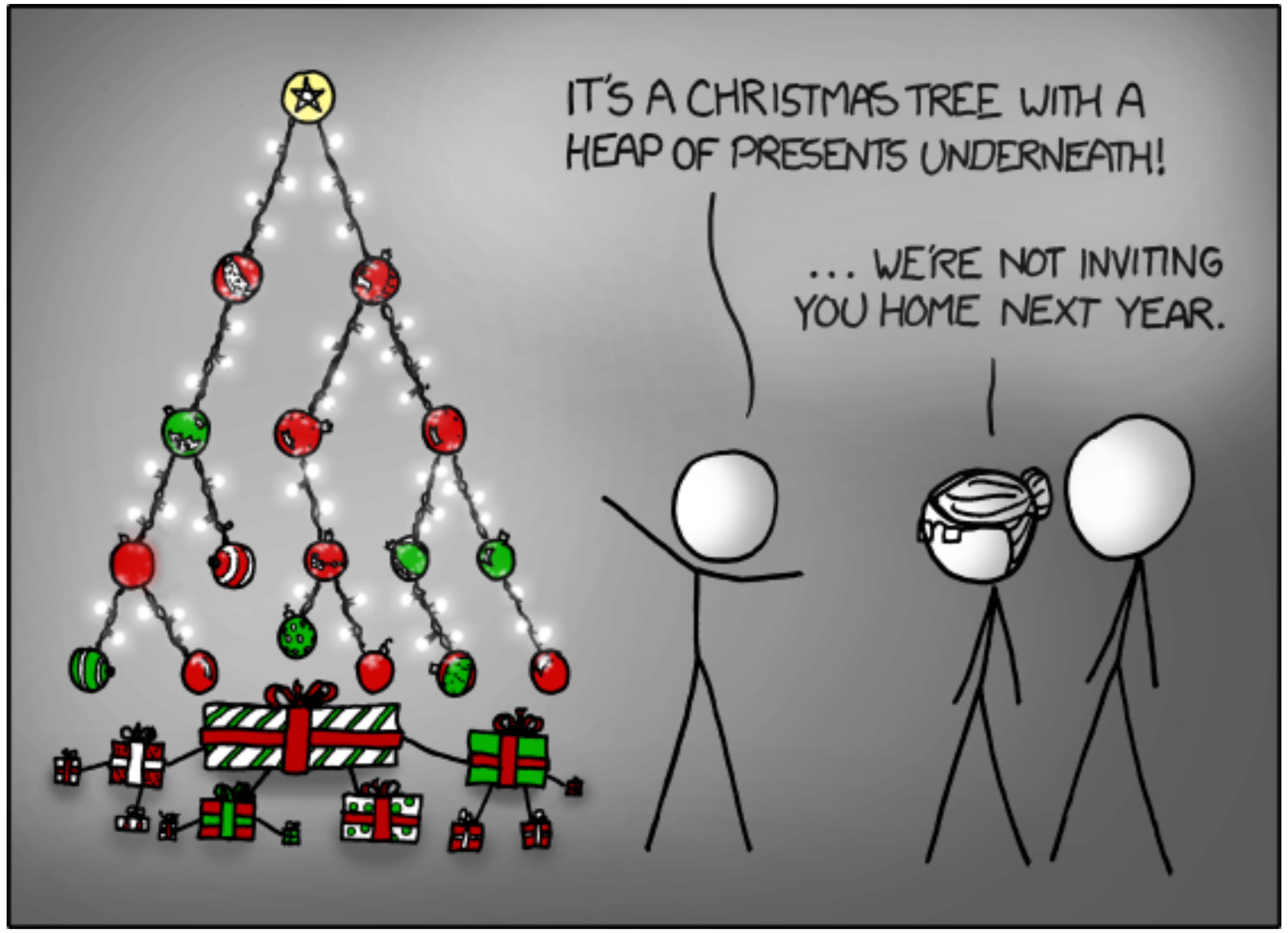
CS-5630 / CS-6630 Visualization for Data Science

Graphs

Alexander Lex
alex@sci.utah.edu

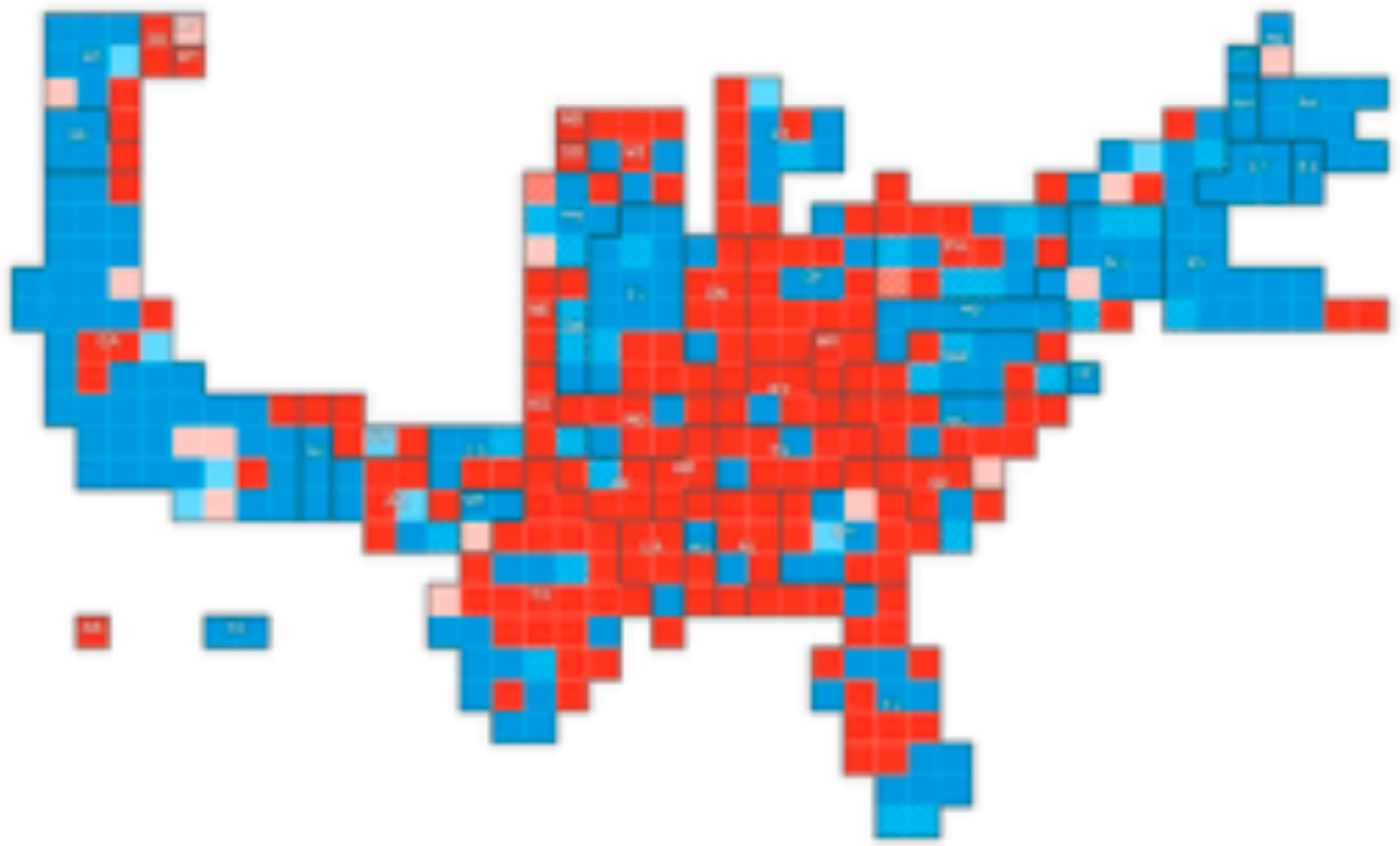
THE UNIVERSITY OF UTAH



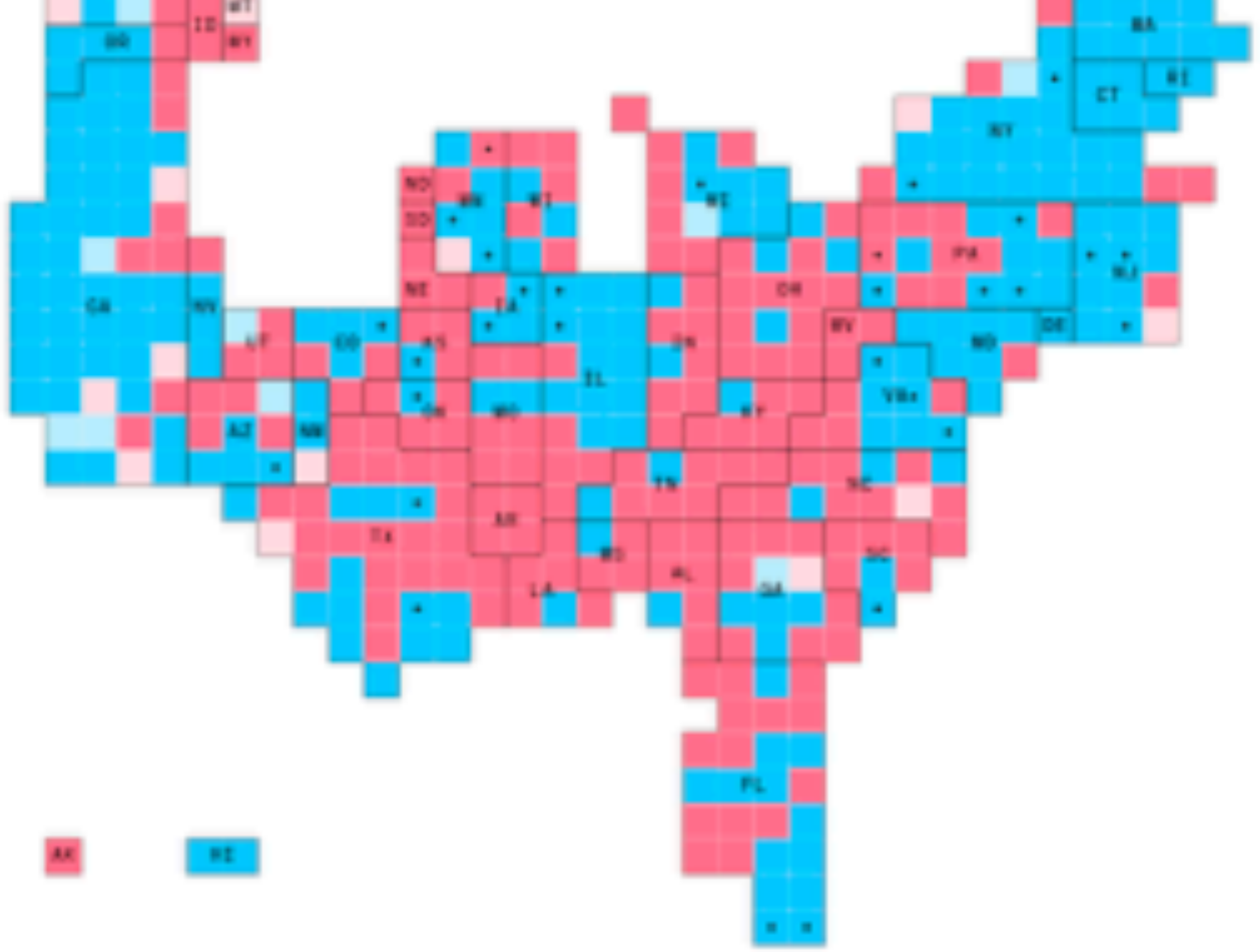
[xkcd]

The Guardian

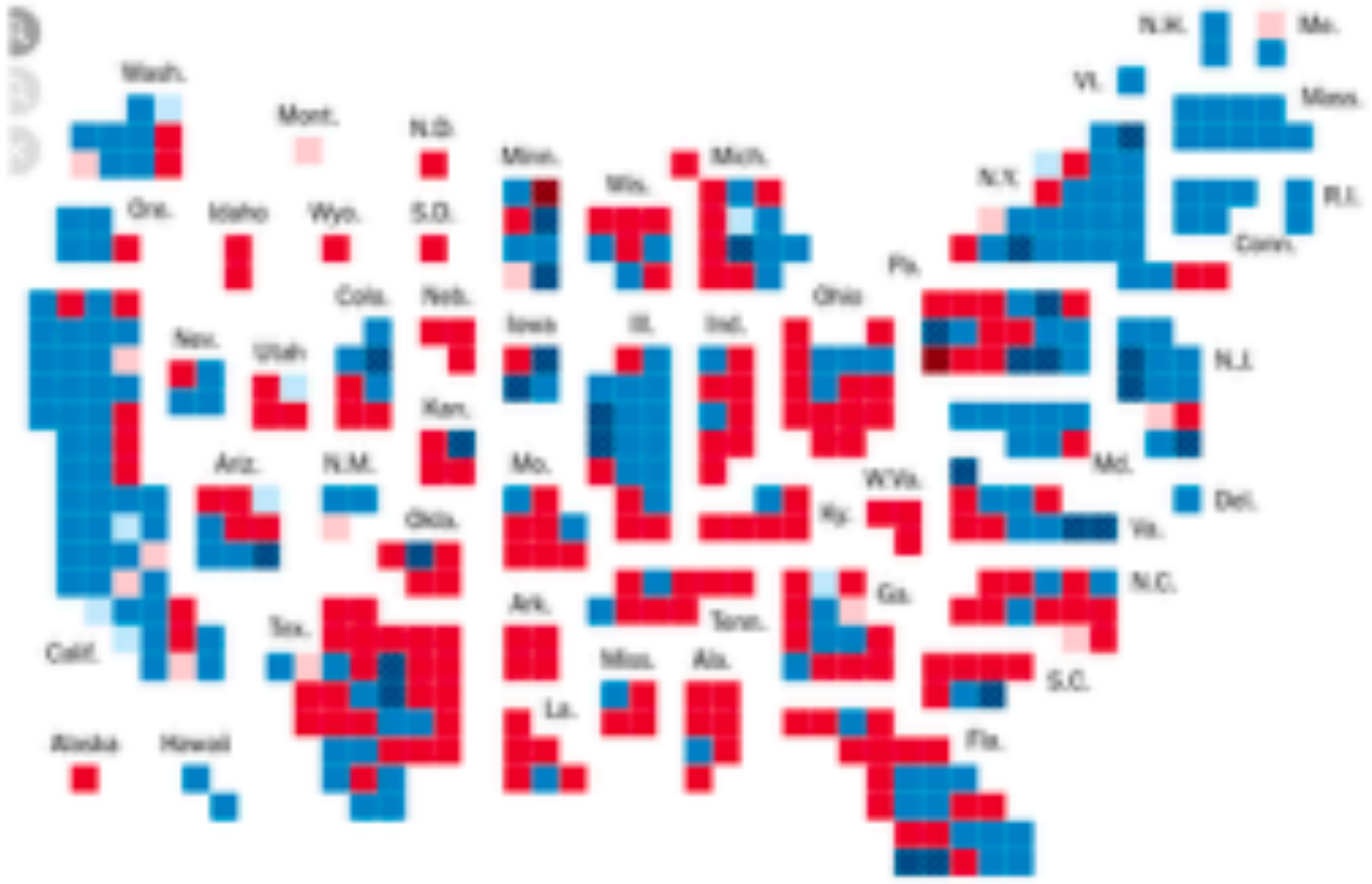
Axios*



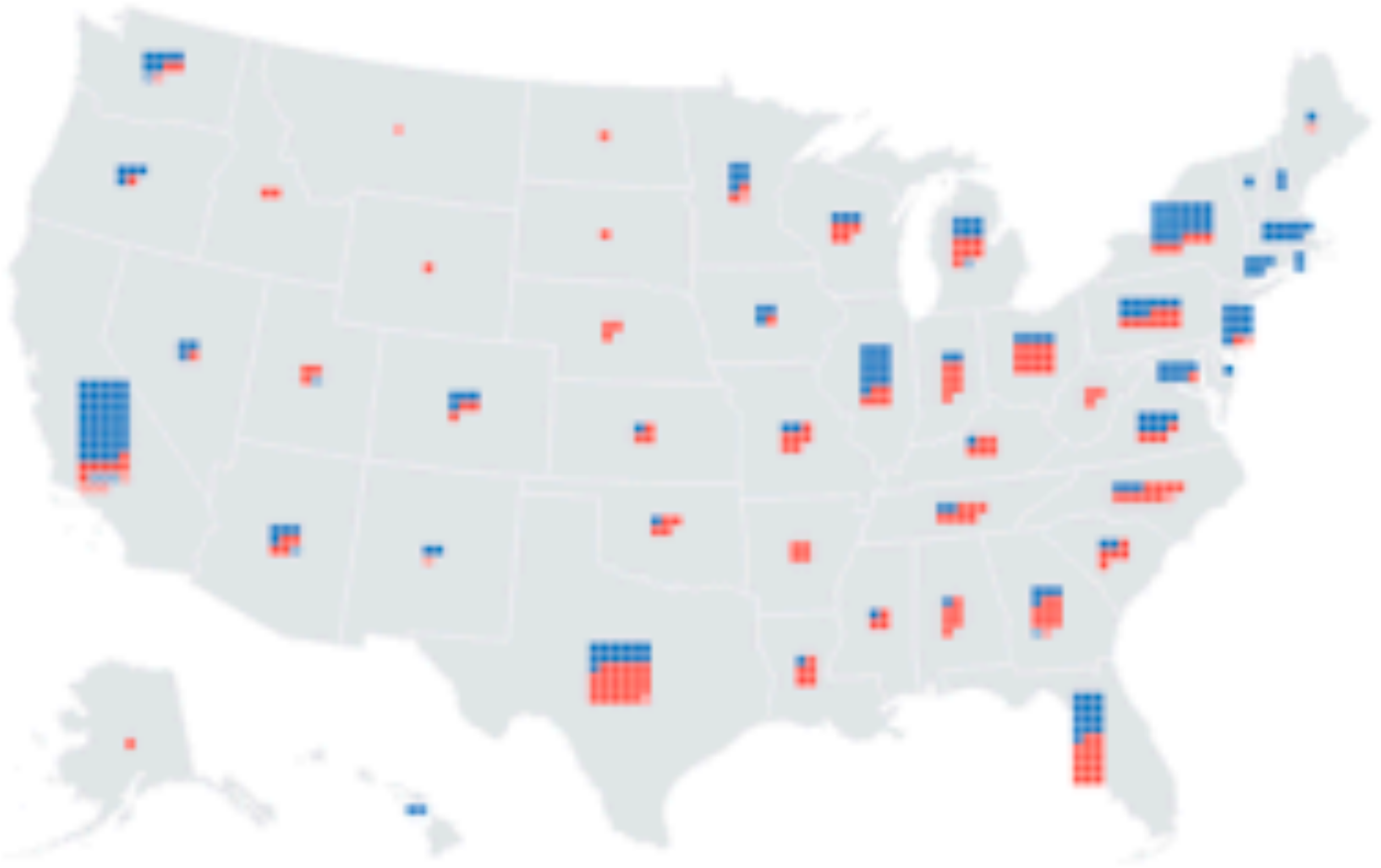
Bloomberg*



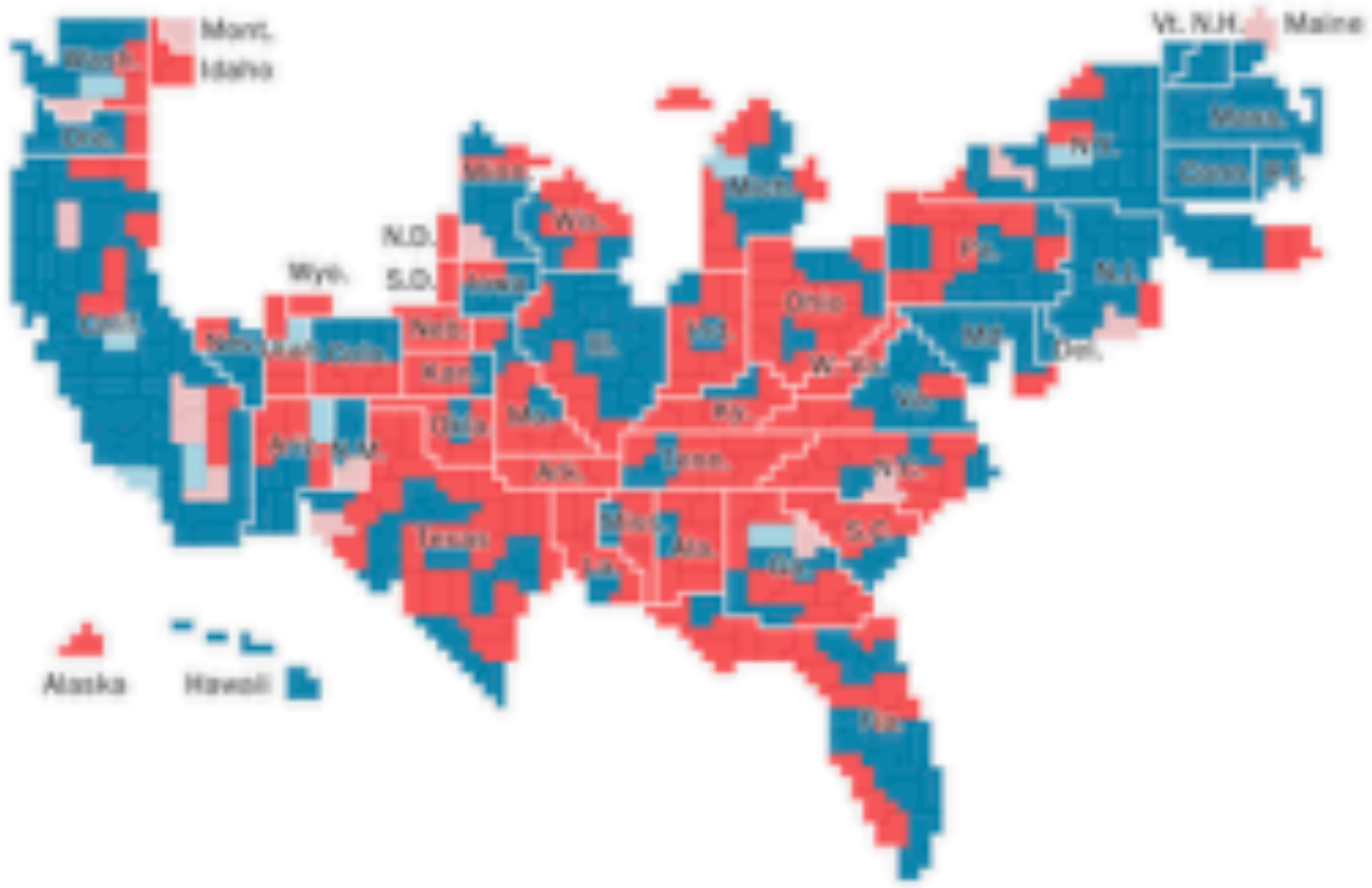
New York Times*



Politico



Washington Post*



https://twitter.com/niko_tinius/status/1060185135918866433

Graph Exercise

Nodes and Node Attributes

Author (# papers)

Carolina (6),

Miriah (42)

Alex (36),

Sean (8),

Marc (40)

Nils (51),

Silvia (110)

Links and Link Attributes

Co-author, co-author - # joint papers

Carolina, Alex - 2

Sean, Miriah - 7

Miriah, Alex - 2

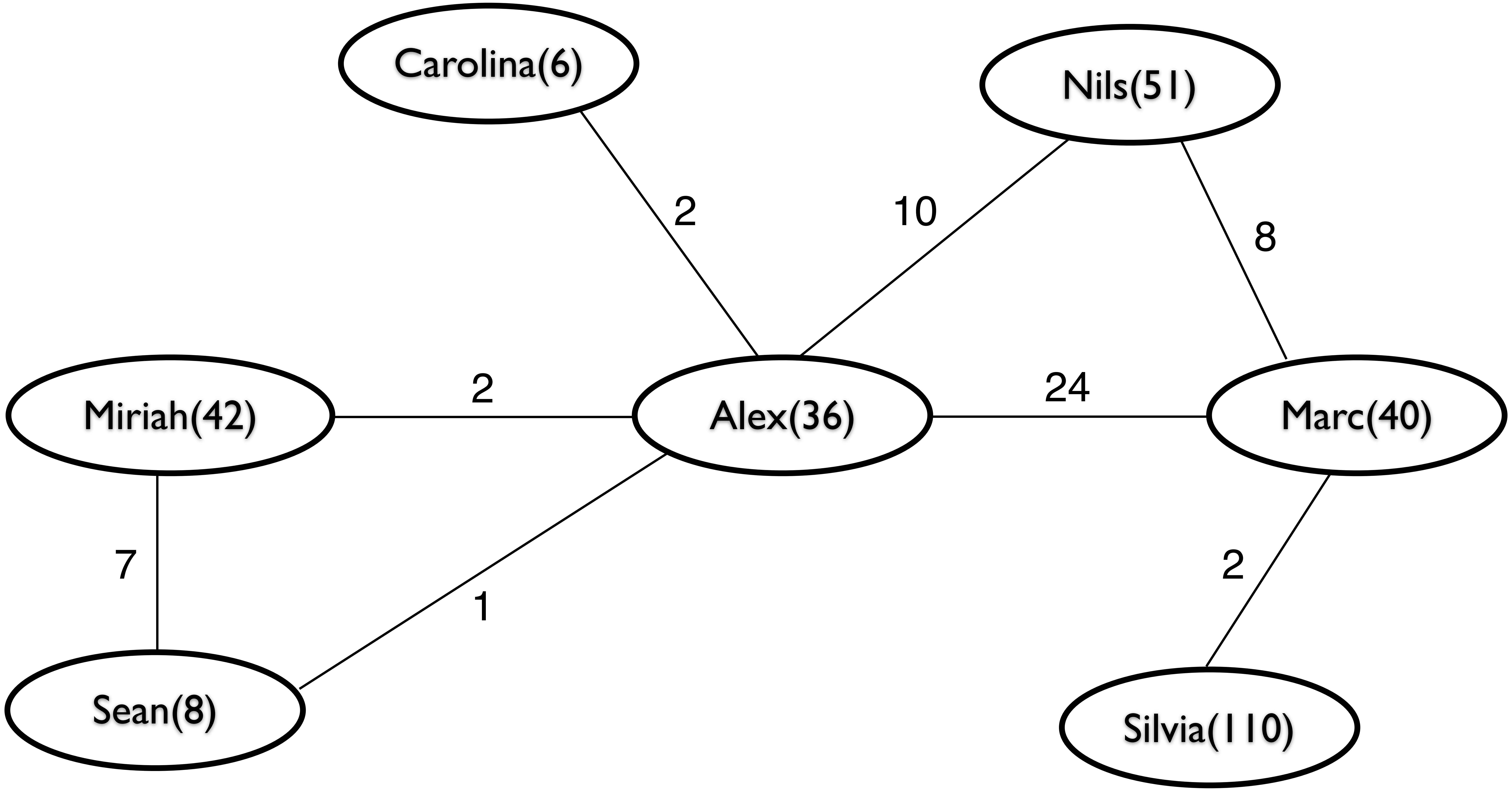
Alex, Sean - 1

Alex, Nils - 10

Alex, Marc - 24

Marc, Silvia - 1

Marc, Nils - 8

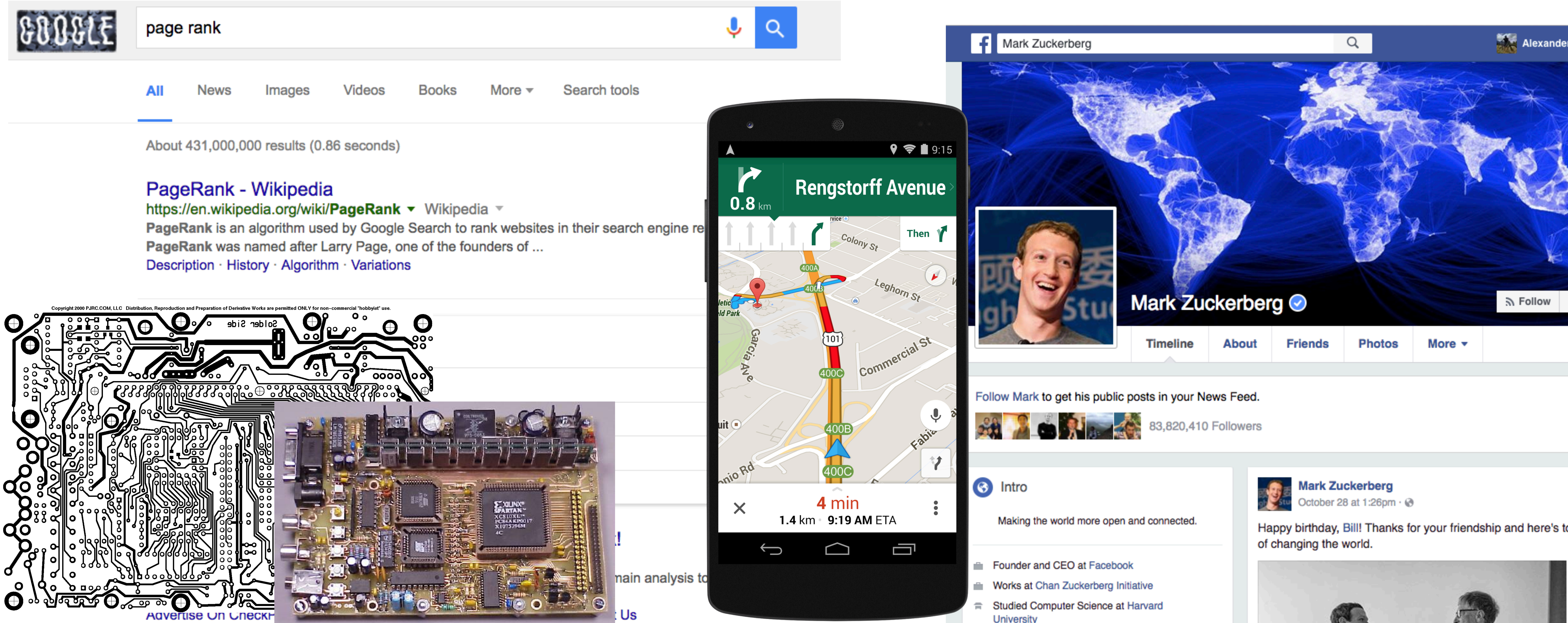

Carolina(6)
Nils(51)
2
10
8
Miriah(42)
2
Alex(36)
24
Marc(40)
7
1
2
Sean(8)
Silvia(110)

	Carolina (6)	Miriah (42)	Alex (36)	Sean (8)	Marc (40)	Nils (51)	Silvia (110)
Carolina (6)			2				
Miriah (42)			2	7			
Alex (36)	2	2		1	14	10	
Sean (8)		7	1				
Marc (40)			14			8	1
Nils (51)			10		8		
Silvia (110)					1		

Graphs

Applications of Graphs

Without graphs, there would be none of these:`




facebook
December 2010

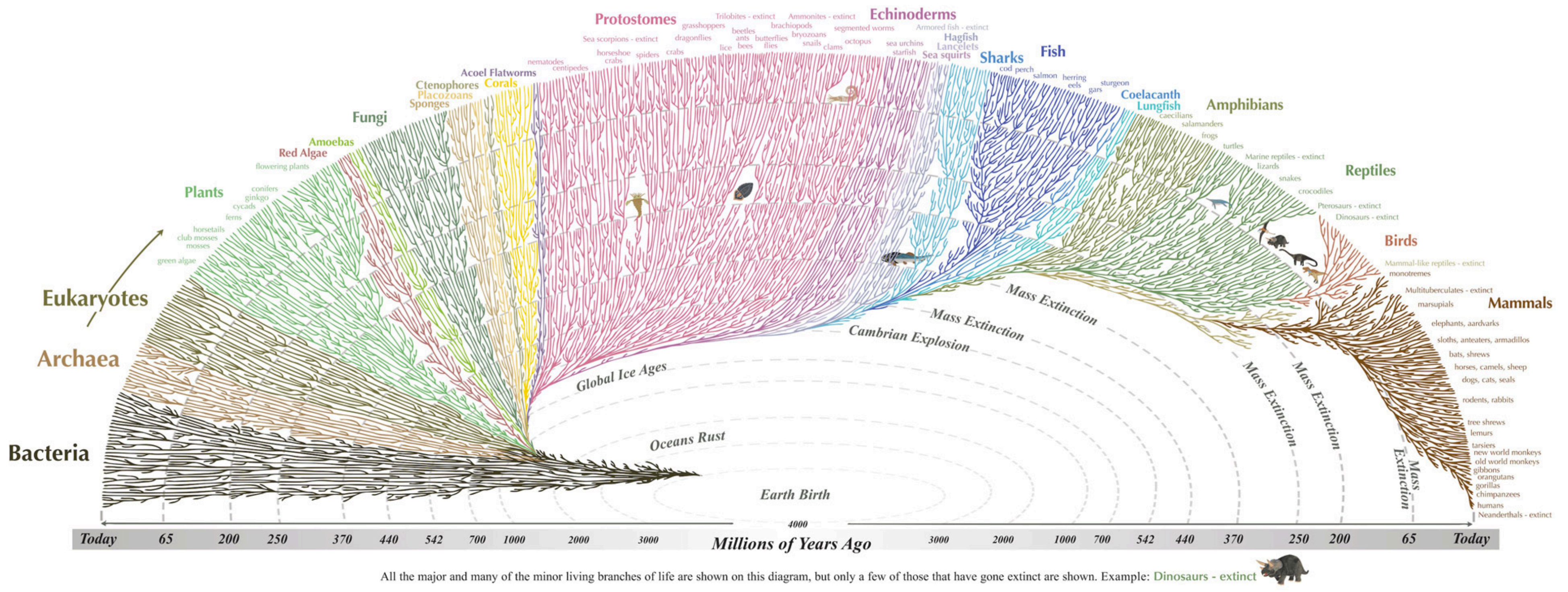
Biological Networks

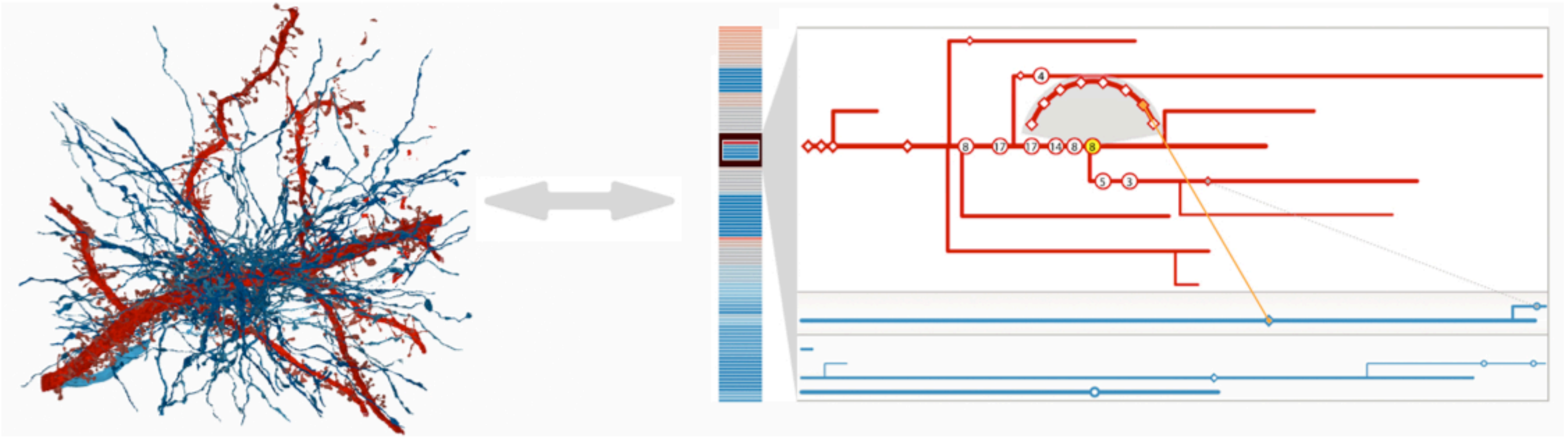
Interaction between genes, proteins and chemical products

The brain: connections between neurons

Your ancestry: the relations between you and your family

Phylogeny: the evolutionary relationships of life



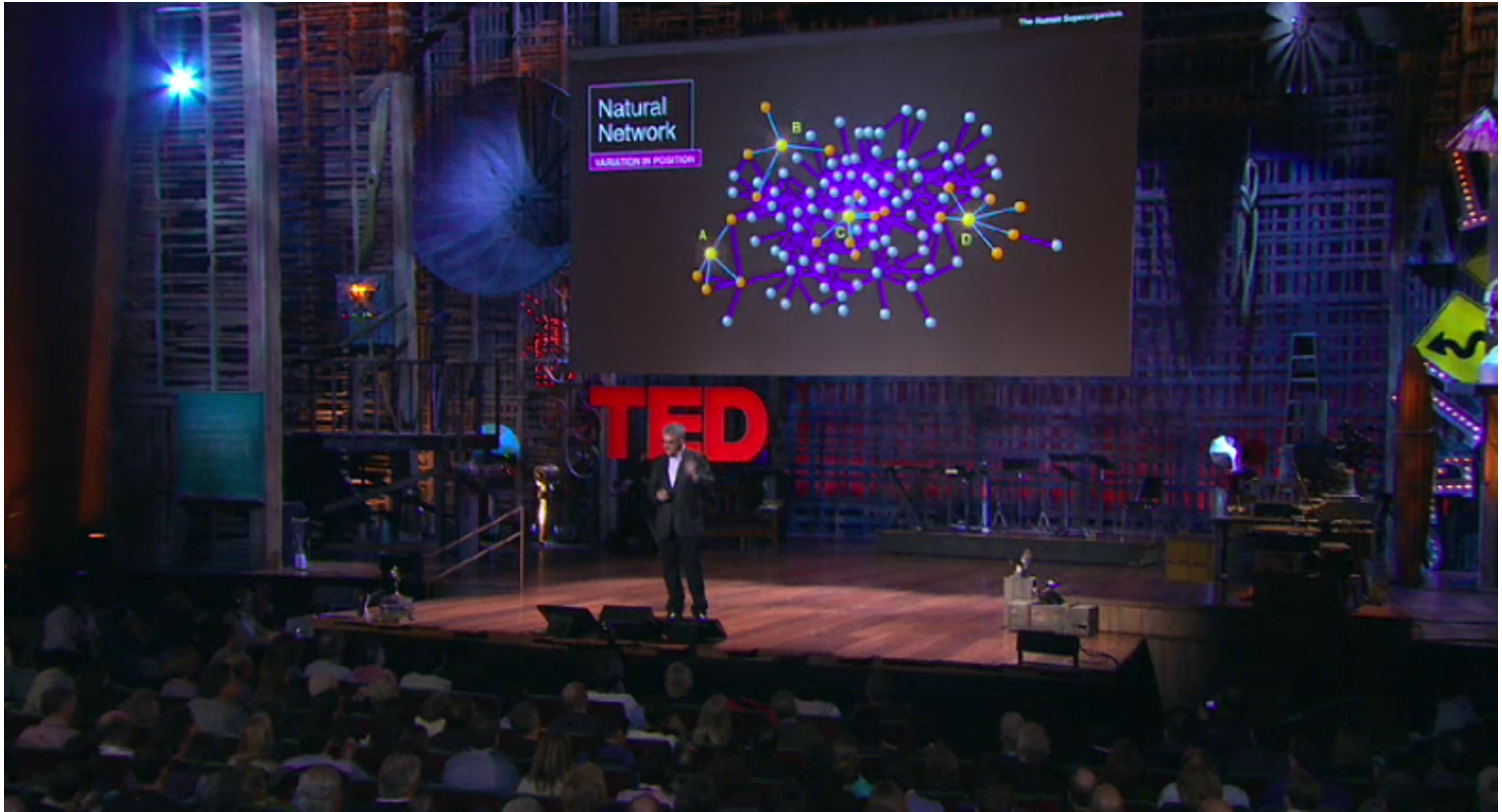


[Beyer 2014]

CITRATE
ATP
MALONATE, OXALO-ACETATE
SUCCINATE
CoA-SH
SUCCINYL-CoA HYDROLASE
PPi, ATP
H₂O
PROTEINS
GLYCYL-tRNA
GLYCINE-tRNA LIGASE
ACETYL-CoA
PyrP
5-AMINOLEVULINATE SYNTHASE
GLYCINE
increased in acute intermittent porph
5-AMINO-LEVULINATE SYNTHASE
FUMARATE
SUCCINATE DEHYDROGENASE
FADH₂ · E
FAD · E
ACCEPTOR (UBIQUINONE)
ACC · H₂ (UBIQUINOL)
ATP
ADP
Pi
GTP (ITP)
CoA-SH
GDP (IDP)
SUCCINYL-CoA LIGASES
SUCCINYL-CoA
HEME
hv (in plants and photo-bacteria)
CO₂
NADH + H⁺
FORMATE DEHYDROGENASE
NAD⁺
FORMATE DEHYDRO-GENASE (CYTOCHROME)
MOLYBDO-PTERIN
2 Cytb-Fe⁺⁺
DIHYDRO-LIPOAMIDE SUCCINYL TRANS-FERASE
DIHYDRO-LIPO-AMIDE DEHY-DRO-GENASE
FAD · E
NADH + H⁺
FORMYL-CoA
FORMYL-CoA HYDRO-LASE
FORMATE
H₂O
CoA-SH
OXALATE DECARBOXY-LASE
OXALYL-CoA DECARBOXY-LASE
OXALATE-CoA-TRANSFERASE
OXO-GLUTARATE SYNTHASE
FERRE-DOXIN (red)
FERRE-DOXIN (ox.)
Deficient in hyper-oxaluria I
PYRUVATE
ALANINE-GLYOXYLATE TRANS-AMINASE
ThPP · E
FADH₂ · E
NADH
ACTH

Michal 2000

Graph Analysis Case Study

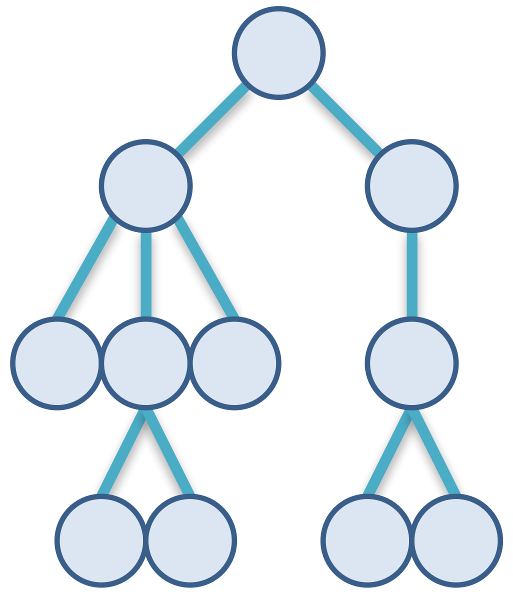


Graph Theory Fundamentals

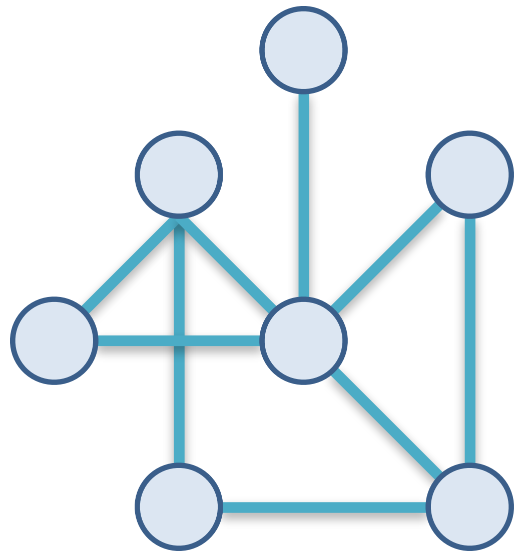
See also “Network Science”, Barabasi

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

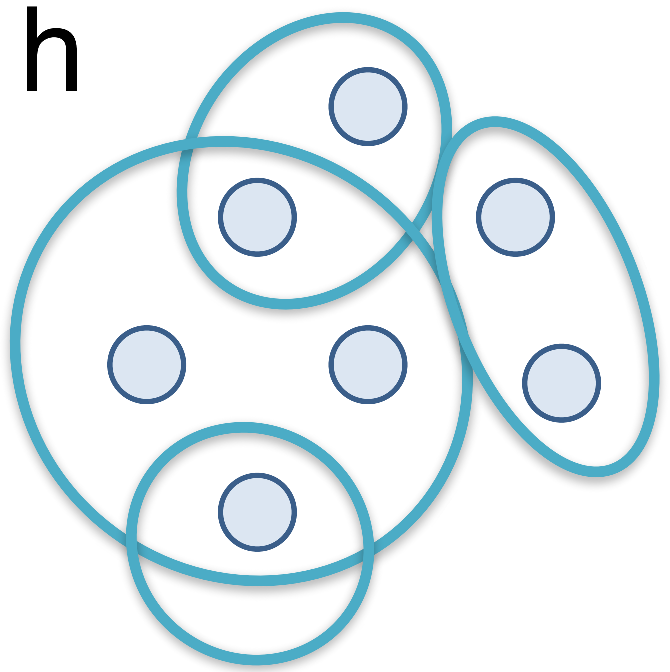
Tree



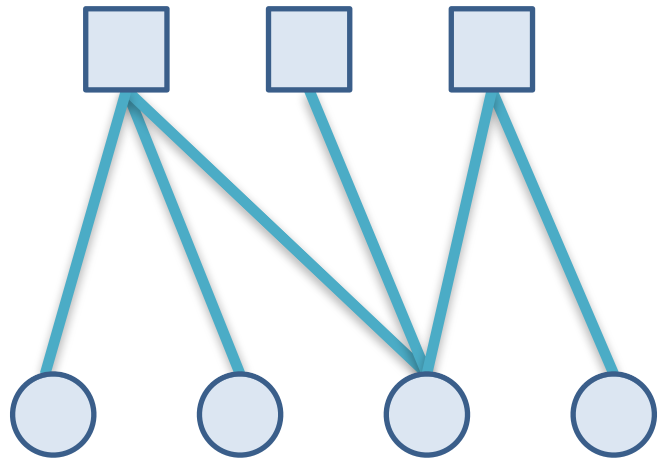
Network



Hypergrap
h

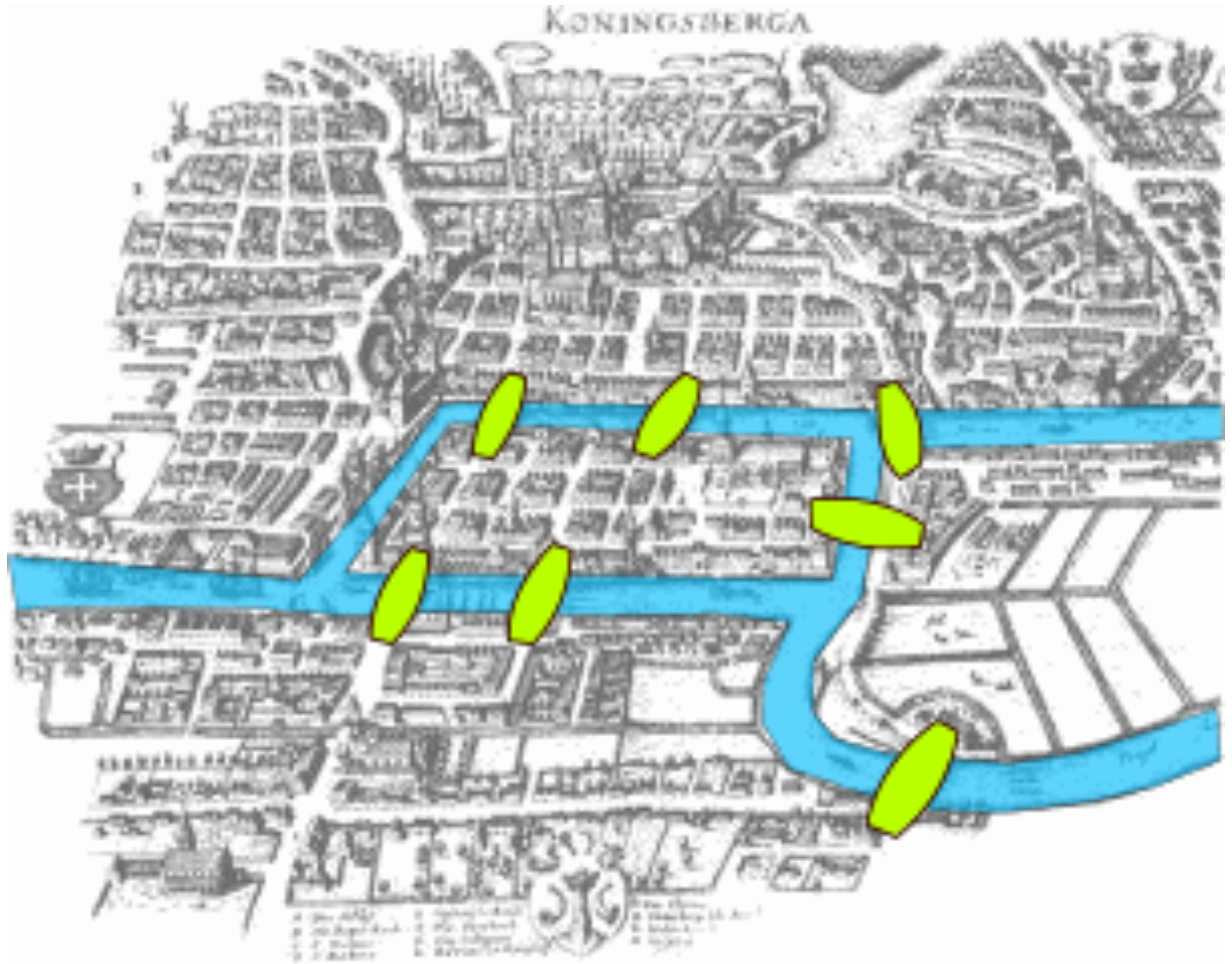


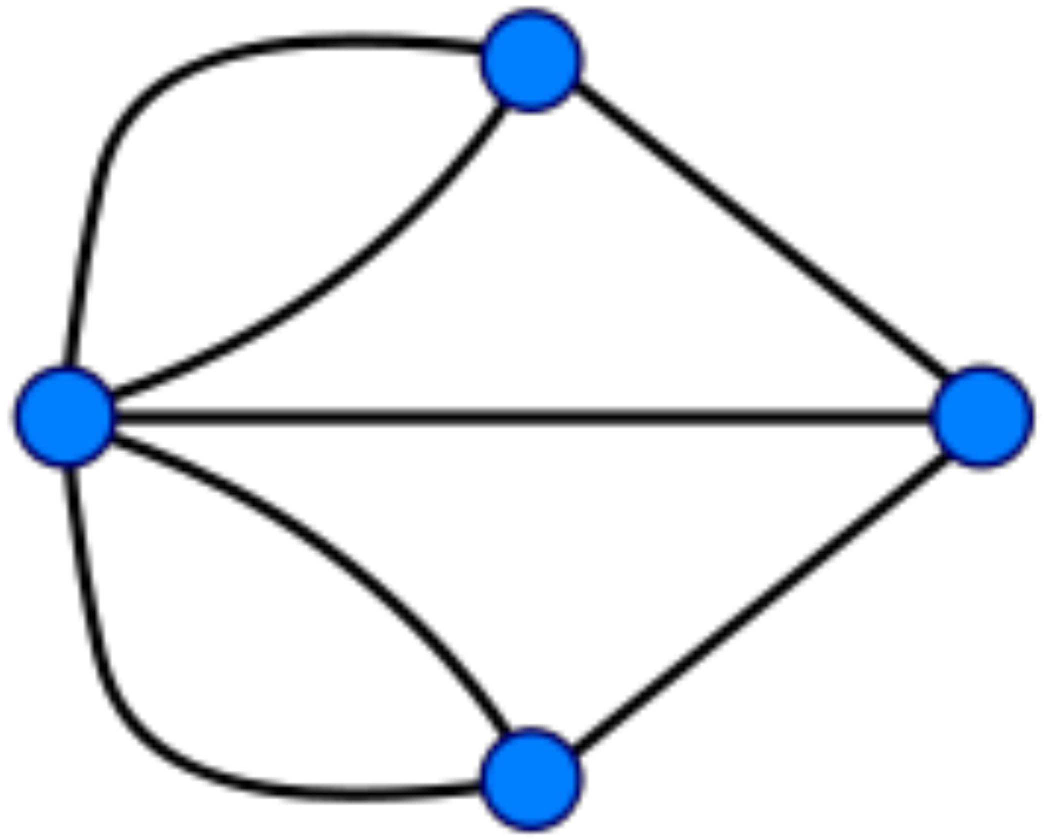
Bipartite Graph



Königsberg Bridge Problem (1736)

Can you take a walk and visit every land mass without crossing a bridge twice?





Leonhard Euler:
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.

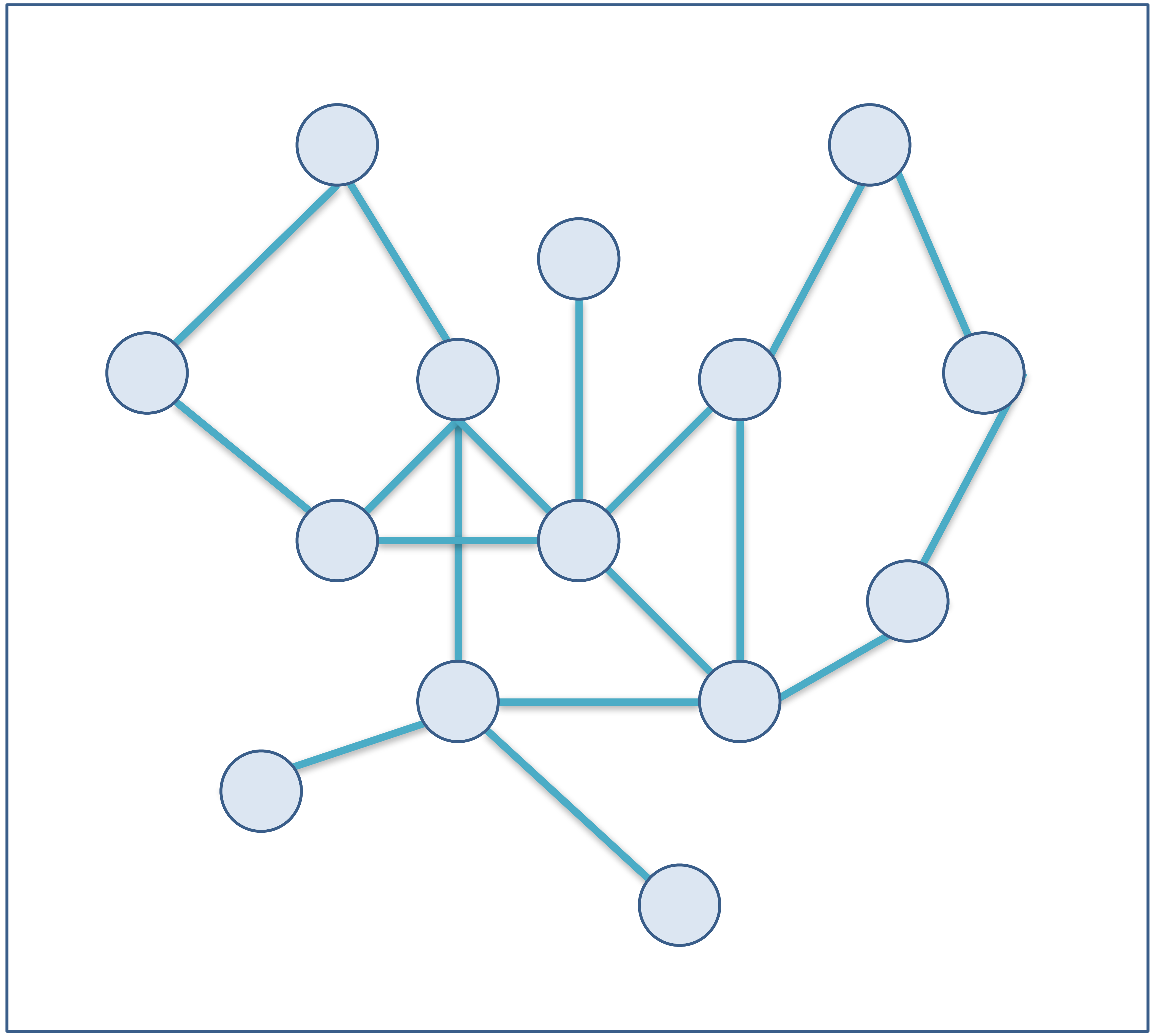
http://barabasi.com/networksciencebook/chapter/2#bridges

Graph Terms

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

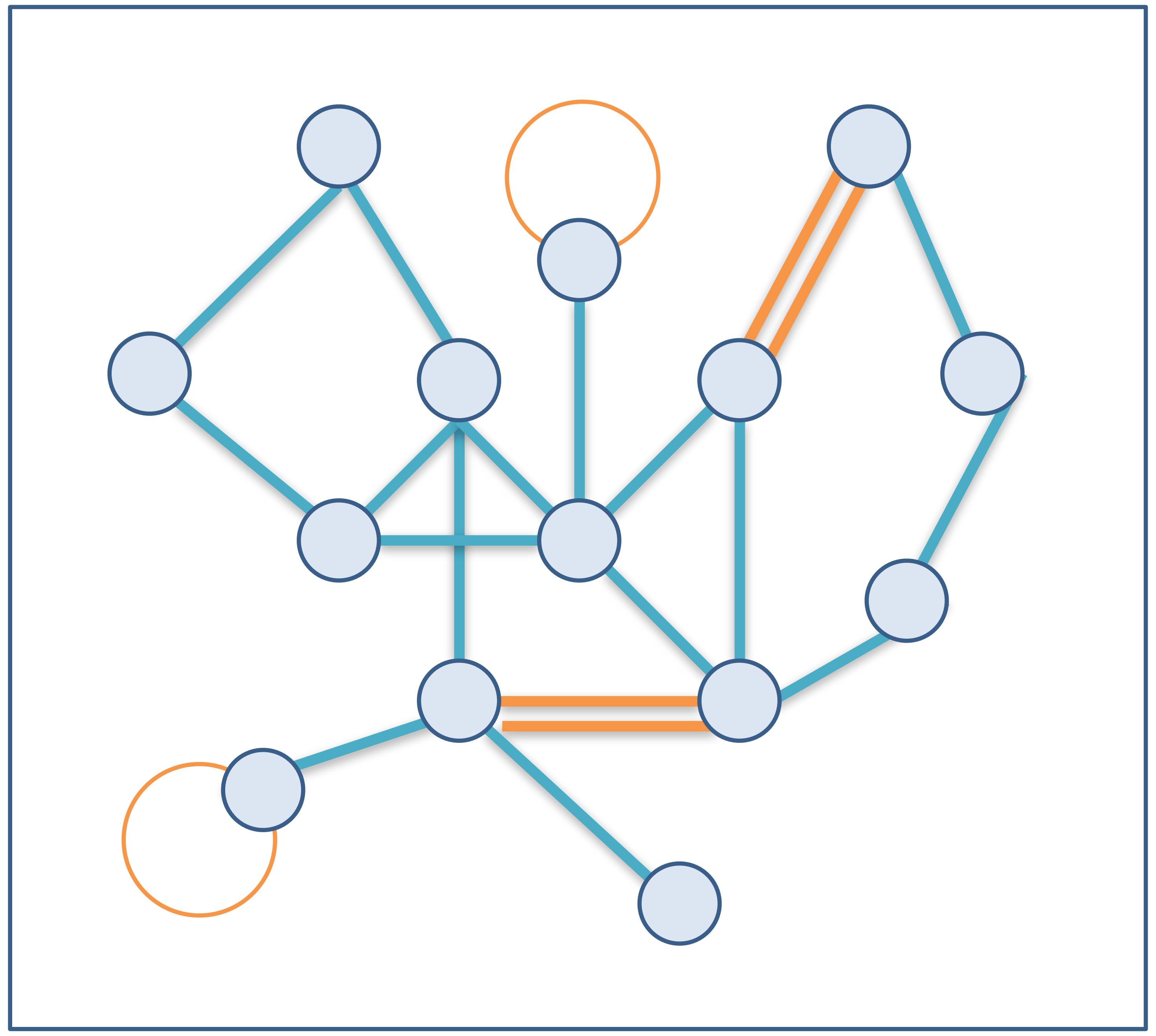
set of **edges E** (also called links) connecting these vertices.

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**



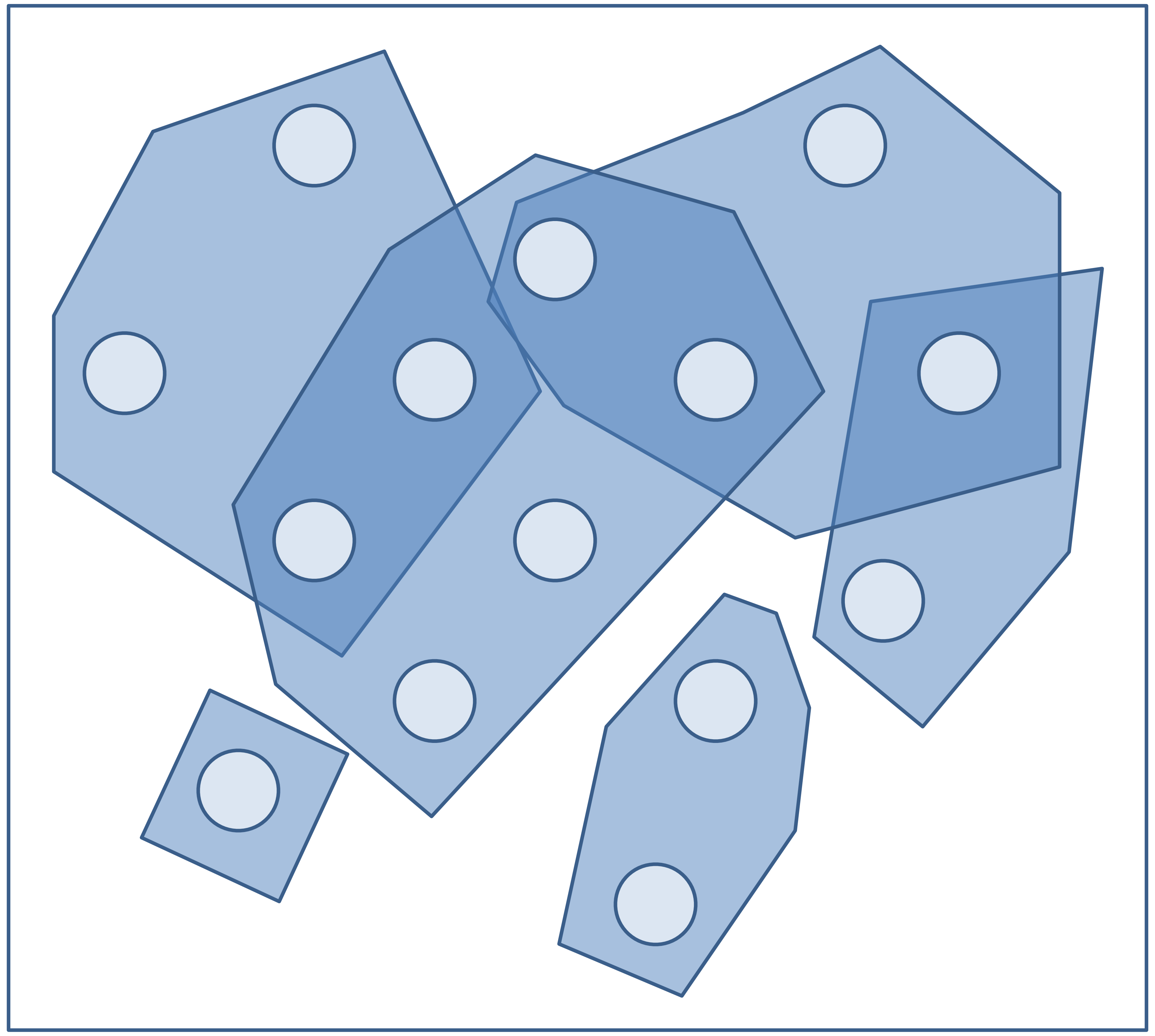
Not a simple graph!
→ A ***general graph***

Graph Term: Directed Graph

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

Graph Terms: Hypergraph

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



Hypergraph Example

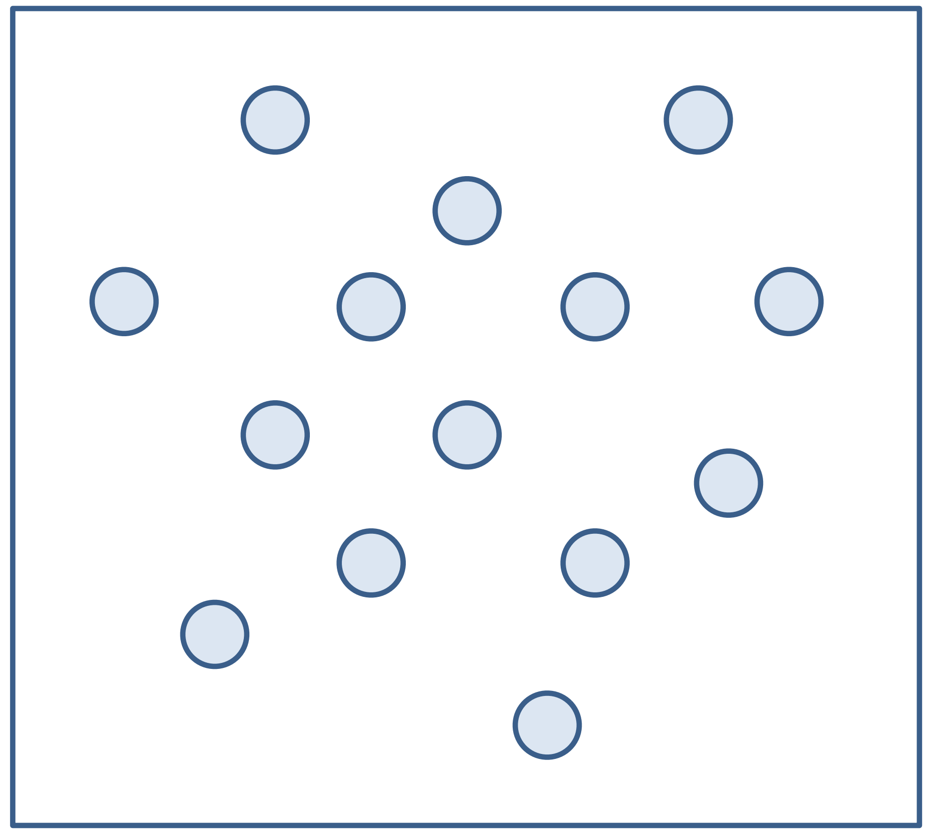
Graph Terms

Independent Set

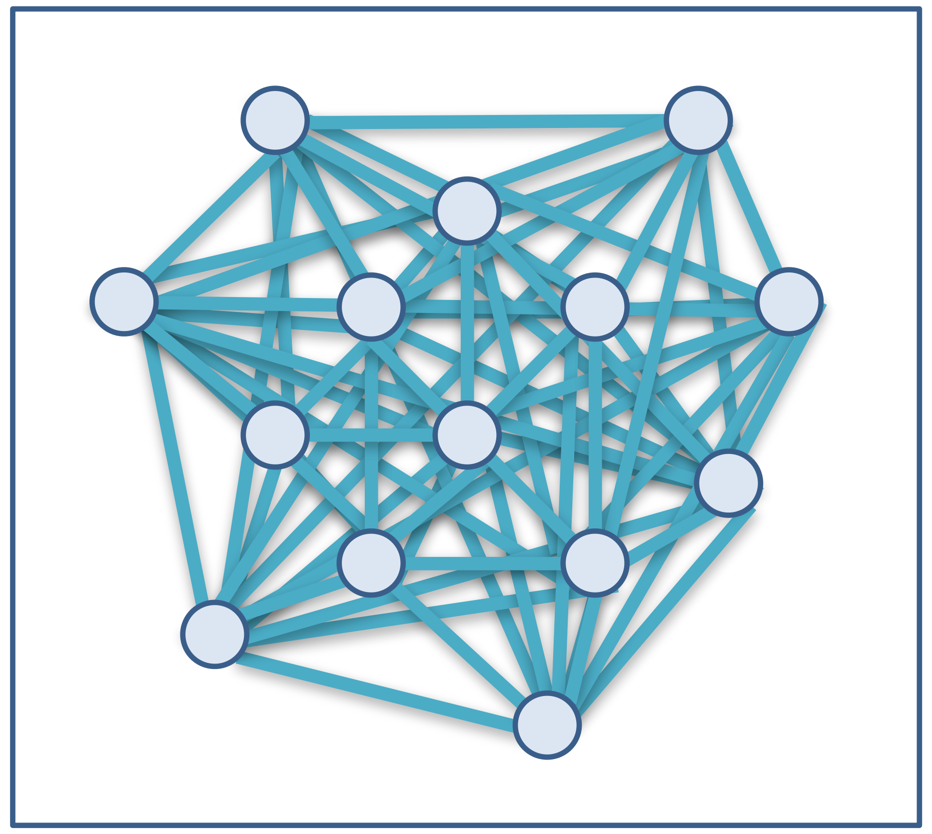
G contains no edges

Clique

G contains all possible edges



Independent Set

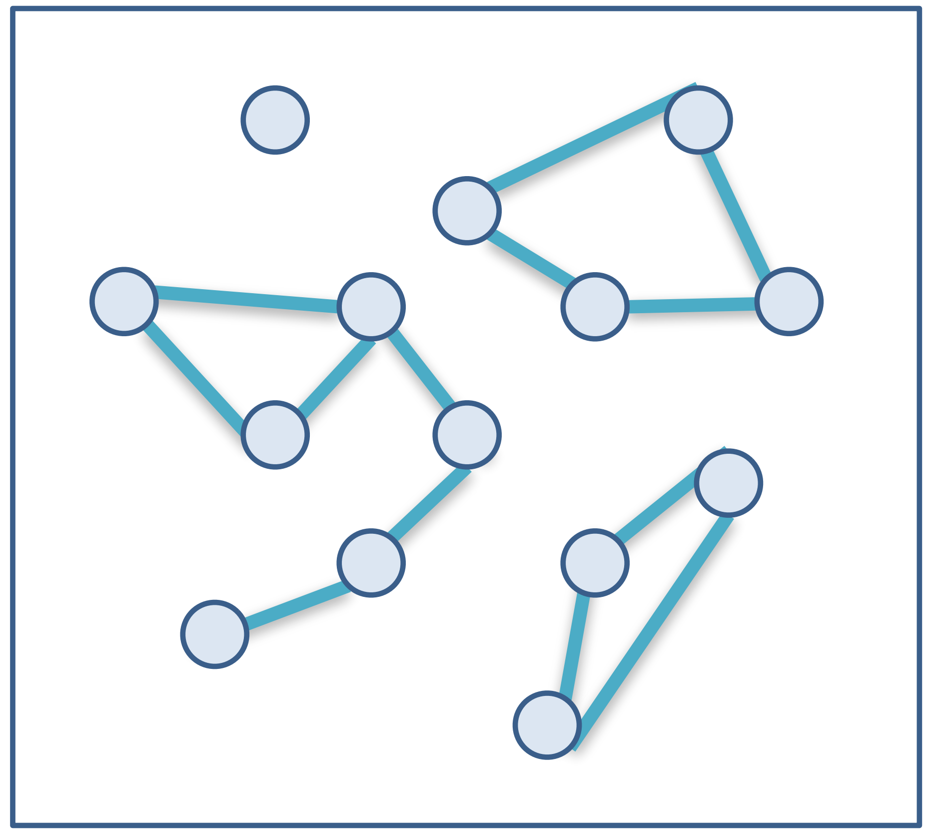


Clique

Unconnected Graphs, Articulation Points

Unconnected graph

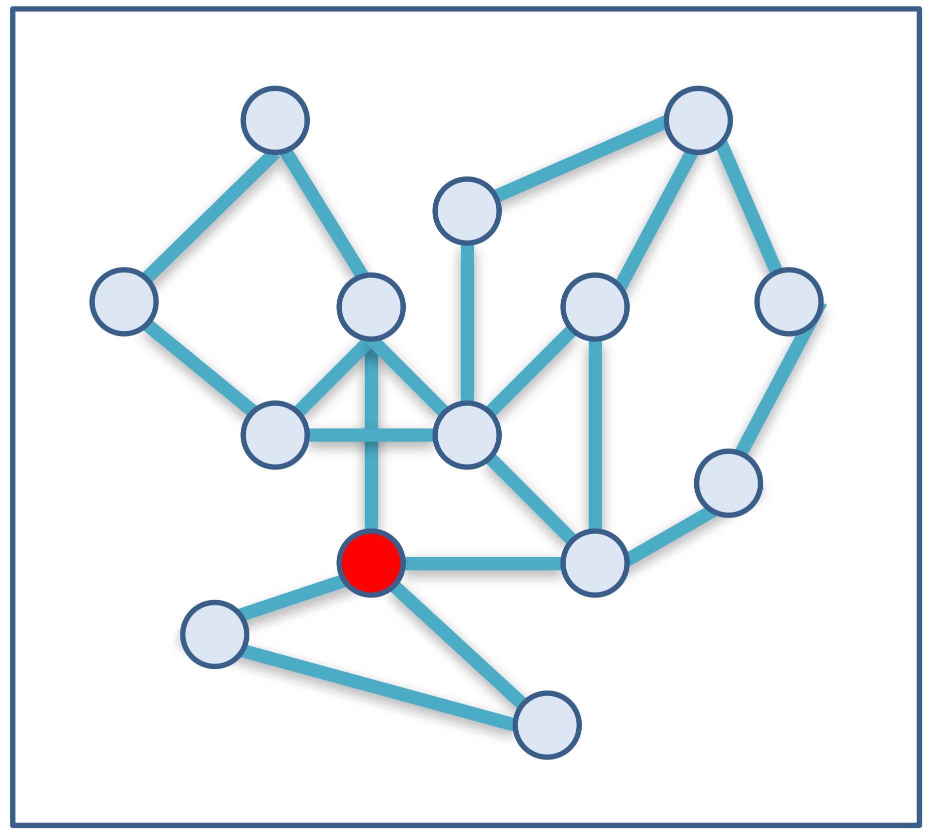
An edge traversal starting from a given vertex cannot reach any other vertex.



Unconnected Graph

Articulation point

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

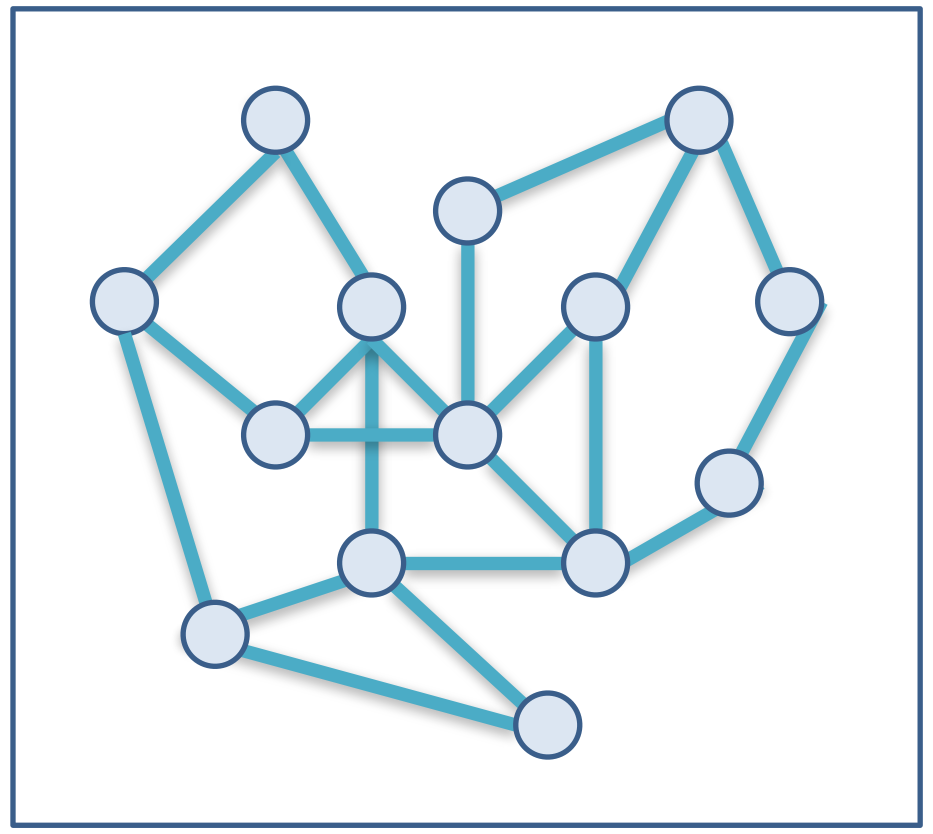


Articulation Point (red)

Biconnected, Bipartite Graphs

Biconnected graph

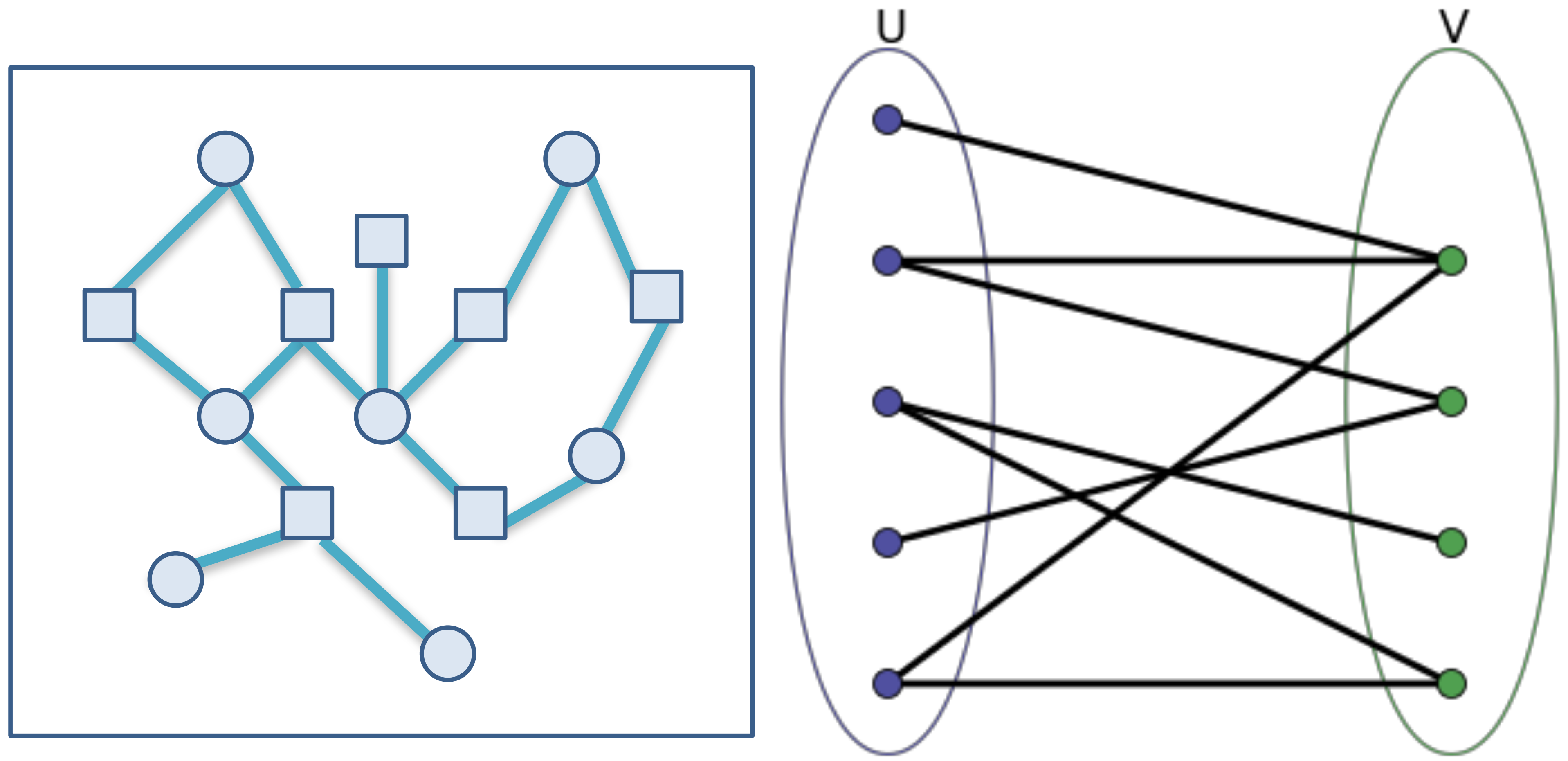
A graph without articulation points.



