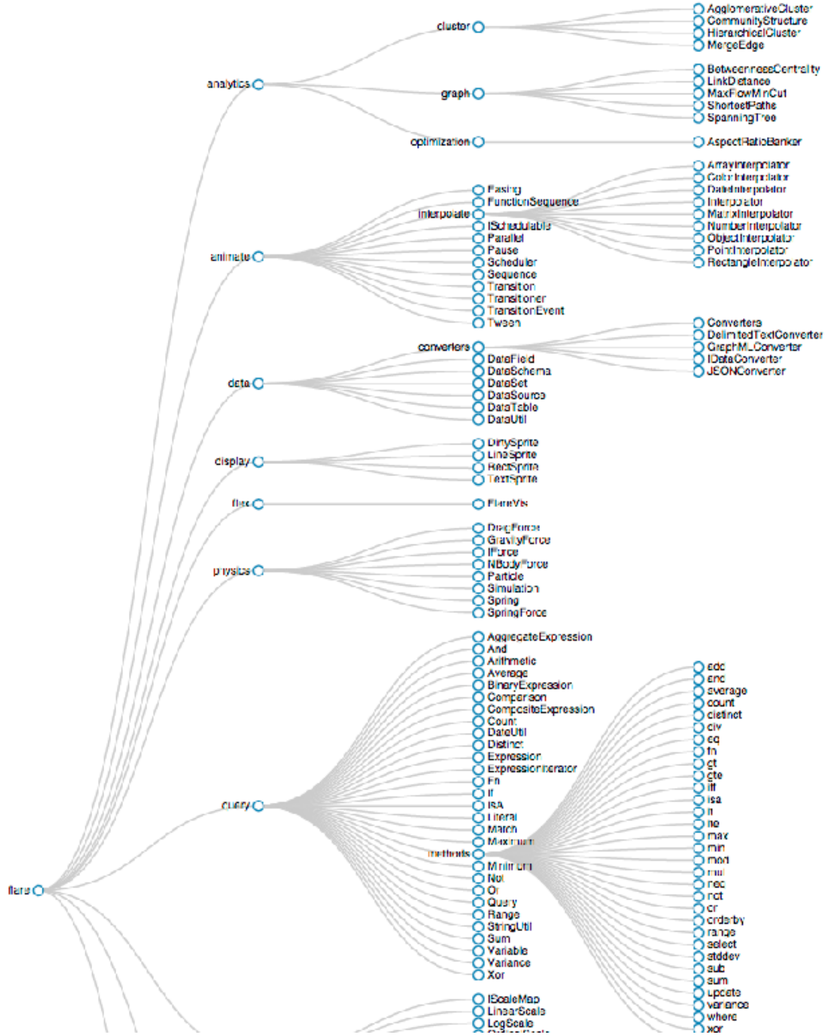


https://www.youtube.com/watch?v=E1PVTitj7h0

Explicit Tree Visualization

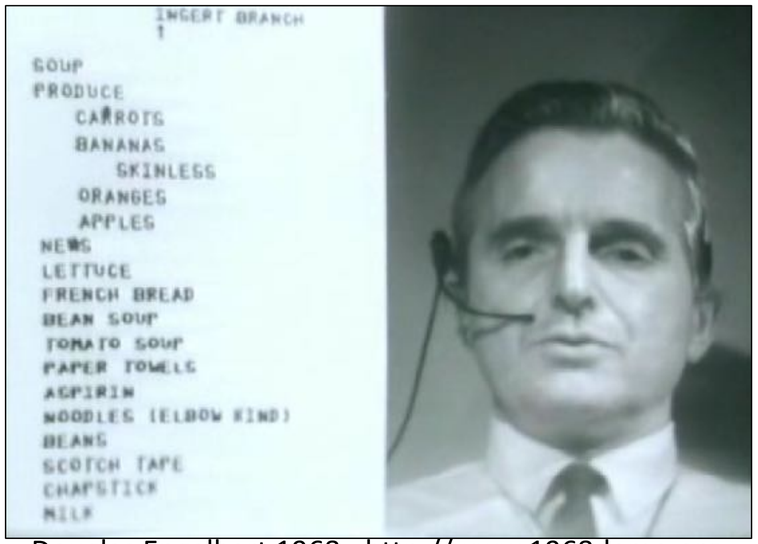
Reingold–
Tilford layout

http://billmill.org/pymag-trees/

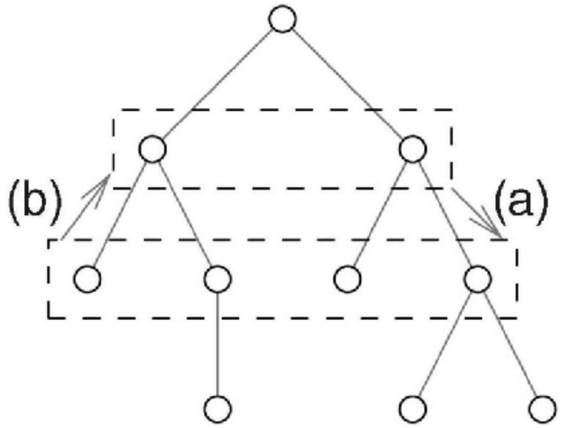


Manipulating Aggregation Levels

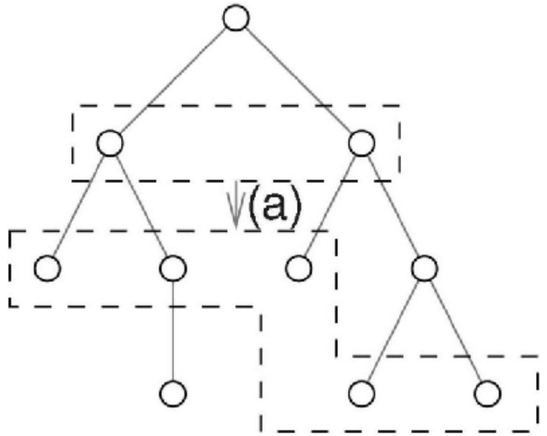
First interactive tree manipulation



Douglas Engelbart 1968 - http://www.1968demo.org



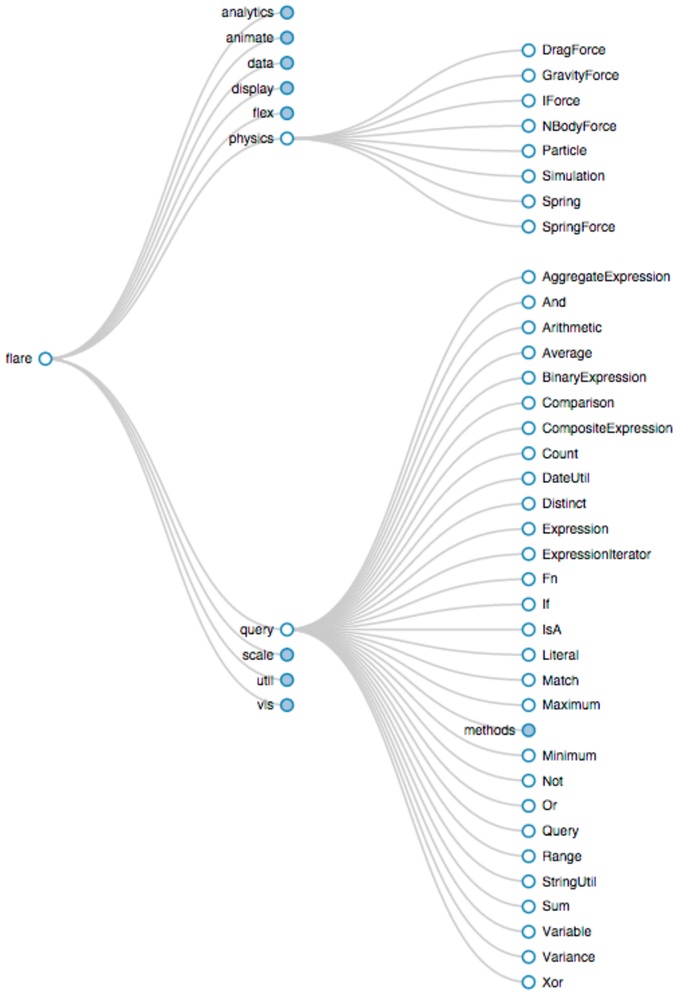
(a) Drill-Down (b) Roll-Up

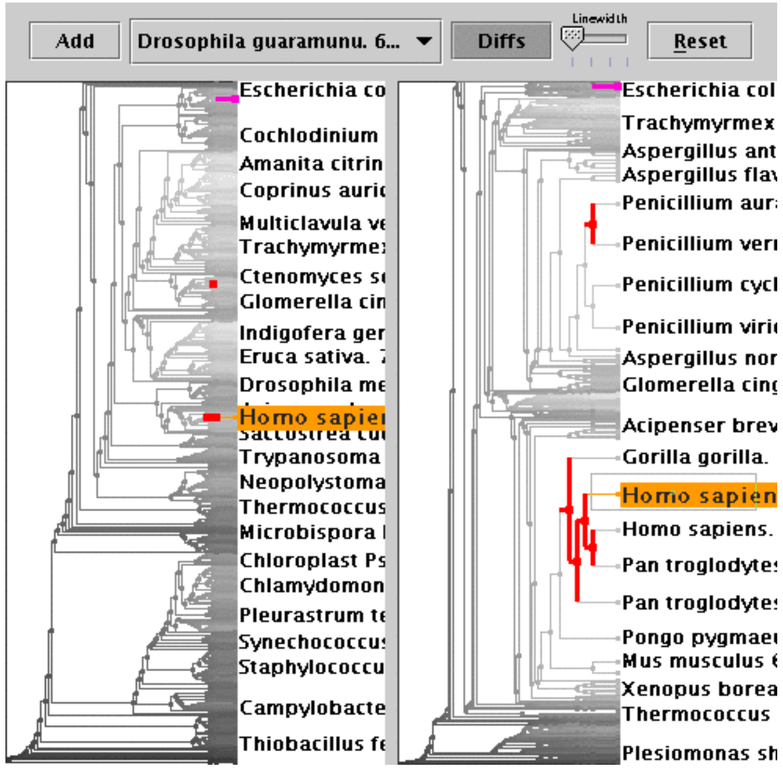


(a) Unbalanced Drill-Down

“The mother of all demos“
https://www.youtube.com/watch?v=yJDv-zdhzMY

Tree Interaction, Tree Comparison





Explicit Representations

Pros:

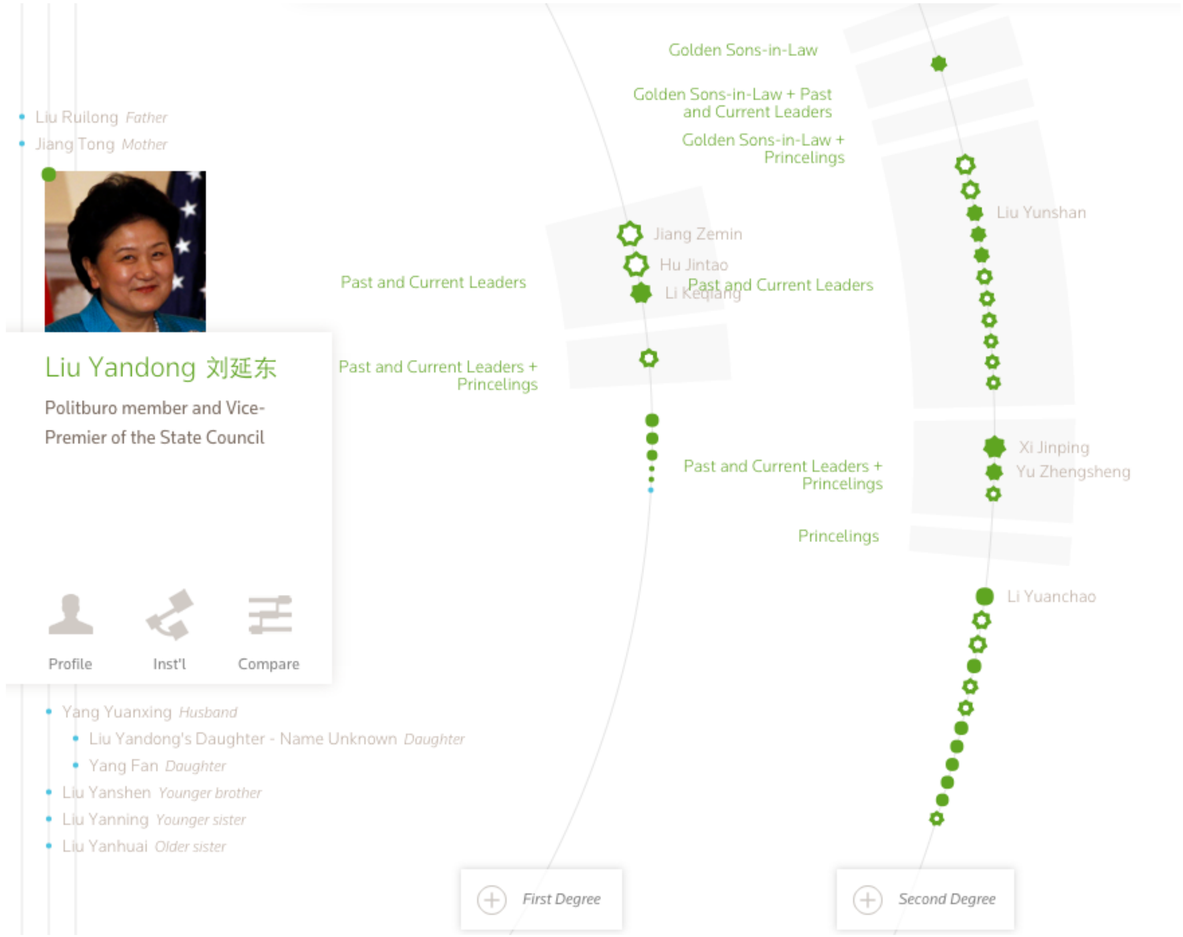
- is able to depict all graph classes
- can be customized by weighing the layout constraints
- very well suited for TBTs, if also a suitable layout is chosen

Cons:

- computation of an optimal graph layout is in NP
  (even just achieving minimal edge crossings is already in NP)
- even heuristics are still slow/complex (e.g., naïve spring embedder is in $O(n^2)$)
- has a tendency to clutter (edge clutter, “hairball”)

Design Critique

Connected China



https://goo.gl/YXkWYX

http://china.fathom.info/

Multivariate Graphs

Networks and Attributes

Attributes can influence topology

Path can be slow / blocked

- best route when driving depends on traffic
- biological network depends on many factors

Challenge: Data Scale & Heterogeneity

Large number of values

- Large datasets have more than 500 experiments

Multiple groups/conditions

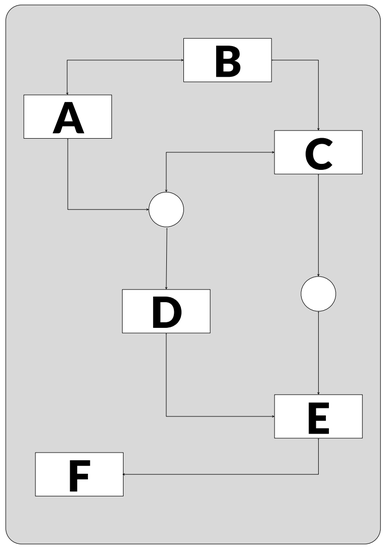
Different types of data

Challenge: Supporting Multiple Tasks

Two central tasks:

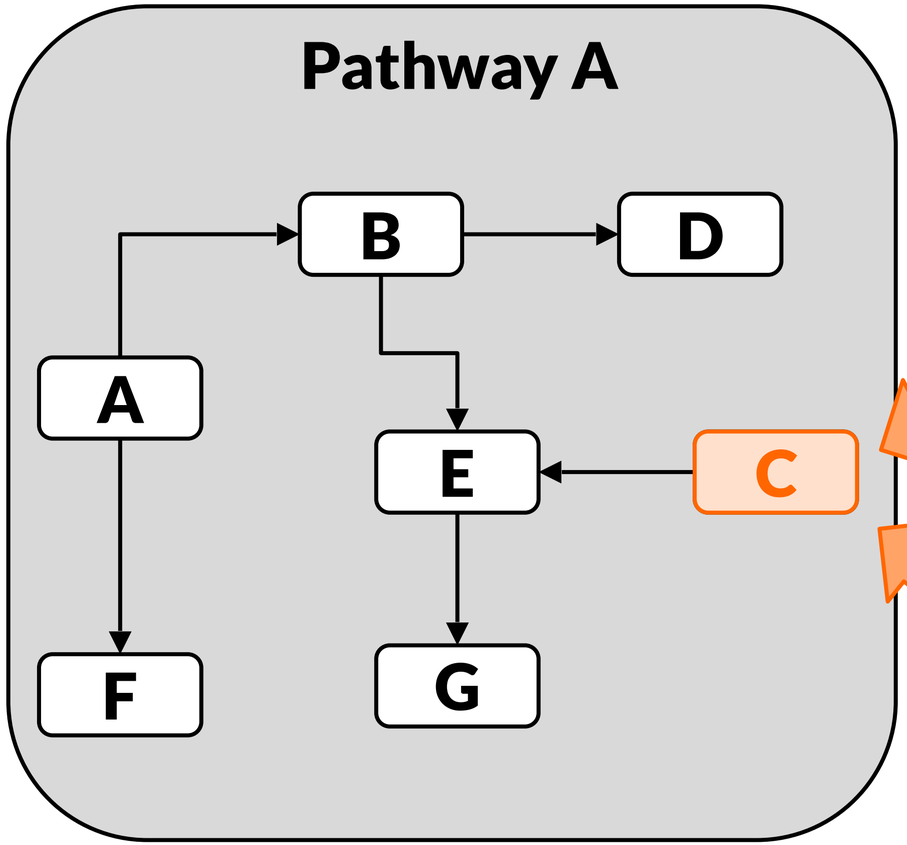
- **Explore topology of network**
- **Explore the attributes of the nodes (experimental data)**

Need to support both!



	Sample 1	Sample 2	Sample 3
Gene 1	1	1.1	0.4
Gene 2	2	0.5	1.2
Gene 3	1.4	0.2	0.5
Gene 4	0.3	0.5	0.7

Many Node Attributes

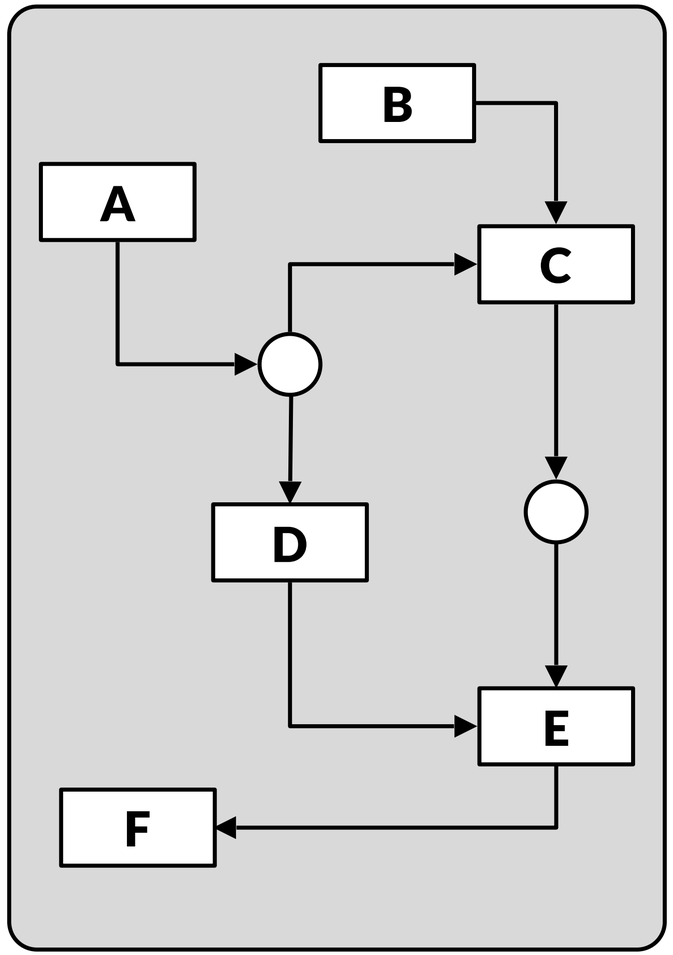


Node	Sample 1	Sample 2	Sample 3	...
A	0.55	0.95	0.83	...
B	0.12	0.42	0.16	...
C	0.33	0.65	0.38	...
...	...	...	...	

