VIS EXAMPLES

FOR
MULTIVARIATE NETWORKS

CMPT 733

STEVEN BERGNER

SURVEY PAPER AND TUTORIAL BY
CAROLINA NOBRE, MARC STREIT, ALEXANDER LEX

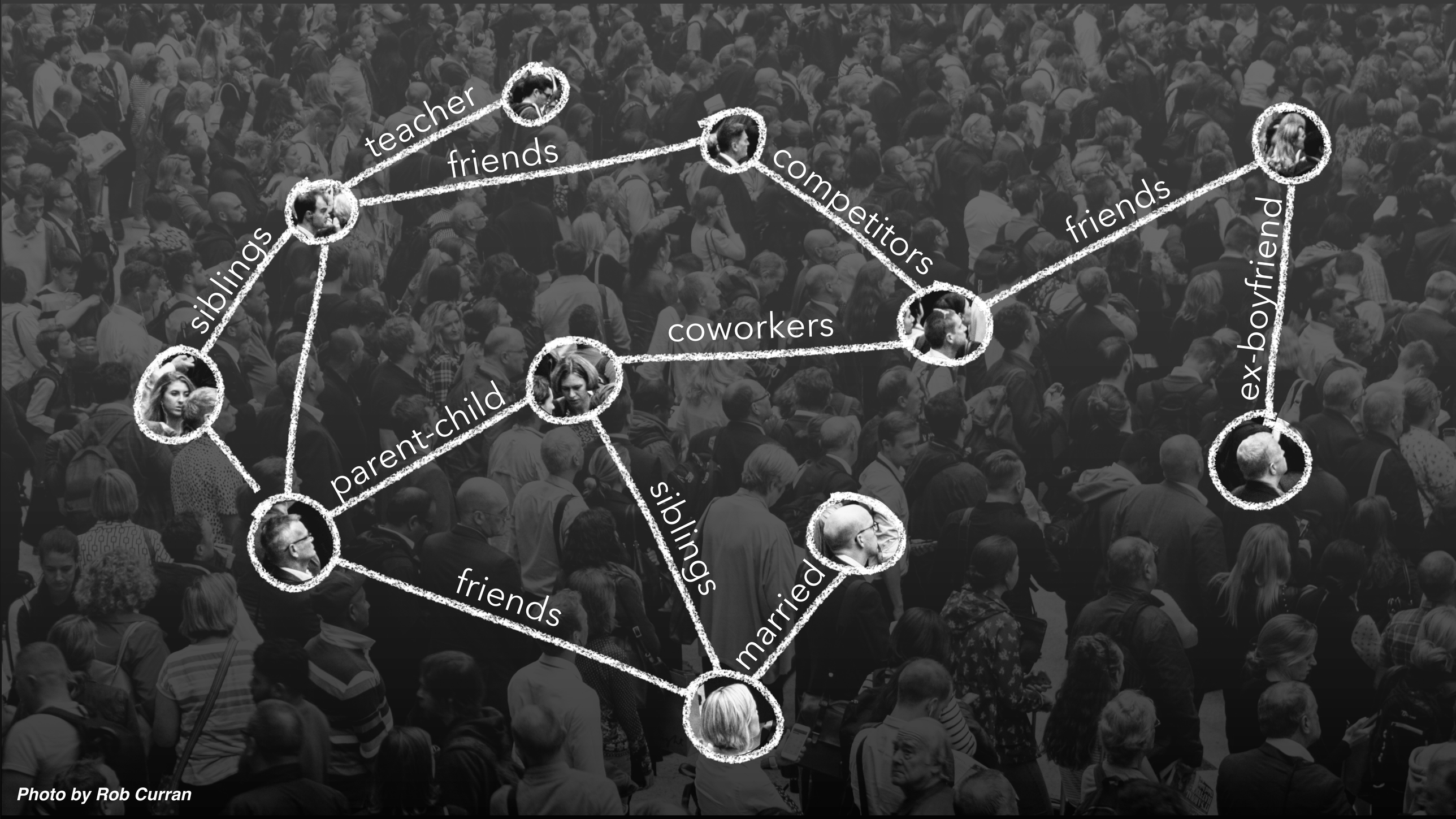


Photo by Rob Curran

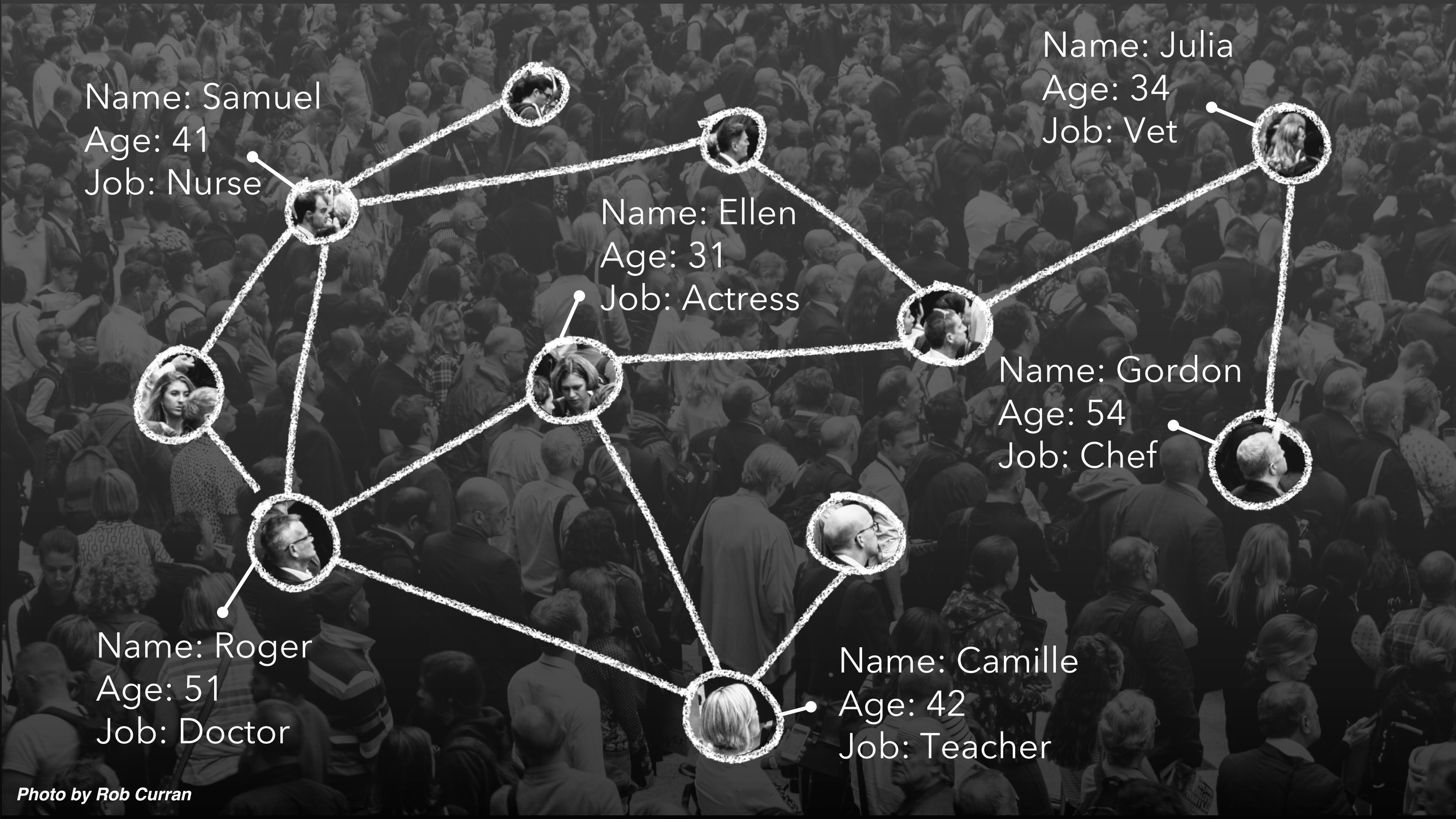


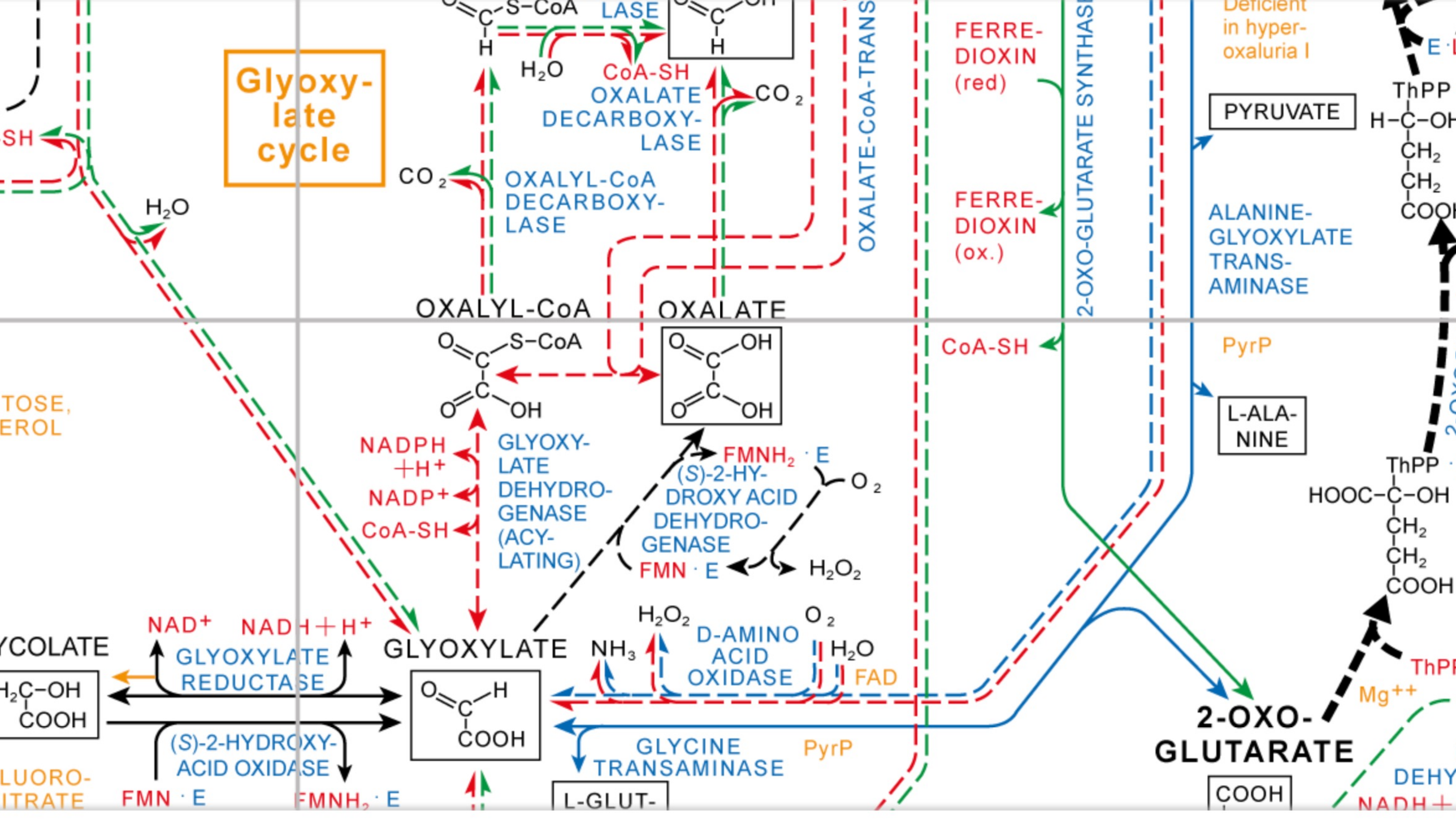
Photo by Rob Curran

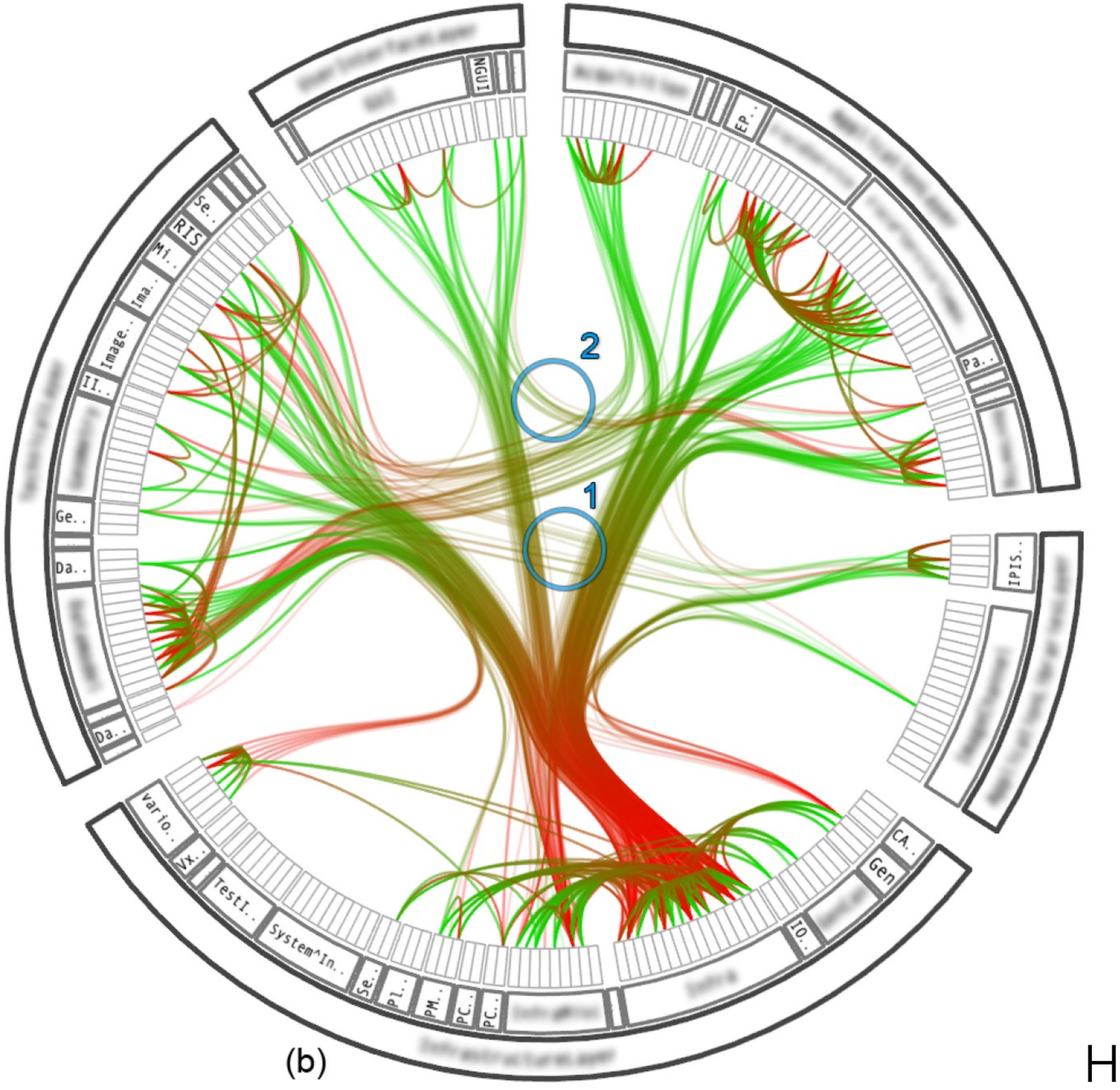
A MULTIVARIATE NETWORK IS

NETWORK TOPOLOGY +

NODE AND EDGE ATTRIBUTES


foursquare
LIVEJOURNAL
vimeo
YouTube
Insta
facebook
my Blogger
twitter
flickr
Like
tumblr.