Biconnected Graph

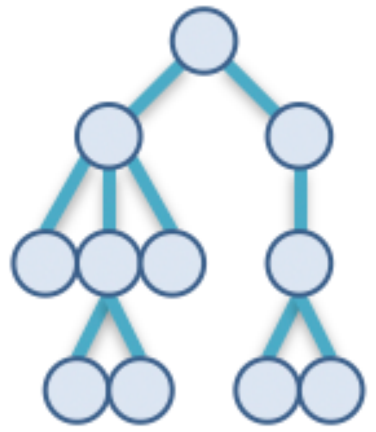
Bipartite graph

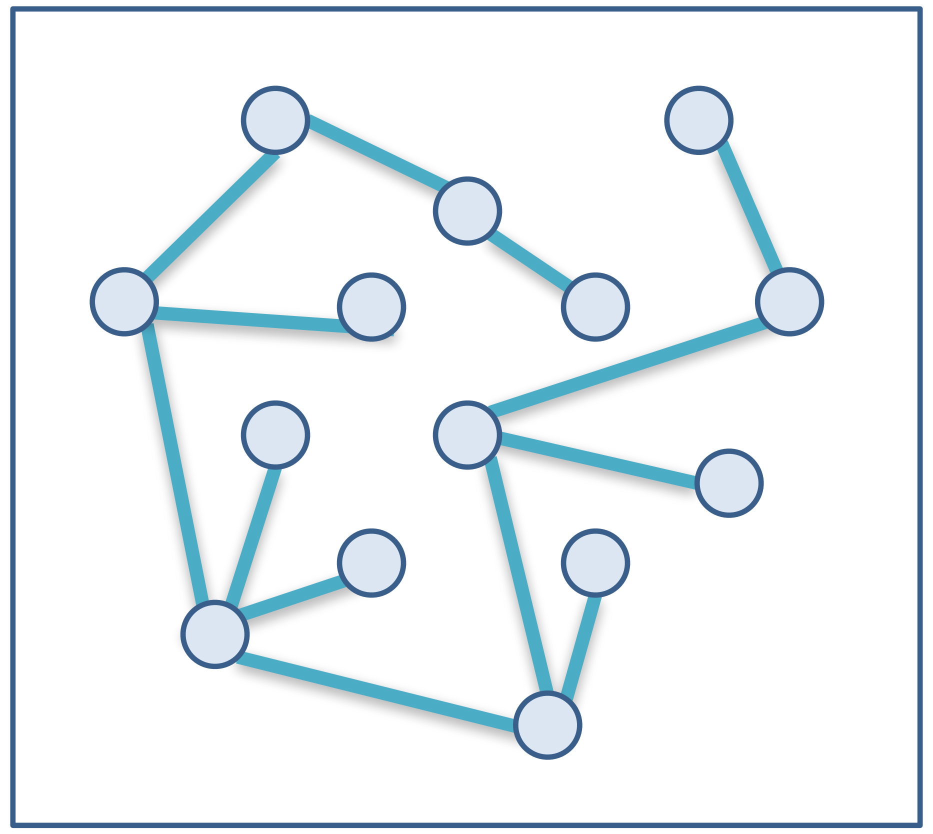
The vertices can be partitioned in two independent sets.



Bipartite Graph

Tree



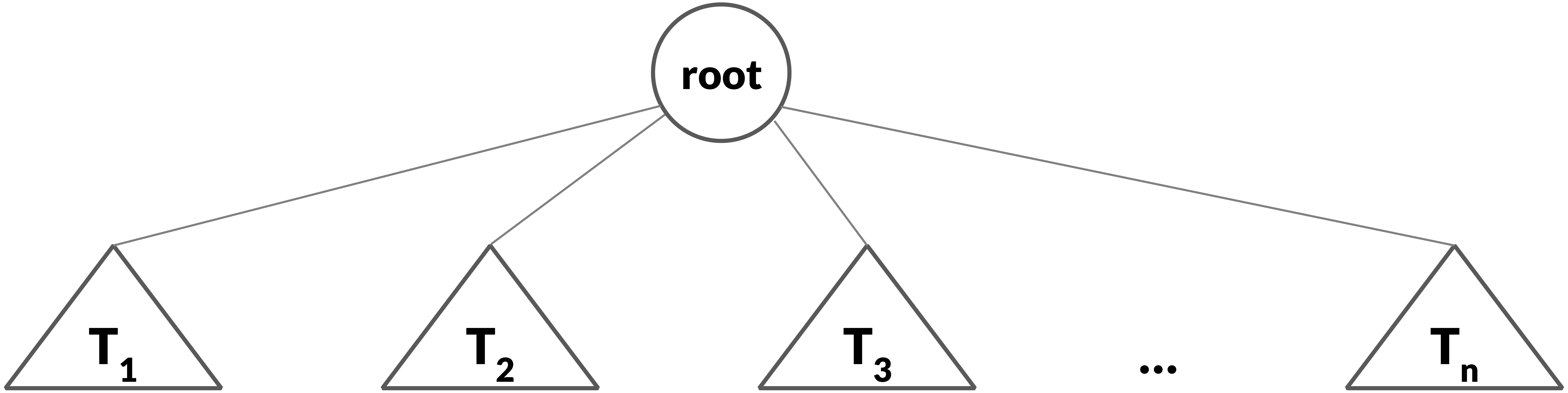


A graph with no cycles - or:

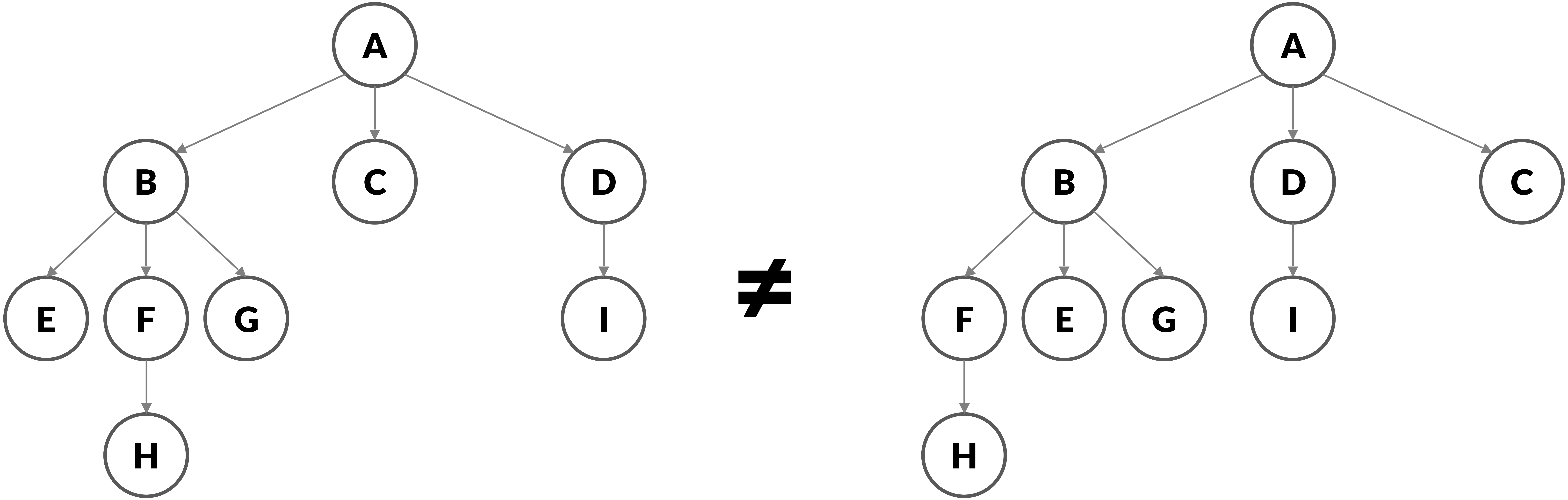
A collection of nodes

contains a root node and 0-n subtrees

subtrees are connected to root by an edge



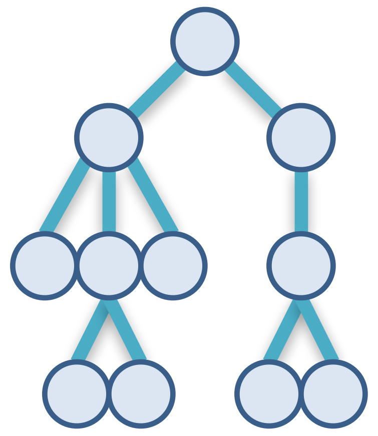
Ordered Tree



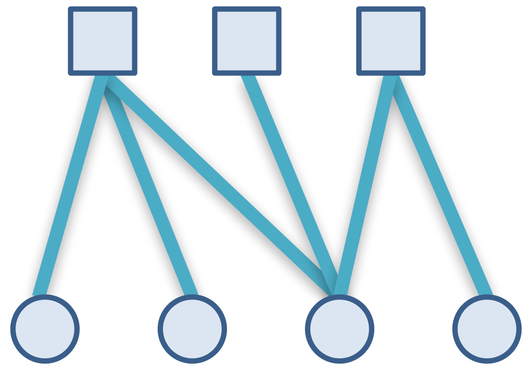
Different Kinds of Graphs

Over 1000 different graph classes

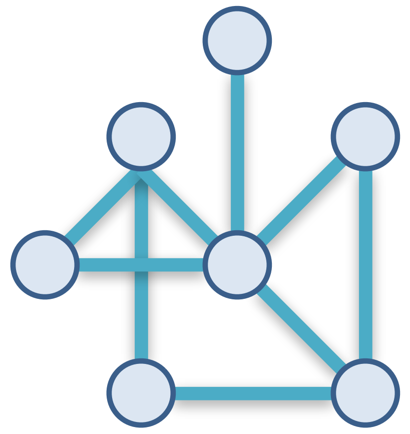
Tree



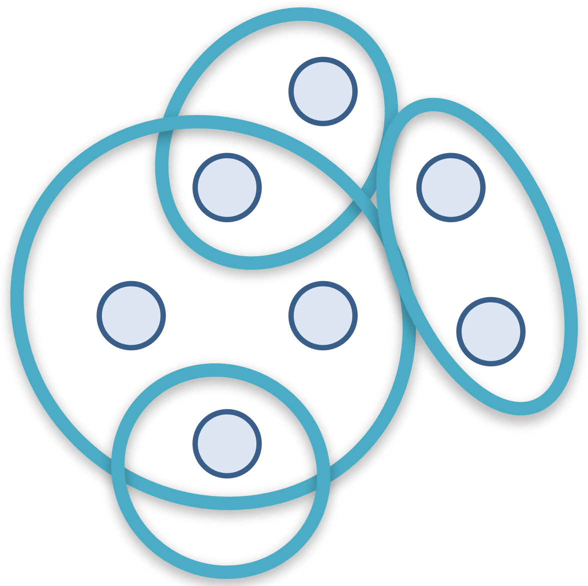
Bipartite Graph

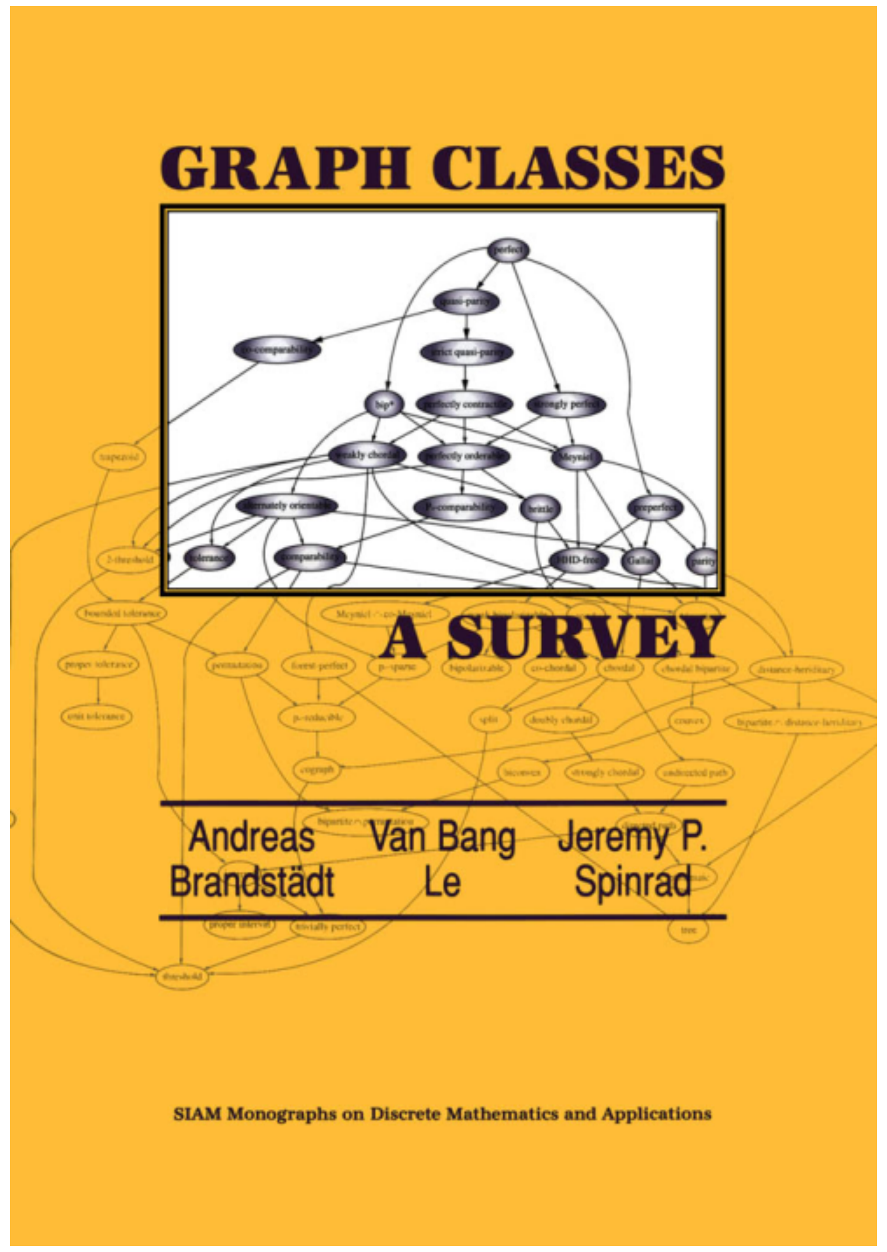


Network



Hypergraph





A. Brandstädt et al. 1999

Degree

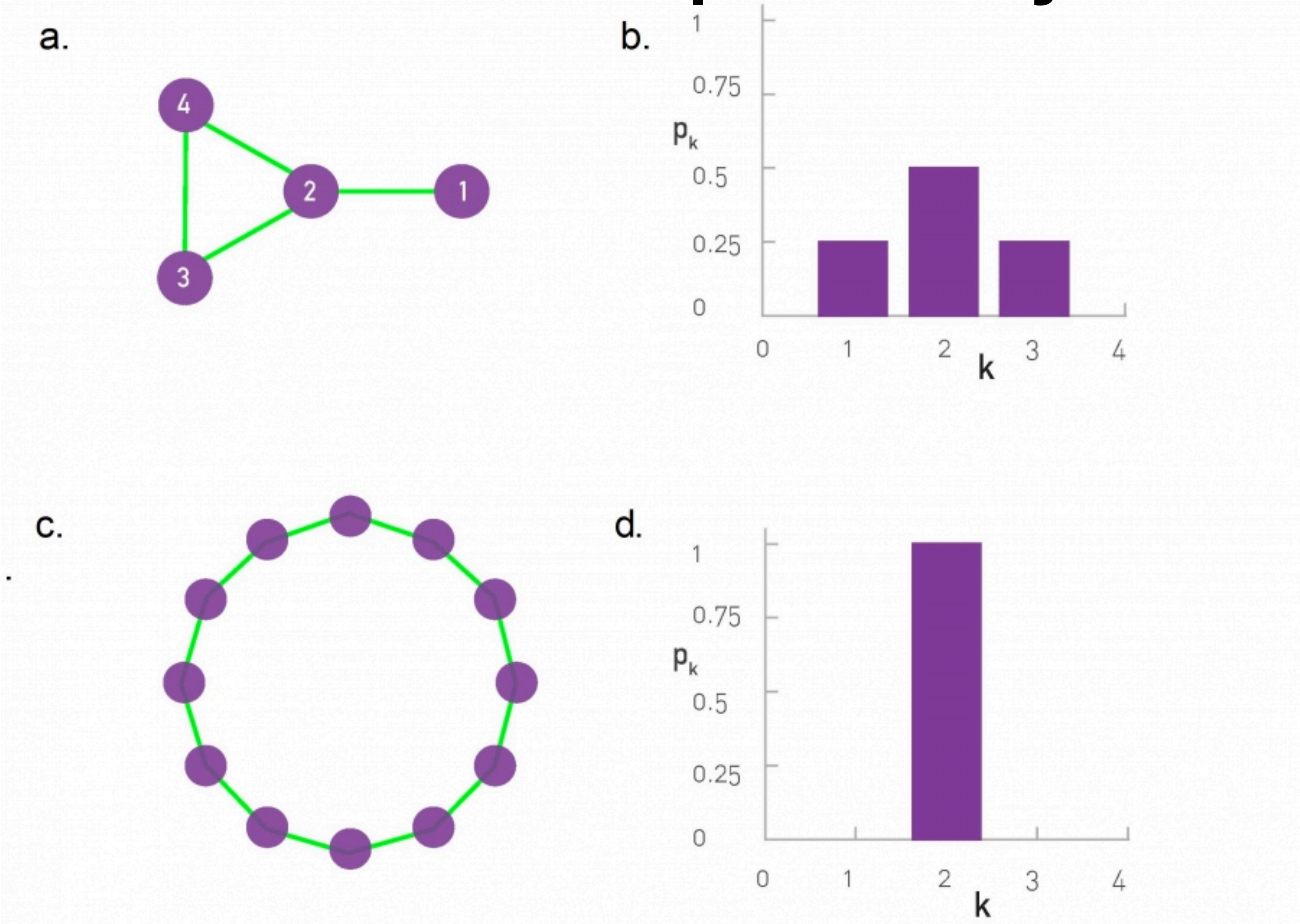
Node degree deg(x)

The number of edges connecting a node. For directed graphs in- and out-degree are considered separately.

Average degree

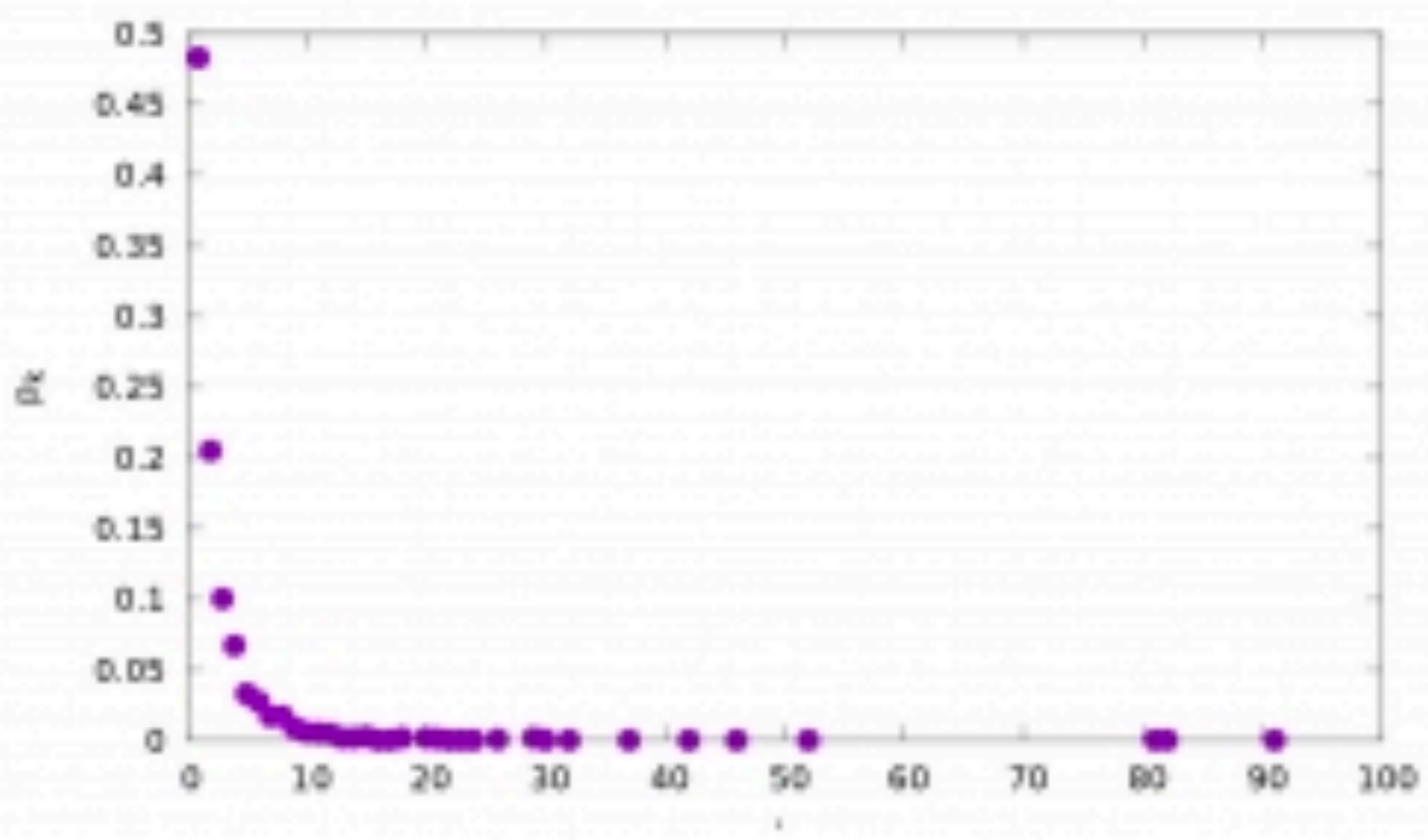
$$\langle k \rangle = \frac{1}{N}\sum_{i=1}^{N} k_i = \frac{2L}{N}$$

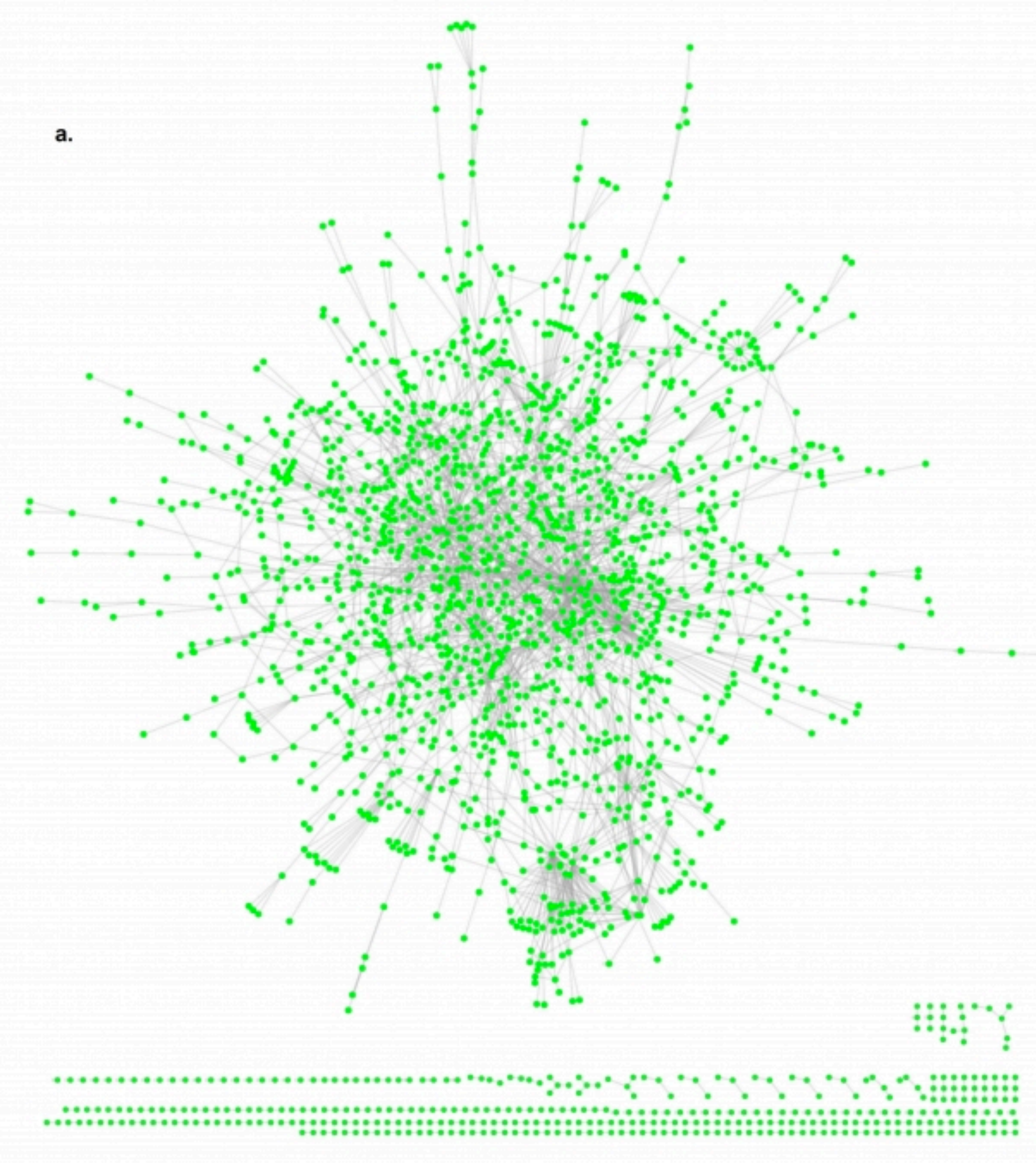
Degree distribution



Degree Distribution of a real Network

Degrees

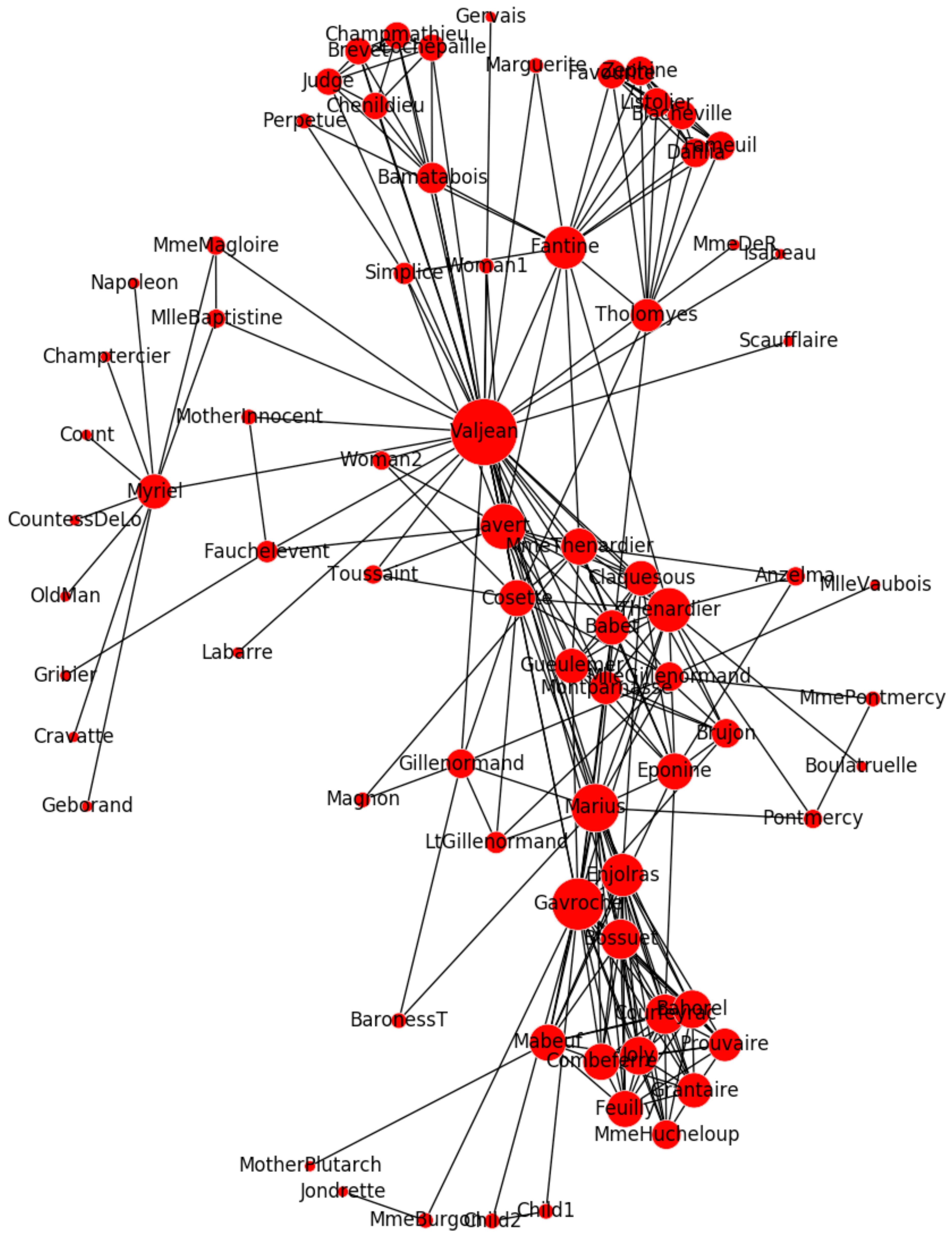




Protein Interaction Network

Degrees

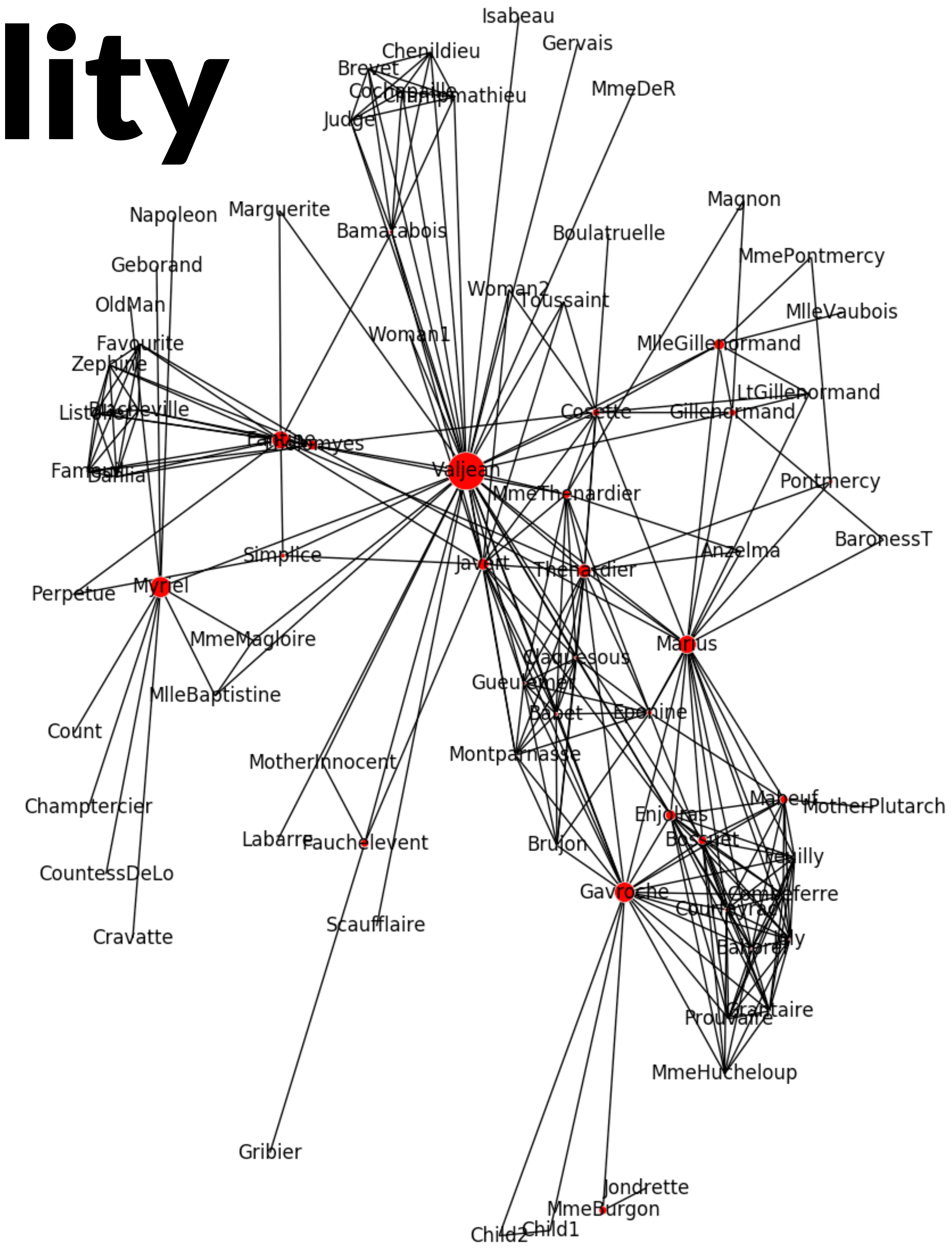
Degree is a measure of local importance



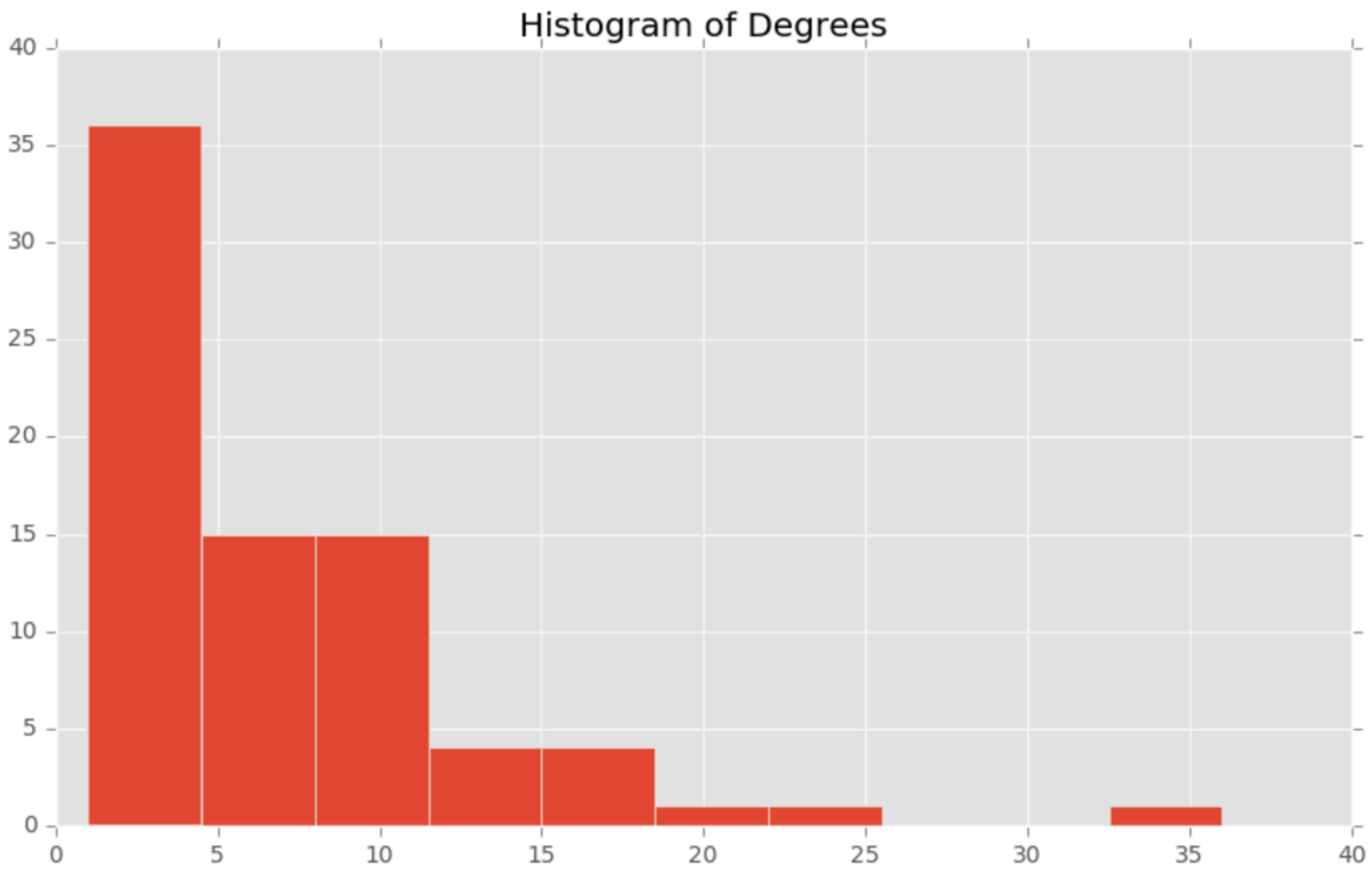
Betweenness Centrality

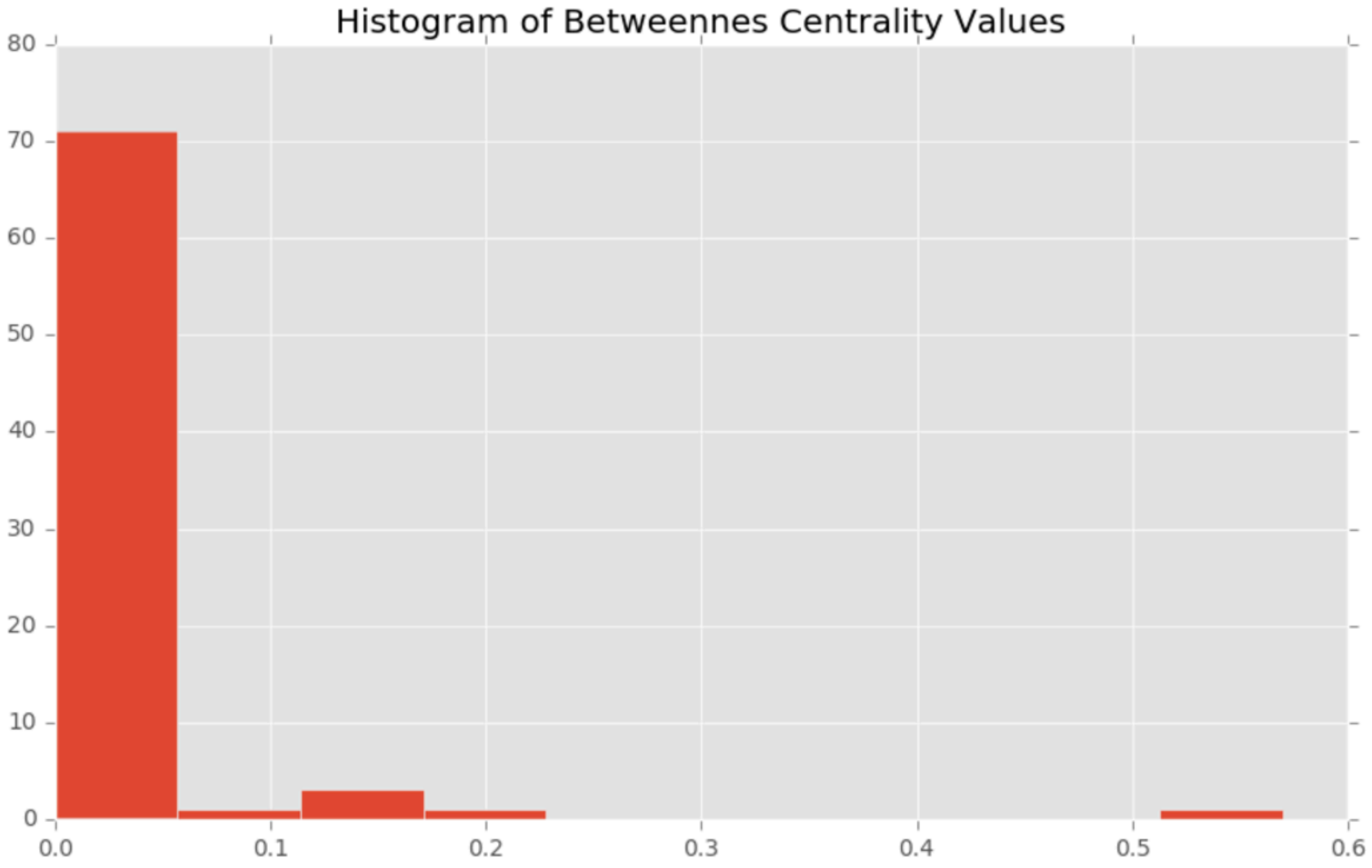
a measure of how many shortest paths pass through a node

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

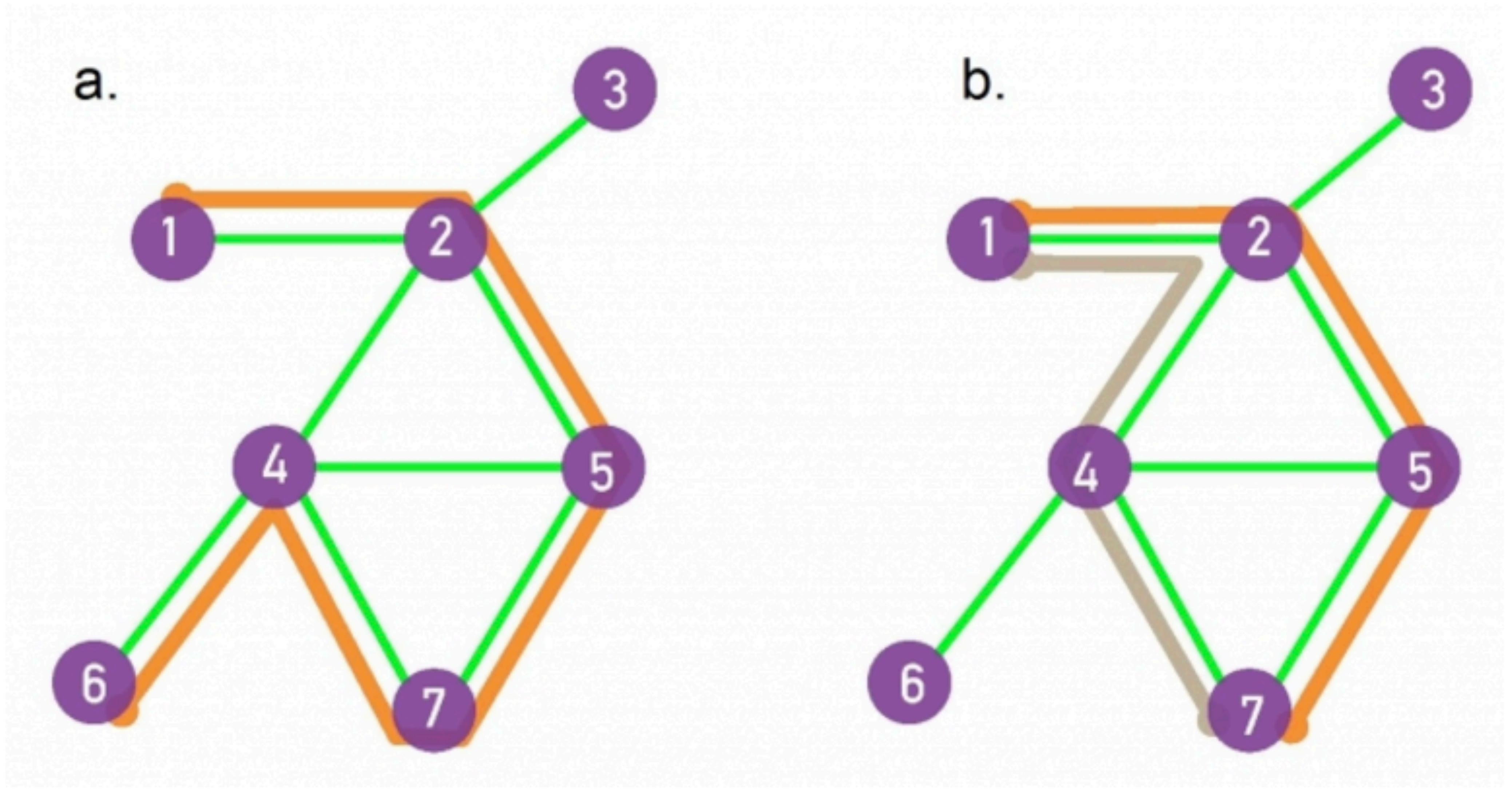
Paths & Distances

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

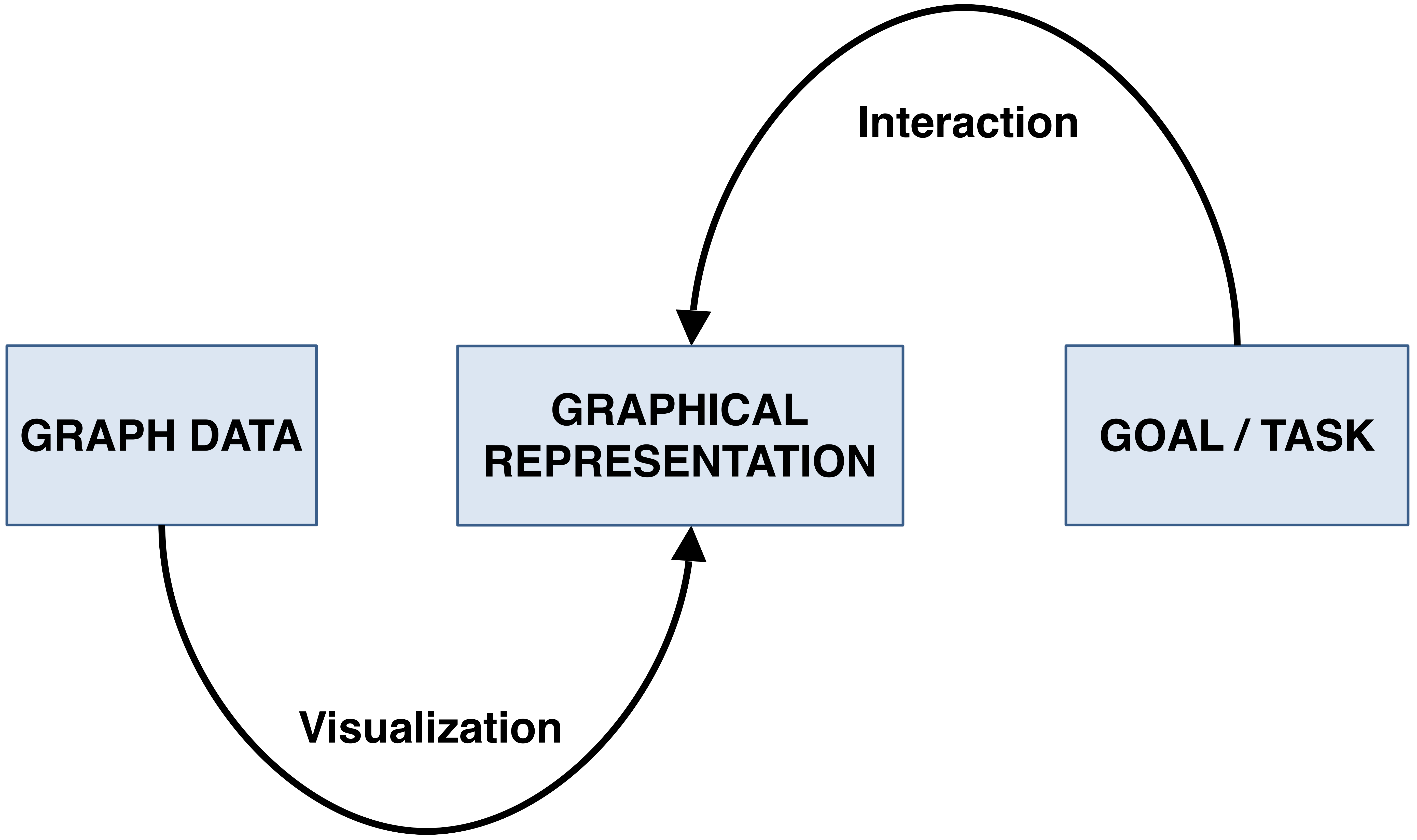
Diameter of graph G
The longest shortest path within G.



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 degree 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

Task Taxonomy for Graph Visualization

Bongshin Lee, Catherine Plaisant,
Cynthia Sims Parr
Human-Computer Interaction Lab
University of Maryland,
College Park, MD 20742, USA
+1-301-405-7445
{bongshin, plaisant, csparr}@cs.umd.edu

Jean-Daniel Fekete,
Nathalie Henry
INRIA Futurs/LRI Bat. 490
Université Paris-Sud,
91405 ORSAY, France
+33-1-69153460
Jean-Daniel.Fekete@inria.fr, nhenry@lri.fr

ABSTRACT

Our goal is to define a list of tasks for graph visualization that has enough detail and specificity to be useful to: 1) designers who want to improve their system and 2) to evaluators who want to compare graph visualization systems. In this paper, we suggest a list of tasks we believe are commonly encountered while analyzing graph data. We define graph specific objects and demonstrate how all complex tasks could be seen as a series of low-level tasks performed on those objects. We believe that our taxonomy, associated with benchmark datasets and specific tasks, would help evaluators generalize results collected through a series of controlled experiments.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Graphical user interfaces (GUI)*

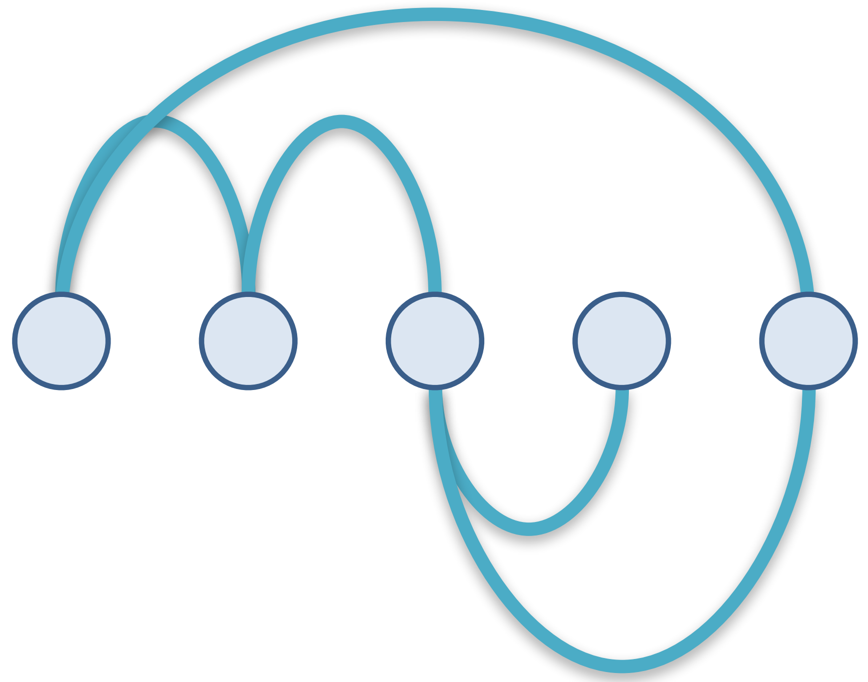
user studies of graph visualization techniques and extracted the tasks used in those studies.

After making those two lists, we considered the set of low-level Visual Analytics tasks proposed by Amar *et al*. [2]. These tasks were extracted from a corpus of questions about tabular data. We realized that our tasks all seem to be compound tasks made up of Amar *et al*'s primitive tasks applied to the graph objects. When some tasks could not be represented with those tasks and objects, we added either an object or a low-level task. In this paper, we demonstrate how all complex tasks could be seen as a series of low-level tasks performed on those objects.

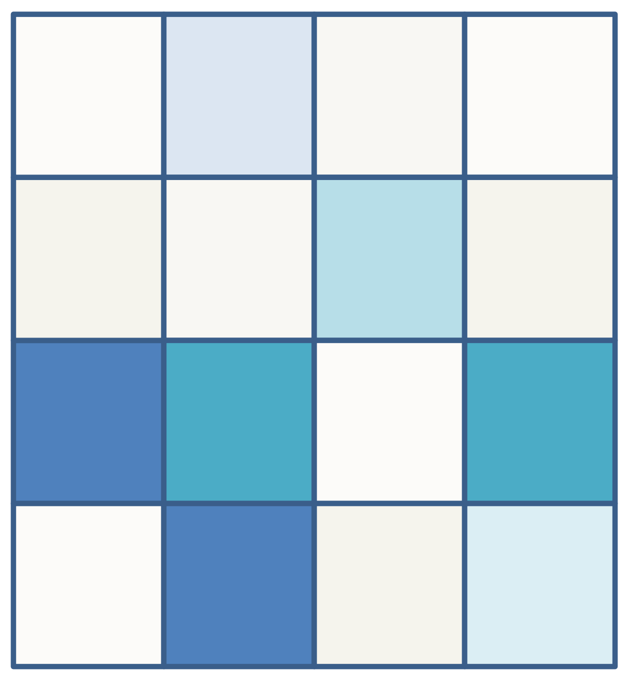
2. GRAPH-SPECIFIC OBJECTS

A graph consists of two types of primitive elements, nodes and links. A subgraph of a graph G is a graph whose nodes and links are subsets of G. There are several meaningful subgraphs such as

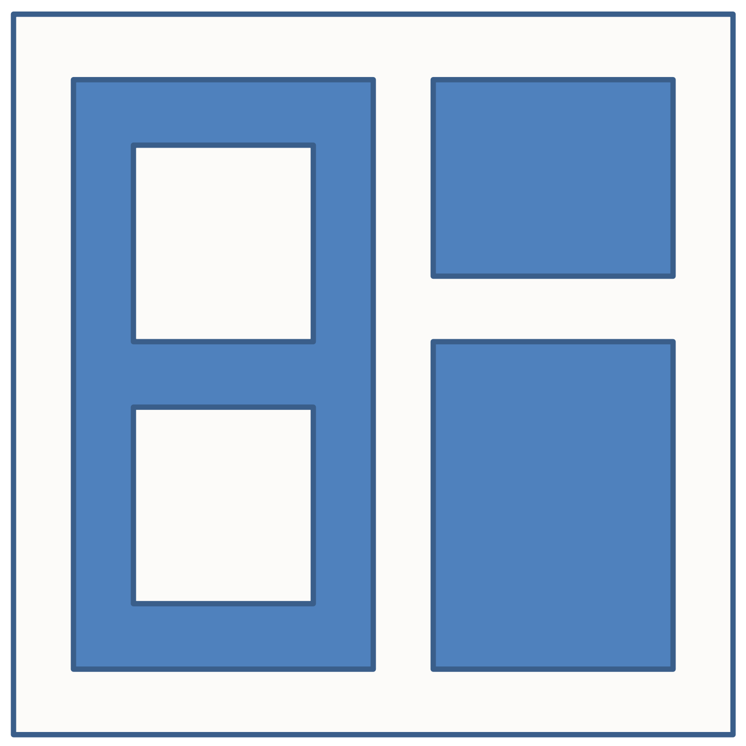
Three Types of Graph Representations



Explicit
(Node-Link)



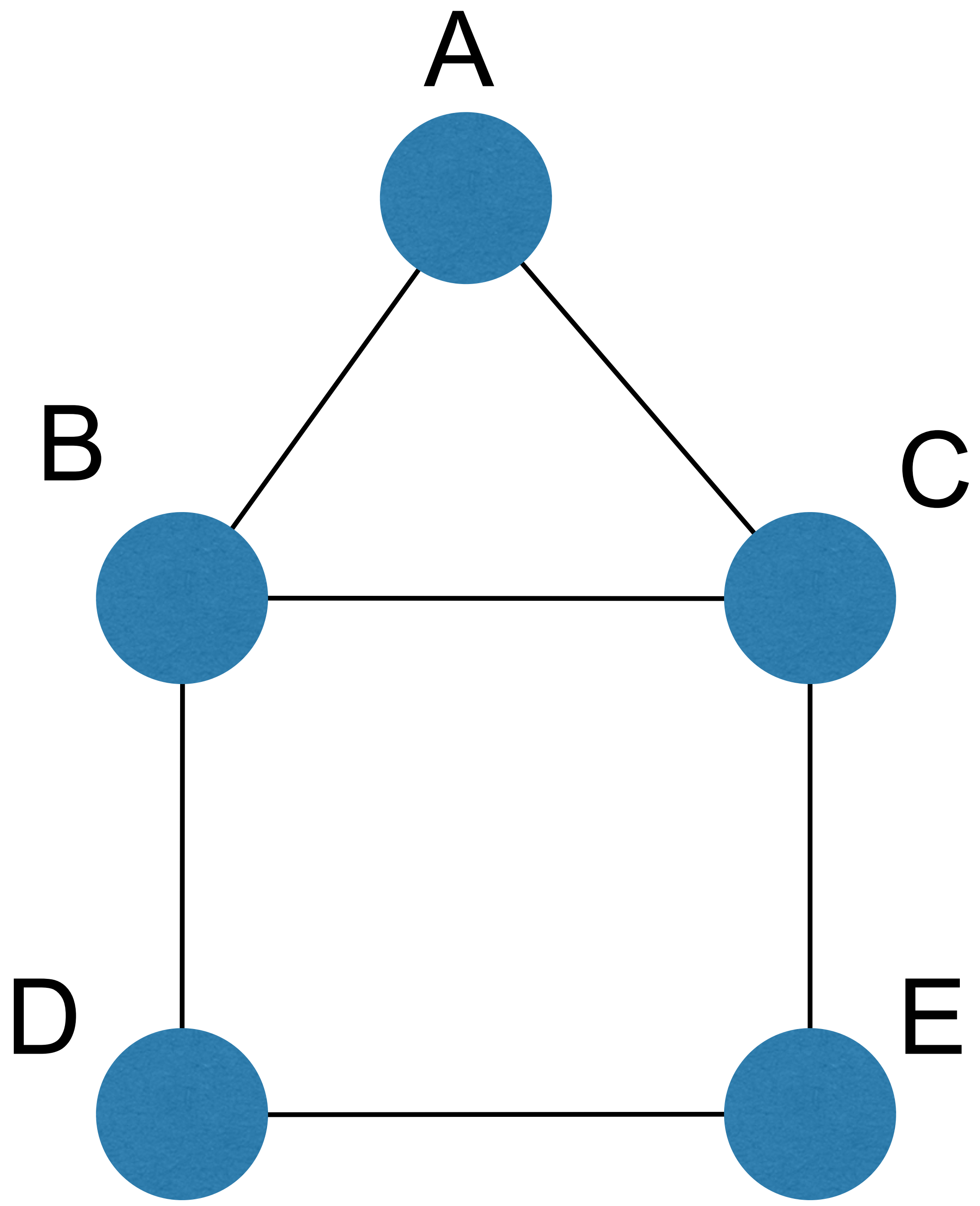
Matrix

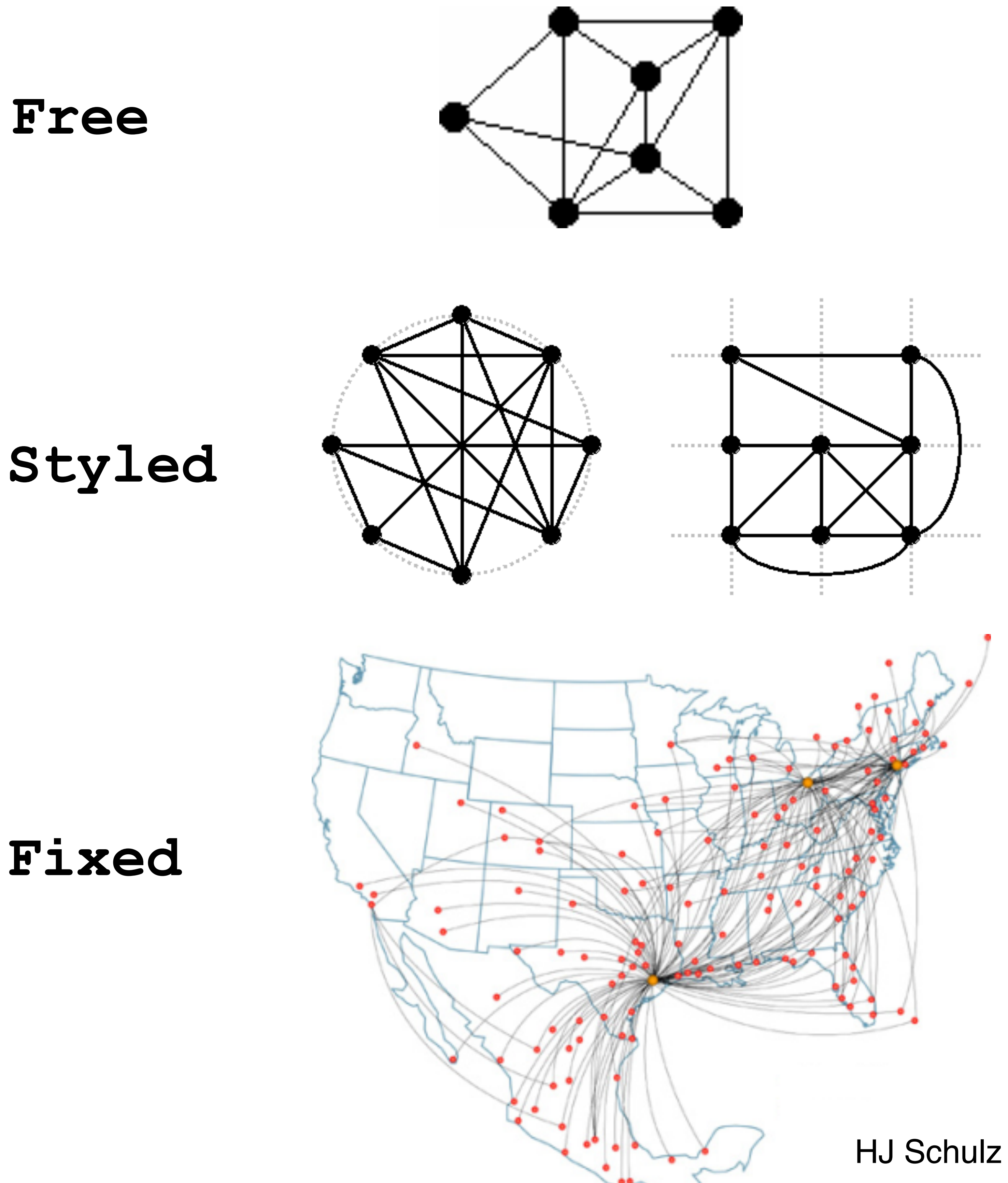


Implicit

Explicit Graph Representations

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





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

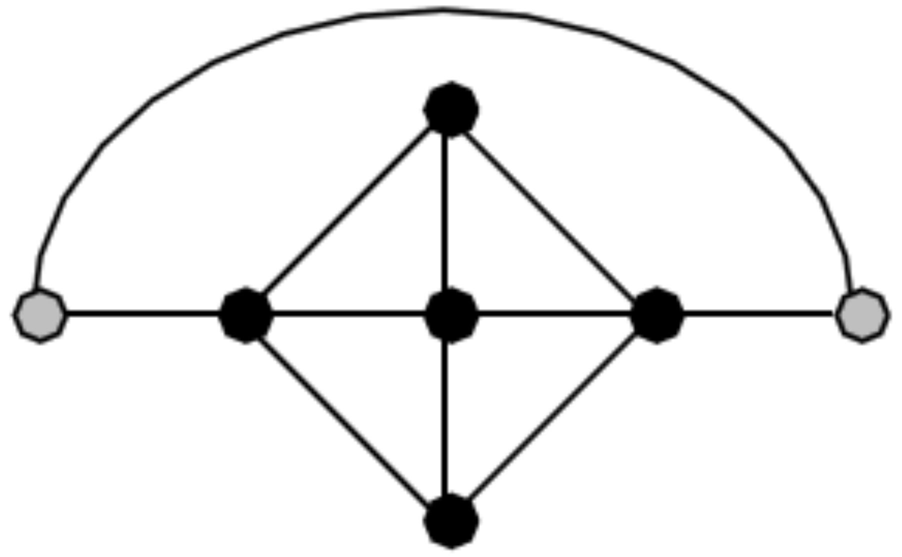
Aspect ratio about 1 (not too long and not too wide)

Symmetry: similar graph structures should look similar

list adapted from Battista et al. 1999

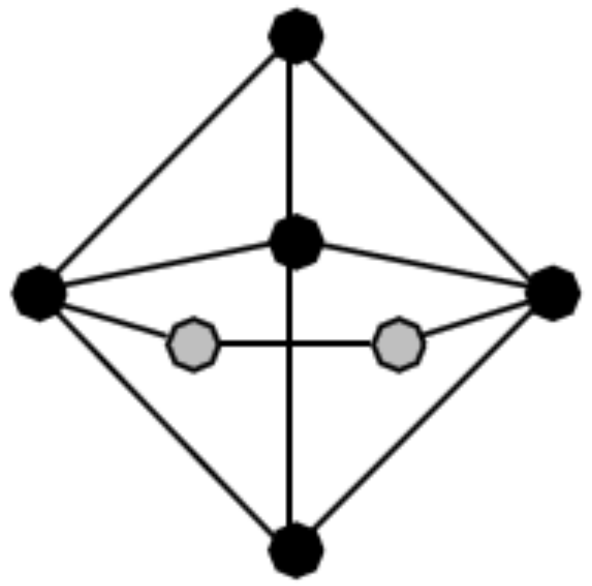
Conflicting Criteria

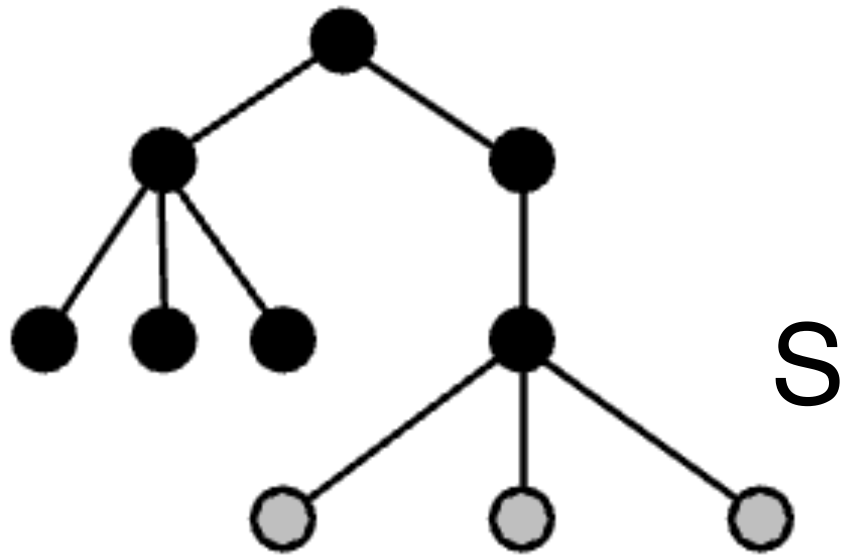
Minimum number
of edge crossings



vs.

Uniform edge
length

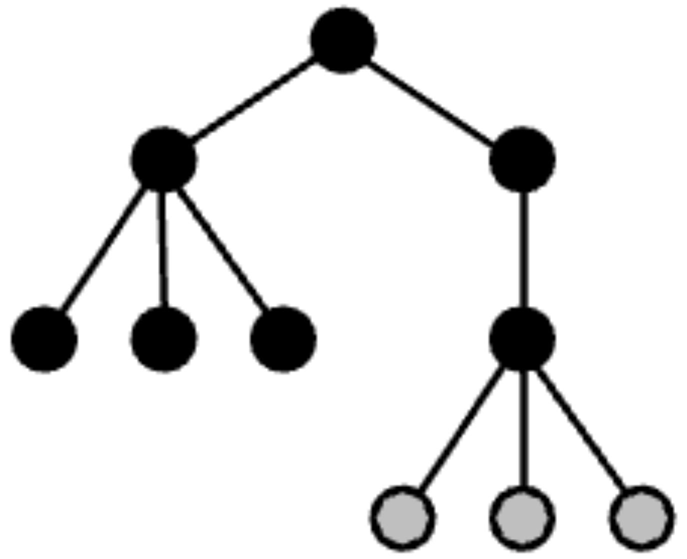




Space utilization

vs.

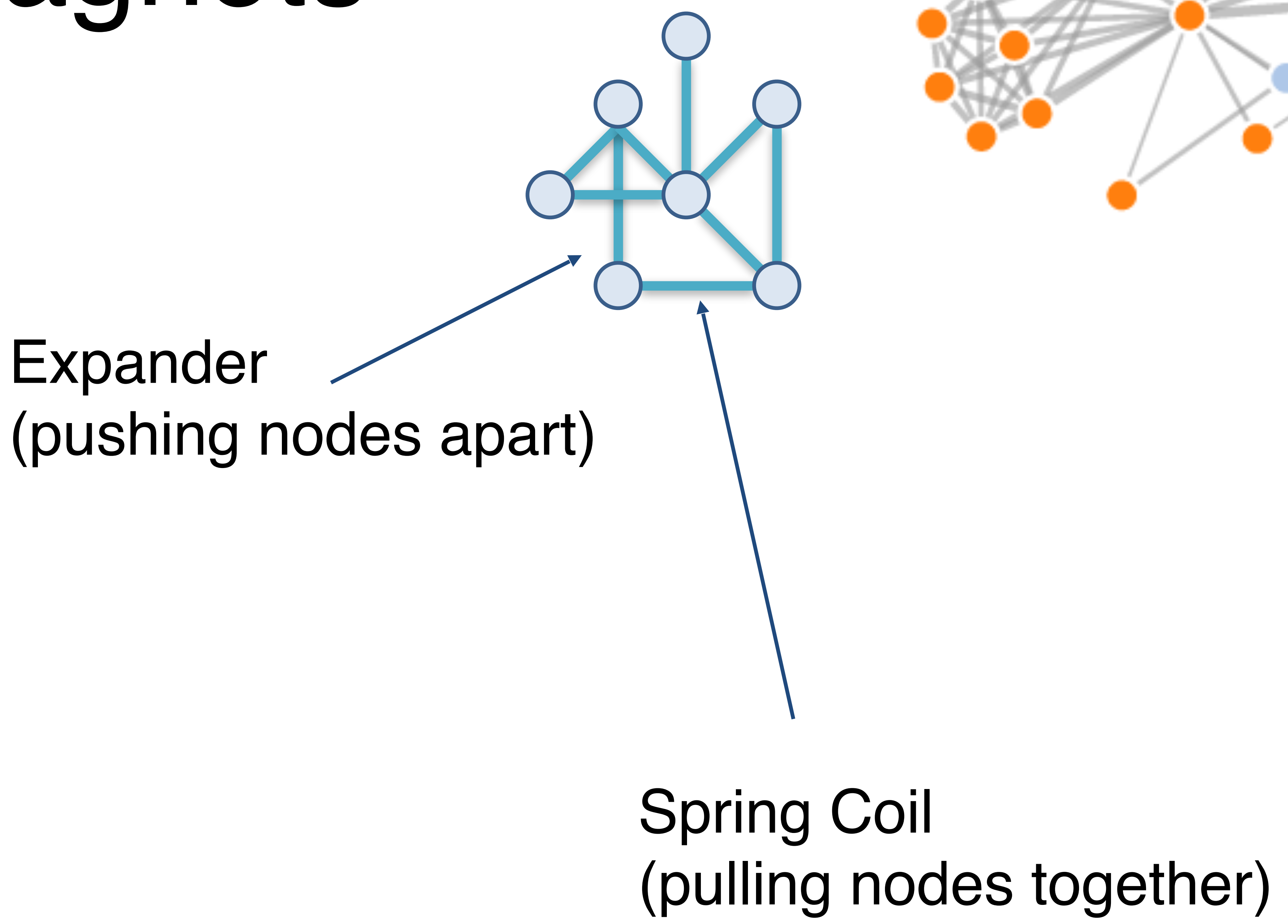
Symmetry



Schulz 2004

Force Directed Layouts

Physics model:
edges = springs,
vertices = repulsive magnets



Algorithm

Place Vertices in random locations

While not equilibrium

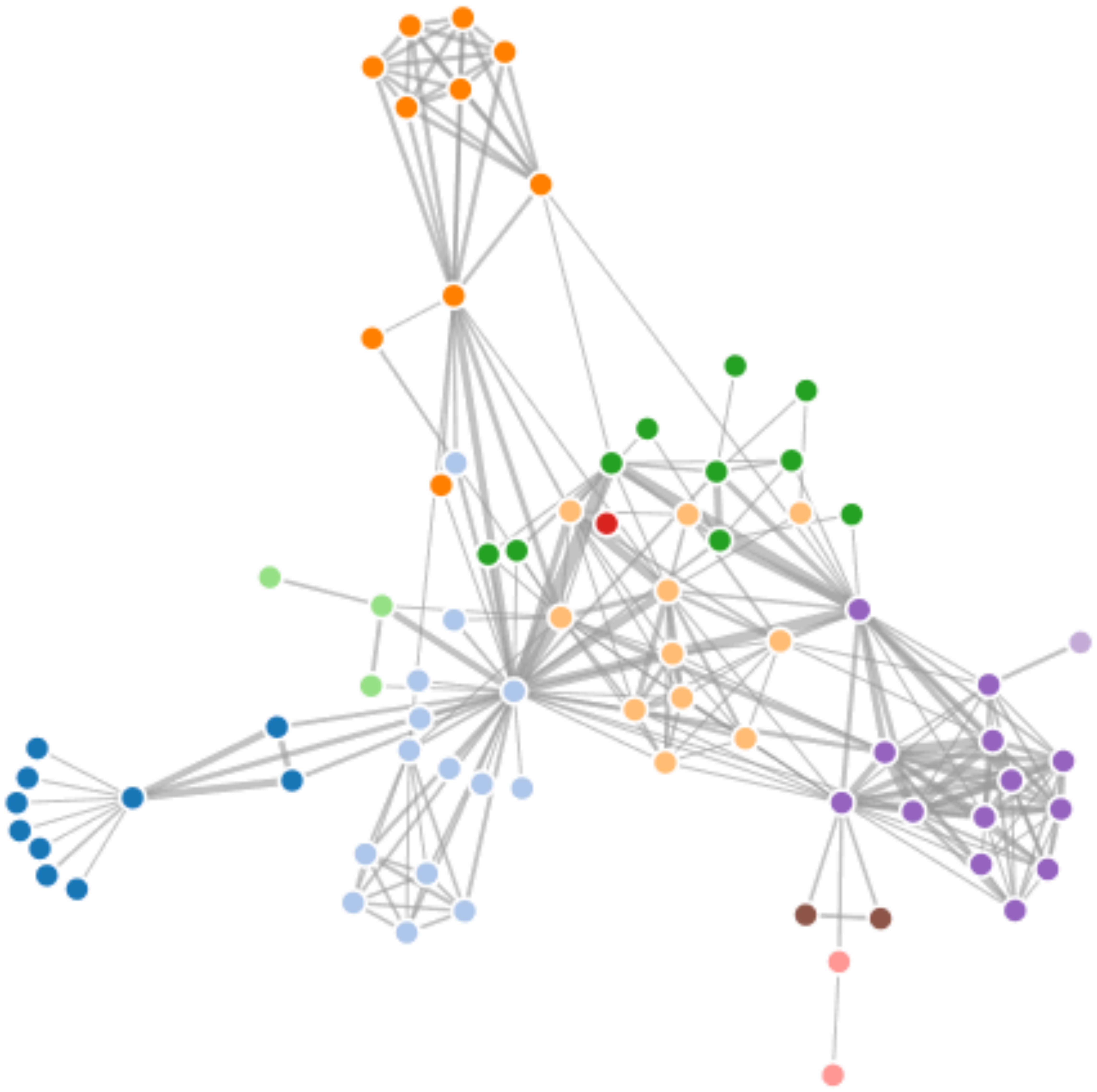
  calculate force on vertex

    sum of

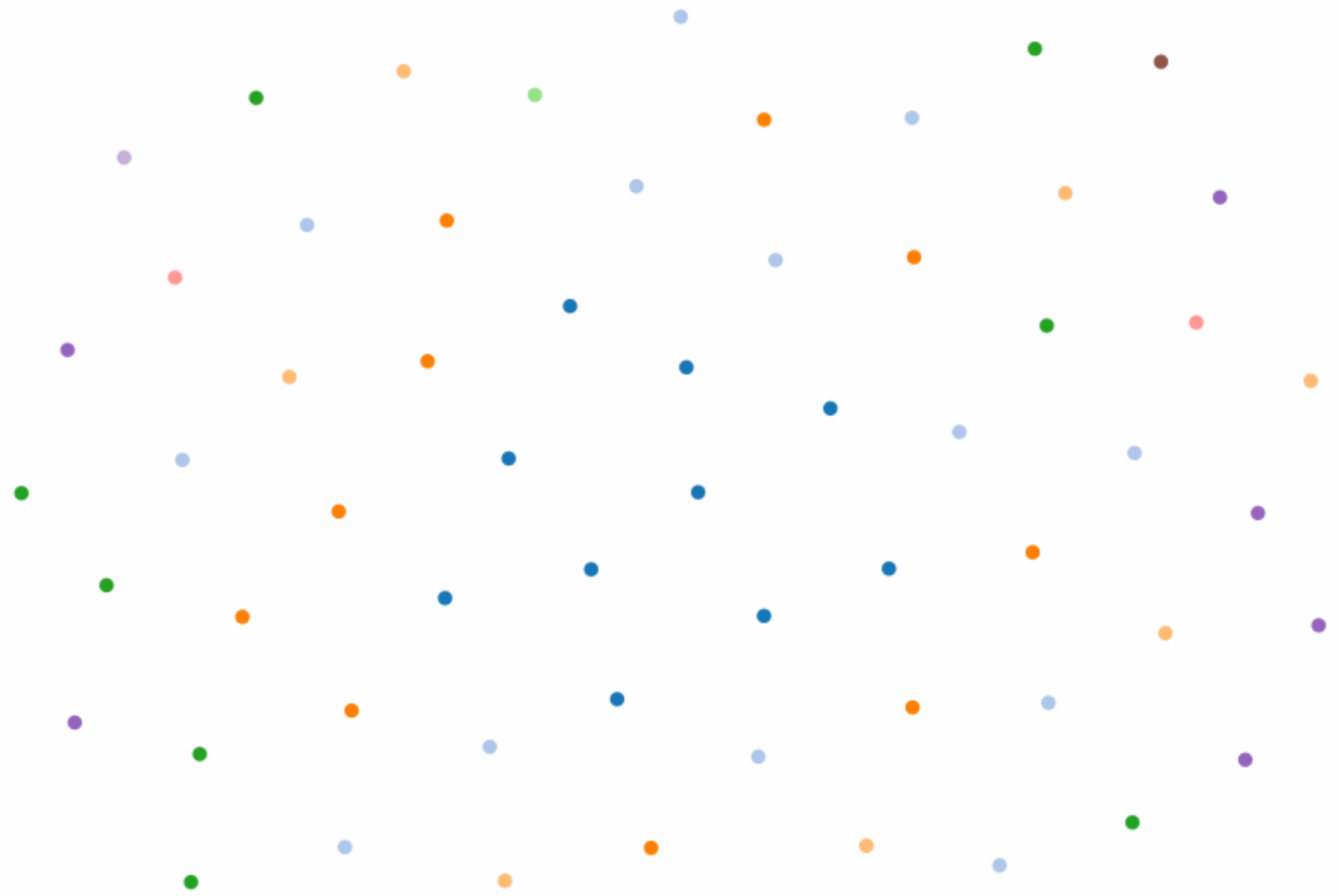
      pairwise repulsion of all nodes

      attraction between connected nodes

  move vertex by c * force on vertex



What happens when there are no links?