Node	Sample 1	Sample 2	Sample 3	...
A	low	low	very high	...
B	normal	low	high	...
C	high	very low	normal	...
...	...	...	...	

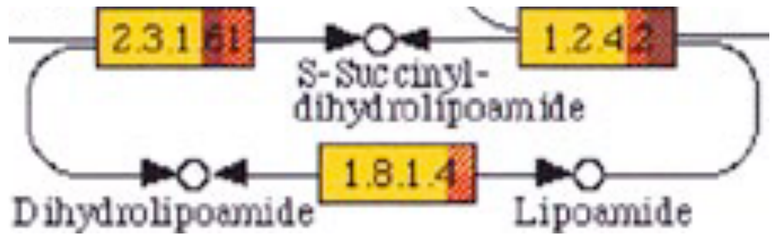
How to visualize attribute data on networks?

Good Old Color Coding



A	-3.4	4.2	5.1	4.2
B	2.8	1.8	1.3	1.1
C	3.1	-2.2	2.4	2.2
D	-3	-2.8	1.6	1.0
E	0.5	0.3	-1.1	1.3
F	0.3	0.3	1.8	-0.3

[Lindroos2002]

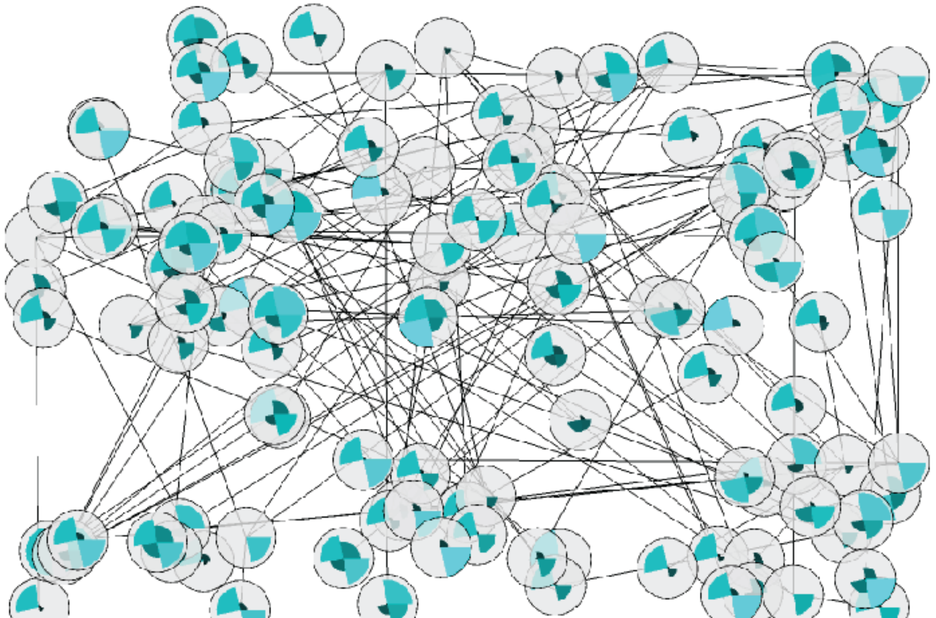


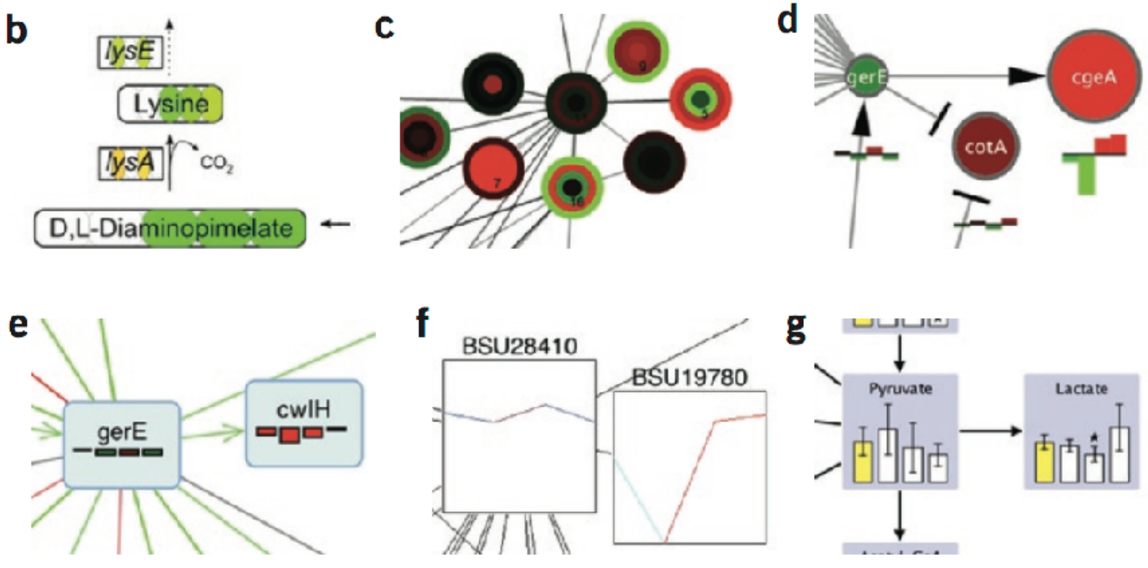
Node Attributes

Coloring

Glyphs

-> Limited in scalability

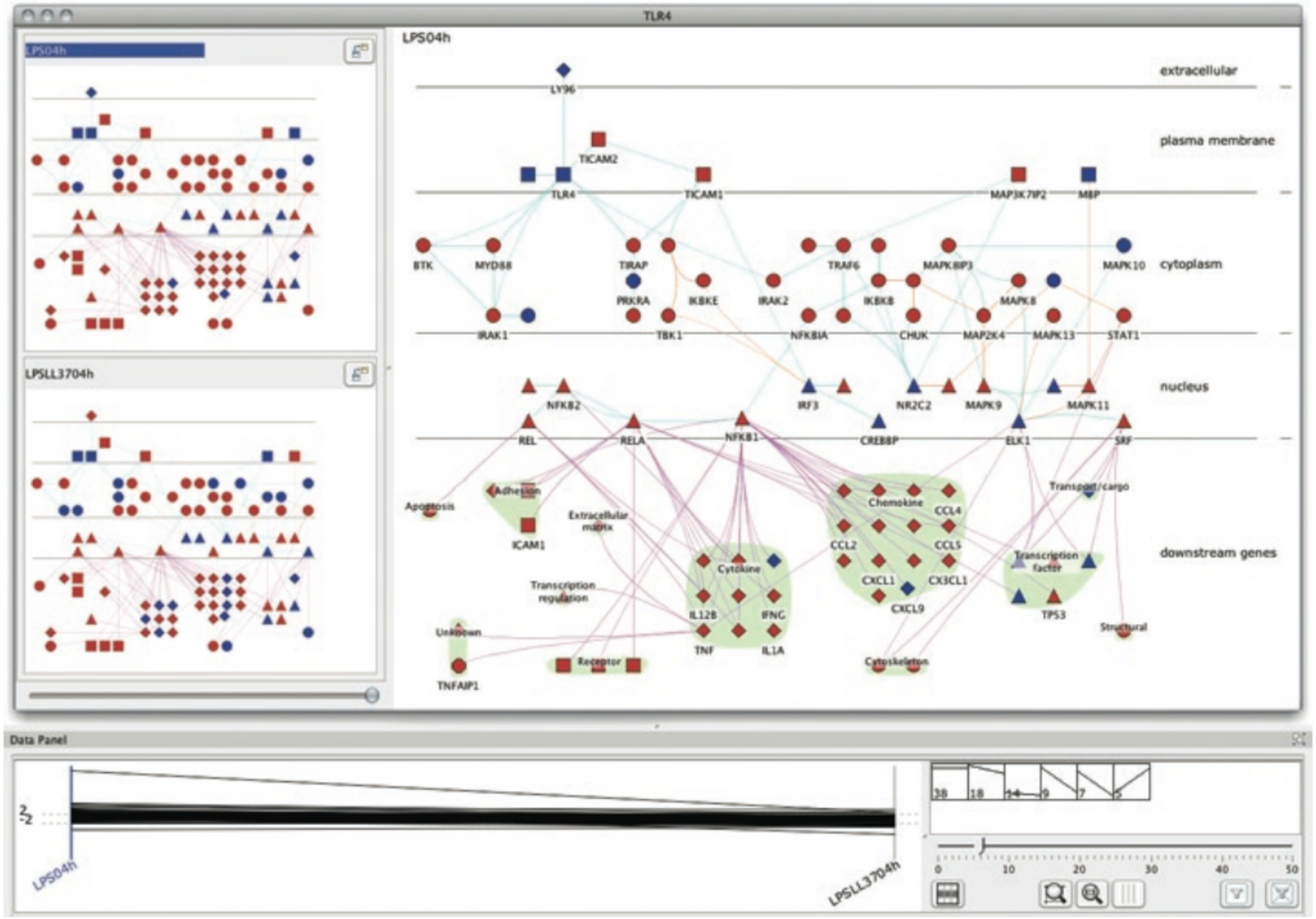




Small Multiples

Cerebral [Barsky, 2008]

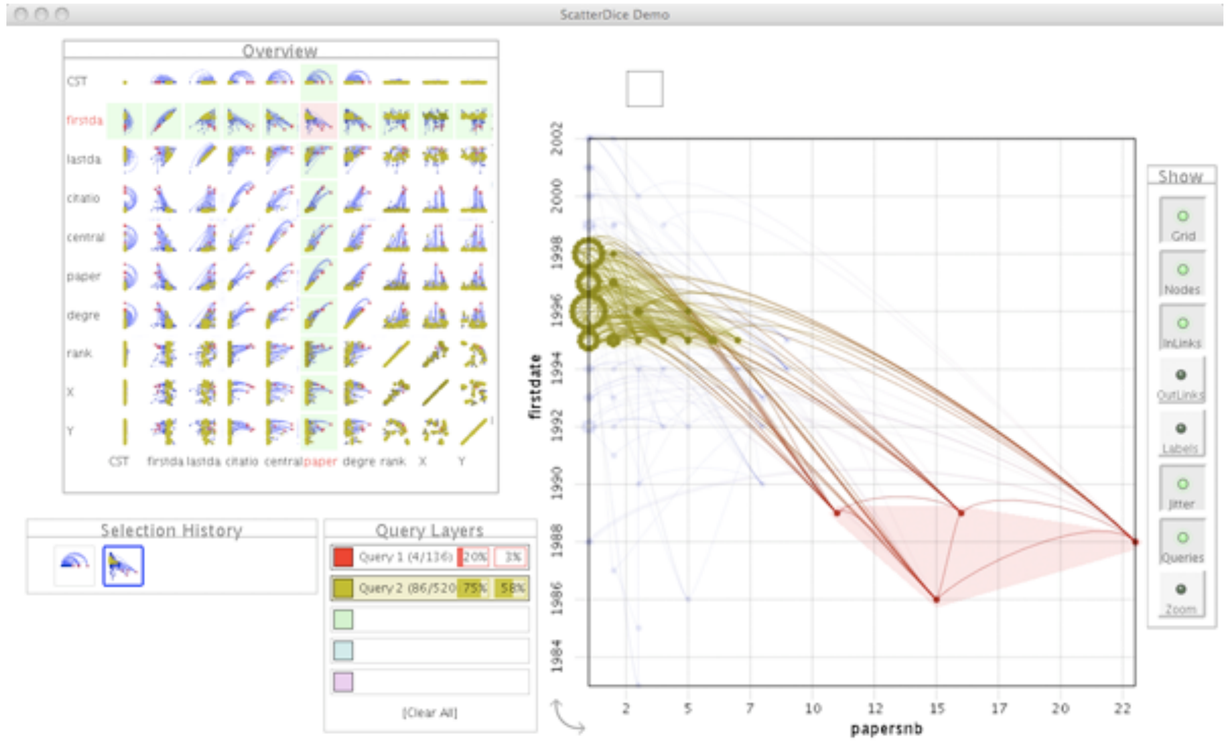
Each dimension in its own window



Data-driven node positioning

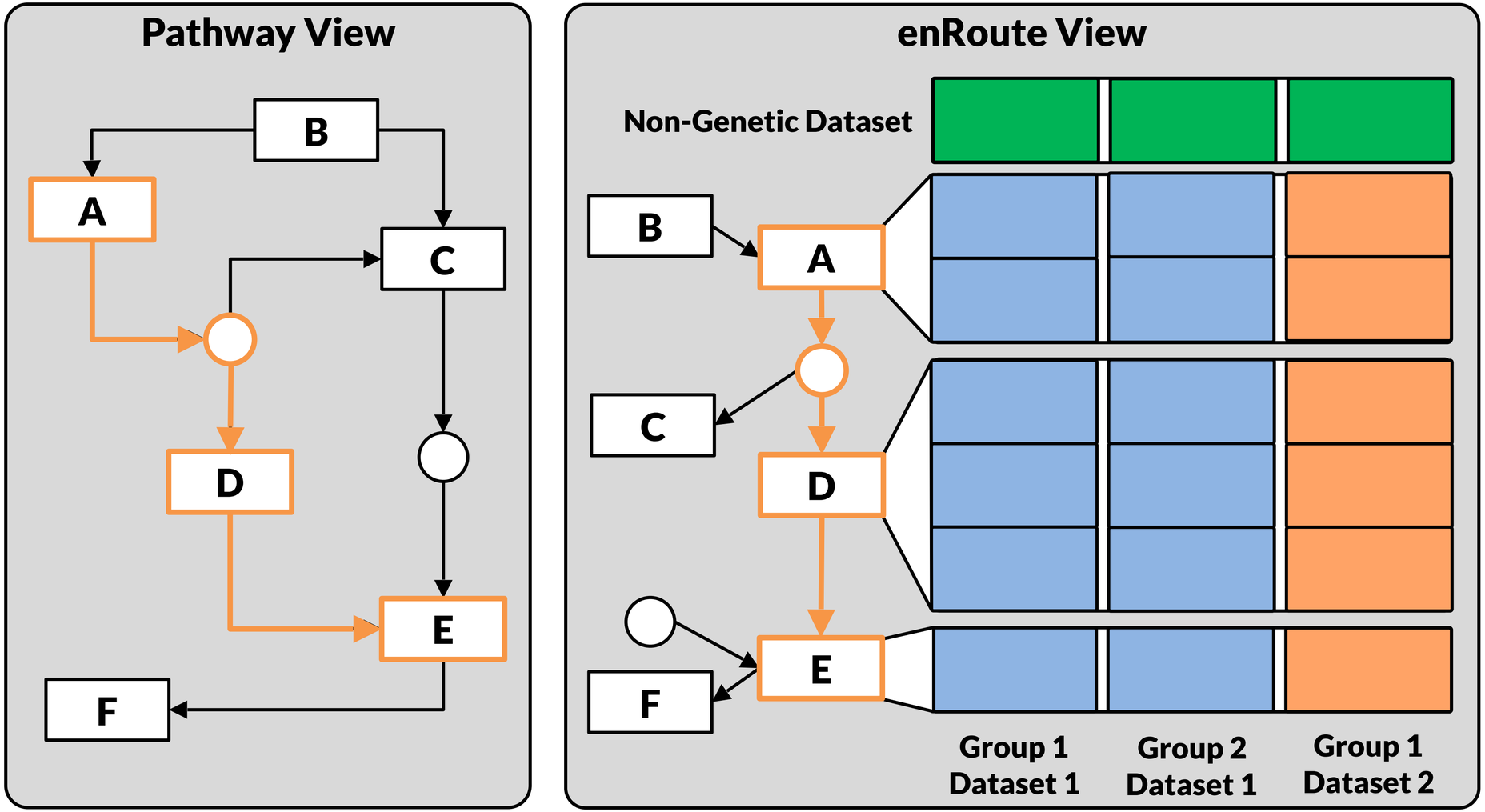
GraphDice

Nodes are laid out according to attribute values

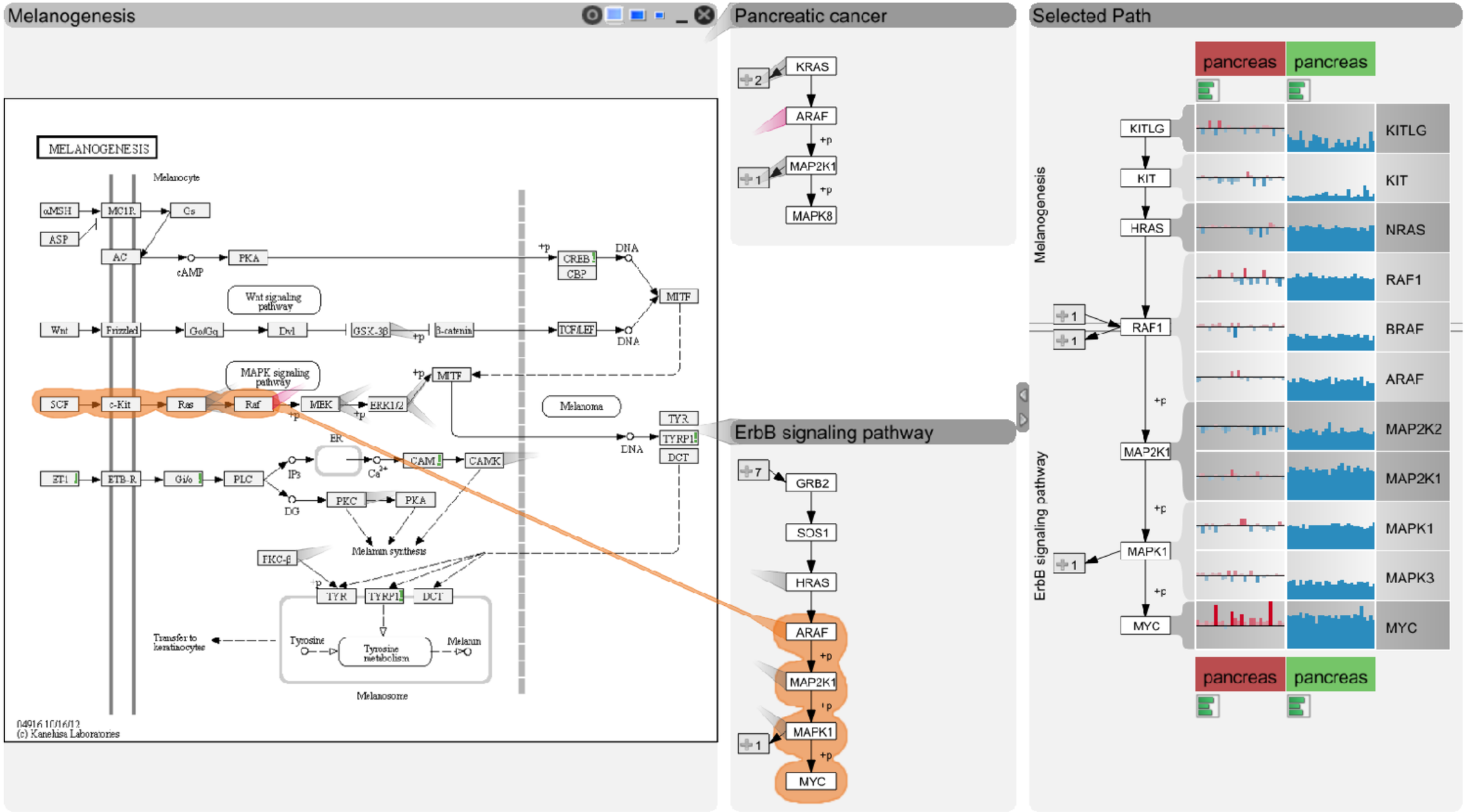


[Bezerianos et al, 2010]