Glyoxy-
late
cycle
LASE
H_2O
CoA-SH
OXALATE
DECARBOXY-
LASE
CO_2
CO_2
OXALYL-CoA
DECARBOXY-
LASE
H_2O
OXALATE-CoA-TRANS
FERRE-
DIOXIN
(red)
FERRE-
DIOXIN
(ox.)
2-OXO-GLUTARATE SYNTHASE
Deficient
in hyper-
oxaluria I
PYRUVATE
ALANINE-
GLYOXYLATE
TRANS-
AMINASE
ThPP
OXALYL-CoA
OXALATE
CoA-SH
PyrP
L-ALA-
NINE
TOSE,
EROL
NADPH
$+H^+$
$NADP^+$
CoA-SH
GLYOXY-
LATE
DEHYDRO-
GENASE
(ACY-
LATING)
$FMNH_2 \cdot E$
(S)-2-HY-
DROXY ACID
DEHYDRO-
GENASE
$FMN \cdot E$
O_2
H_2O_2
NAD^+
$NADH+H^+$
GLYOXYLATE
REDUCTASE
YCOLATE
GLYOXYLATE
NH_3
H_2O_2
D-AMINO
ACID
OXIDASE
O_2
H_2O
FAD
(S)-2-HYDROXY-
ACID OXIDASE
$FMN \cdot E$
$FMNH_2 \cdot E$
GLYCINE
TRANSAMINASE
PyrP
L-GLUT-
2-OXO-
GLUTARATE
COOH
Mg^{++}
ThPP
DEHY
$NADH+$

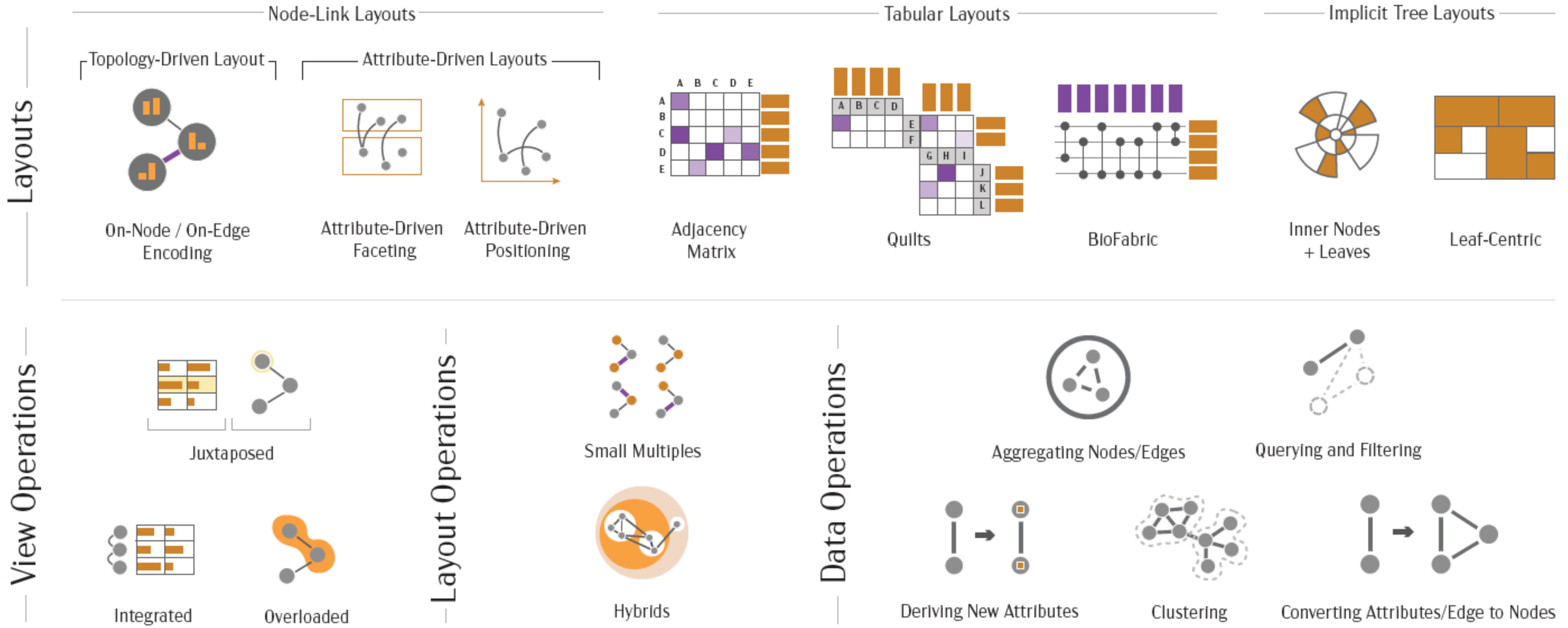


Holten and Wijk, 2009

The State of the Art in Visualizing Multivariate Networks

C. Nobre[1], M. Meyer[1], M. Streit[2], and A. Lex[1]

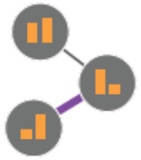
[1]University of Utah, Utah, USA
[2]Johannes Kepler University Linz, Austria



Layouts

Node-Link Layouts

Topology-Driven Layout



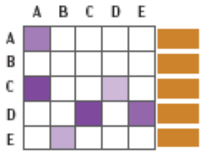
On-Node / On-Edge Encoding

Attribute-Driven Layouts

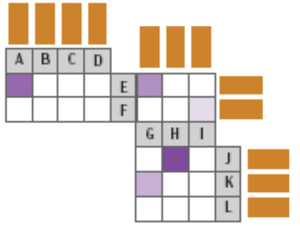
Attribute-Driven Faceting

Attribute-Driven Positioning

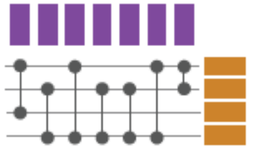
Tabular Layouts



Adjacency Matrix



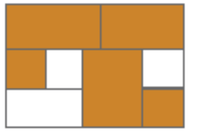
Quilts



BioFabric

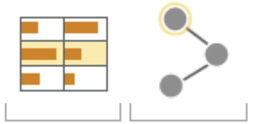
Implicit Tree Layouts

Inner Nodes + Leaves



Leaf-Centric

View Operations



Juxtaposed

Integrated

Overloaded

Layout Operations

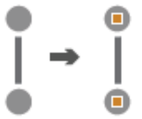
Small Multiples

Hybrids

Data Operations

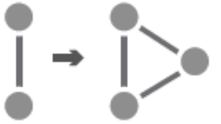
Aggregating Nodes/Edges

Querying and Filtering



Deriving New Attributes

Clustering

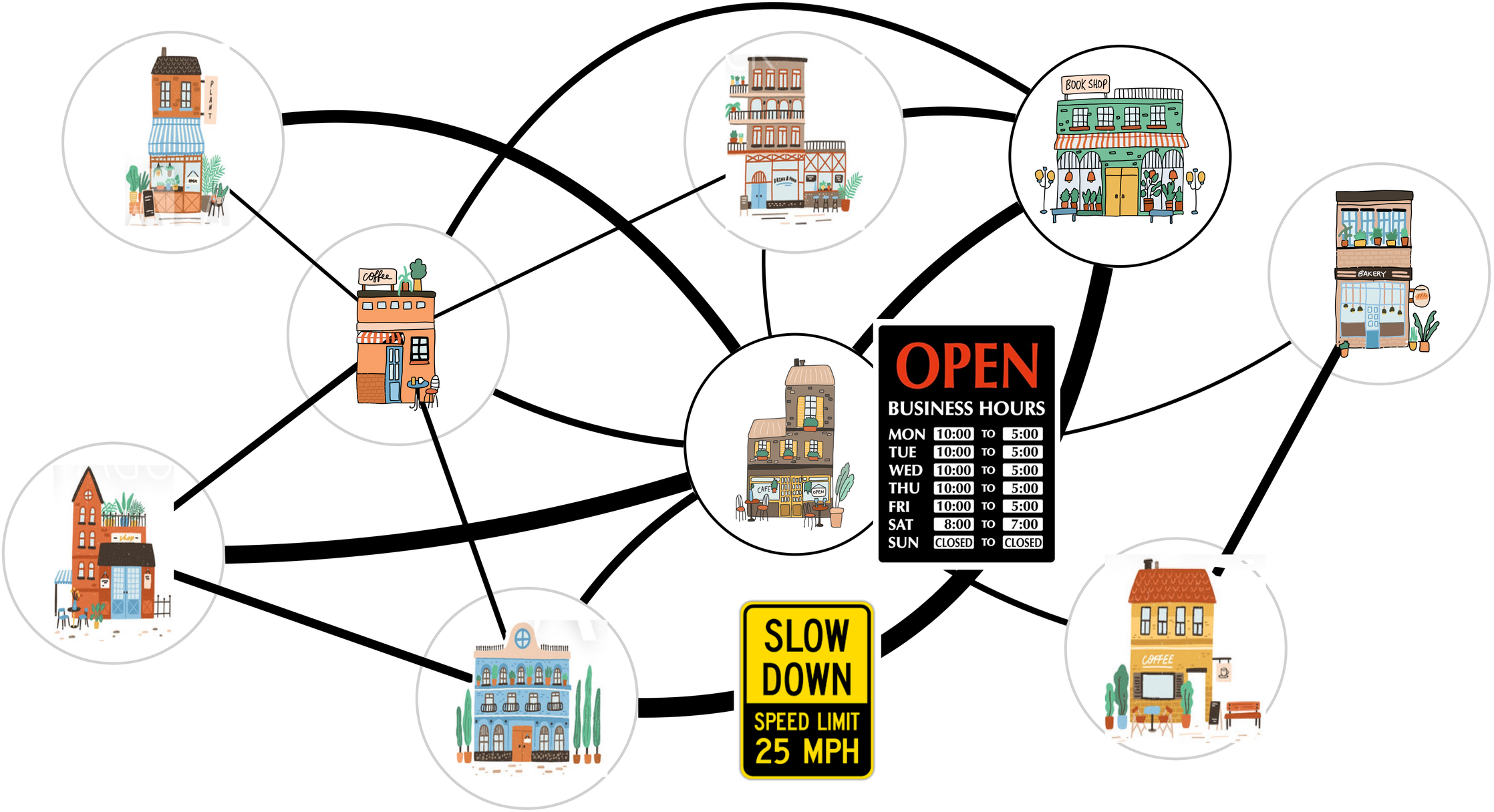


Converting Attributes/Edge to Nodes

MVNV Tasks

How is an MVN task different than a regular graph task?

MVN Tasks rely on both the topology of the network and the attributes of the nodes and edges



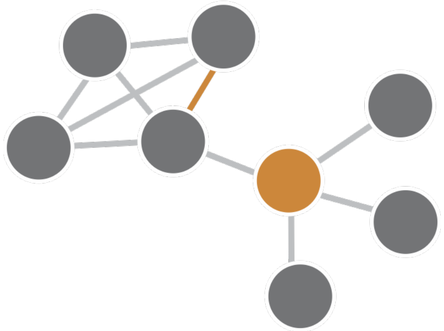
What is an efficient way I can complete all my errands?

- What is the fastest route to get all my errands done?

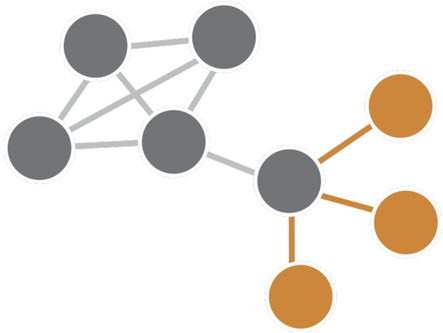
Tasks that rely on the **topology** of the network and the **attributes** of the nodes and edges