Properties

Generally good layout

Uniform edge length

Clusters commonly visible

Not deterministic

Computationally expensive: $O(n^3)$

n^2 in every step, it takes about n cycles to reach equilibrium

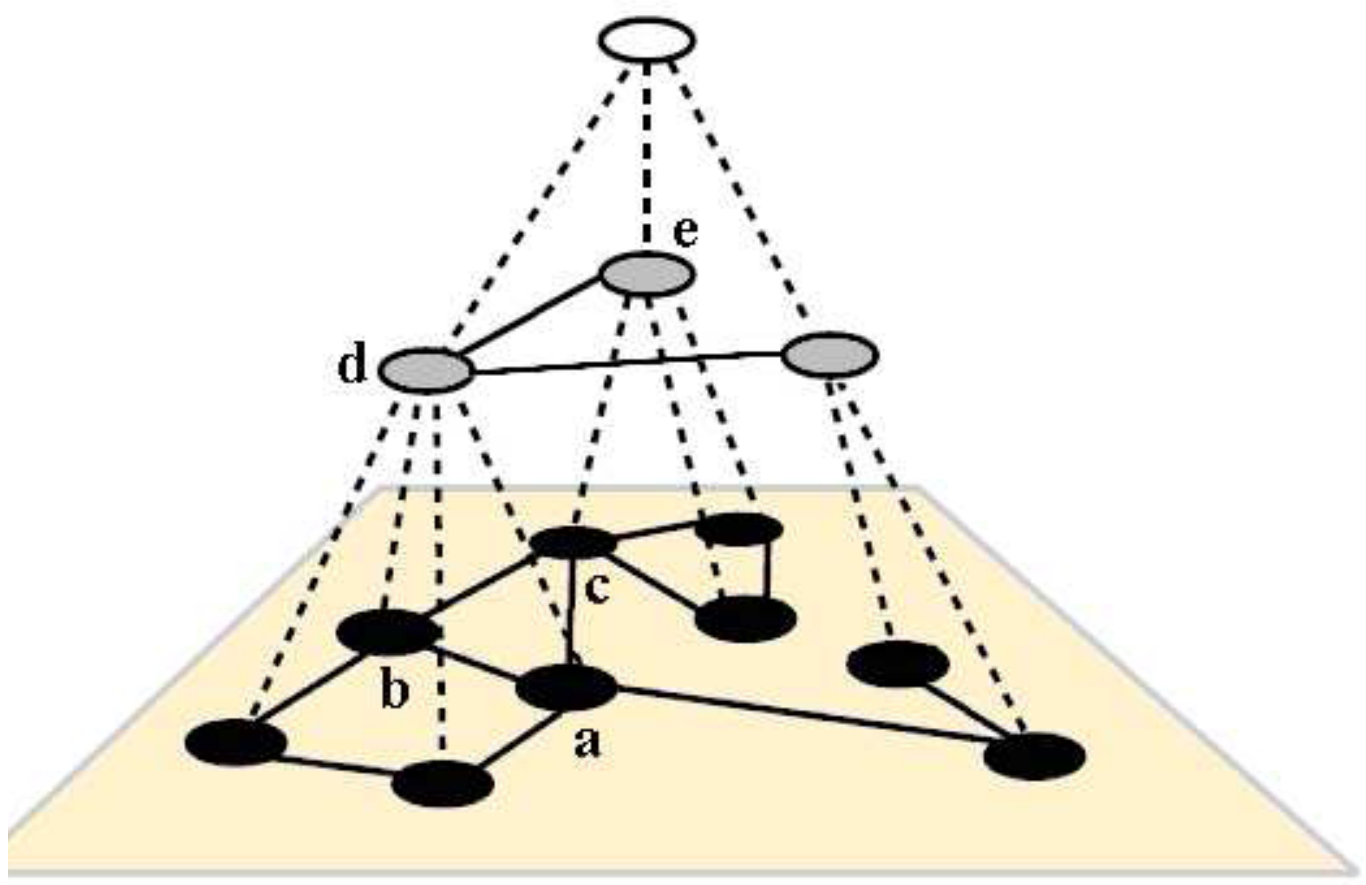
Limit (interactive): ~1000 nodes

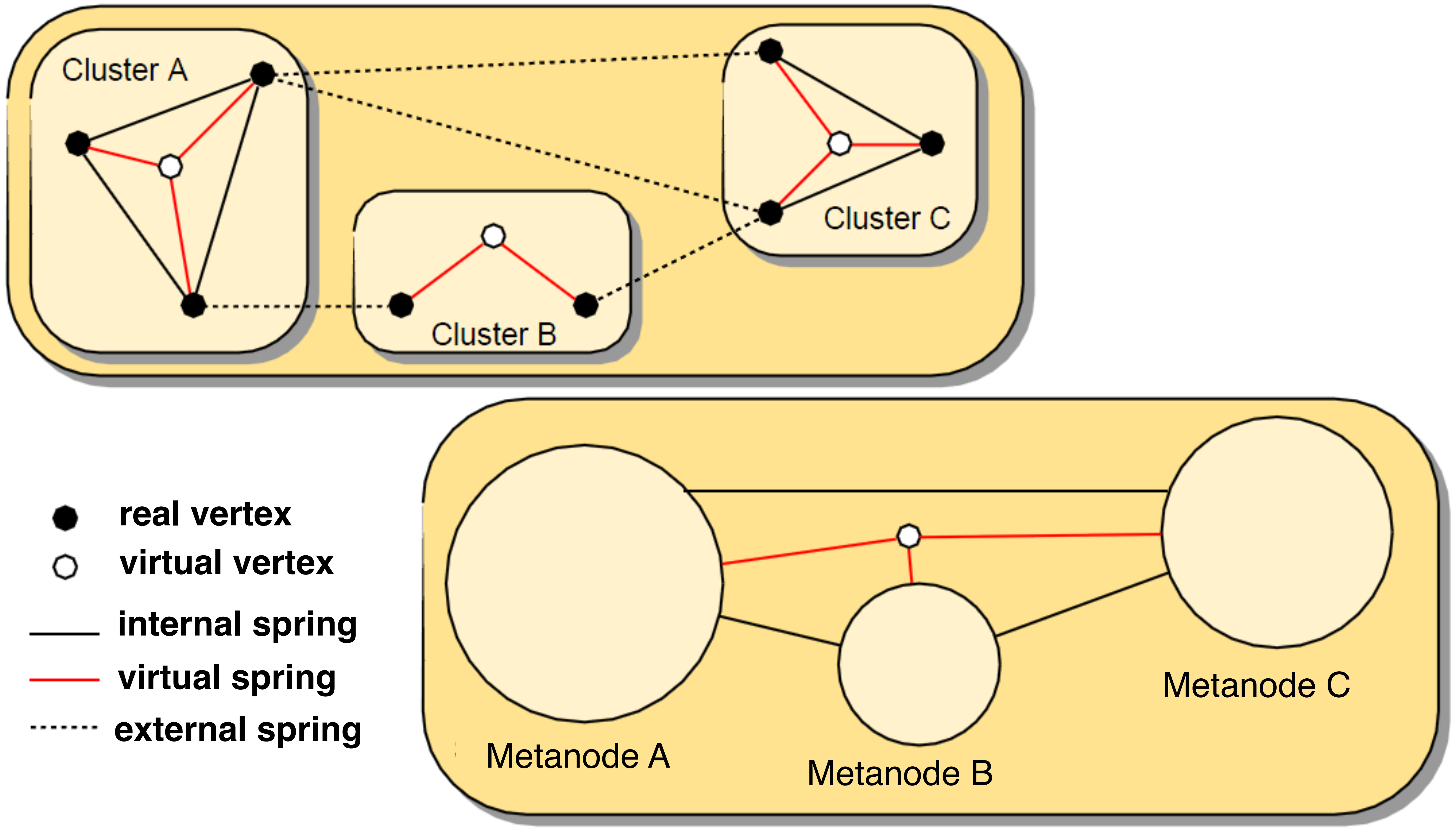
in practice: damping, center of gravity

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

Giant Hairball

Adress Computational Scalability: Multilevel Approaches





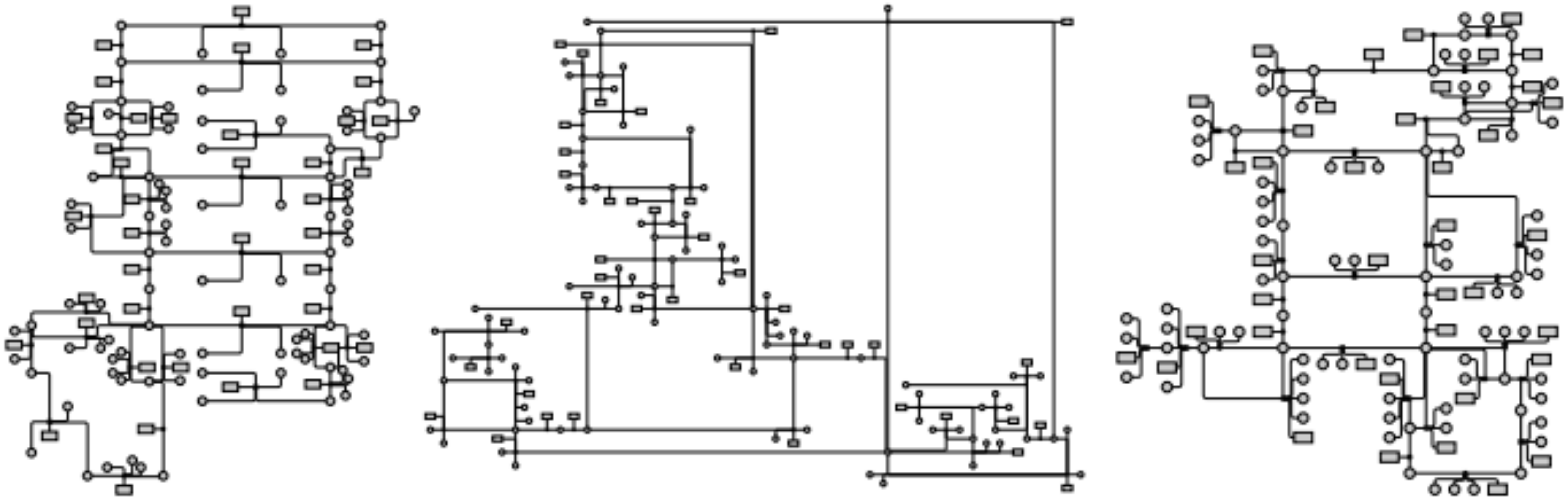
[Schulz 2004]

HOLA: Human-like Orthogonal Layout

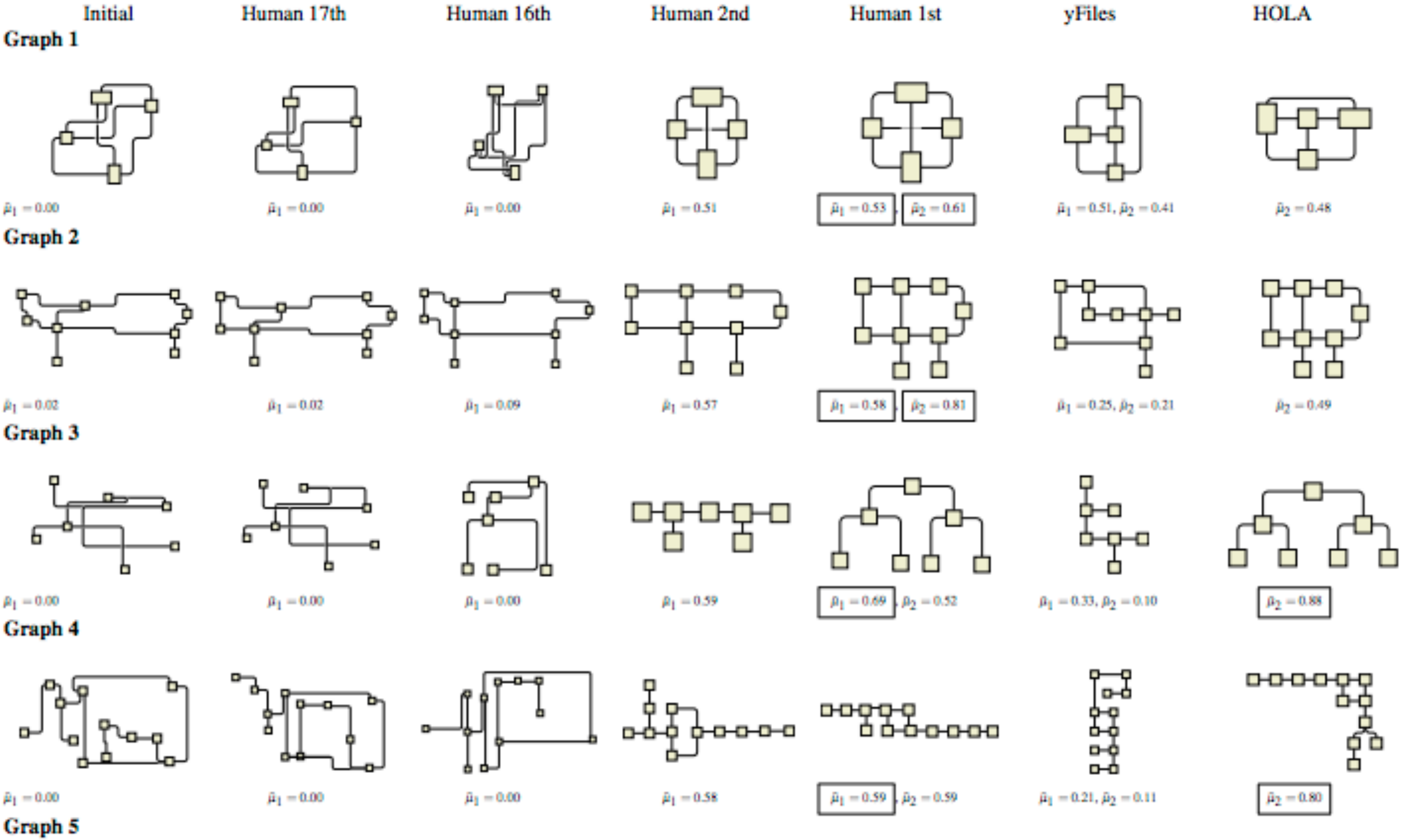
Study how humans lay-out a graph

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

Graphs in 3D

Why, why not visualize graphs in 3D?

Why, why not use AR/VR?

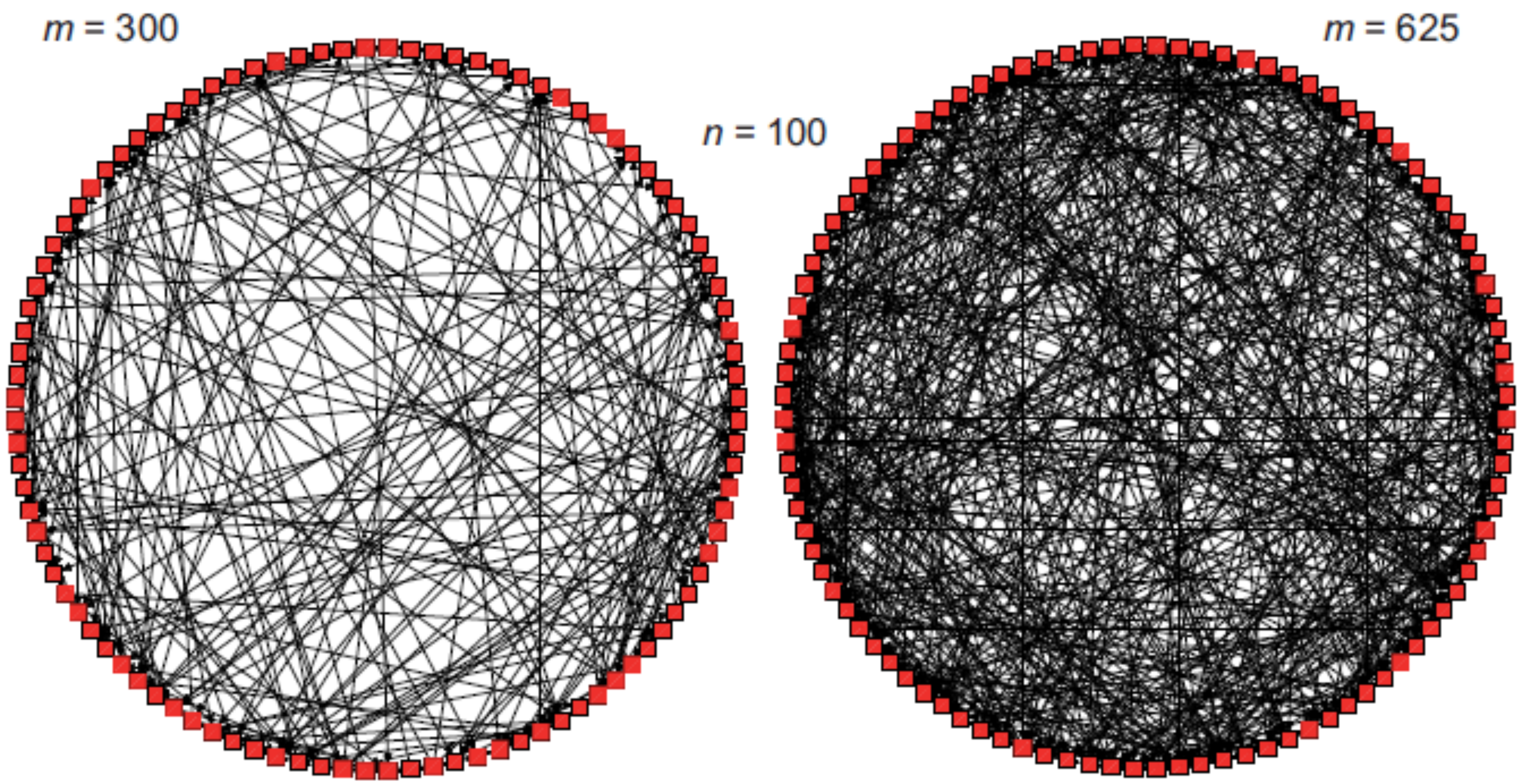
https://twitter.com/alexsigaras/status/860560655031685121

Styled / Restricted Layouts

Circular Layout

Node ordering

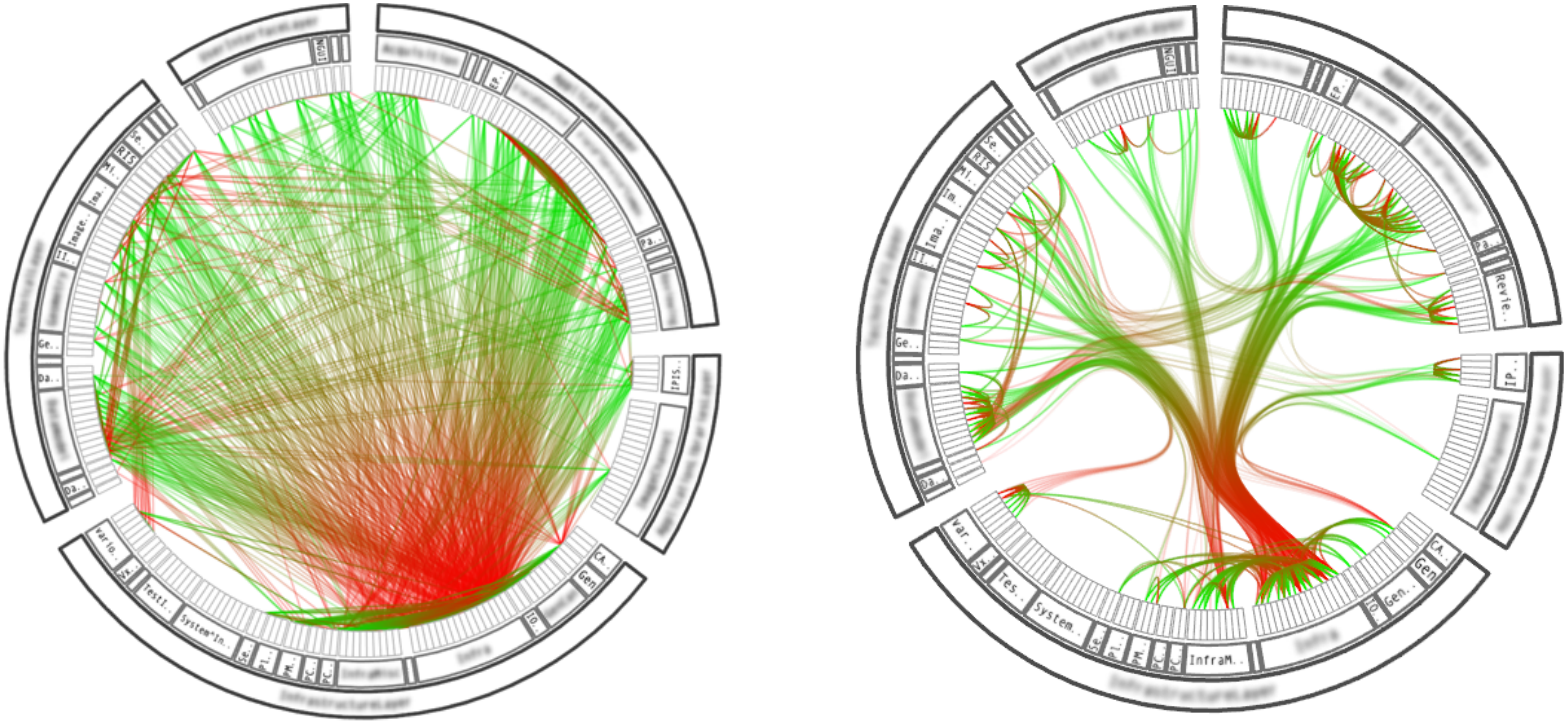
Edge Clutter



ca. 3% of all possible edges

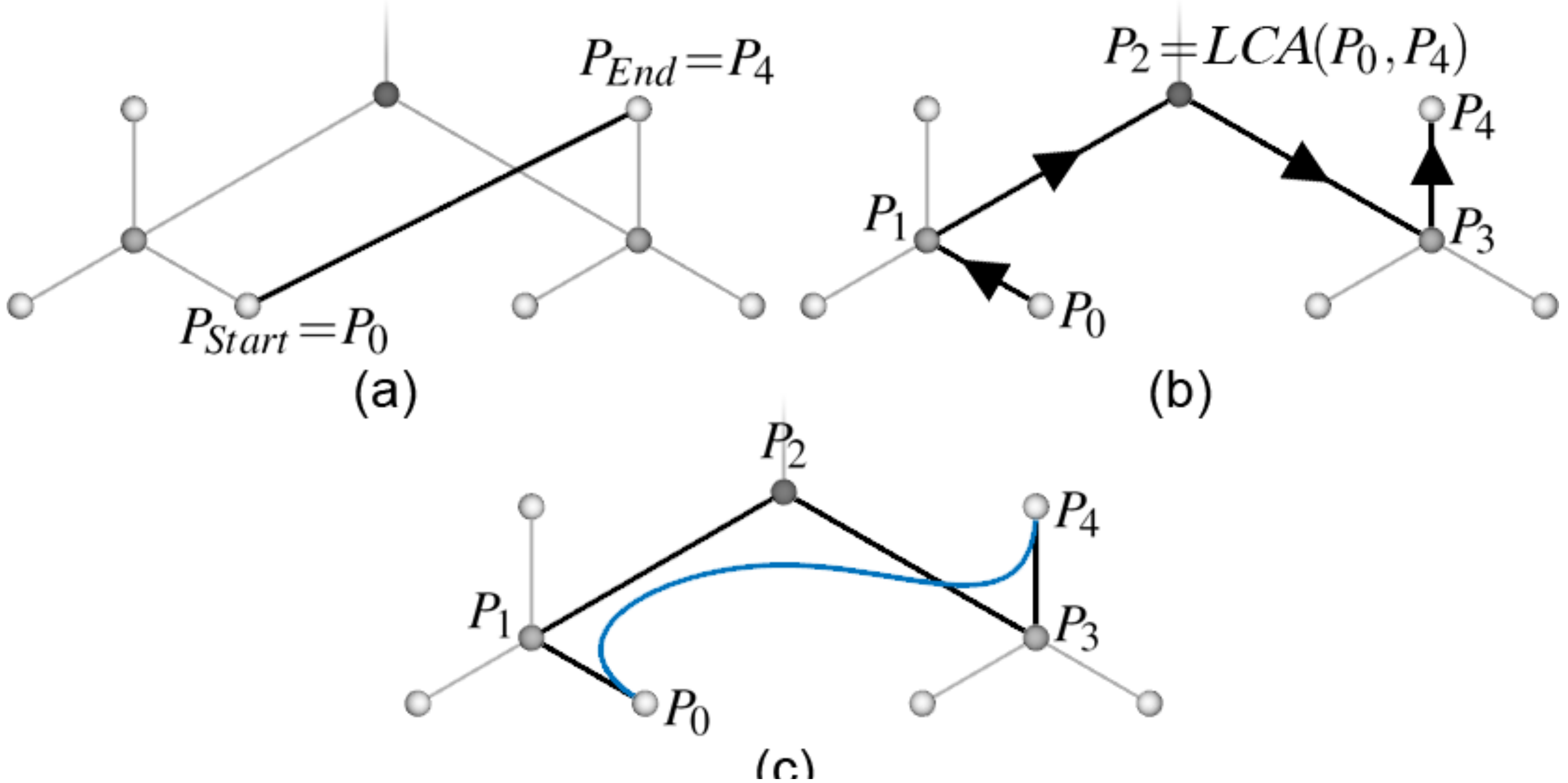
ca. 6,3% of all possible edges

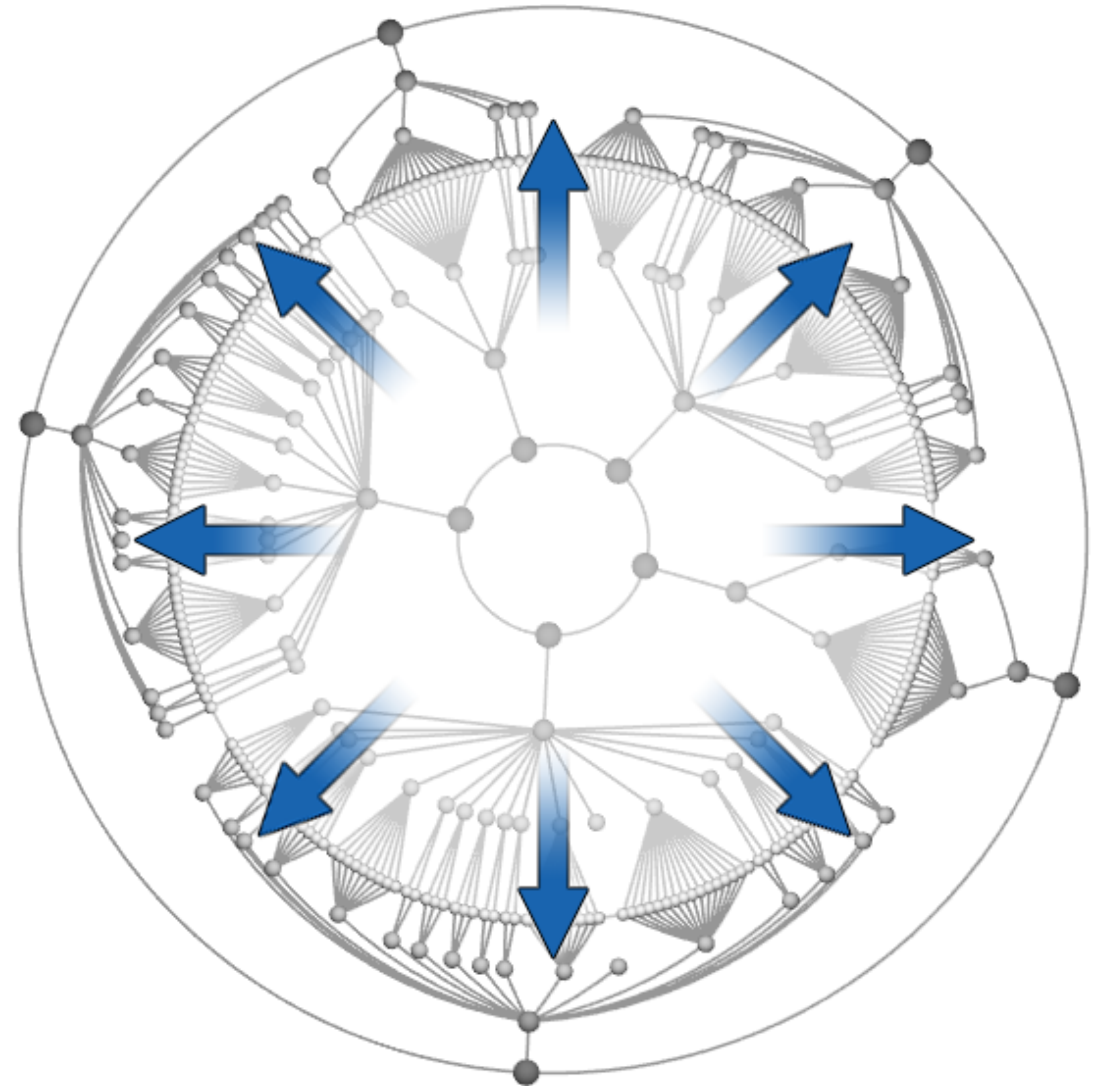
Reduce Clutter: Edge Bundling

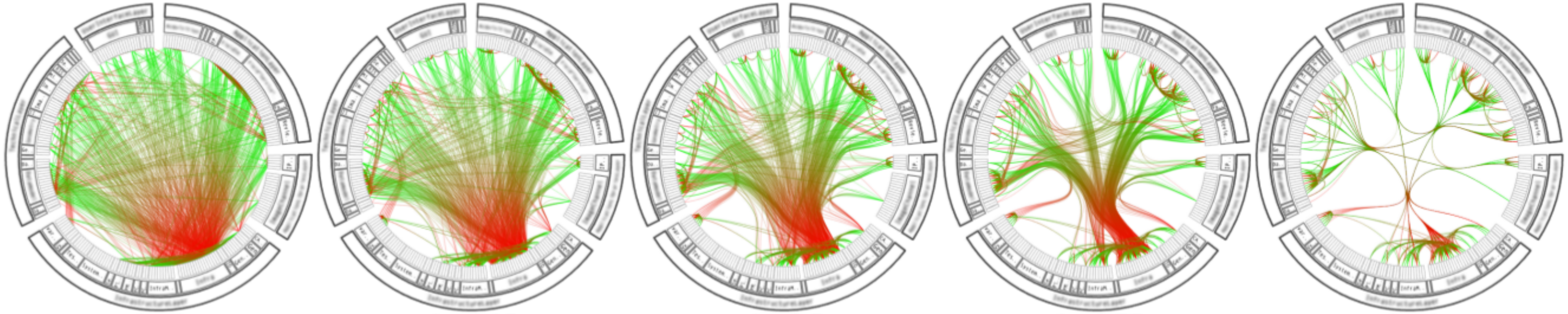


Holten et al. 2006

Hierarchical Edge Bundling





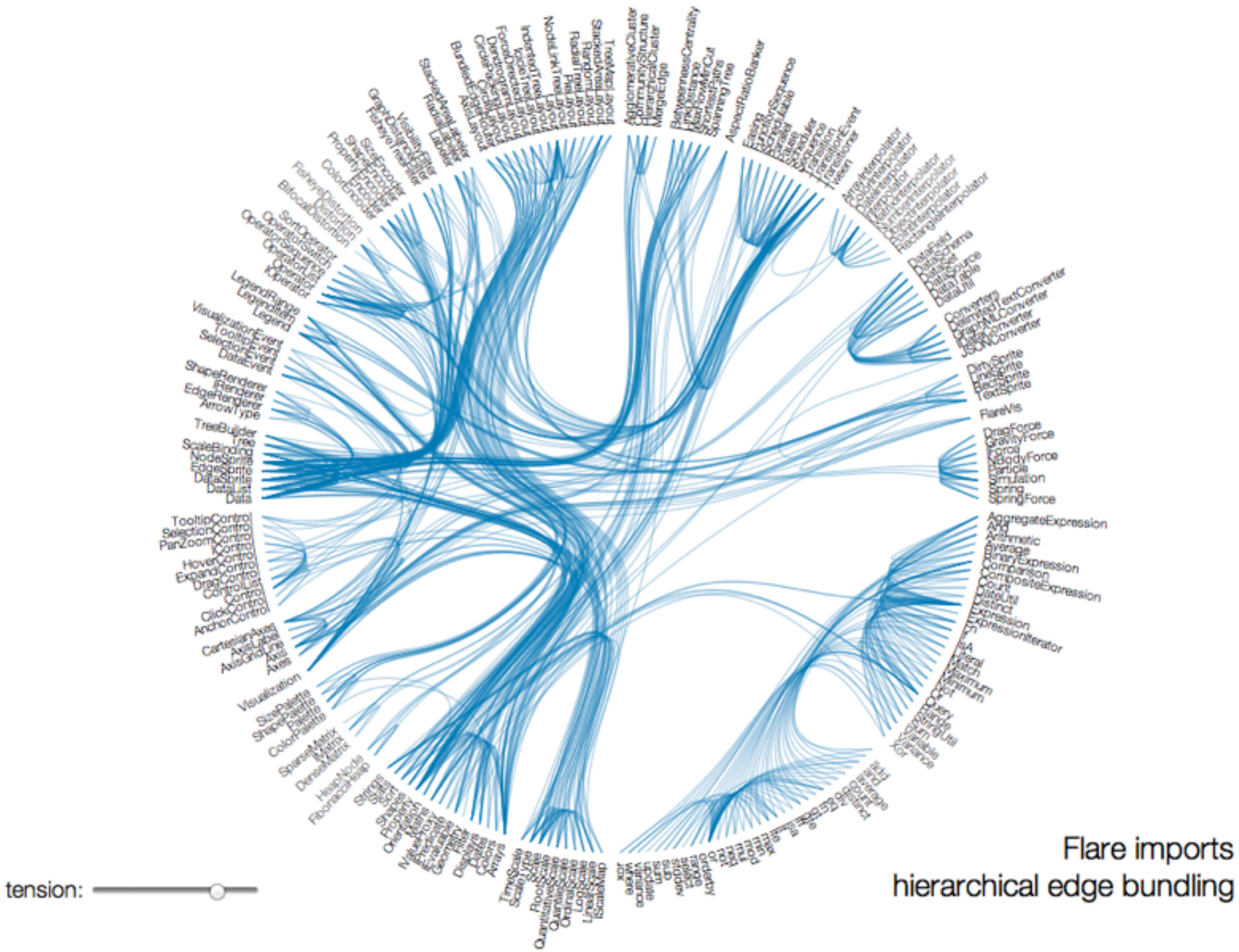


$\beta = 0$ $\beta = 0.25$ $\beta = 0.5$ $\beta = 0.75$ $\beta = 1$

Bundling Strength

Holten et al. 2006

Bundling Strength



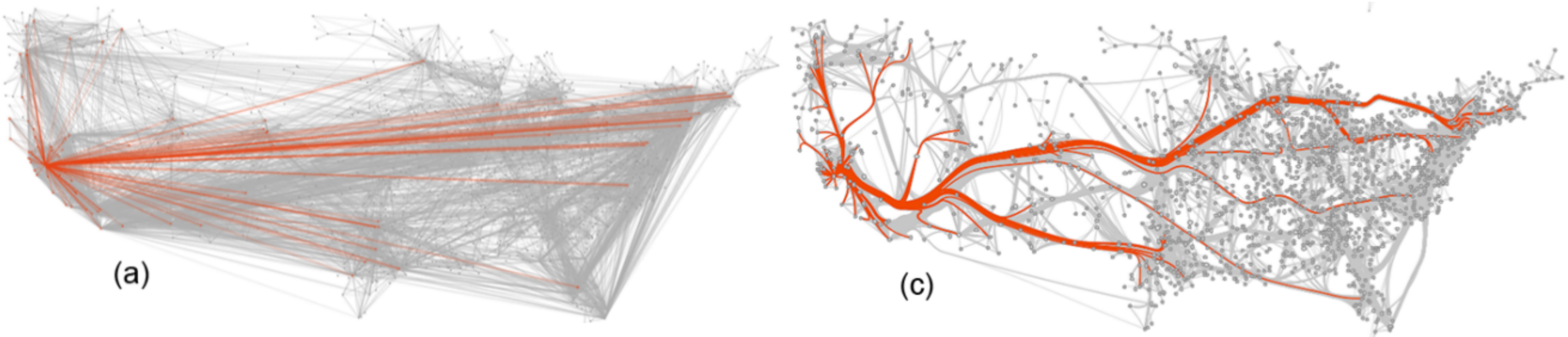
mbostock.github.com/d3/talk/20111116/bundle.html

Michael Bostock

Fixed Layouts

Can’t vary position of nodes

Edge routing important

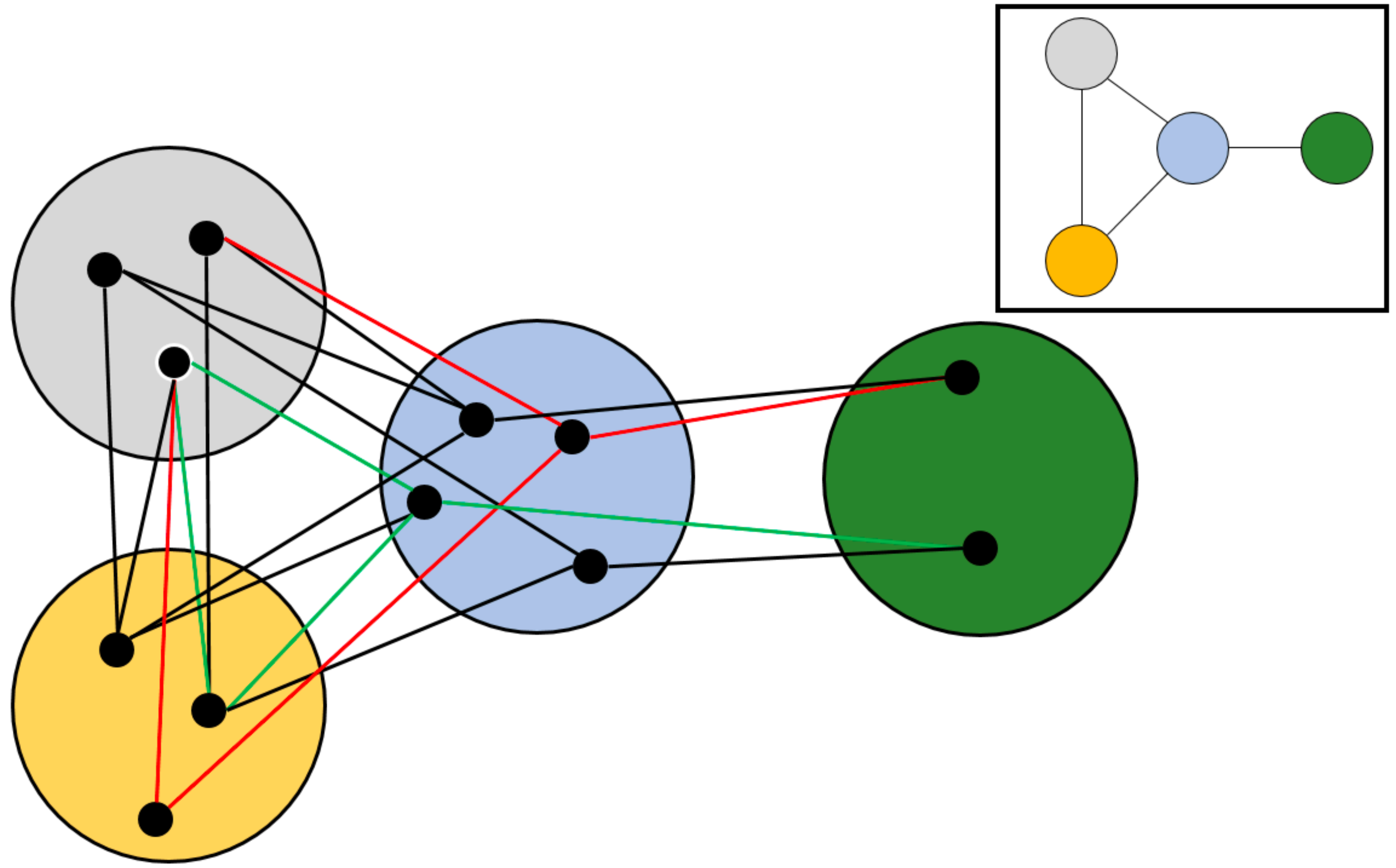


Supernodes / Aggregation

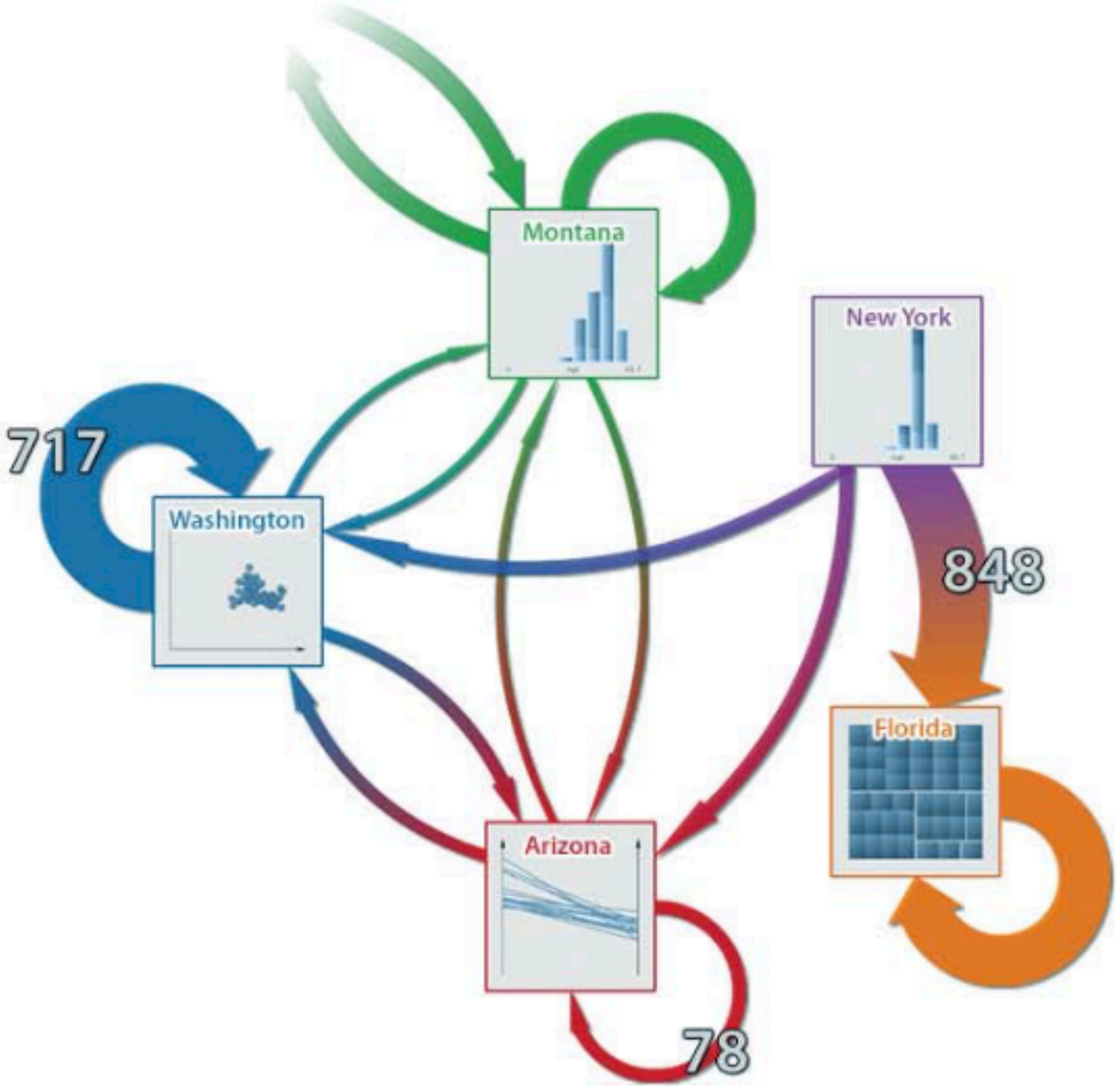
Supernodes: aggregate of nodes

manual or algorithmic

- clustering



Aggregation

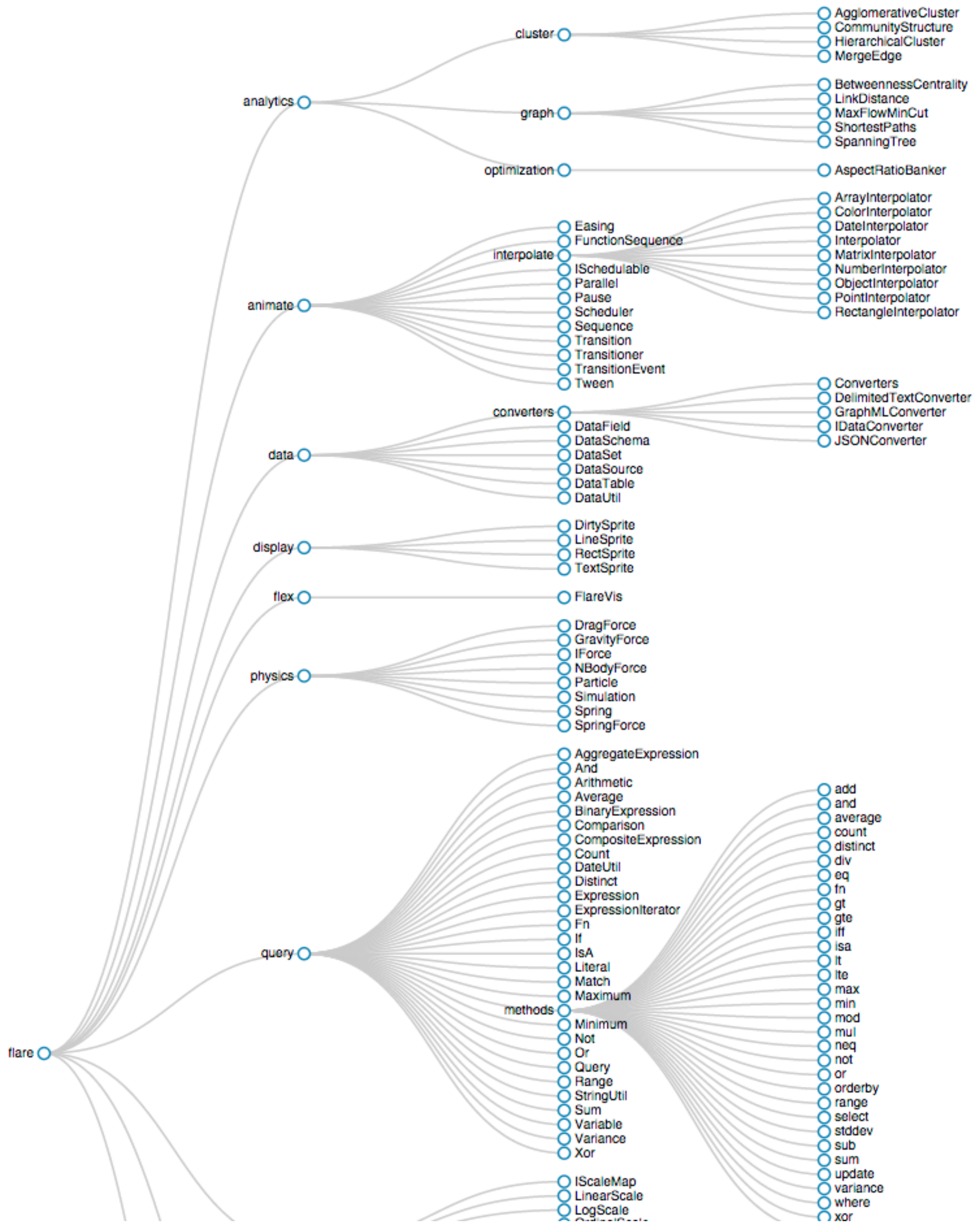


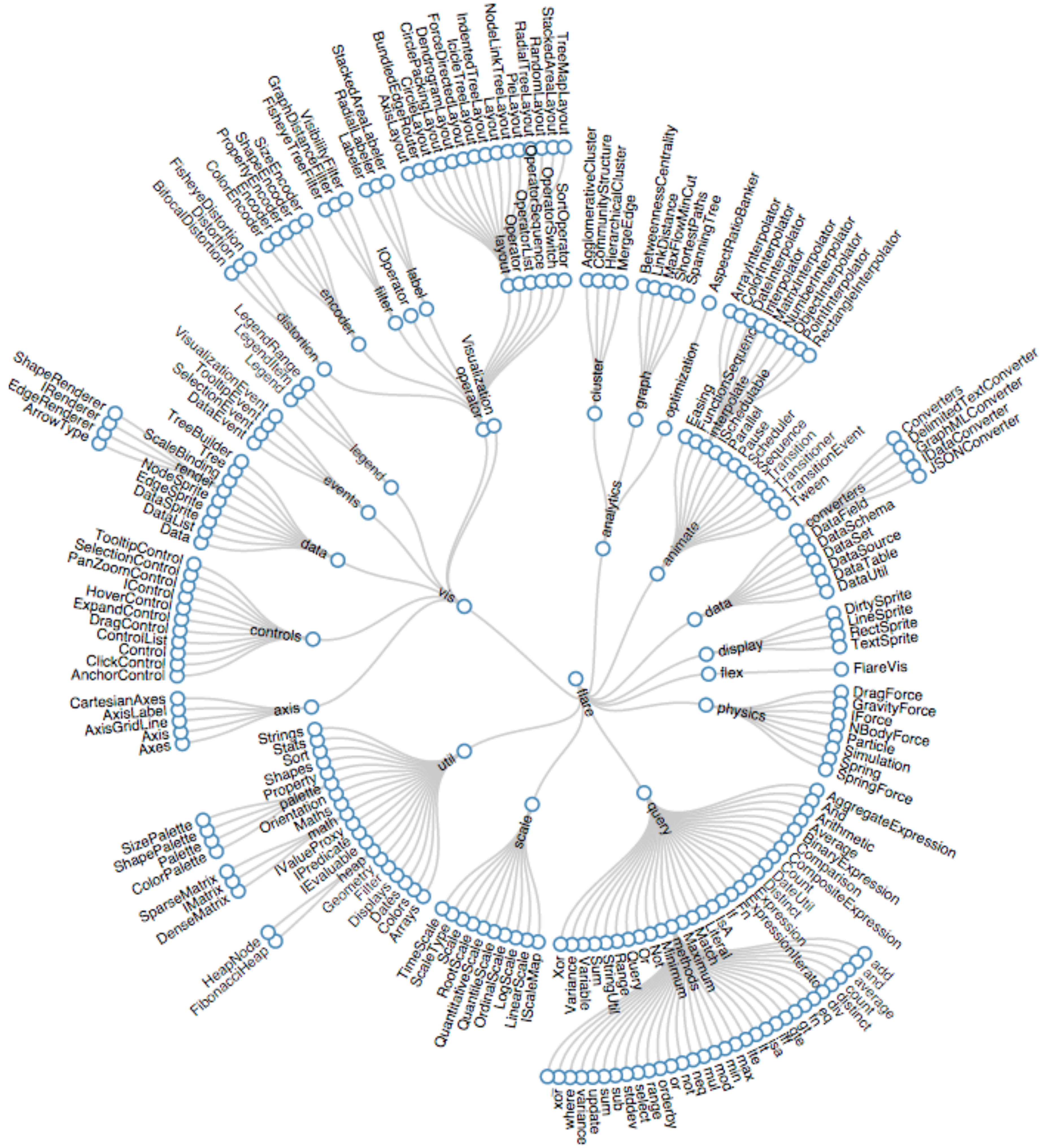
https://youtu.be/E1PVTitj7h0?t=57

Explicit Tree Visualization

Reingold–
Tilford layout

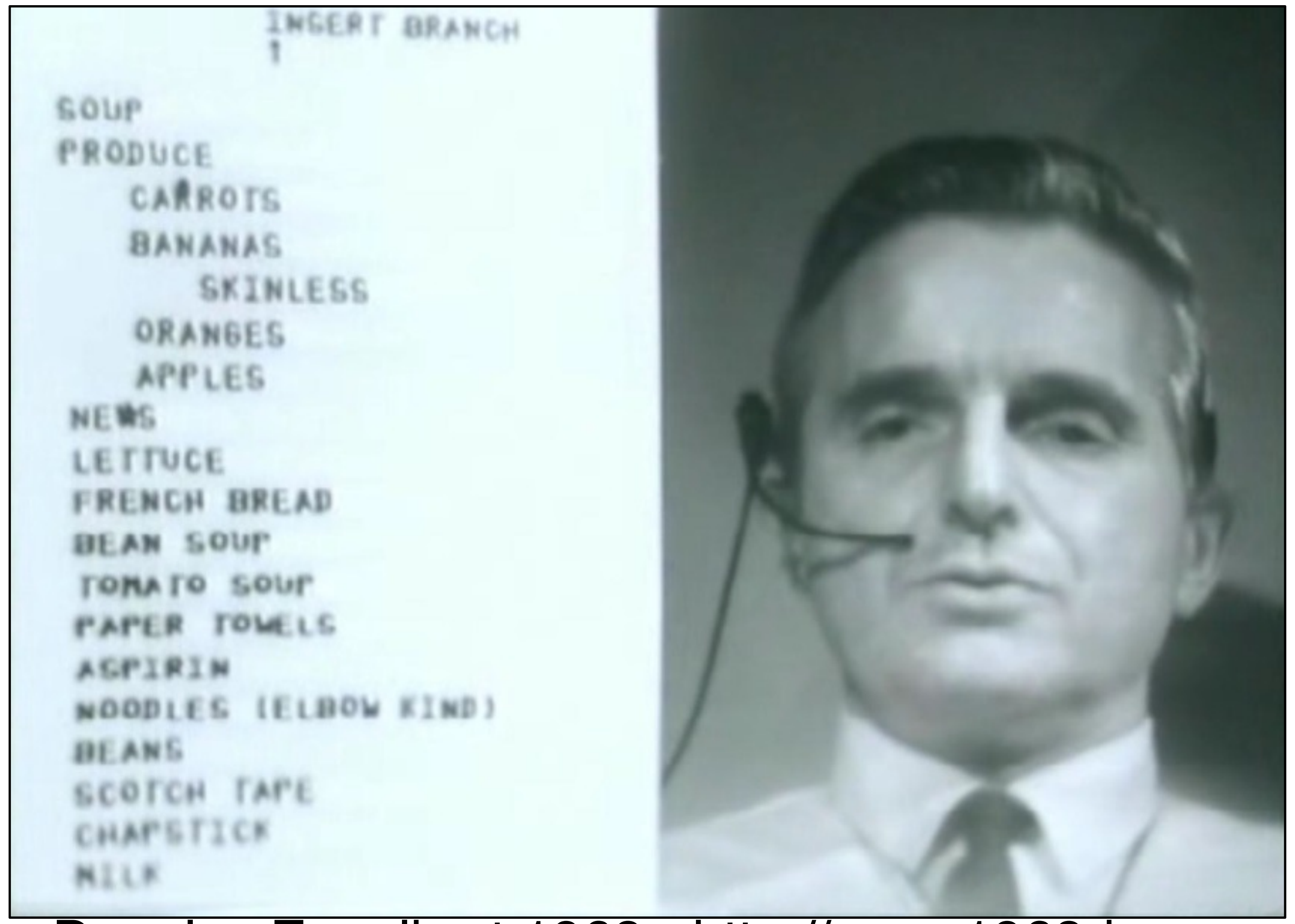
http://billmill.org/pymag-trees/



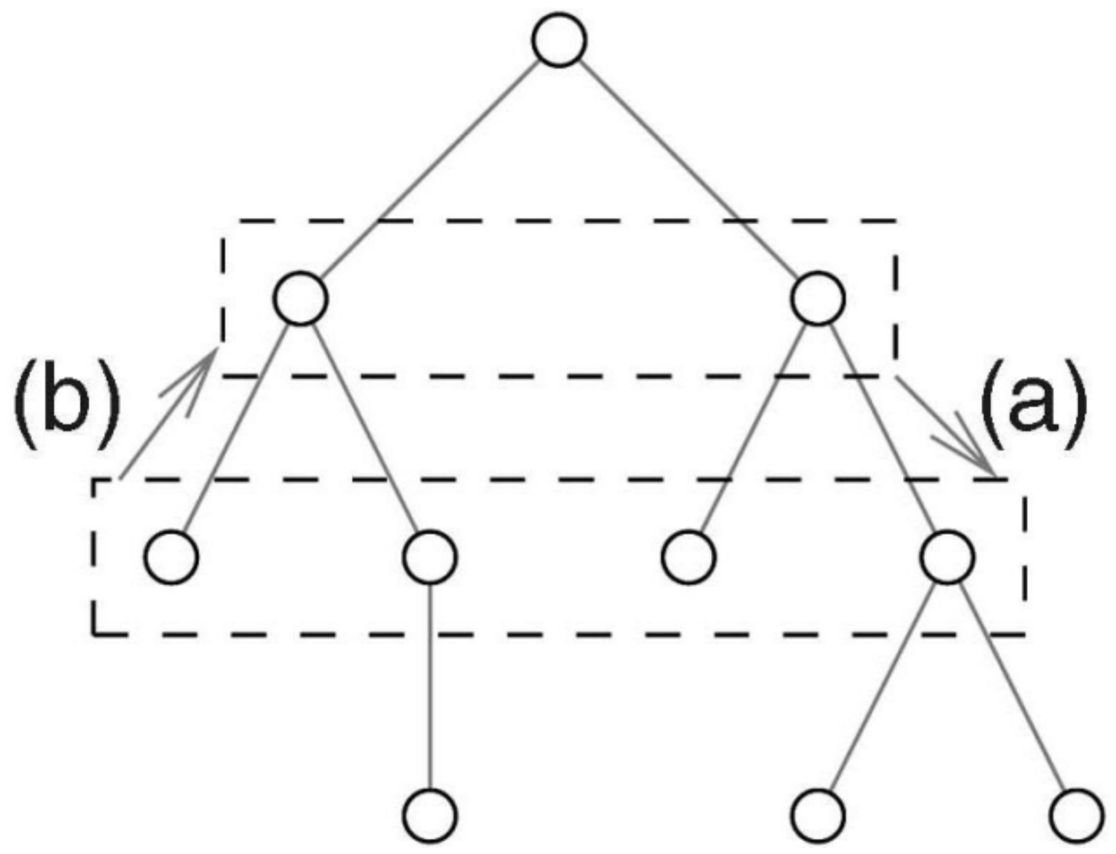


Manipulating Aggregation Levels

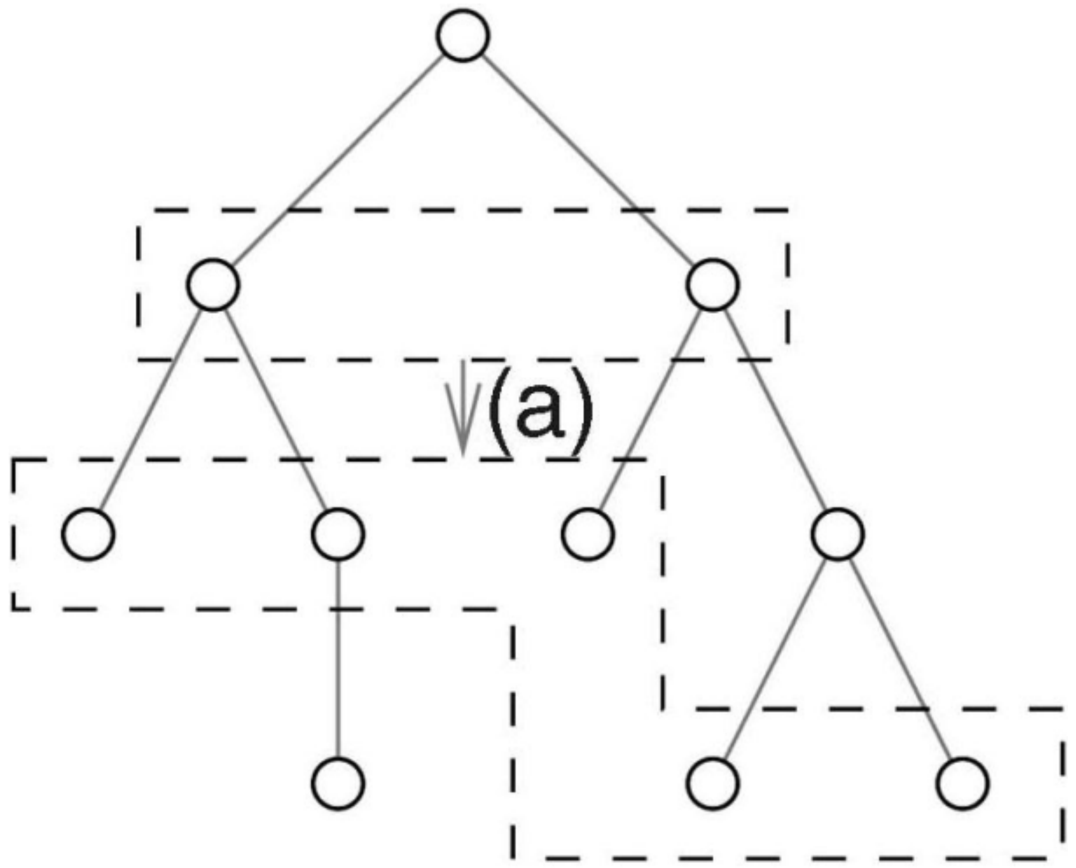
First interactive tree manipulation



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



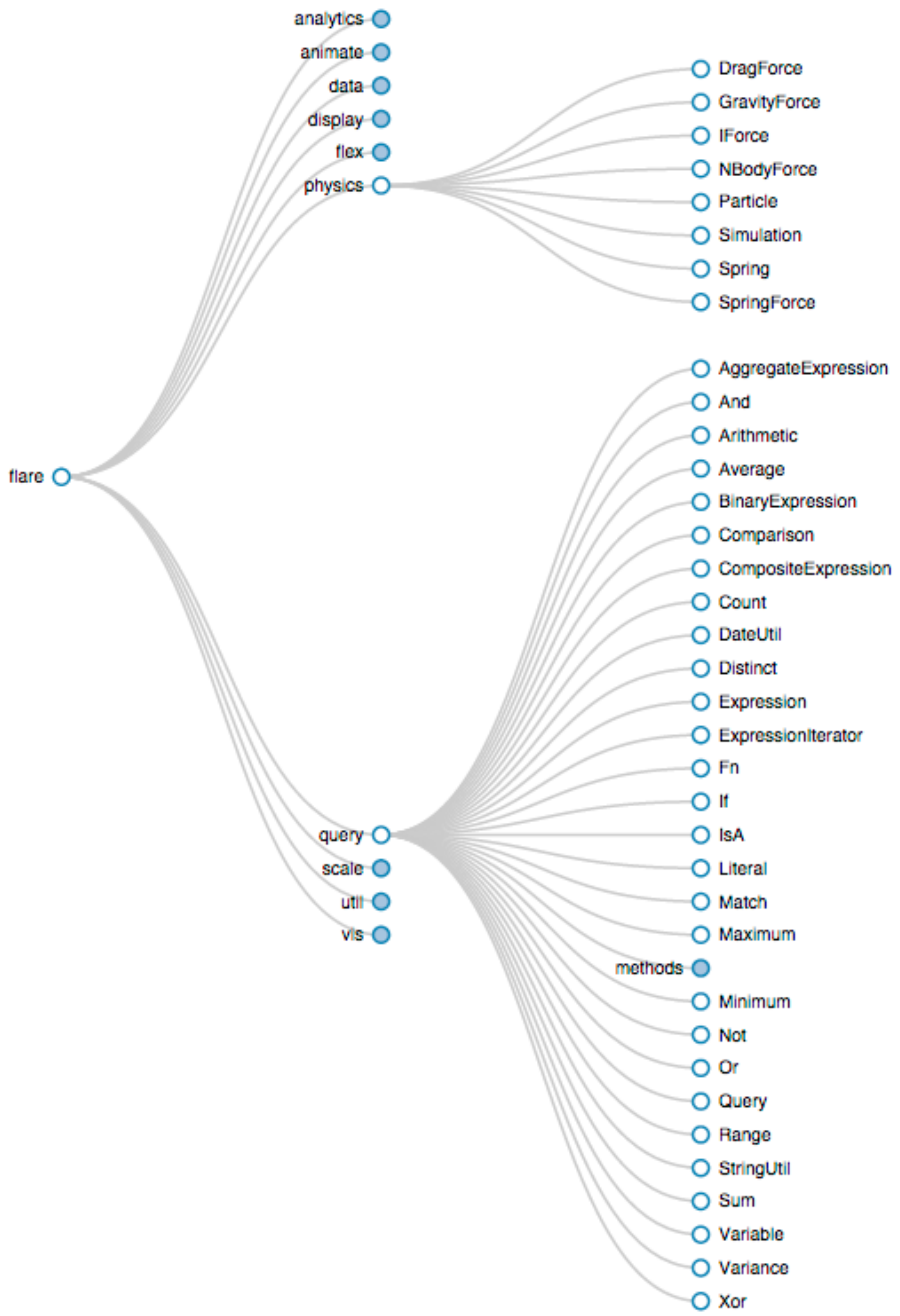
(a) Drill-Down (b) Roll-Up

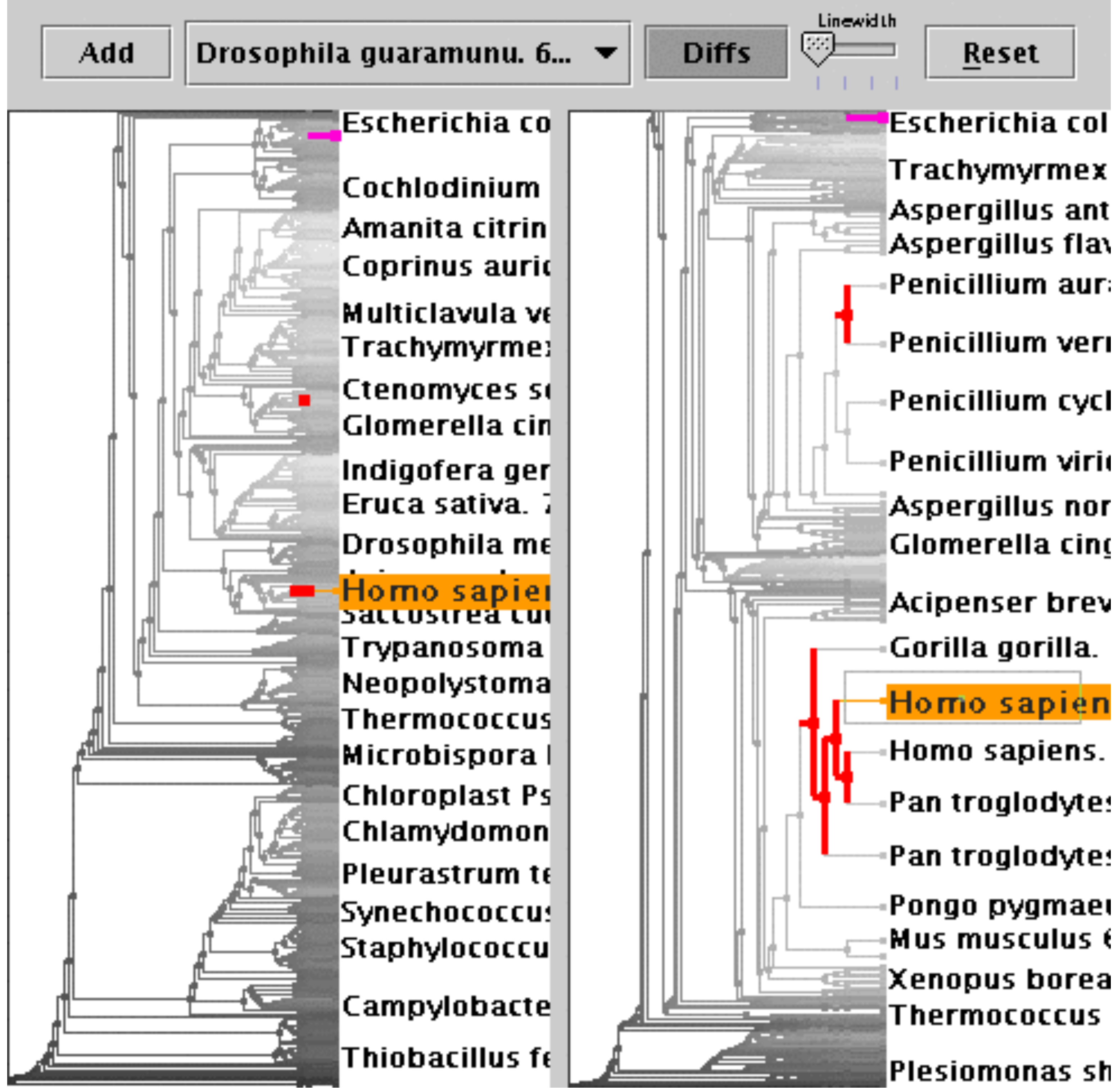


(a) Unbalanced Drill-Down

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

Tree Interaction, Tree Comparison





Explicit Representations

Pros:

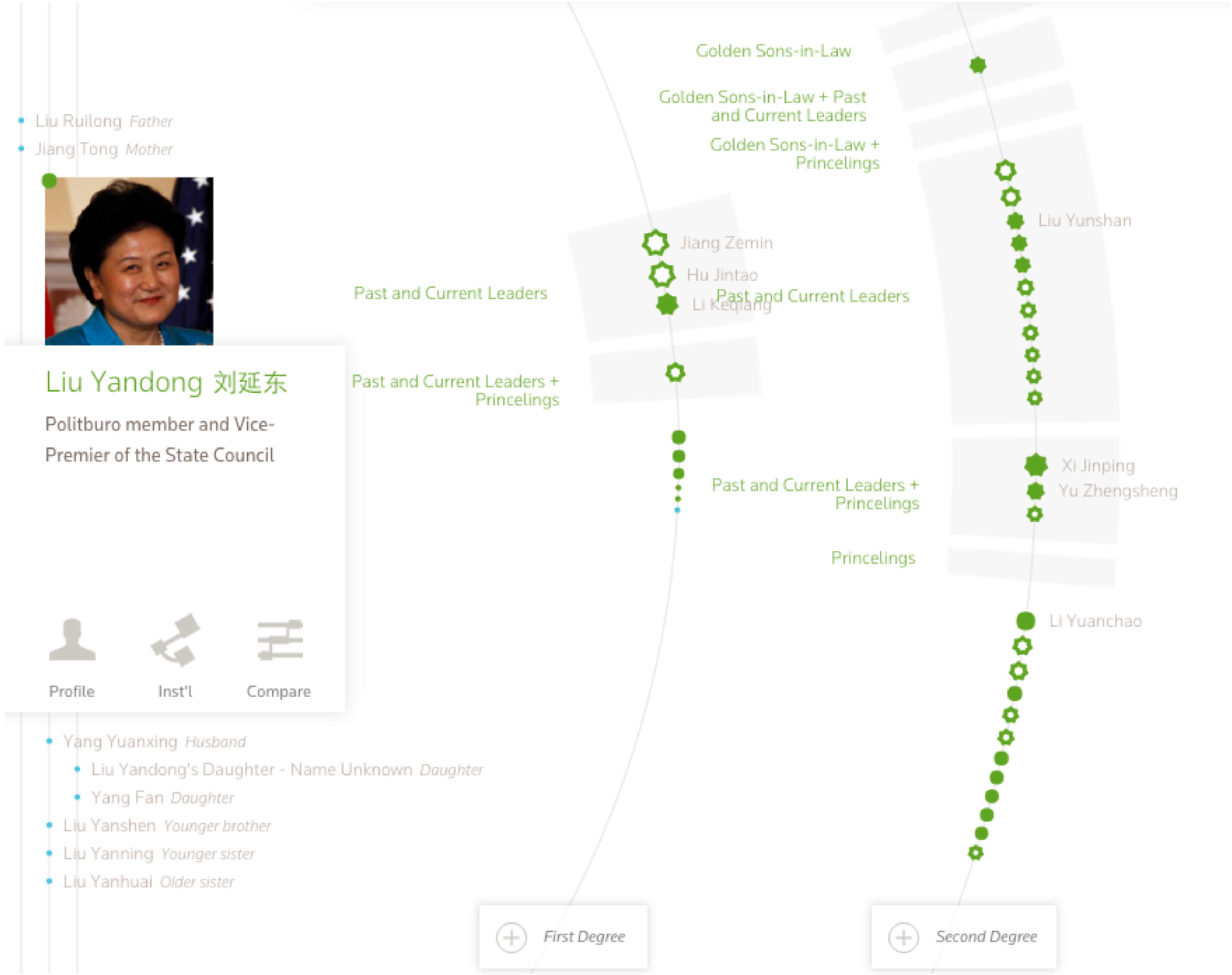
- able to depict all graph classes
- can be customized by weighing the layout constraints
- very well suited for TBTs, if also a suitable layout is chosen