Path Extraction: enRoute



enRoute


Melanogenesis
MELANOGENESIS
Melanocyte
αMSH
MC1R
Gs
ASP
AC
cAMP
PKA
CREB
CBP
DNA
MITF
Wnt signaling pathway
Wnt
Frizzled
Gq/G11
Dvl
GSK-3β
β-catenin
TCF/LEF
MAPK signaling pathway
SCF
c-Kit
Ras
Raf
MEK
ERK1/2
Melanoma
TYR
TYRP1
DCT
ET1
ETB-R
Gi/o
PLC
IP3
DG
ER
CAM
CAMK
PKC
PKA
Melanin synthesis
FKC-β
Transfer to keratinocytes
Tyrosine
Tyrosine metabolism
Melanin
Melanosome
04916 10/16/12
(c) Kanehisa Laboratories
Pancreatic cancer
KRAS
ARAF
MAP2K1
MAPK8
ErbB signaling pathway
GRB2
SOS1
HRAS
ARAF
MAP2K1
MAPK1
MYC
Selected Path
pancreas
pancreas
KITLG
KIT
HRAS
RAF1
MAP2K1
MAPK1
MYC
Melanogenesis
ErbB signaling pathway
KITLG
KIT
NRAS
RAF1
BRAF
ARAF
MAP2K2
MAP2K1
MAPK1
MAPK3
MYC
pancreas
pancreas

File Data Window View Help

Entourage

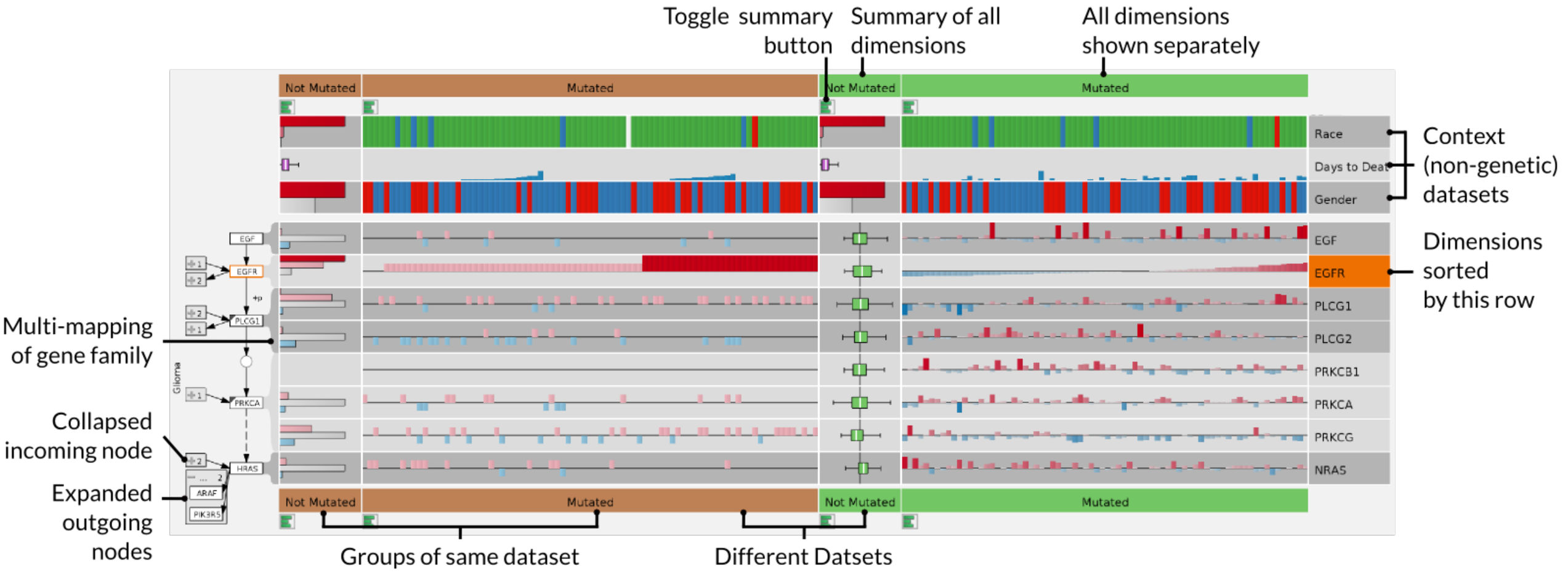
Pathways

Pathway

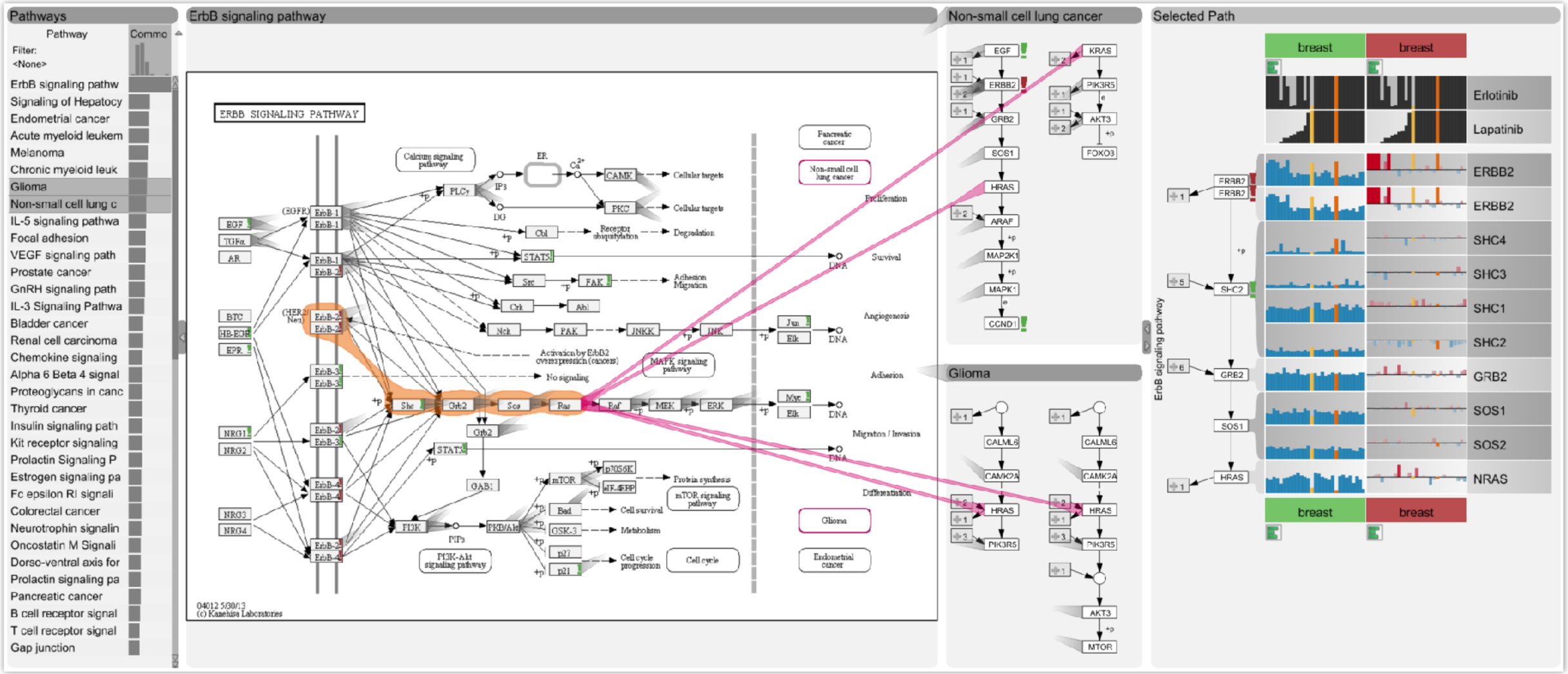
Filter:
<None>

1 C donor
2-Oxocarboxylic acid
ABC transporters
ABC-family proteins
ACE Inhibitor Pathwa
Acetylcholine Synthes
Acute myeloid leukem
Adherens junction
Adipocyte TarBase
Adipocytokine signali
Adipogenesis
Advanced glycosylatio
Aflatoxin B1 metaboli
African trypanosomias
AGE/RAGE pathway
AhR pathway
Alanine and aspartate
Alanine, aspartate an
Alcoholism
Aldosterone-regulated
Allograft rejection
Allograft rejection
Alpha 6 Beta 4 signal
alpha-Linolenic acid
Alzheimer's disease
Alzheimers Disease
amino acid conjugatio
amino acid conjugatio
Amino sugar and nucl
Aminoacyl-tRNA bios
Amoebiasis
Amphetamine addicti
AMPK signaling
Amyotrophic lateral sc
Androgen receptor si
Angiogenesis
Angiogenesis
angiogenesis overvie
Antigen processing an
APC/C-mediated degra
Apoptosis
Apoptosis
Apoptosis Meta Path
Apoptosis Modulation
Apoptosis Modulation
Apoptosis, anoikis an

Selected Path


Toggle summary button
Summary of all dimensions
All dimensions shown separately
Not Mutated
Mutated
Not Mutated
Mutated
Race
Days to Death
Gender
Context (non-genetic) datasets
EGF
EGFR
PLCG1
PLCG2
PRKCB1
PRKCA
PRKCG
NRAS
Dimensions sorted by this row
Multi-mapping of gene family
Collapsed incoming node
Expanded outgoing nodes
Glioma
EGF
EGFR
PLCG1
PRKCA
HRAS
ARAF
PIK3R5
Not Mutated
Mutated
Not Mutated
Mutated
Groups of same dataset
Different Datsets

Case Study: CCLE Data



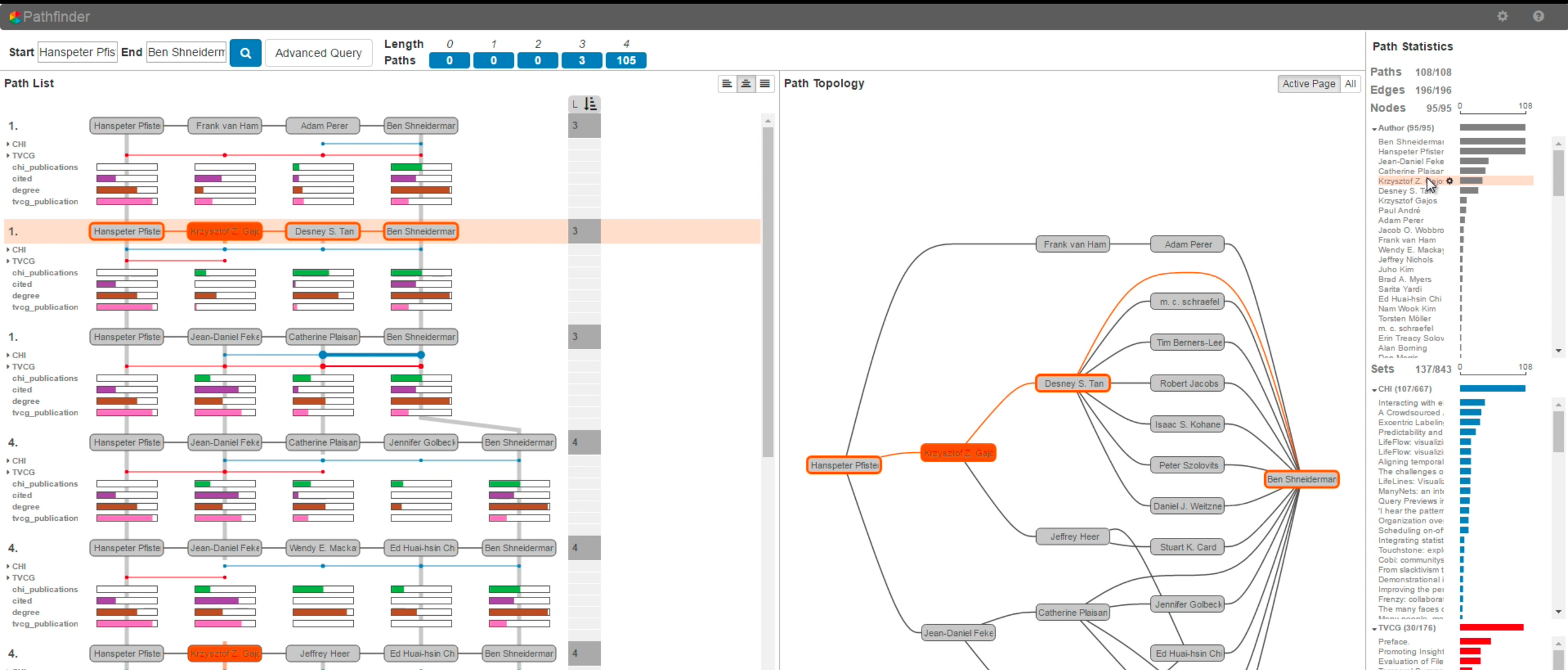

Pancreatic cancer
Non-small cell lung cancer
Proliferation
Survival
Angiogenesis
Adhesion
Migration / Invasion
Differentiation
Glioma
DNA
Jun
Elk
Myc
+p
ERBB2
GRB2
SOS1
HRAS
ARAF
MAP2K1
MAPK1
CCND1
PIK3R5
AKT3
FOXO3
CALML6
CAMK2A
ErbB signaling pathway
SHC2
Erlotinib
Lapatinib
ERBB2
SHC4
SHC3
SHC1
SHC2
GRB2
SOS1
SOS2
NRAS
breast
breast

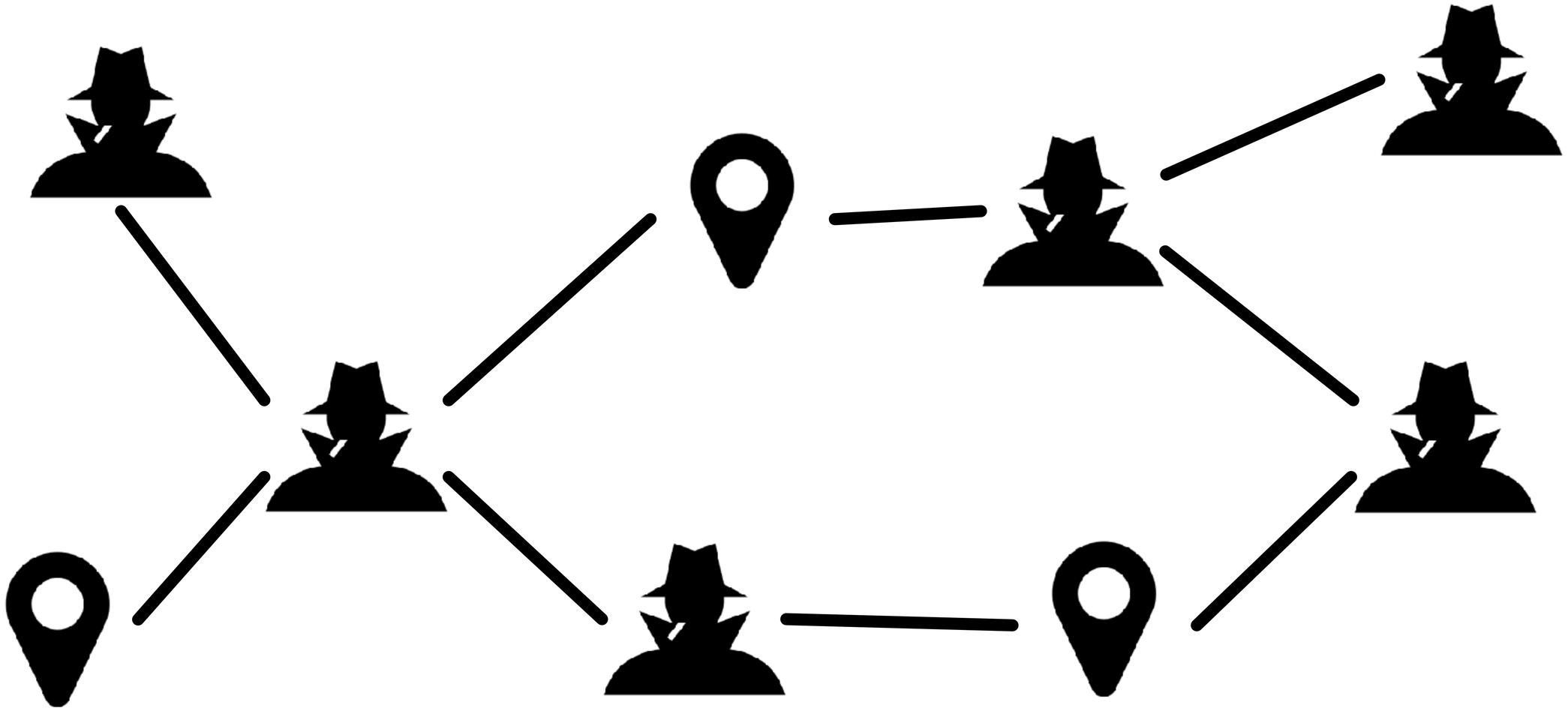
[EuroVis '16]

Honorable Mention Award

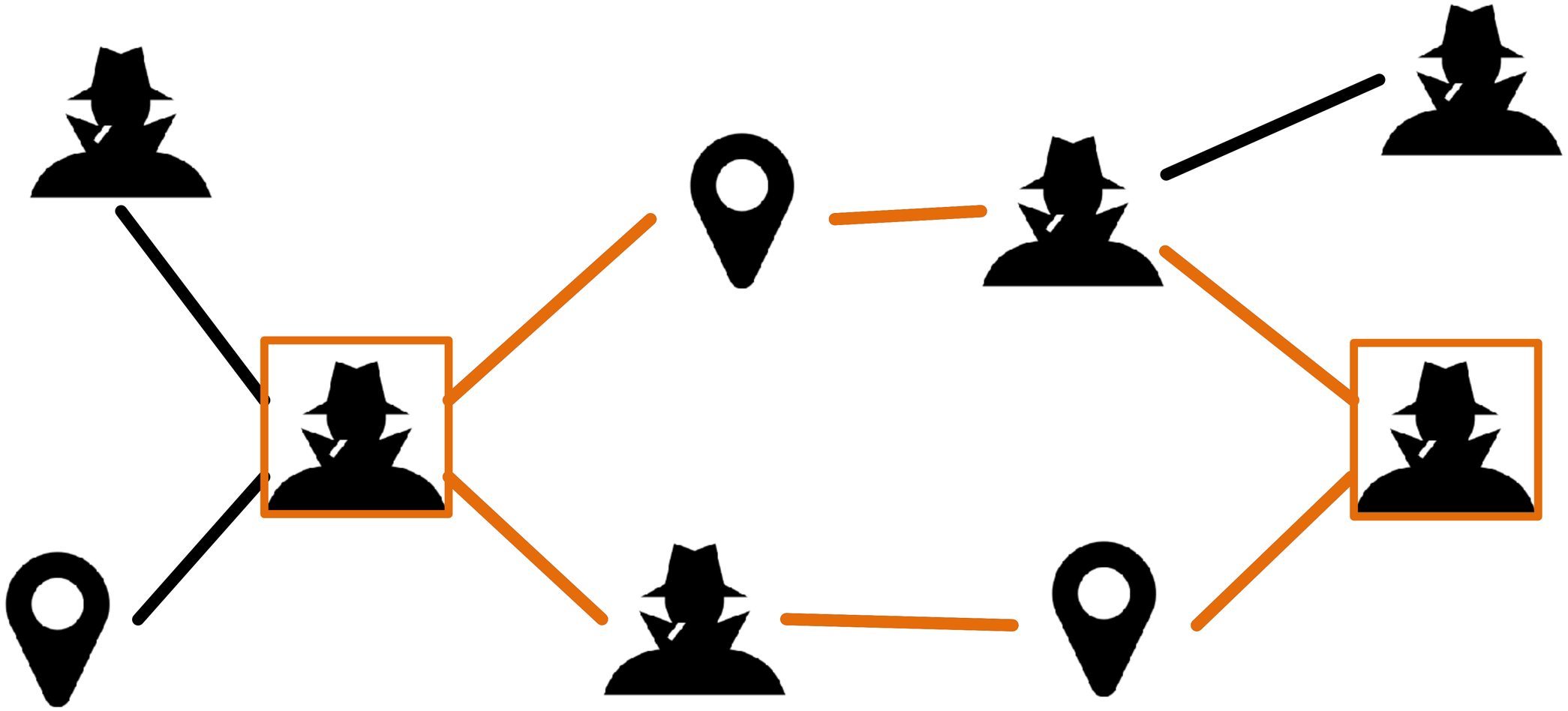
Pathfinder:

Visual Analysis of Paths in Graphs

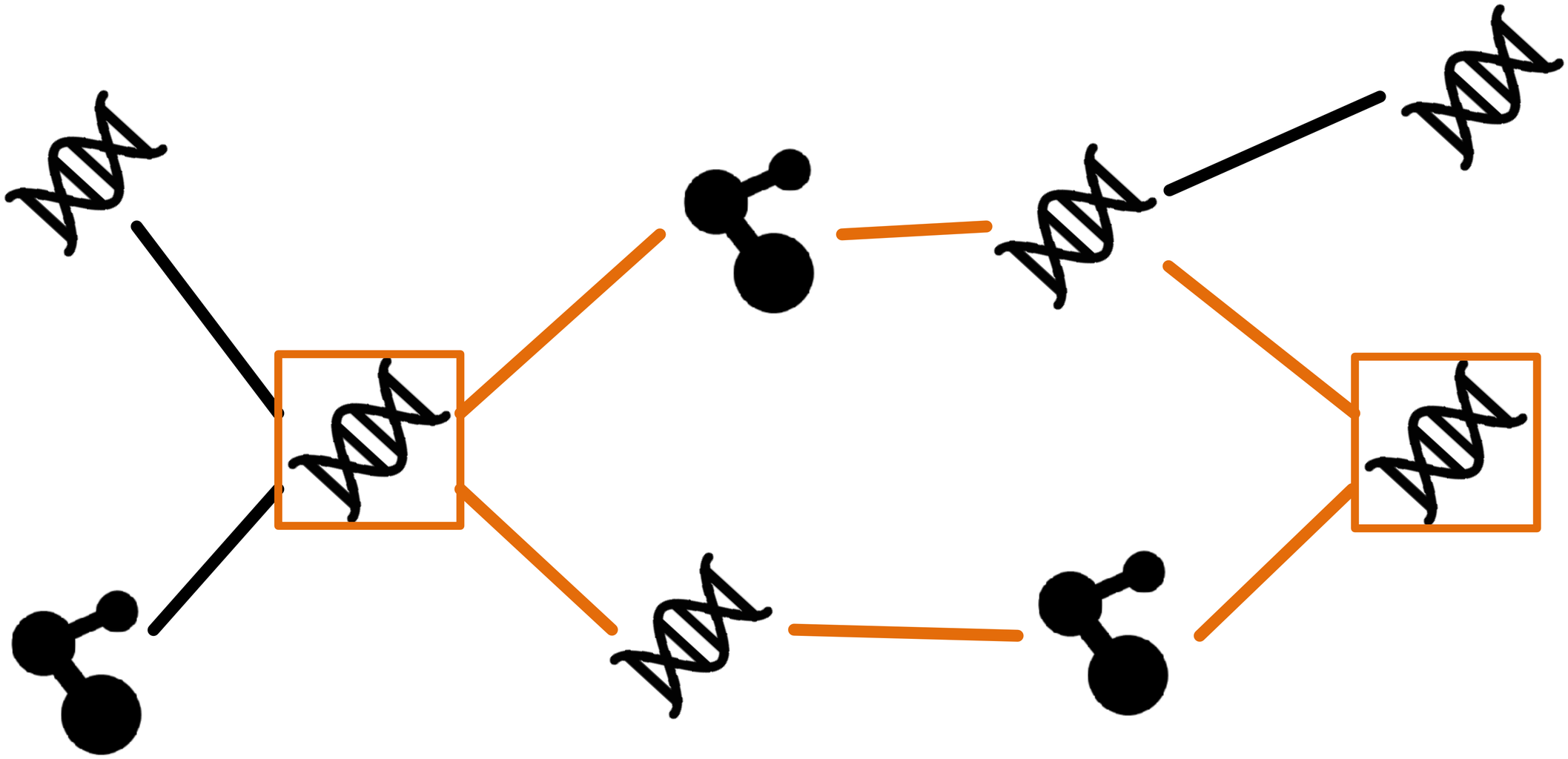




Intelligence Data: How are two suspects connected?



Intelligence Data: How are two suspects connected?



Biological Network: How do two genes interact?

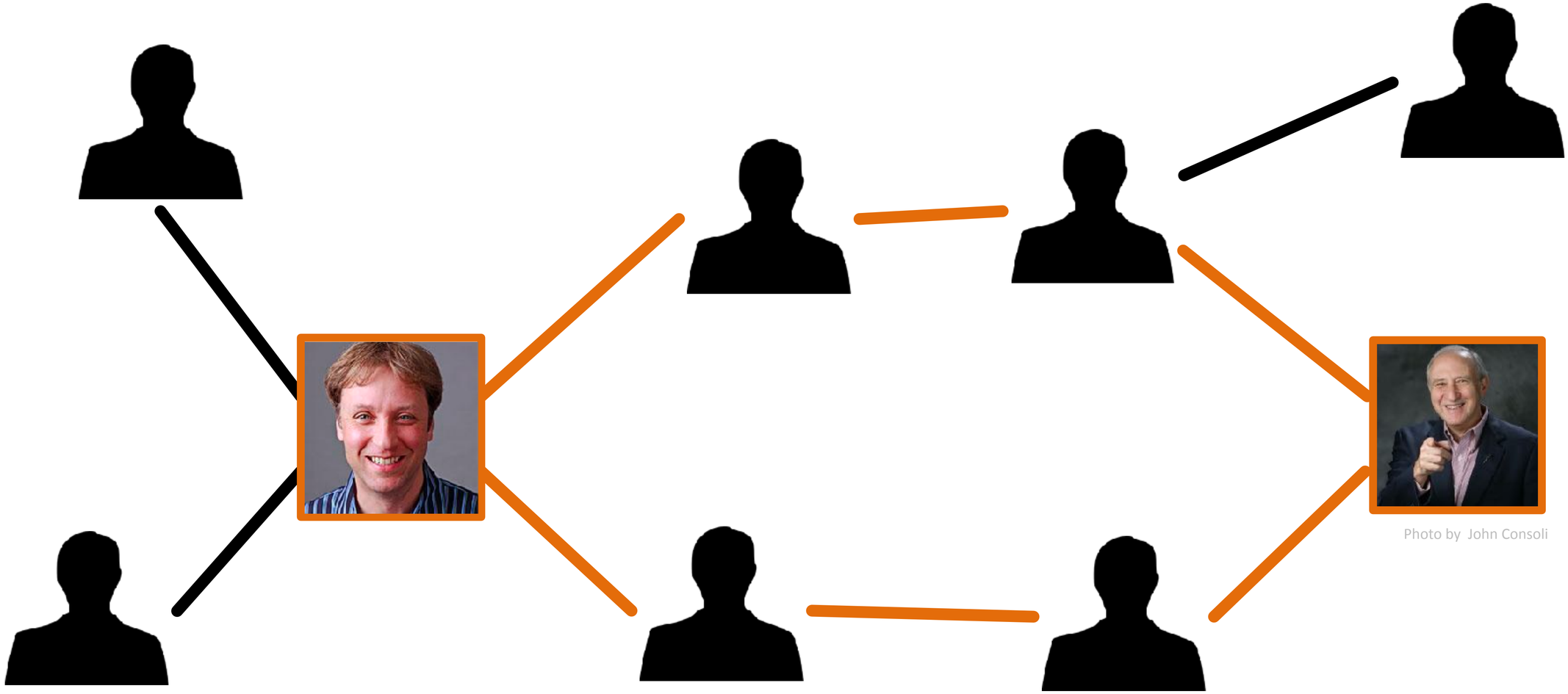
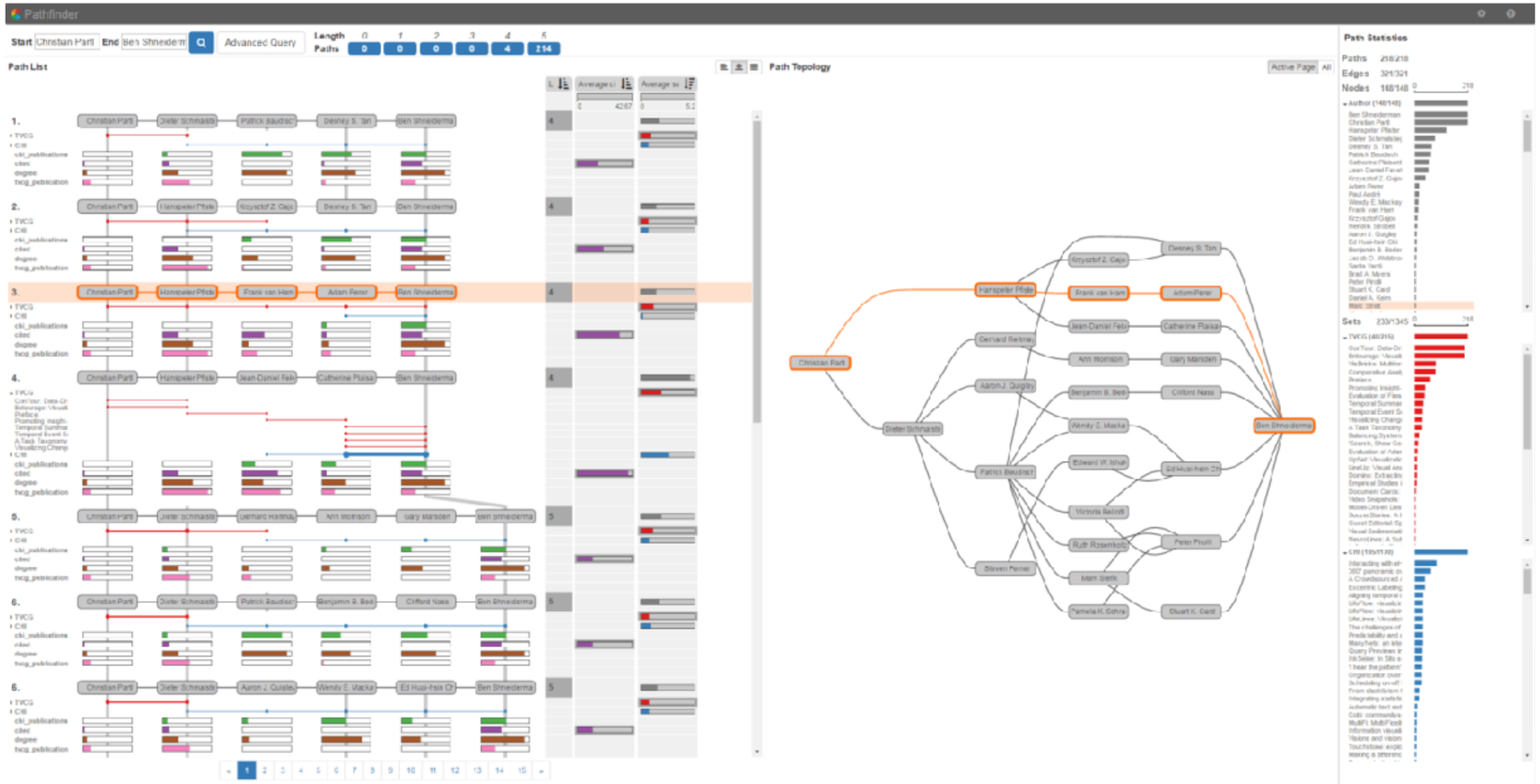


Photo by John Consoli

Coauthor Network: How is HP Pfister connected to Ben Shneiderman?

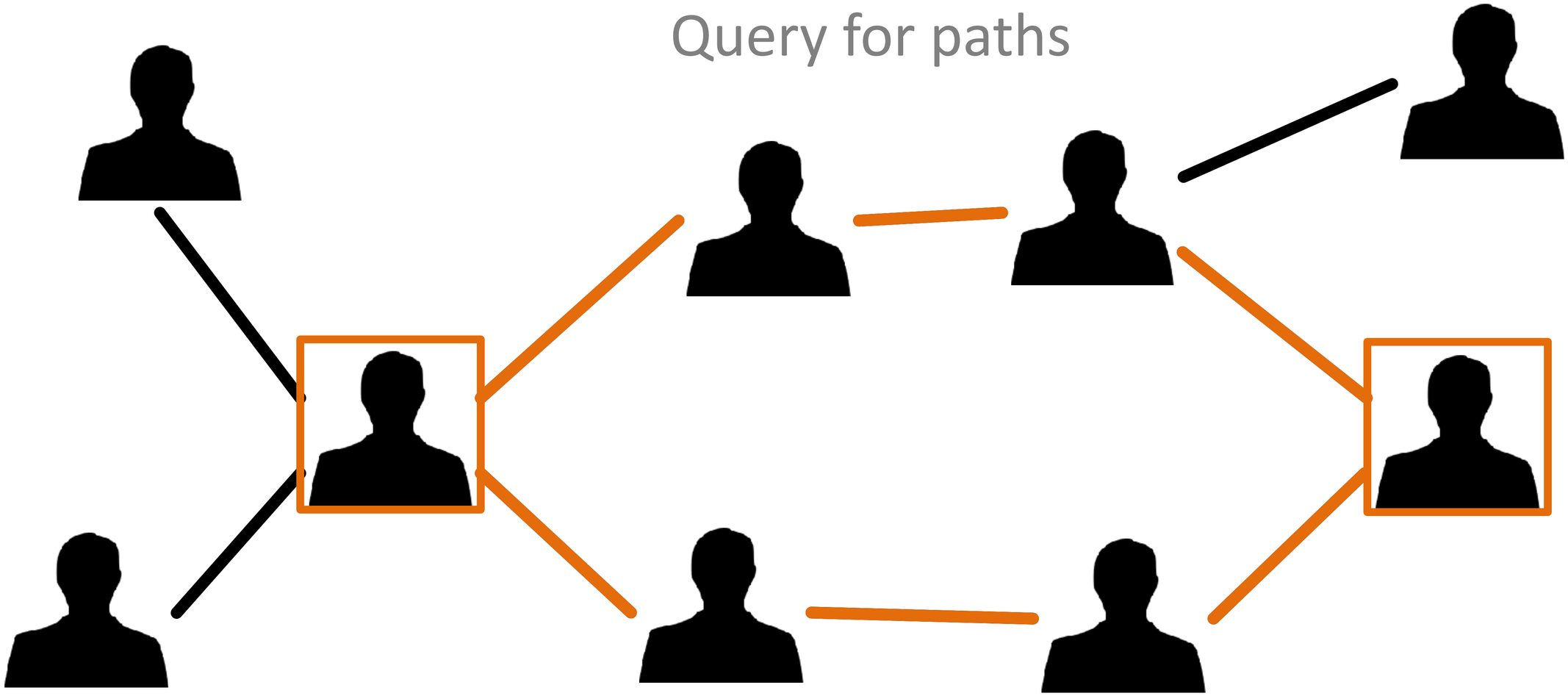
Pathfinder



Visual Analysis of Paths in Large Multivariate Graphs

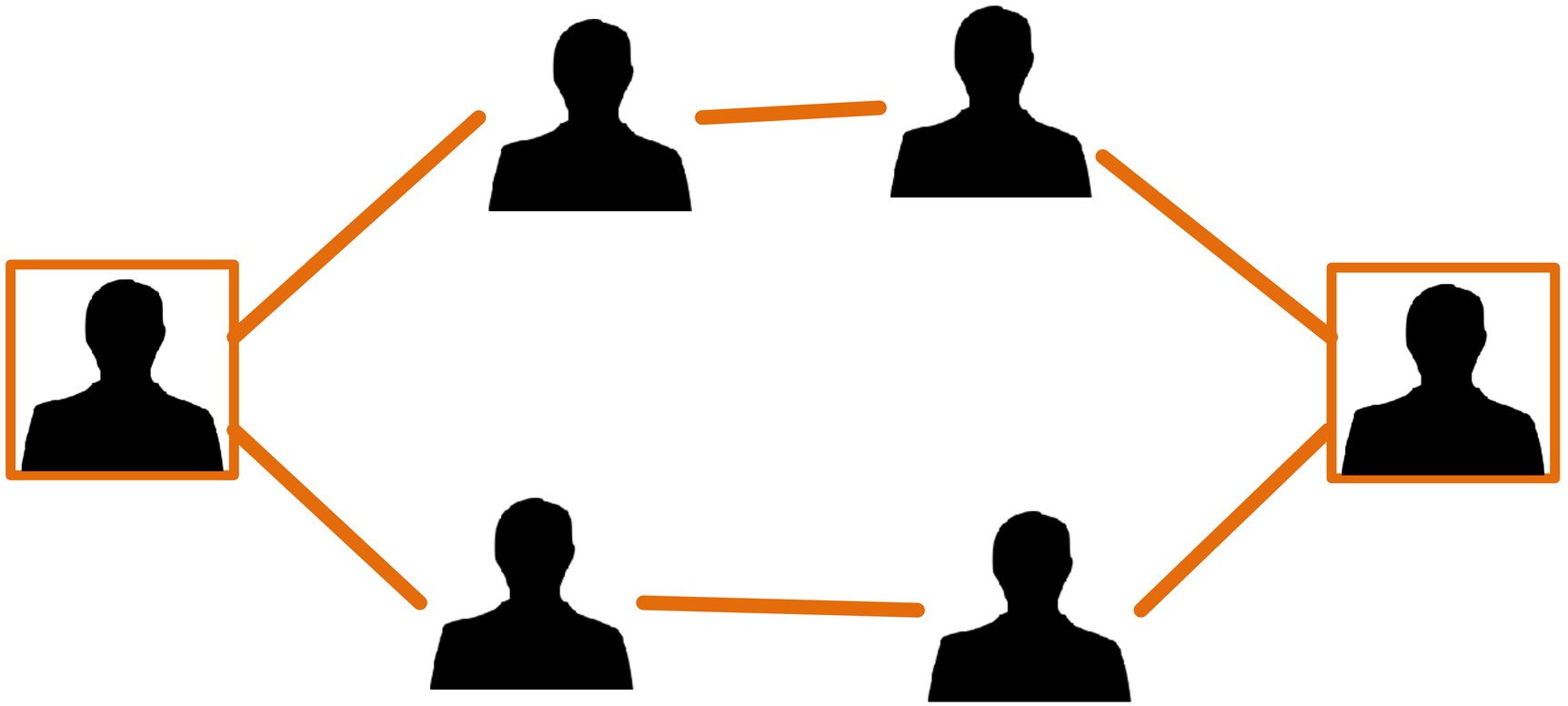
Pathfinder Approach

Query for paths



Pathfinder Approach

Shows node-edge diagram...