MVNV tasks are applied to topological structures

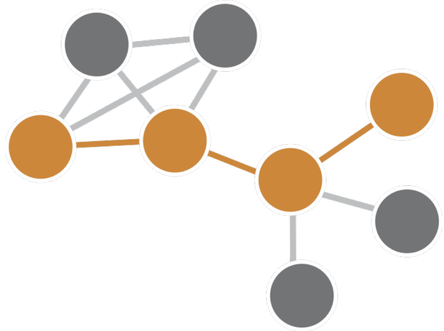
Single Node/Edge



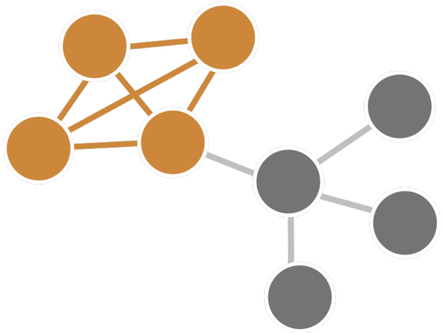
Node Neighbors



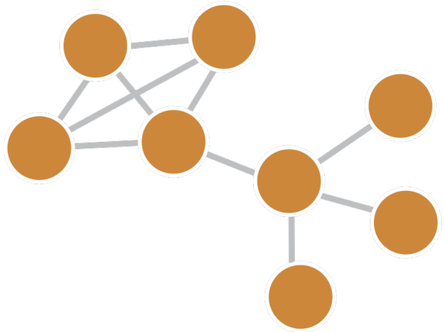
Path



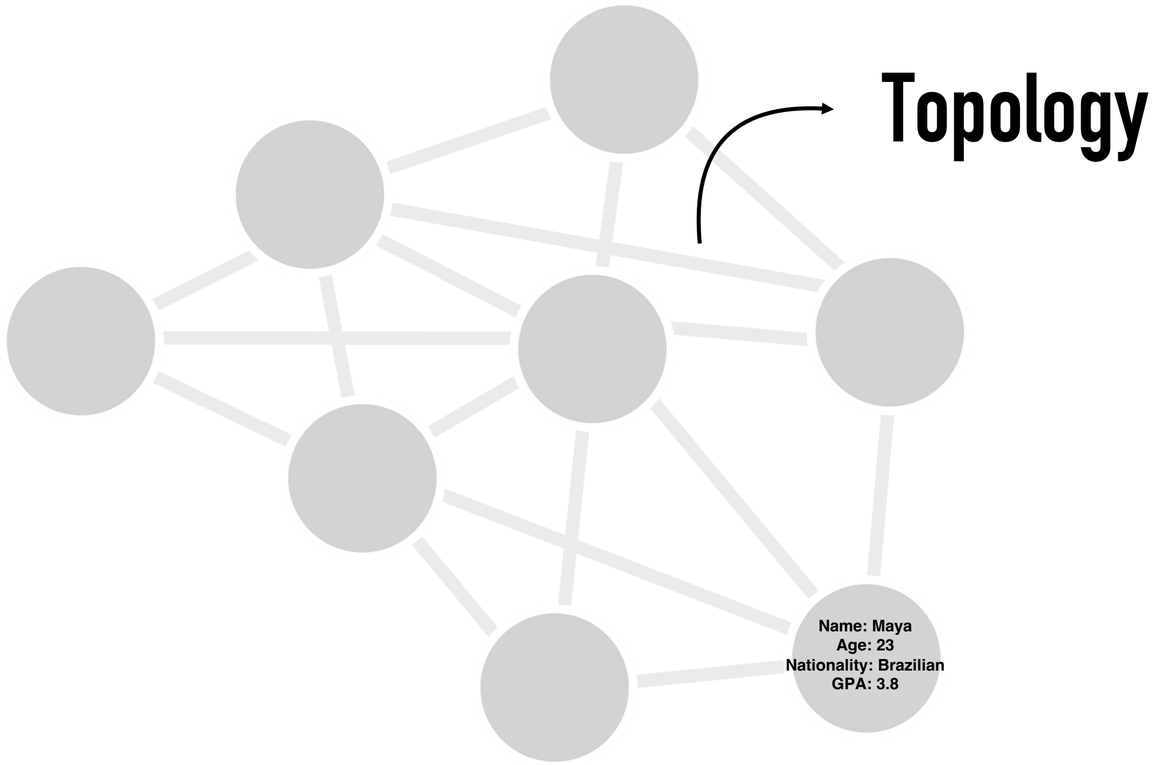
Cluster

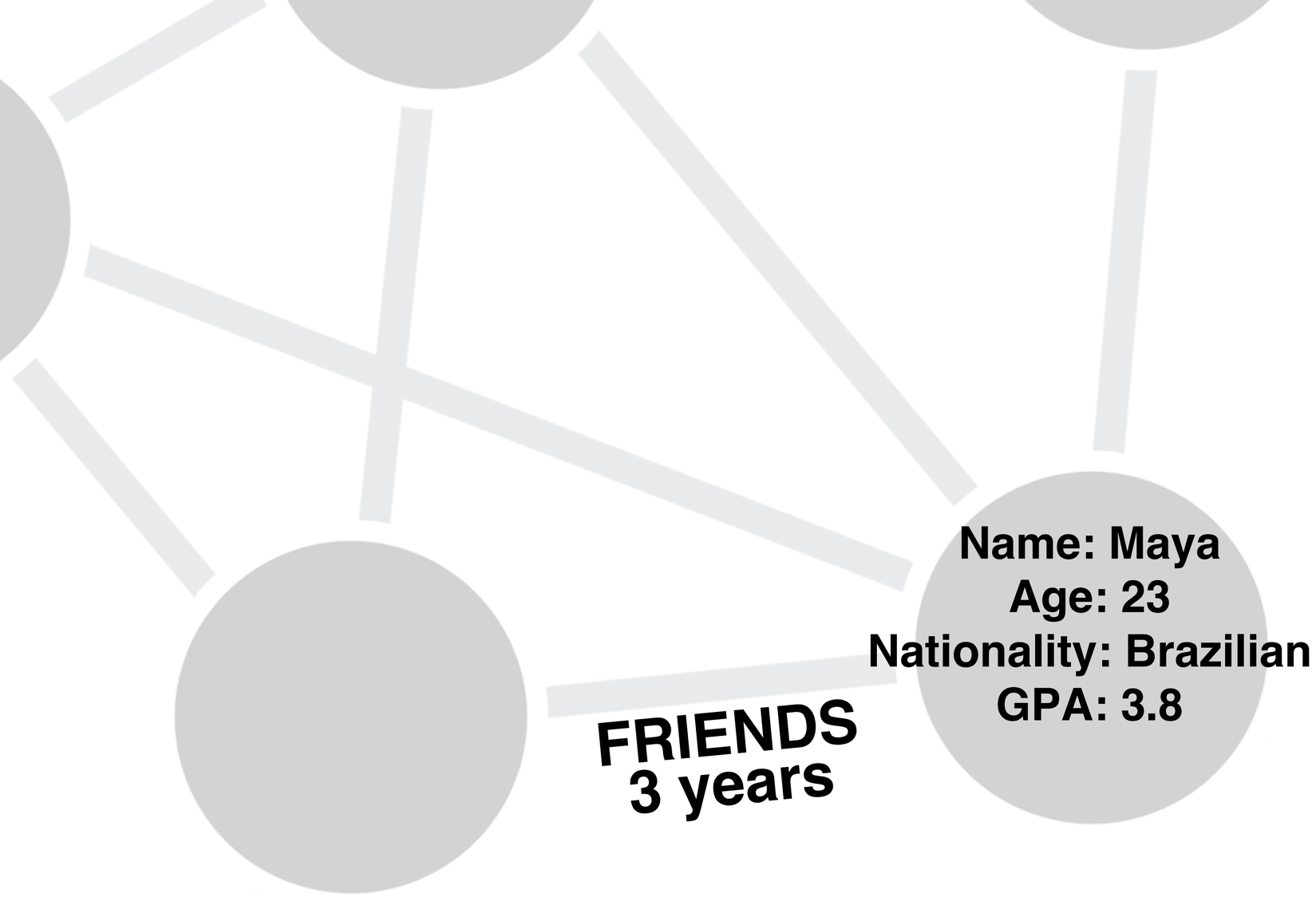


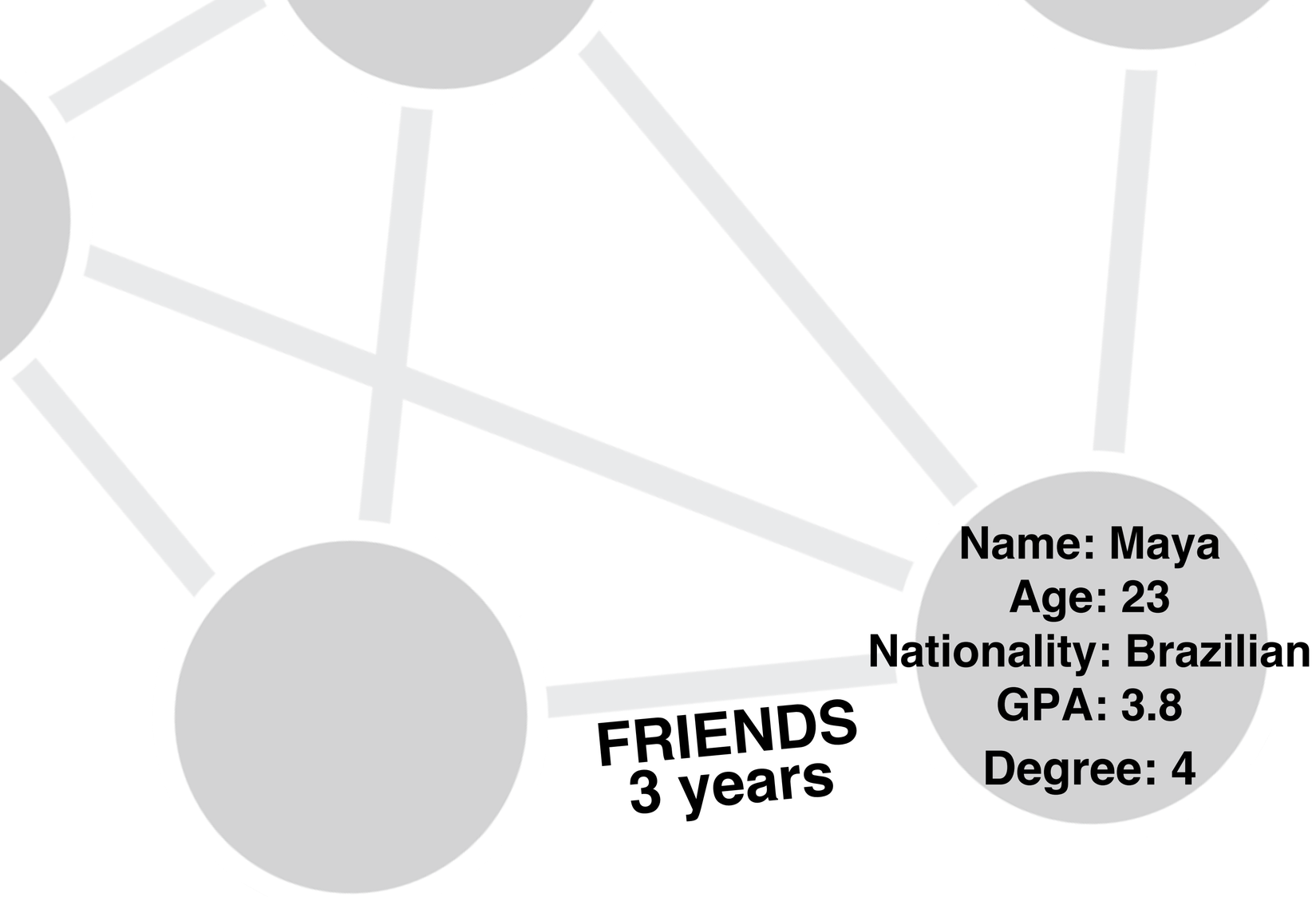
Network/Subnetwork

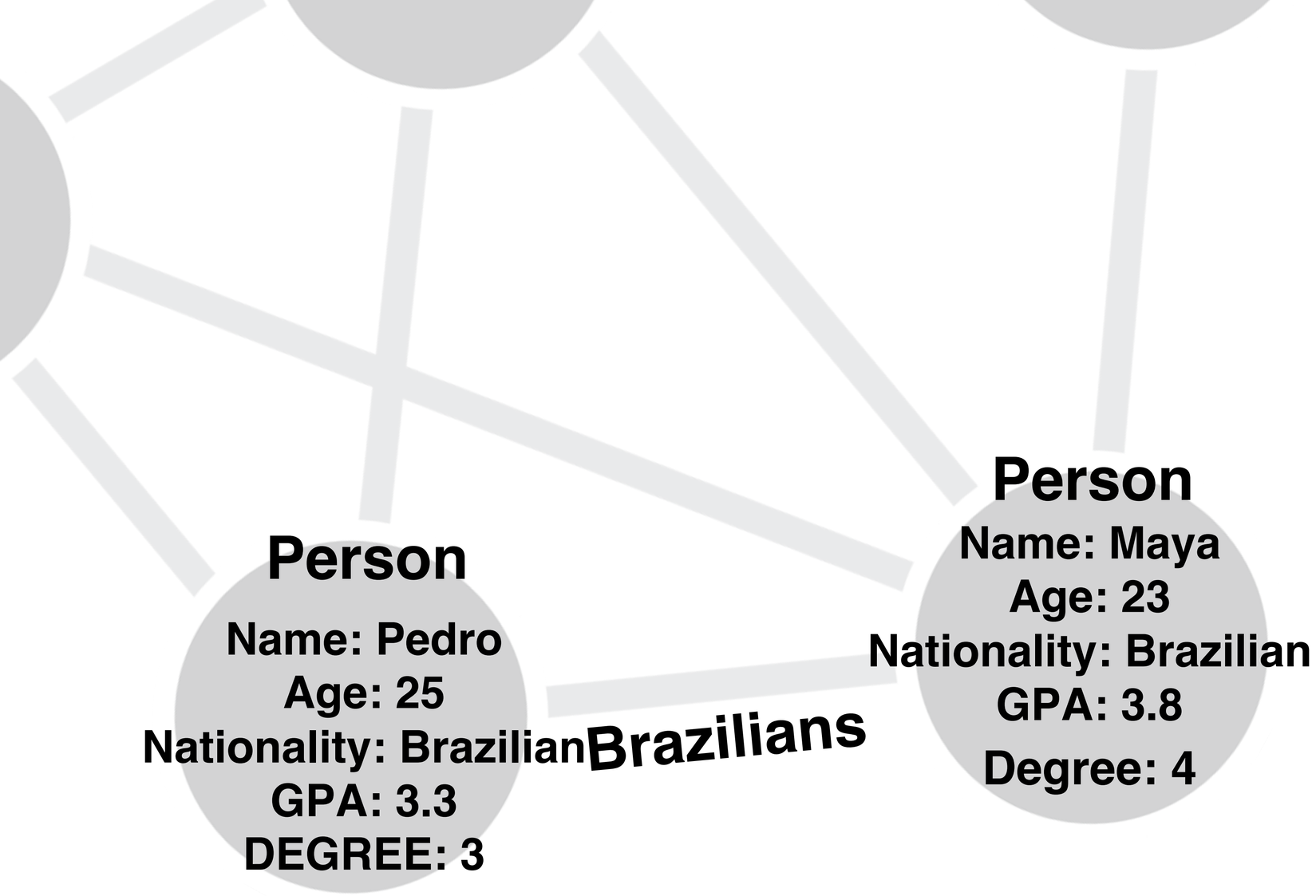


Network and Attribute Characteristics

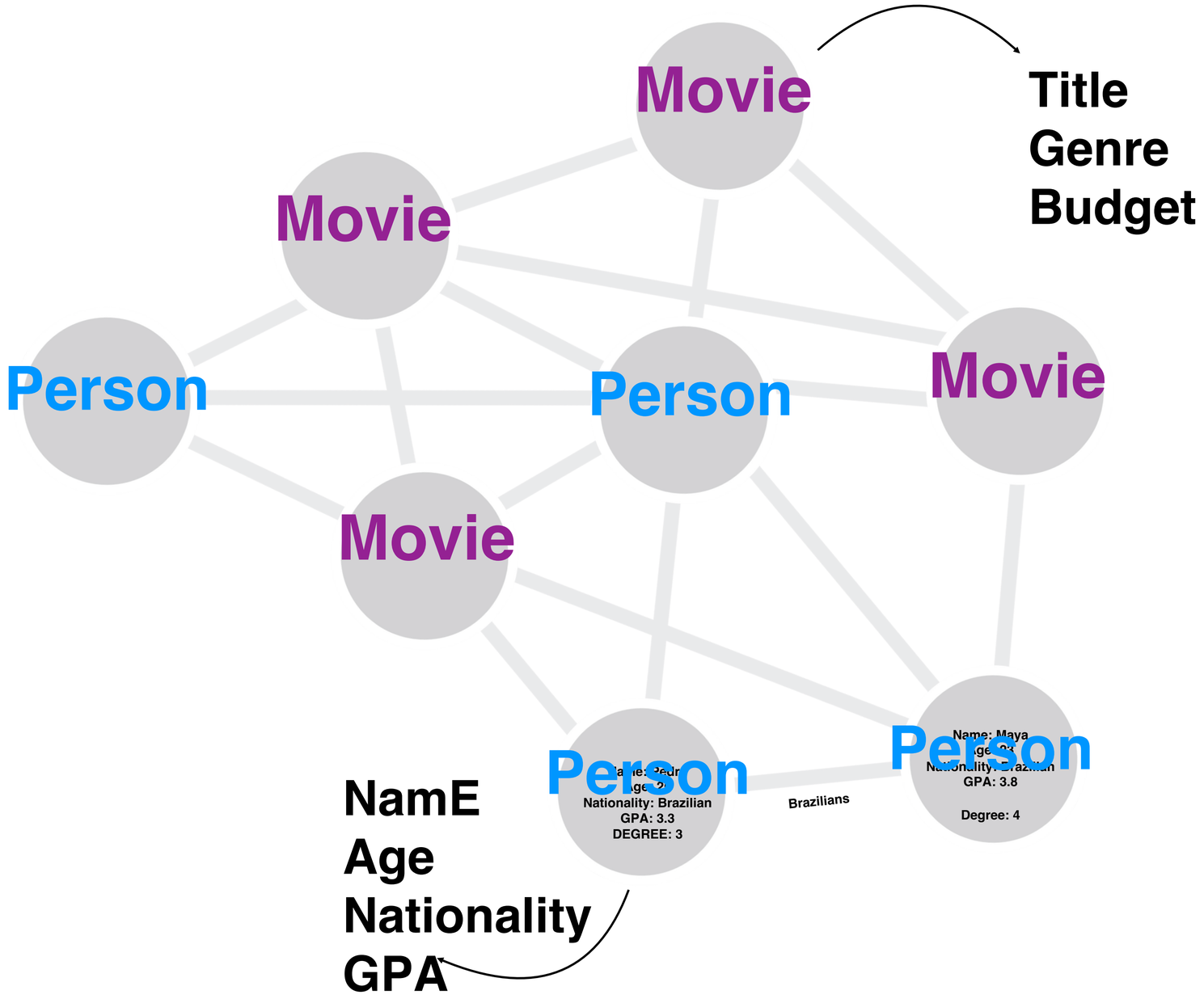

Topology
Name: Maya
Age: 23
Nationality: Brazilian
GPA: 3.8


Name: Maya
Age: 23
Nationality: Brazilian
GPA: 3.8
FRIENDS
3 years

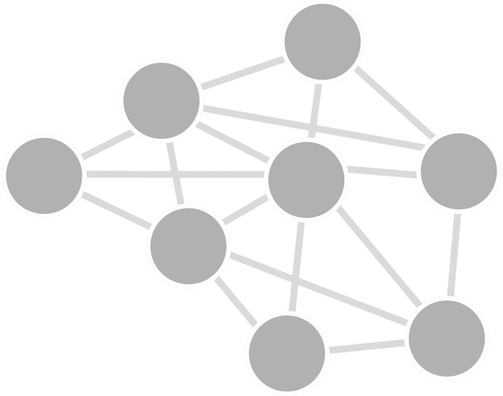

Name: Maya
Age: 23
Nationality: Brazilian
GPA: 3.8
Degree: 4
FRIENDS
3 years


Person
Name: Pedro
Age: 25
Nationality: Brazilian
GPA: 3.3
DEGREE: 3
Brazilians
Person
Name: Maya
Age: 23
Nationality: Brazilian
GPA: 3.8
Degree: 4

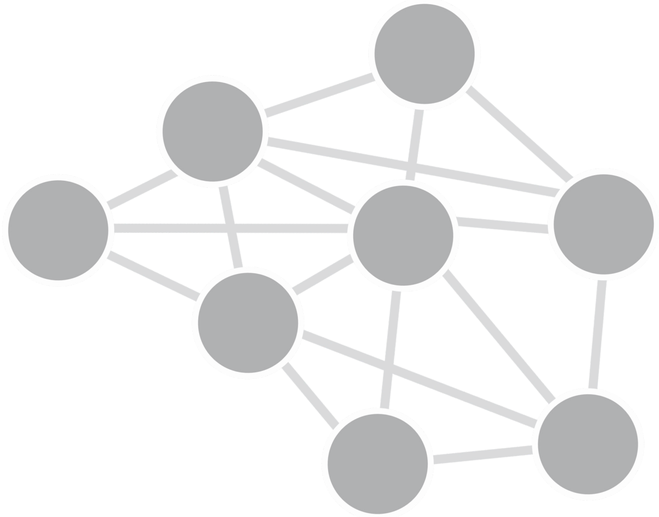
ity


Movie
Title
Genre
Budget
Movie
Movie
Person
Person
Movie
Movie
Person
Name: Maya