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(n3))
- has a tendency to clutter (edge clutter, “hairball”)

Design Critique

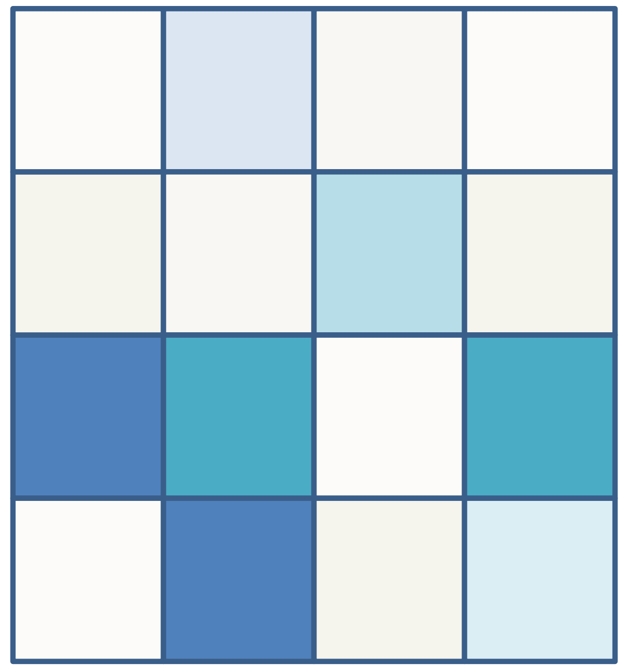
Connected China



https://goo.gl/YXkWYX

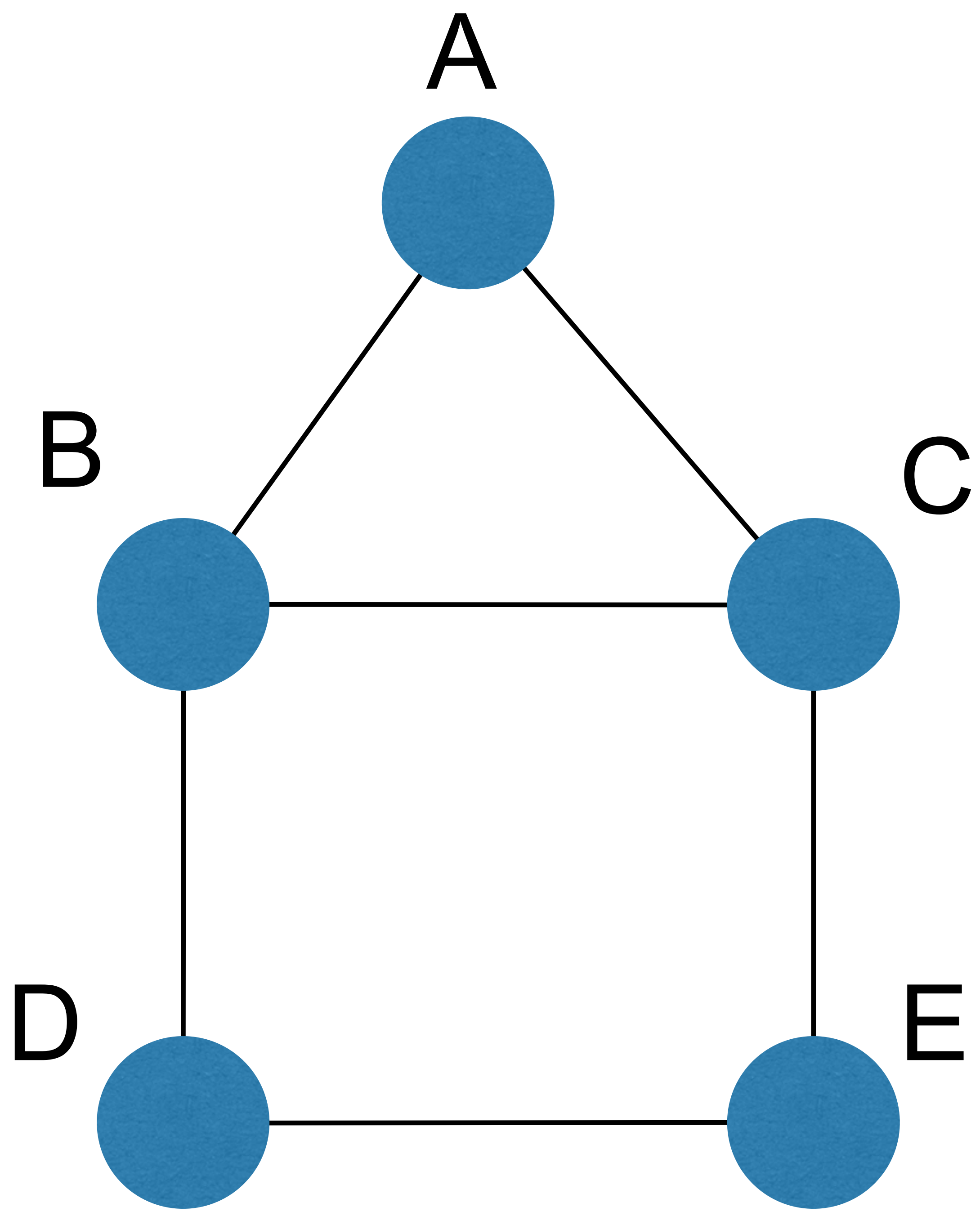
http://china.fathom.info/

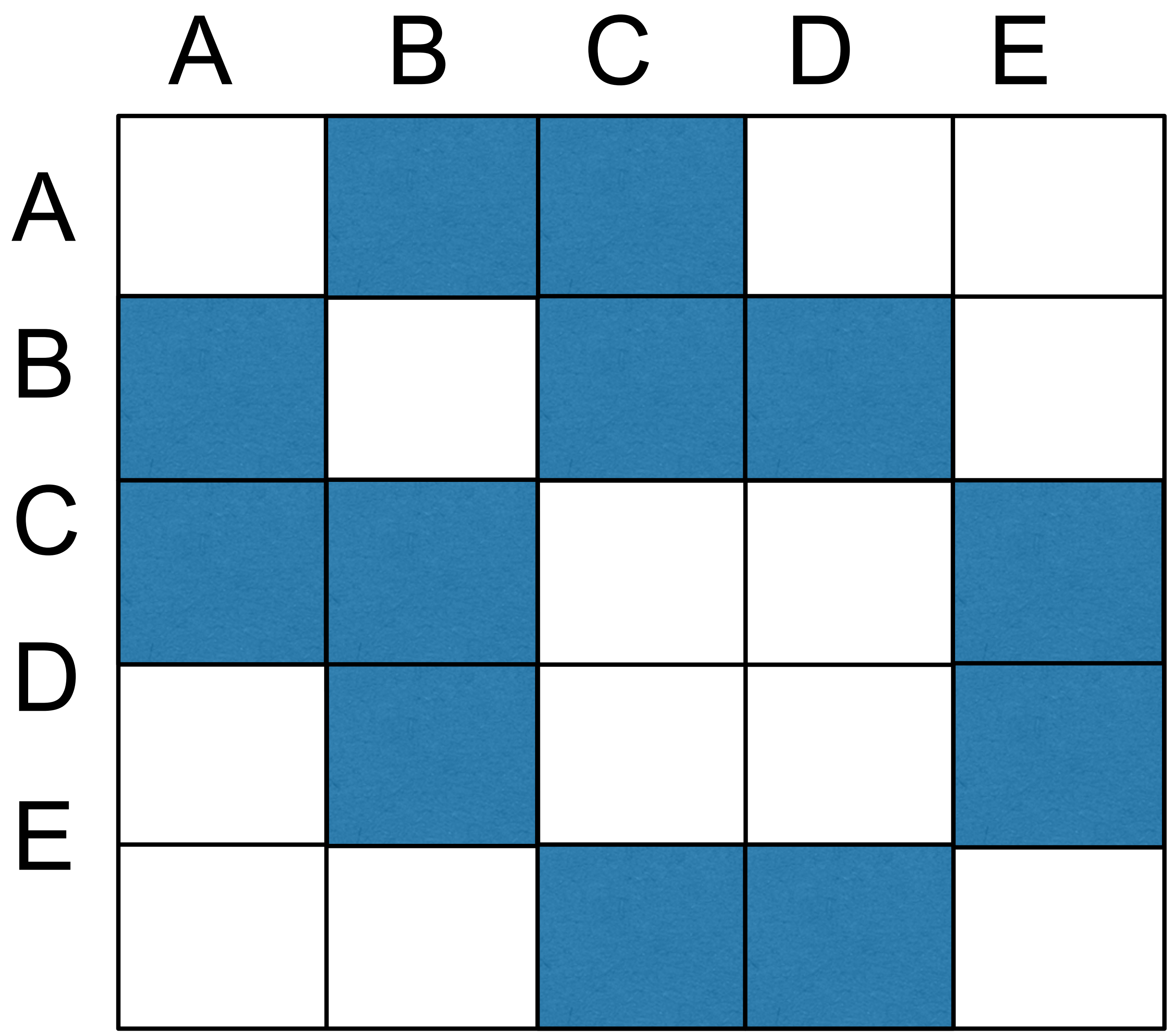
Matrix Representations



Matrix Representations

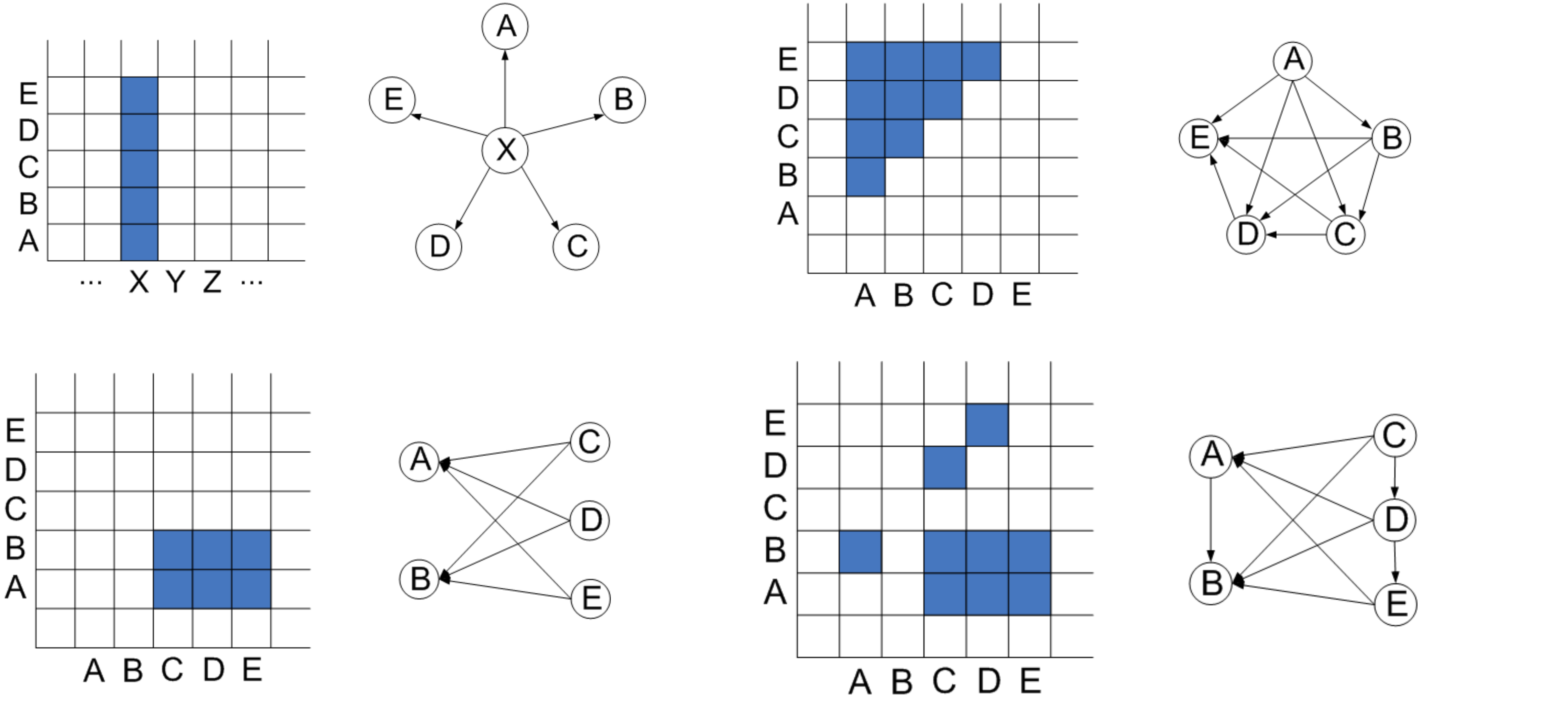
Instead of node link diagram, use adjacency matrix





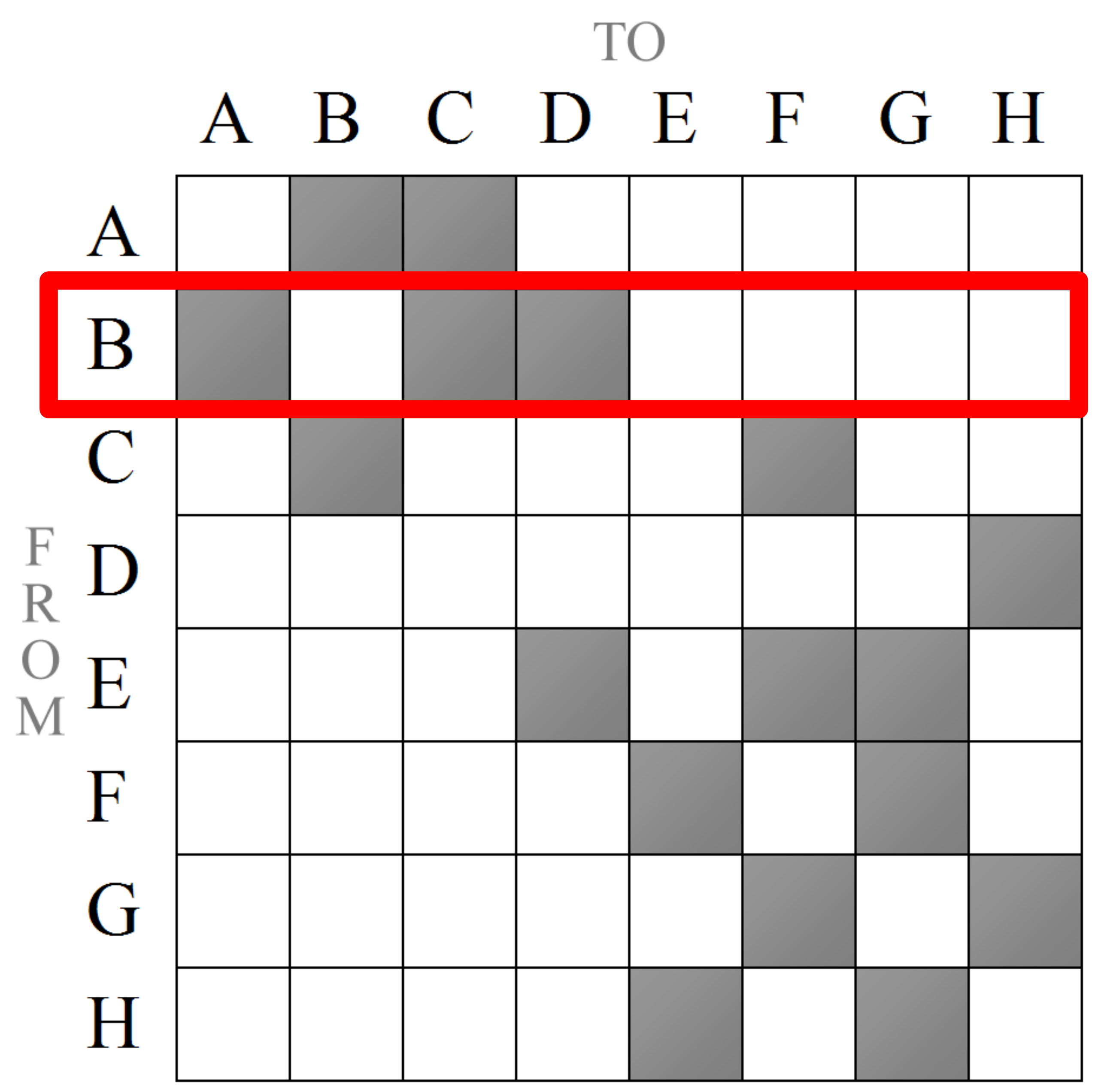
Matrix Representations

Examples:

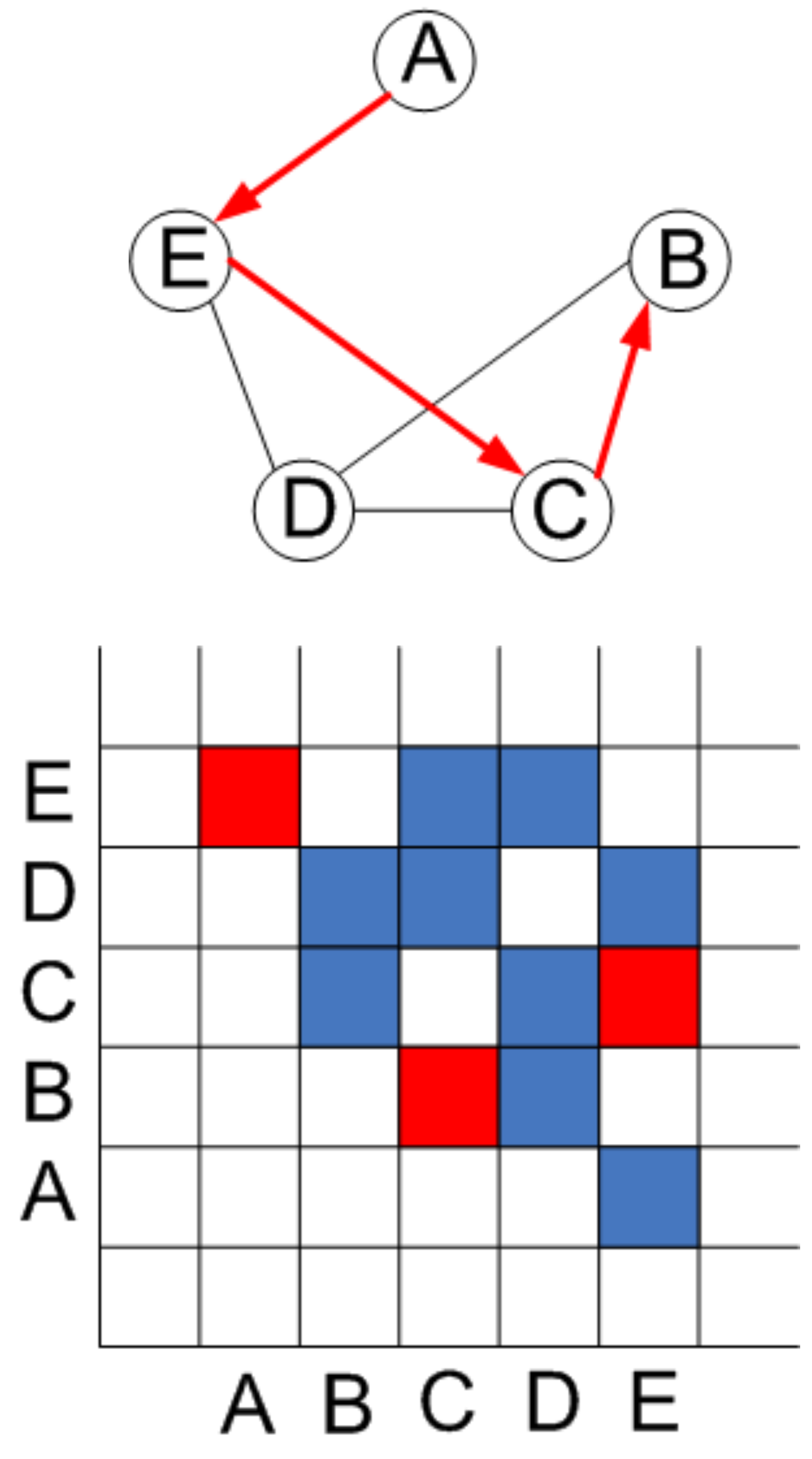


HJ Schulz 2007

Matrix Representations

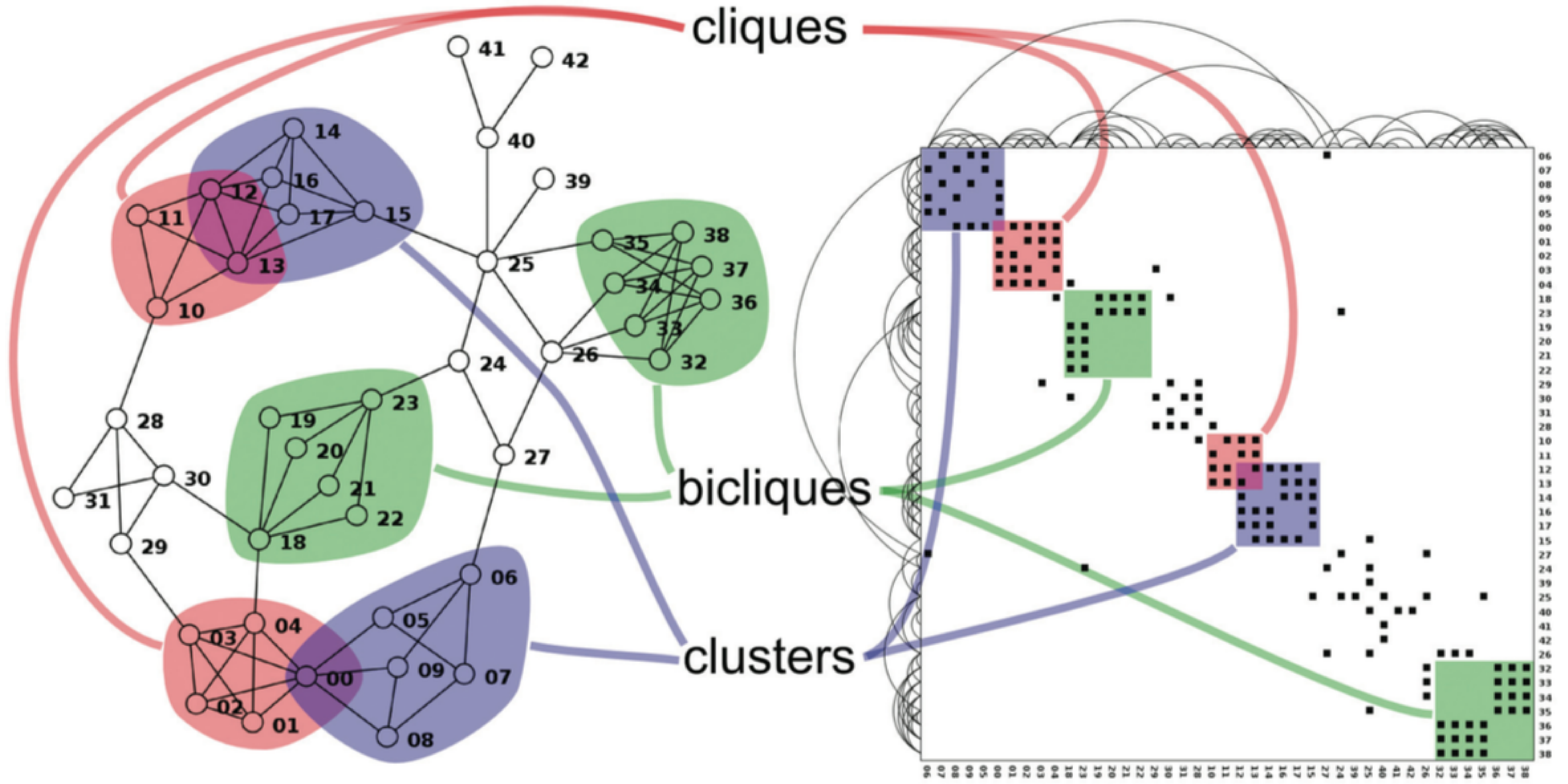


Well suited for
neighborhood-related TBTs



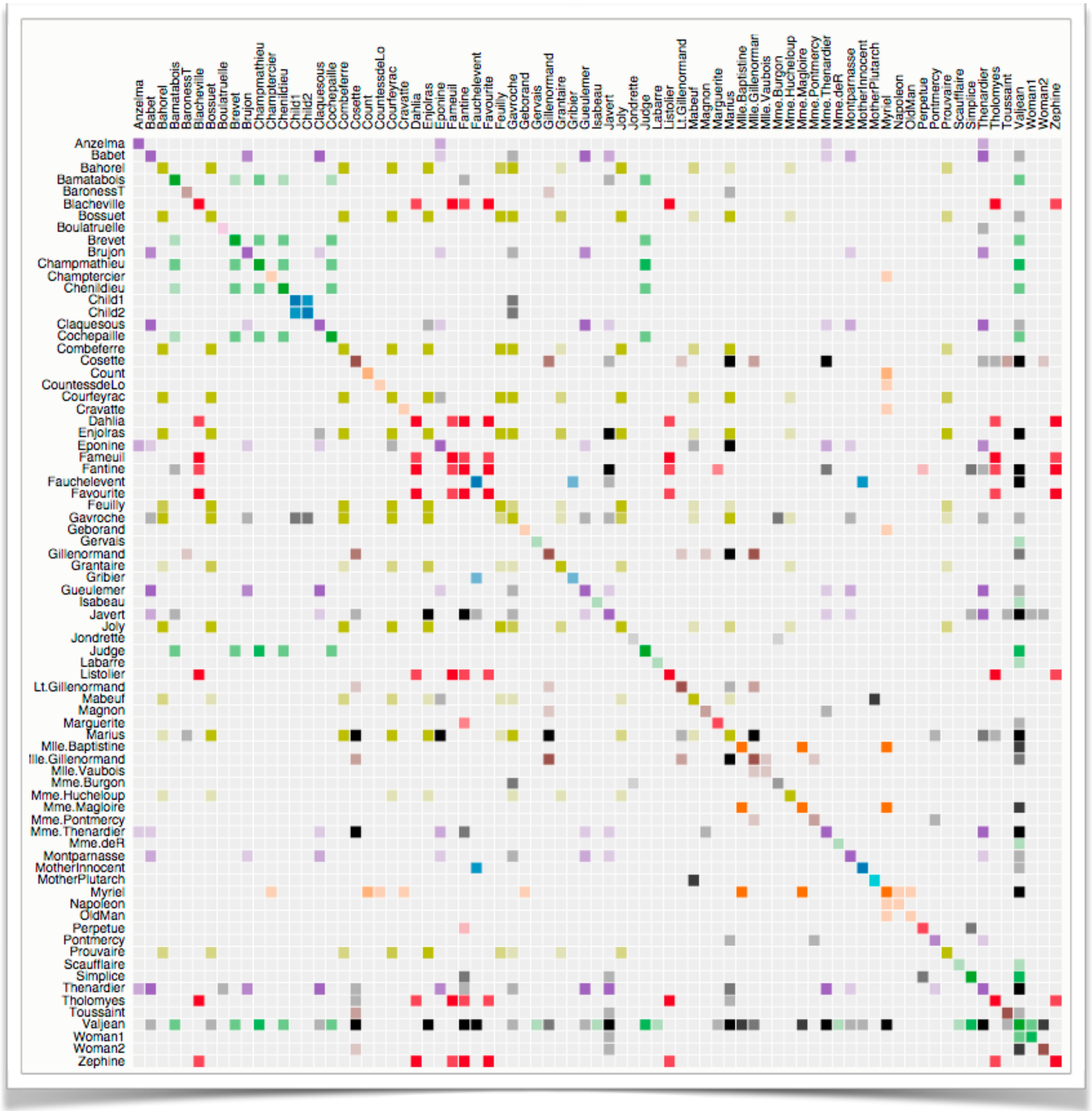
Not suited for
path-related TBTs

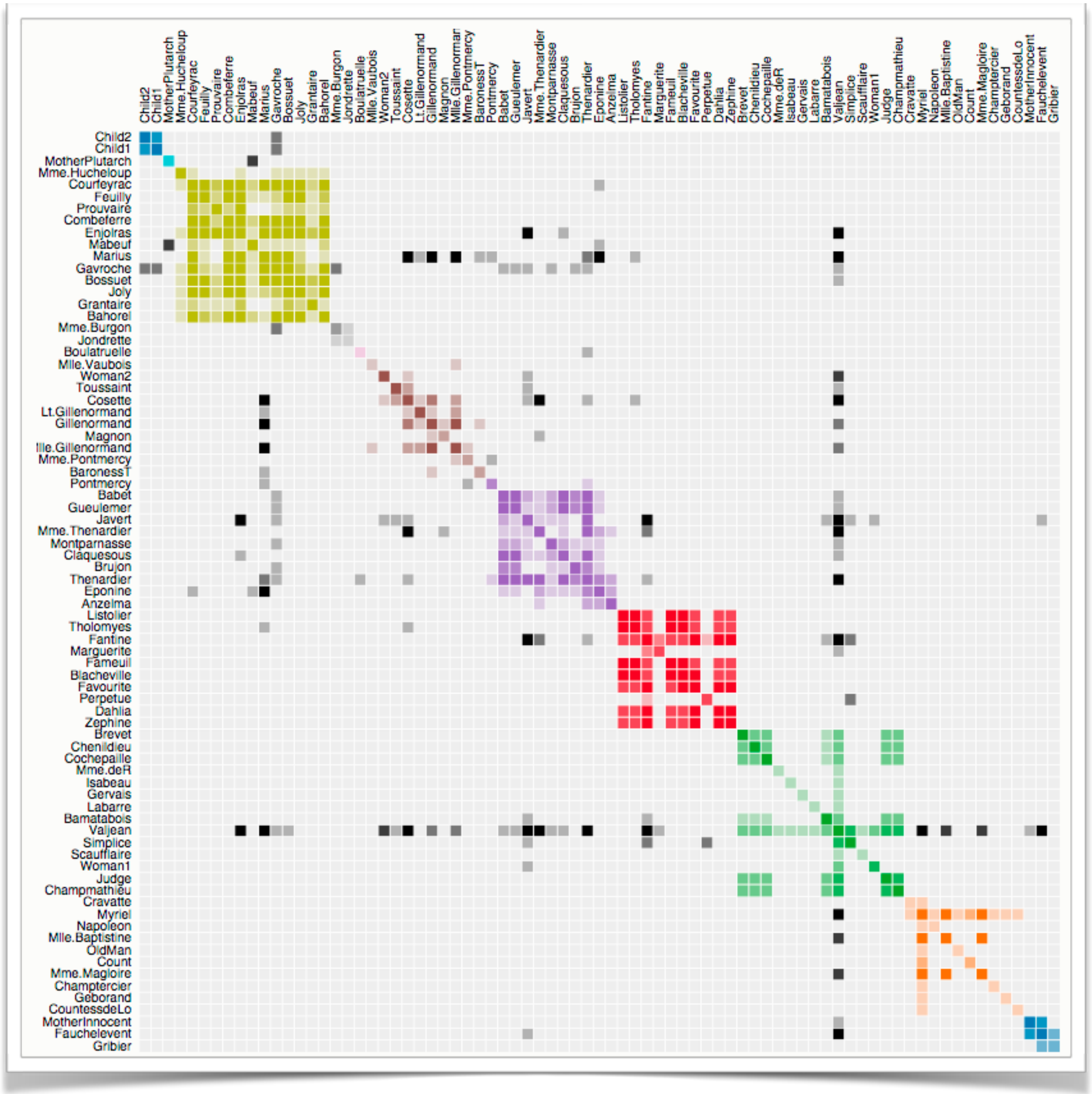
van Ham et al. 2009
Shen et al. 2007



McGuffin 2012

Order Critical!





Matrix Representations

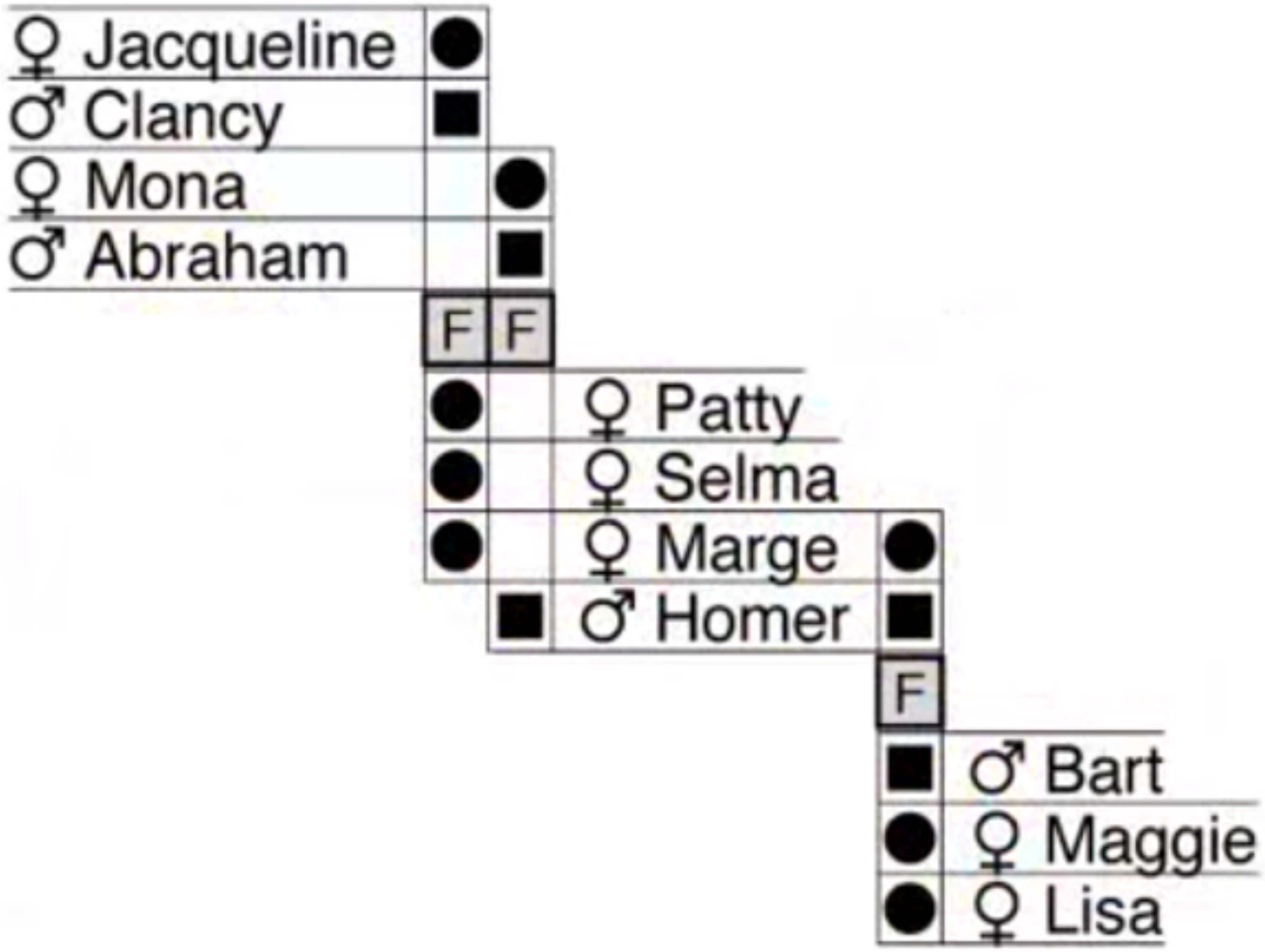
Pros:

- 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

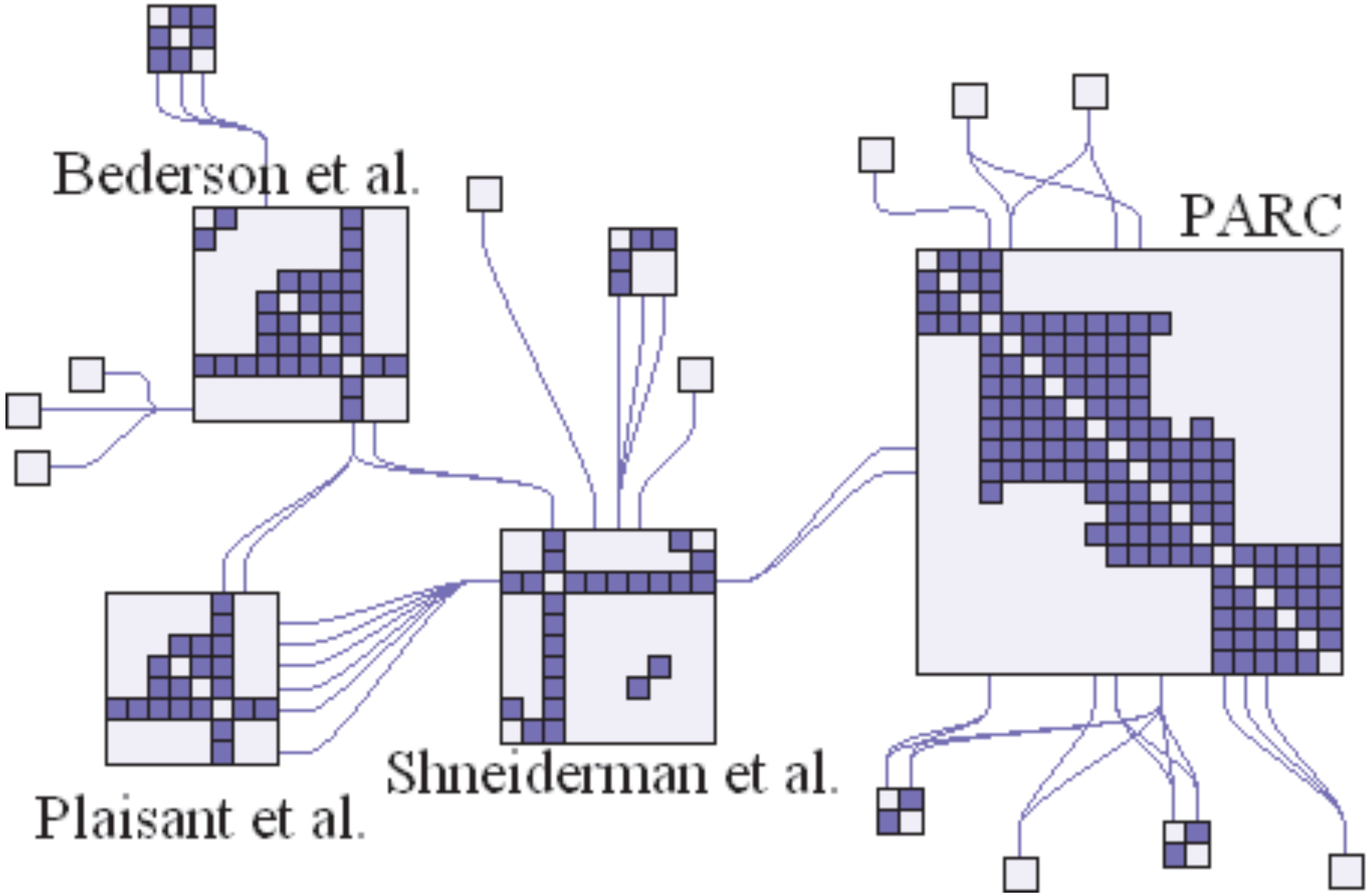
Cons:

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

Special Case: Genealogy



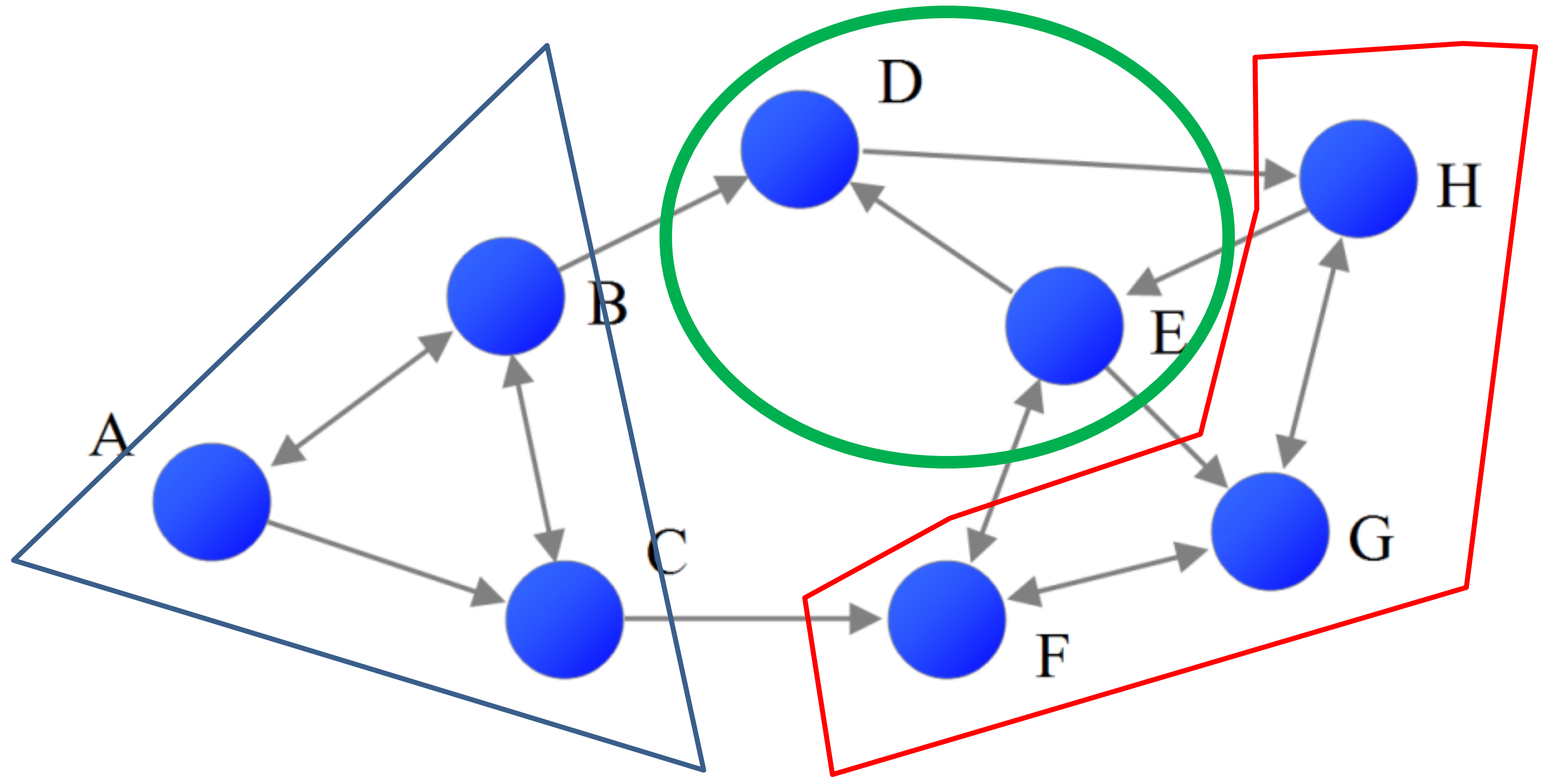
Hybrid Explicit/Matrix

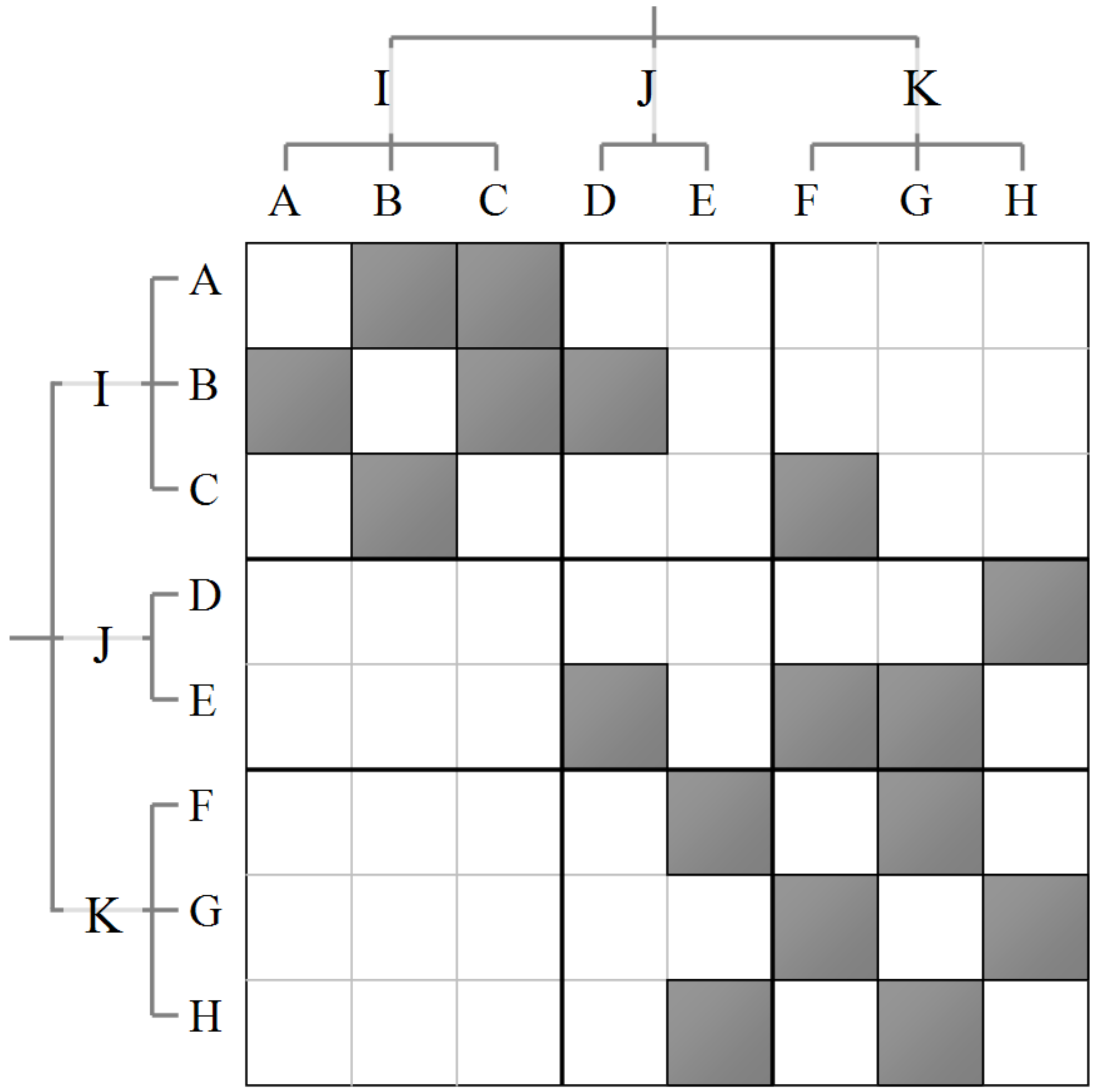


NodeTrix
[Henry et al. 2007]

Matrix Representations

Problem #1: used screen real estate is quadratic in the number of nodes
Solution approach: hierarchization of the representation

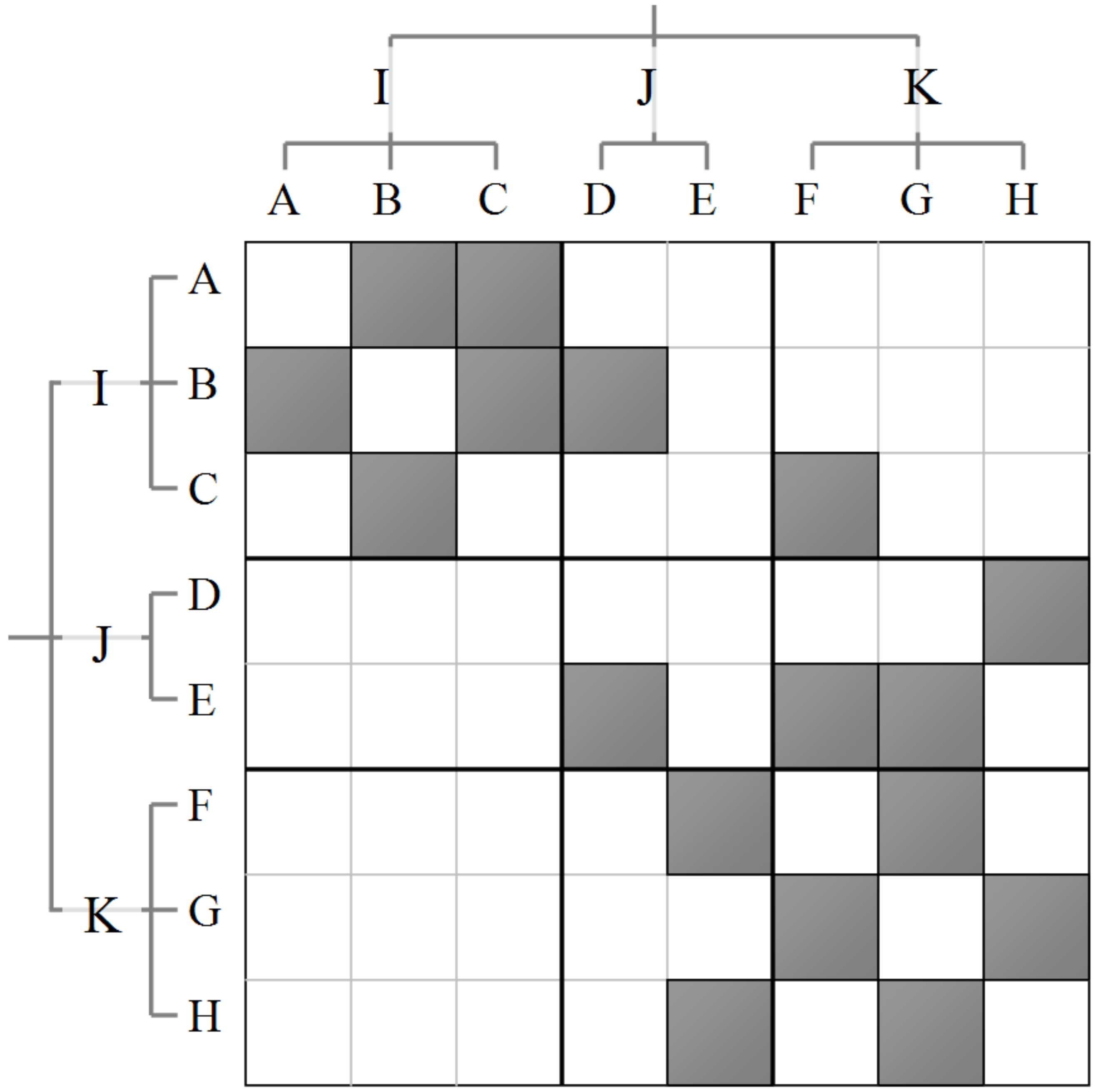


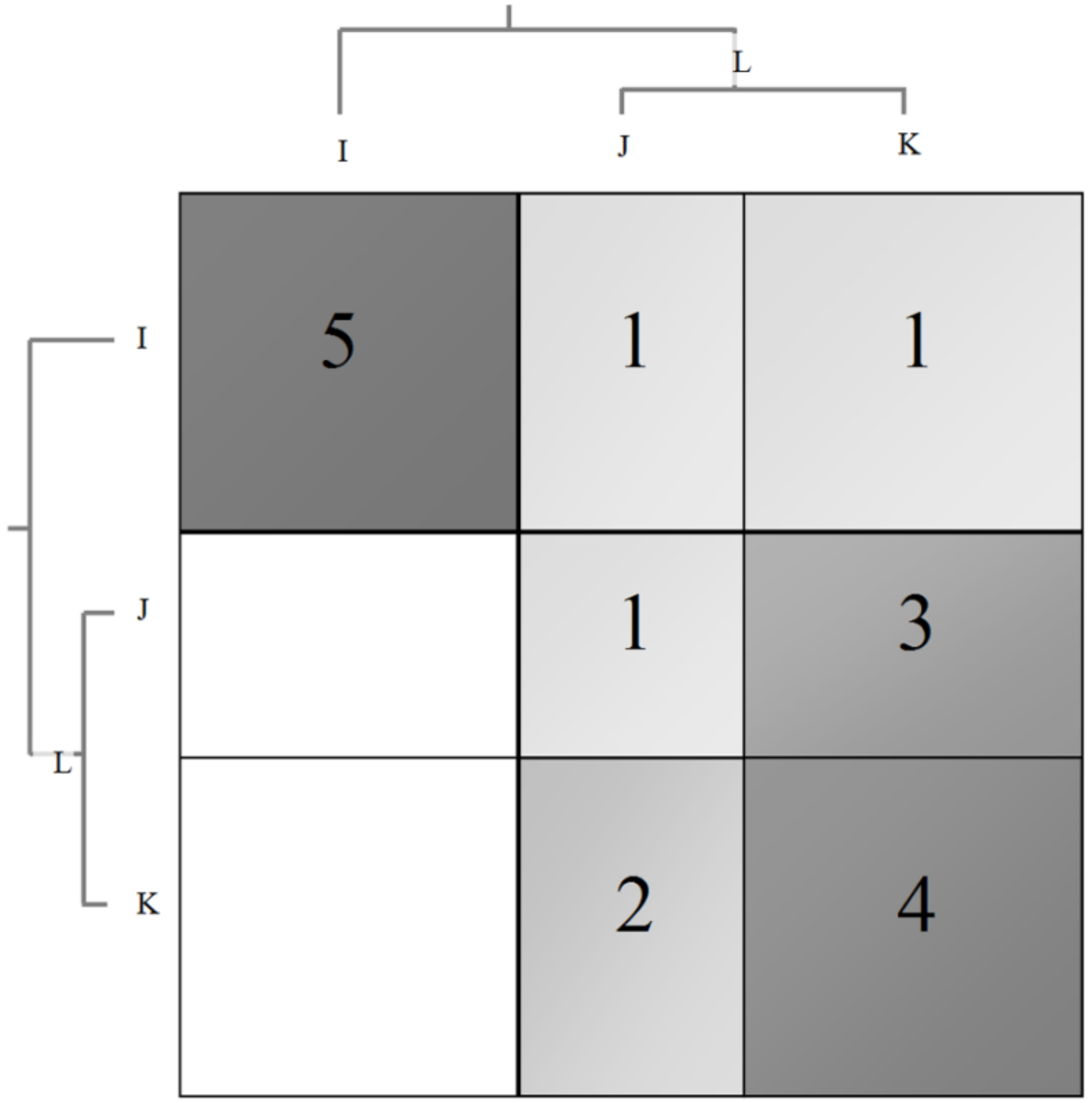


[van Ham et al. 2009]

Matrix Representations

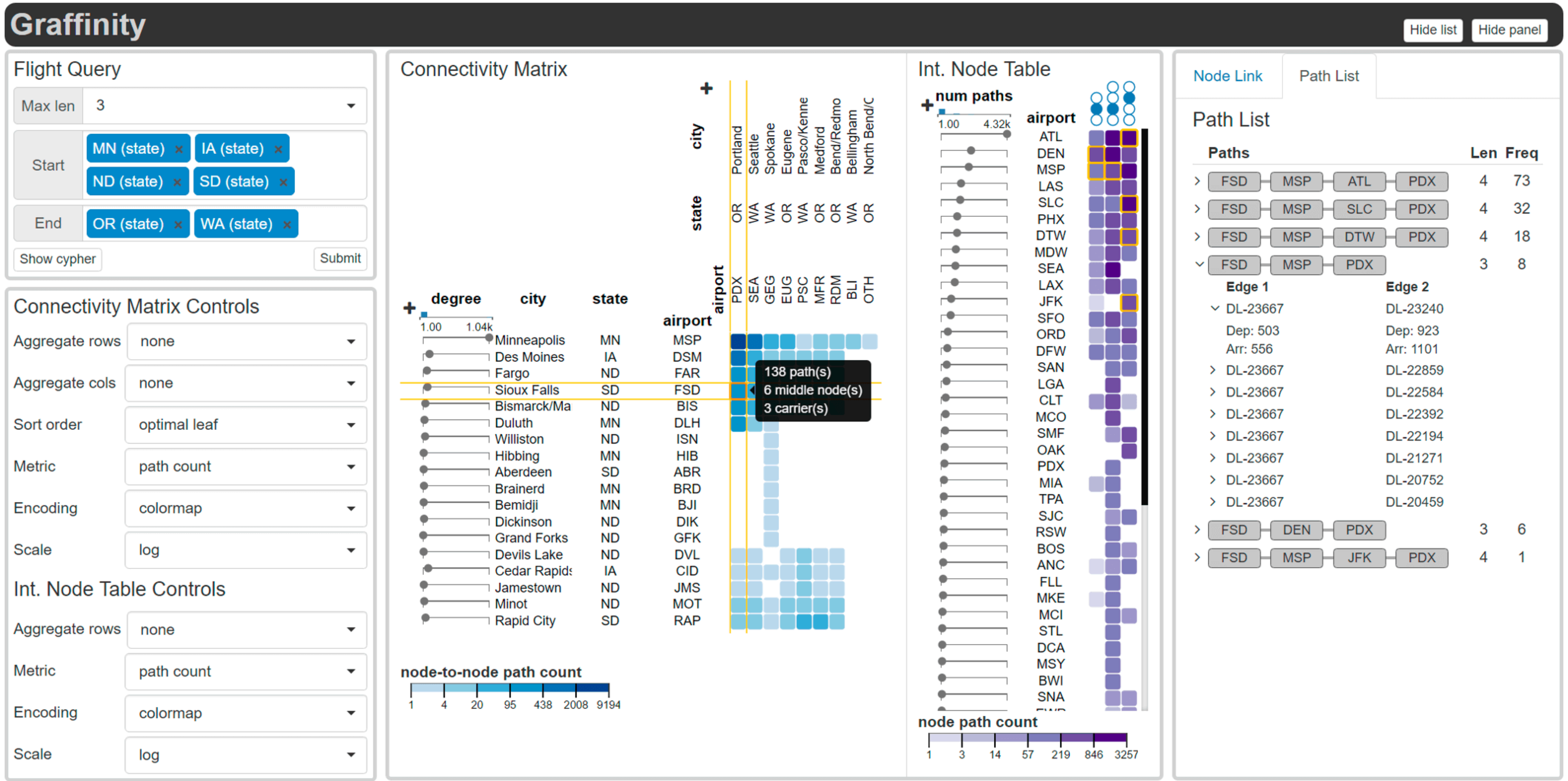
Problem #1: used screen real estate is quadratic in the number of nodes
Solution approach: hierarchization of the representation





[van Ham et al. 2009]

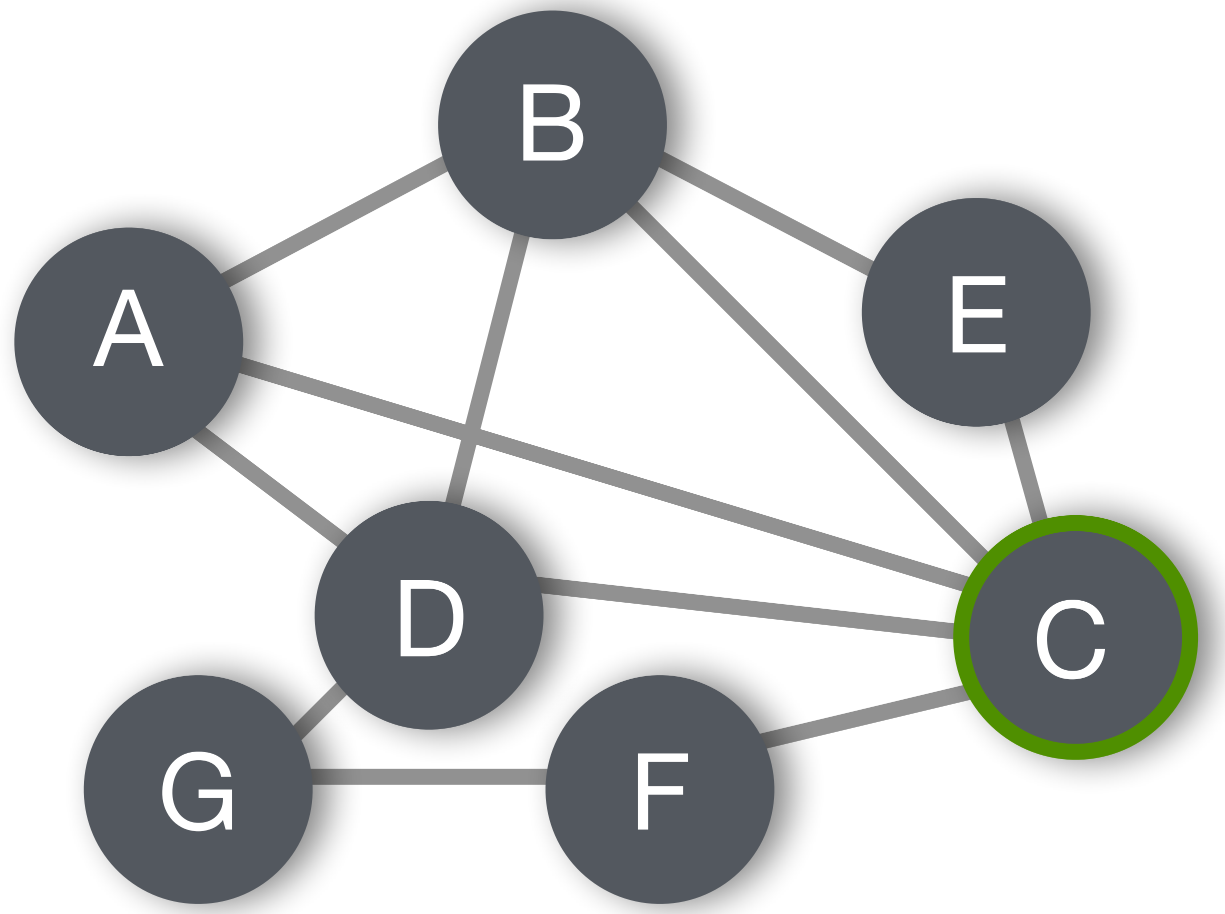
Higher-Order Connectivity



[Kerzner et al., Graffinity, 2017]

Multivariate Networks

What is a Multivariate Network?



Node	Year	Citations	Type	...
A	2006	13	Journal	...
B	2009	36	Conference	...
C	2017	5	Journal	...
...	...	...	...	

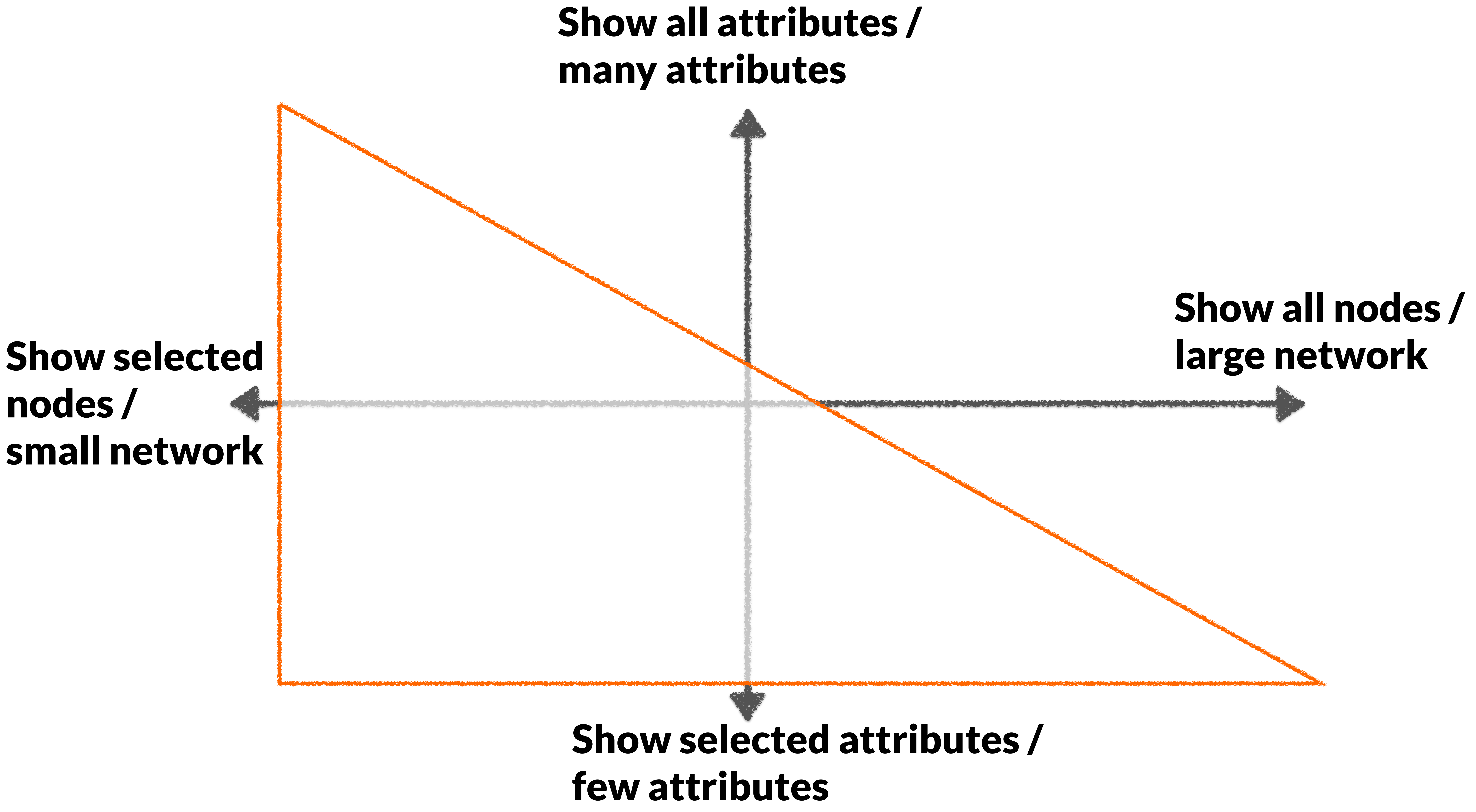
Challenge: Consider topology and attributes simultaneously

Networks and Attributes

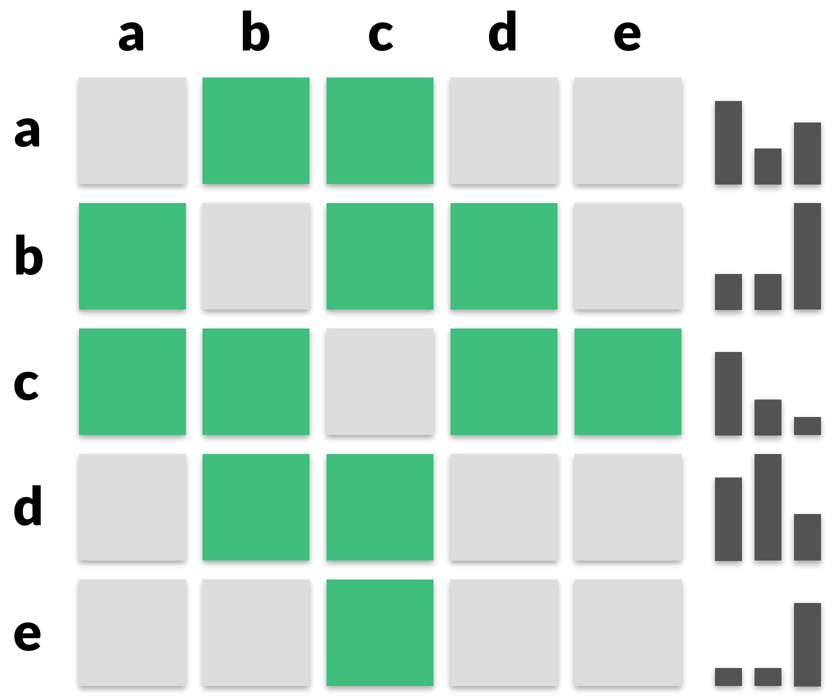
Attributes can influence topology

Path can be slow / blocked

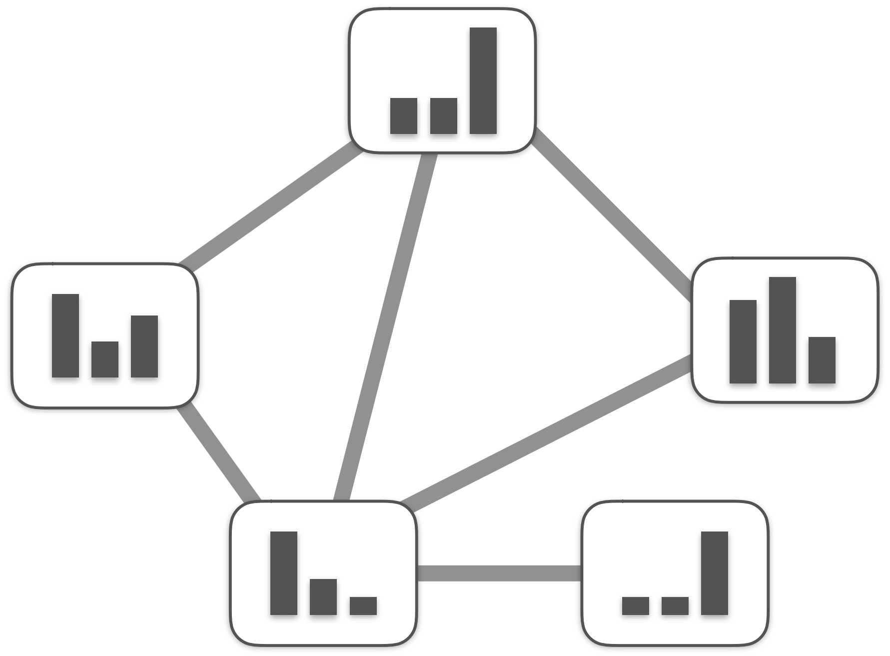
- best route when driving depends on traffic
- biological network depends on many factors


Show all attributes / many attributes
Show selected nodes / small network
Show all nodes / large network
Show selected attributes / few attributes

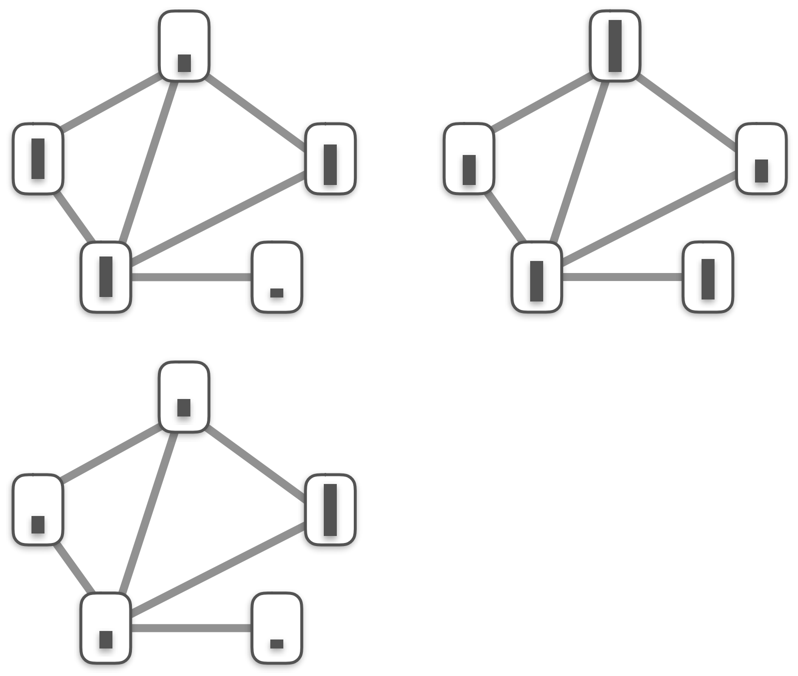
Multivariate Network Visualization Strategies