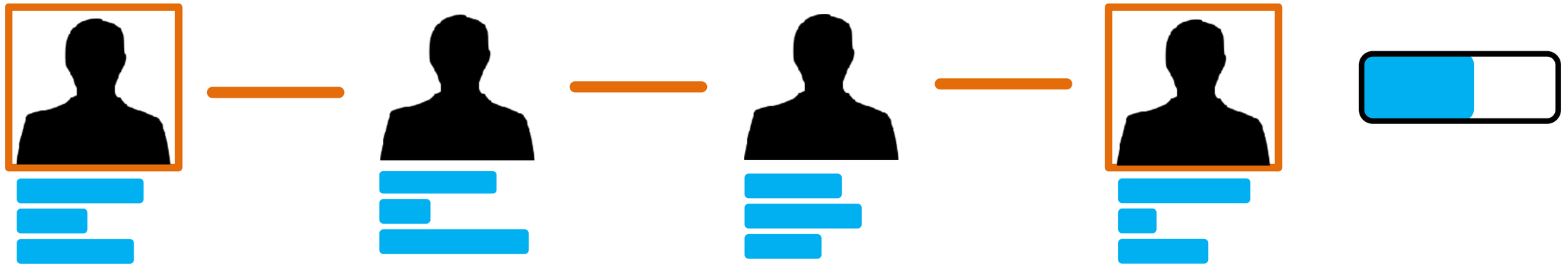


Pathfinder Approach

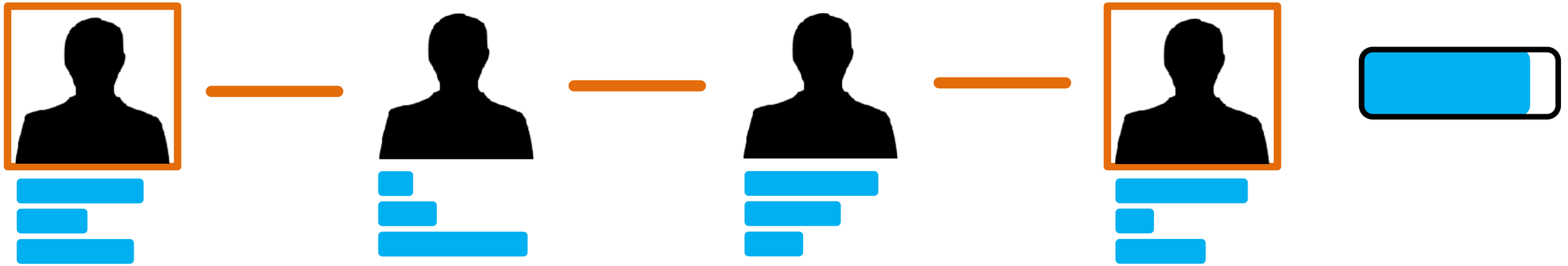
Update ranking to identify important paths

Path Score

1.

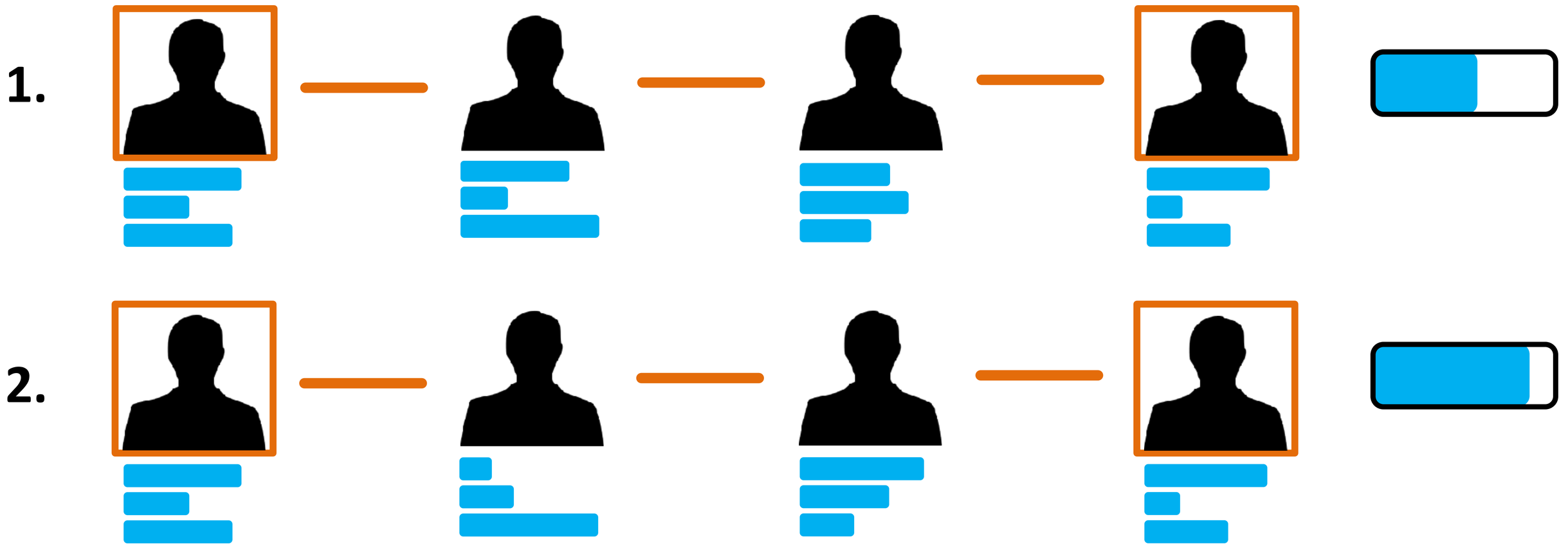


2.



Pathfinder Approach

Update ranking to identify important paths



Pathfinder

Start End Advanced Query Length Paths

Path Statistics

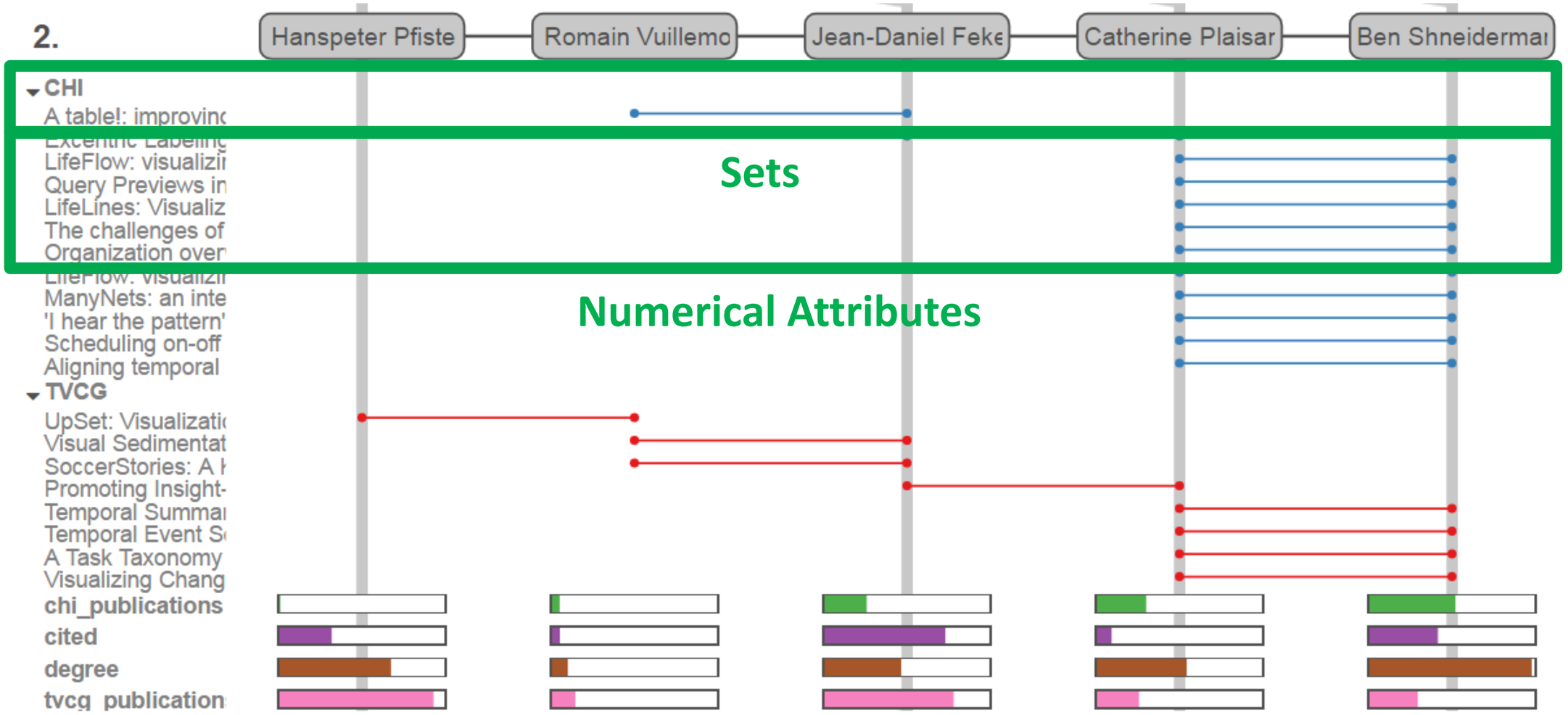
Path List

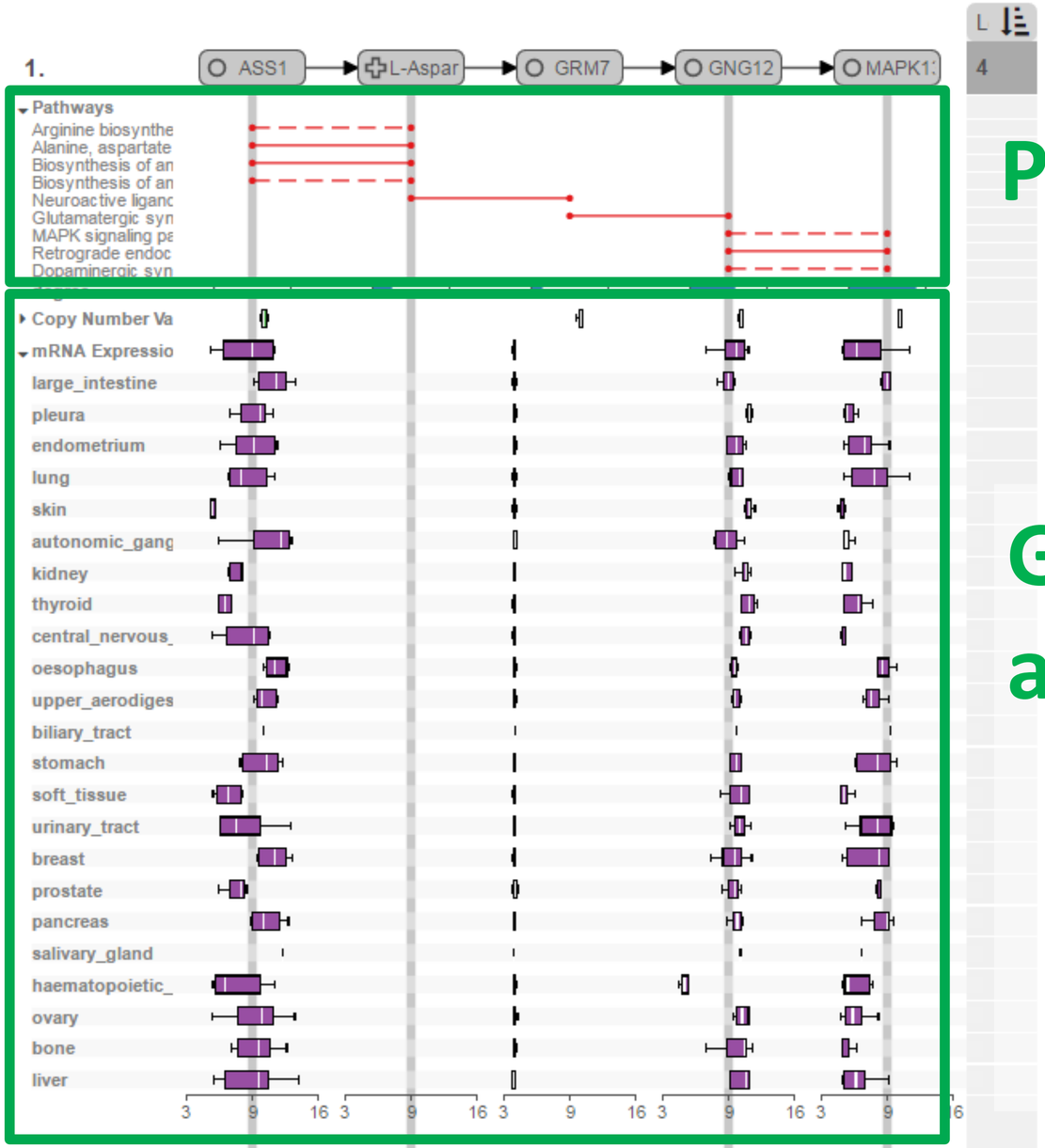
Path Topology Active Page All

Query Interface

« »

Path Representation

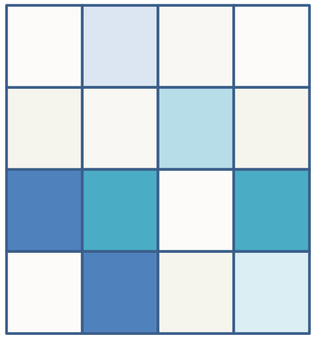




Pathways

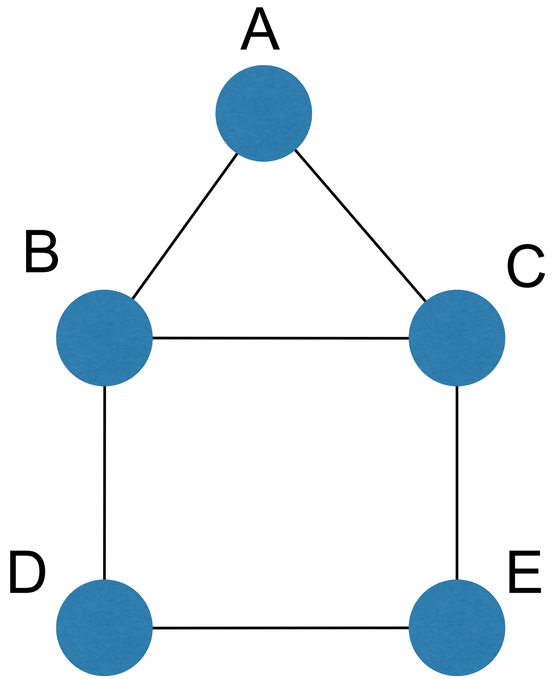
Grouped Copy Number and Gene Expression Data

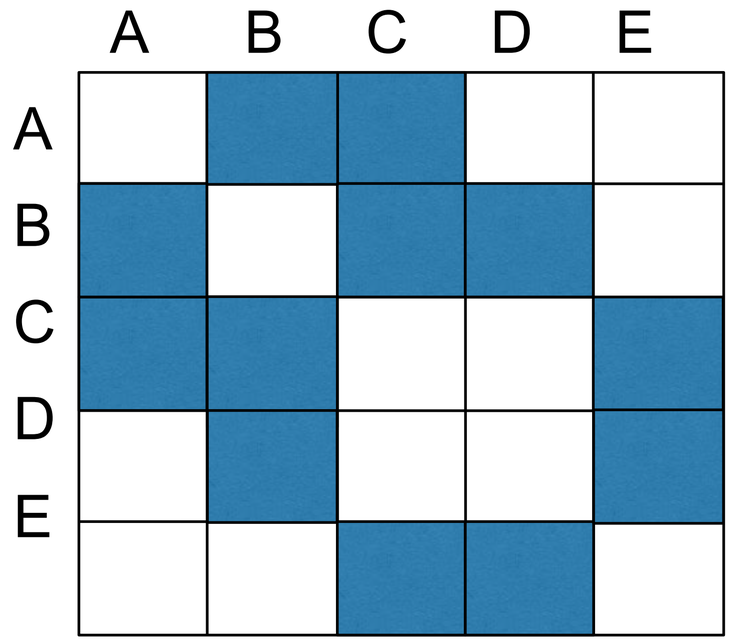
Matrix Representations



Matrix Representations

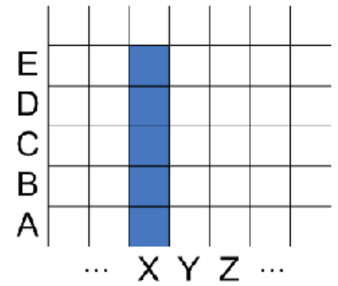
Instead of node link diagram, use adjacency matrix

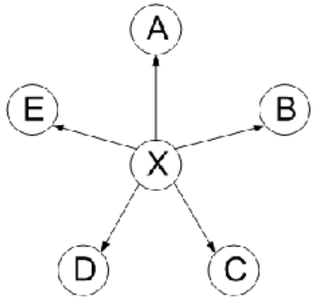


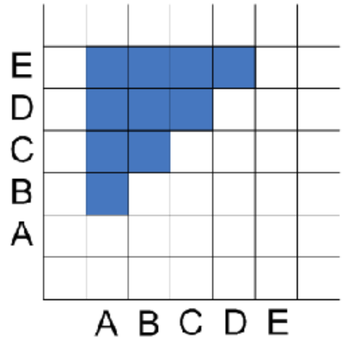


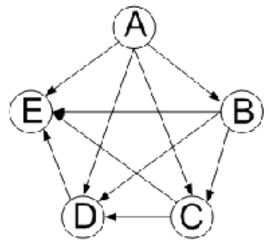
Matrix Representations

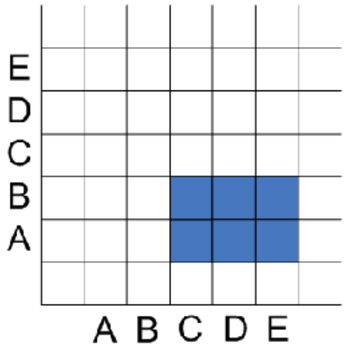
Examples:

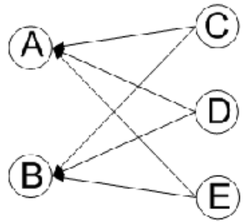


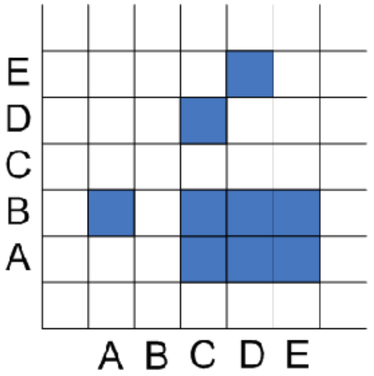


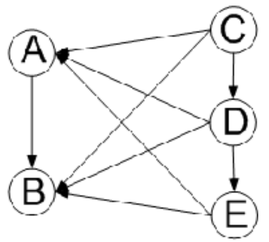




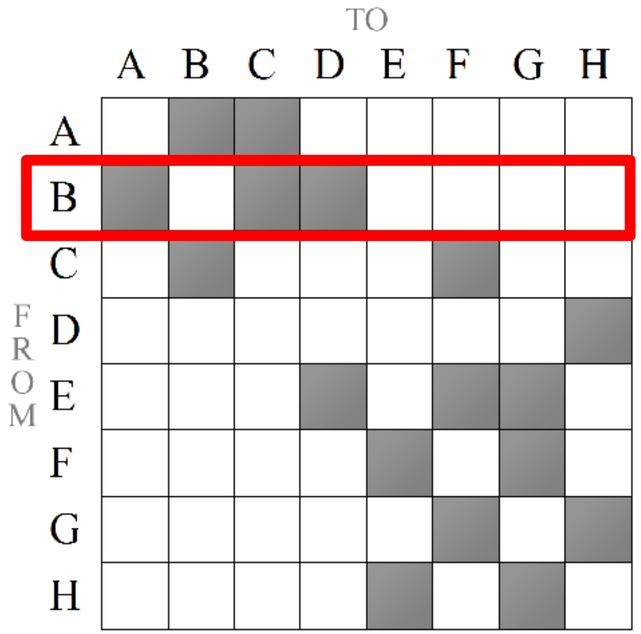




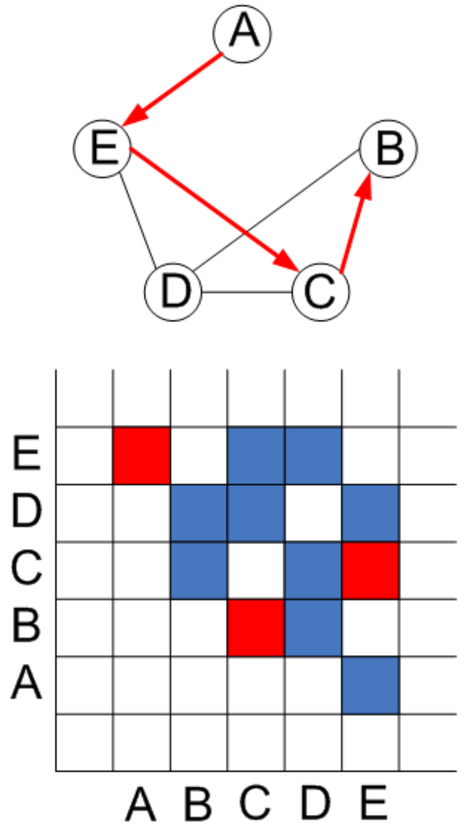




Matrix Representations



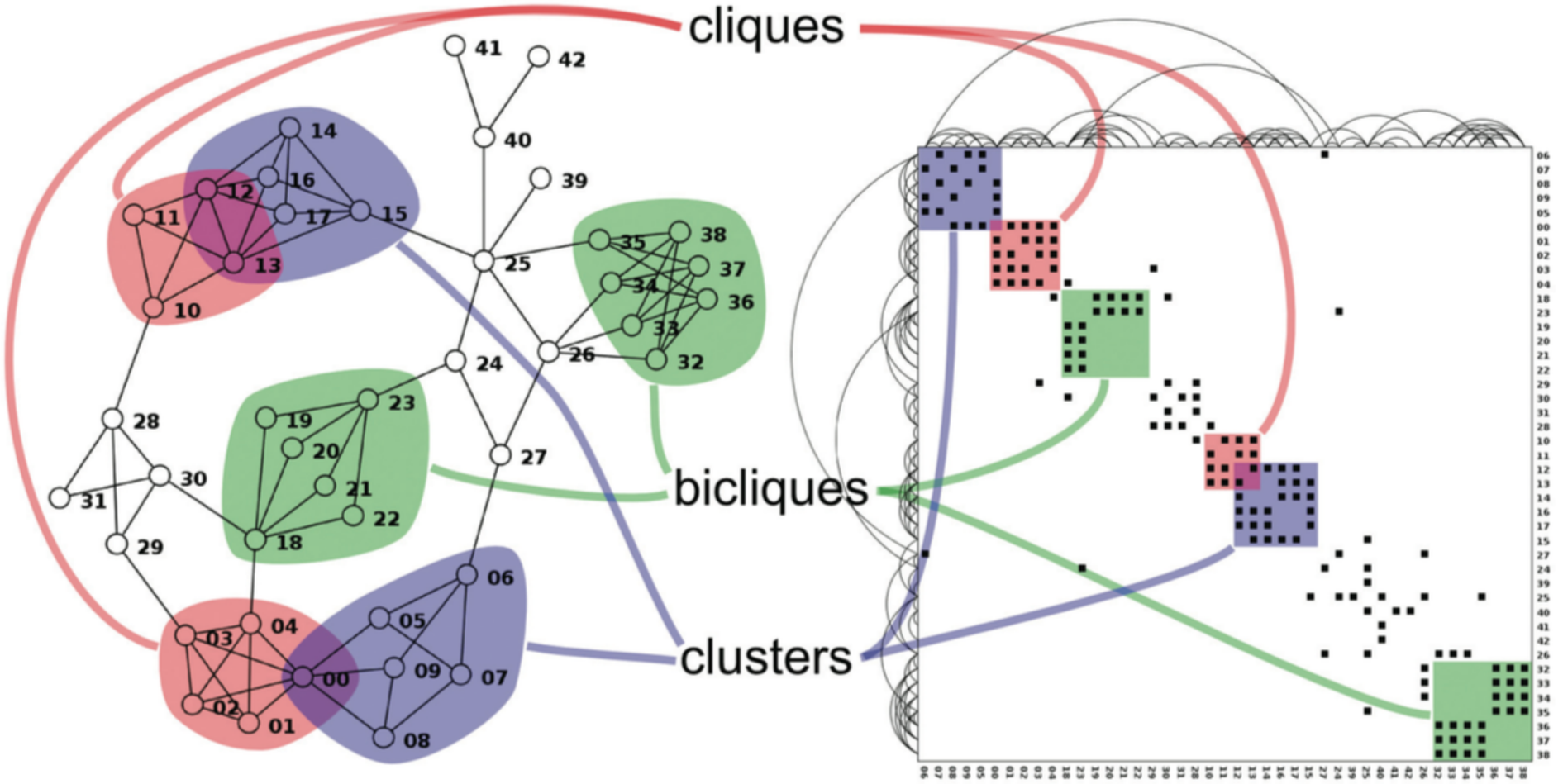
Well suited for
neighborhood-related TBTs



Not suited for
path-related TBTs

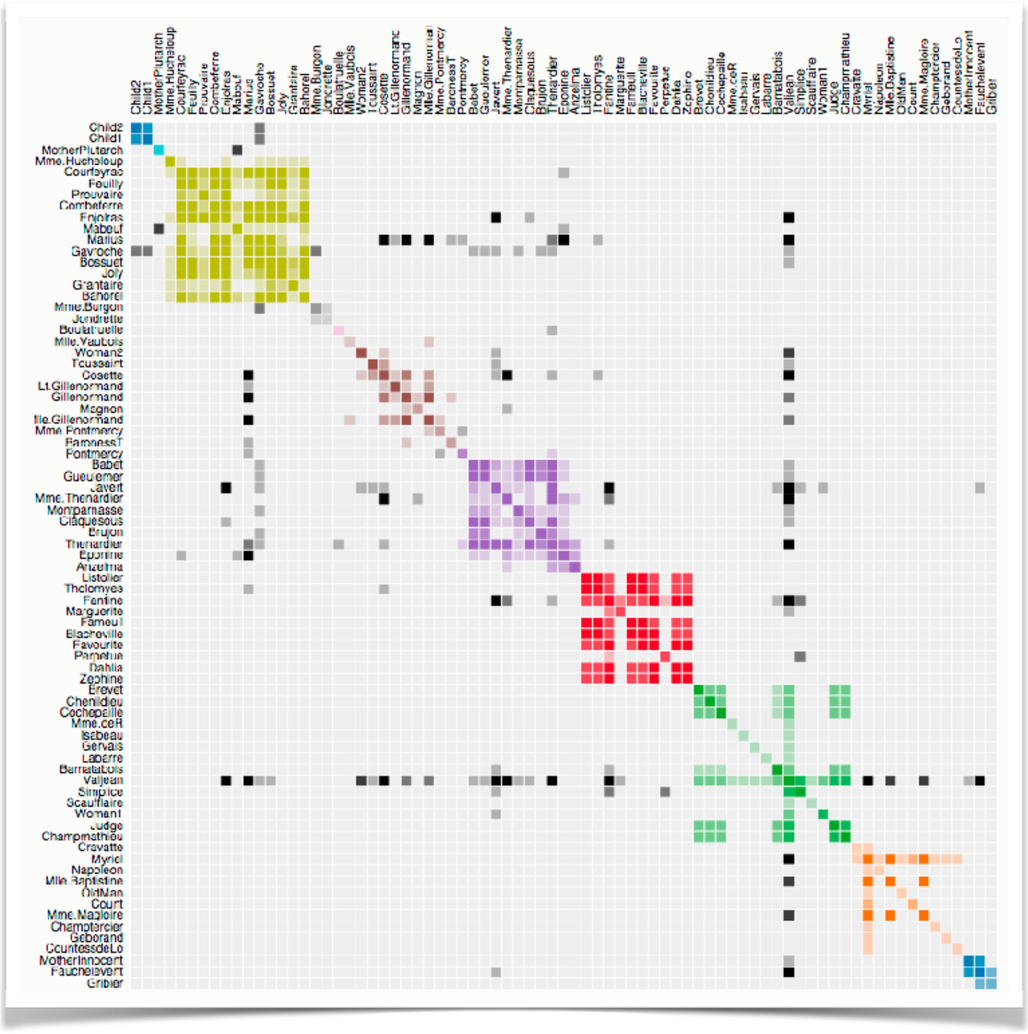
van Ham et al. 2009

Shen et al. 2007



McGuffin 2012

Order Critical!



Matrix Representations

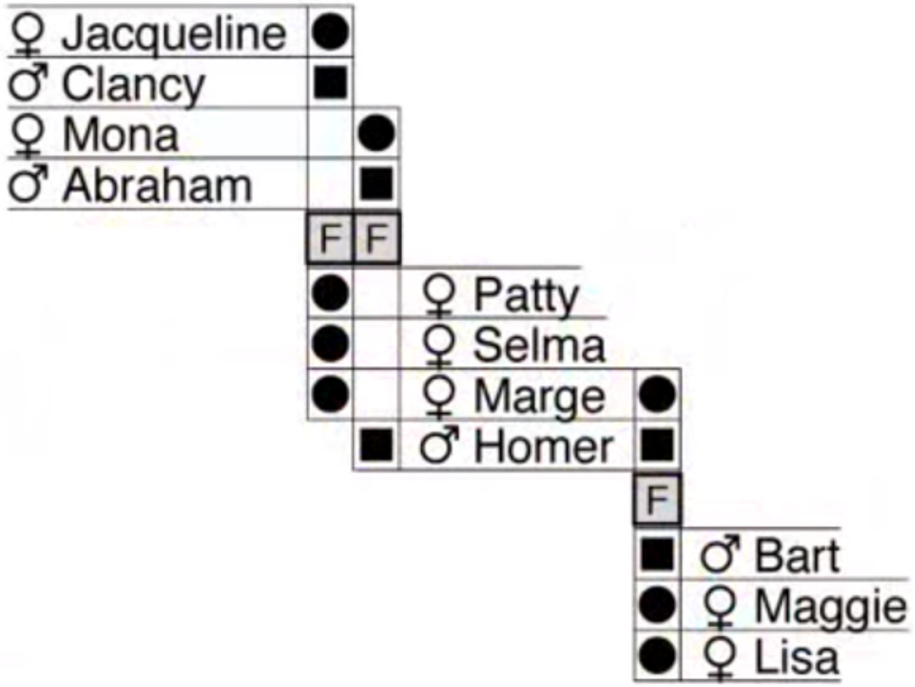
Pros:

- can represent **all graph classes** except for hypergraphs
- puts **focus on the edge set**, not so much on the node set
- simple grid -> **no elaborate layout** or rendering needed
- well suited for **ABT on edges** via coloring of the matrix cells
- well suited for **neighborhood-related TBTs** via traversing rows/columns

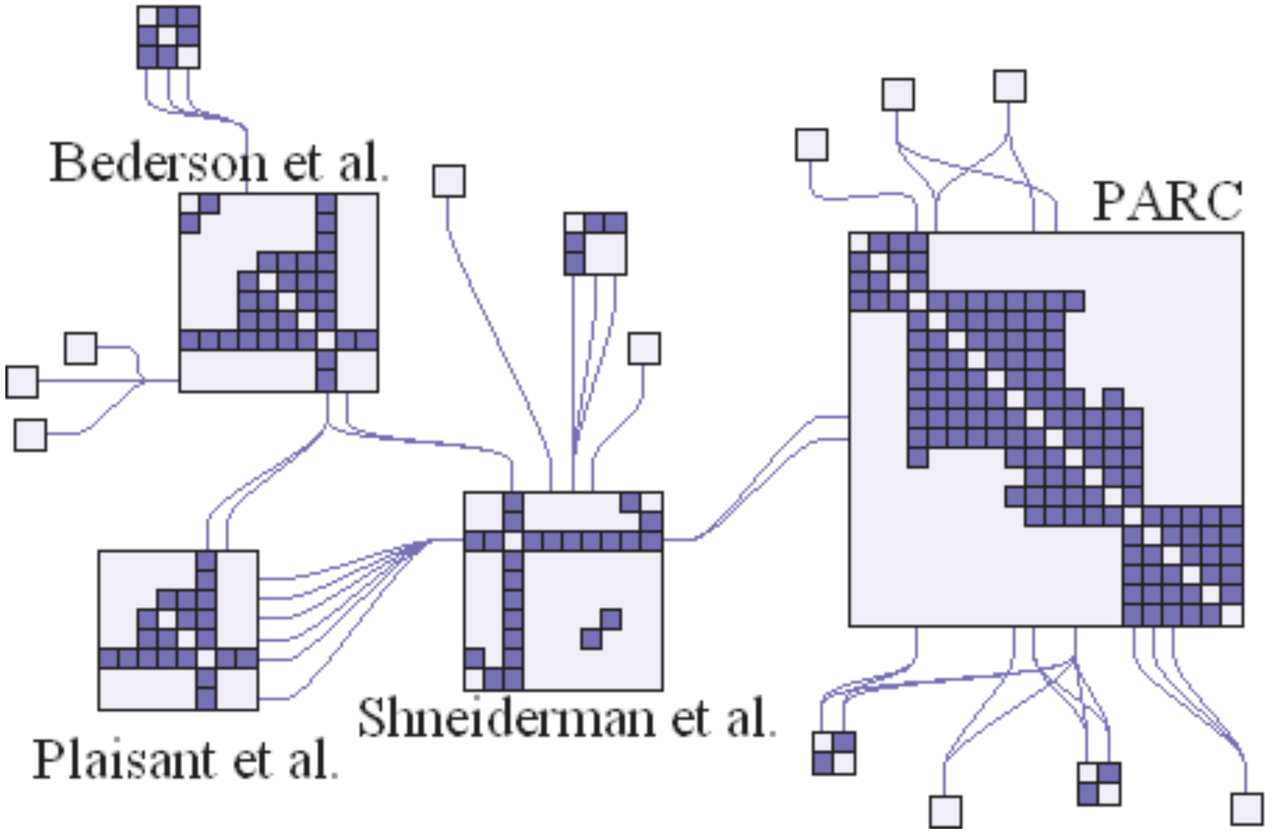
Cons:

- quadratic screen space requirement (any possible edge takes up space)
- not suited for path-related TBTs

Special Case: Genealogy



Hybrid Explicit/Matrix

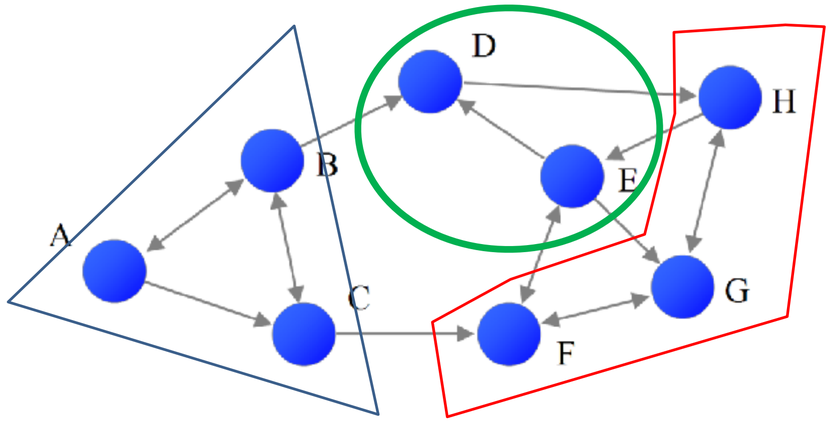


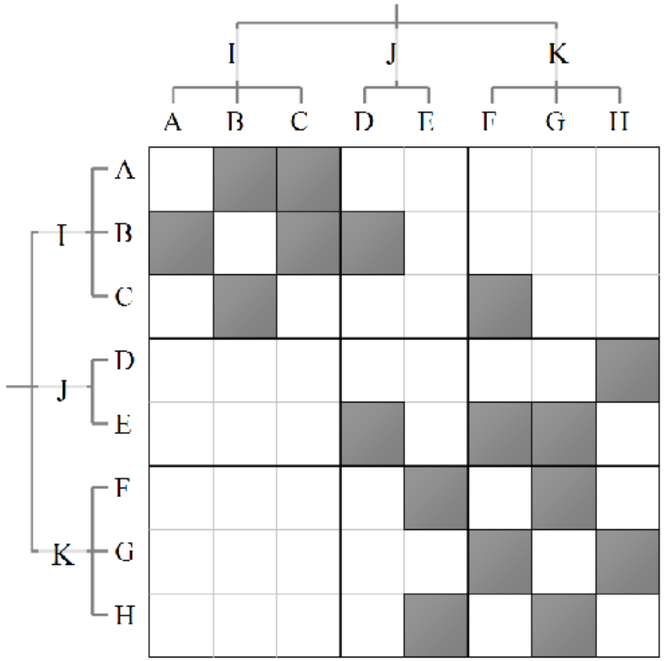
NodeTrix
[Henry et al. 2007]

Matrix Representations

Problem #1: used screen real estate is quadratic in the number of nodes

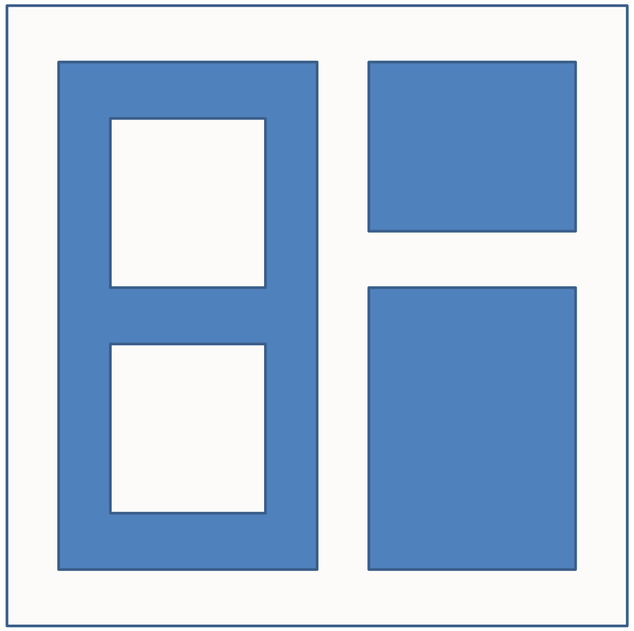
Solution approach: hierarchization of the representation