GPA: 3.8
Degree: 4
Person
Nationality: Brazilian
GPA: 3.3
DEGREE: 3
Brazilians
NamE
Age
Nationality
GPA

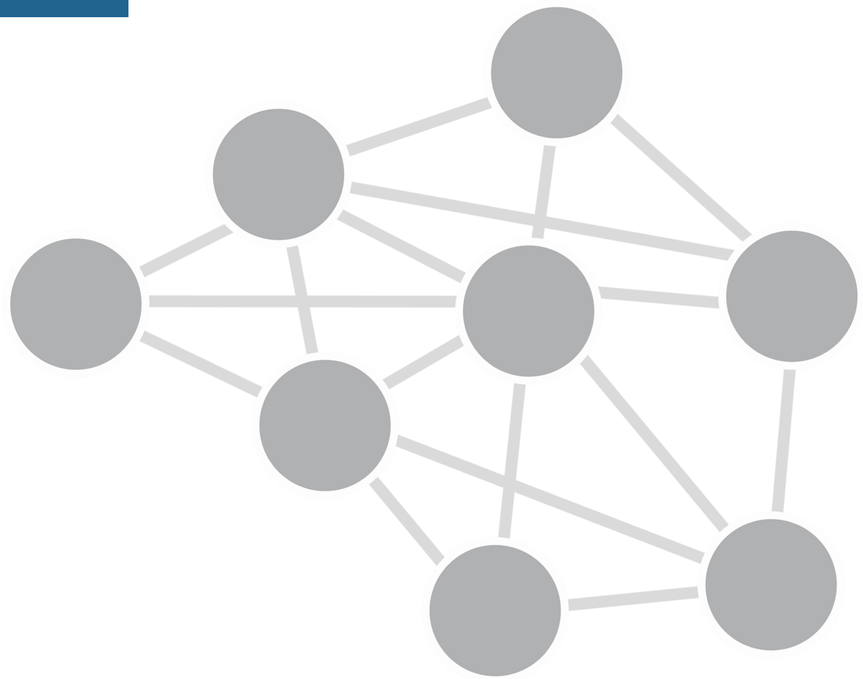
Network Size



Small
<100

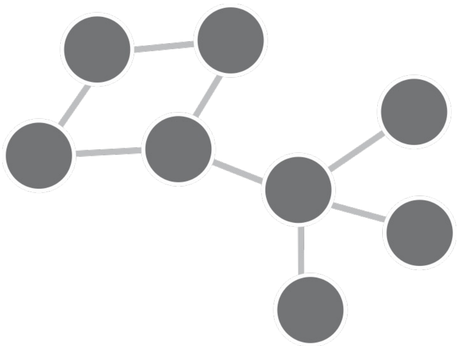


Medium
100–1000

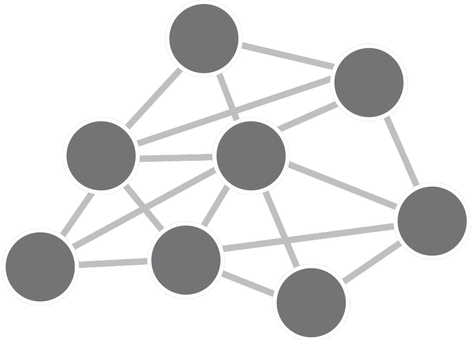


Large
>1000

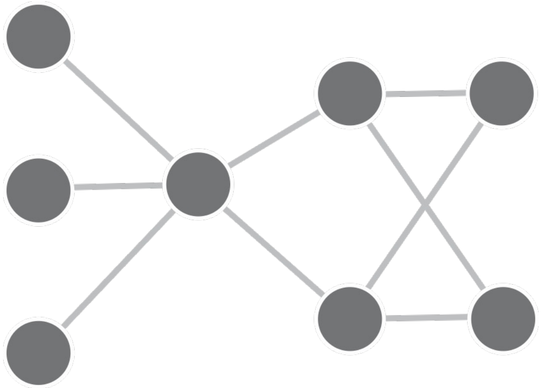
Network Types



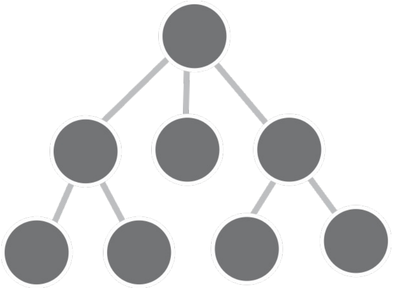
Sparse



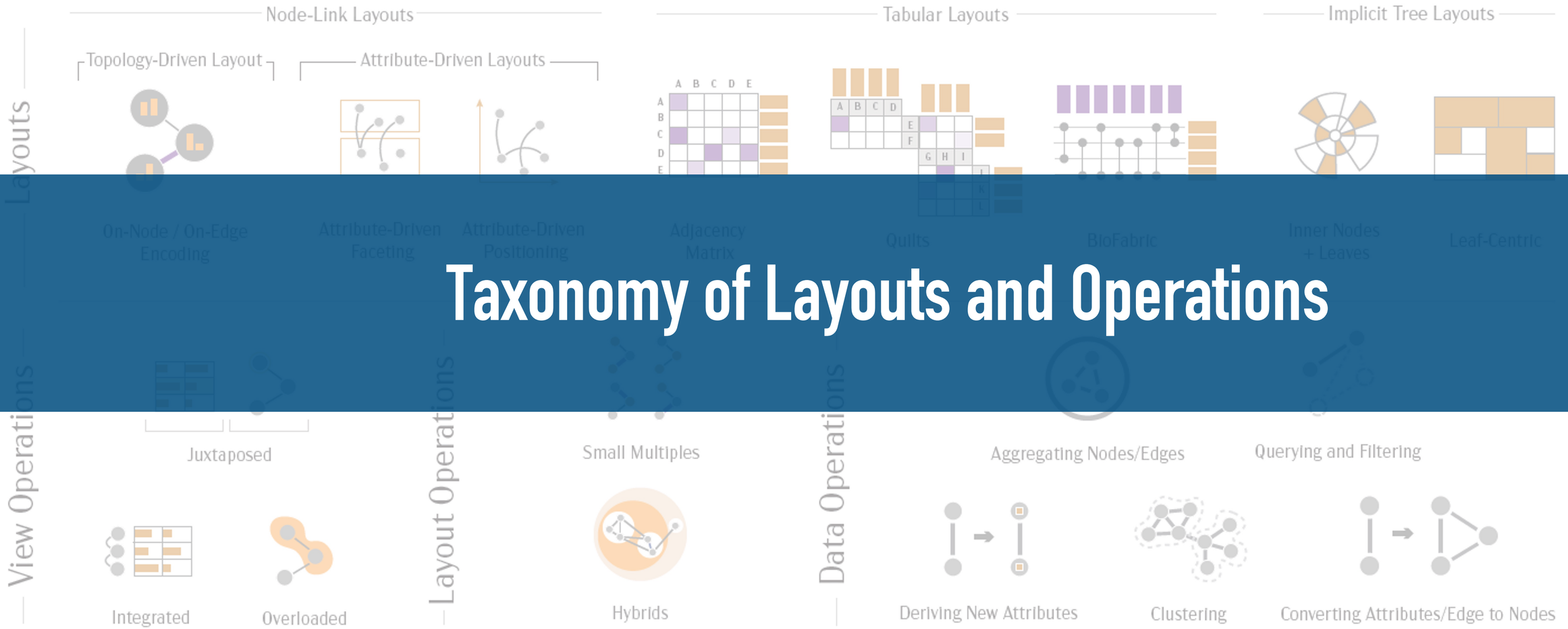
Dense



Layered



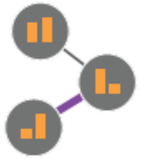
Trees


Node-Link Layouts
Tabular Layouts
Implicit Tree Layouts
Topology-Driven Layout
Attribute-Driven Layouts
On-Node / On-Edge Encoding
Attribute-Driven Faceting
Attribute-Driven Positioning
Adjacency Matrix
Quilts
BioFabric
Inner Nodes + Leaves
Leaf-Centric
Layouts
Taxonomy of Layouts and Operations
View Operations
Juxtaposed
Integrated
Overloaded
Layout Operations
Small Multiples
Hybrids
Data Operations
Aggregating Nodes/Edges
Querying and Filtering
Deriving New Attributes
Clustering
Converting Attributes/Edge to Nodes