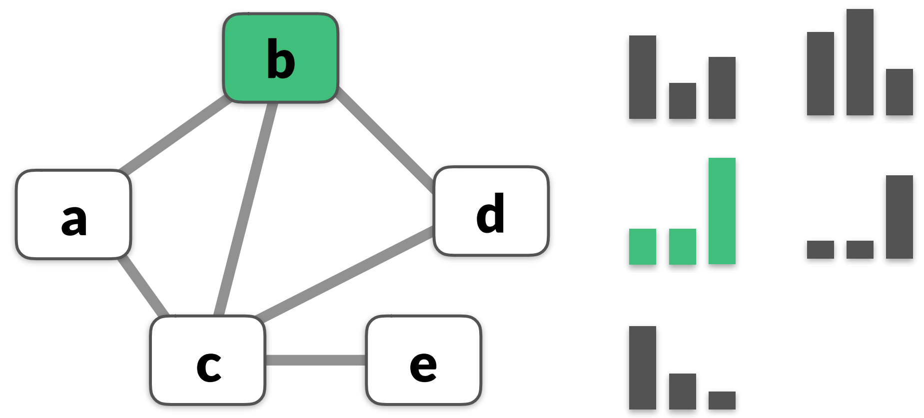
Matrices



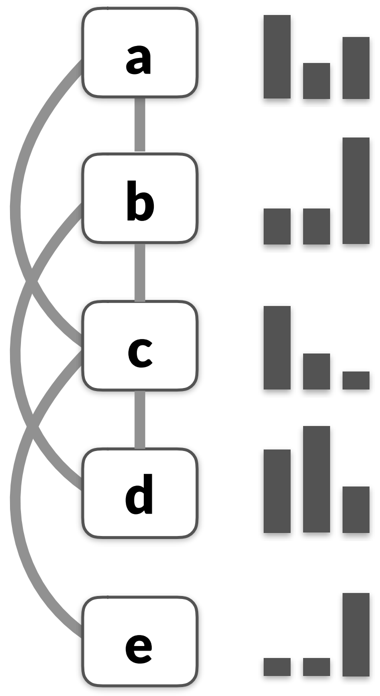
On-Node Encoding



Small Multiples

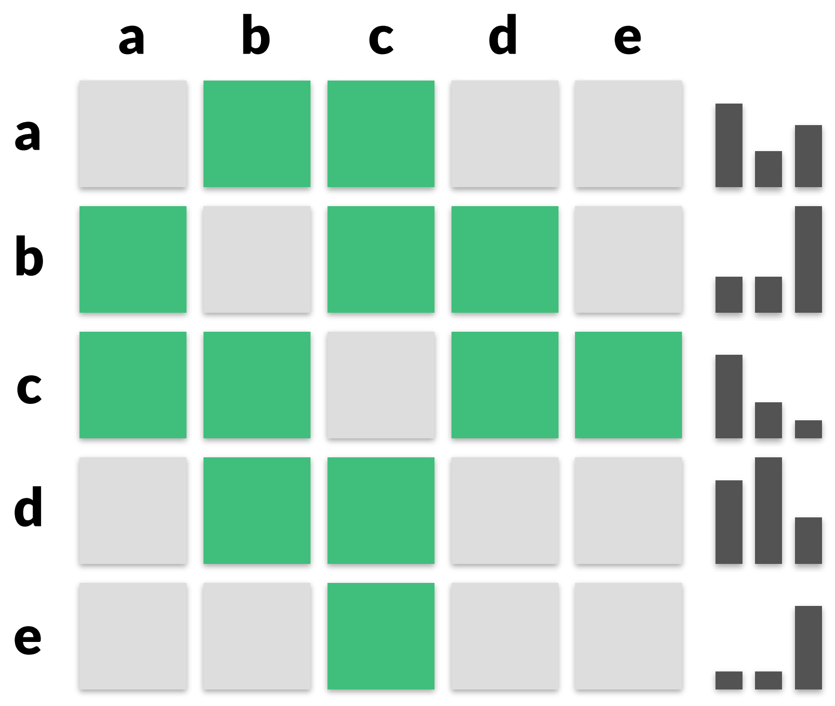


Multiple Coordinated Views



Layout Adaption

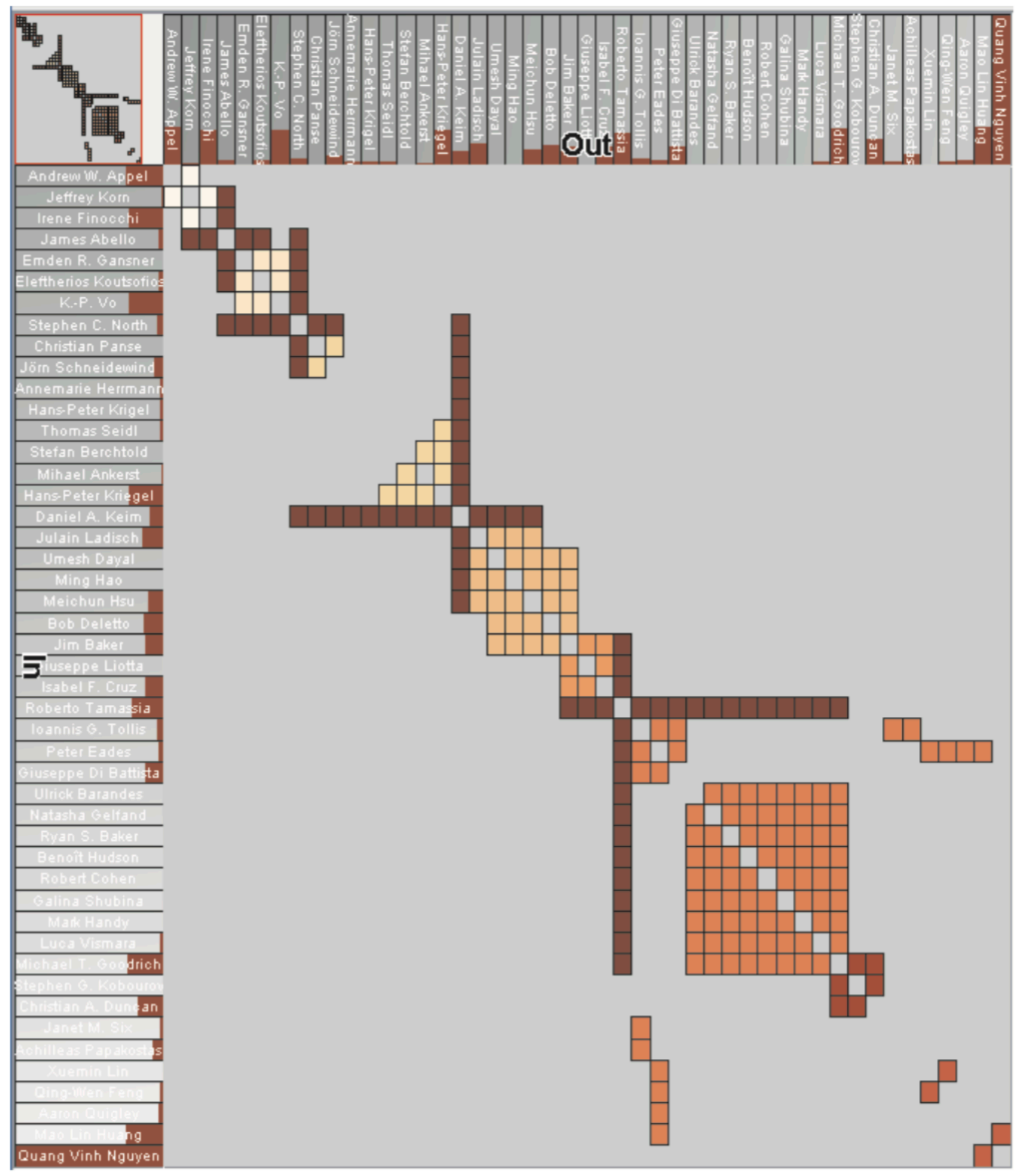
Matrices



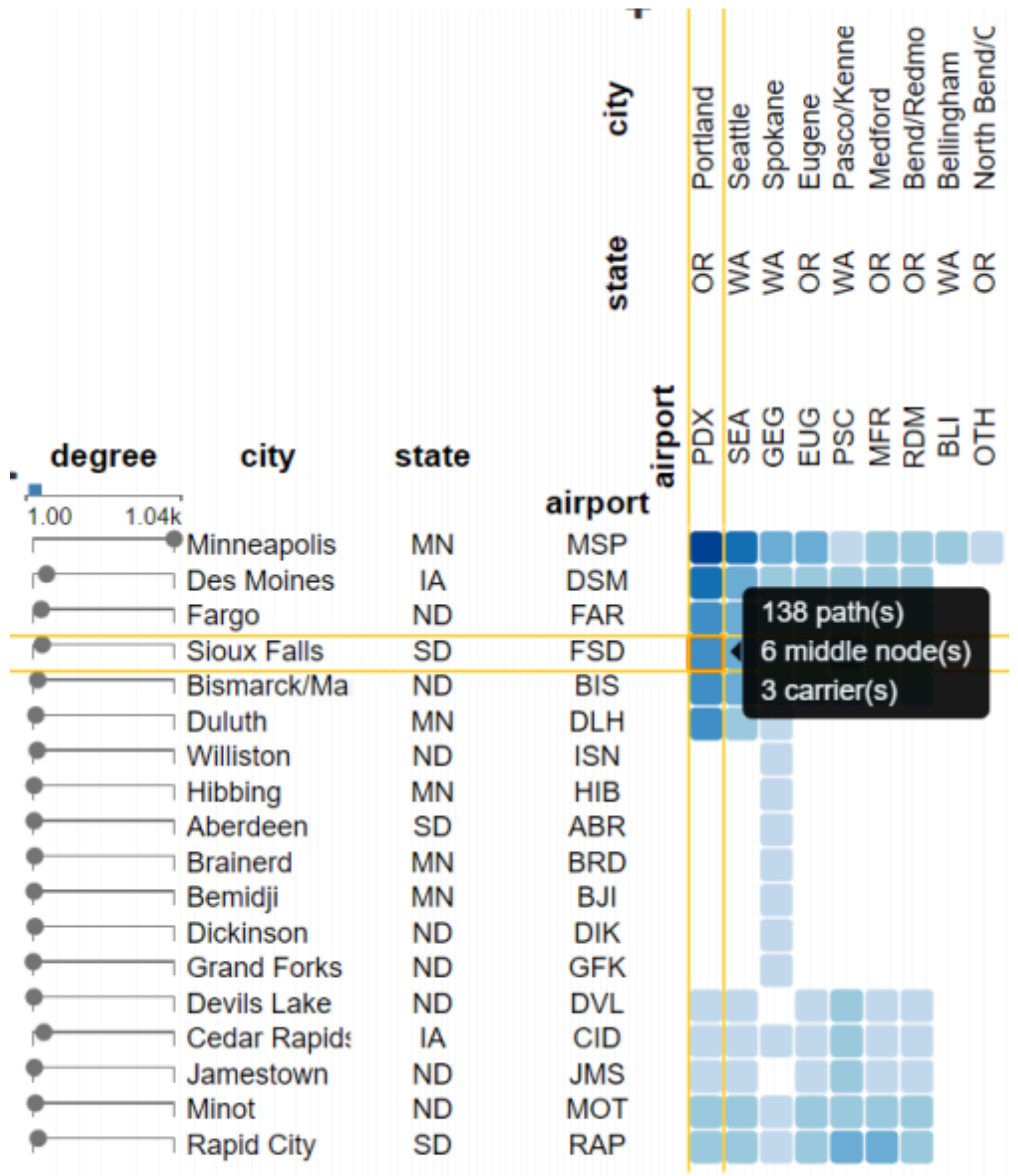
Easy to encode edge attributes in cells

Easy to encode node attributes adjacent to matrix

Common pros and cons of matrices

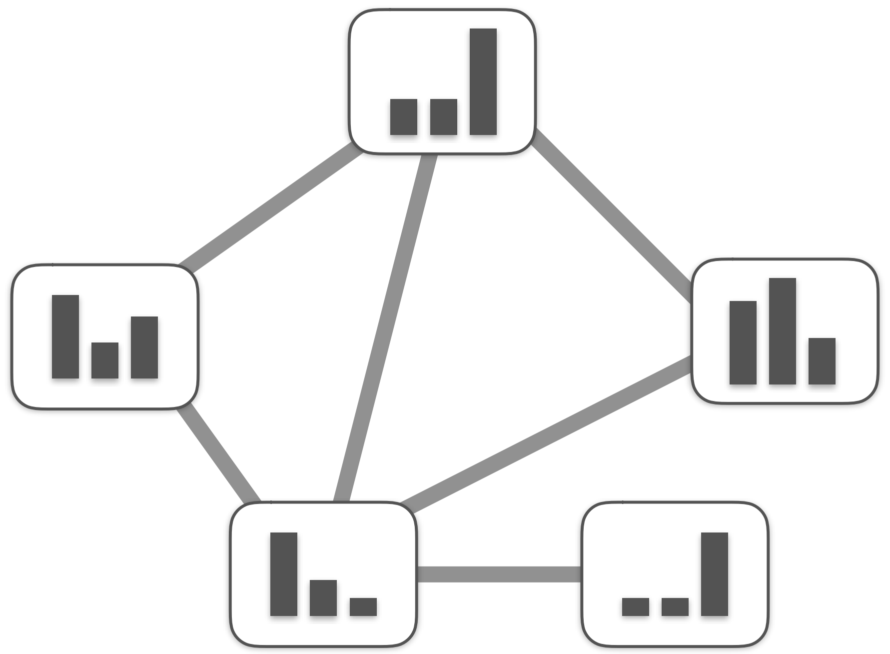


[Henry and Fekete, MatrixExplorer, 2006]



[Kerzner et al., Graffinity, 2017]

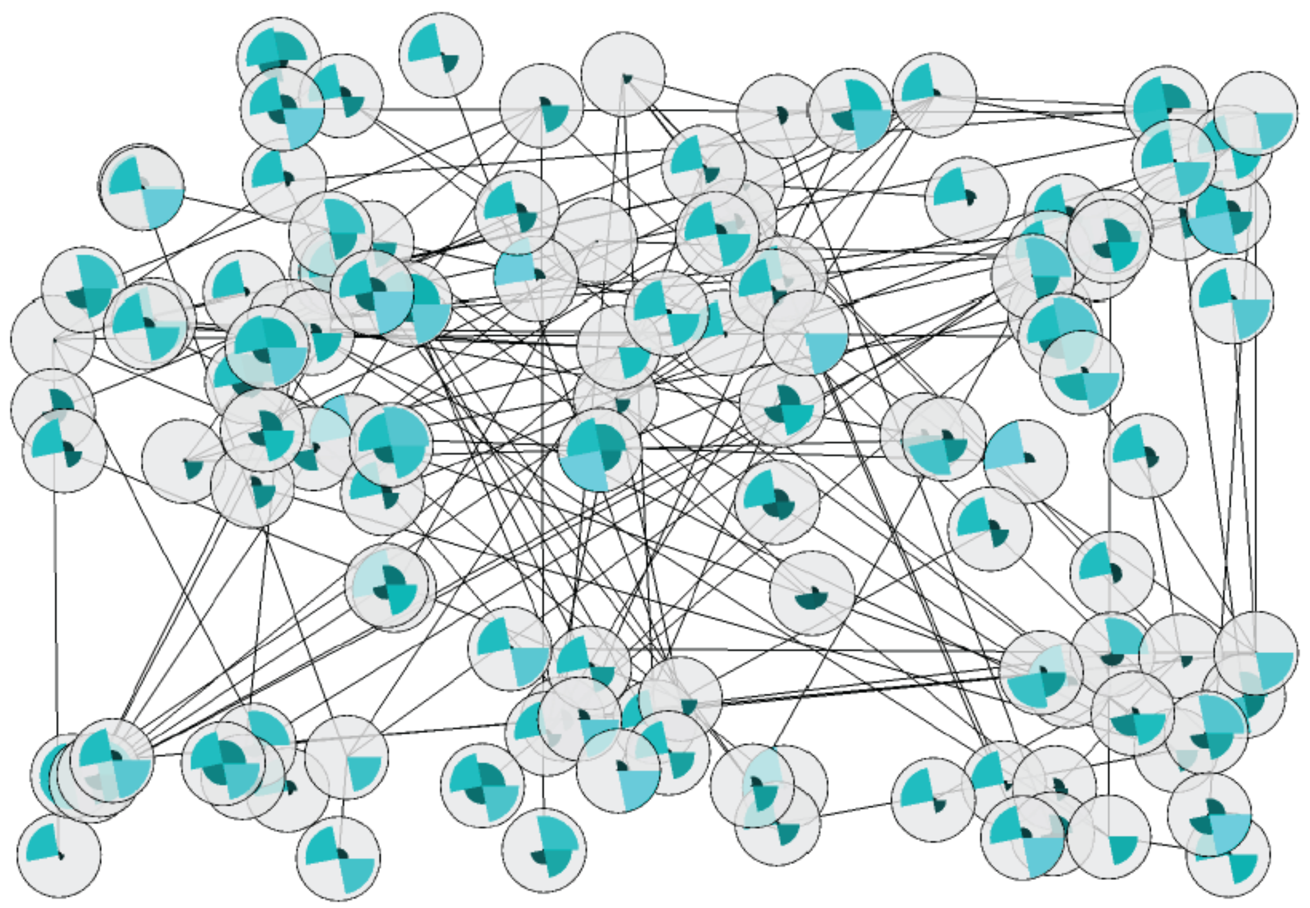
On-Node Encoding



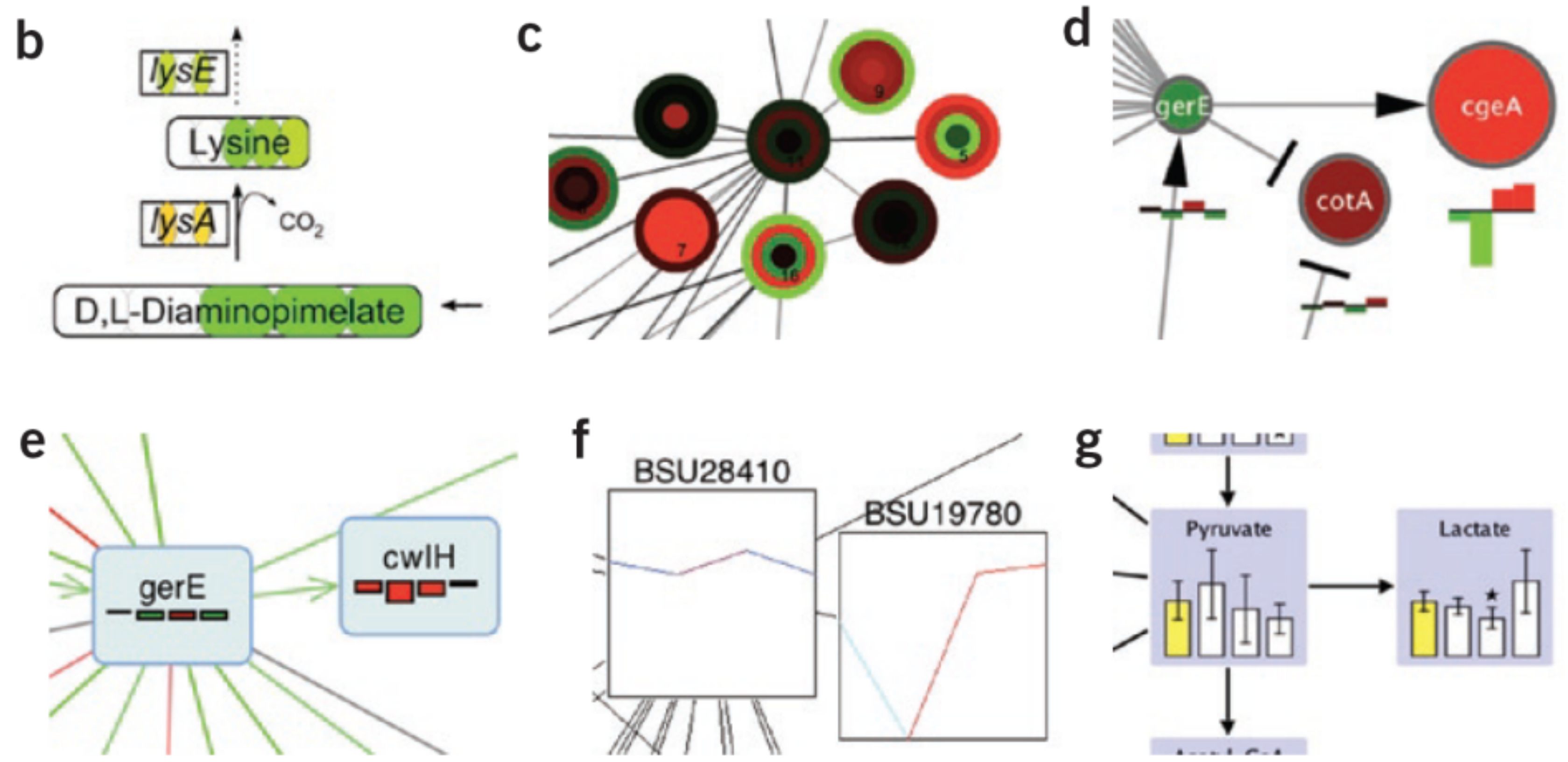
Canonical way to visualize single attribute. Widely supported.

Ideal for topology-attribute interaction

Tricky for multiple attributes

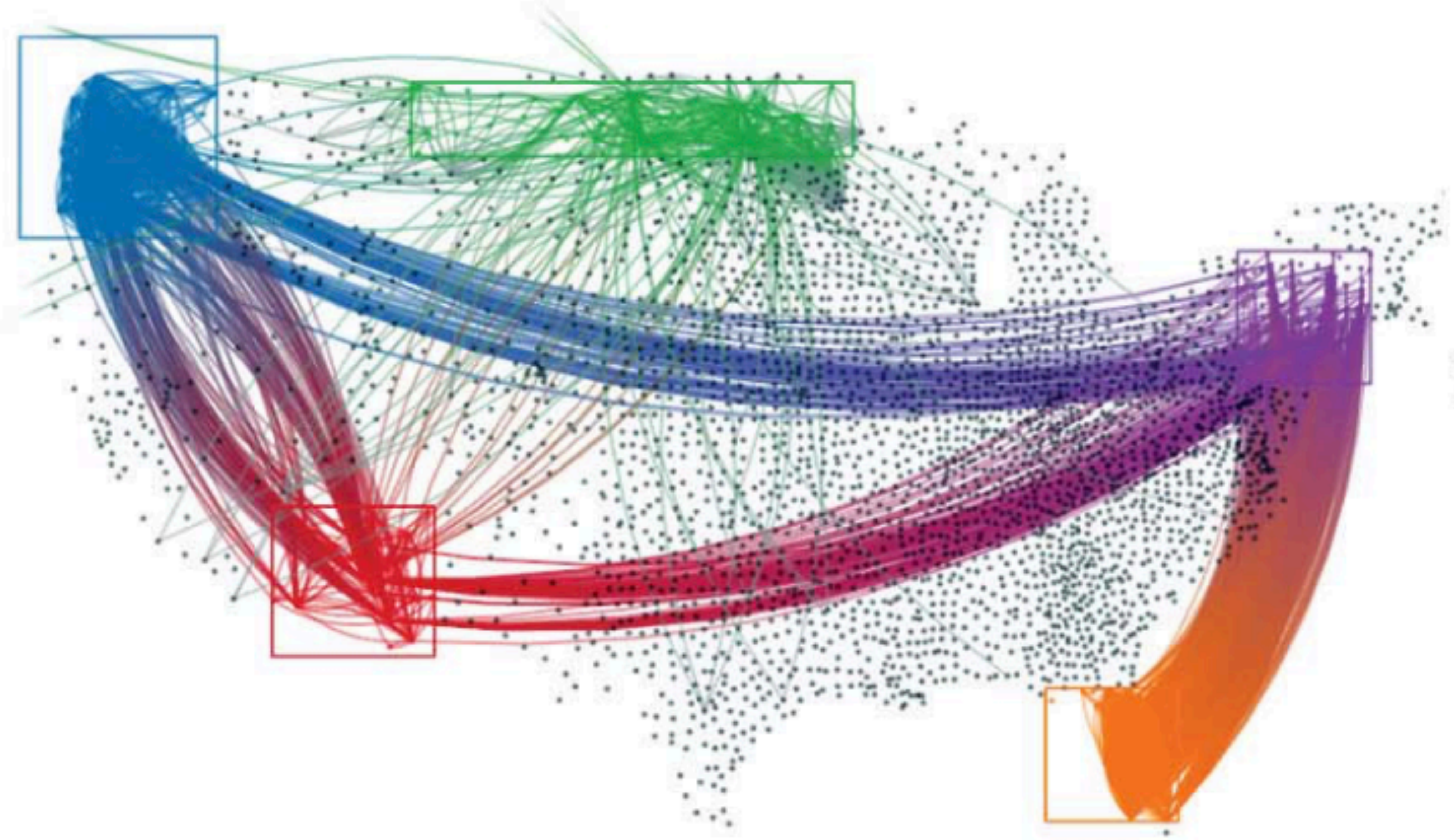


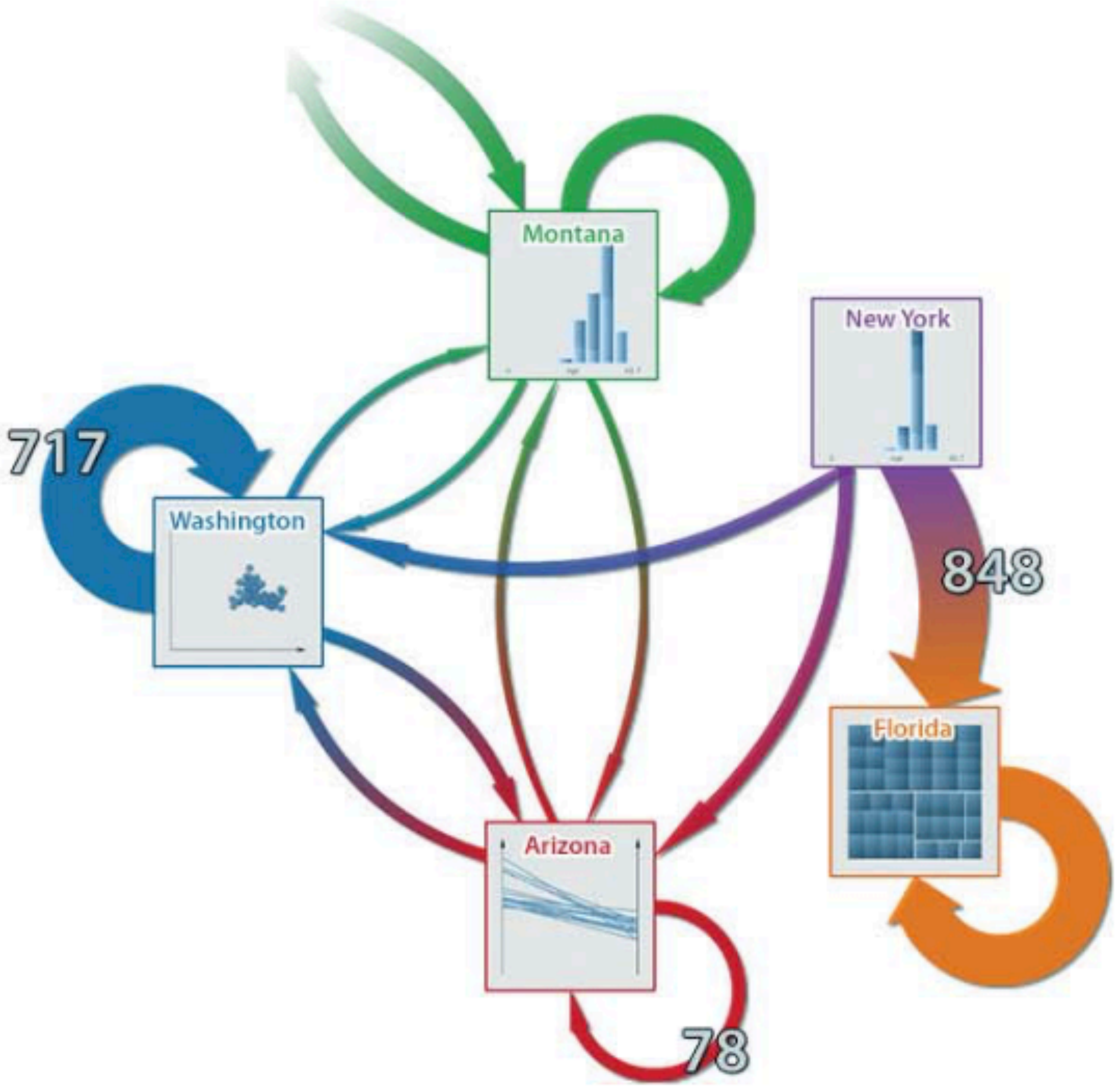
[McDonnel2009]



[Gehlenborg2010]

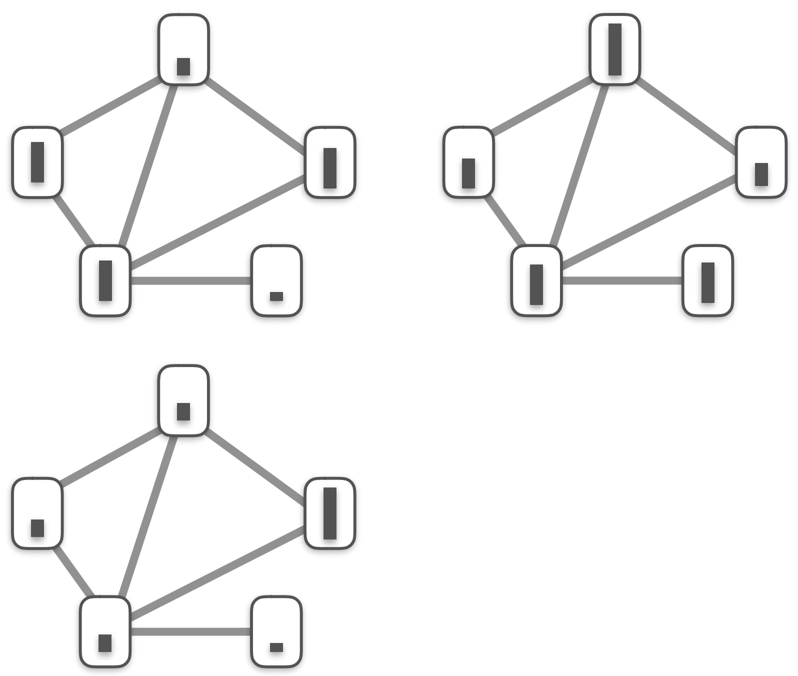
On-Node Encoding for Aggregates





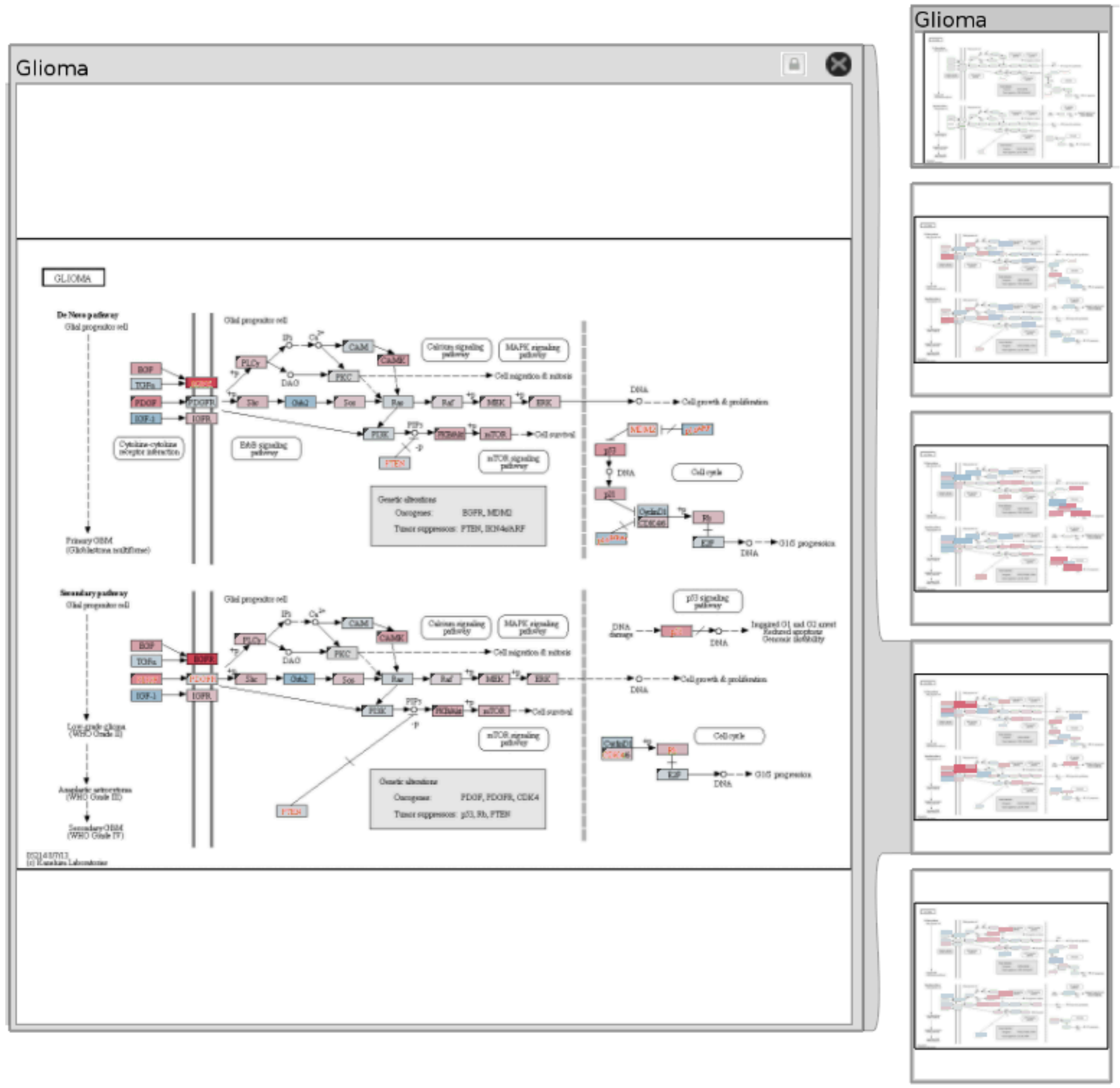
[van den Elzen and van Wijk, 2014]

Small Multiples

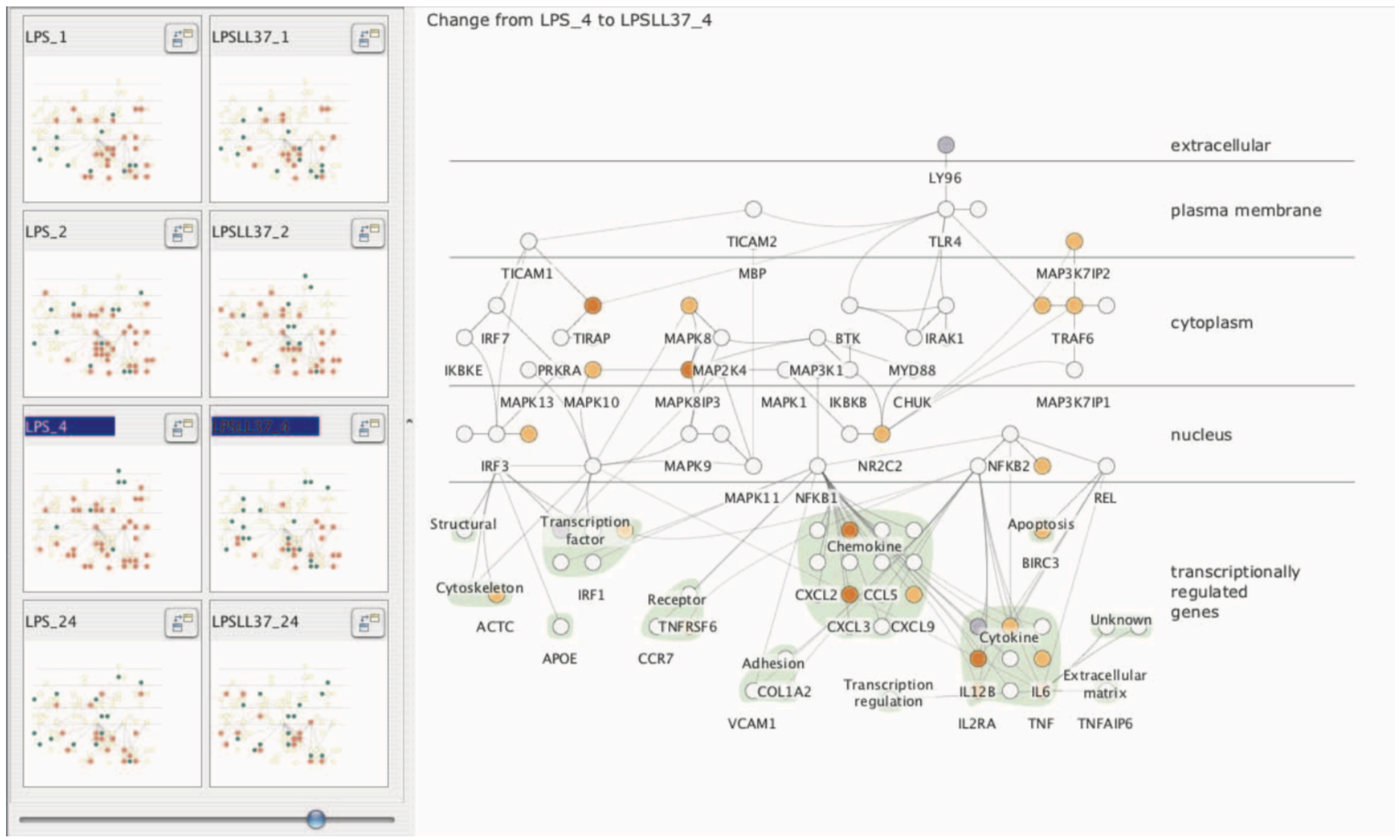


On-node encoding with small multiples

Graphs tend to be small, combine with focus graph



[Lex et al., StratomeX, 2012]

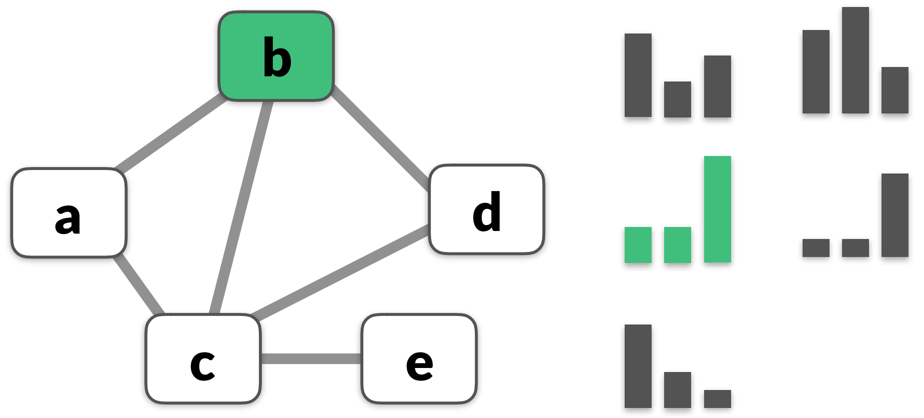


[Barsky et al., Cerebral, 2008]

https://www.sporcle.com/games/Cutthroat/spot-the-difference-i

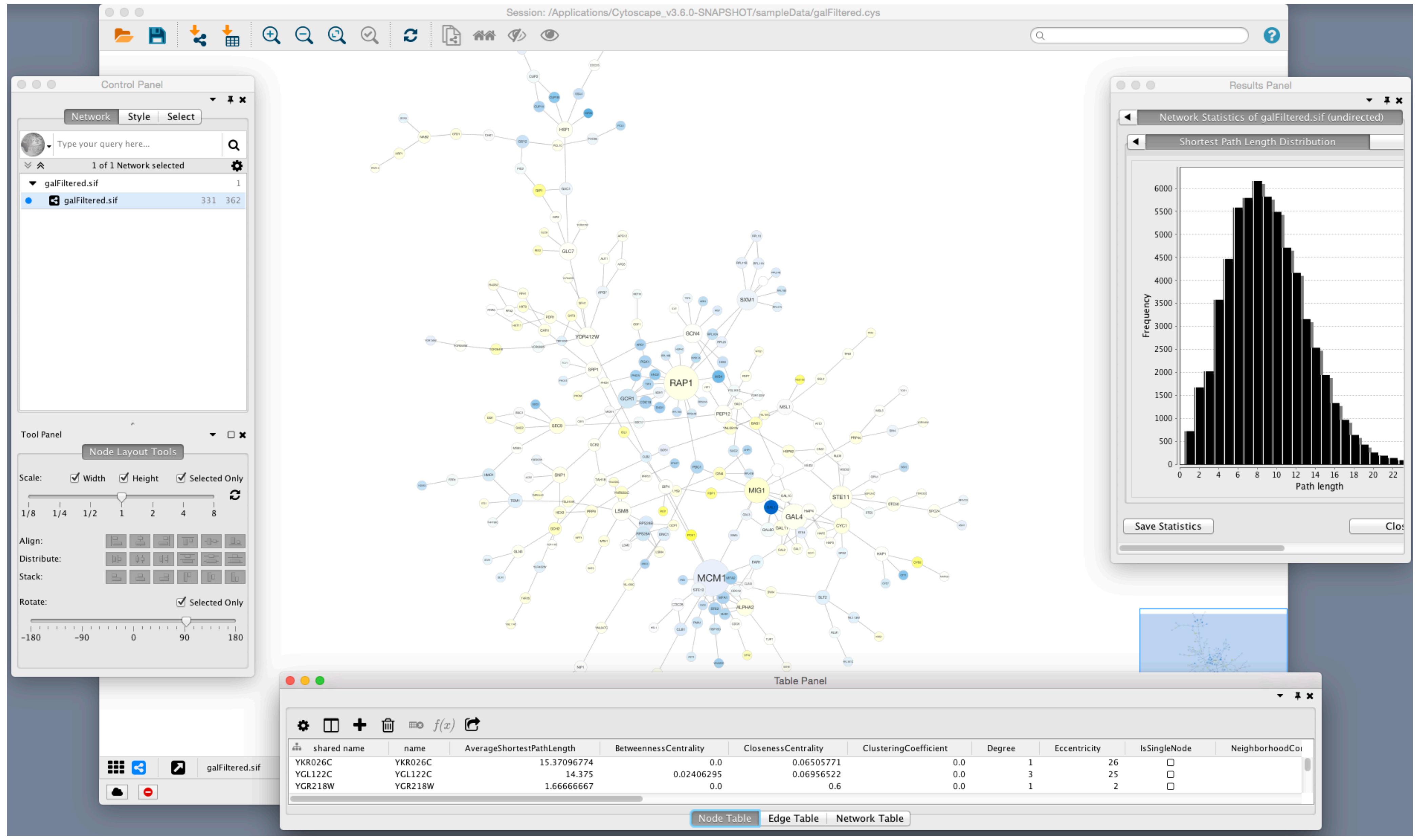
https://www.sporcle.com/games/Cutthroat/spot-the-difference-i

Multiple Coordinated Views

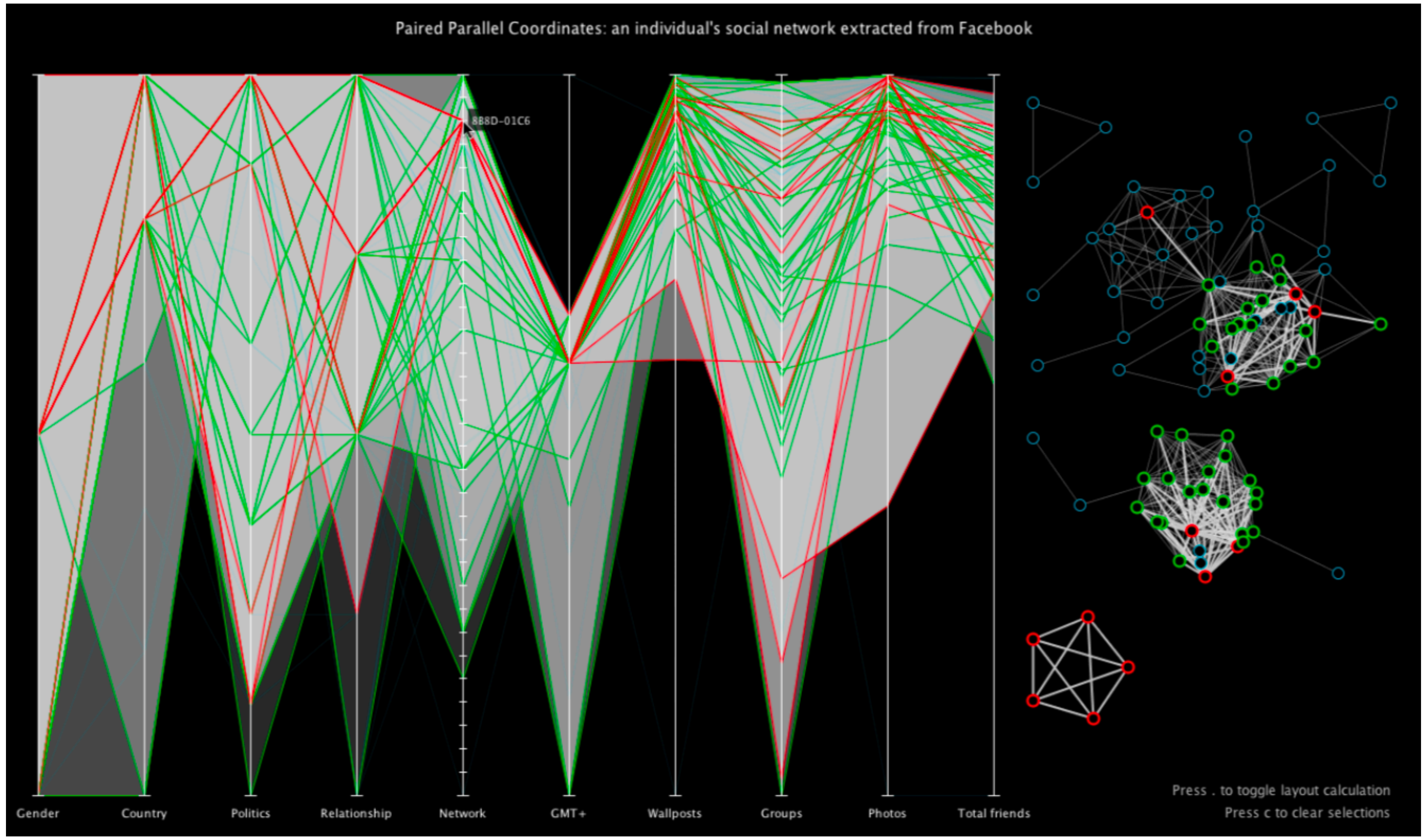


Can optimize for topology and attributes at the same time

Lacking whith regards to interplay

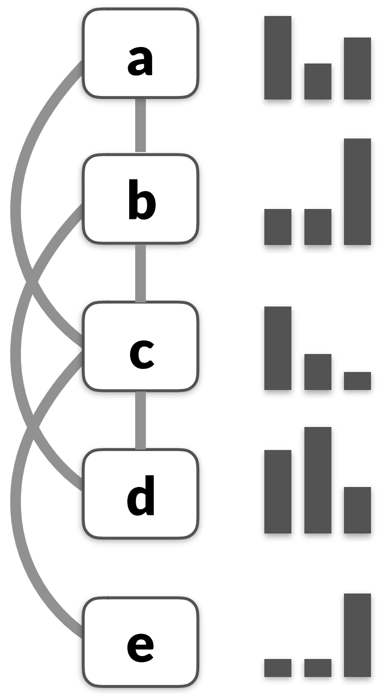


[Shannon et al., Cytoscape, 2003]



[Shannon et al., 2008]

Layout Adaption



Adapt node position in a node-link diagram so that it is well suited for attribute visualization

Layout driven by attributes

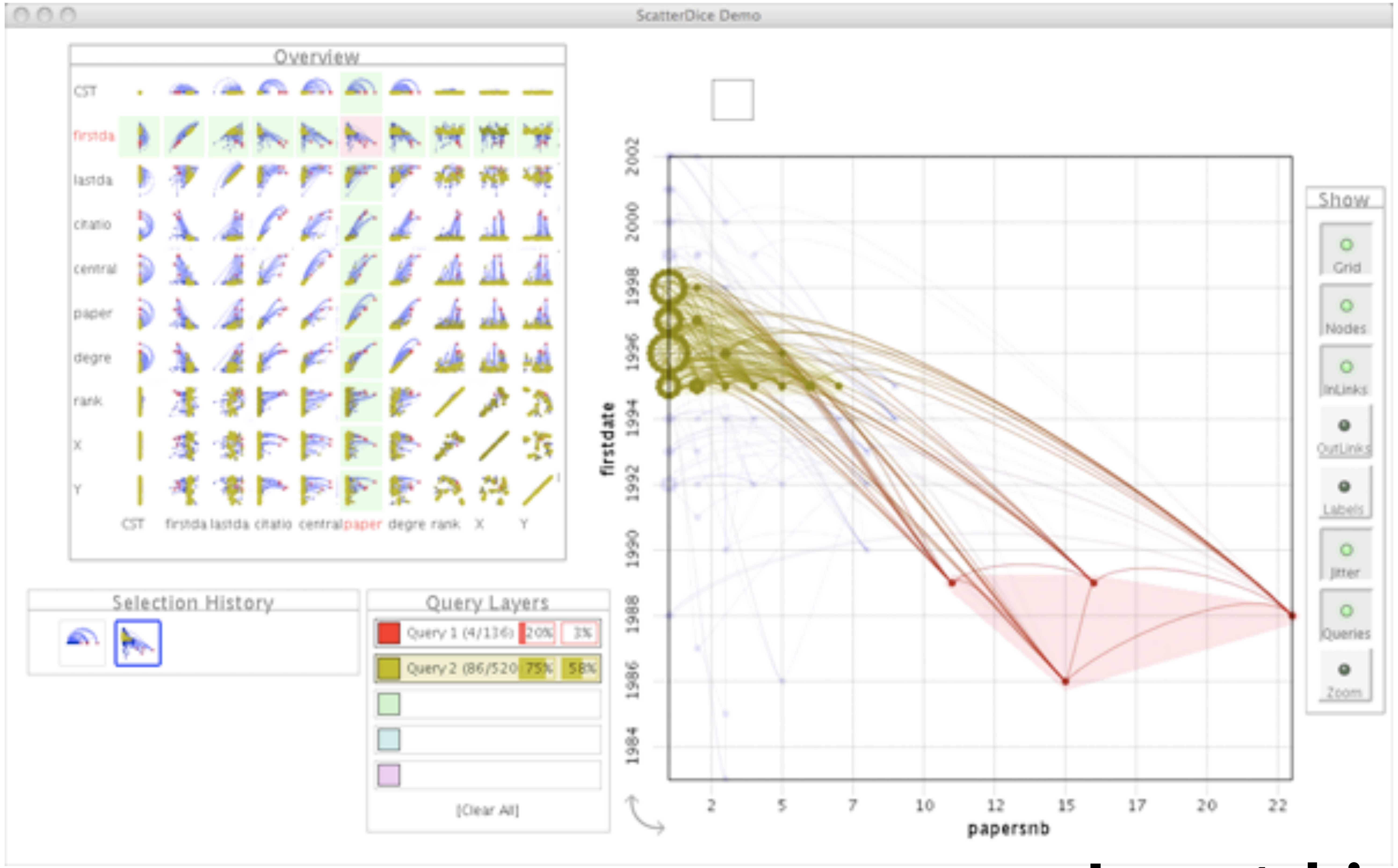
Layout driven by Topology

Fixed Layout

Node position defined by attribute values.

Focus on relationship of limited number of attributes

Topology hard to read



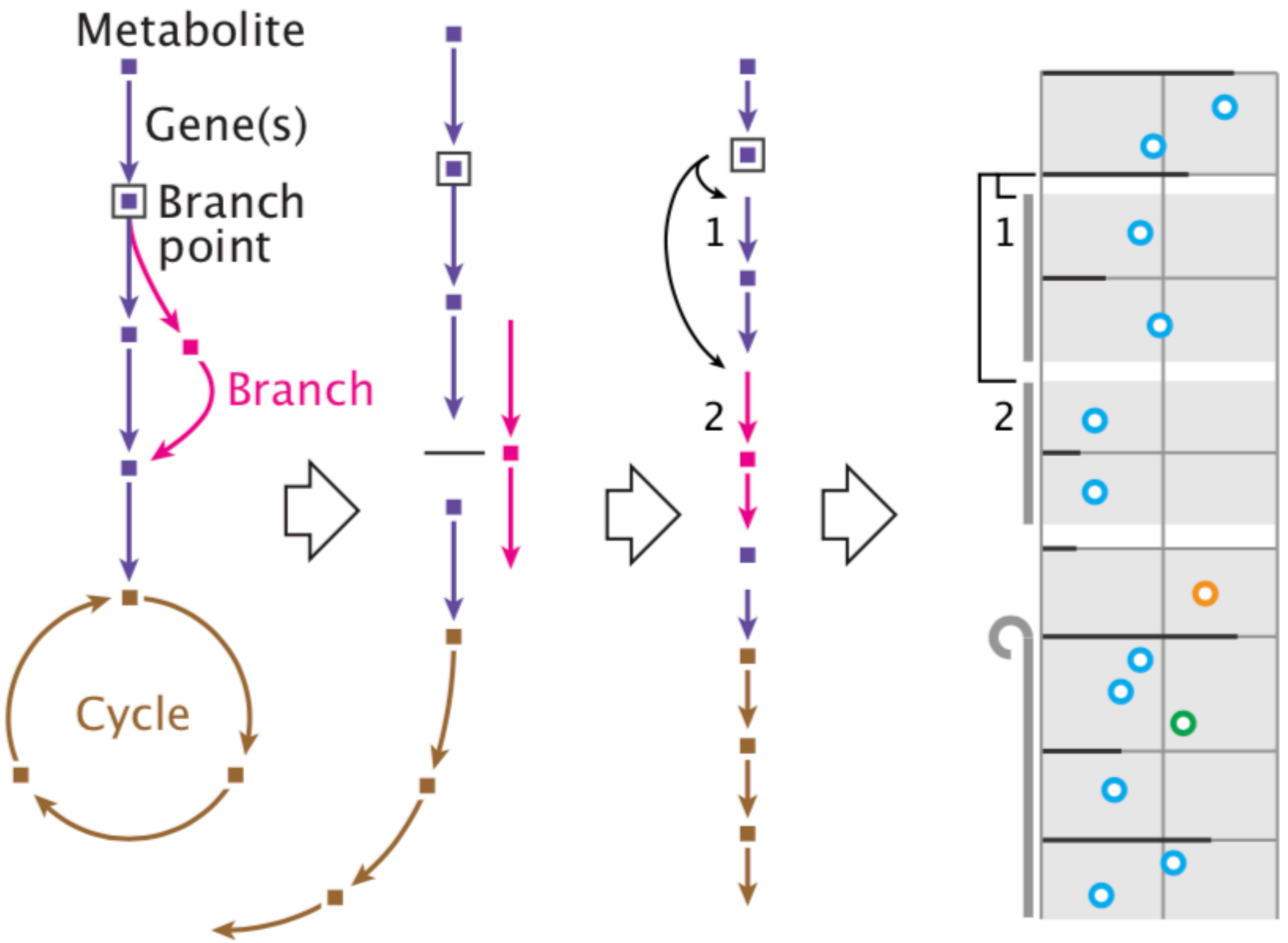
[Bezerianos et al, 2010]

Layout driven by attributes

Layout driven by Topology

Linearization Strategy:
Layout that enables
juxtaposition with attribute
visualizations

Complete Linearization: Pathline



[Meyer et al, Pathline, 2010]

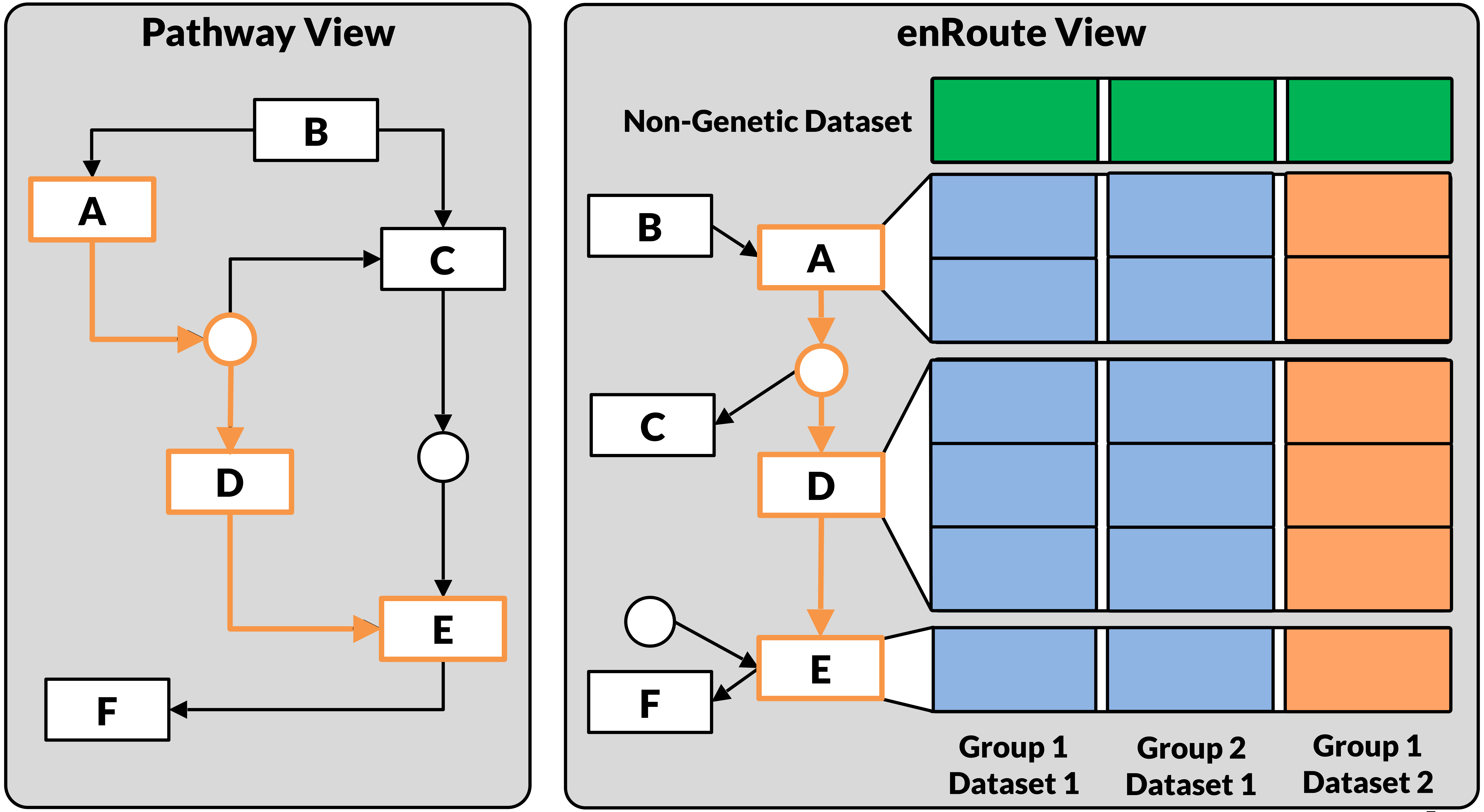
Good solution for smaller graphs

Hard to keep track of topology for complex graphs

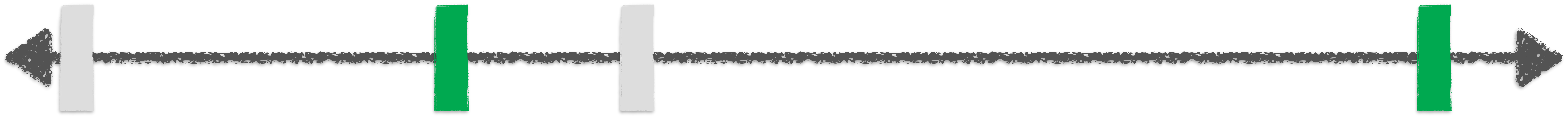
Layout driven by attributes

Layout driven by Topology

Selective Linearization: enRoute

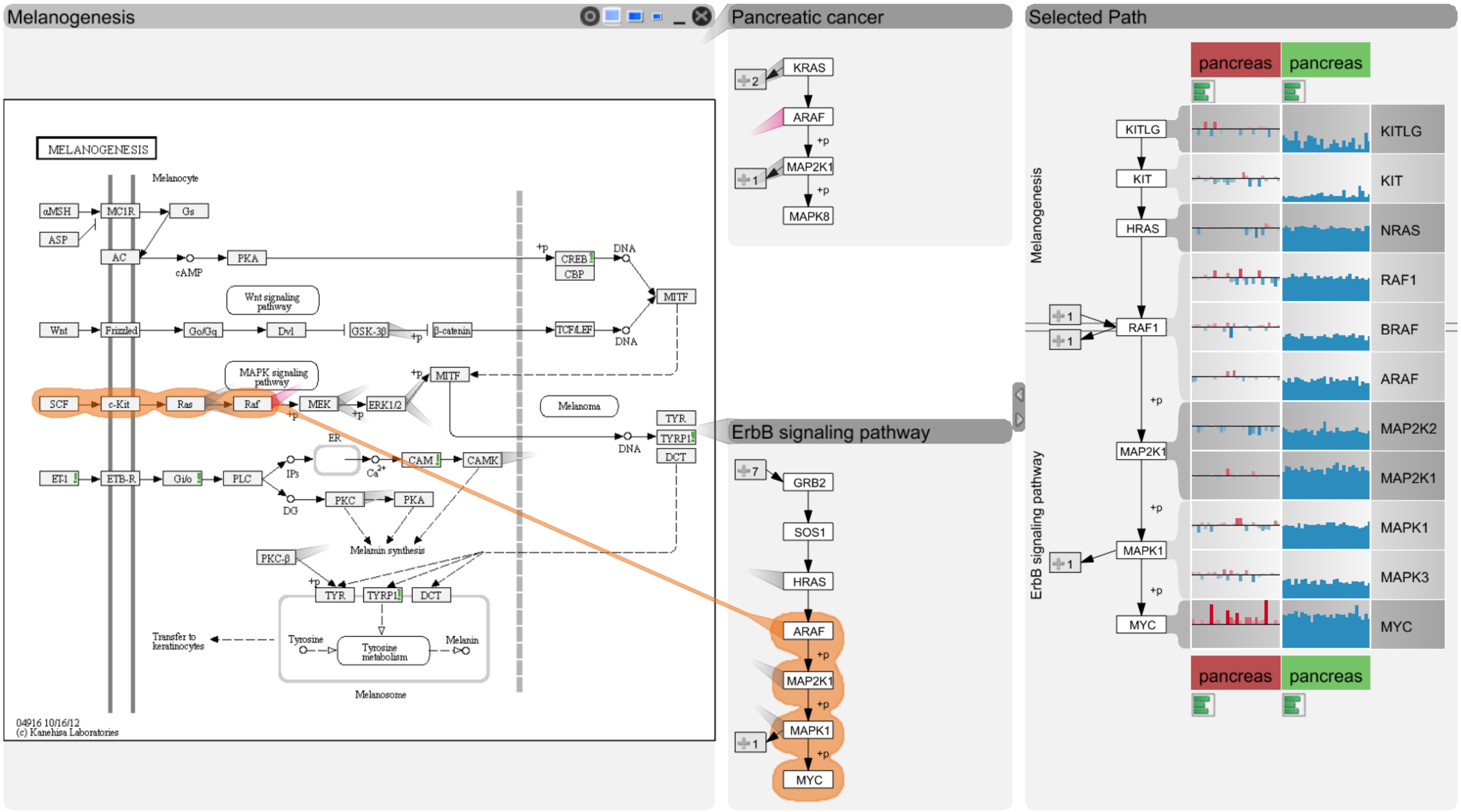


Layout driven by attributes



Layout driven by Topology

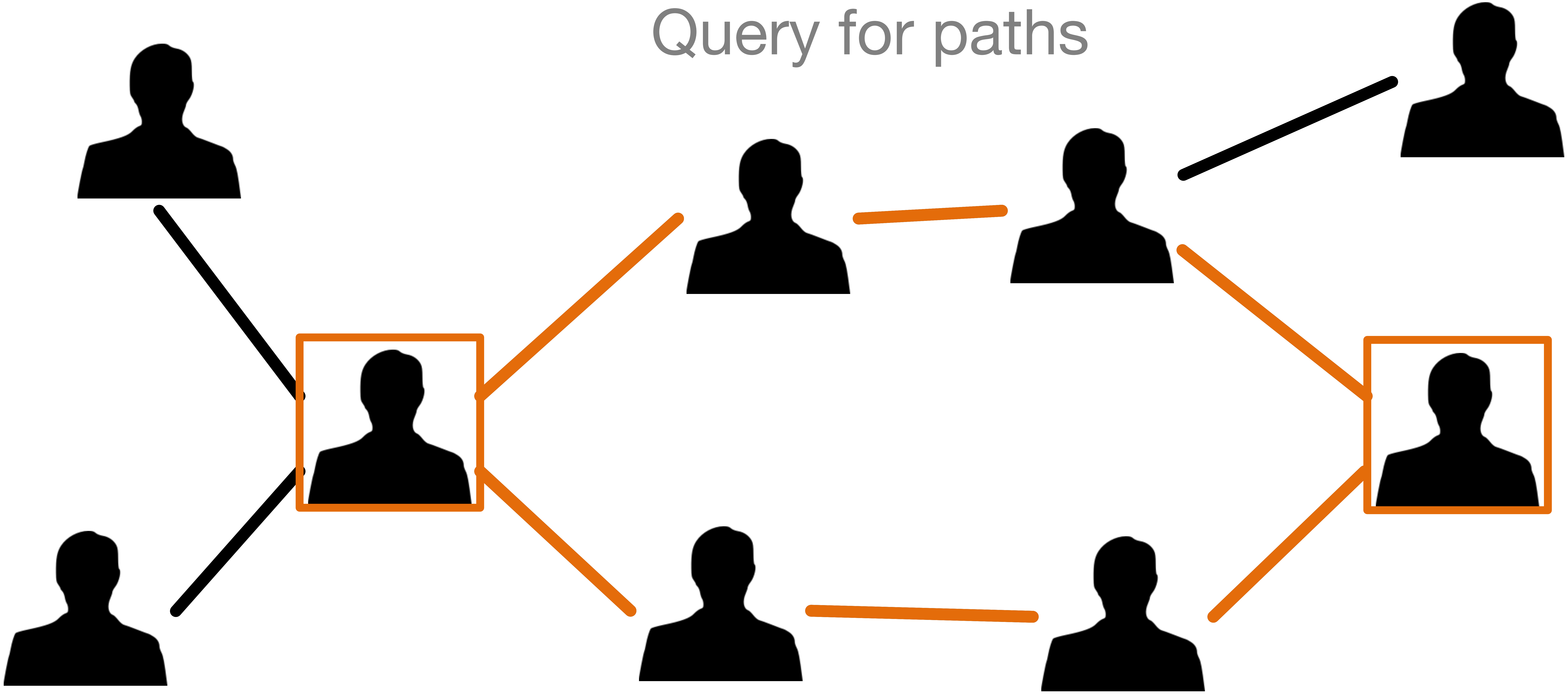
enRoute



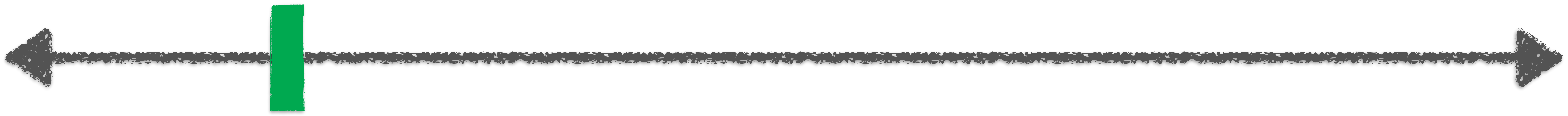
Melanogenesis
Pancreatic cancer
Selected Path
MELANOGENESIS
Melanocyte
αMSH
MC1R
Gs
ASP
AC
cAMP
PKA
CREB
CBP
DNA
MITF
Wnt signaling pathway
Wnt
Frizzled
Go/Gq
Dvl
GSK-3β
β-catenin
TCF/LEF
MAPK signaling pathway
SCF
c-Kit
Ras
Raf
MEK
ERK1/2
Melanoma
TYR
TYRP1
DCT
ER
ET-1
ETB-R
Gi/o
PLC
IP3
Ca2+
CAM
CAMK
DG
PKC
PKC-β
Melanin synthesis
Transfer to keratinocytes
Tyrosine
Tyrosine metabolism
Melanin
Melanosome
04916 10/16/12
(c) Kanehisa Laboratories
KRAS
ARAF
MAP2K1
MAPK8
ErbB signaling pathway
GRB2
SOS1
HRAS
MAPK1
MYC
pancreas
pancreas
Melanogenesis
ErbB signaling pathway
KITLG
KIT
NRAS
RAF1
BRAF
MAP2K2
MAPK3

Selective Linearization: Pathfinder

Query for paths

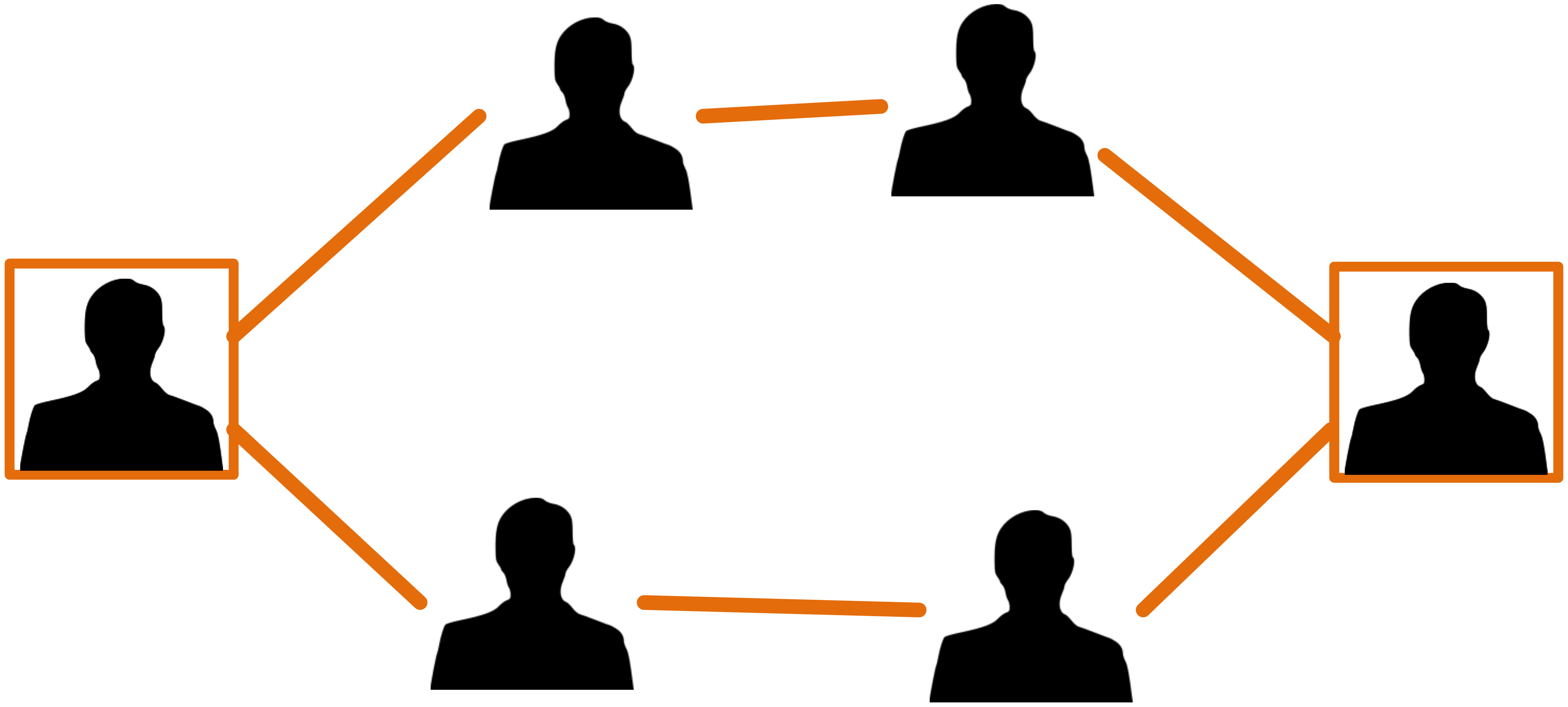


Layout driven by attributes



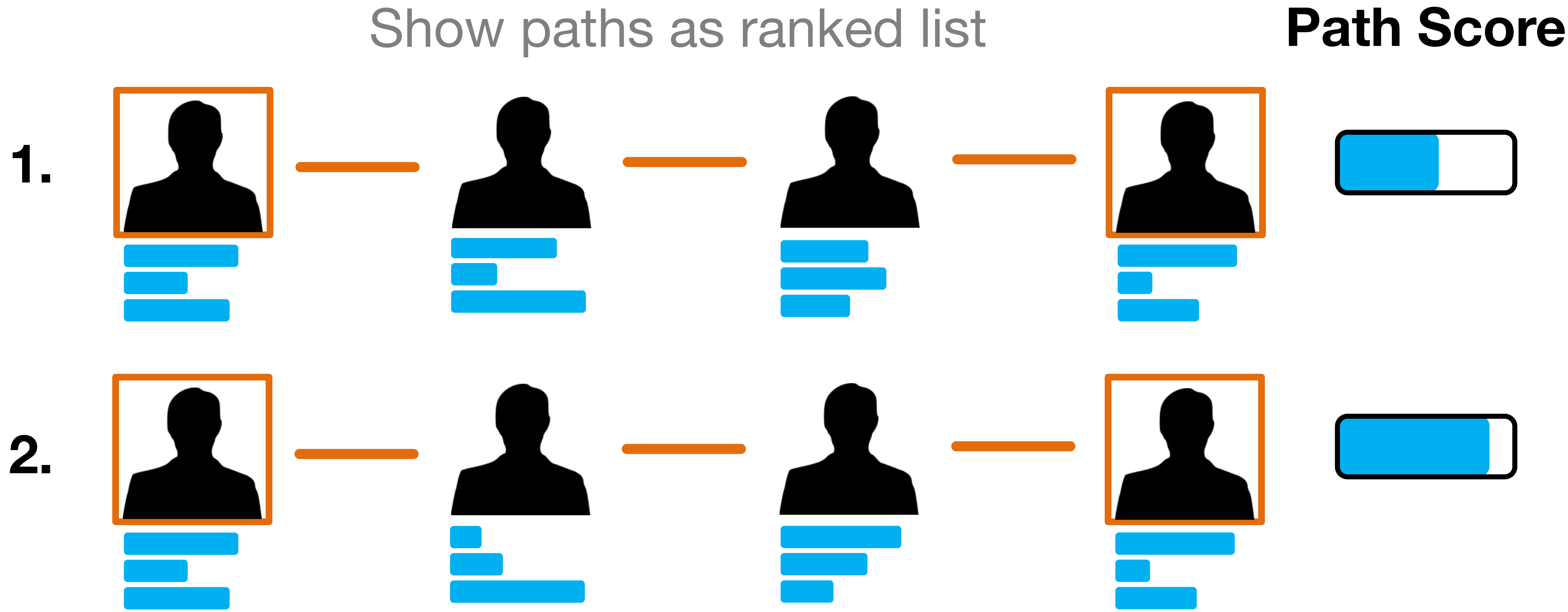
Layout driven by Topology

Selective Linearization: Pathfinder



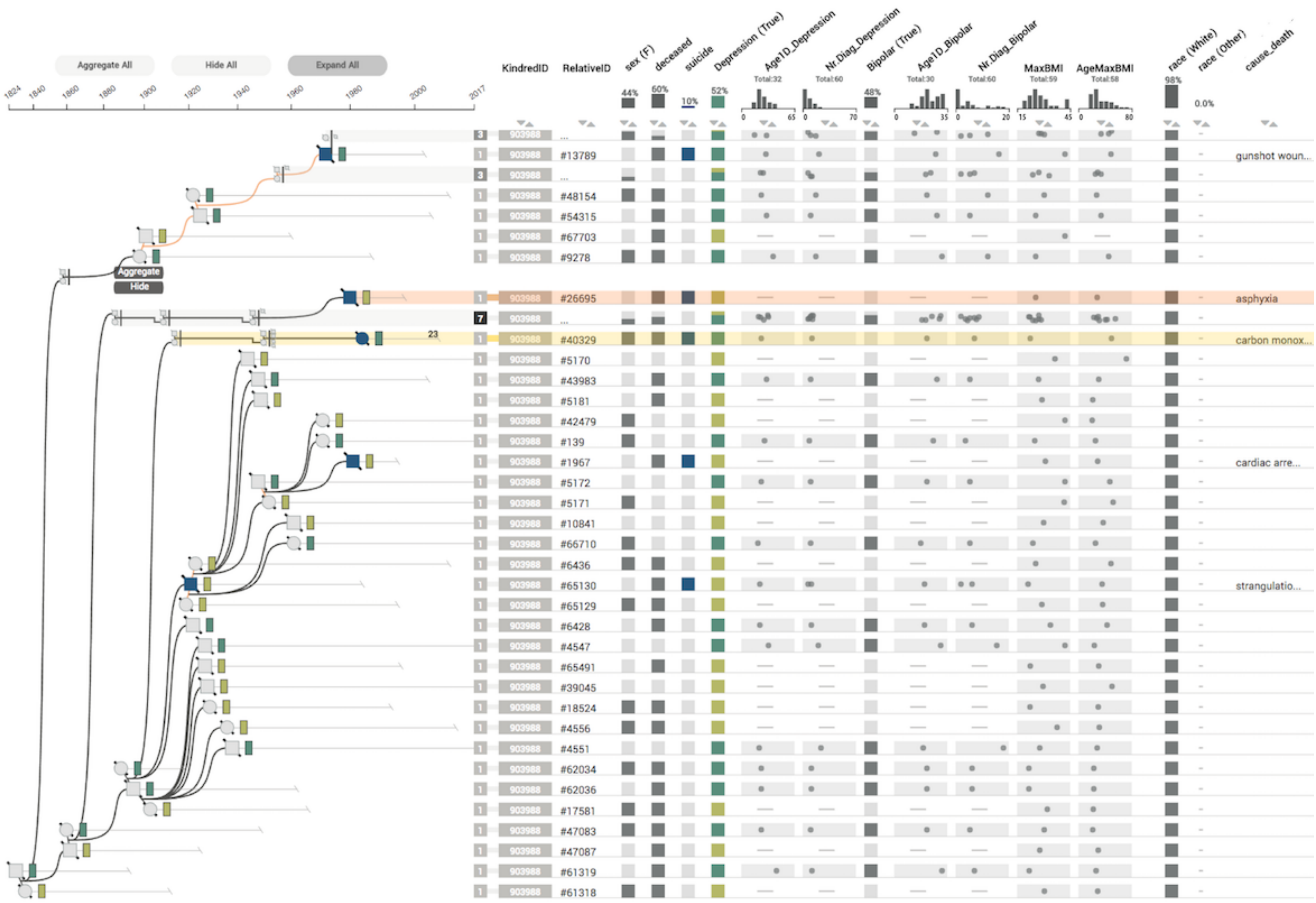
Selective Linearization: Pathfinder

Show paths as ranked list

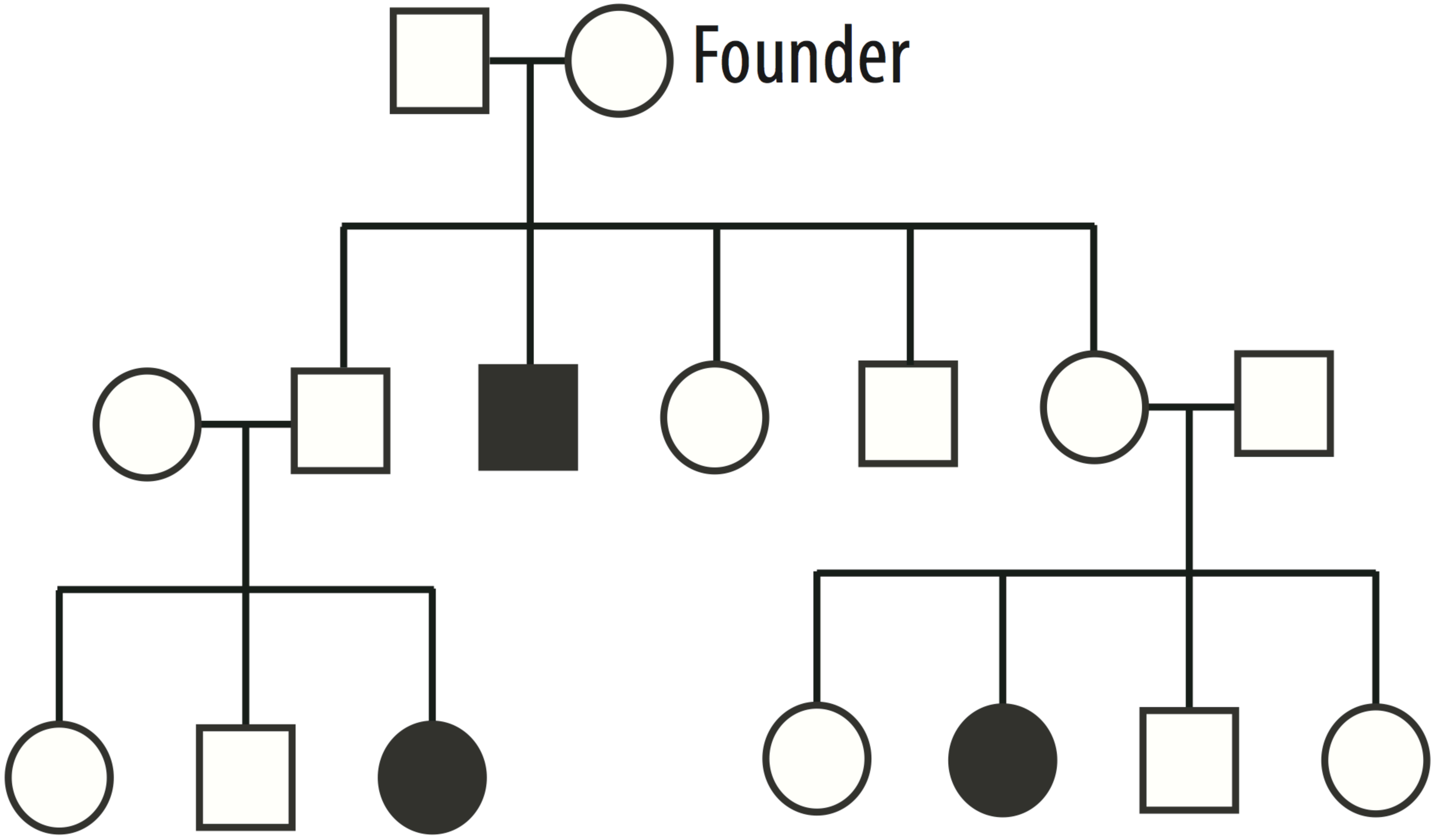


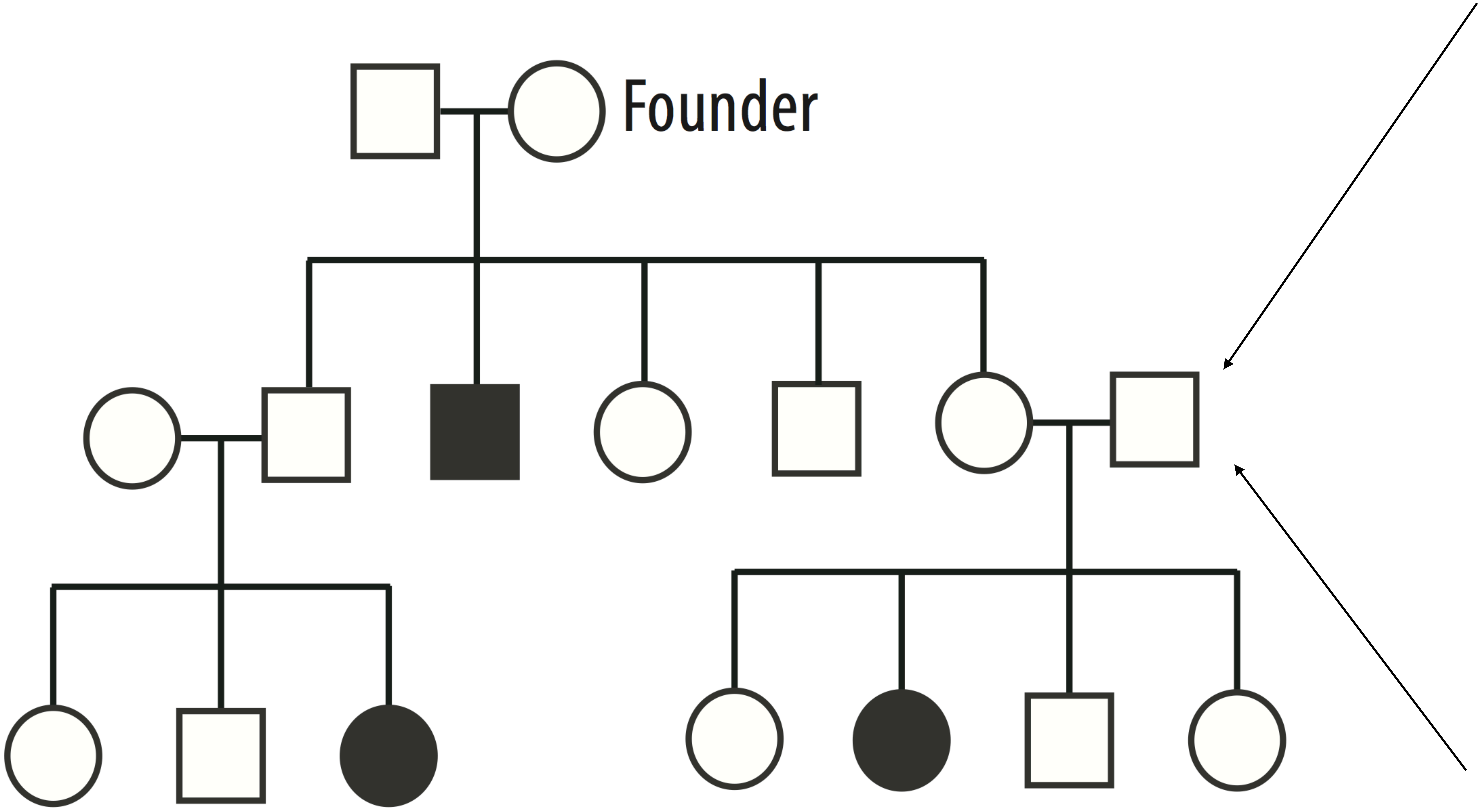

Pathfinder
Start
End
Advanced Query
Length
Paths
Path Statistics
Path List
Query Interface
Path Topology
Active Page
All
«
»