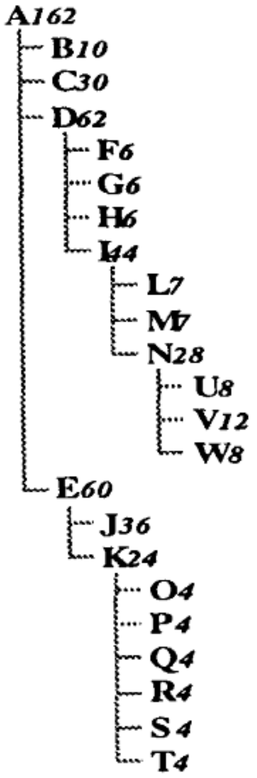


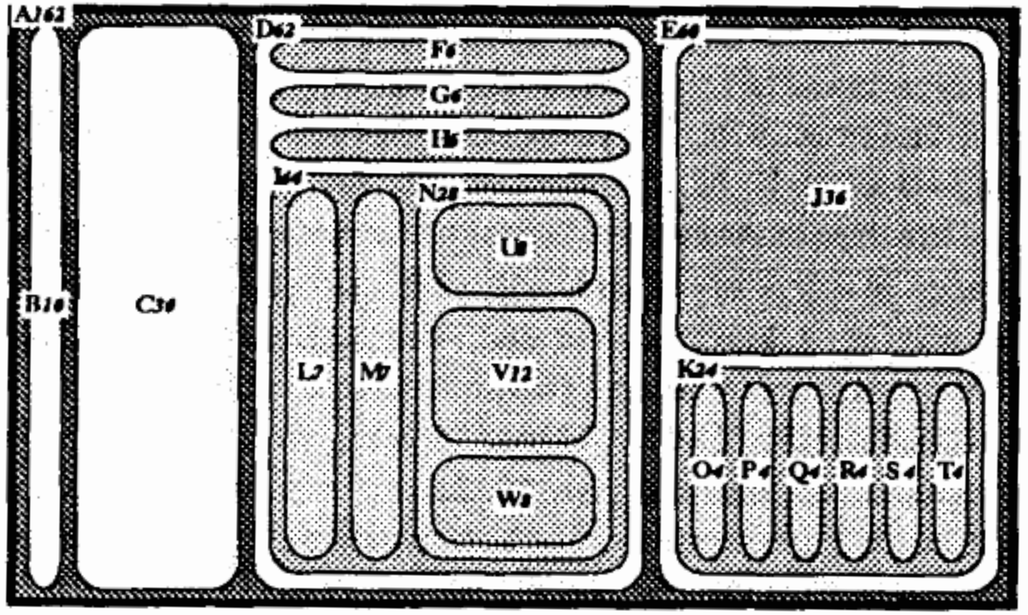
[van Ham et al. 2009]

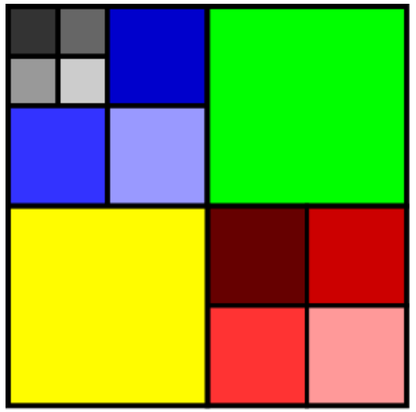
Implicit Layouts for Trees

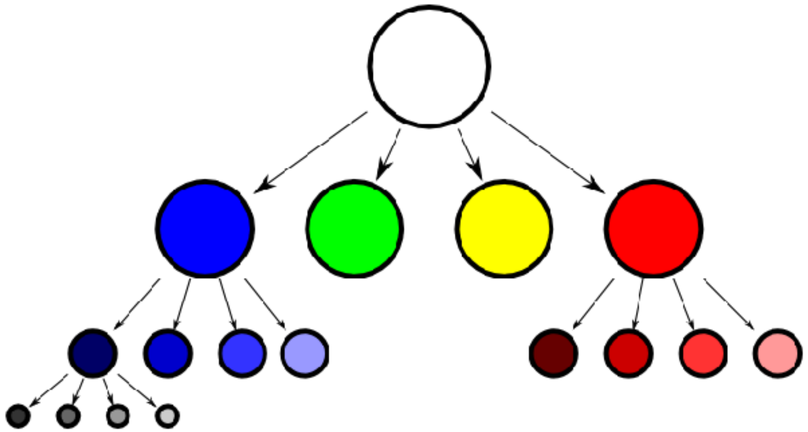


Tree Maps



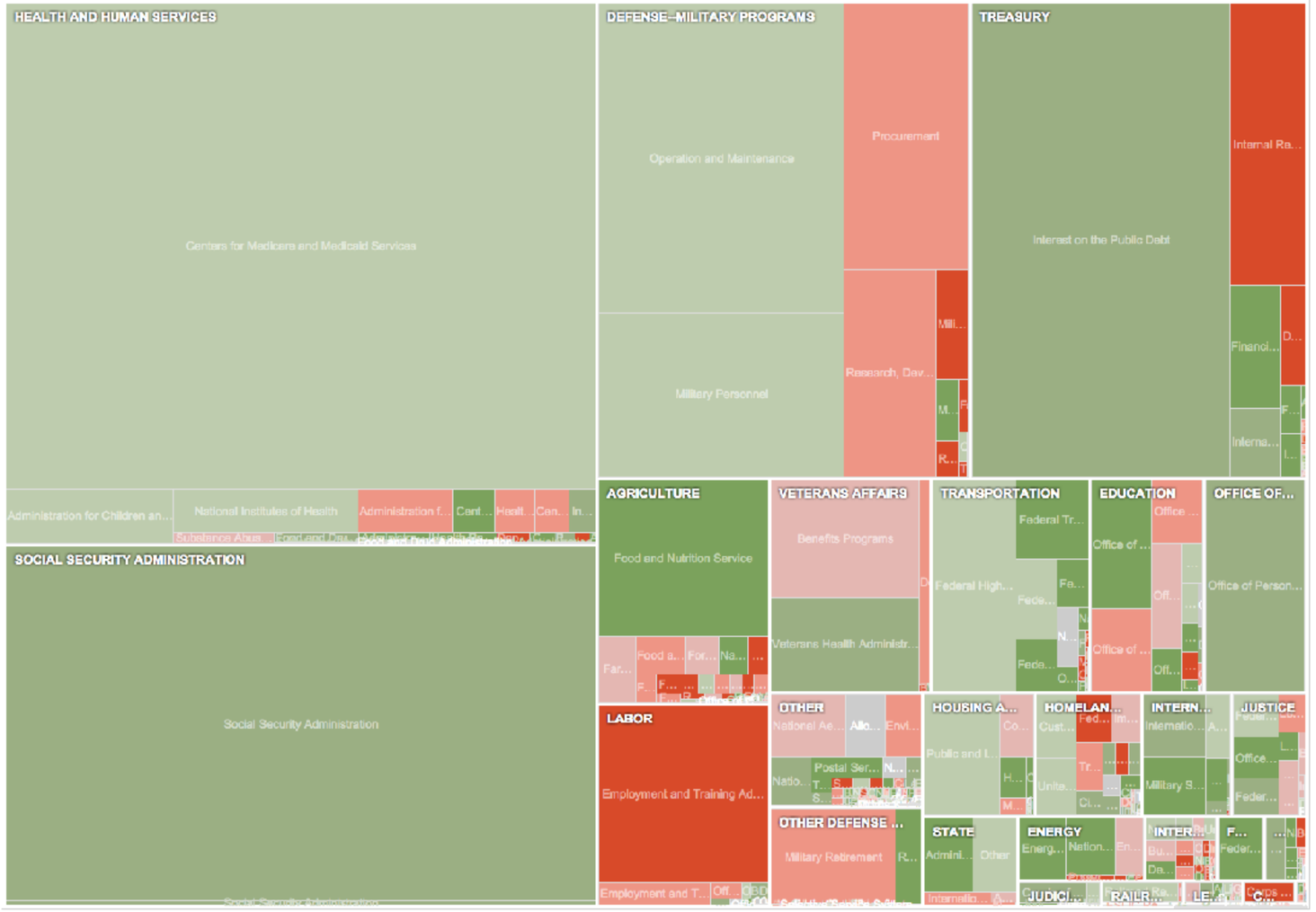




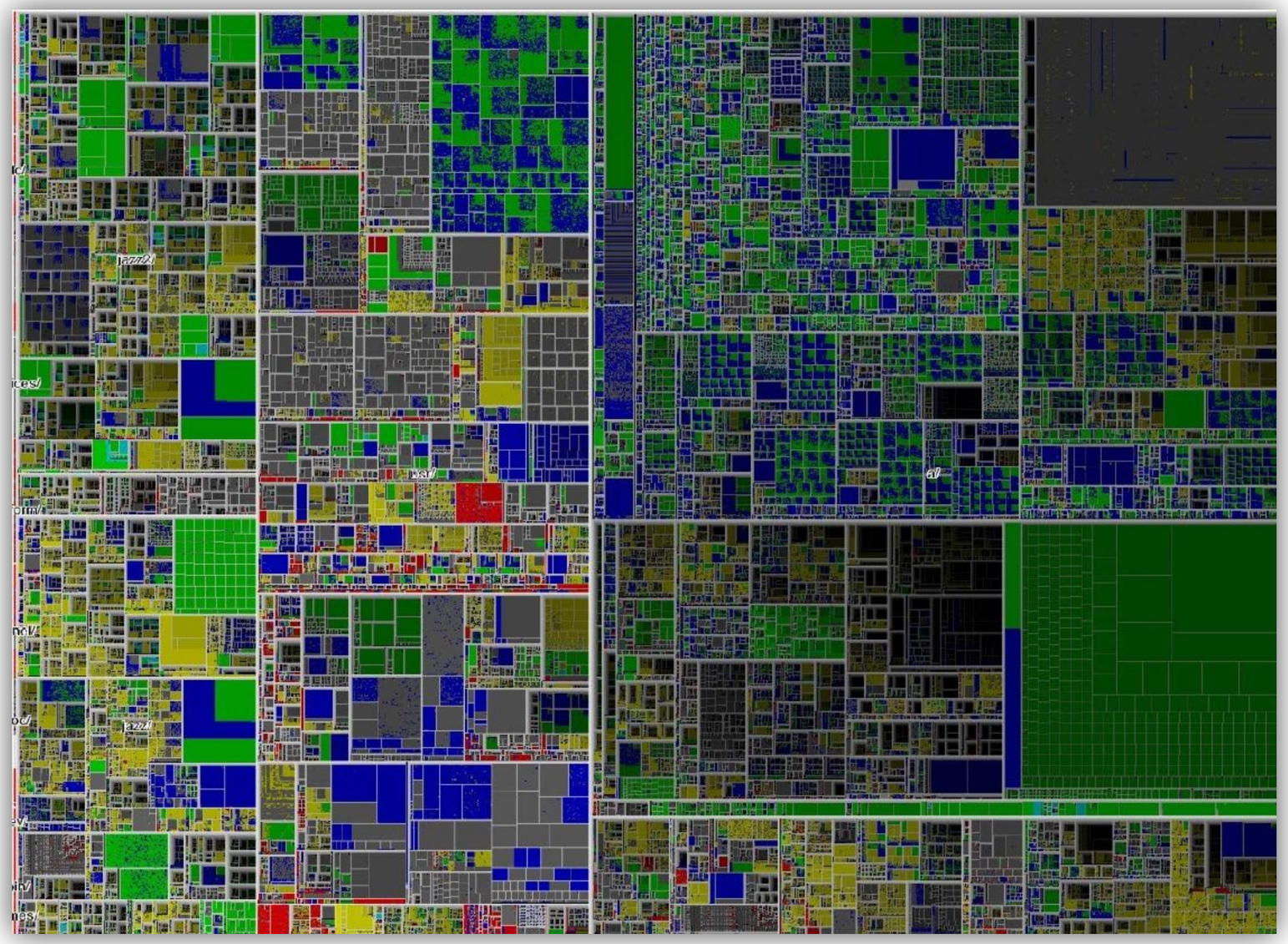


Johnson and Shneiderman 1991

Zoomable Treemap



Example: Interactive TreeMap of a Million Items



Fekete et al. 2002

Sunburst: Radial Layout

[Sunburst by John Stasko, Implementation in Caleydo by Christian Partl]

Implicit Representations

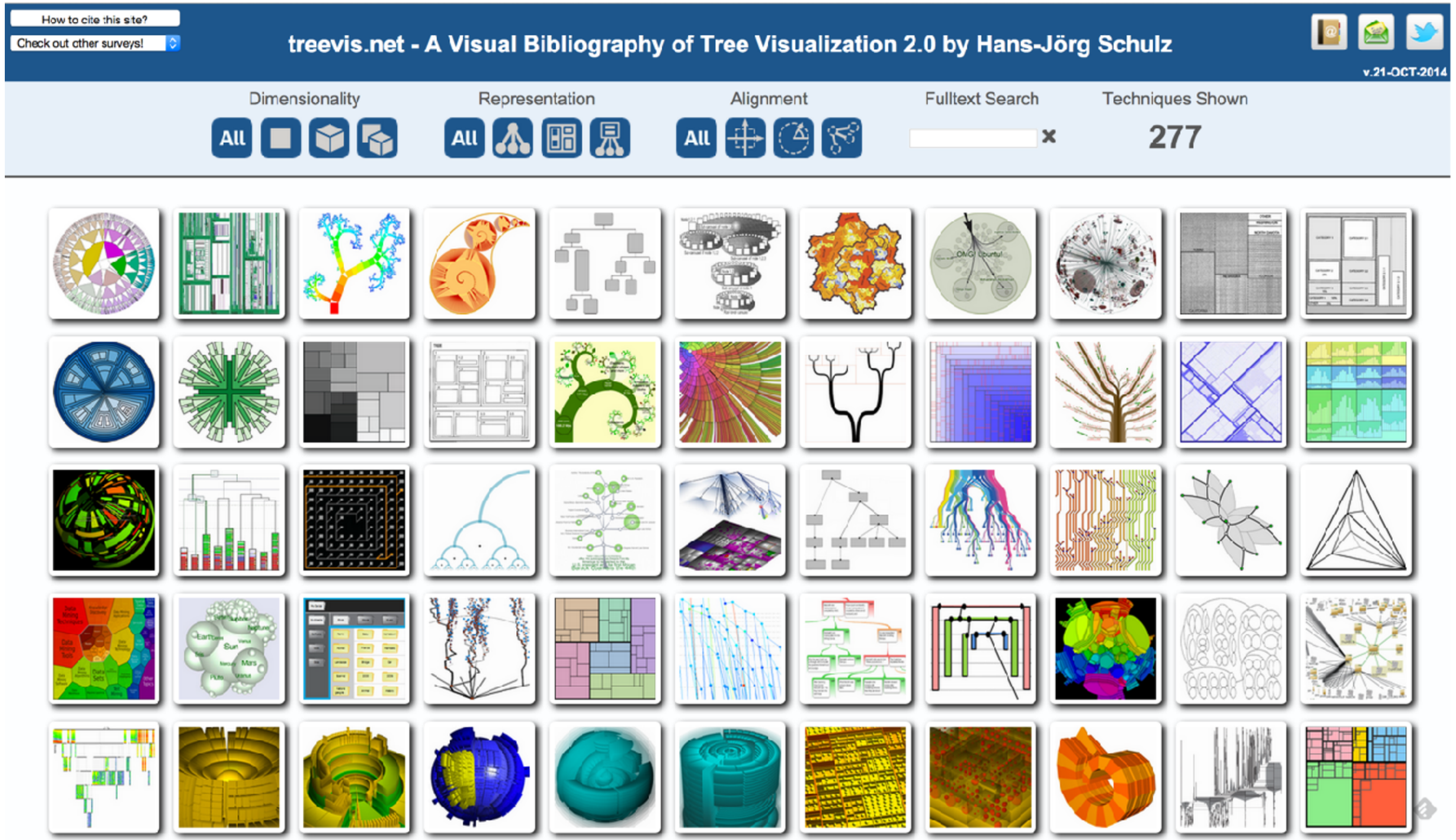
Pros:

- space-efficient because of the lack of explicitly drawn edges: scale well up to very large graphs
- in most cases well suited for ABTs on the node set
- depending on the spatial encoding also useful for TBTs

Cons:

- can only represent trees
- since the node positions are used to represent edges, they can no longer be freely arranged (e.g., to reflect geographical positions)
- useless to pursue any task on the edges
- spatial relations such as overlap or inclusion lead to occlusion

Tree Visualization Reference



Visualizing Time Varying Graphs

Up to now: given graphs were static

Extension: given is a sequence of graphs

either the sequence is given in full (offline)

or the sequence is streamed (online)

Variants:

varying linkage:

node set is fixed, only edges change over time

varying attributes:

graph structure is fixed, only attributes change

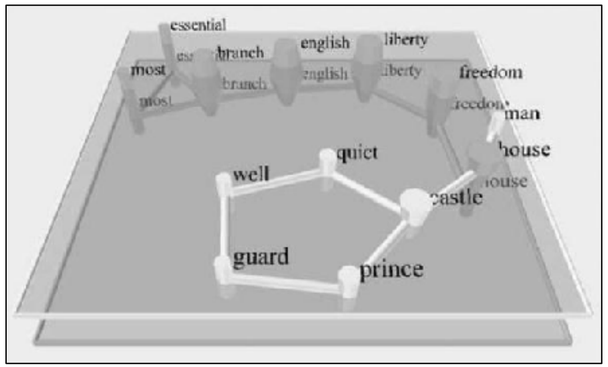
Visualizing Time Varying Graphs

Animation

- Map time to time

Layering

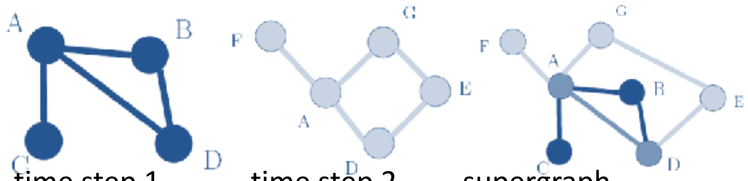
- Layout graph in 2D and use 3rd dimension to show time
- For small graphs with few time steps