Layouts

Node-Link Layouts

Topology-Driven Layout



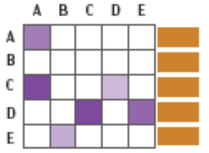
On-Node / On-Edge Encoding

Attribute-Driven Layouts

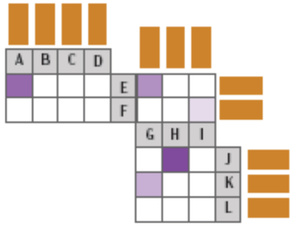
Attribute-Driven Faceting

Attribute-Driven Positioning

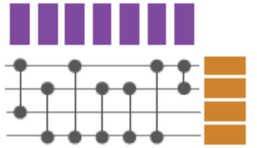
Tabular Layouts



Adjacency Matrix



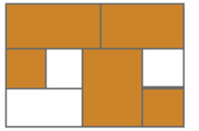
Quilts



BioFabric

Implicit Tree Layouts

Inner Nodes + Leaves



Leaf-Centric

View Operations

Juxtaposed

Integrated

Overloaded

Layout Operations

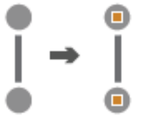
Small Multiples

Hybrids

Data Operations

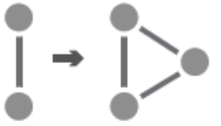
Aggregating Nodes/Edges

Querying and Filtering



Deriving New Attributes

Clustering

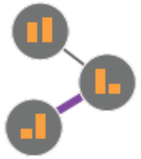


Converting Attributes/Edge to Nodes

Layouts

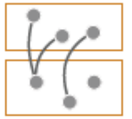
Node-Link Layouts

Topology-Driven Layout



On-Node / On-Edge Encoding

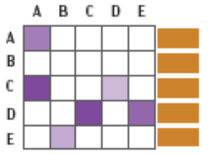
Attribute-Driven Layouts



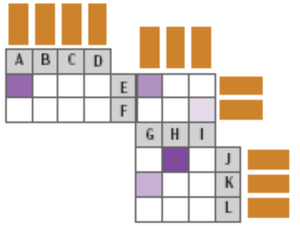
Attribute-Driven Faceting

Attribute-Driven Positioning

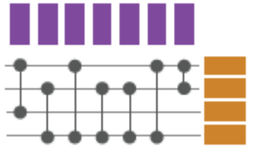
Tabular Layouts



Adjacency Matrix



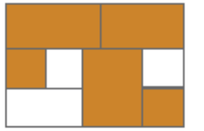
Quilts



BioFabric

Implicit Tree Layouts

Inner Nodes + Leaves



Leaf-Centric

View Operations

Juxtaposed

Integrated

Overloaded

Layout Operations

Small Multiples

Hybrids

Data Operations

Aggregating Nodes/Edges

Querying and Filtering

Deriving New Attributes

Clustering

Converting Attributes/Edge to Nodes

Layouts

Node-Link Layouts

Topology-Driven Layout

On-Node / On-Edge Encoding

Attribute-Driven Layouts

Attribute-Driven Faceting

Attribute-Driven Positioning

Tabular Layouts

Adjacency Matrix

Quilts

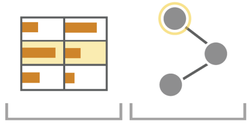
BioFabric

Implicit Tree Layouts

Inner Nodes + Leaves

Leaf-Centric

View Operations

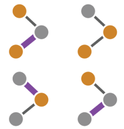


Juxtaposed

Integrated

Overloaded

Layout Operations



Small Multiples

Hybrids

Data Operations

Aggregating Nodes/Edges

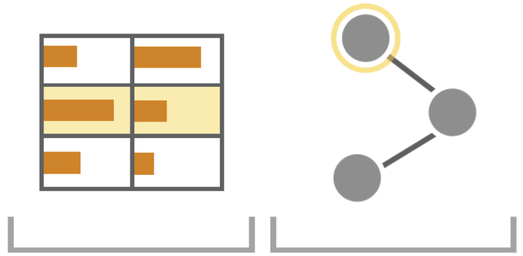
Querying and Filtering

Deriving New Attributes

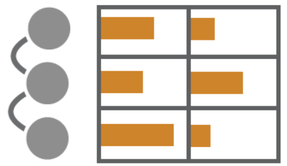
Clustering

Converting Attributes/Edge to Nodes

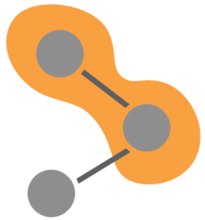
View Operations



Juxtaposed



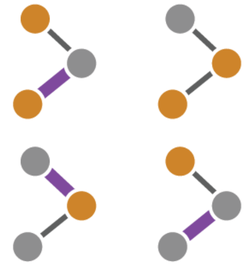
Integrated



Overloaded

Separate views for Topology and Attributes

Layout Operations



Small Multiples

Hybrids

Multiple layouts for Topology or Attributes

Layouts

Node-Link Layouts

Topology-Driven Layout

On-Node / On-Edge Encoding

Attribute-Driven Layouts

Attribute-Driven Faceting

Attribute-Driven Positioning

Tabular Layouts

Adjacency Matrix

Quilts

BioFabric

Implicit Tree Layouts

Inner Nodes + Leaves

Leaf-Centric

View Operations

Juxtaposed

Integrated

Overloaded

Layout Operations

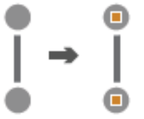
Small Multiples

Hybrids

Data Operations

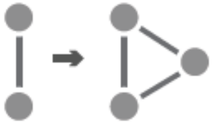
Aggregating Nodes/Edges

Querying and Filtering



Deriving New Attributes

Clustering



Converting Attributes/Edge to Nodes

		Size			Type				Node Attributes				Edge Attributes				Topolog. Structures				
		Small <100 nodes)	Medium (<1,000)	Large (>1,000 nodes)	Complex (sparse)	Complex (dense)	Layered/K-Partite	Trees	Few (<5)	Several (≥5)	Homog. (1 type)	Hetero. (>1 type)	Few (<3)	Several (≥3)	Homog. (1 type)	Hetero. (>1 type)	Single node/edge	Neighbors	Paths	Clusters	Entire/sub network
Node-Link Layouts	On-node/edge encoding	3	2	1	3	1	3	3	2	1	3	2	2	1	3	1	3	3	2	2	2
	Attr.-driven faceting	3	1	1	3	1	3	1	3	1	3	3	2	1	2	1	3	2	1	1	1
	Attr.-driven positioning	3	1	1	3	1	1	1	3	1	3	1	2	1	2	1	3	2	1	1	2
Tabular Layouts	Adjacency matrix	3	1	1	2	3	2	1	2	3	3	2	3	2	3	2	3	3	1	3	2
	Quilts	3	1	1	3	1	3	3	3	3	3	3	3	3	3	2	3	3	2	2	2
	BioFabric	3	1	1	3	3	2	1	3	3	3	3	3	3	3	3	3	1	1	1	2
Implicit	Inner nodes & leaves	3	2	1	0	0	0	3	3	1	3	1	0	0	0	0	3	3	3	0	3
	Leaves	3	2	2	0	0	0	3	3	1	3	1	0	0	0	0	3	2	1	0	3
View Operations	Juxtaposed	3	2	1	3	1	3	3	3	3	3	3	3	3	3	3	2	1	1	2	2
	Integrated	3	2	1	3	1	3	3	3	3	3	3	2	2	3	3	3	3	3	1	2
	Overloaded	3	2	1	3	1	3	3	3	1	3	1	1	1	1	1	3	3	2	3	2

0 Does *not* support

1 Supports poorly

2 Supports

3 Optimized for

Layouts

Node-Link Layouts

Topology-Driven Layout

On-Node / On-Edge Encoding

Attribute-Driven Layouts

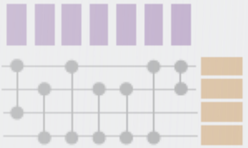
Attribute-Driven Faceting

Attribute-Driven Positioning

Tabular Layouts

Adjacency Matrix

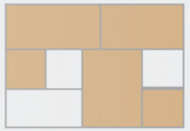
Quilts



BioFabric

Implicit Tree Layouts

Inner Nodes + Leaves



Leaf-Centric

LAYOUTS

View Operations

Integrated

Overloaded

VIEW OPERATIONS

Layout Operations

Hybrids

LAYOUT OPERATIONS

Data Operations

Aggregating No

Querying and Filtering

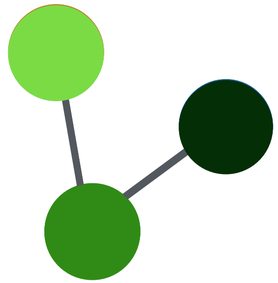
Deriving New Attributes

Clustering

Converting Attributes/Edge to Nodes

DATA OPERATIONS

Node-Link Diagram with on-node encoding

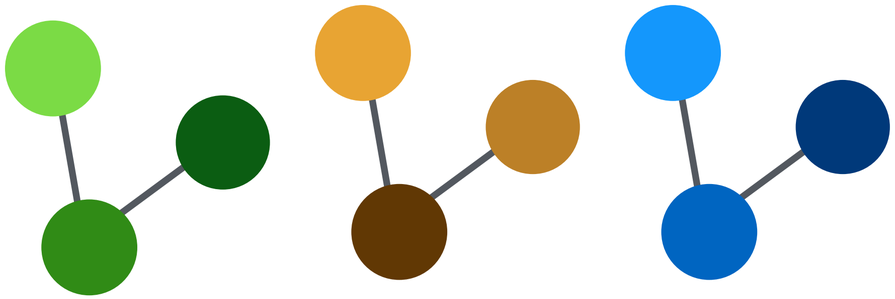


LAYOUTS

VIEW OPERATIONS	LAYOUT OPERATIONS	DATA OPERATIONS

LAYOUTS		
VIEW OPERATIONS	LAYOUT OPERATIONS	DATA OPERATIONS

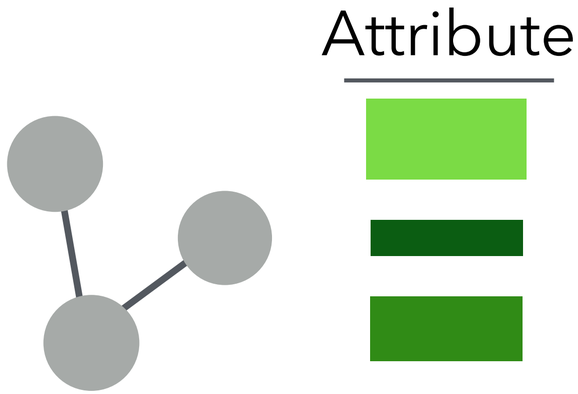
Small Multiples



LAYOUTS

VIEW OPERATIONS	LAYOUT OPERATIONS	DATA OPERATIONS

Juxtaposed Views



LAYOUTS

VIEW OPERATIONS	LAYOUT OPERATIONS	DATA OPERATIONS

Filter Data

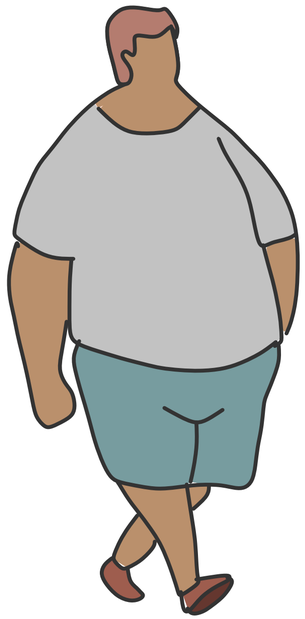
Attribute

Friends

Friends

Co-workers

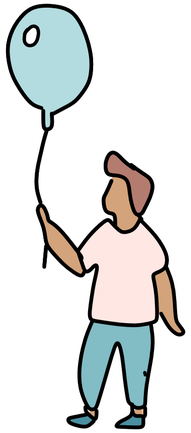
Mother



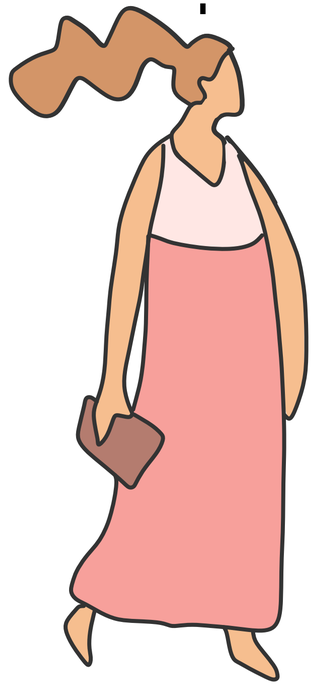
Cole

Tom

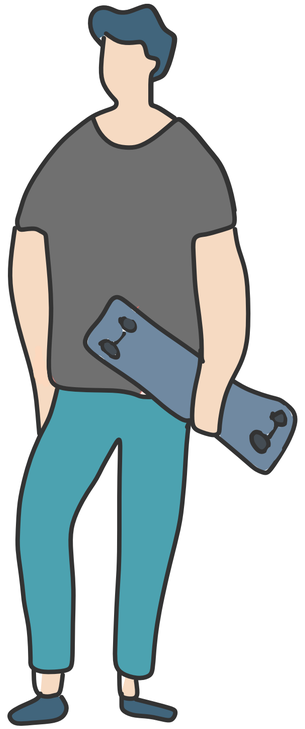
Abby



Jon



Sue

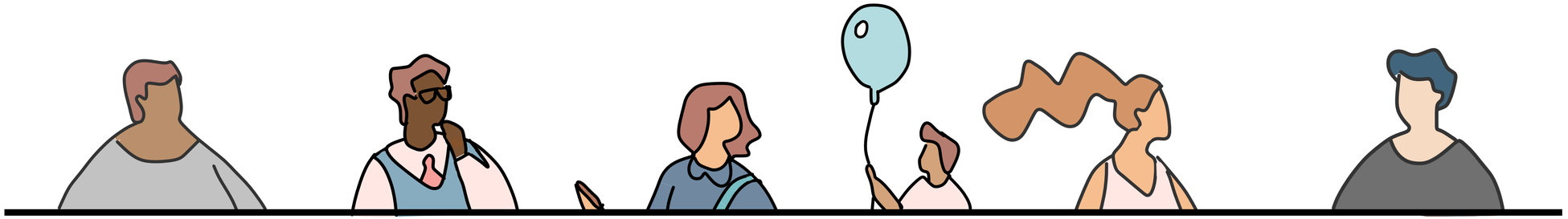


Mark

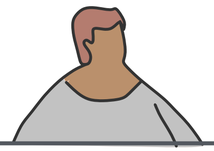
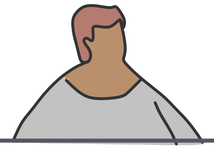
Dating

Married

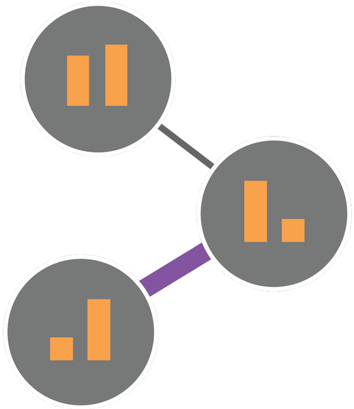
Soccer Coach

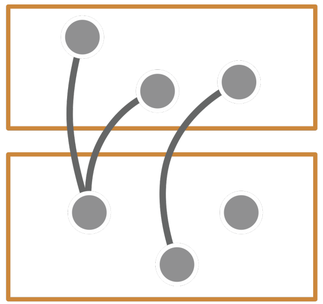


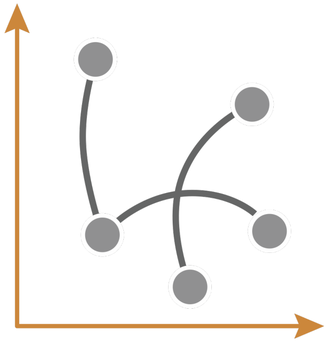
Name	**Cole**	**Tom**	**Abby**	**Jon**	**Sue**	**Mark**
Beverage	Port	Beer	Port	Coke	Coke	Beer
Day 1	1	0	4	3	3	5
Day 2	0	2	5	3	5	5
Day 3	4	1	2	2	4	3

Source	Target	Type	Duration
		Co-workers	3 years
		Soccer Coach	2 years
		Dating	1 year
		Mother / Son	7 years
		Friends	12 years
		Friends	3 years
		Married	6 years