Linearizing a Tree: Lineage



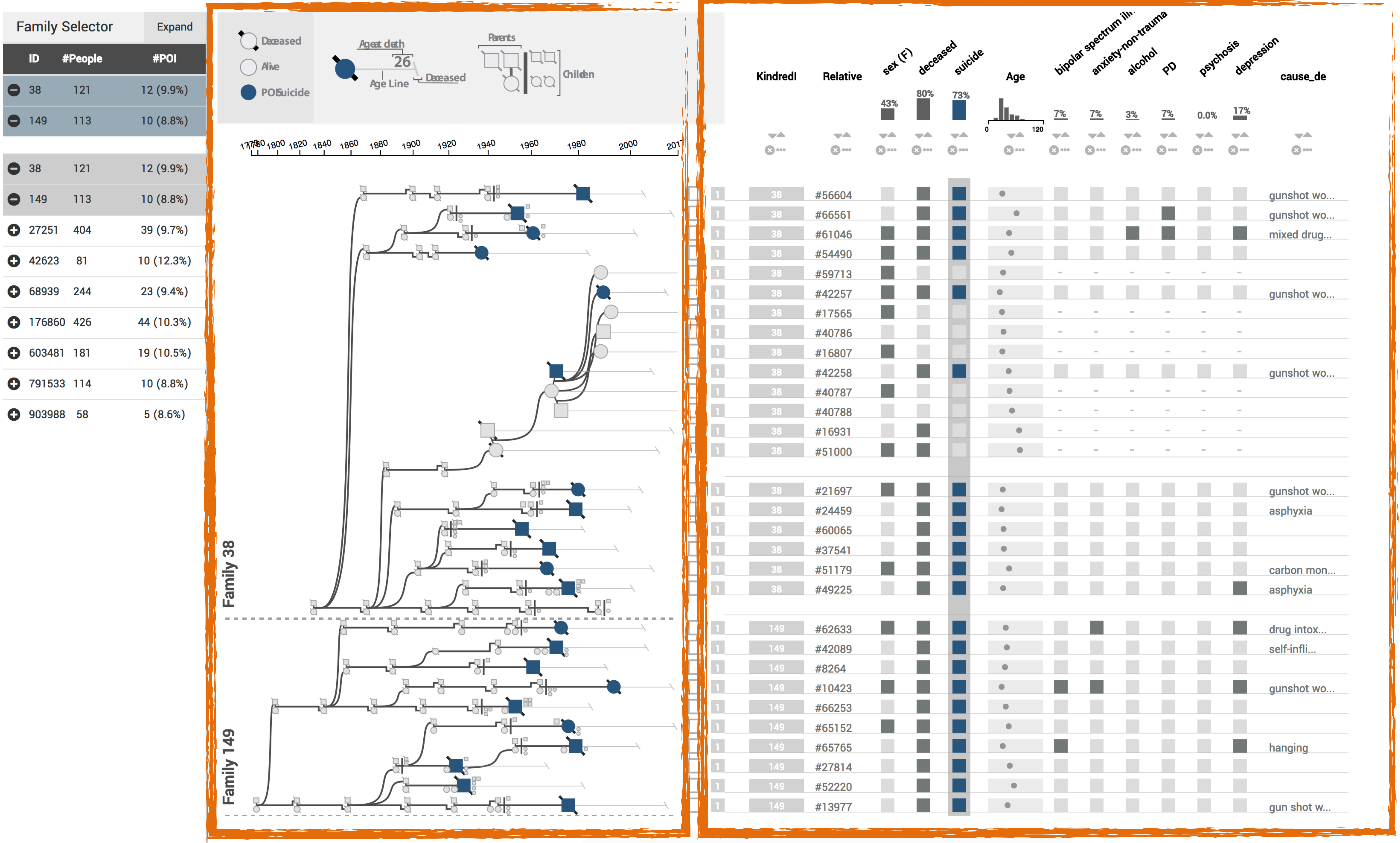
[Nobre et al, Lineage, 2018]


Founder



Age
Sex
Race
Bipolar
 # diagnosis bipolar
Depression
 # diagnosis depression
Asthma
 # diagnosis asthma
Obesity
Schizophrenia
Cause of Death
Weapon Used

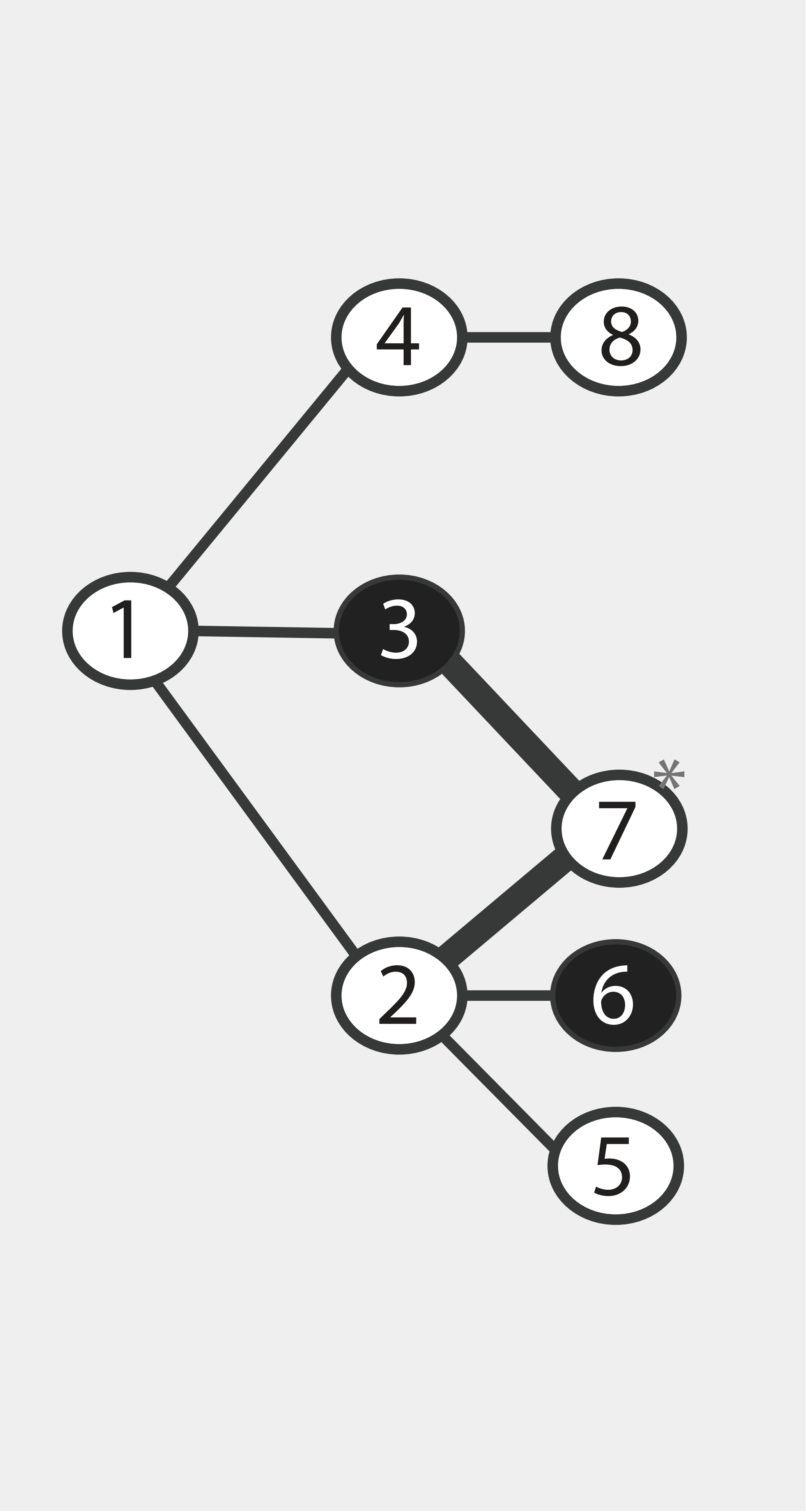
Genealogy with ~400 members rendered with Progeny



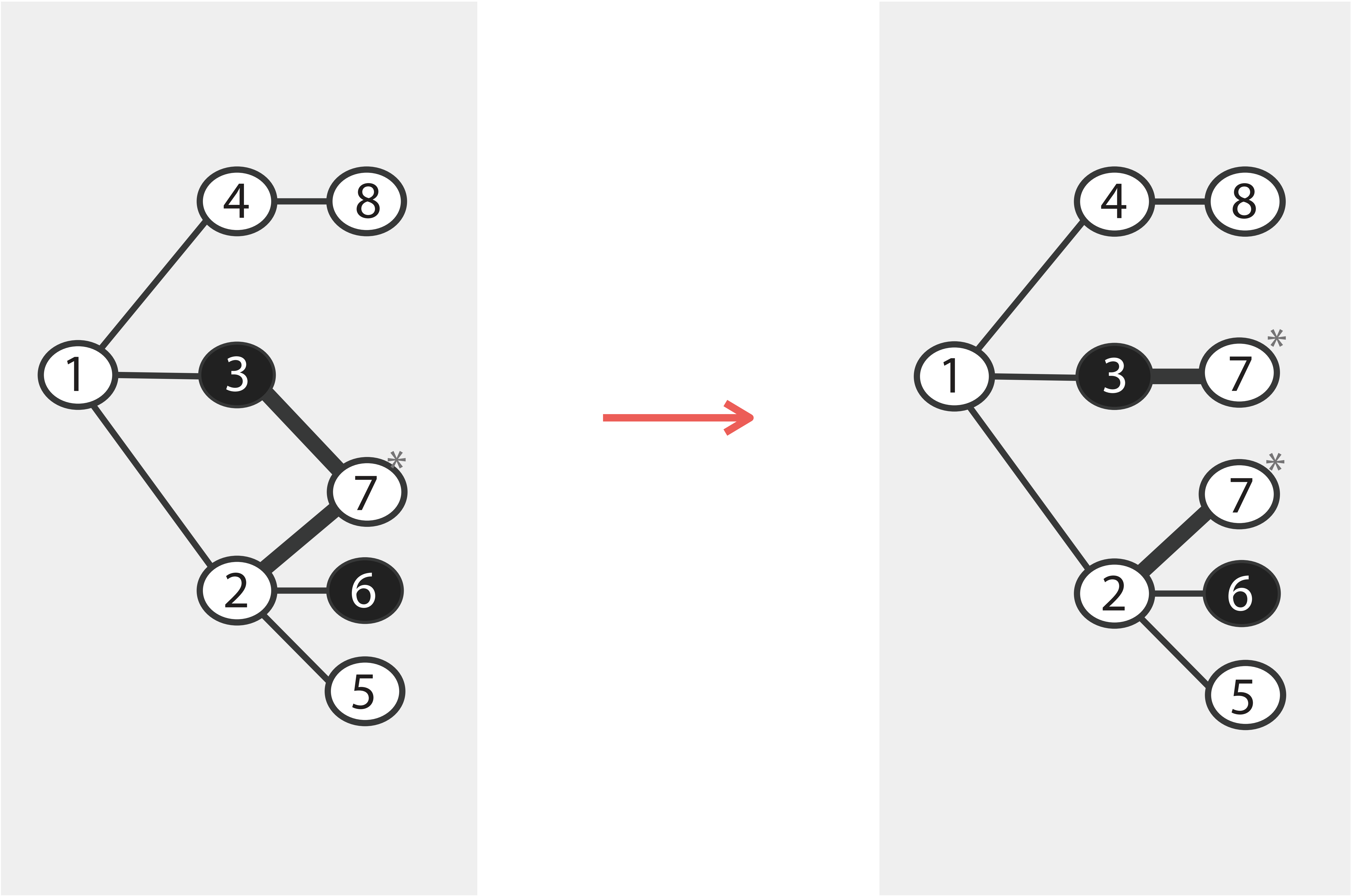
1. De-cycle and linearize graph

2. Plot attributes in table

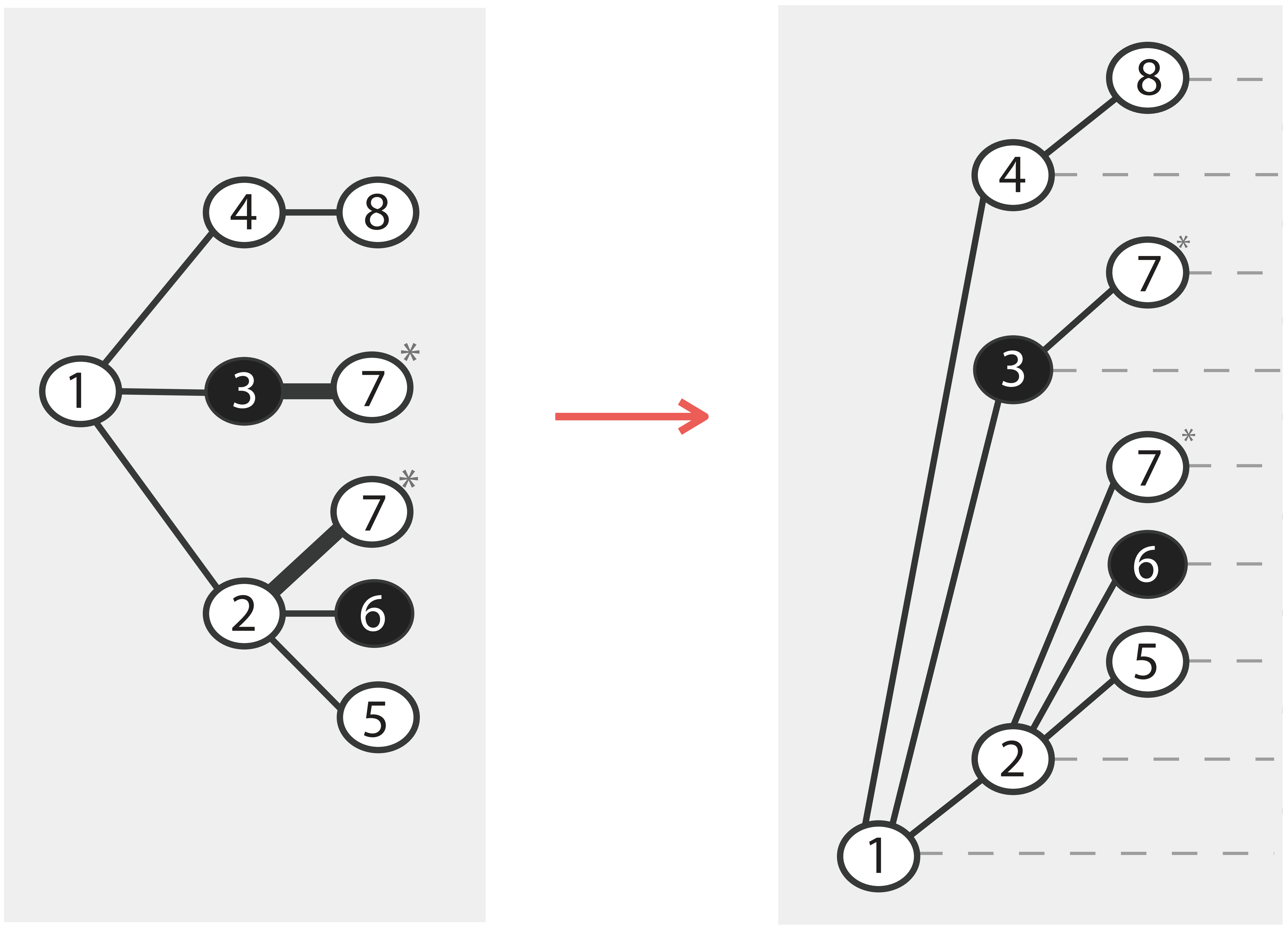
De-Cycling



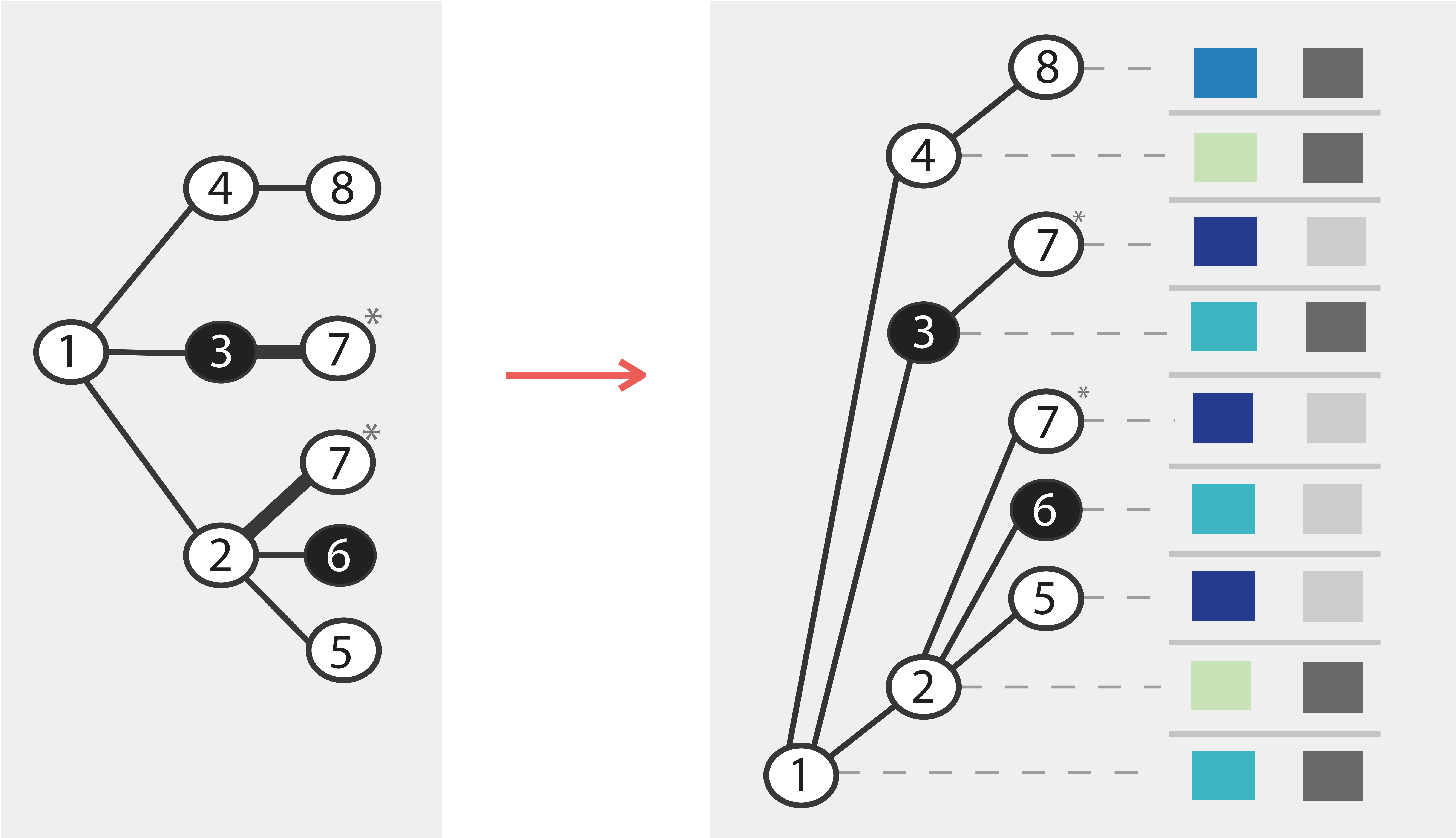
De-Cycling

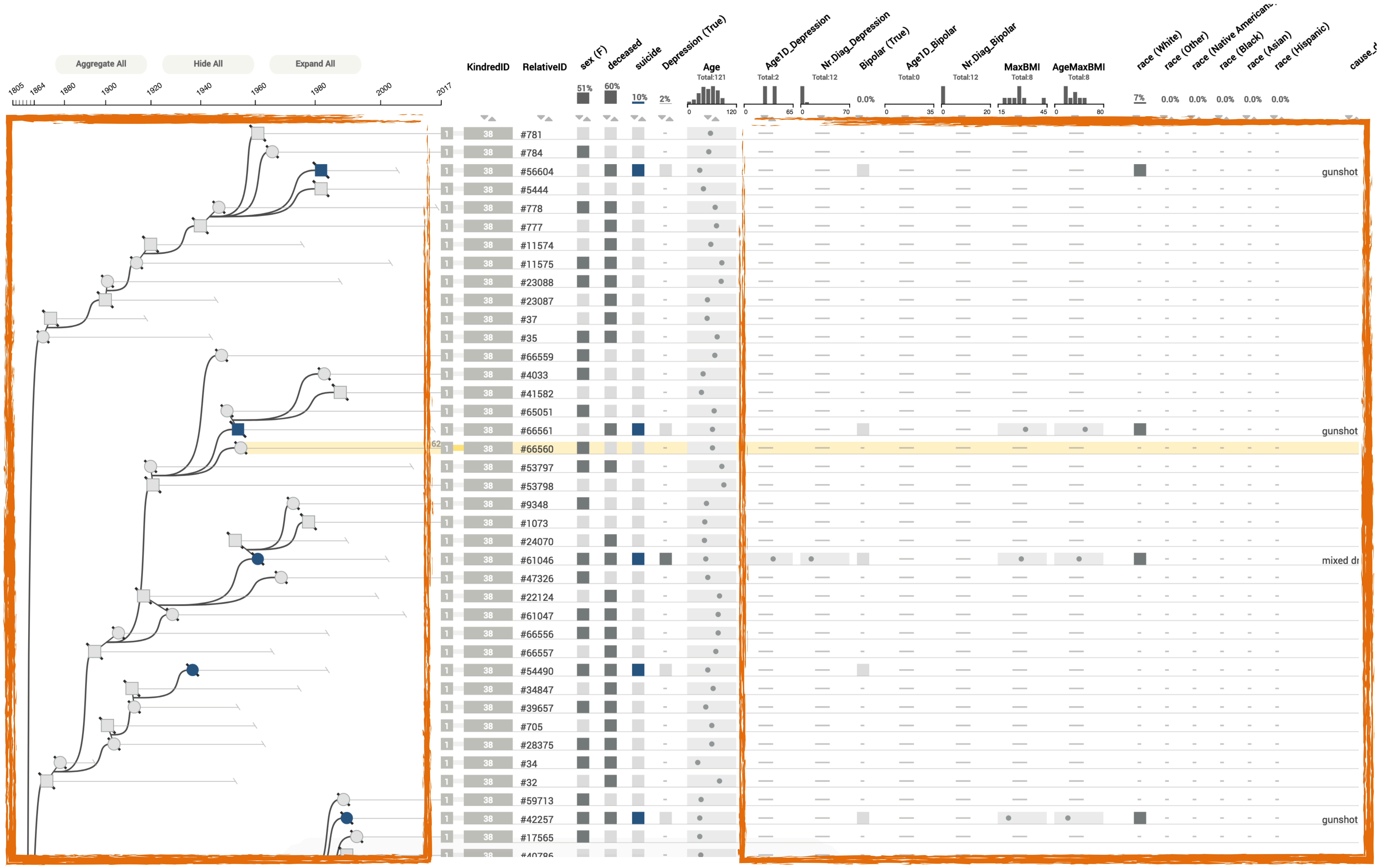


Linearization



Linearization





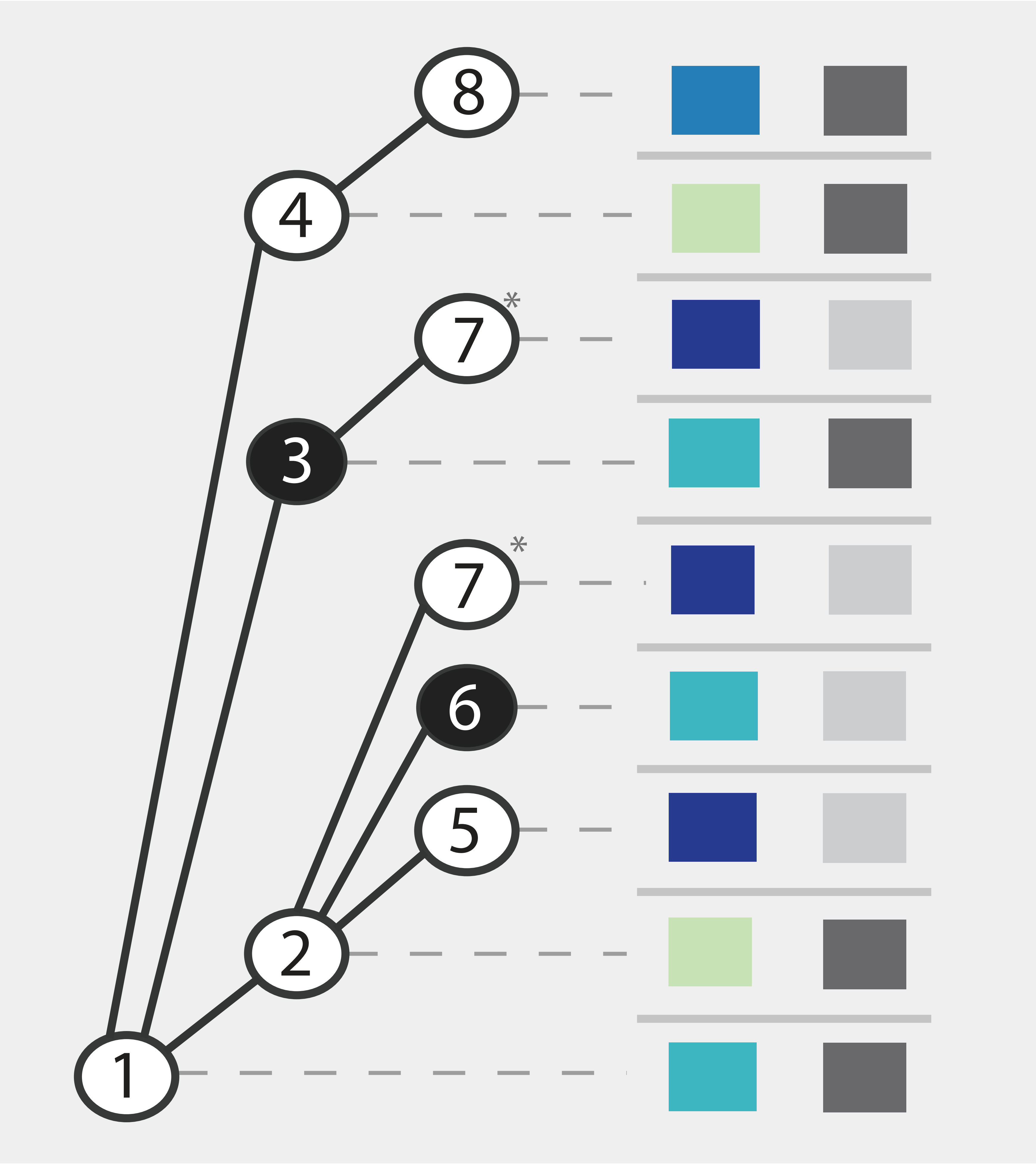
Can’t show many people

Lots of missing data

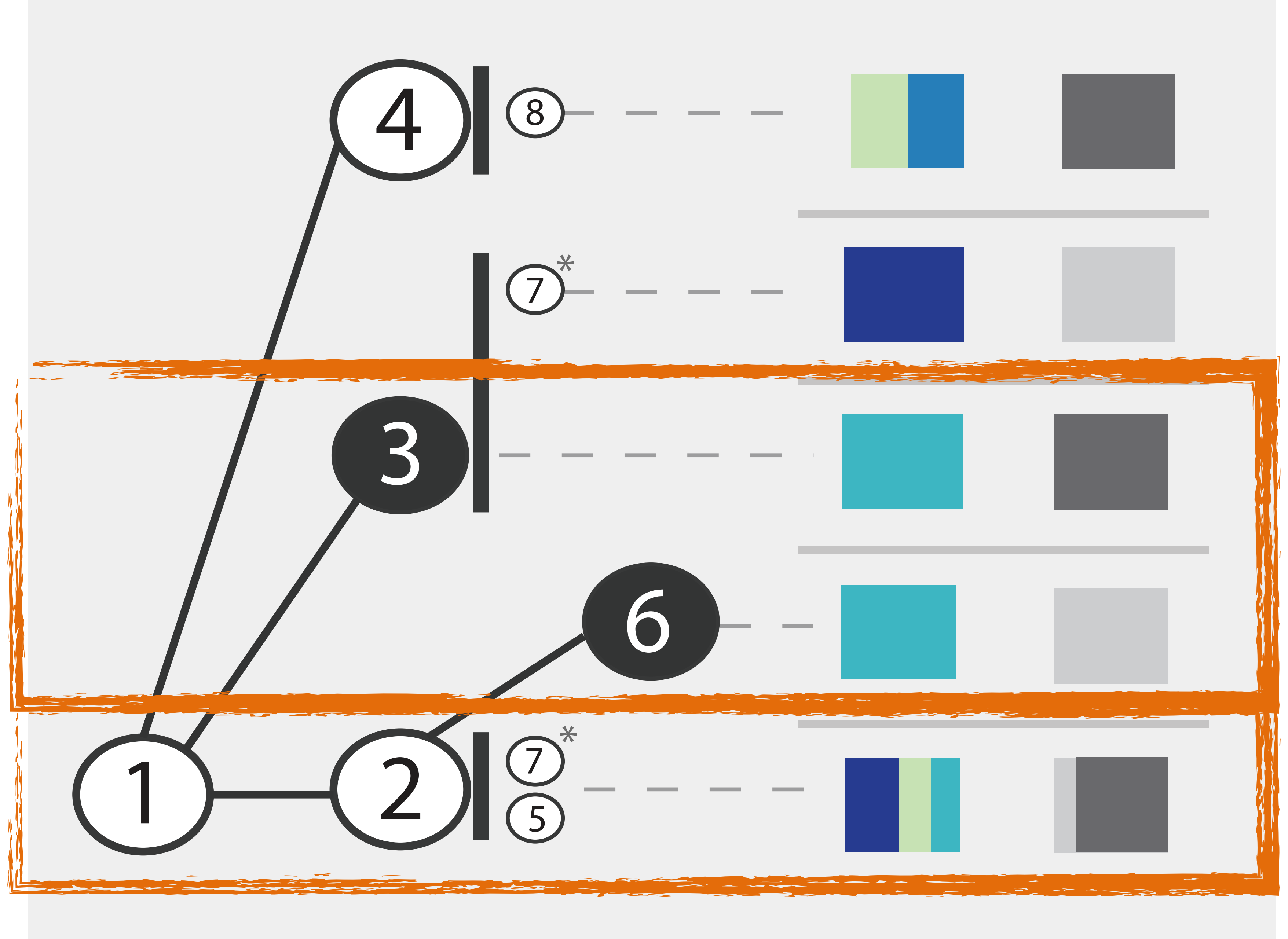
Aggregation

People of Interest

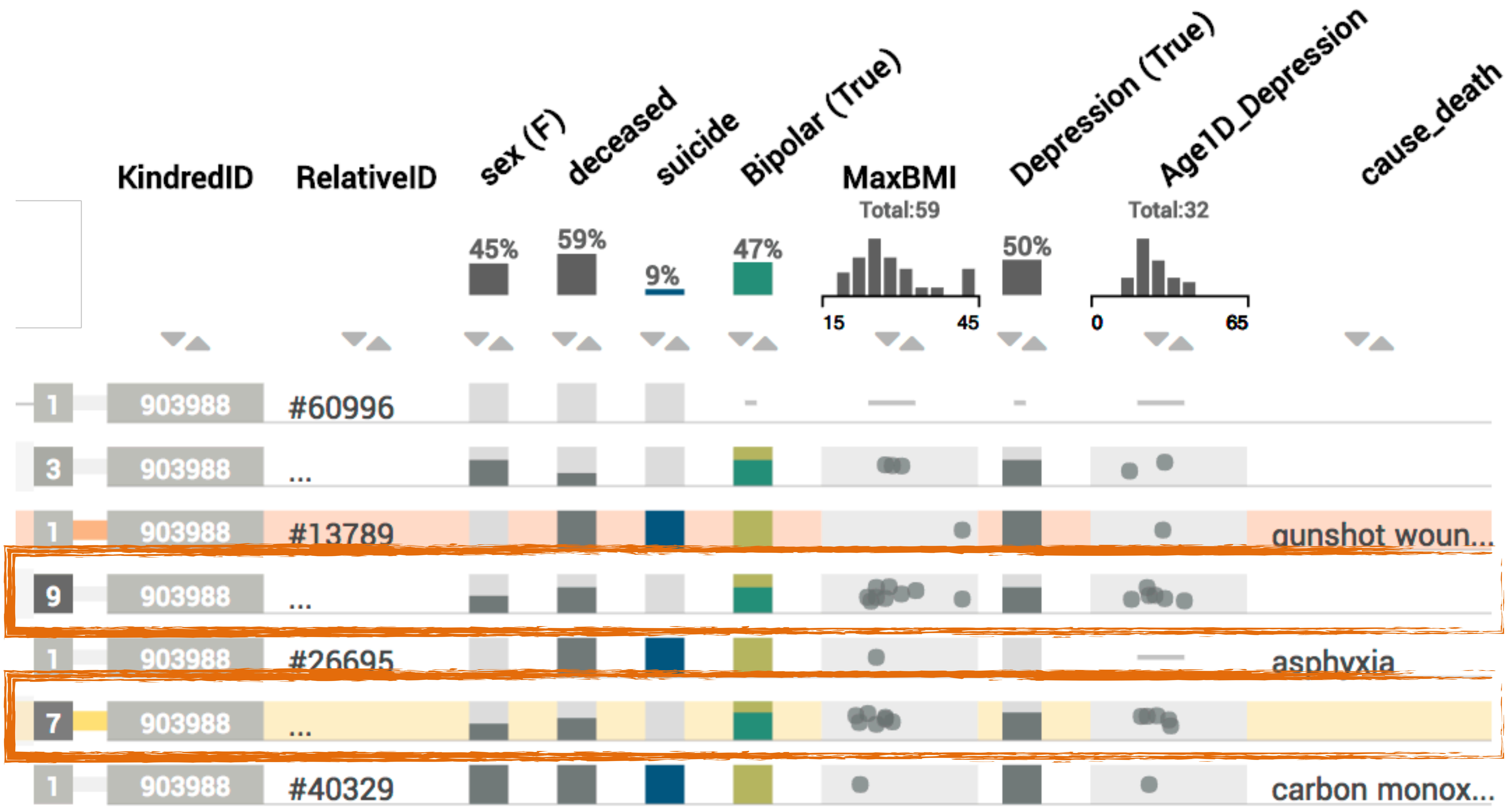
Aggregation



One row for every person of interest

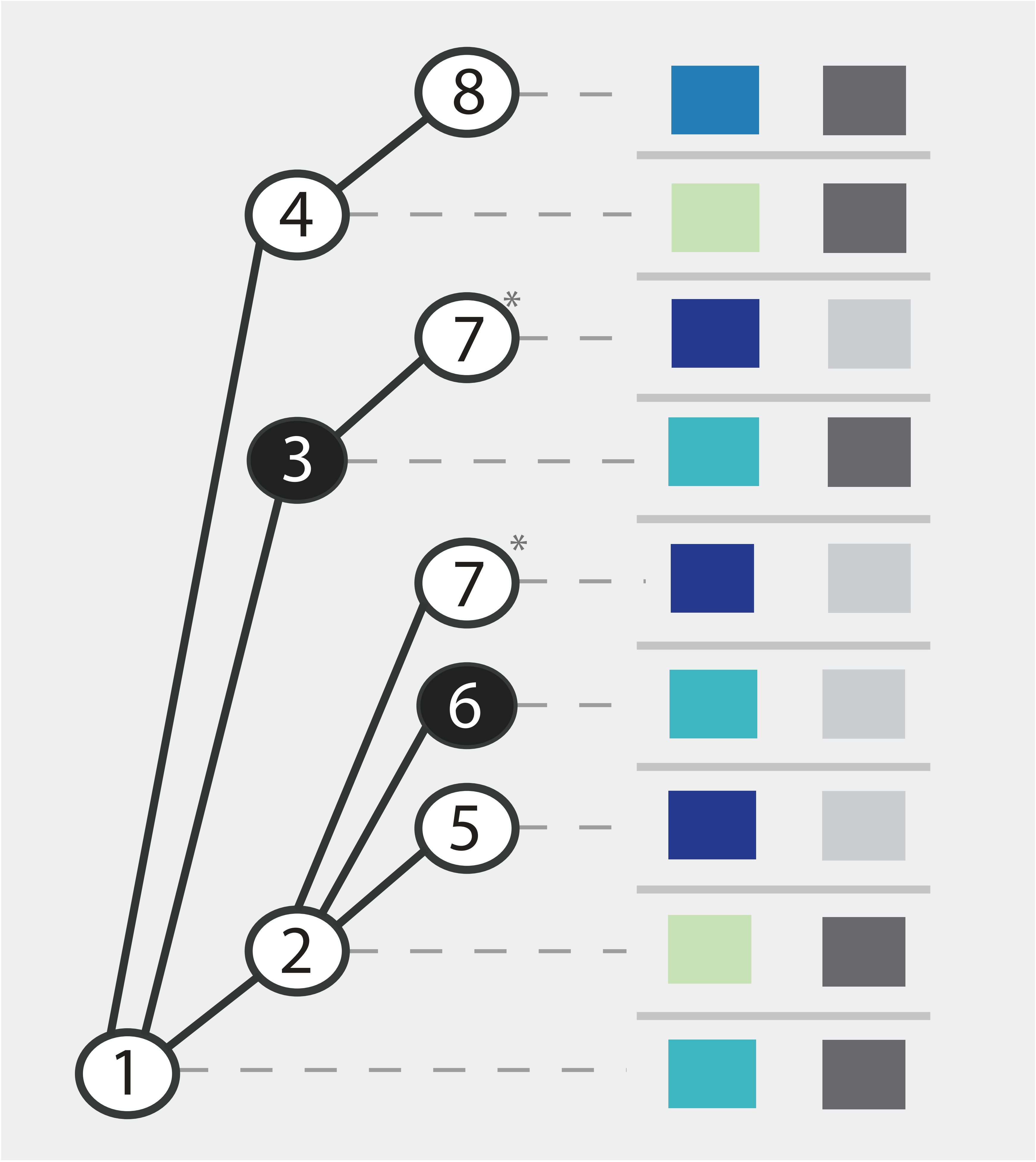


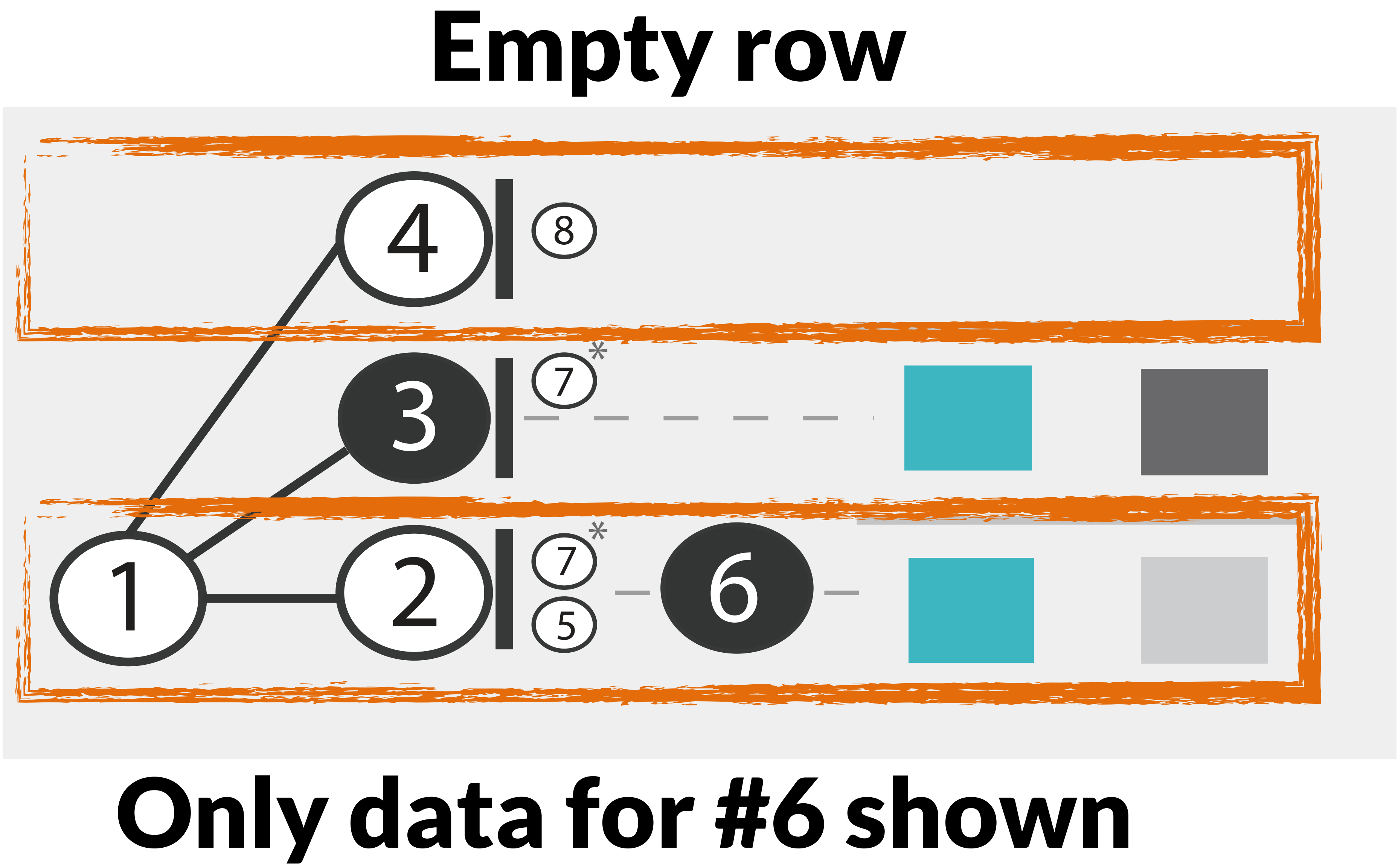
Others have to share a row

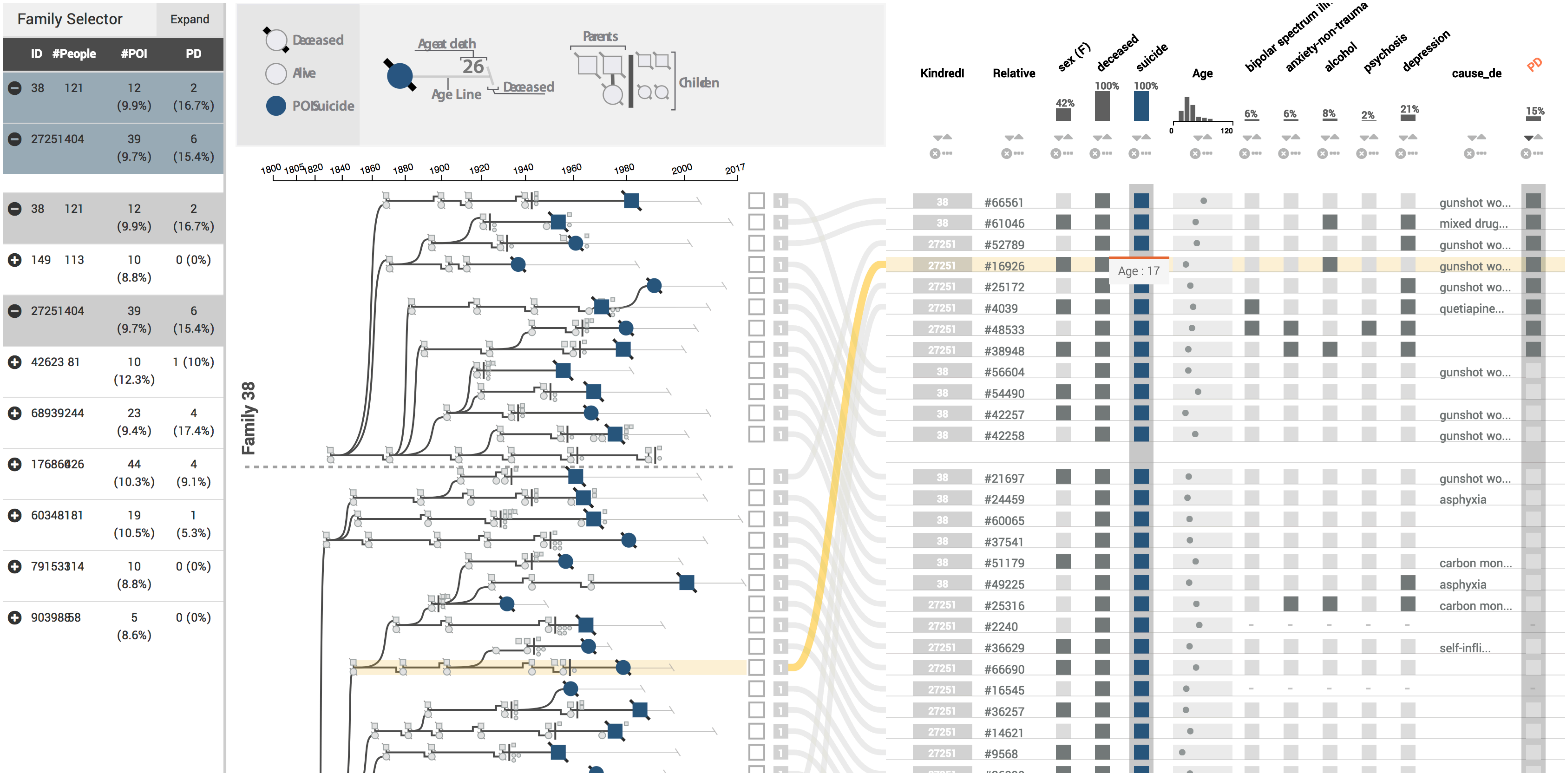


Aggregated Rows

More Aggressive: Hiding



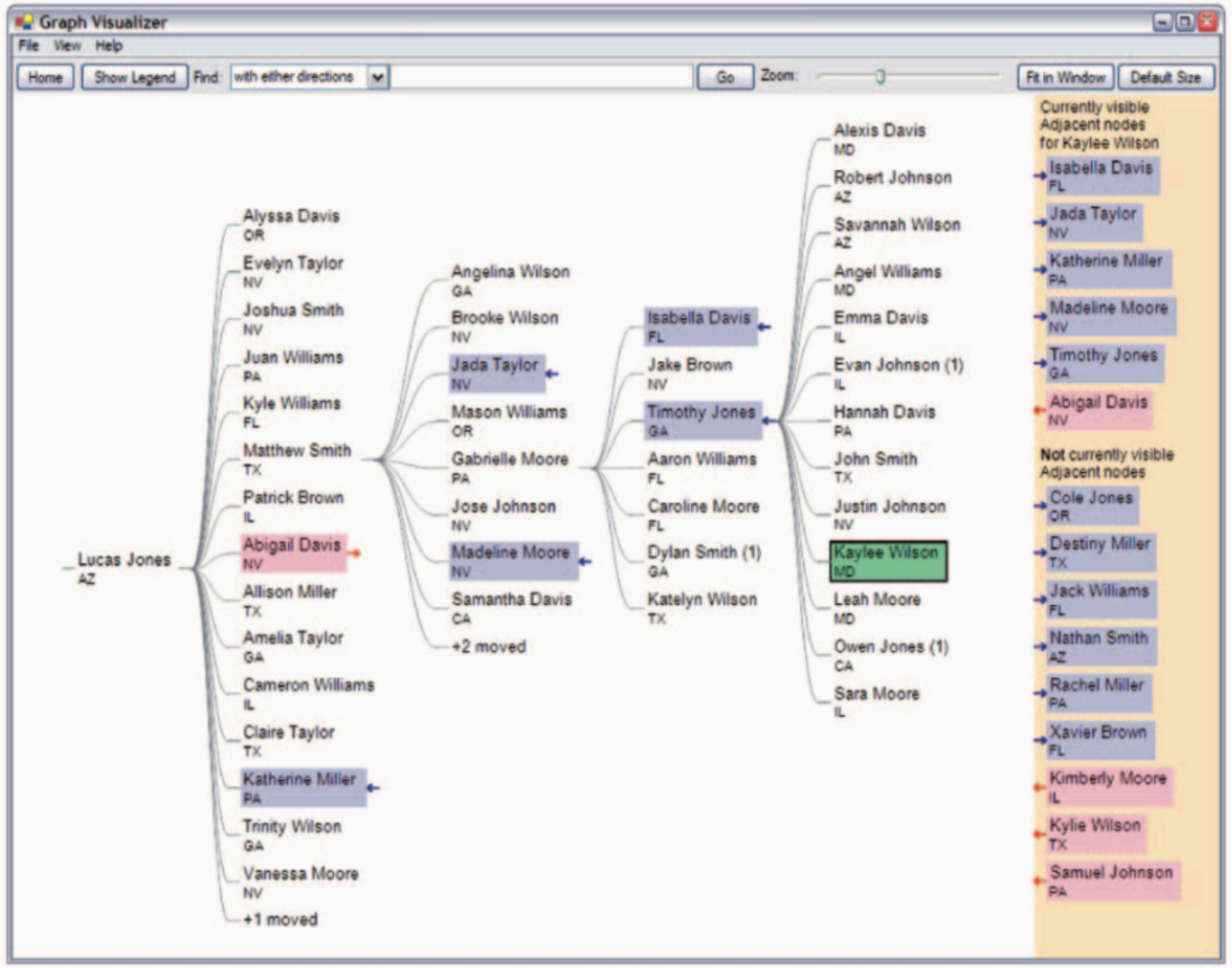


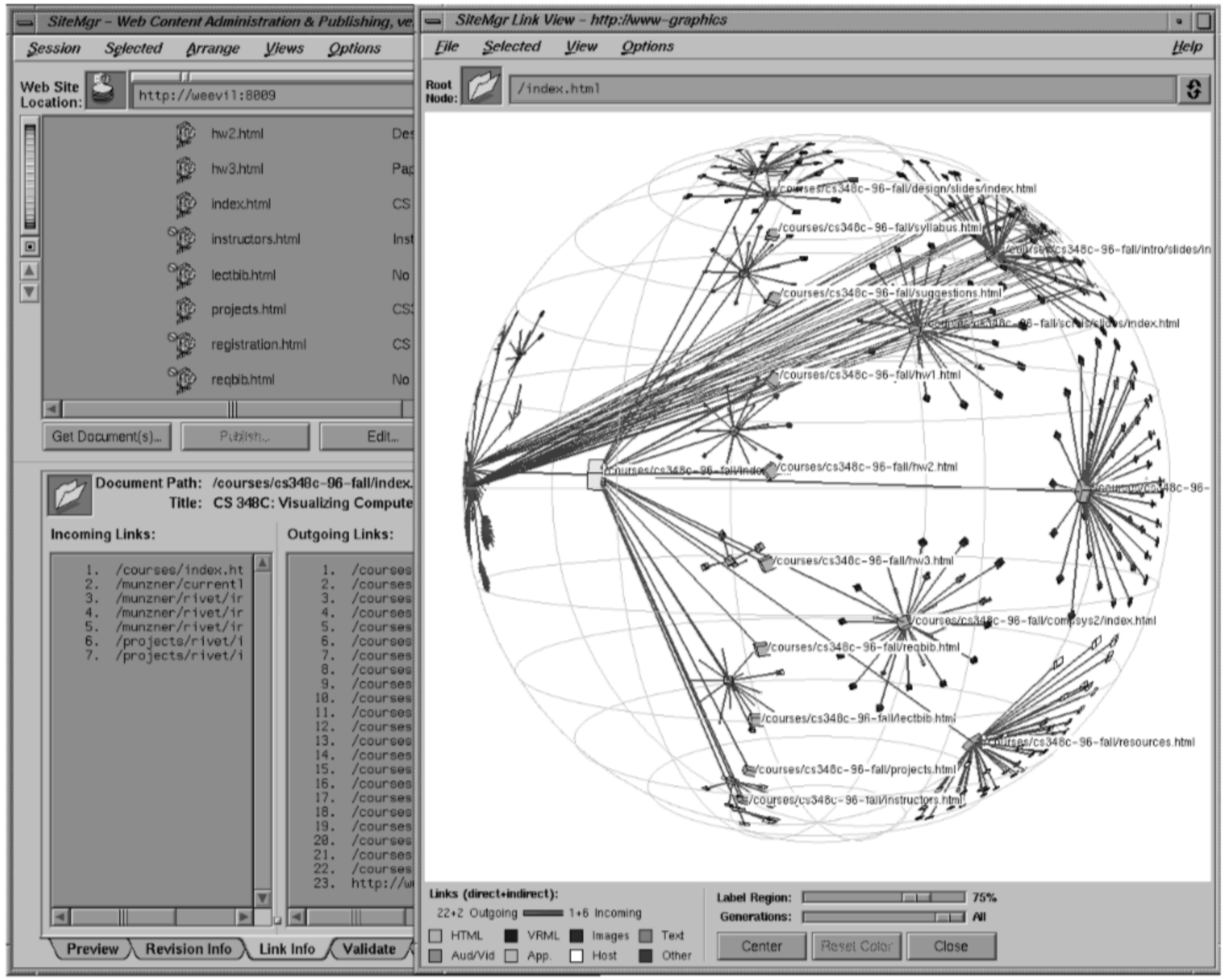


Family Selector
Expand
ID #People #POI PD
38 121 12 (9.9%) 2 (16.7%)
27251404 39 (9.7%) 6 (15.4%)
149 113 10 (8.8%) 0 (0%)
42623 81 10 (12.3%) 1 (10%)
68939244 23 (9.4%) 4 (17.4%)
60348181 19 (10.5%) 1 (5.3%)
Deceased
Alive
POI Suicide
Age at death
26
Age Line
Deceased
Parents
Children
Family 38
Kindredl
Relative
sex (F)
deceased
suicide
Age
bipolar spectrum illn
anxiety-non-trauma
alcohol
psychosis
depression
cause_de
PD
42%
100%
100%
6%
6%
8%
2%
21%
15%
Age : 17
gunshot wo...
mixed drug...
quetiapine...
asphyxia
carbon mon...
self-infli...

Could we use something like
Lineage for general
Multivariate Networks?

Use a Spanning Tree to Visualize a Graph

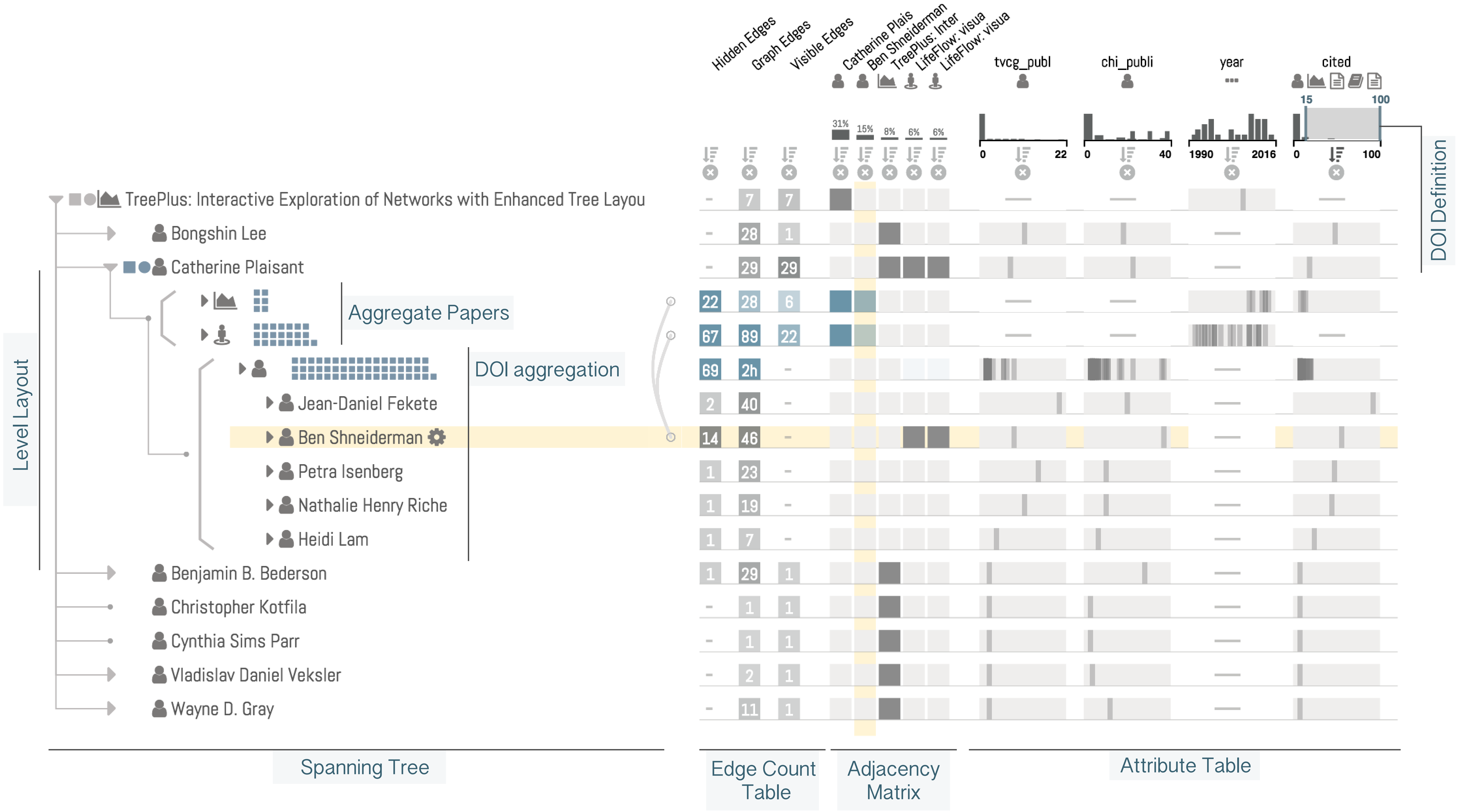




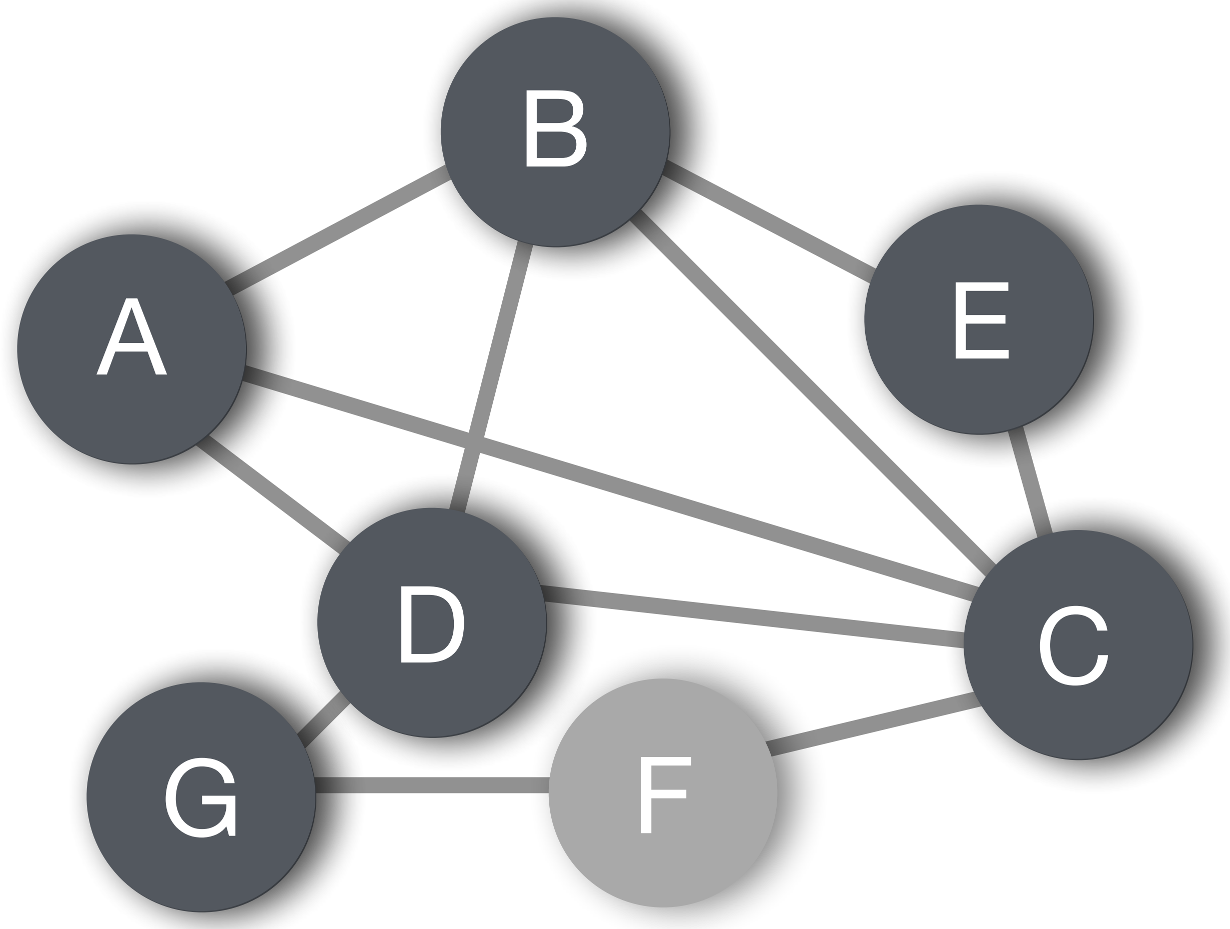
[Lee et al., TreePlus, 2006]

[Munzner, H3Viewer, 1998]

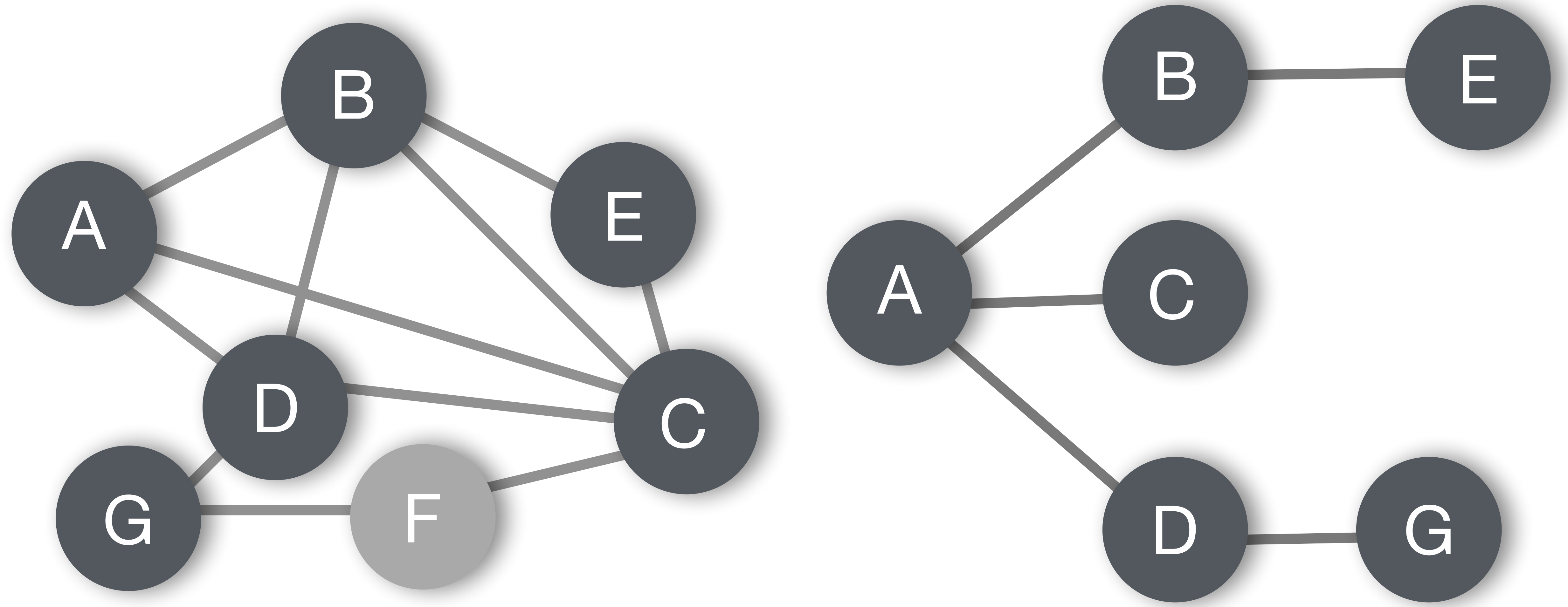
Linearizing a Spanning Tree: Juniper

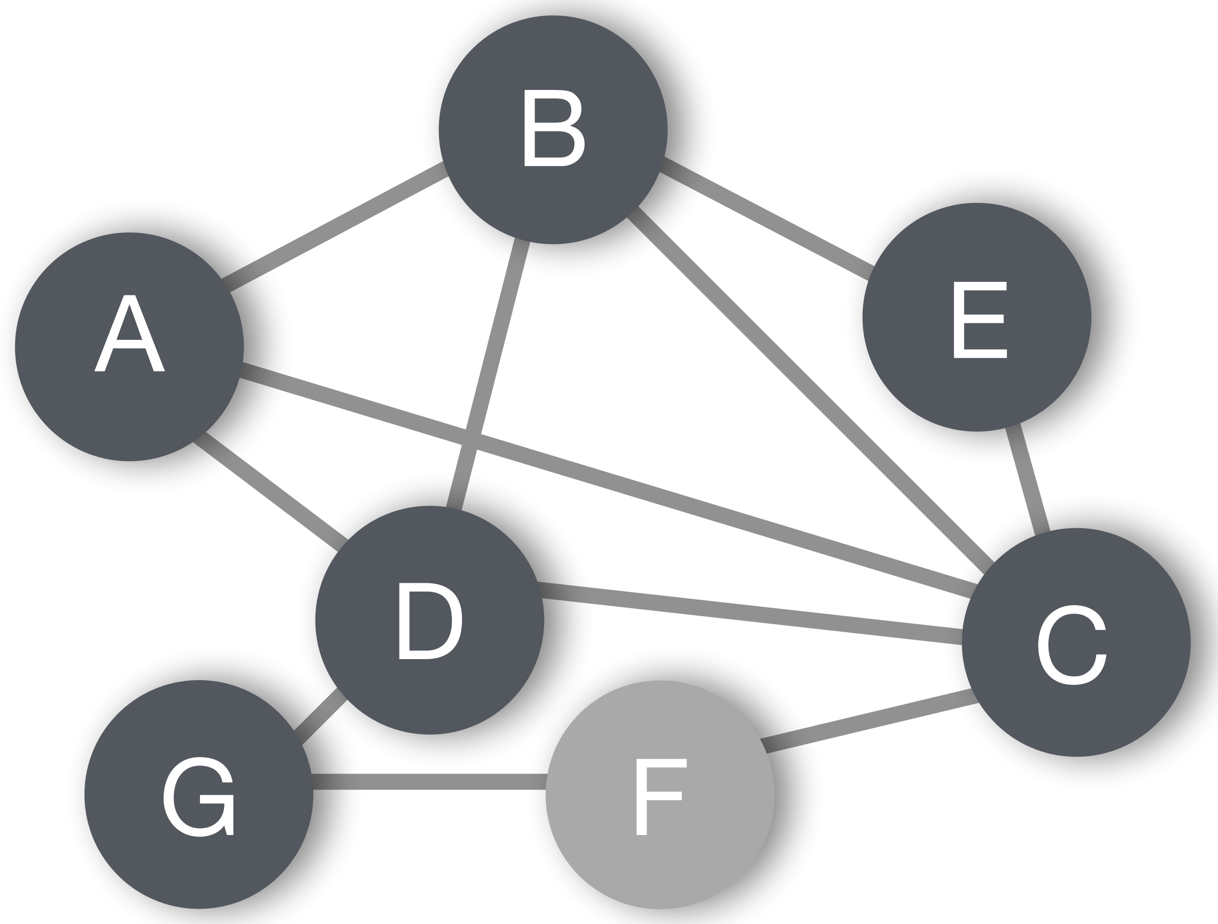


[Nobre et al, Juniper, Preprint 2018]