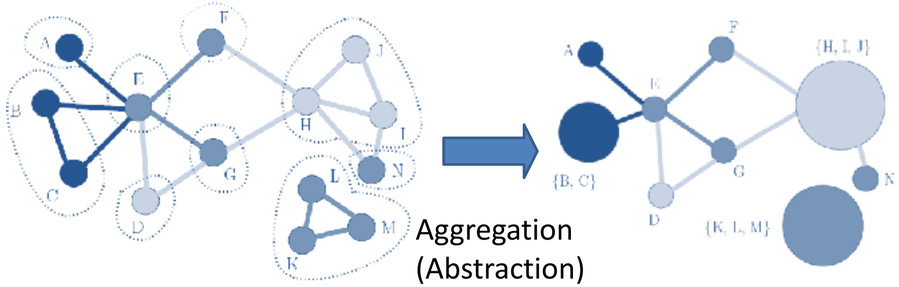
Brandes & Corman 2003

Supergraph

- Aggregate all time steps into a supergraph
- Use colors etc. to represent time



Aggregation



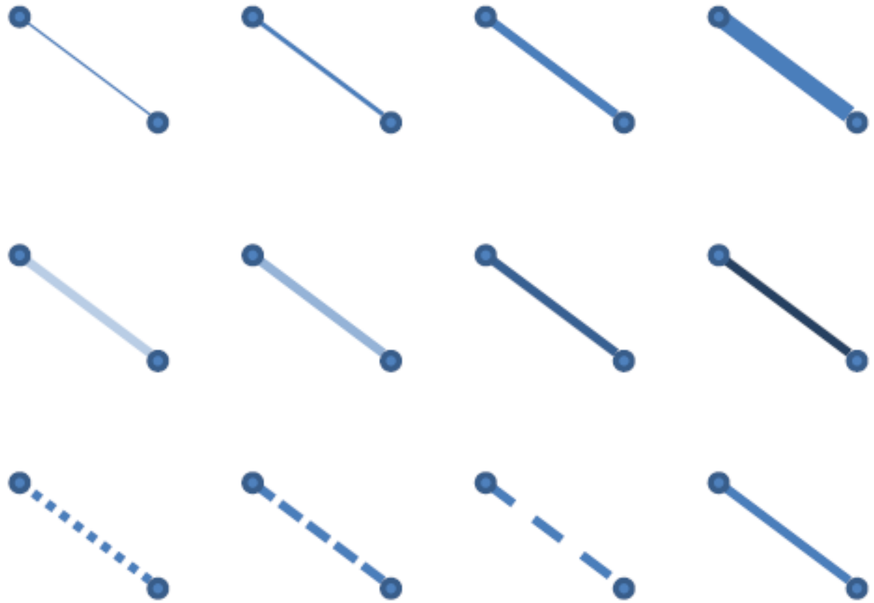
Visualizing Edge Attributes

Most common ways to encode edge attributes

Quantitative: Width

Ordinal: Saturation

Nominal: Style



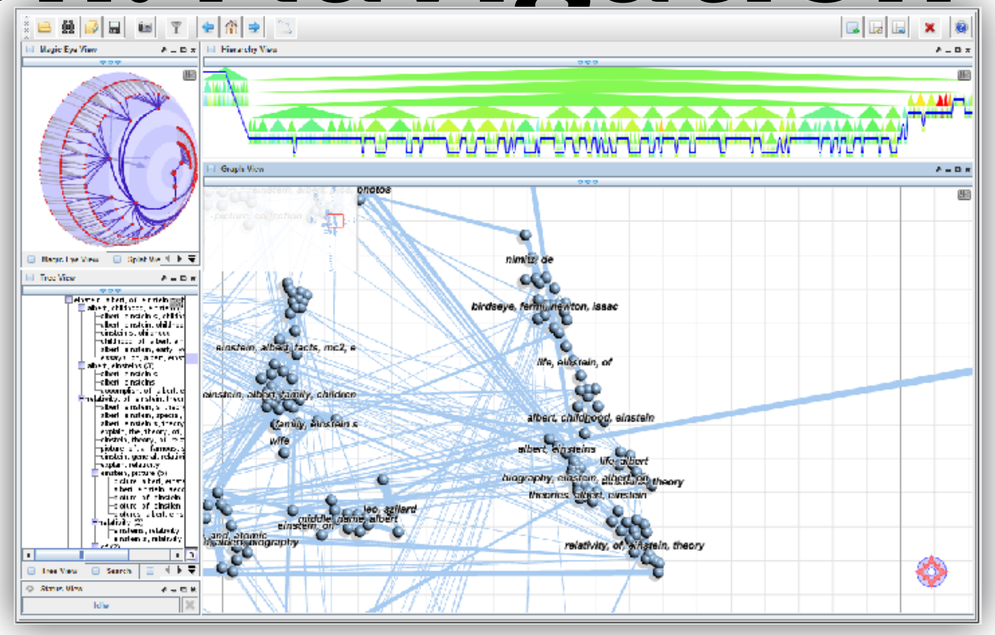
Graph Interaction: Navigation

Standard techniques

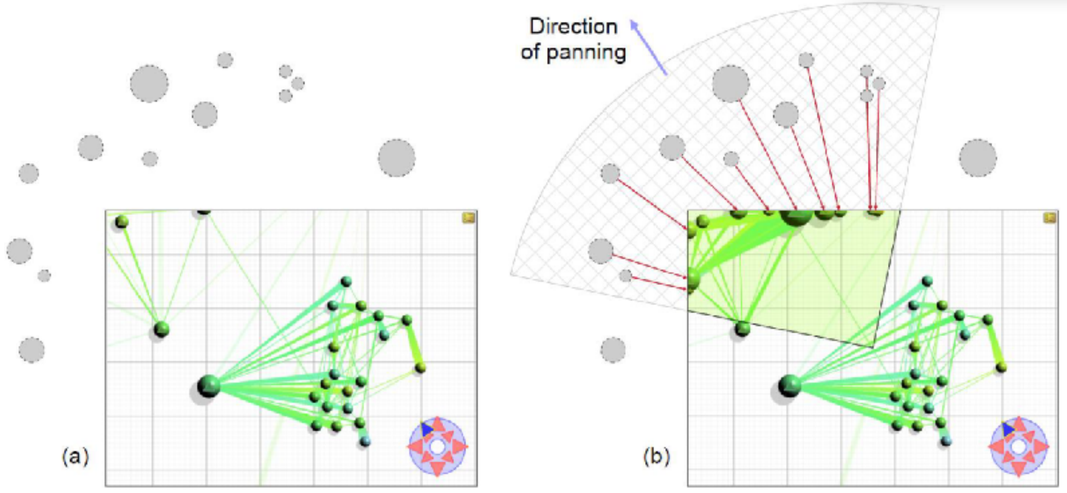
e.g., overview+detail

Edge-based traveling

Radar view for foresighted panning



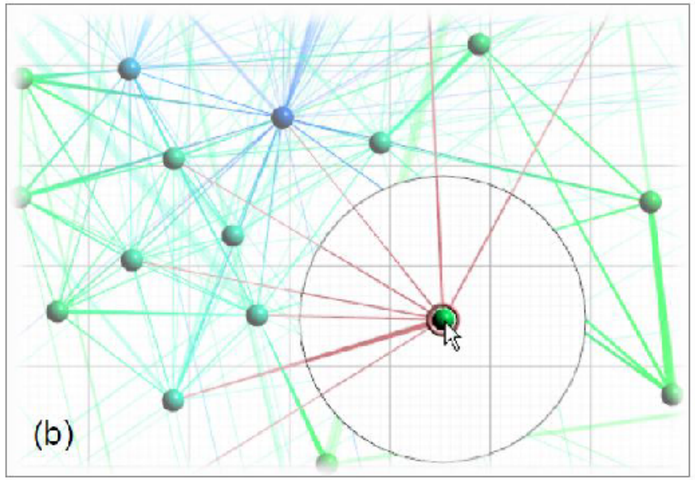
[Tominski et al. 2010]



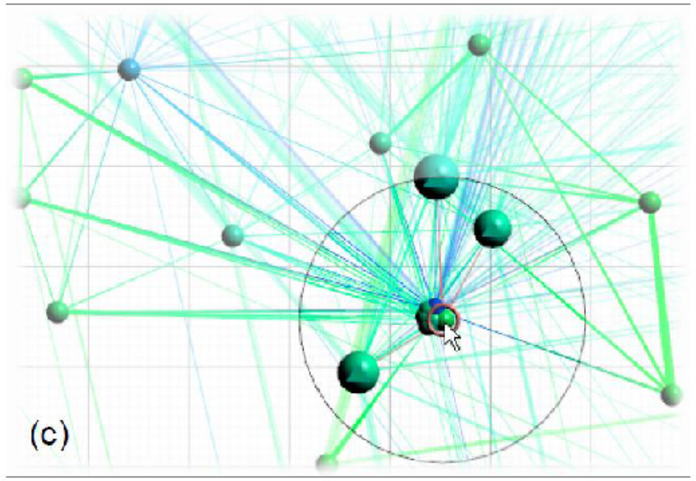
[Tominski et al. 2010]

Graph Interaction: Manipulation

Details-on-demand: smart lenses (semantic lenses)



Local-Edge-Lens shows only edges incident to the nodes inside



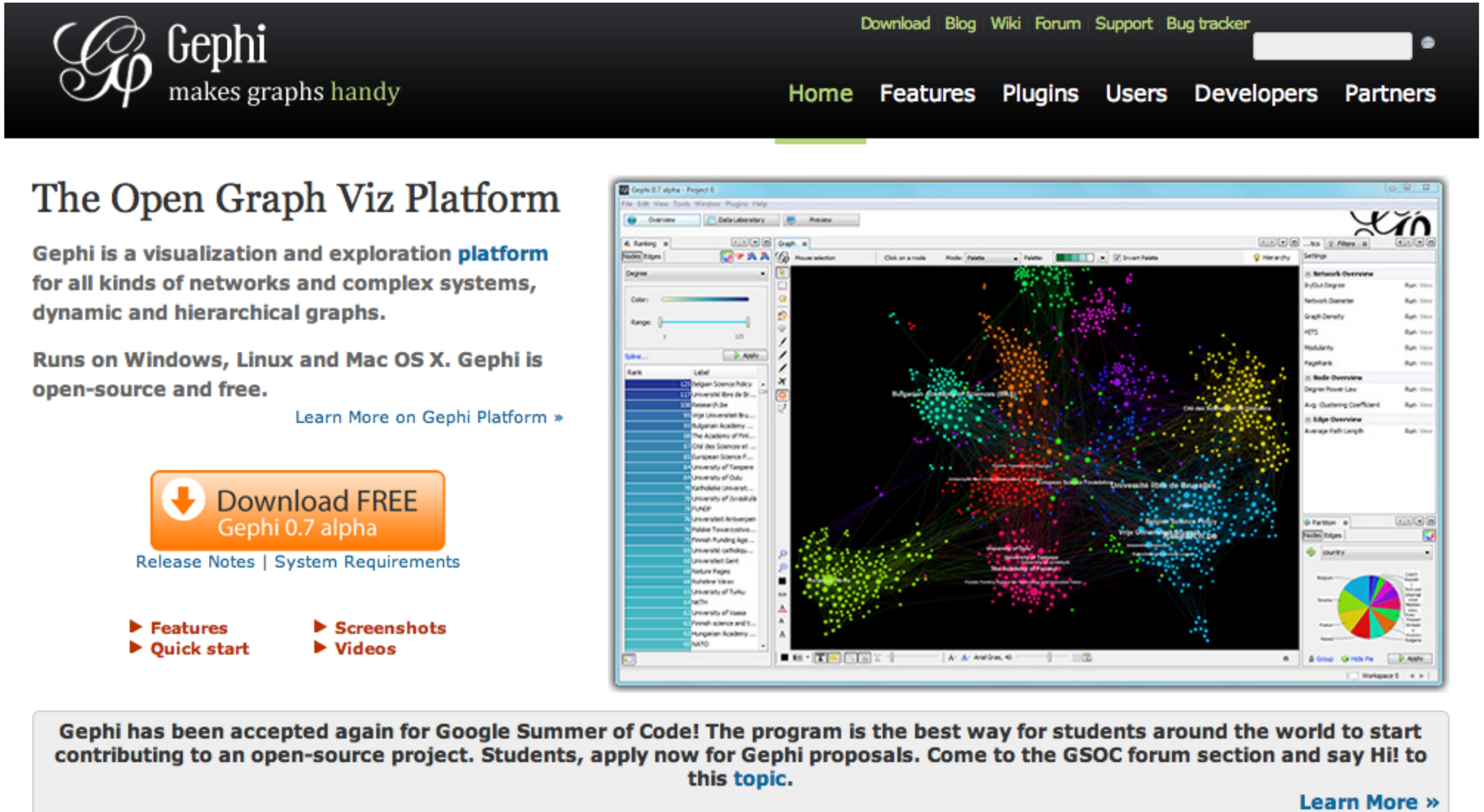
Bring-Neighbors-Lens gathers all neighbors of the center node

[Tominski et al. 2009]

Graph Tools & Applications

Gephi

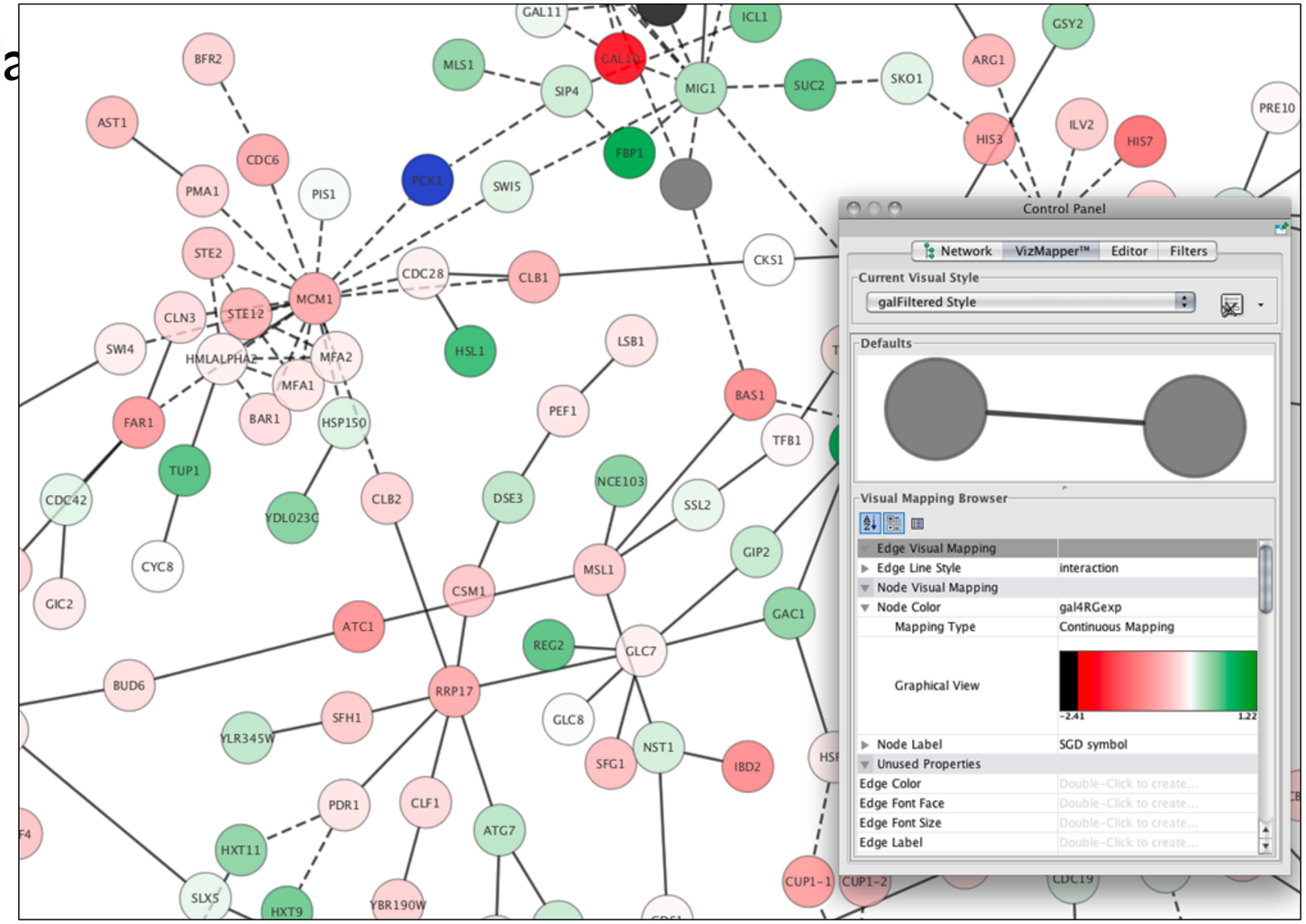
http://gephi.org



Cytoscape

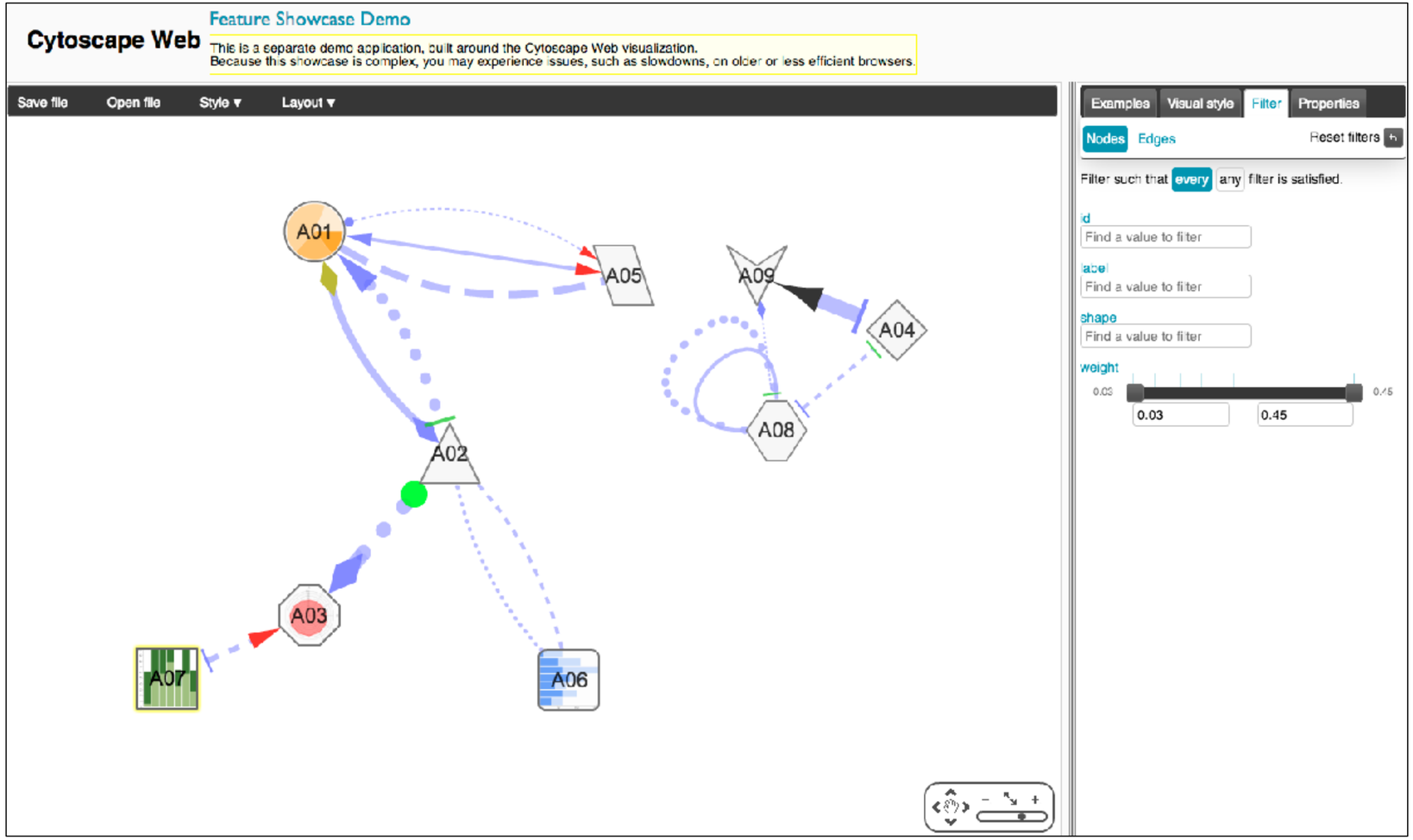
http://www.cytoscape.org/

Open source pla



Cytoscape Web

http://cytoscapeweb.cytoscape.org/



NetworkX

https://networkx.github.io/

NetworkX

NetworkX Home | Documentation | Download | Developer (Github)

High-productivity software for complex networks

NetworkX is a Python language software package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

Documentation
all documentation

Examples
using the library

Reference
all functions and methods

Features

- Python language data structures for graphs, digraphs, and multigraphs.
- Nodes can be "anything" (e.g. text, images, XML records)
- Edges can hold arbitrary data (e.g. weights, time-series)
- Generators for classic graphs, random graphs, and synthetic networks
- Standard graph algorithms
- Network structure and analysis measures
- Open source BSD license
- Well tested: more than 1800 unit tests, >90% code coverage
- Additional benefits from Python: fast prototyping, easy to teach, multi-platform

Versions

Latest Release

1.8.1 - 4 August 2013
downloads | docs | pdf

Development

1.9dev
github | docs | pdf
build passing
coverage 83%

Contact

Mailing list
Issue tracker
Developer guide