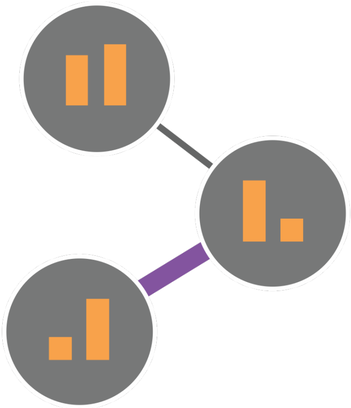
Node-Link Layouts



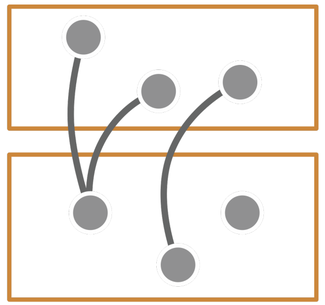


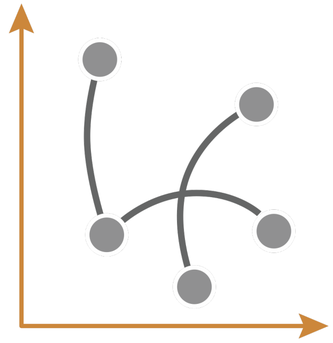


Topology Driven Layout

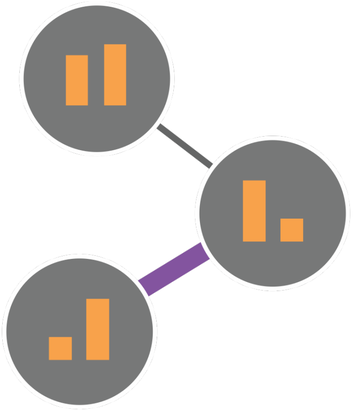


Attribute Driven Layouts



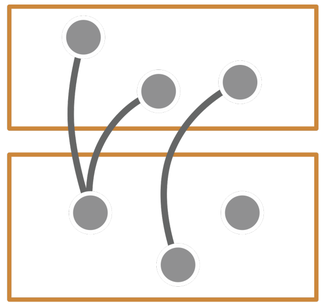


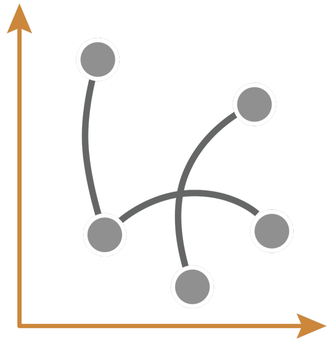
Topology Driven Layout



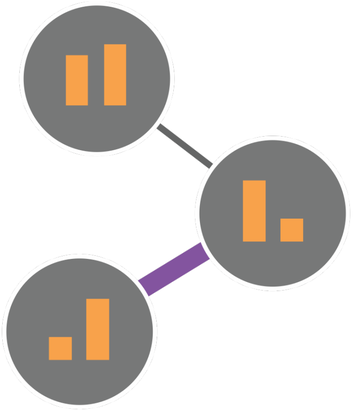
On-Node / On-Edge Encoding

Attribute Driven Layouts



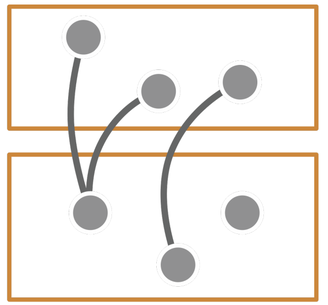


Topology Driven Layout

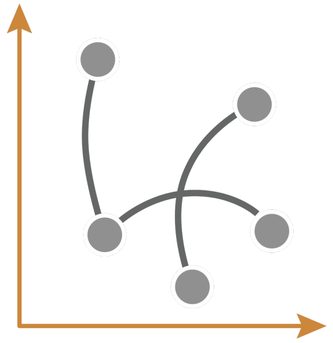


On-Node / On-Edge Encoding

Attribute Driven Layouts

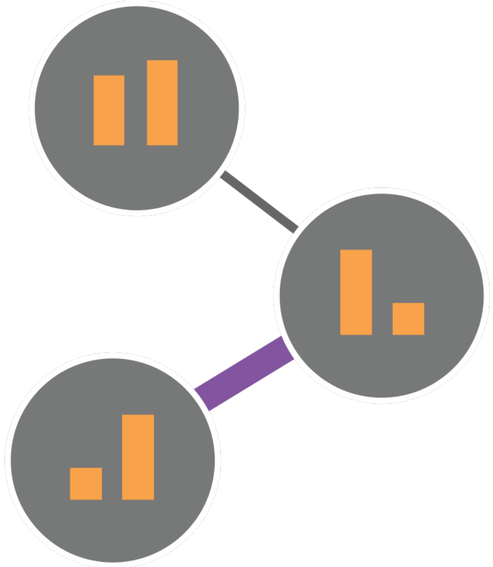


Attribute-Driven Faceting

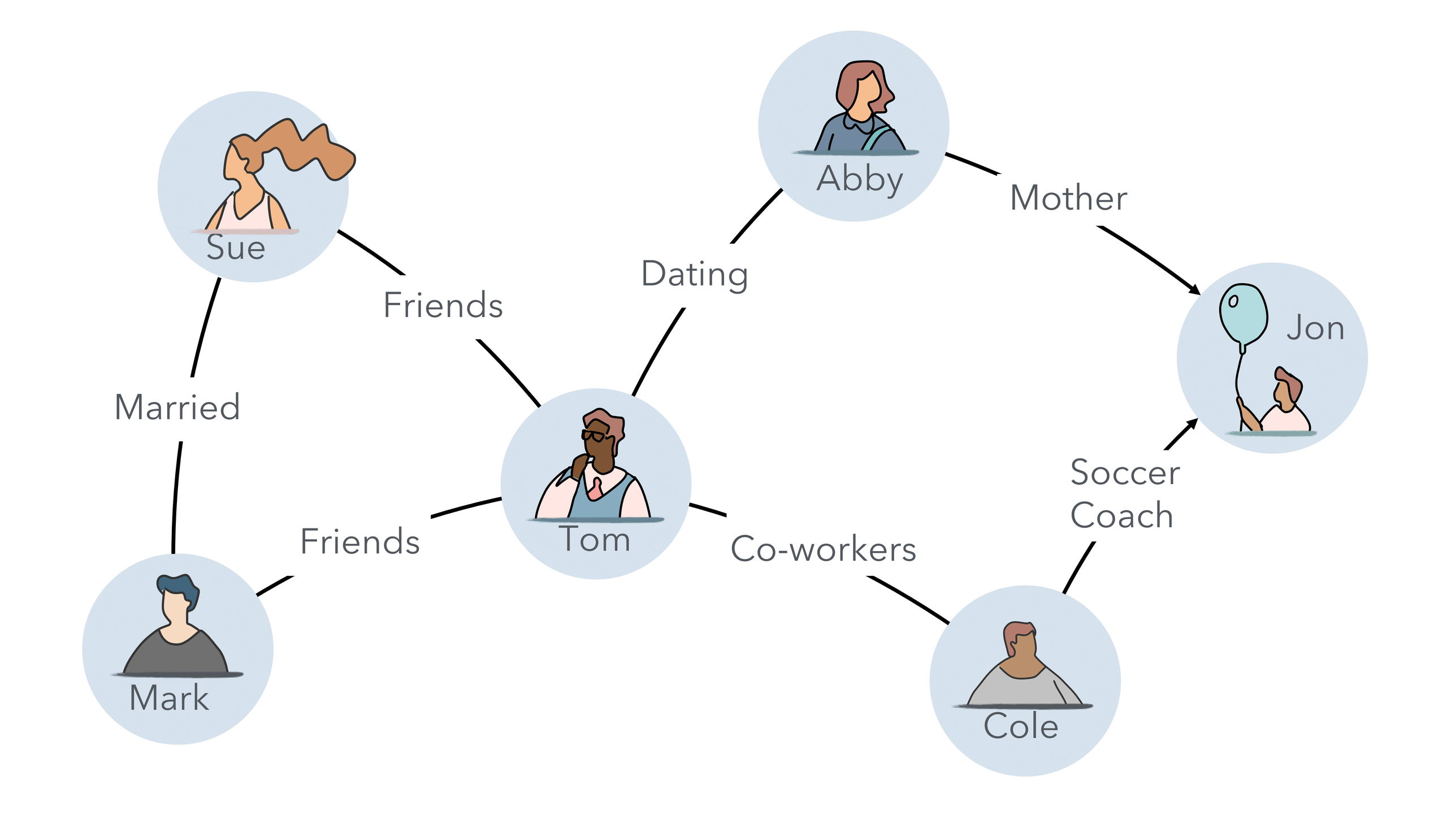


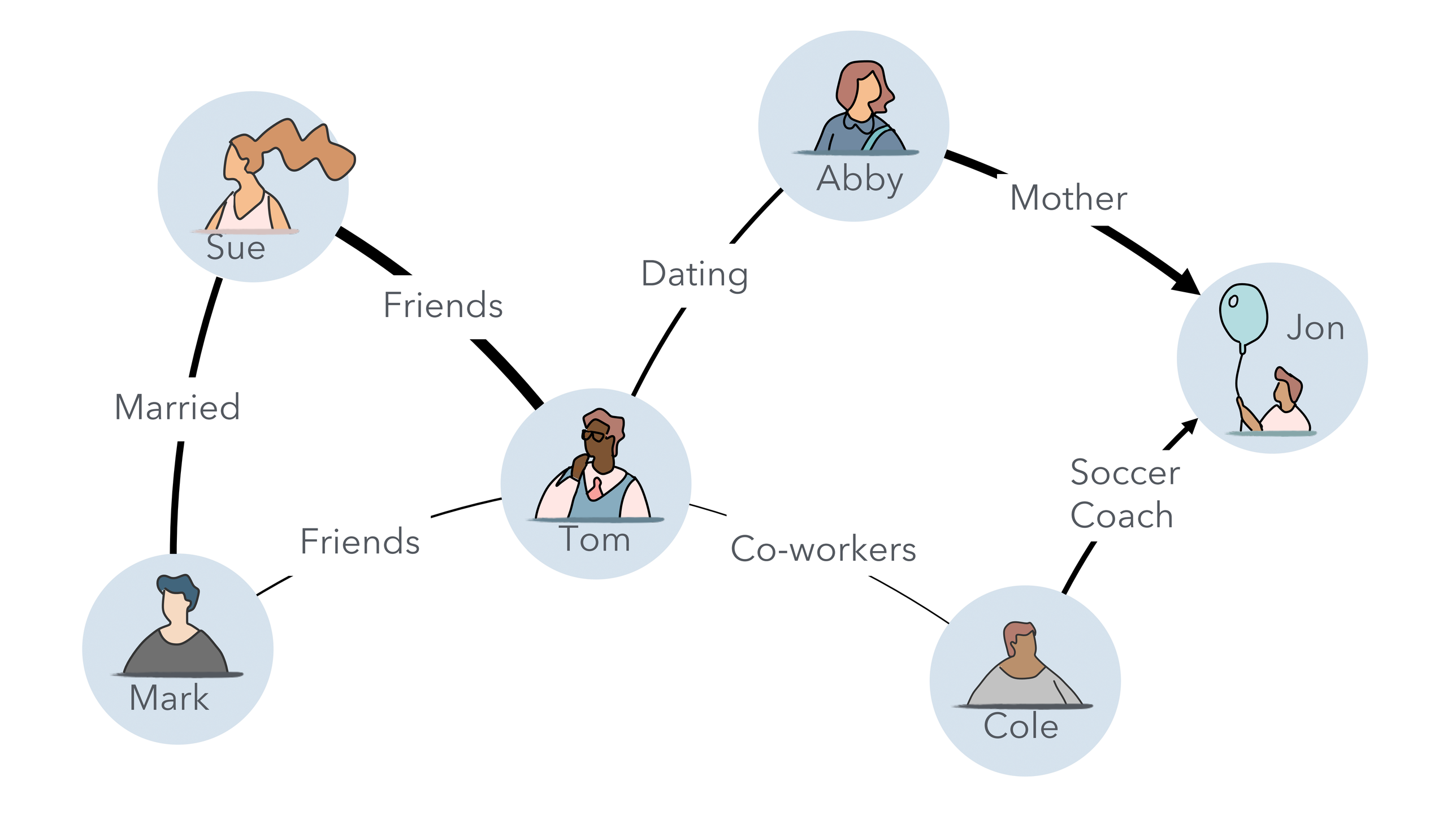
Attribute-Driven Positioning

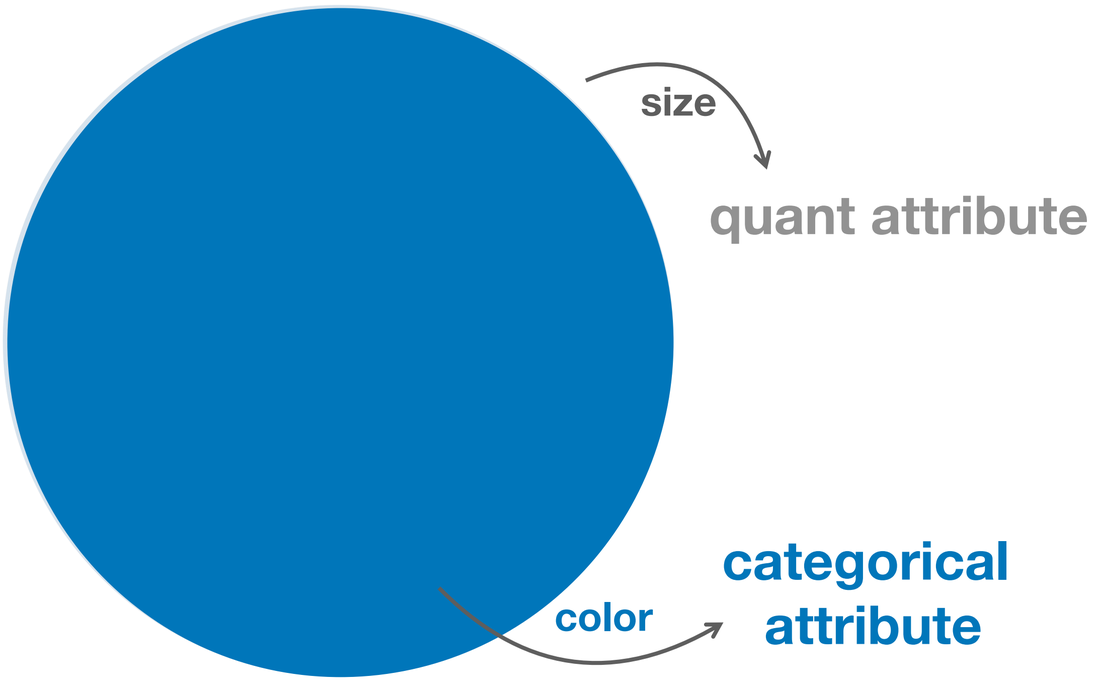
On-Node / On-Edge Encoding

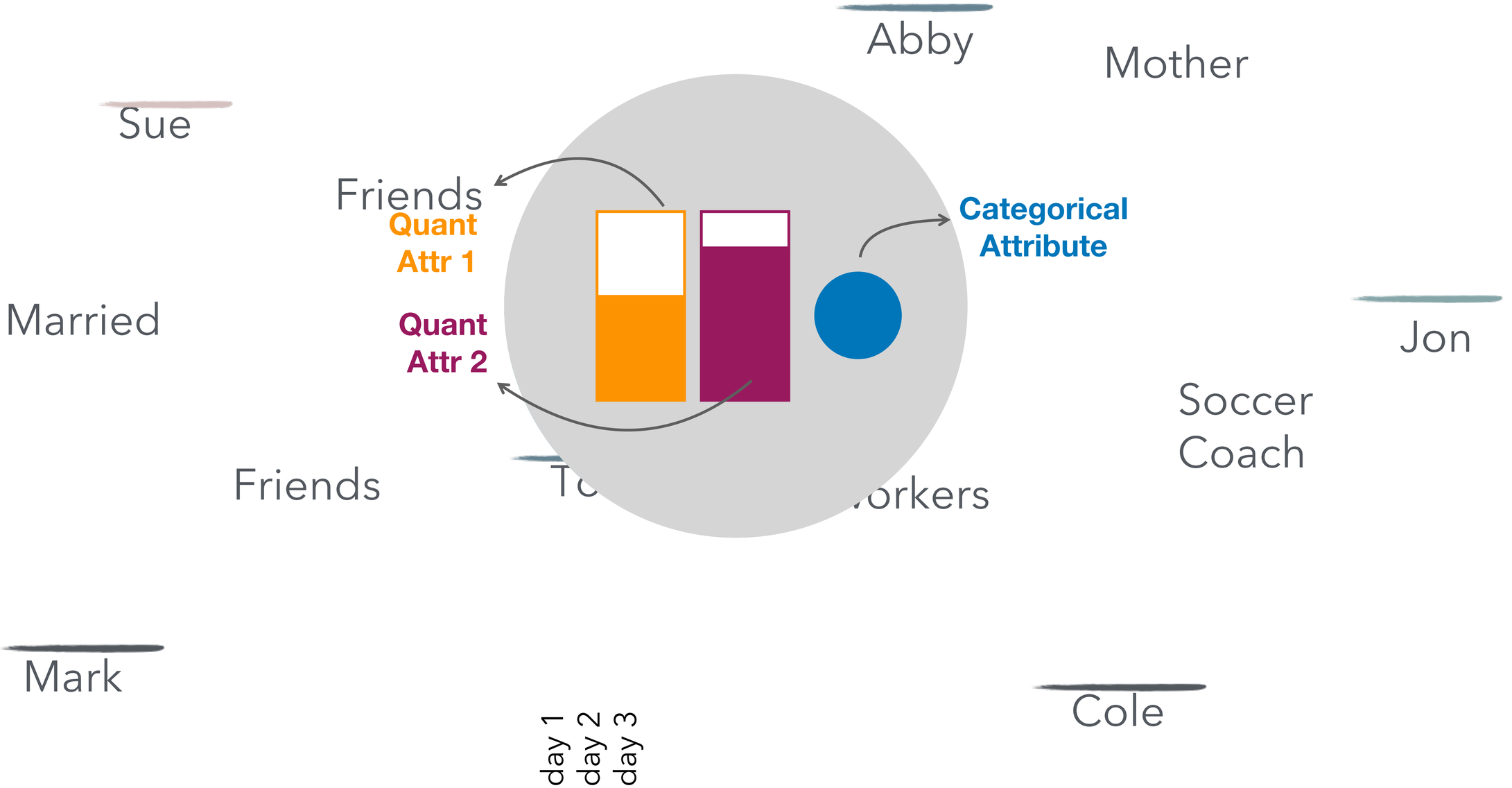


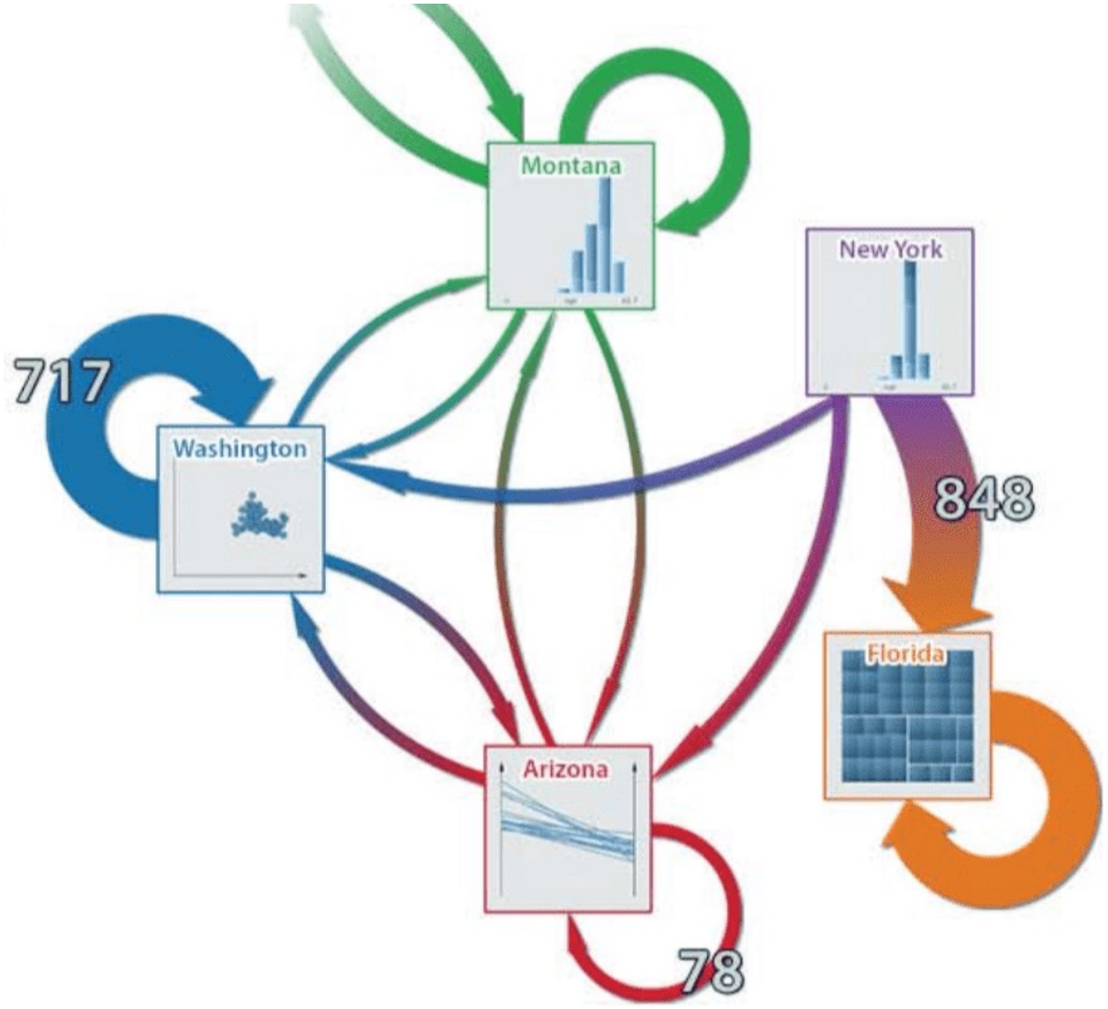

Mark
Sue
Tom
Abby
Cole
Jon


Sue
Abby
Mother
Friends
Dating
Jon
Married
Tom
Soccer
Coach