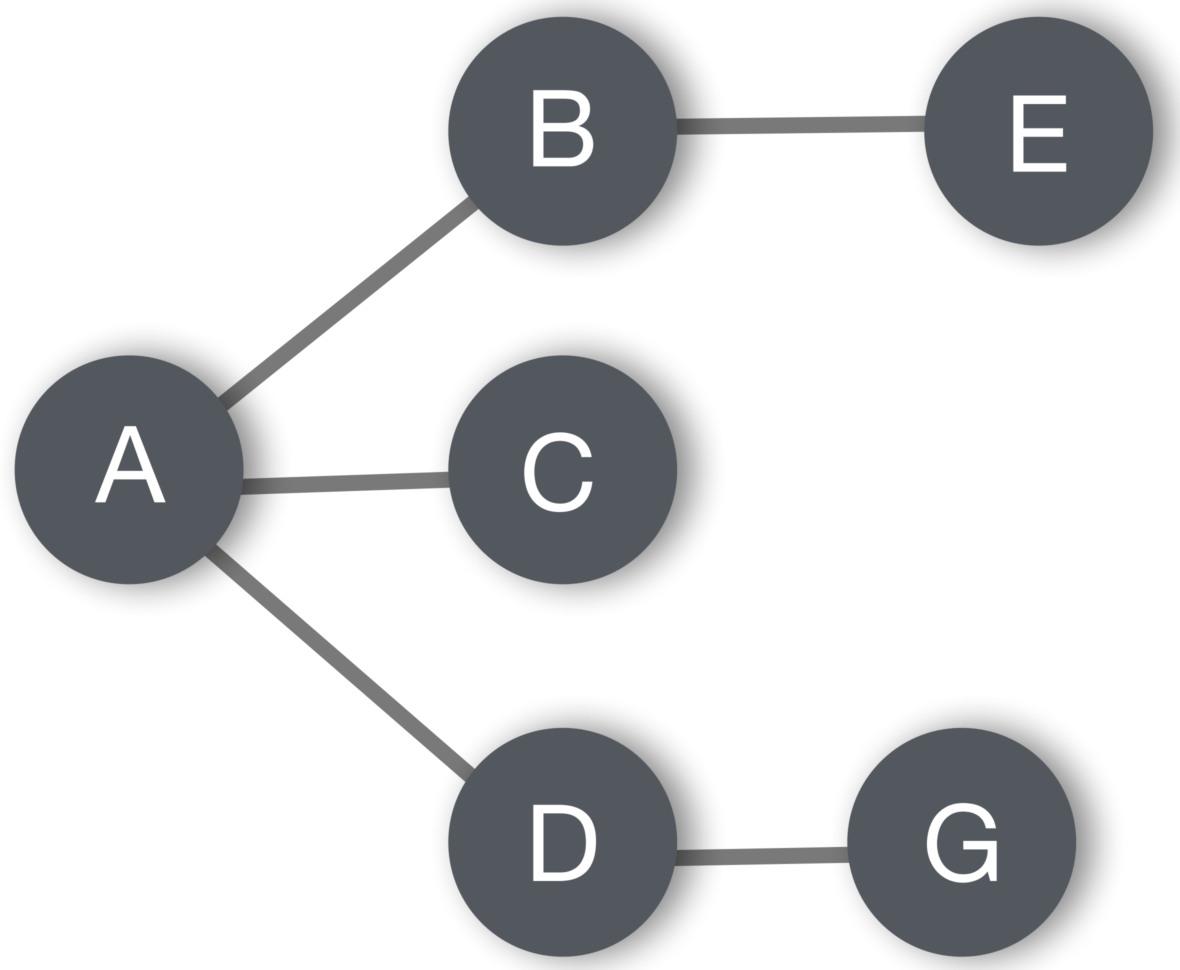


B
A
E
D
C
G
F

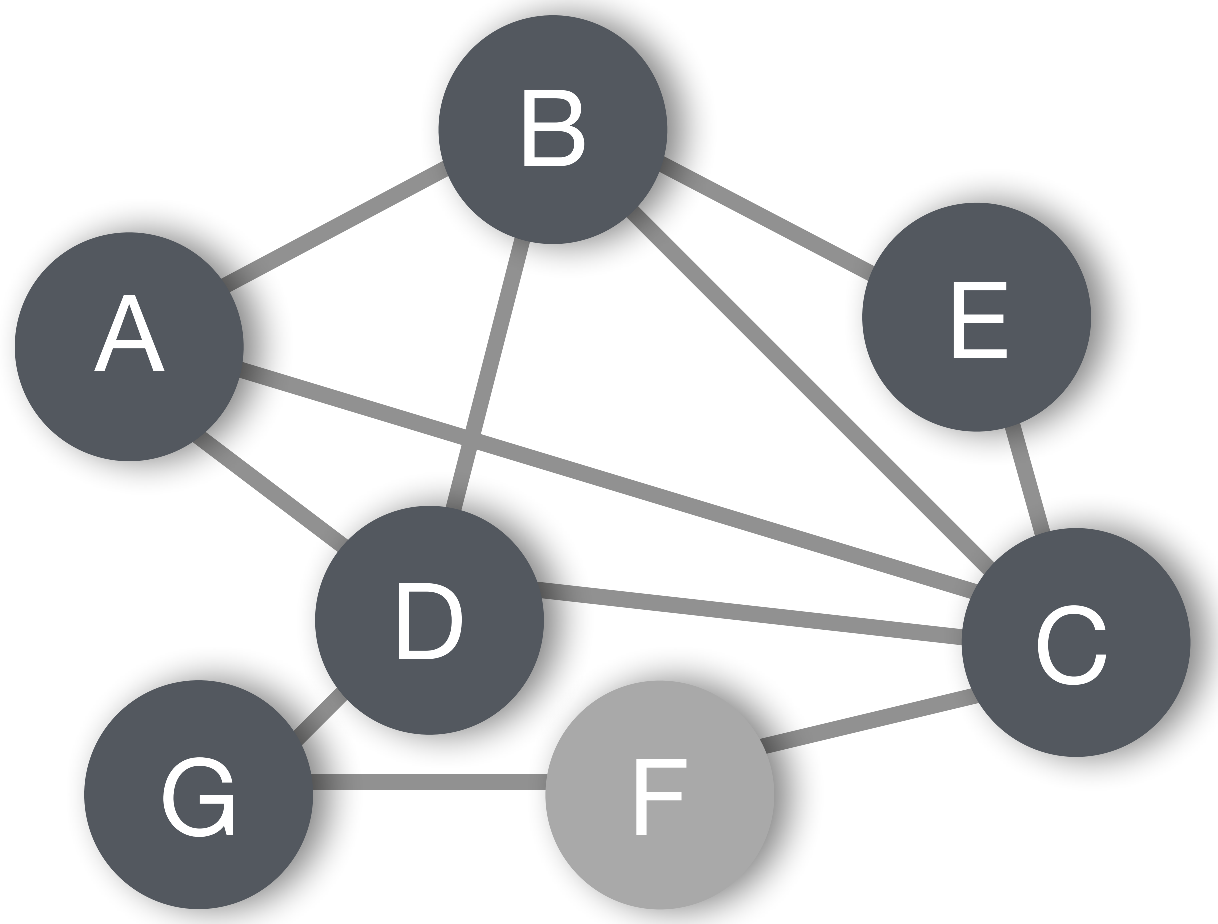

B
A
E
D
C
G
F
B
E
A
C
D
G

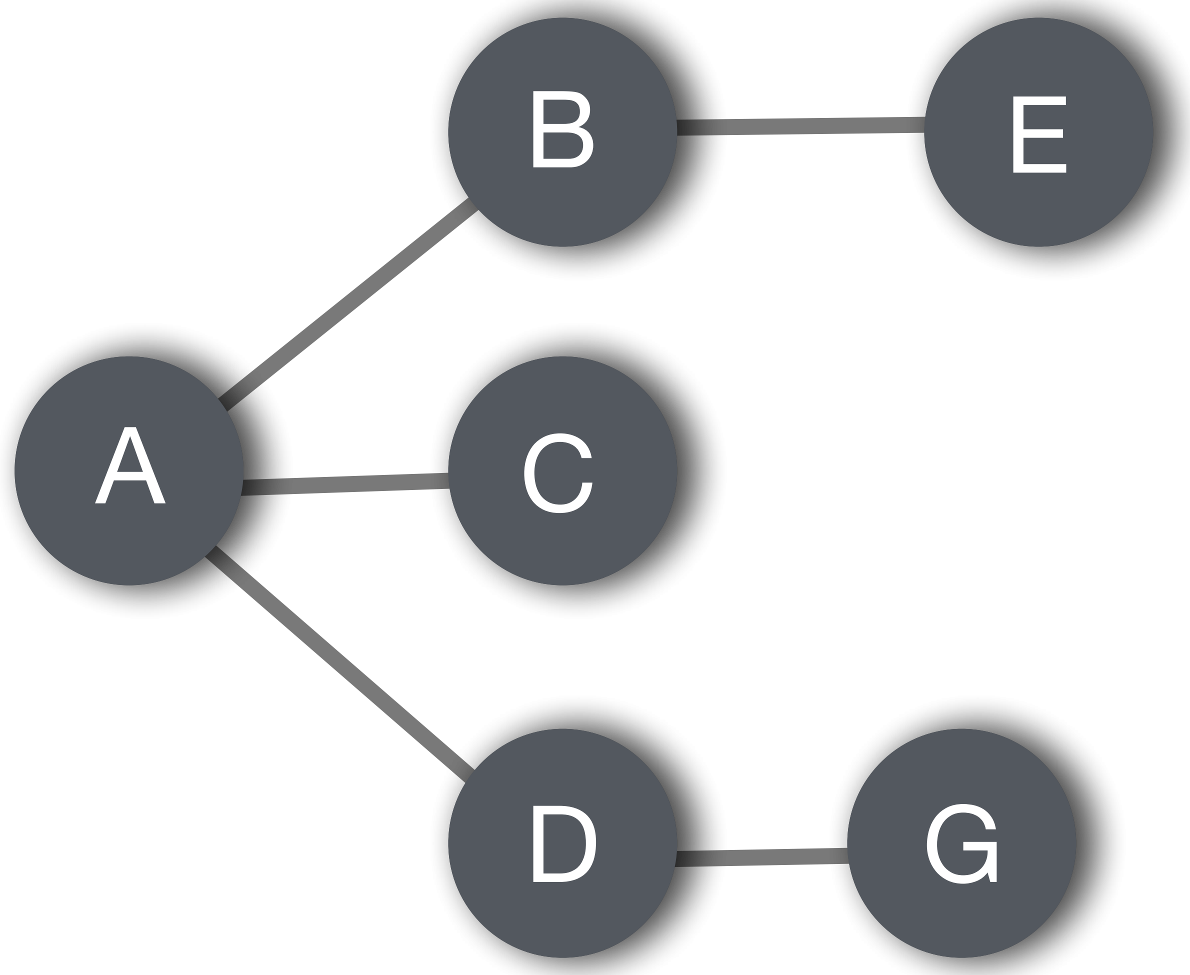


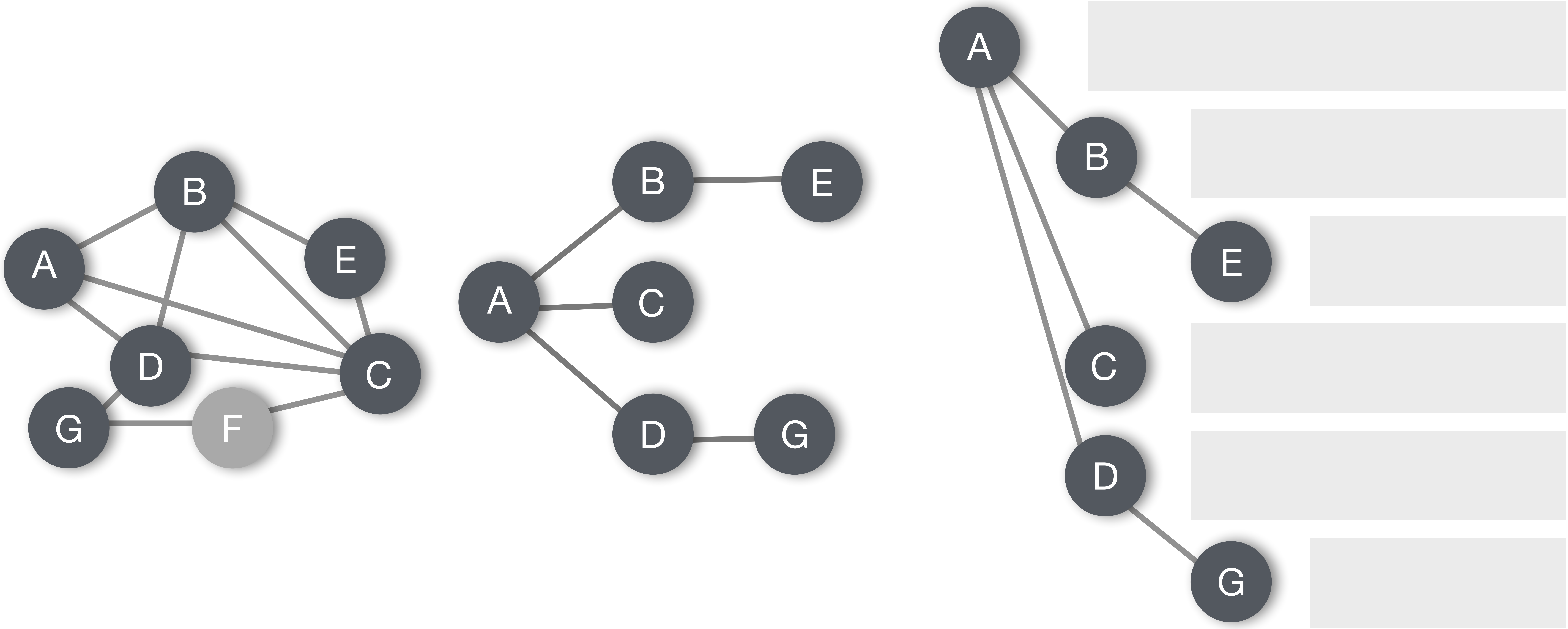
B
A
E
D
C
G
F

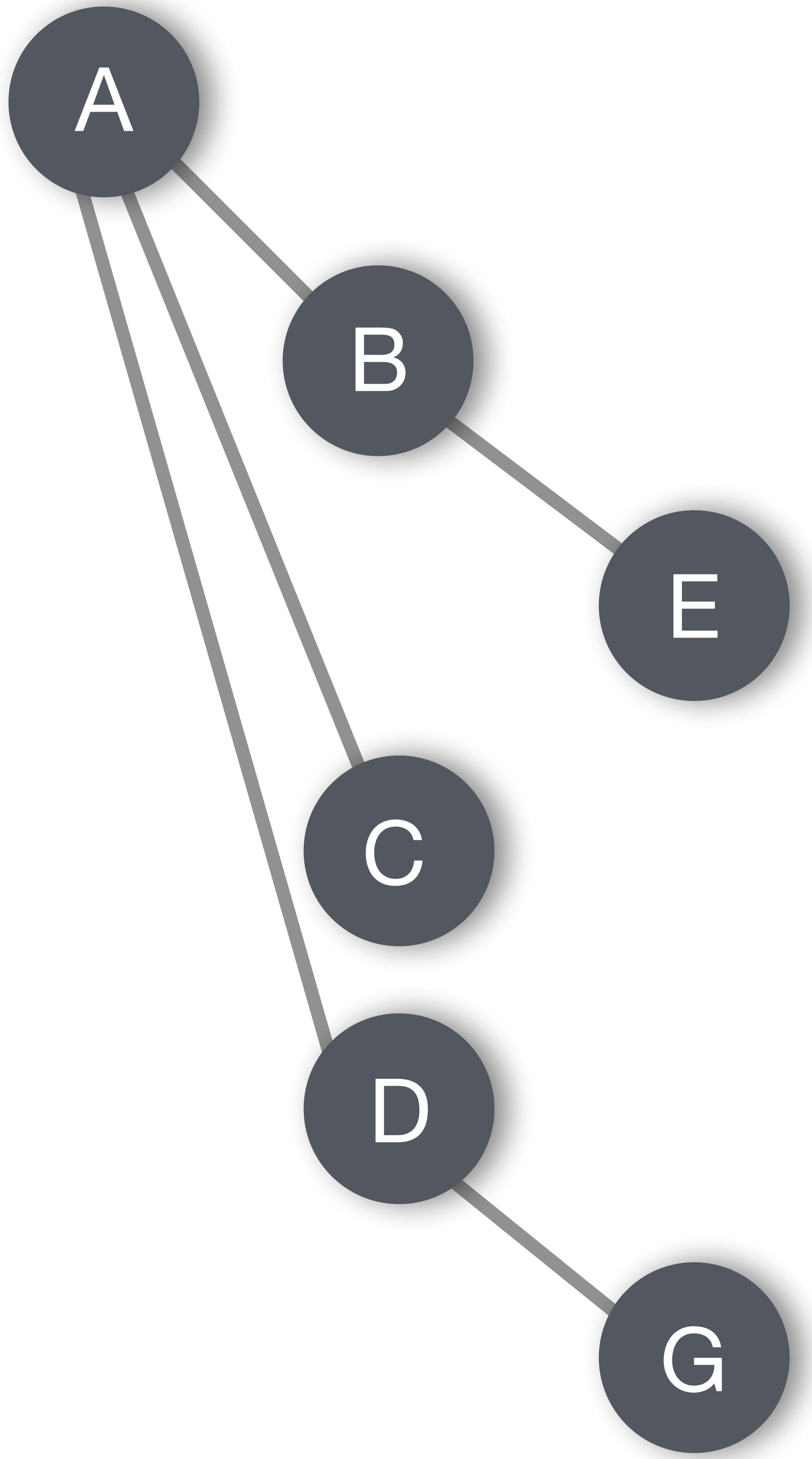


B
E
A
C
D
G


A
B
E
D
C
G
F

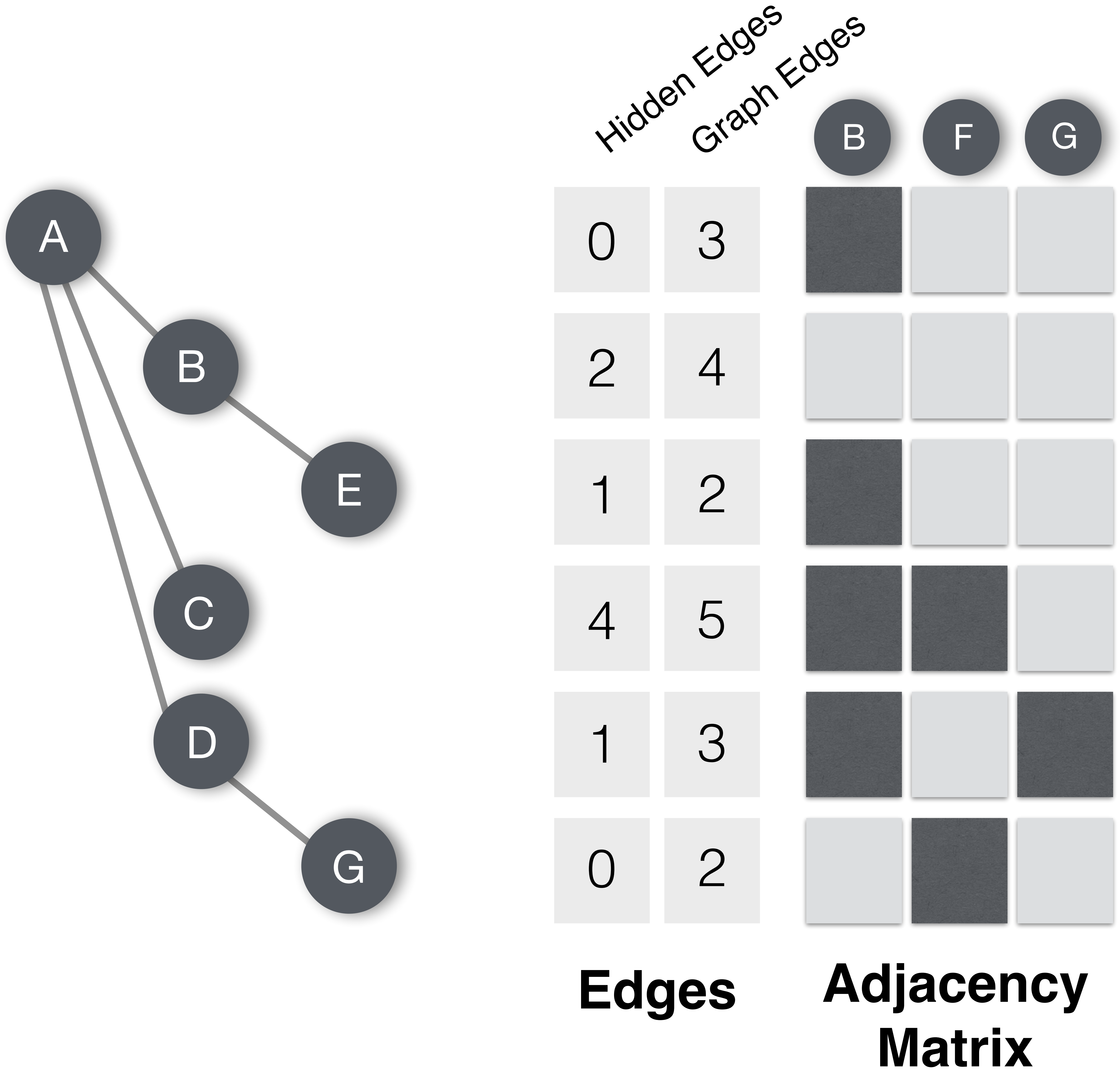

B
E
A
C
D
G

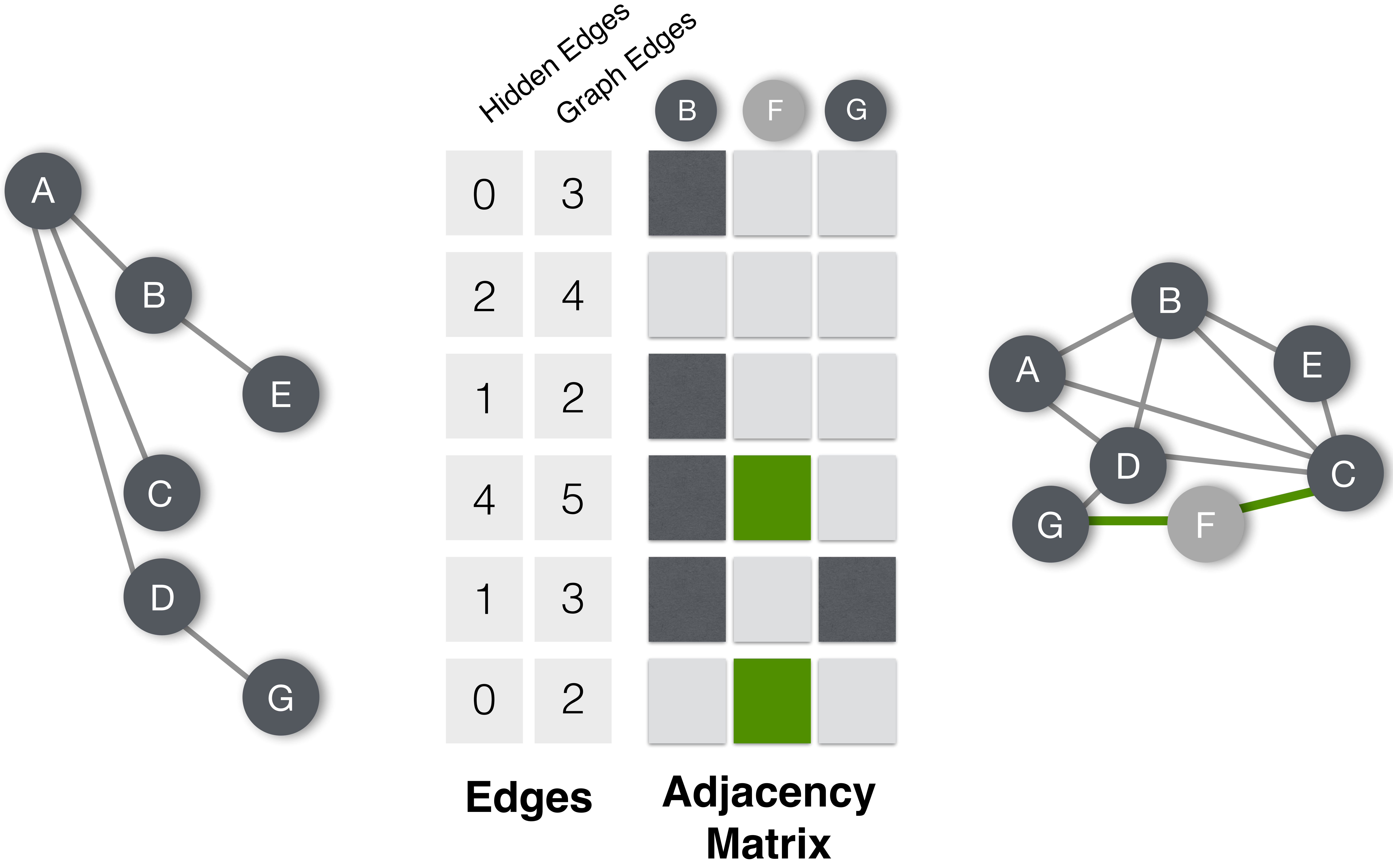

A
B
E
C
D
G
F
A
B
E
C
D
G
A
B
E
C
D
G



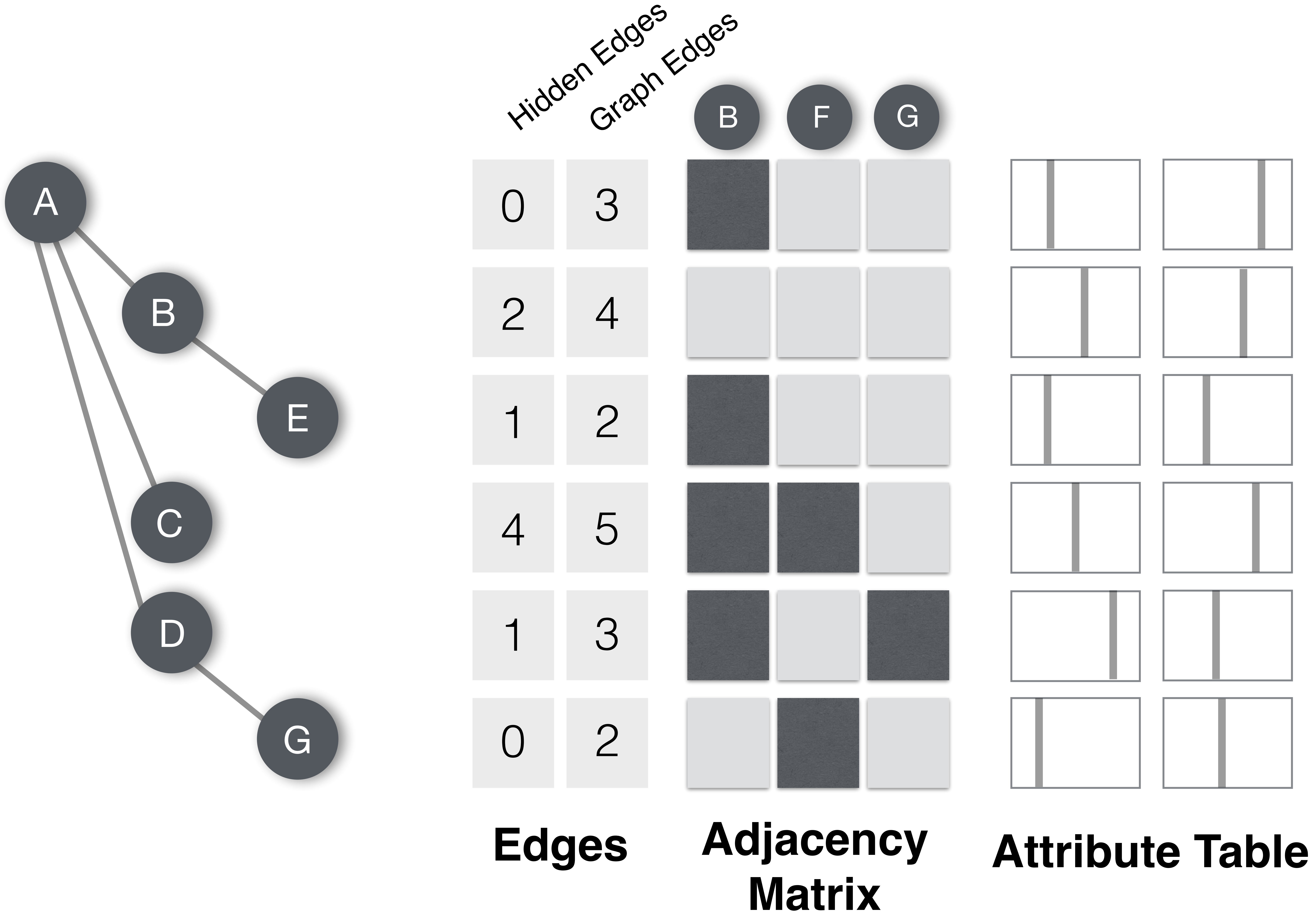
Hidden Edges	Graph Edges
0	3
2	4
1	2
4	5
1	3
0	2

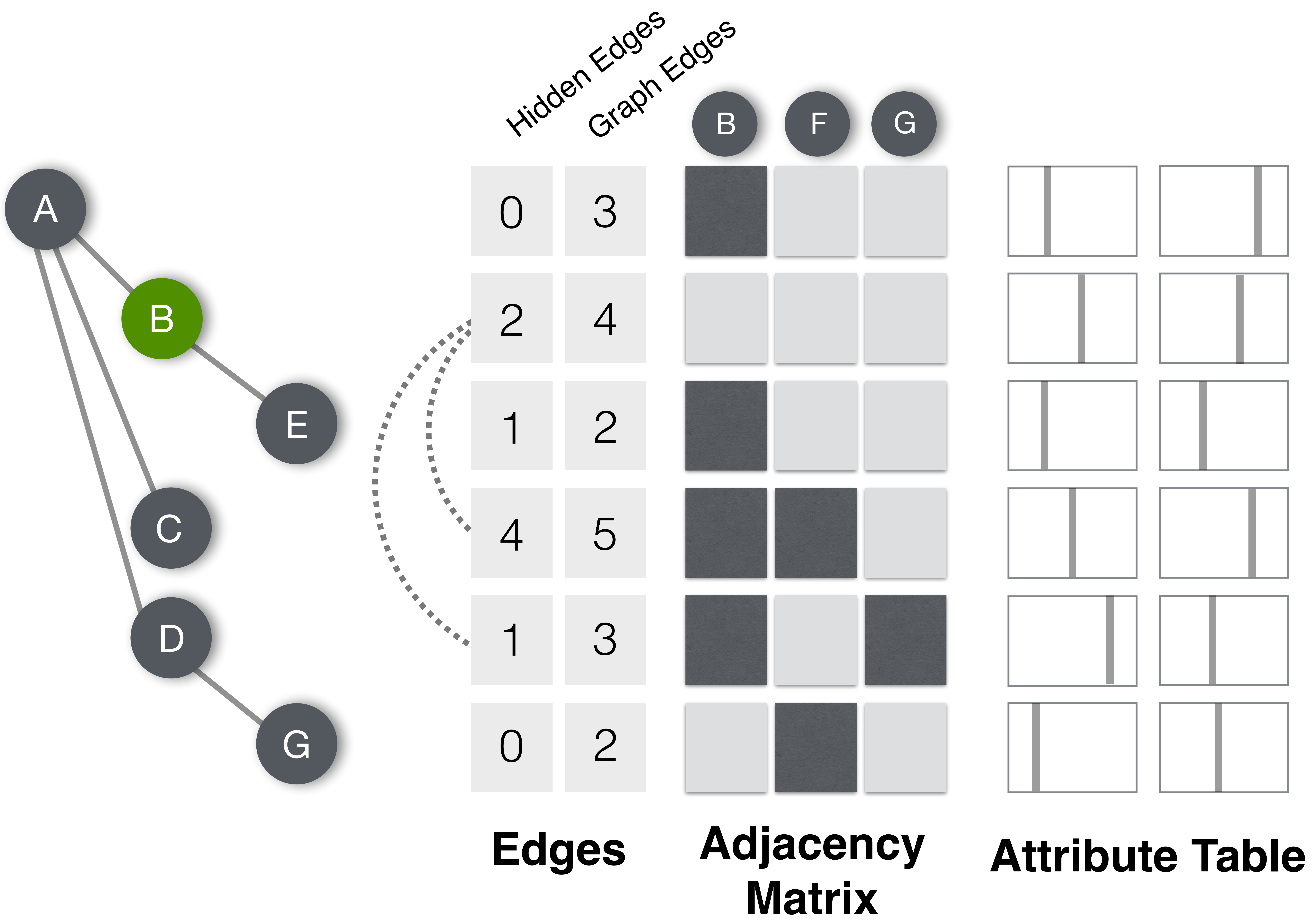
Edges


A
B
E
C
D
G
Hidden Edges
Graph Edges
B
F
G
0 3
2 4
1 2
4 5
1 3
0 2
Edges
Adjacency Matrix

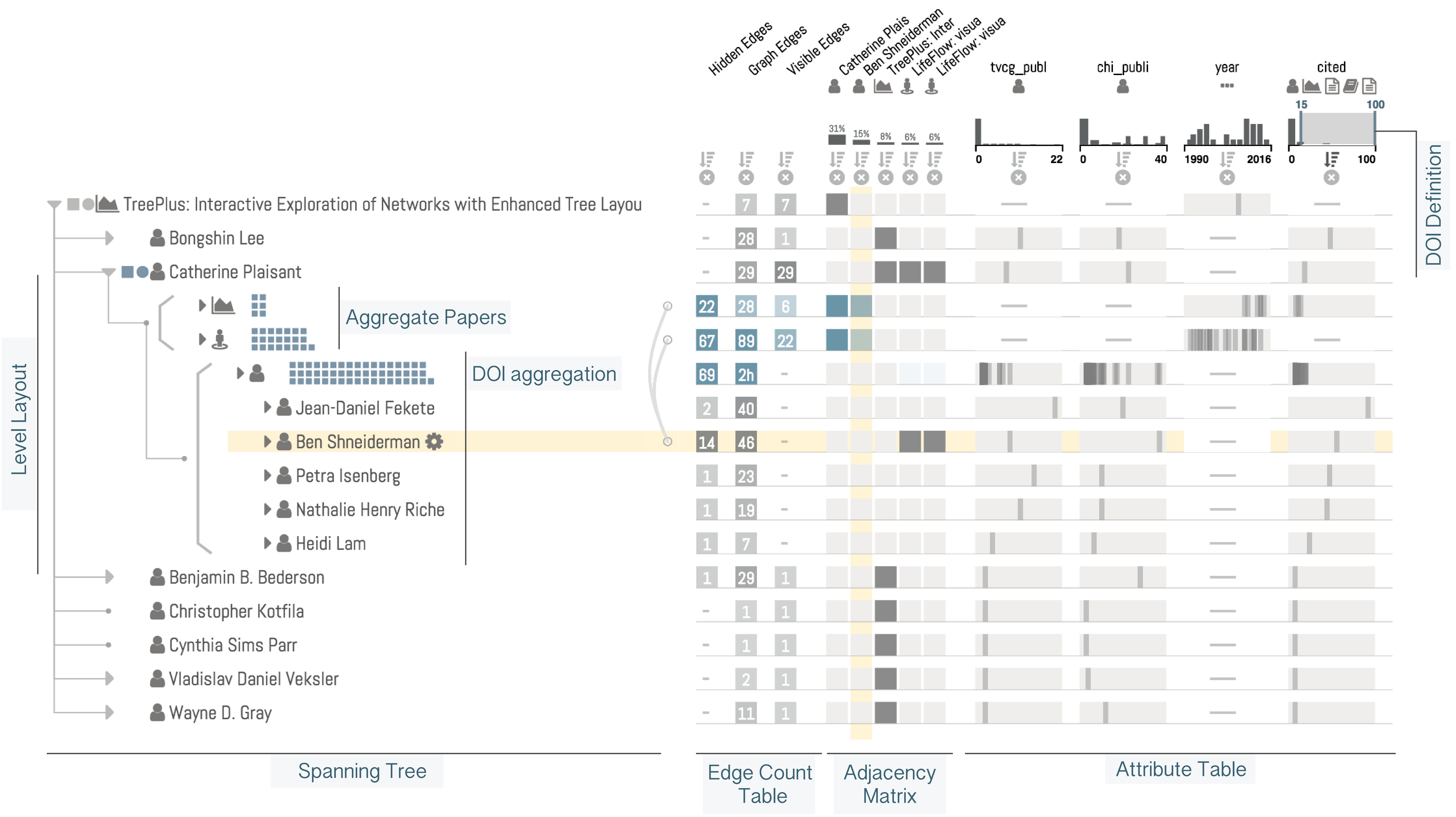


A
B
E
C
D
G
Hidden Edges
Graph Edges
0 3
2 4
1 2
4 5
1 3
0 2
B
F
G
Edges
Adjacency Matrix
B
A
E
D
C
G
F


A
B
C
D
E
G
Hidden Edges
Graph Edges
0 3
2 4
1 2
4 5
1 3
0 2
B
F
G
Edges
Adjacency Matrix
Attribute Table



A
B
E
C
D
G
Hidden Edges
Graph Edges
B
F
G
0
3
2
4
1
2
4
5
1
3
0
2
Edges
Adjacency Matrix
Attribute Table



Hidden Edges
Graph Edges
Visible Edges
Catherine Plais
Ben Shneiderman
TreePlus: Inter
LifeFlow: visua
LifeFlow: visua
tvcg_publ
chi_publi
year
cited
31%
15%
8%
6%
6%
0
22
0
40
1990
2016
15
100
0
100
TreePlus: Interactive Exploration of Networks with Enhanced Tree Layou
Bongshin Lee
Catherine Plaisant
Aggregate Papers
DOI aggregation
Jean-Daniel Fekete
Ben Shneiderman
Petra Isenberg
Nathalie Henry Riche
Heidi Lam
Benjamin B. Bederson
Christopher Kotfila
Cynthia Sims Parr
Vladislav Daniel Veksler
Wayne D. Gray
Level Layout
DOI Definition
Spanning Tree
Edge Count Table
Adjacency Matrix
Attribute Table

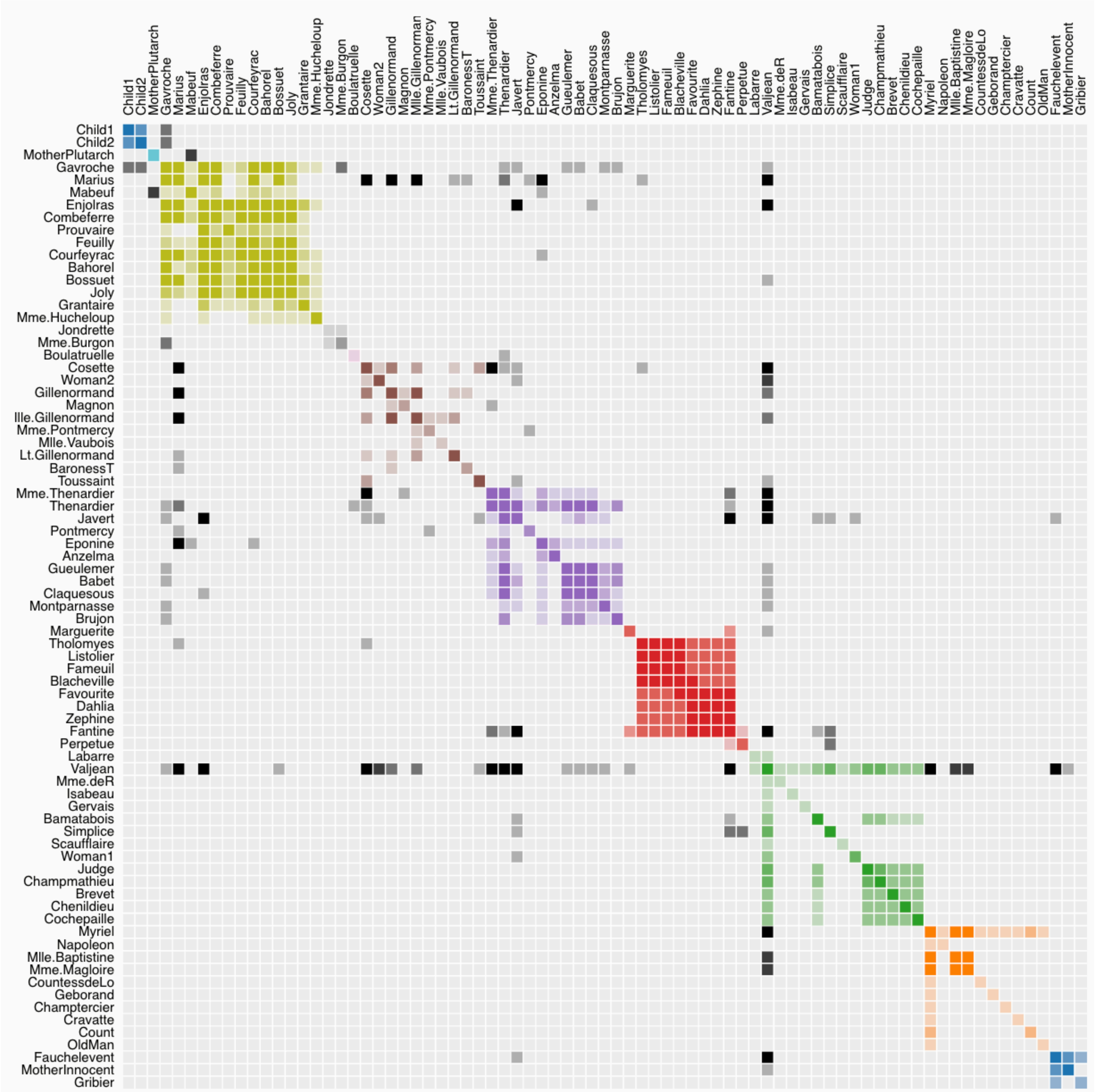
Conclusions

Linearization & Juxtaposition are good options for visualizing Multivariate Graphs

Many tasks are local, leverage the “Search, Show Context, Expand on Demand” principle for multivariate networks

Visualizing Edge Attributes

Good Choice: Matrix



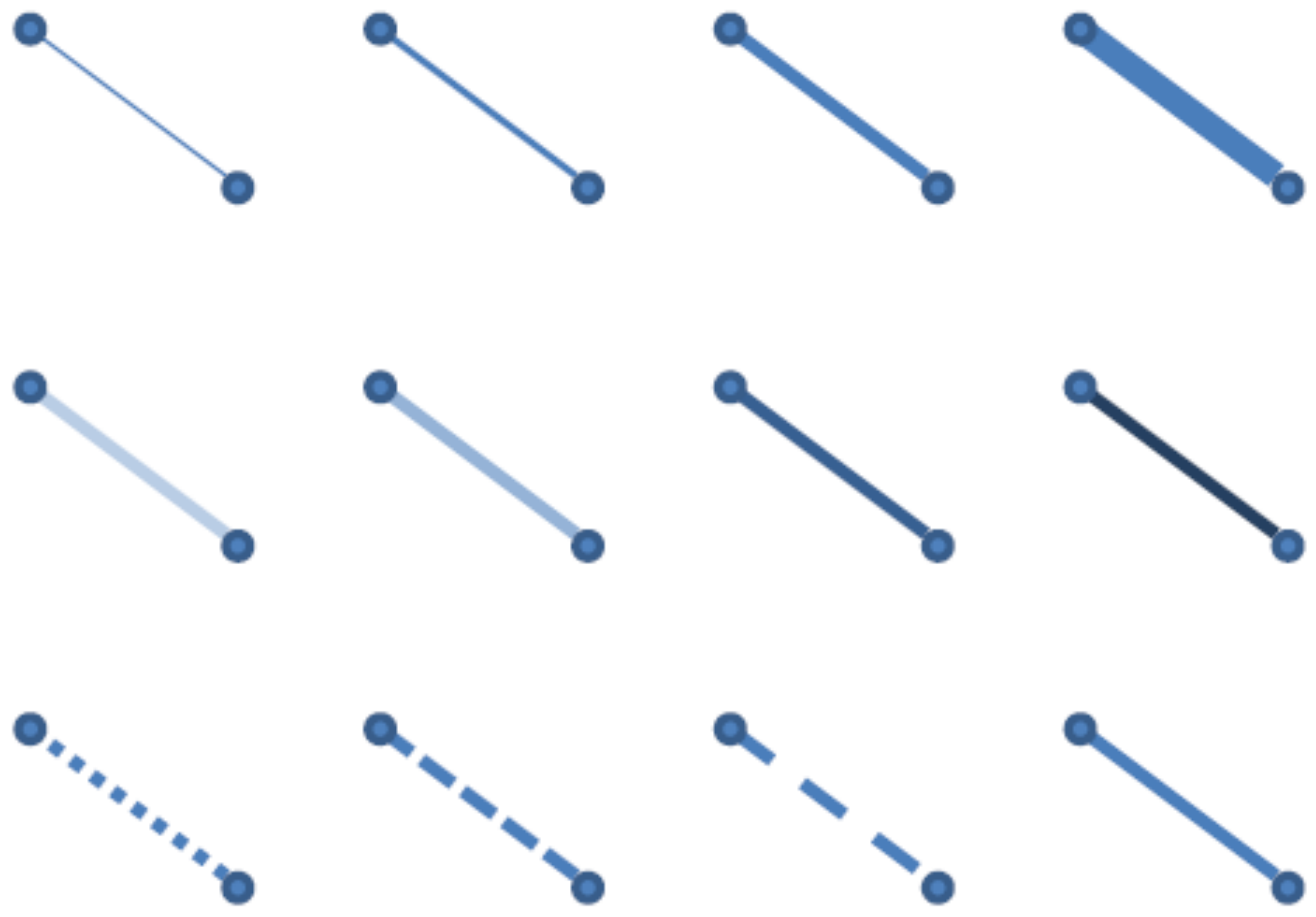
Visualizing Edge Attributes

Most common ways to encode edge attributes

Quantitative: Width

Ordinal: Saturation

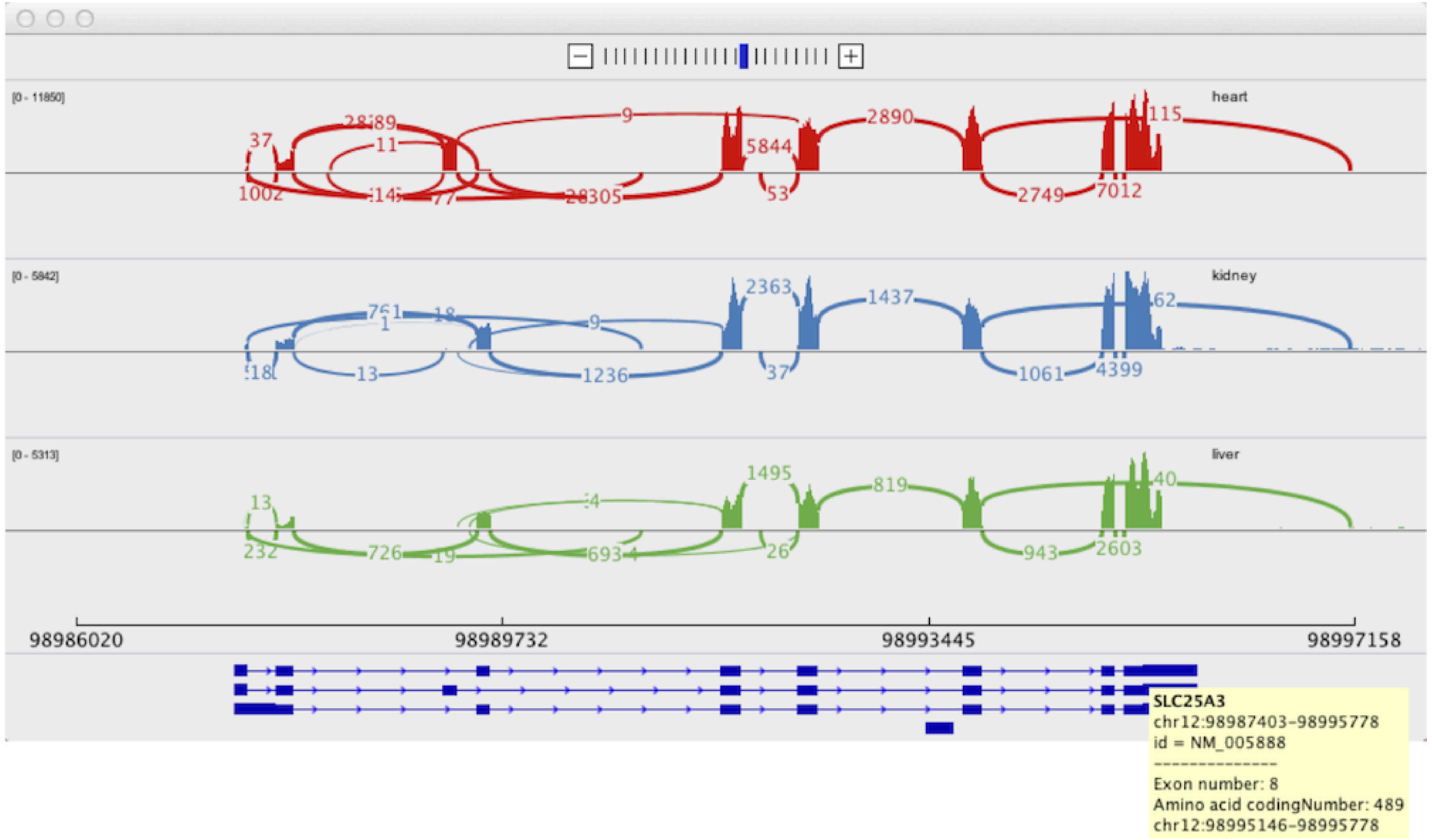
Nominal: Style, Color



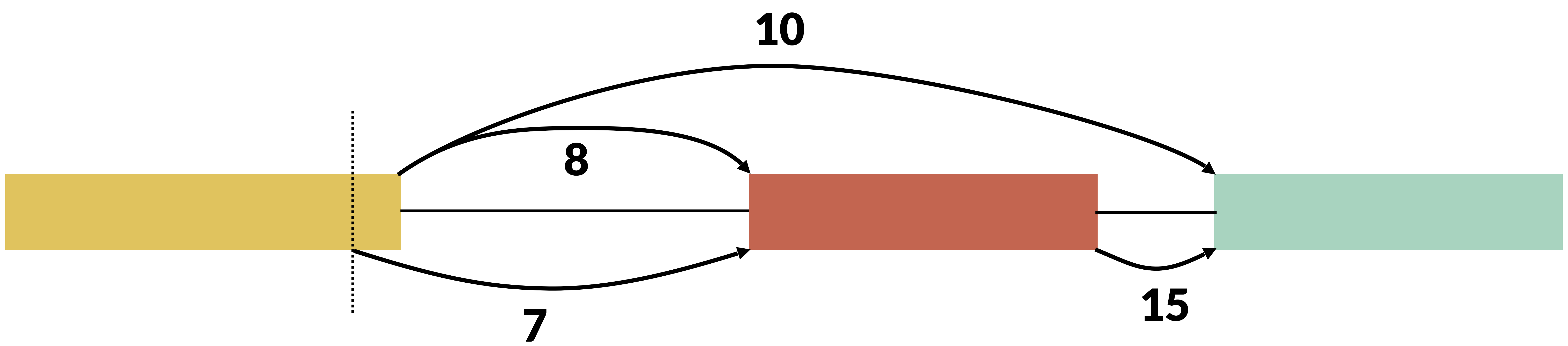
Visualizing Edge Attributes

In practice very limited

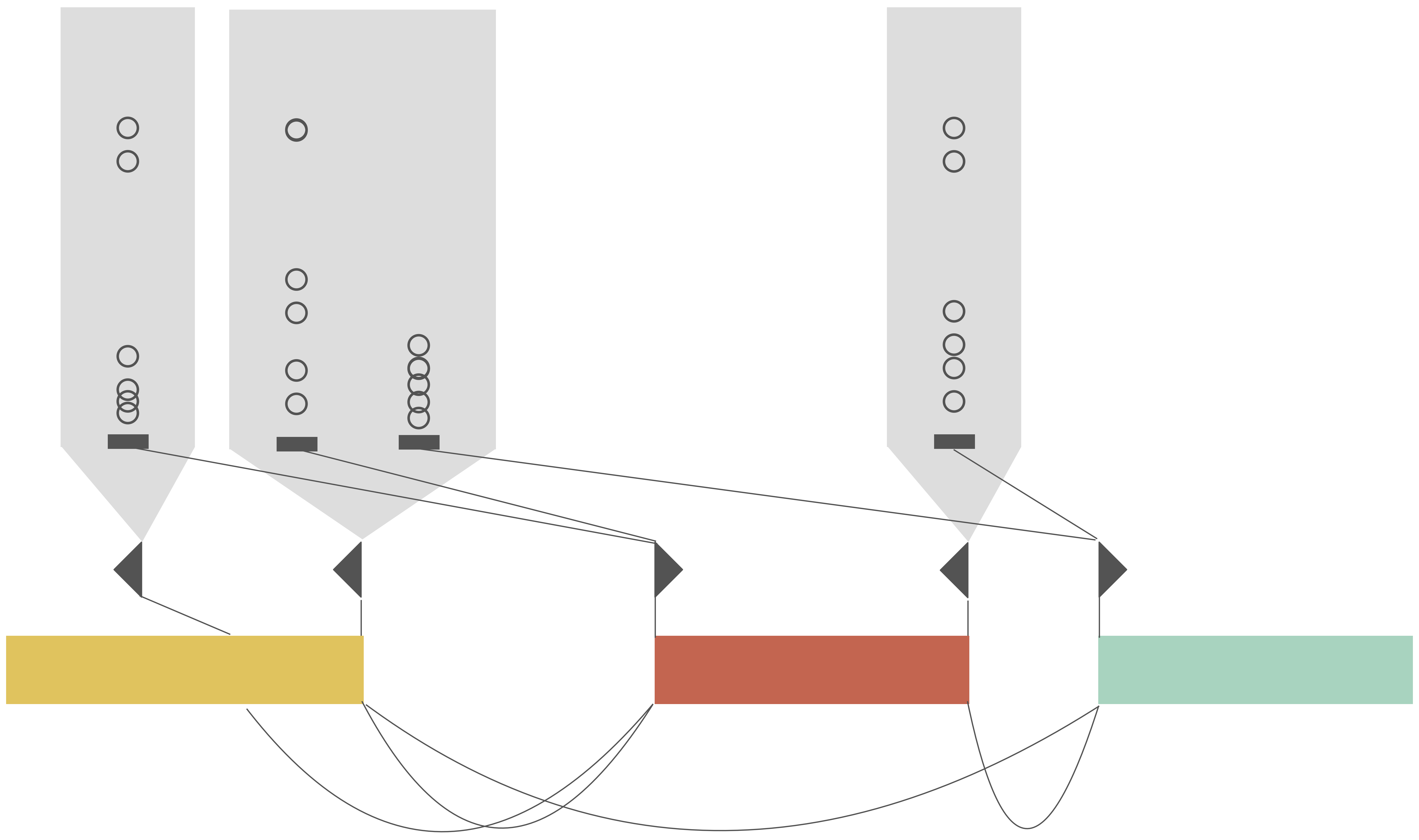
Example: Sashimi Plots



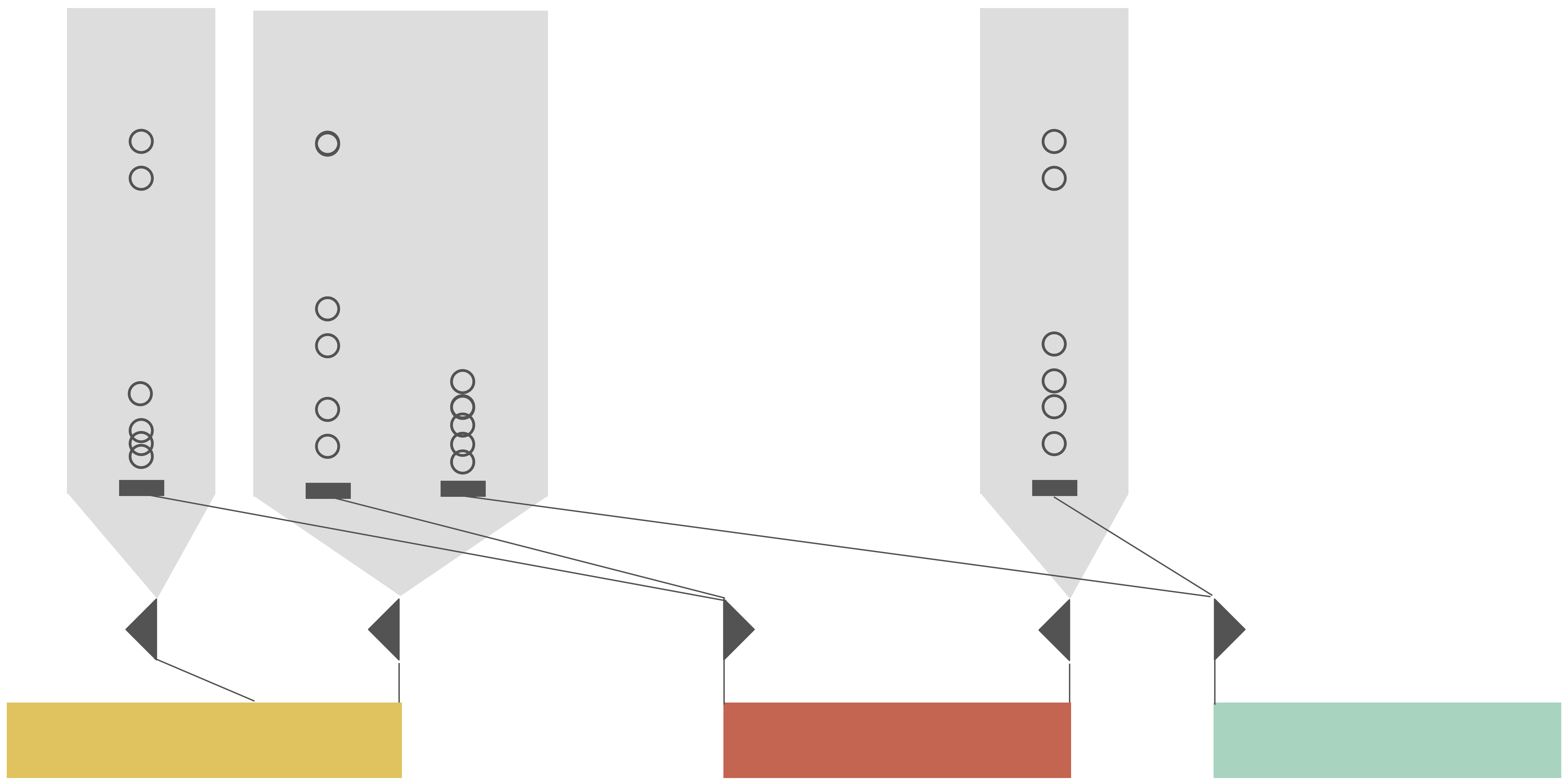
What's the Problem?



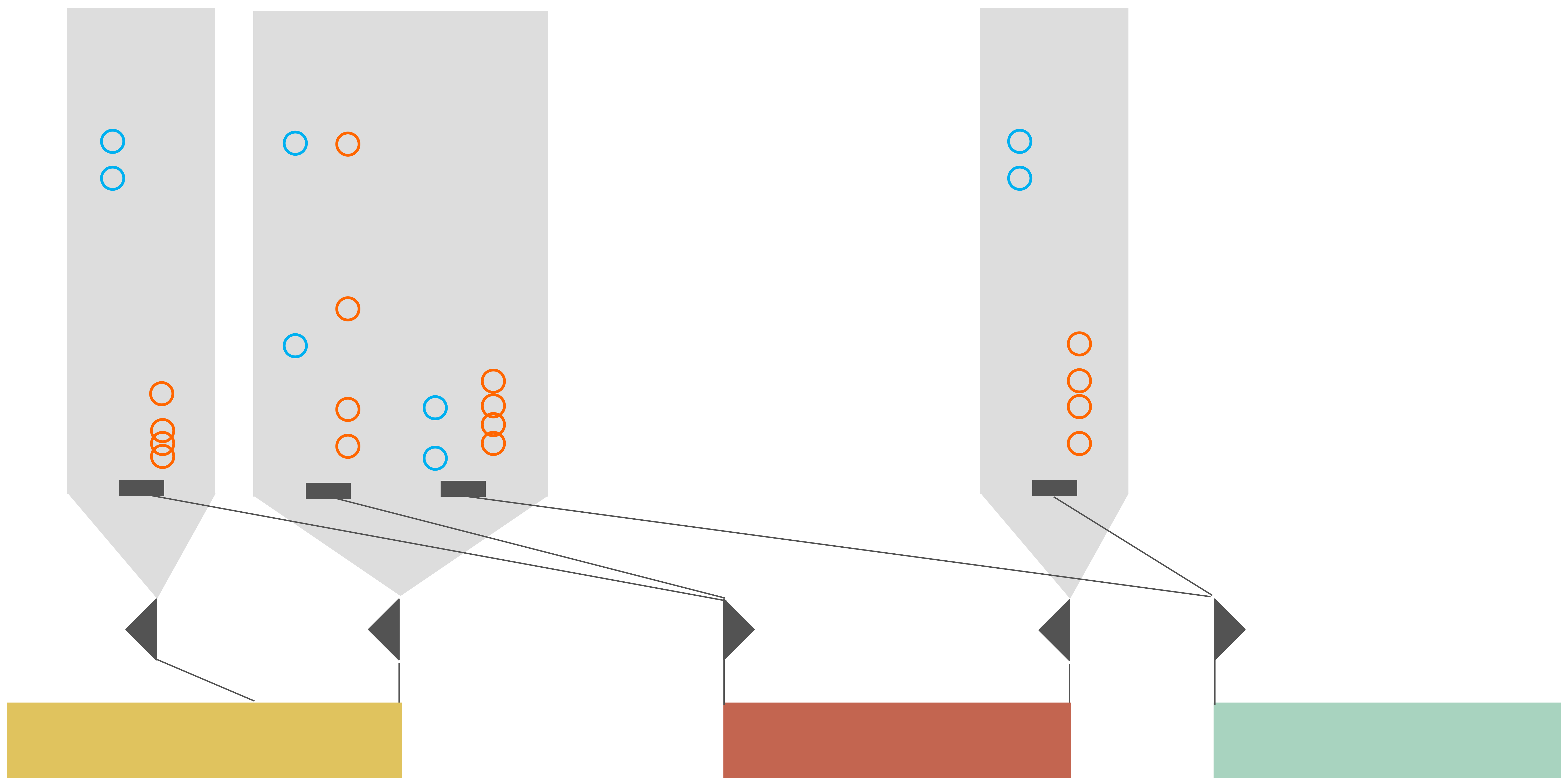
Junction View



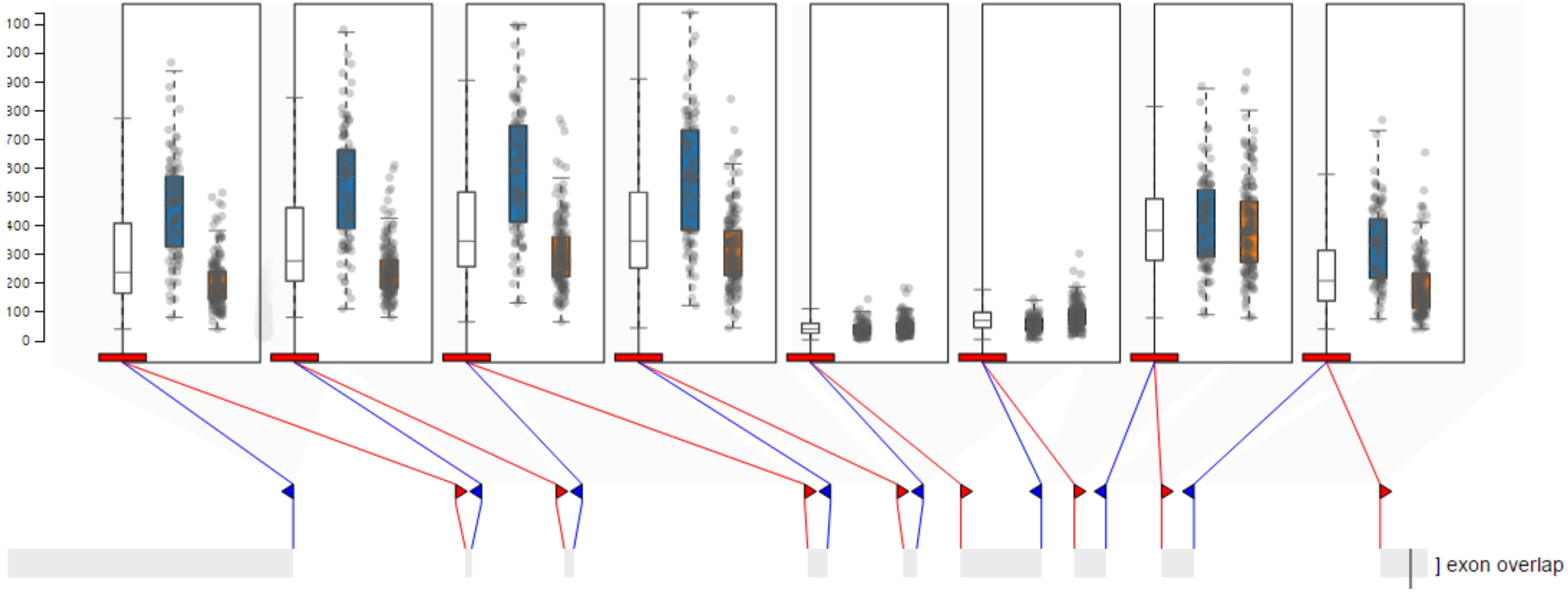
Junction View - Group Comparison



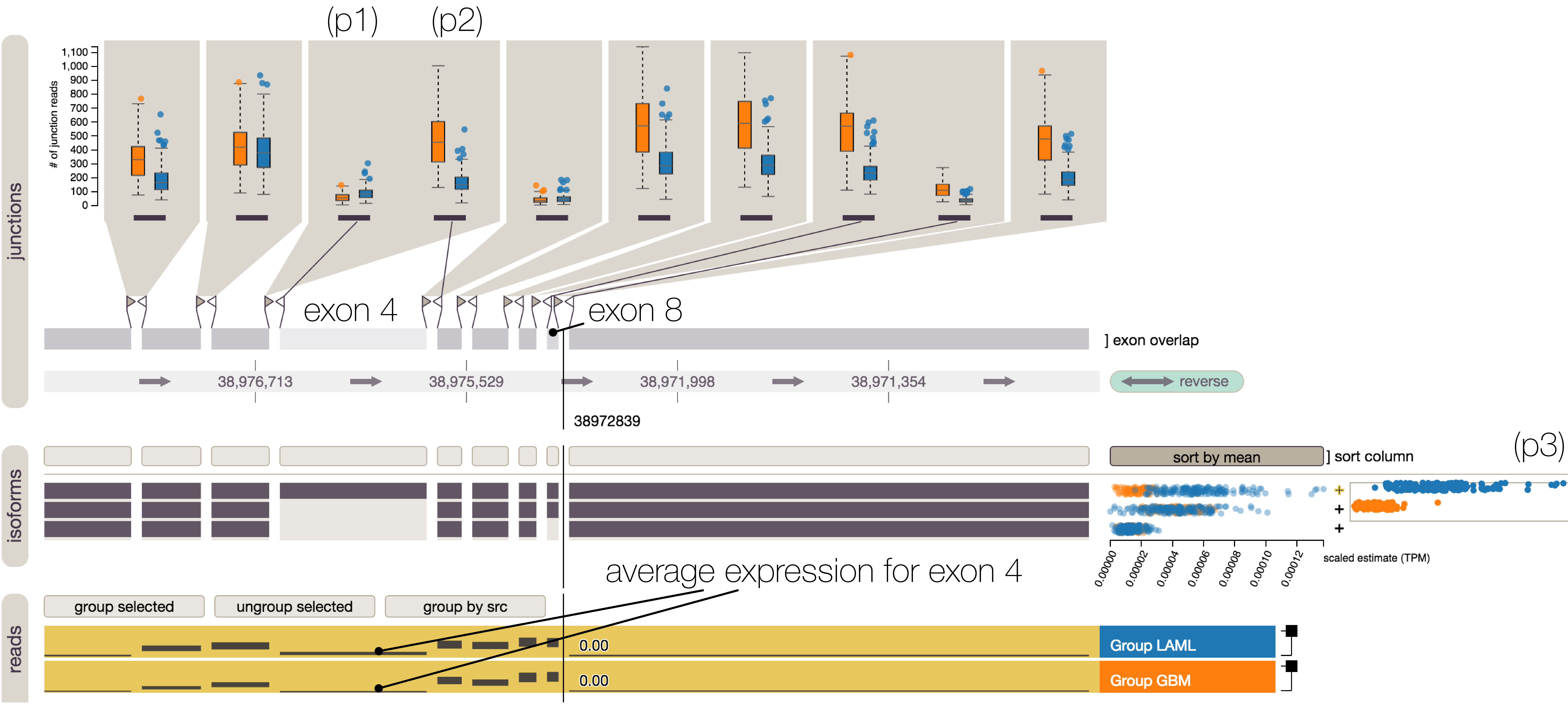
Junction View - Group Comparison



Junction View - Group Comparison



Case Study: Leukemia vs Glioblastoma



Trees

Tree-Exercise

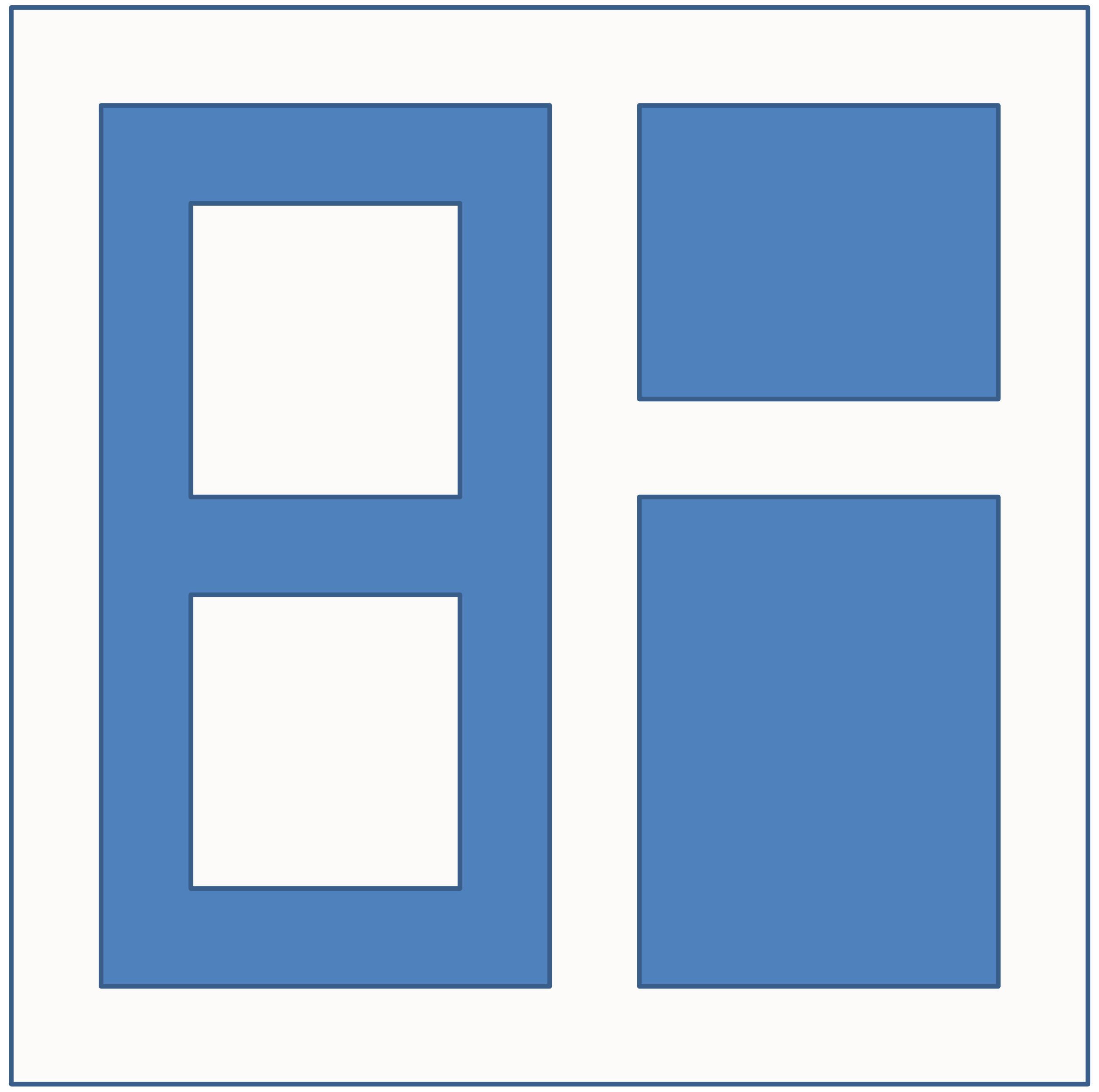
Tree Exercise

Here is part of a directory structure used for the material for this class and the relative file size.

datavis-17/
- lectures/
  - Intro.key (110 MB)
  - perception/
    - Perception.key (113 MB)
    - Blindness.mov (15MB)
  - Data.key (12 MB)
  - Graphs.key (180 MB)
- exams/
  - Exam1-solution.doc (5MB)
  - Exam1.doc (1MB)
- exercise/
  - Graph.doc (3MB)
  - Graph-video.doc (210MB)

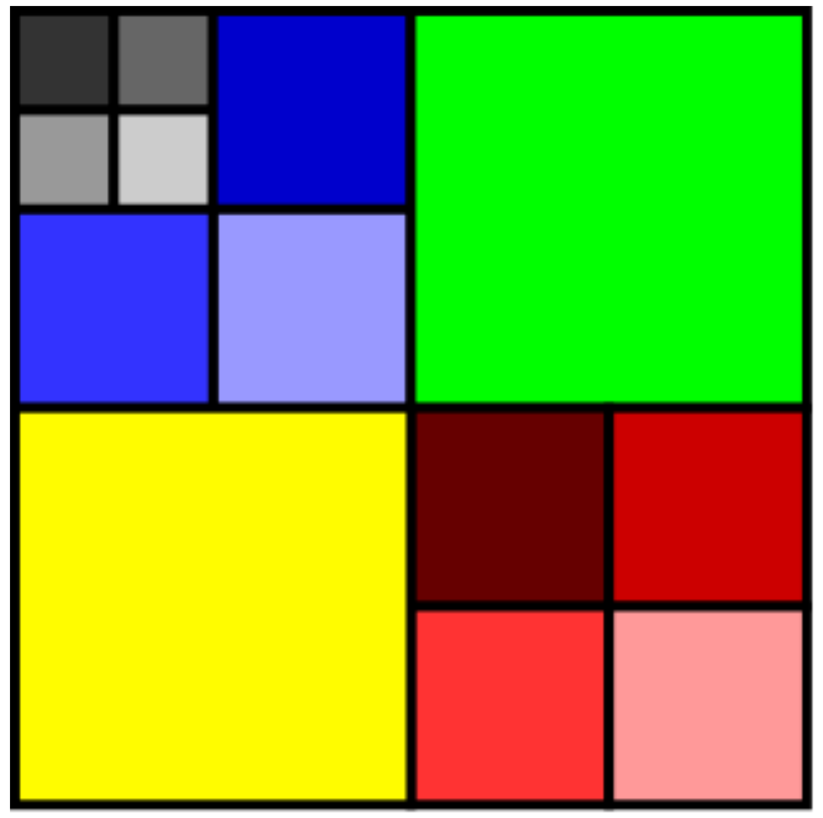
Sketch two different visualizations that show both, the directory structure and the size of the directories and the contained files.

Implicit Layouts for Trees

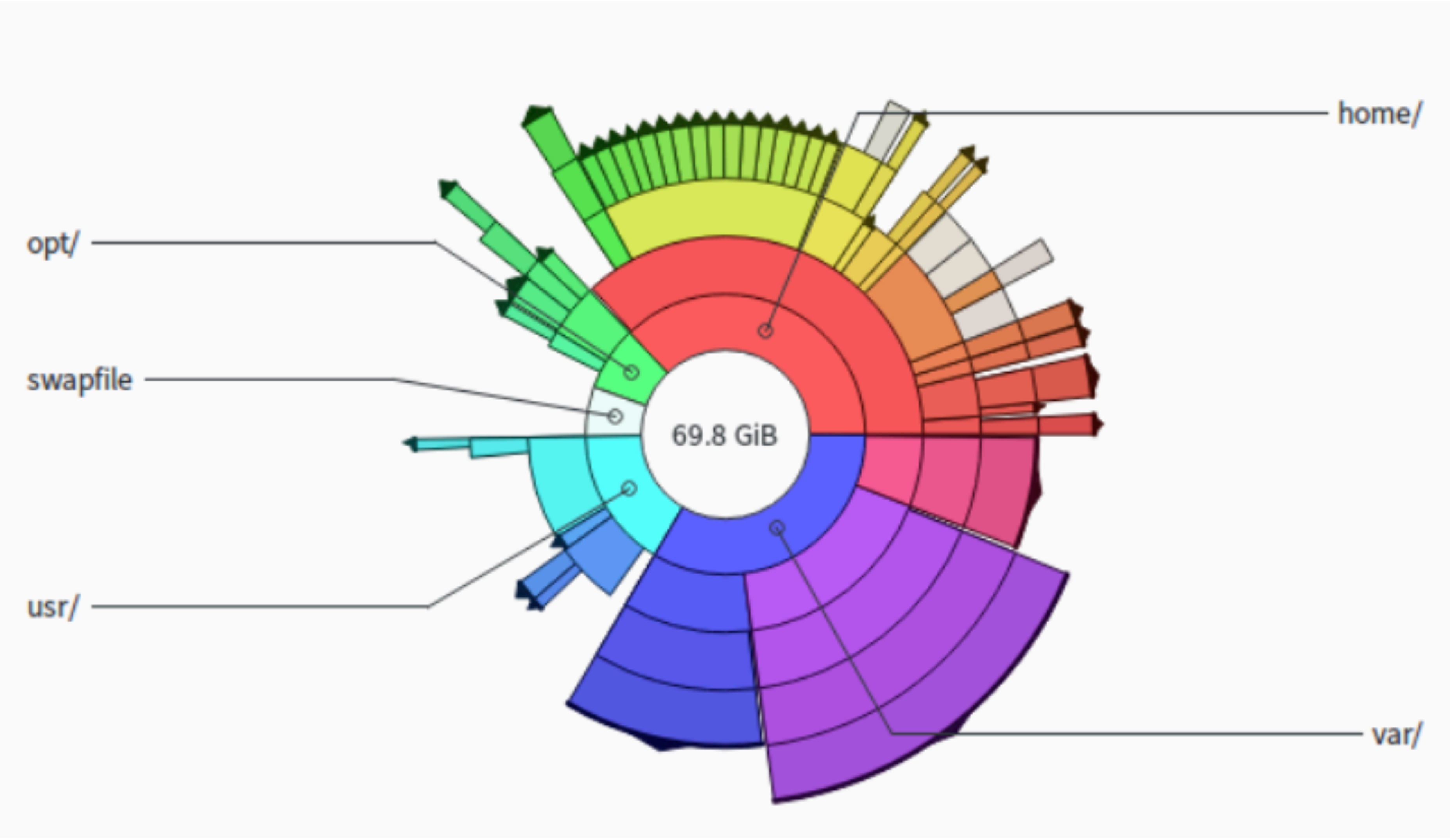


Implicit Layout Options

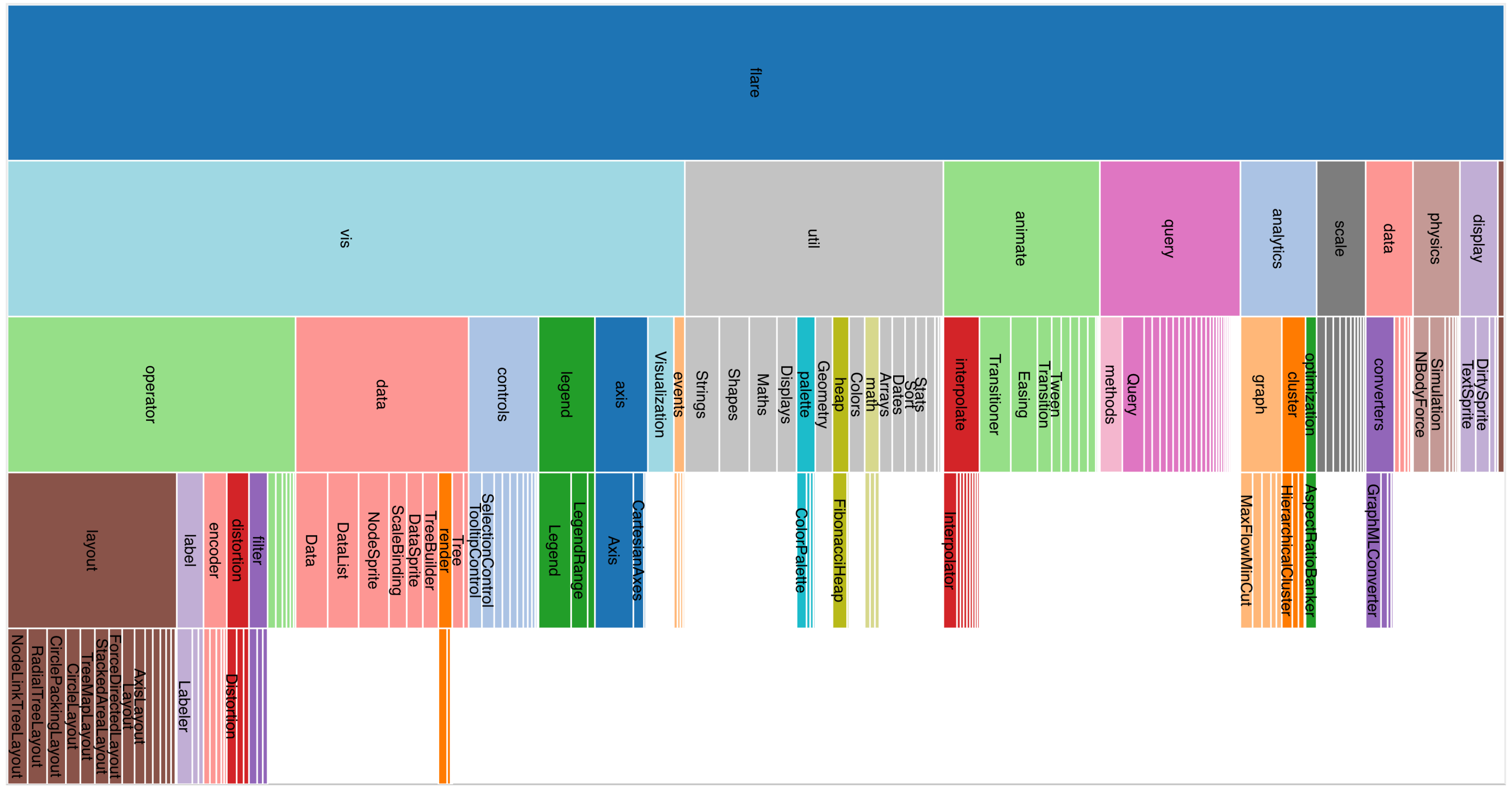
Treemap



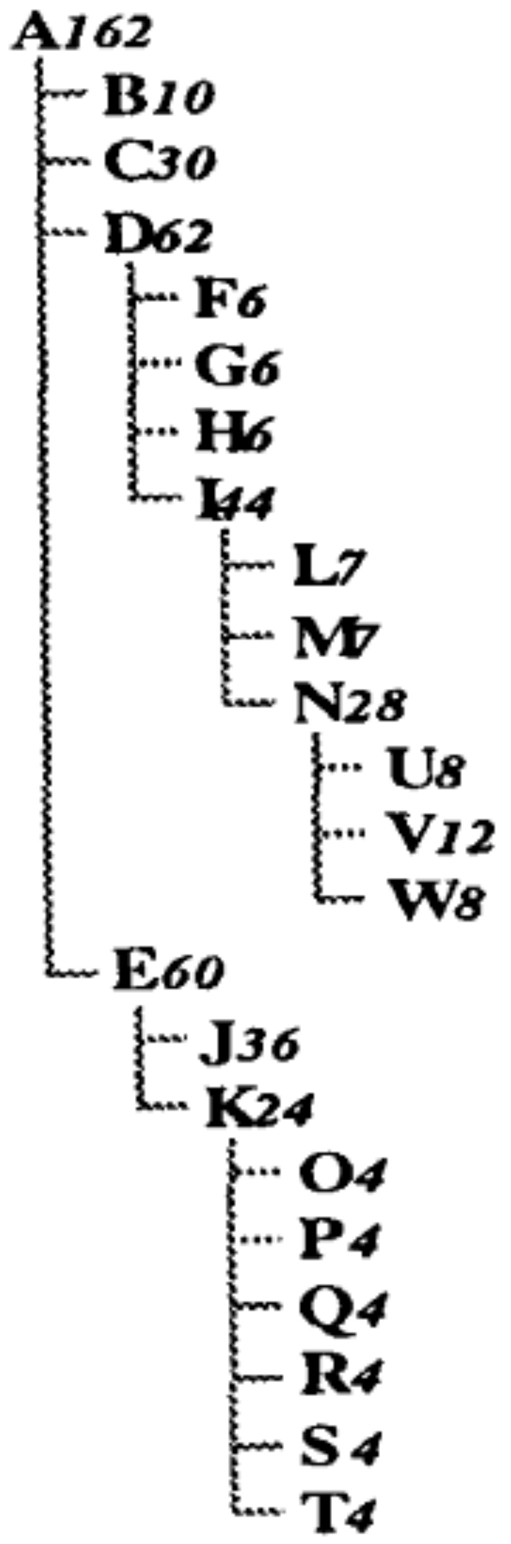
Sunburst

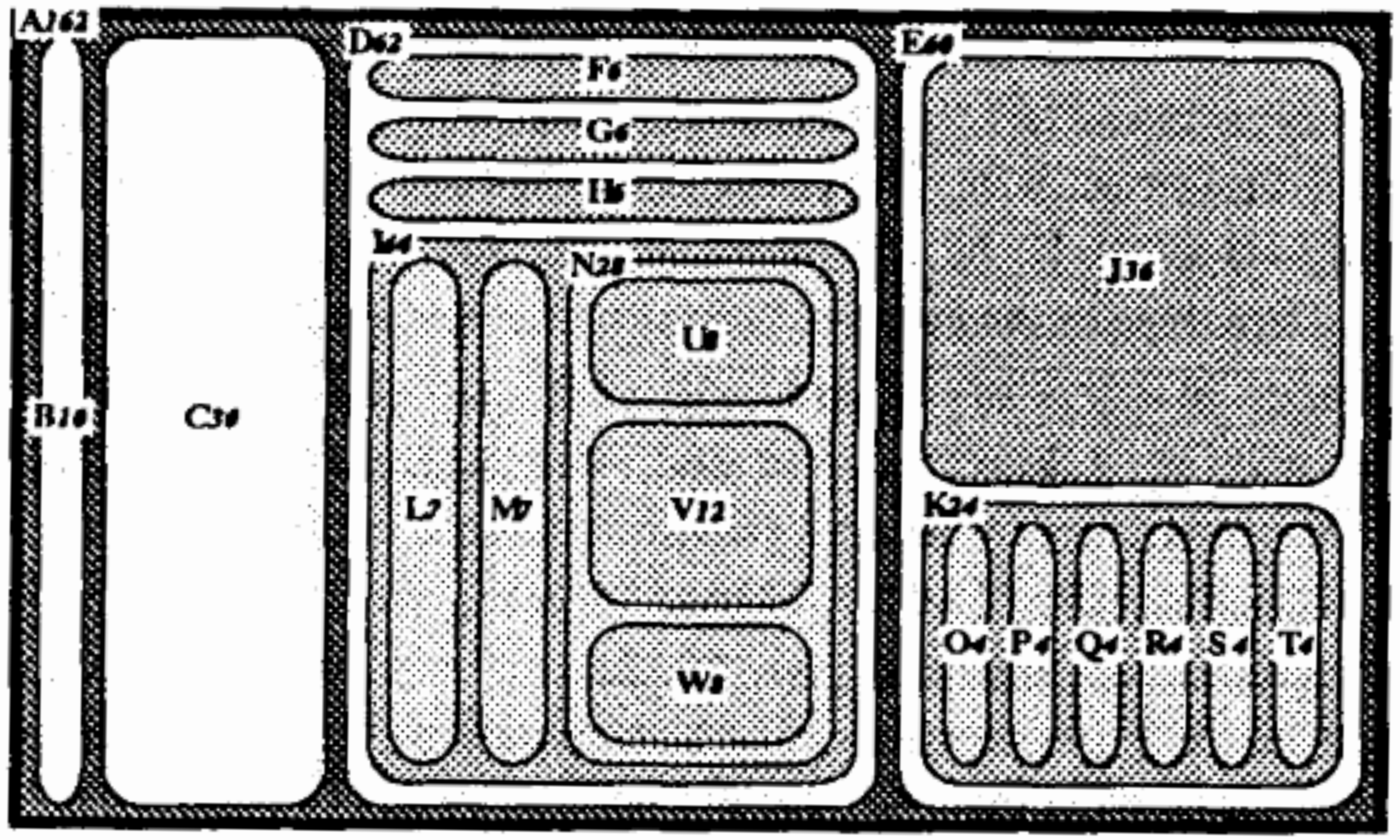


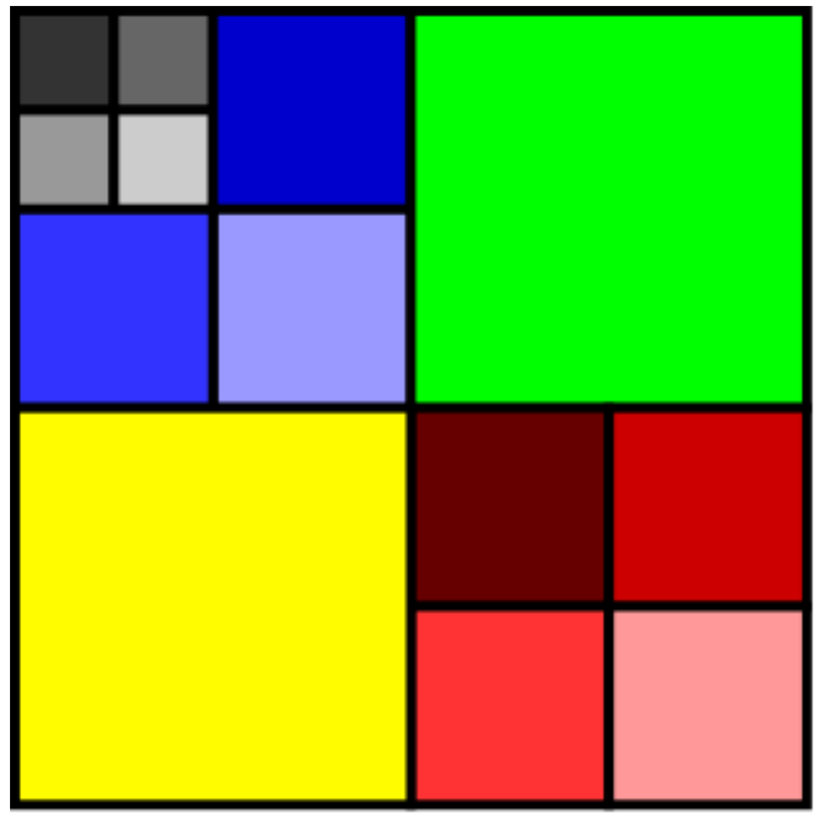
Icicle Plot

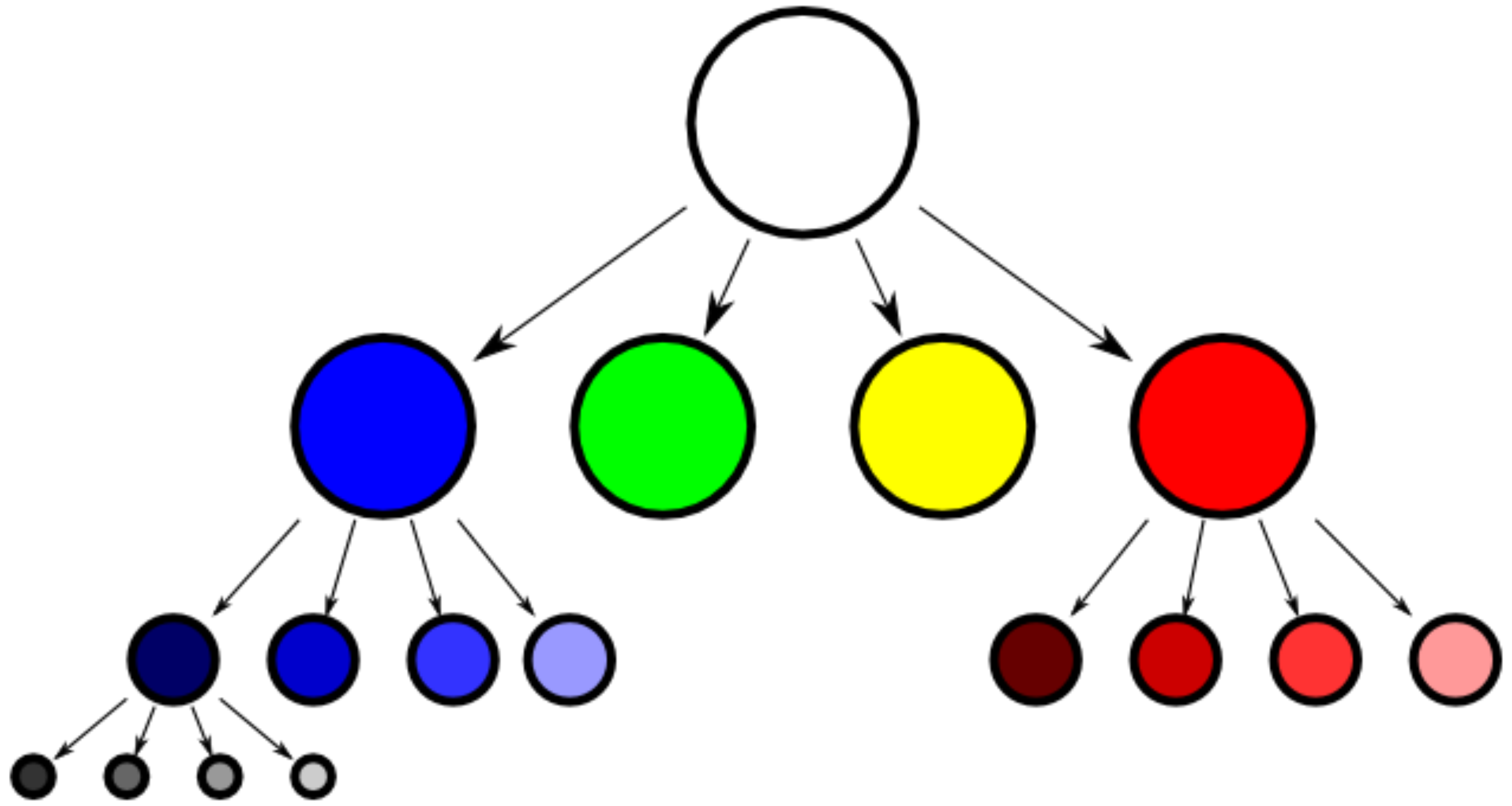


Tree Maps



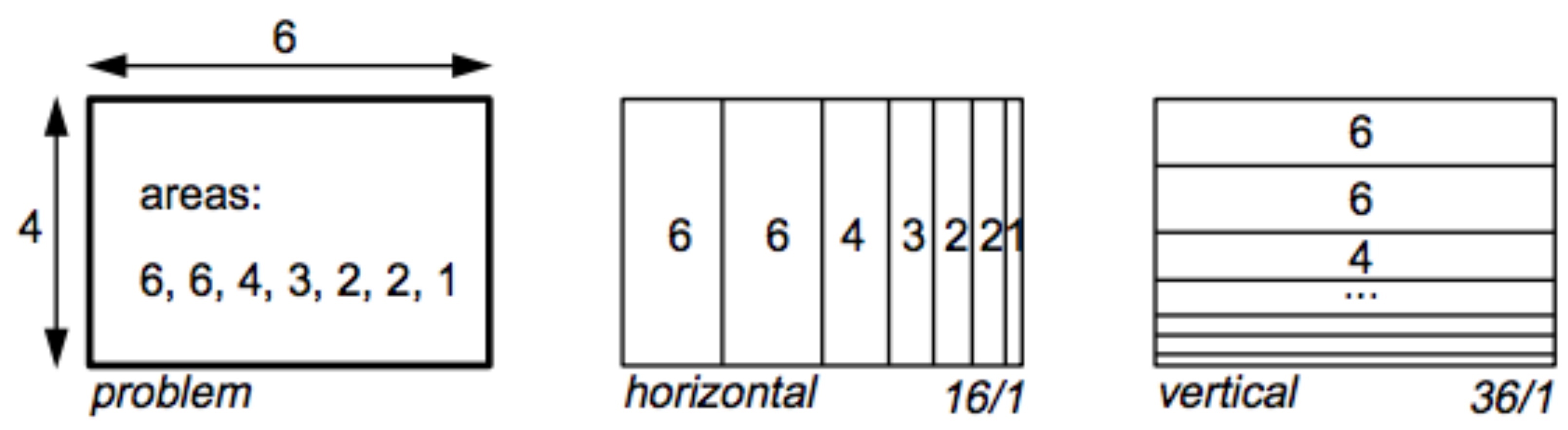






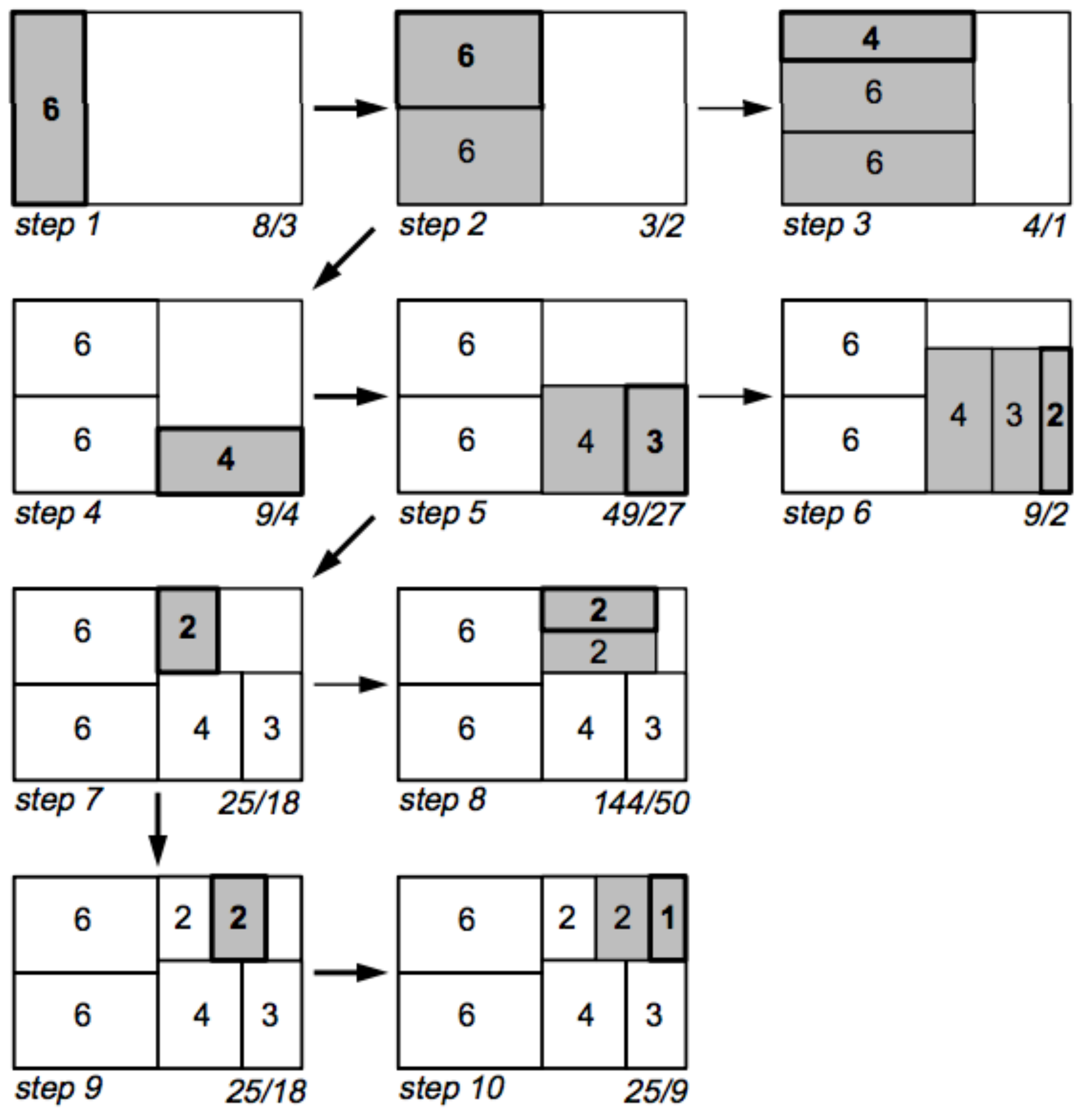
Johnson and Shneiderman 1991

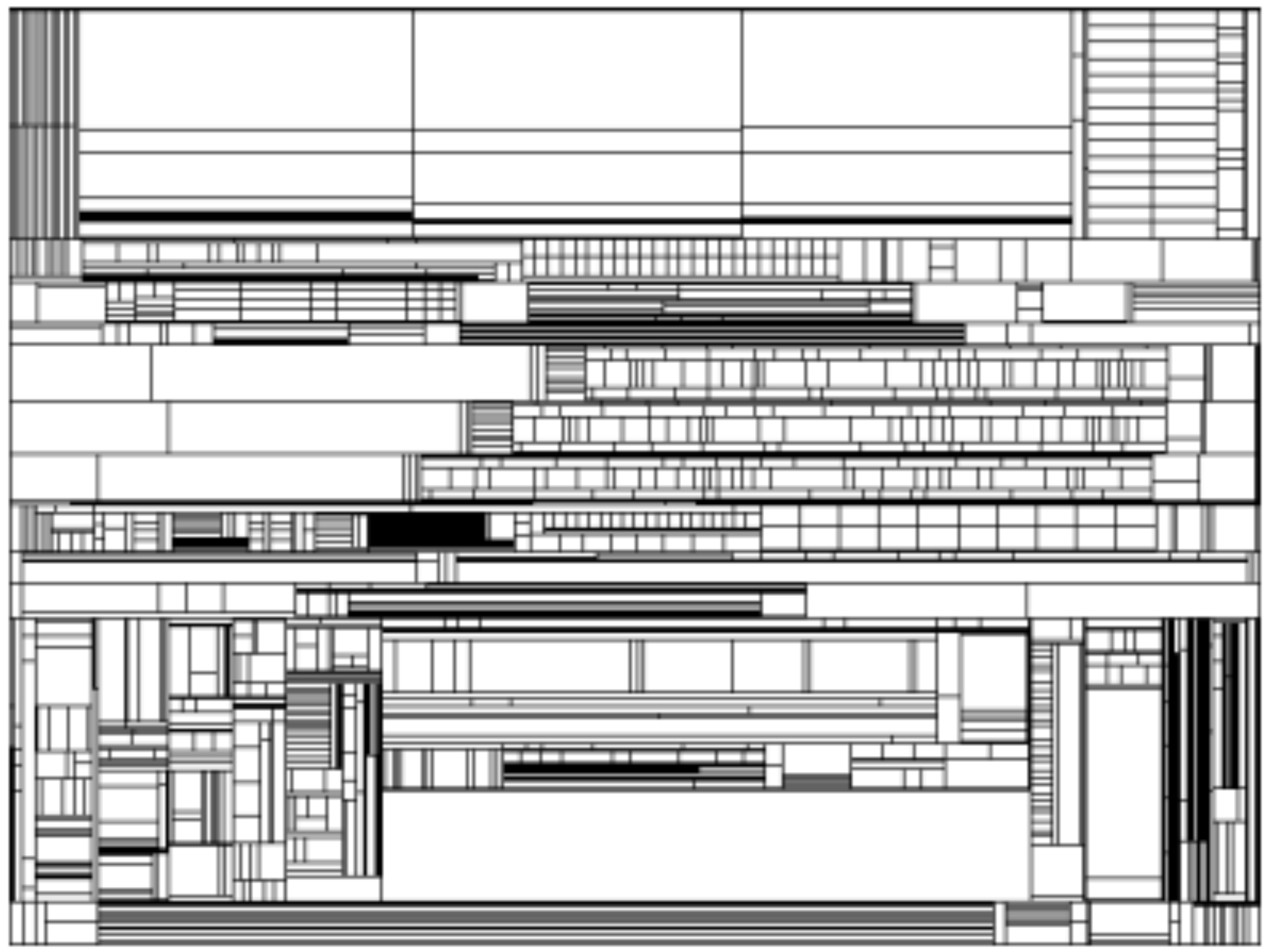
Squarified Treemaps



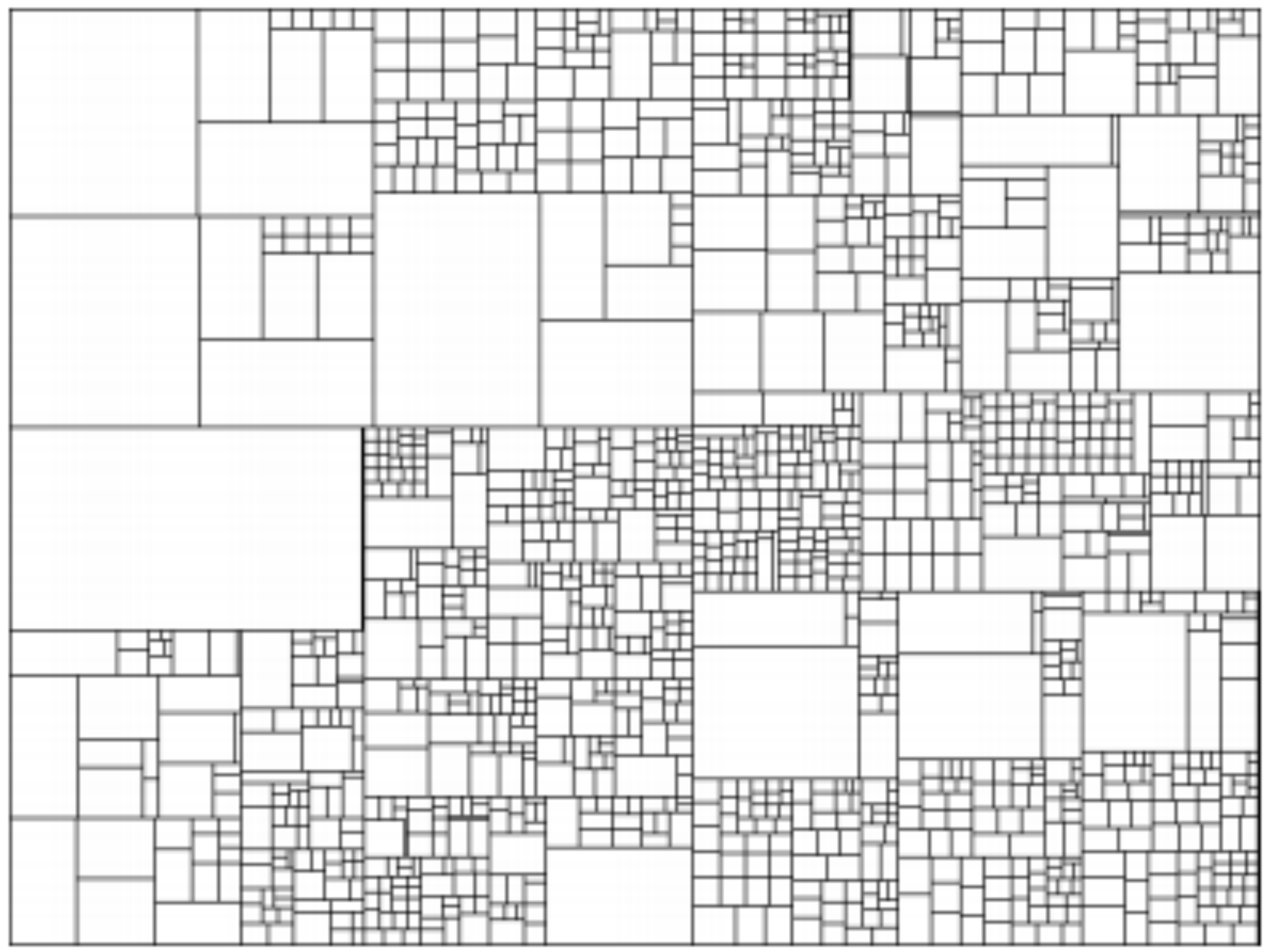
Original Algorithm lead to thin slices

Squarified treemaps [Bruls, Huizing, Van Wijk 2000]



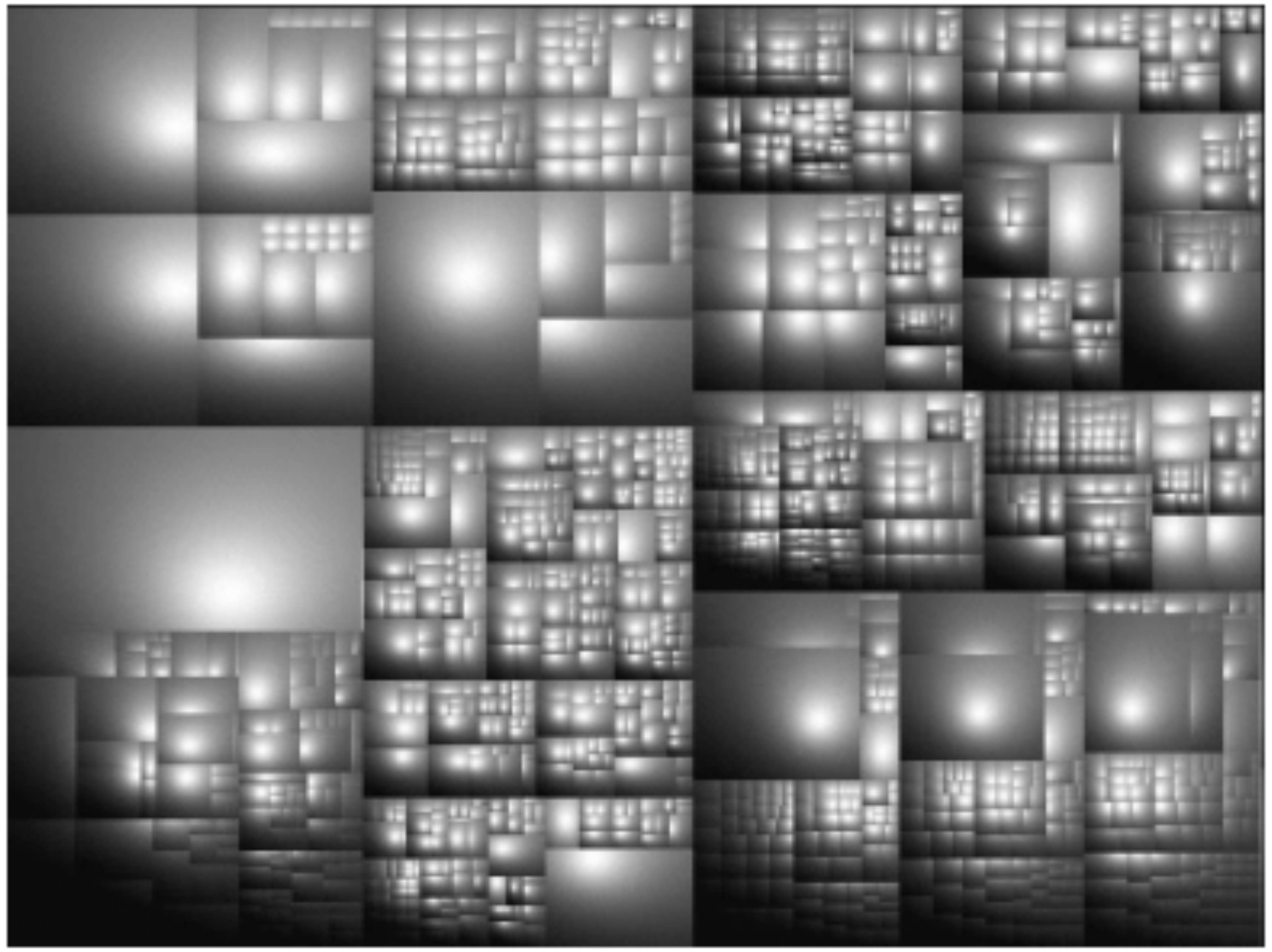


Before

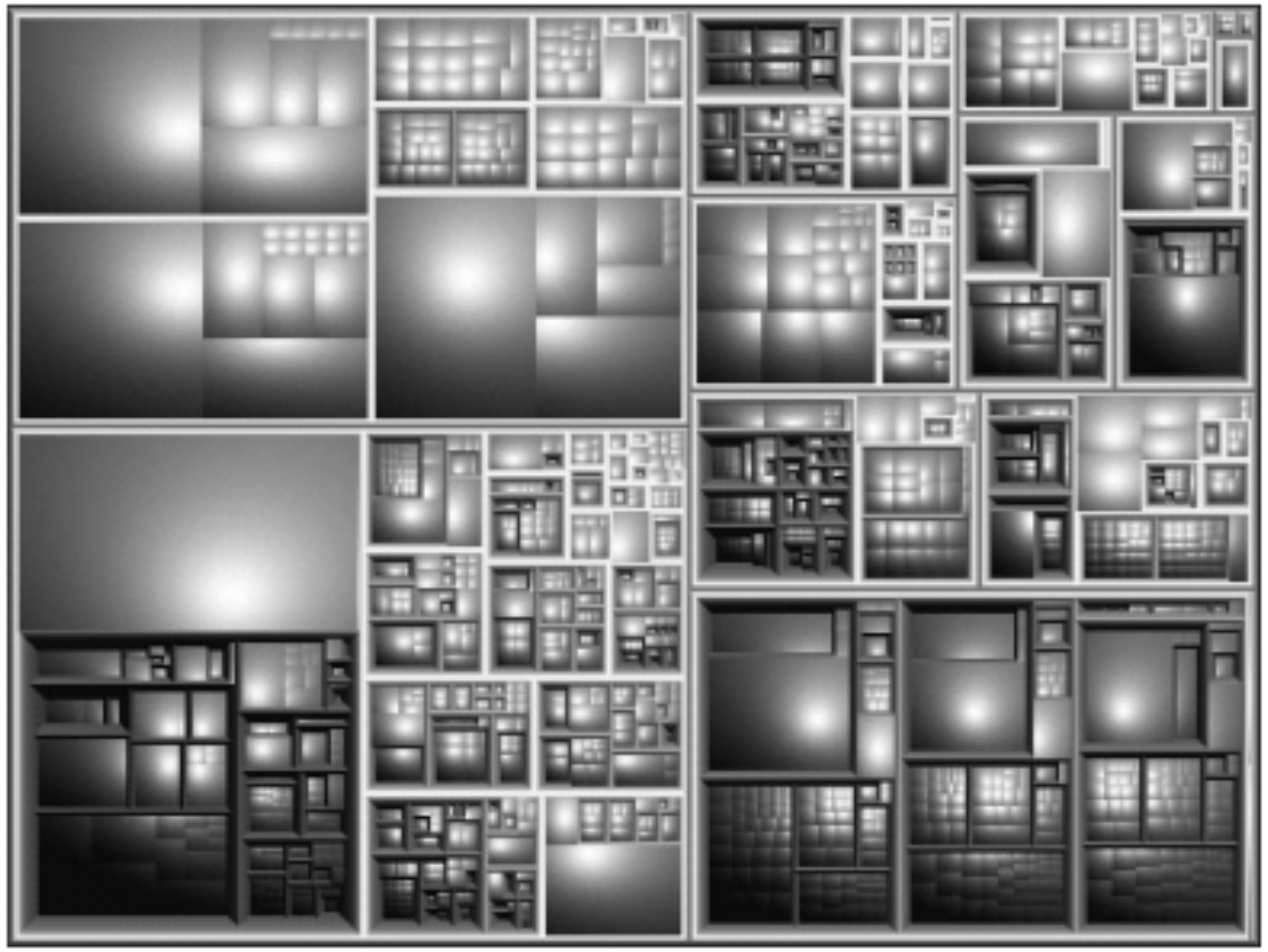


After

Seeing Tree Structure

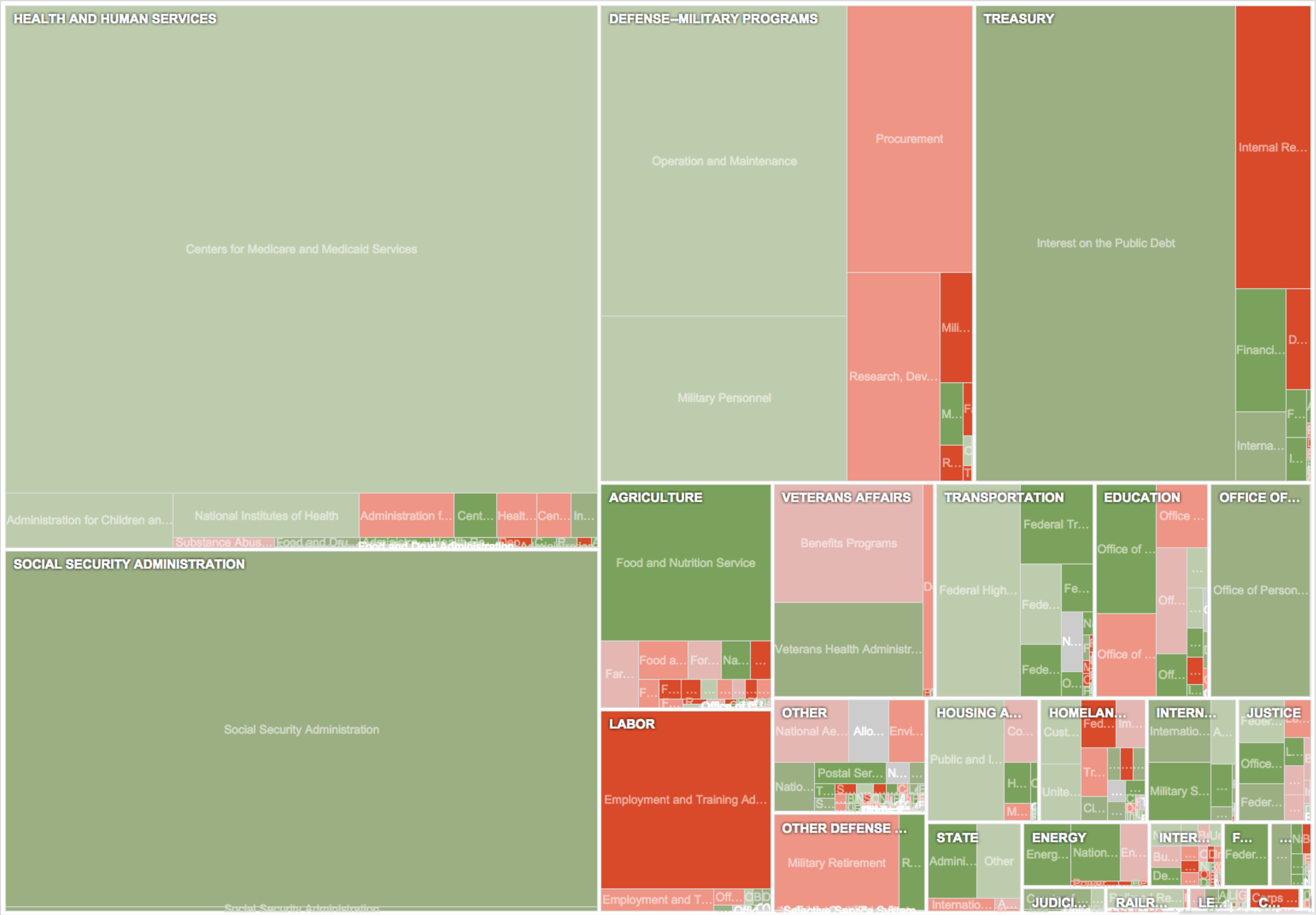


Unframed



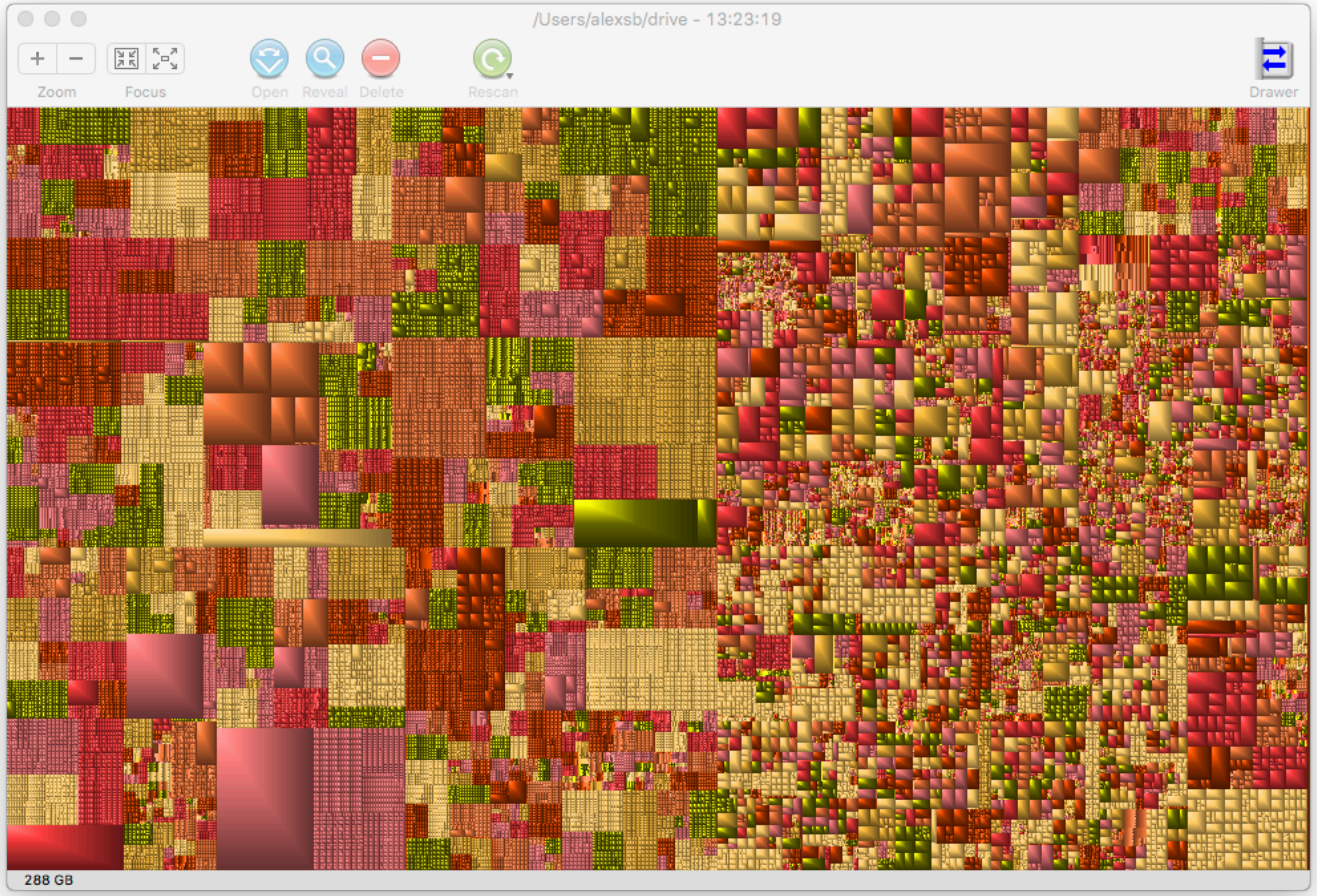
Framed

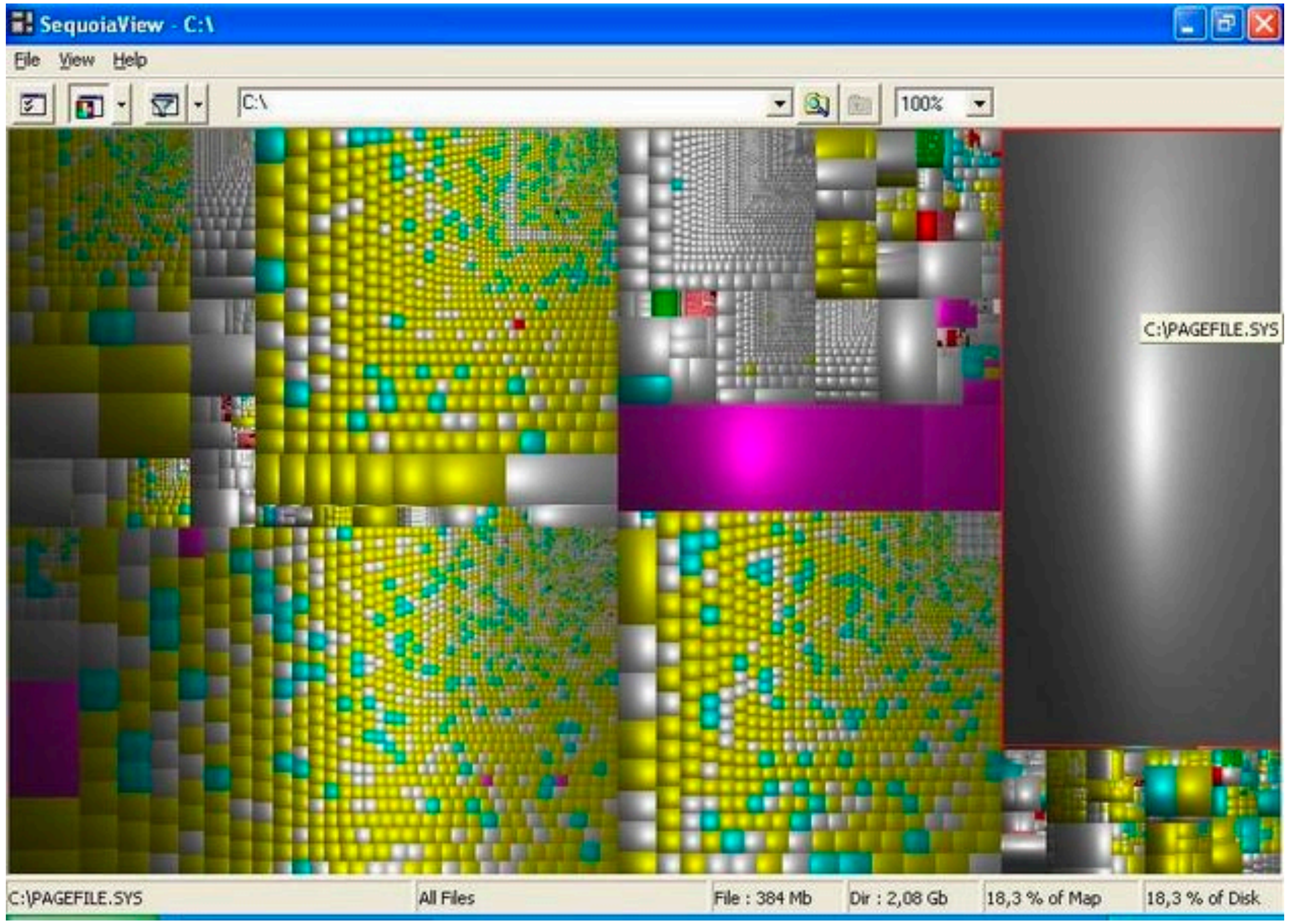
Zoomable Treemap



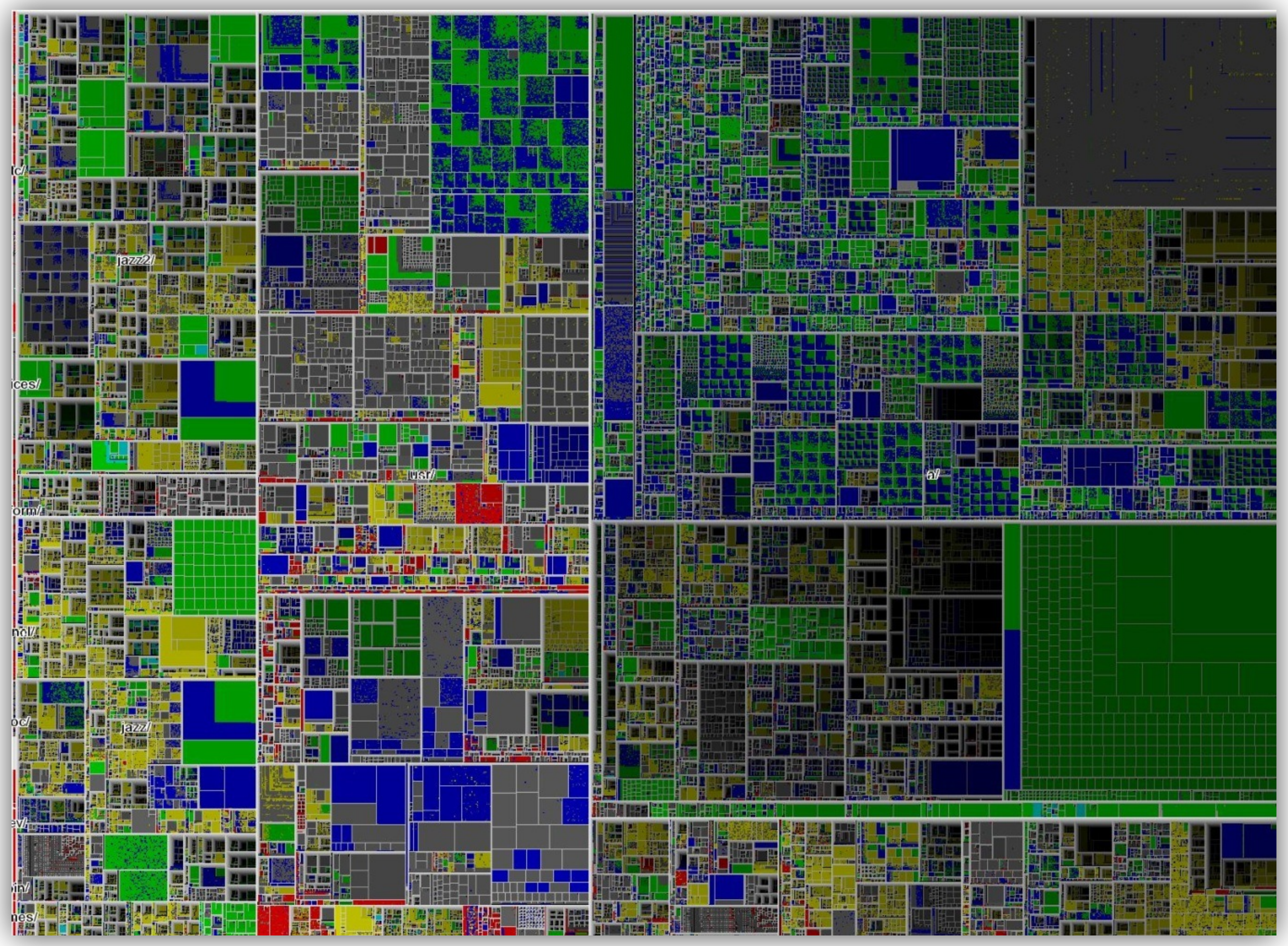
Software

Mac: GrandPerspective   Windows: Sequoia View





Example: Interactive TreeMap of a Million Items

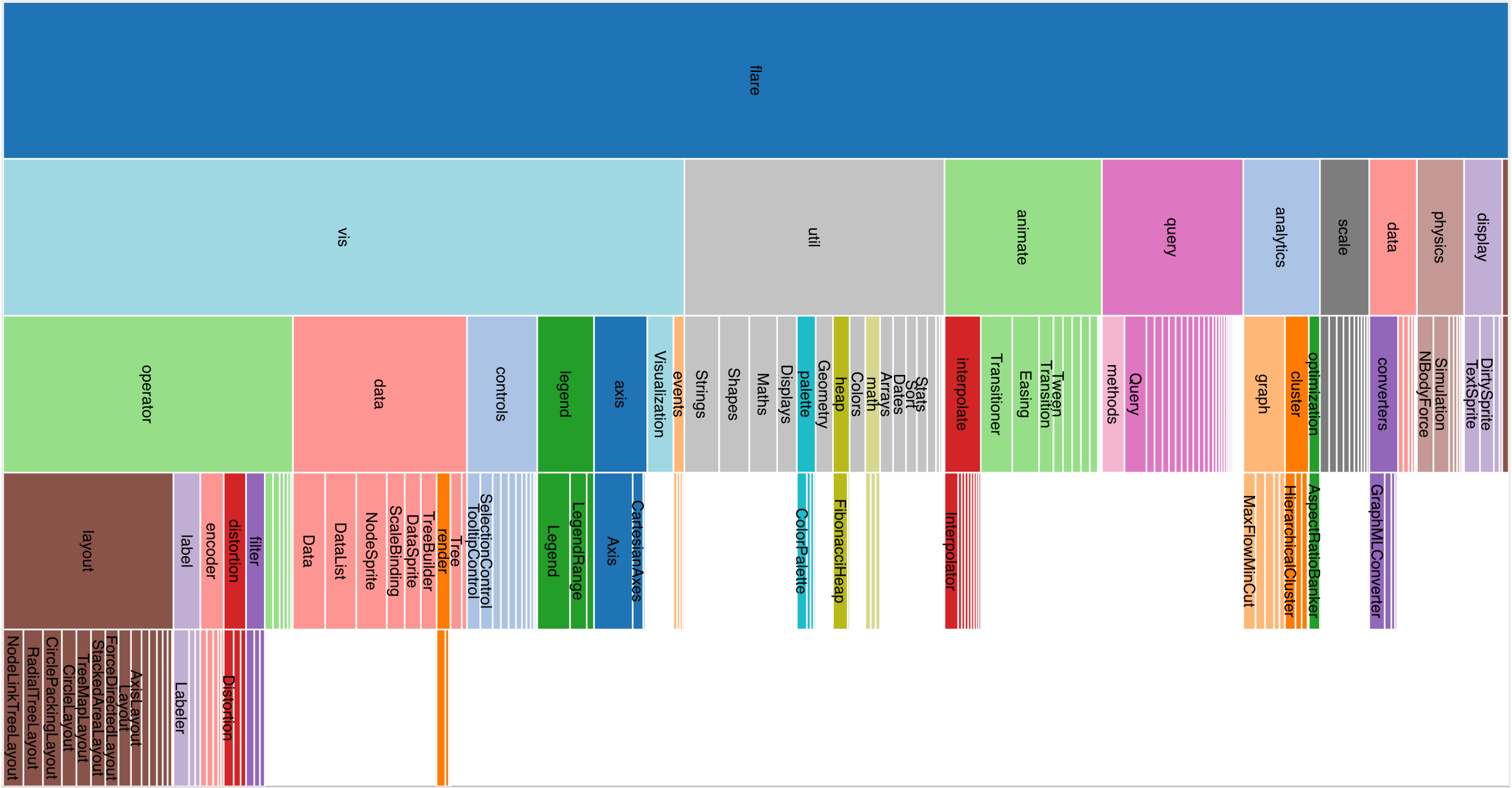


Fekete et al. 2002

Sunburst: Radial Layout

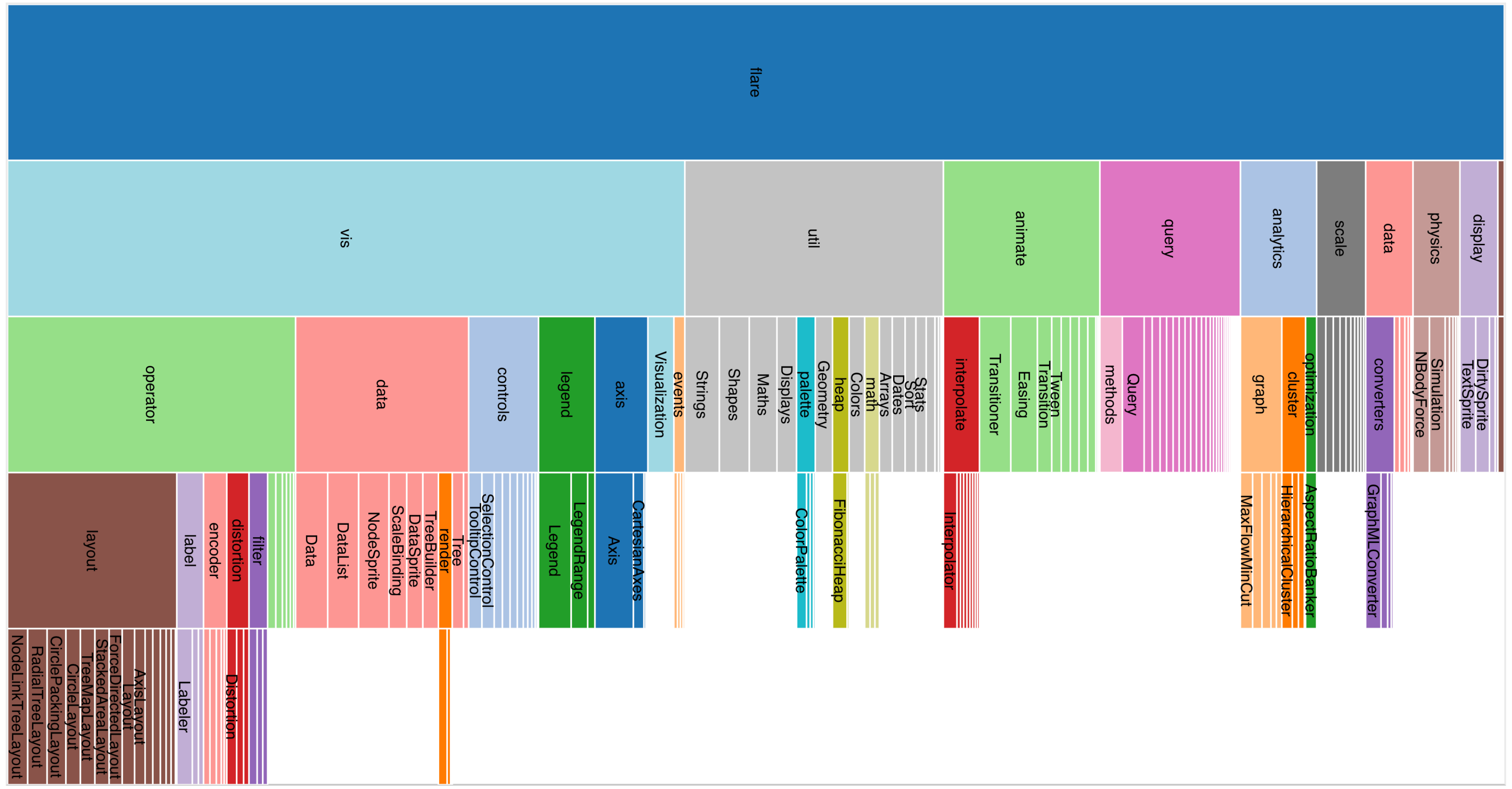
[Sunburst by John Stasko, Implementation in Caleydo by Christian Partl]

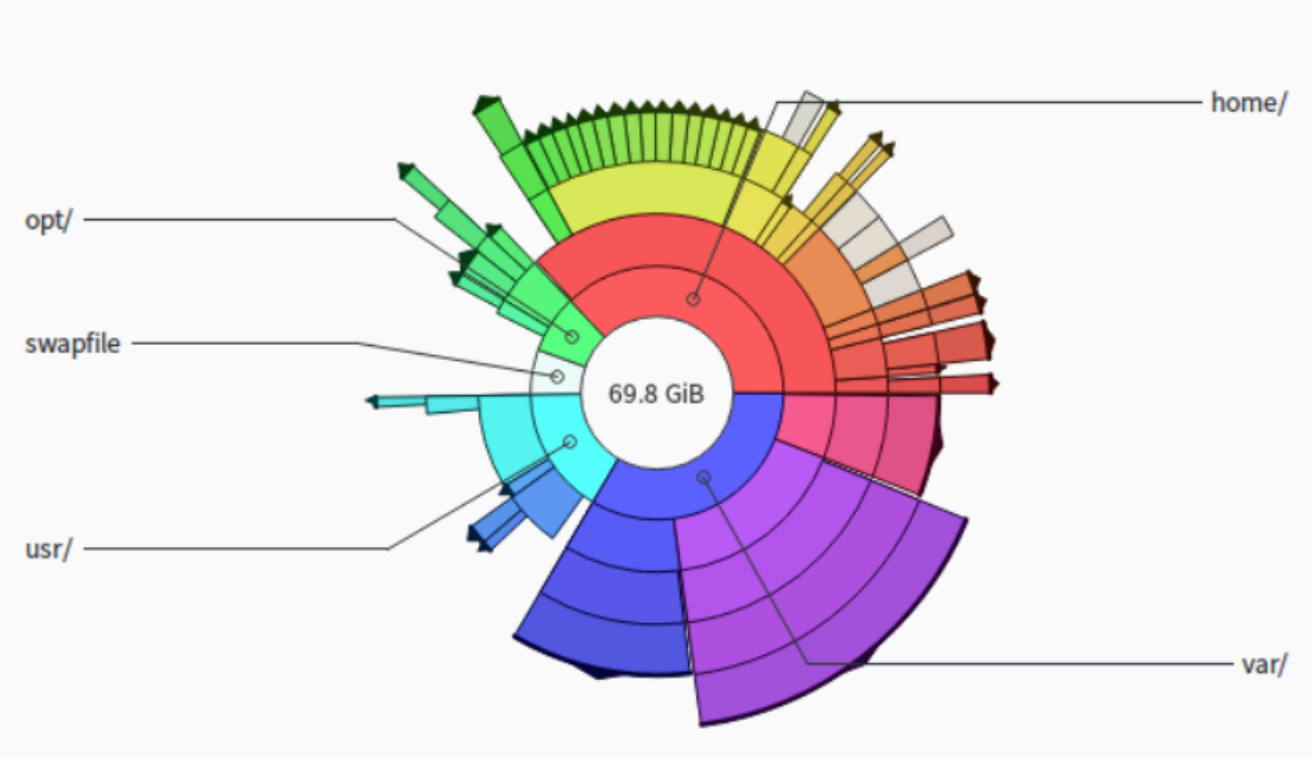
Icicle Plot

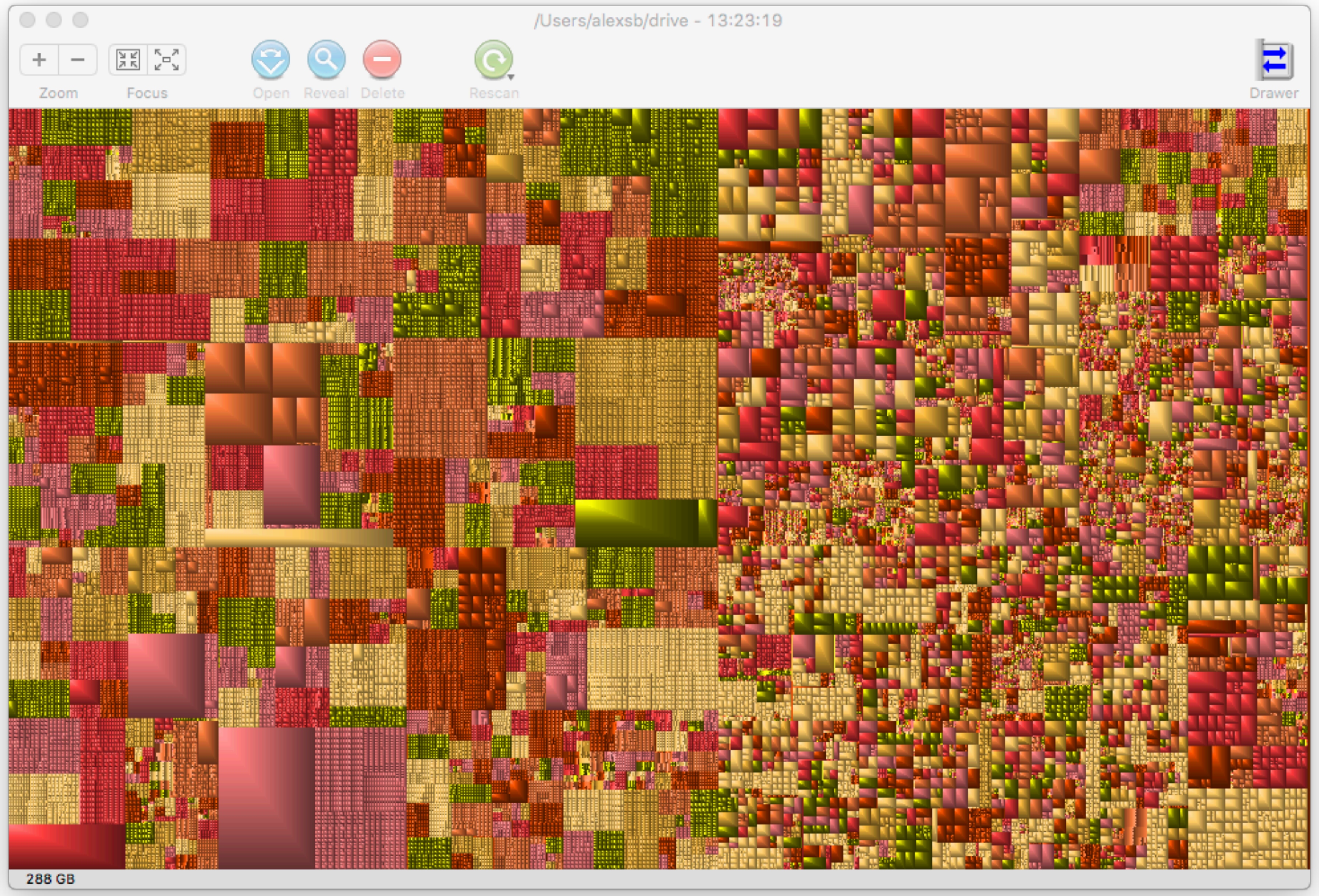


https://bl.ocks.org/mbostock/1005873

Differences? Pros, Cons?







Implicit Representations

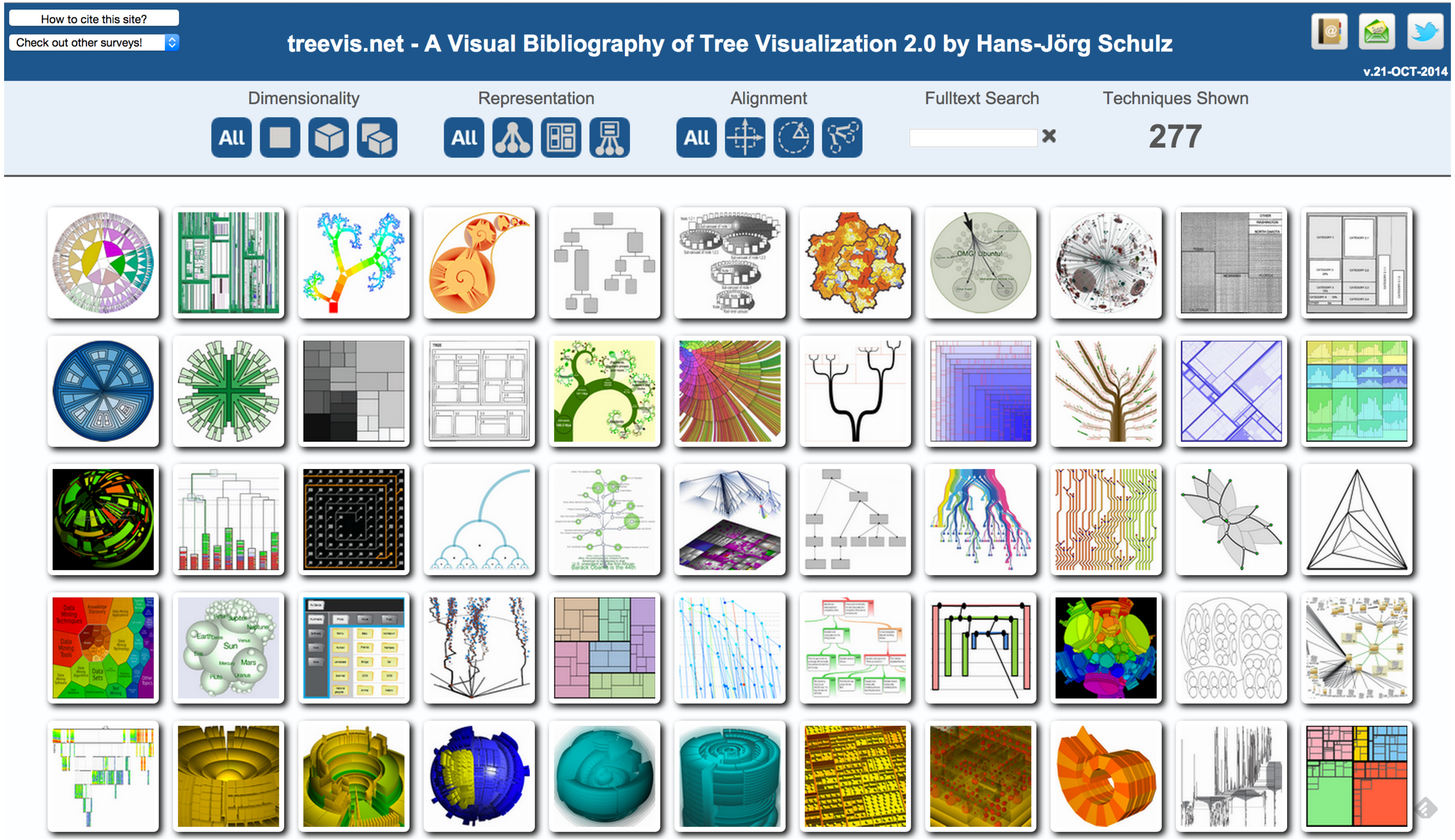
Pros:

- 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

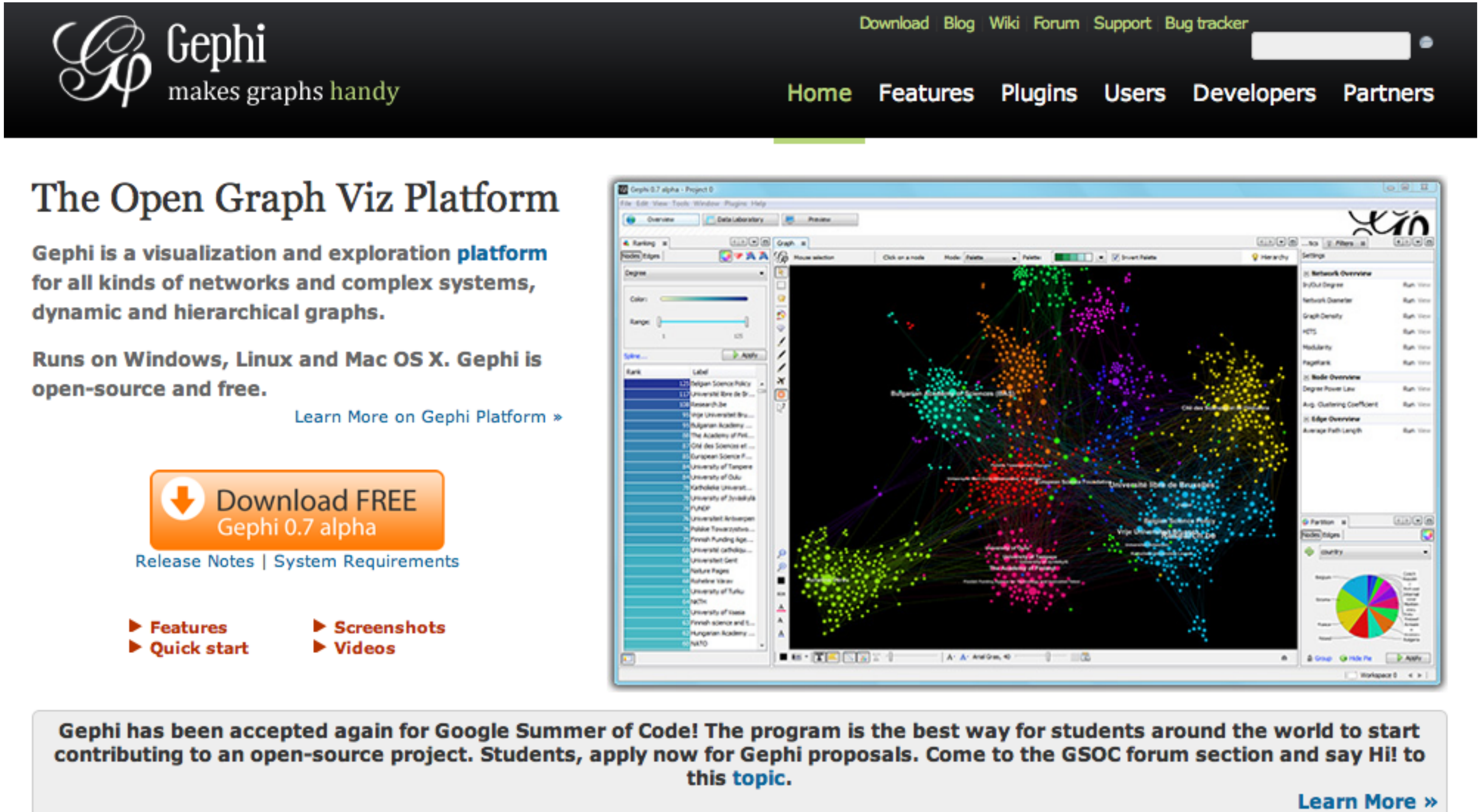
Tree Visualization Reference



Graph Tools & Applications

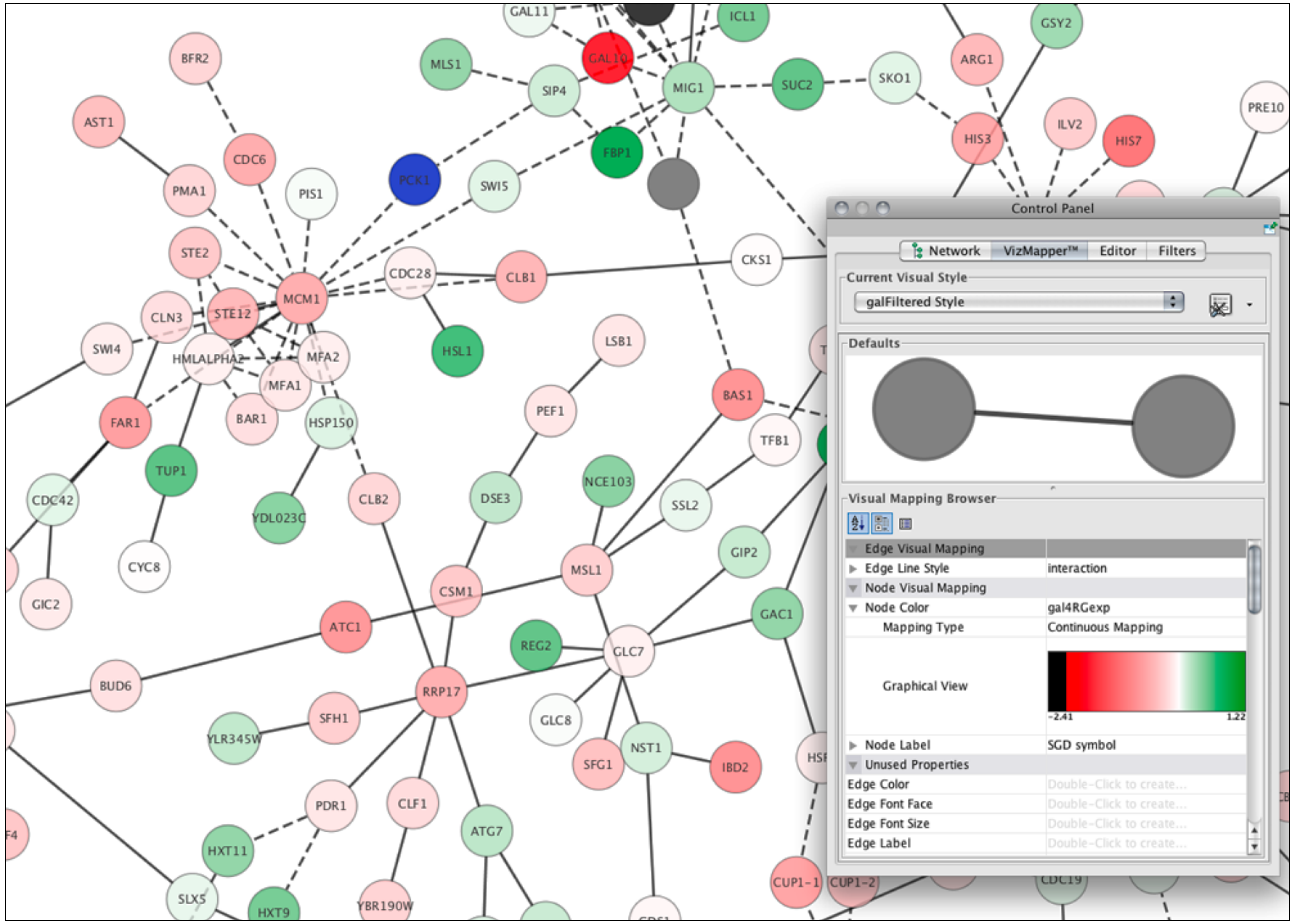
Gephi

http://gephi.org



Cytoscape

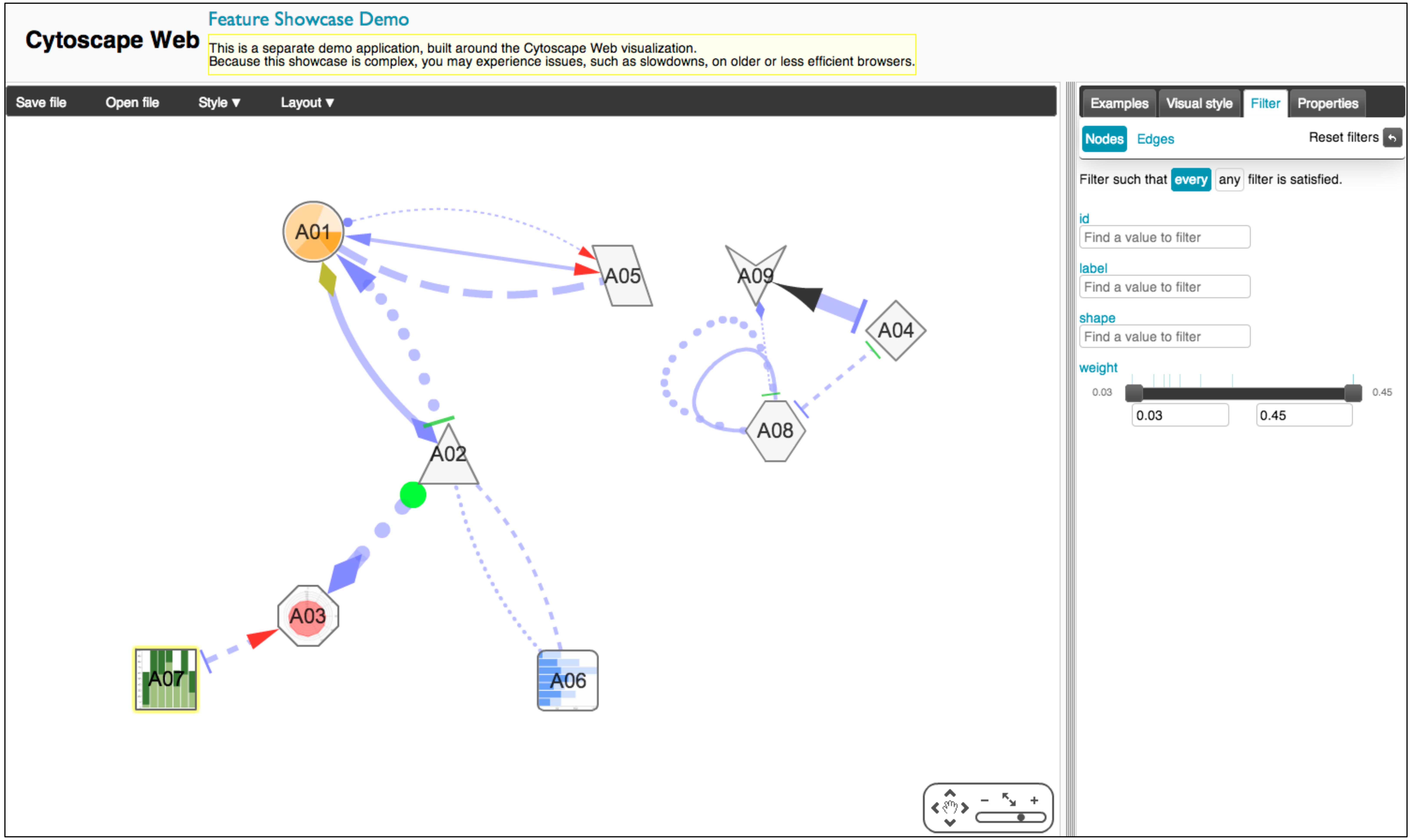
Open source platform for complex network analysis



http://www.cytoscape.org/

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