Friends
Co-workers
Mark
Cole

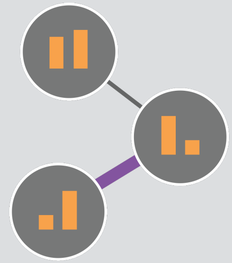

Sue
Abby
Mother
Friends
Dating
Jon
Married
Tom
Soccer
Coach
Friends
Co-workers
Mark
Cole


size
quant attribute
color
categorical
attribute

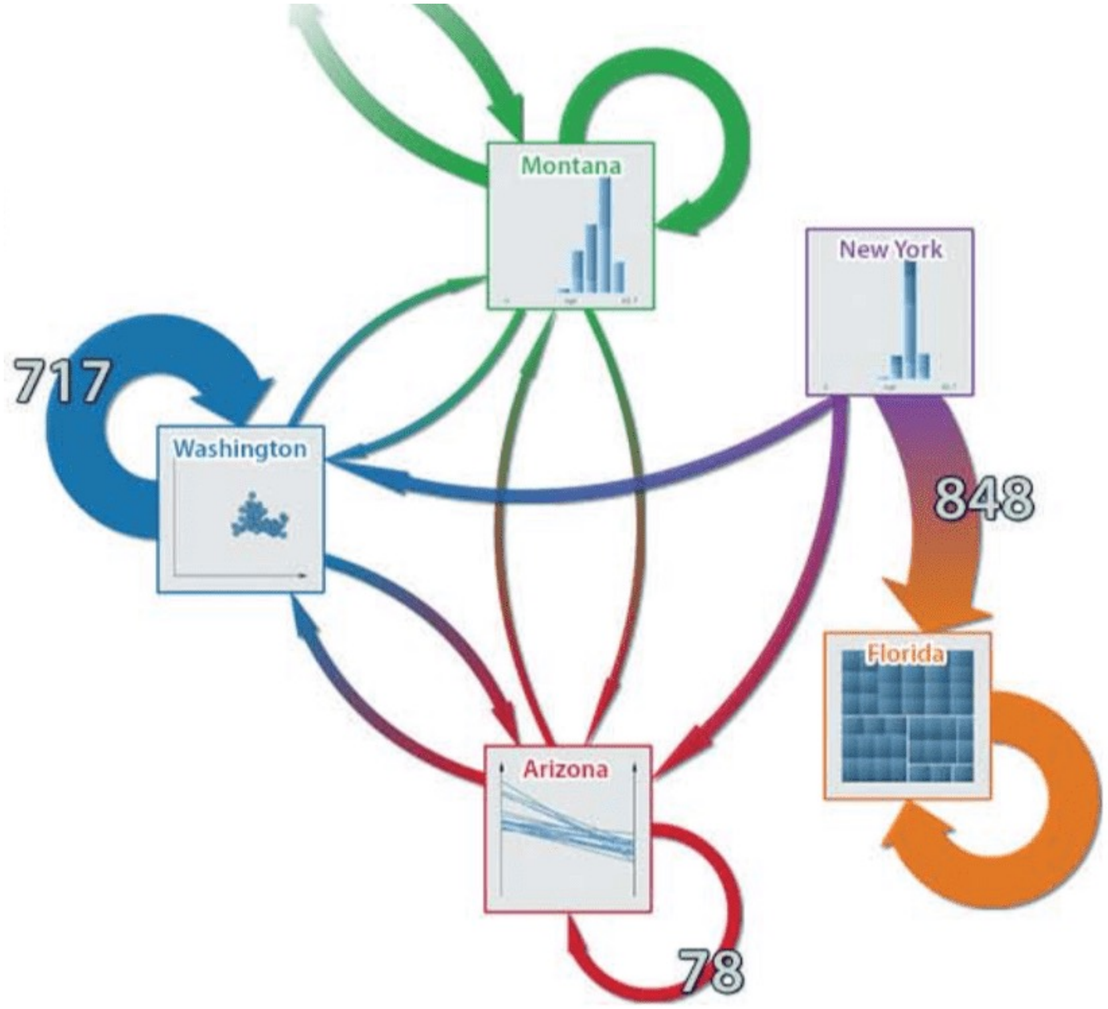

Abby
Mother
Sue
Friends
Quant
Attr 1
Quant
Attr 2
Categorical
Attribute
Married
Jon
Soccer
Coach
Friends
orkers
Mark
Cole
day 1
day 2
day 3



Elzen and Wijk, 2014

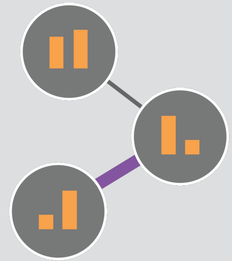


On-Node / On-Edge Encoding

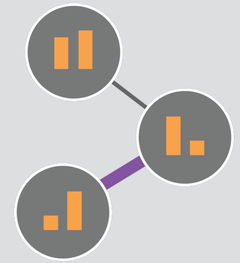


Elzen and Wijk, 2014

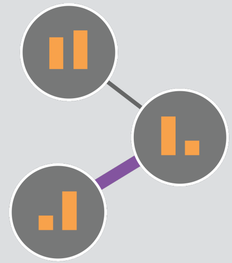
Aggregating Nodes/Edges



On-Node / On-Edge Encoding



On-Node / On-Edge Encoding



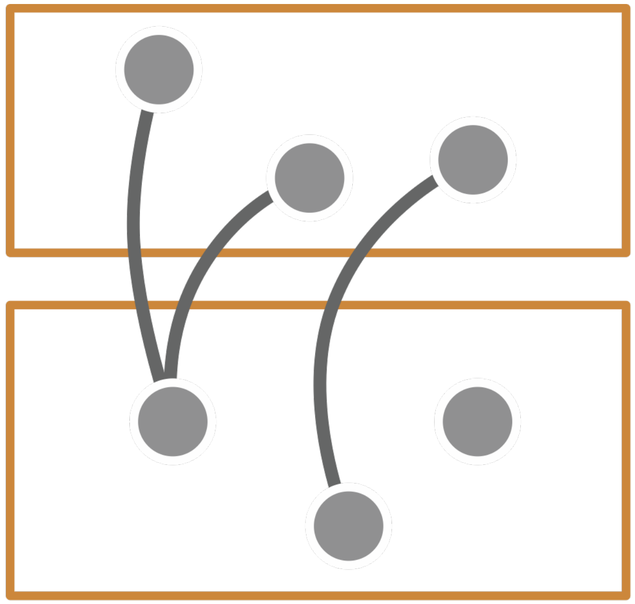
On-Node / On-Edge Encoding

Is easily understood by most users
Works well for all types of networks

Scalability.
Node size leaves little space to encode attributes.

Recommended for small networks when only a few (usually under five) attributes on the nodes are shown, or in combination with a zooming/filtering strategy

Attribute-Driven Faceting




Coca-Cola
classic


BEER


PORTO
TAWNY PORT
e 0,75cl
20
years old


Coca-Cola
classic


BEER

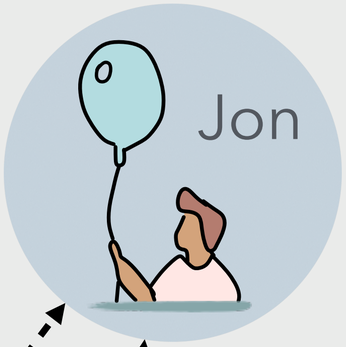

PORTO
TAWNY PORT
20
years old


Coca-Cola
classic


BEER


PORTO
TAWNY PORT
20
years old

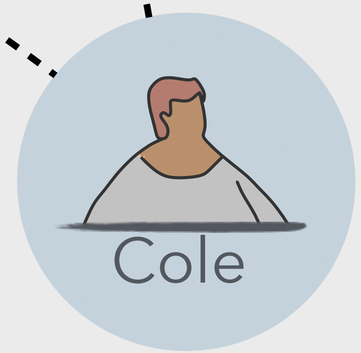

Sue


Jon

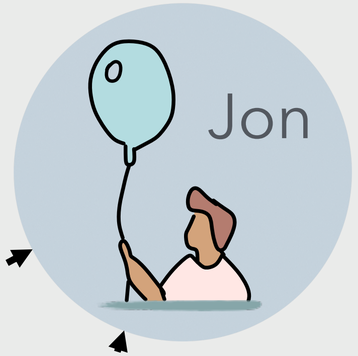

Mark


Tom


Abby


Cole


Sue


Jon

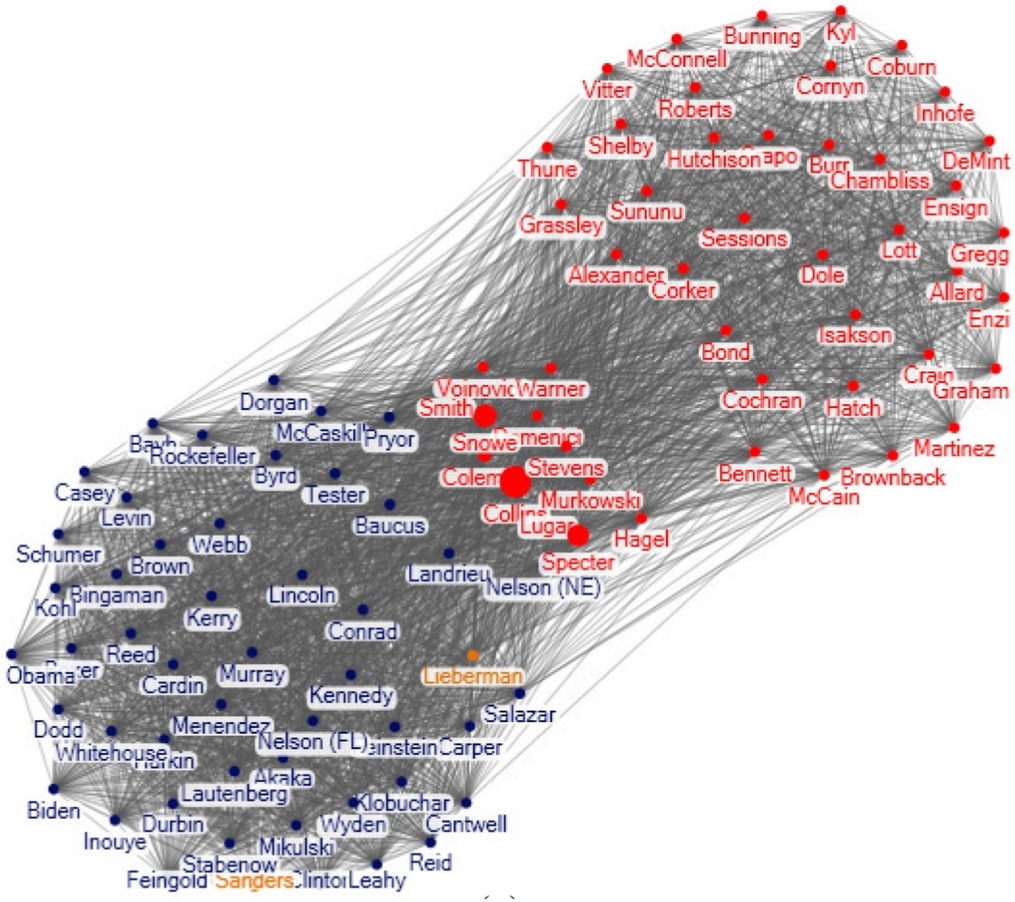

Mark

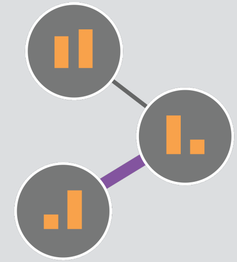

Tom


Abby


Cole

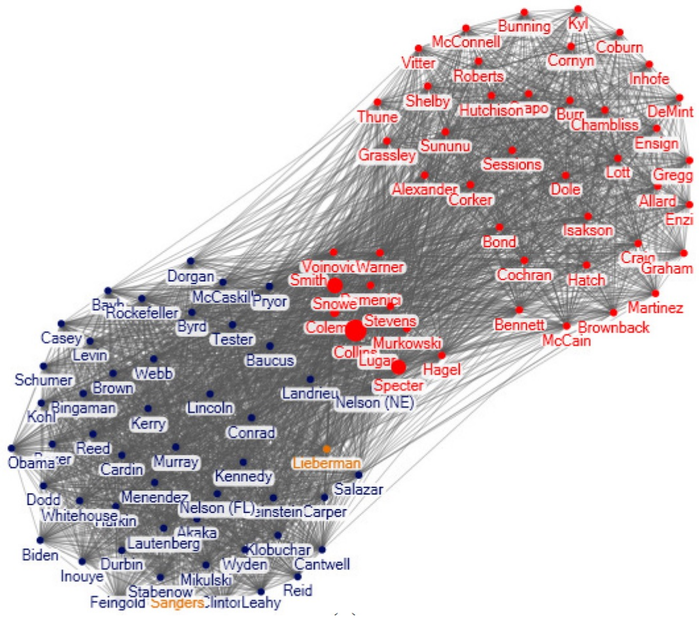
Group-in-a-box *Rodrigues et al. 2011*

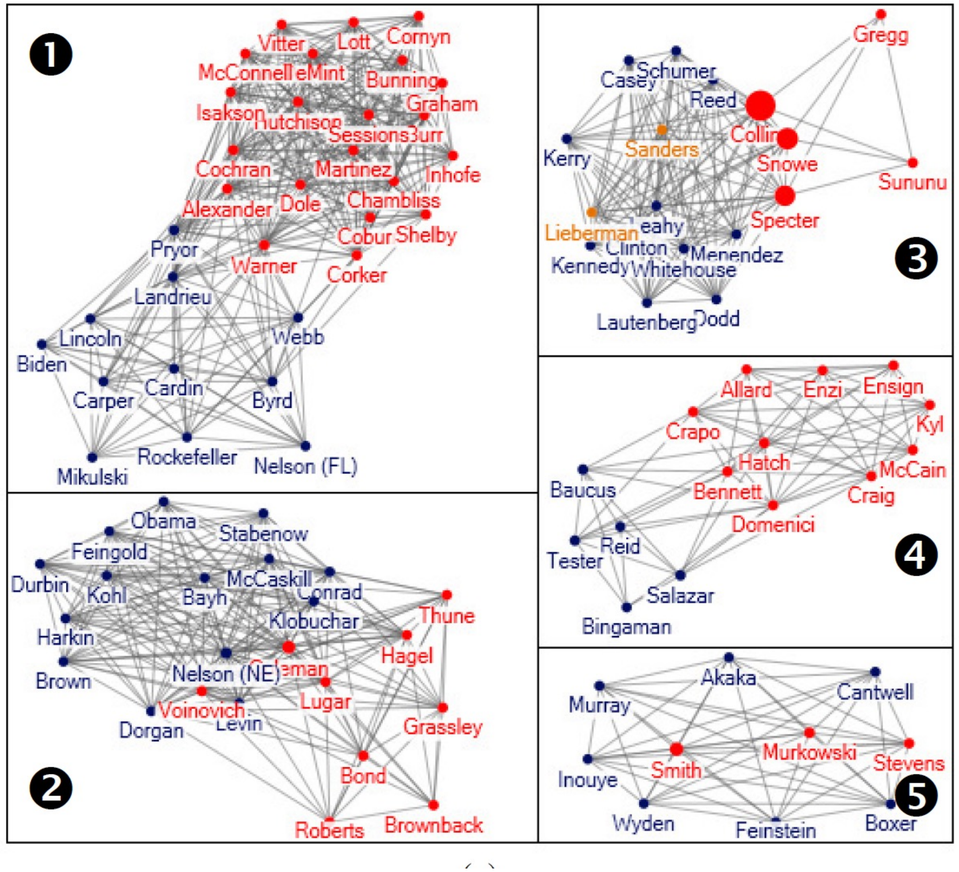


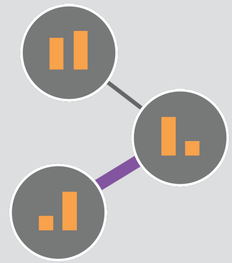


On-Node / On-Edge Encoding

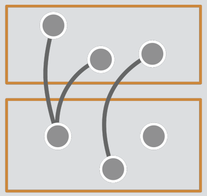
Group-in-a-box *Rodrigues et al. 2011*





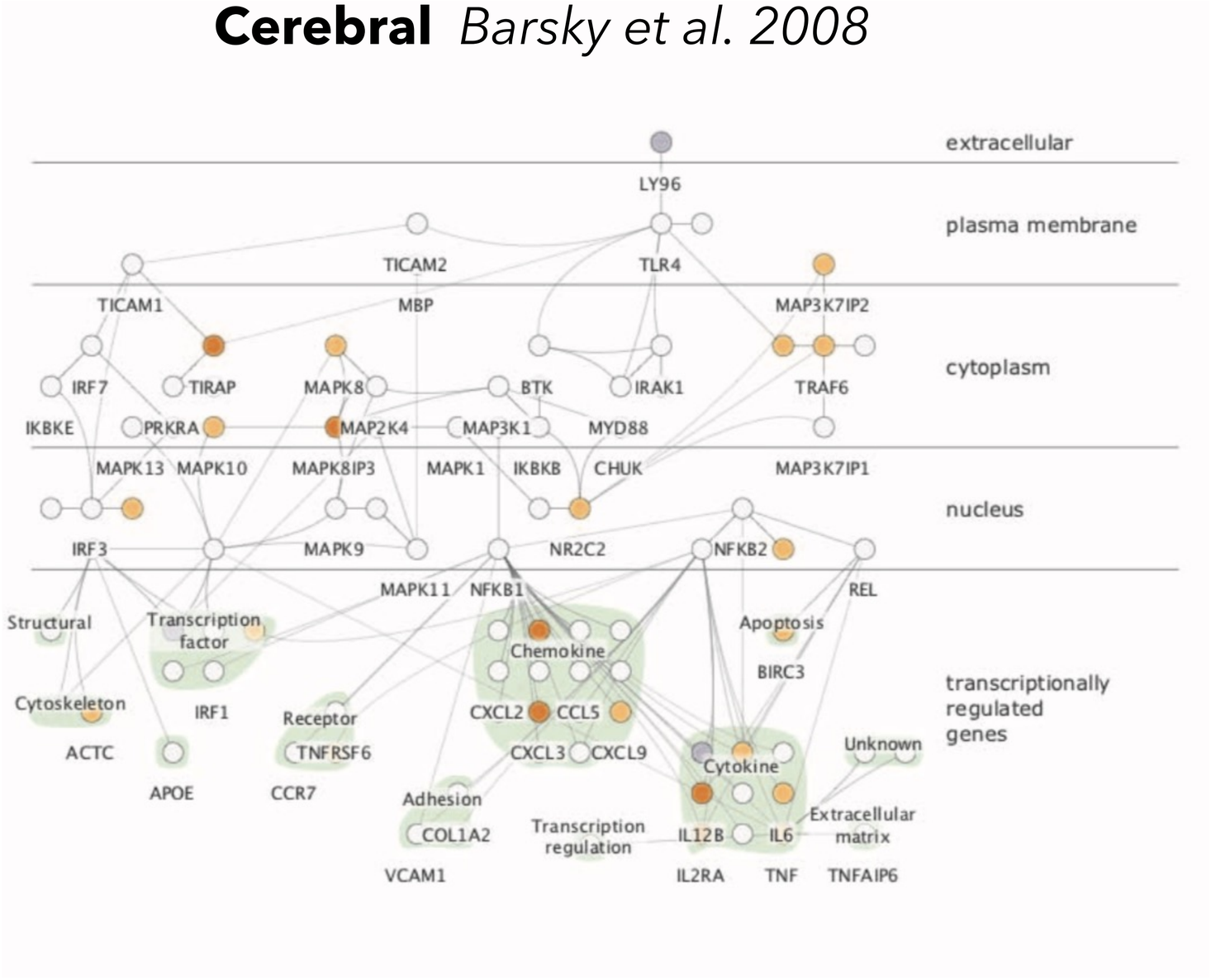


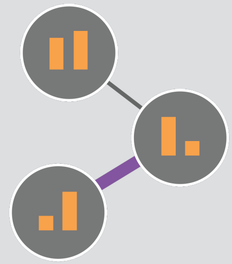
On-Node / On-Edge Encoding



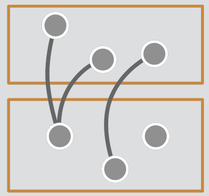
Attribute-Driven Faceting

Cerebral *Barsky et al. 2008*

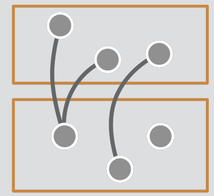




On-Node / On-Edge Encoding



Attribute-Driven Faceting



Attribute-Driven Faceting

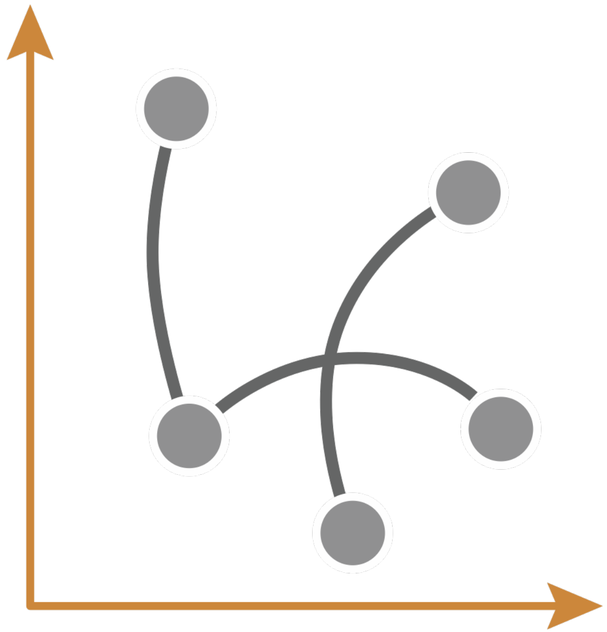
Well suited for networks with different node types or with an important categorical or set-like attribute.

Less scalable with respect to the number of nodes and network density than node-link layouts.

Neighborhoods, paths, and clusters are not easily visible if they span different facets.

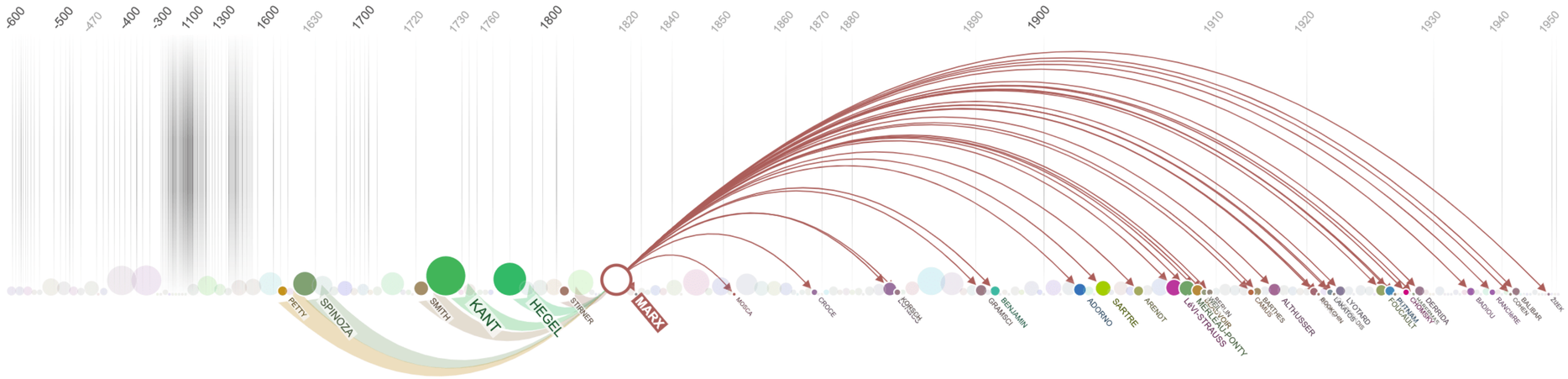
Recommended for networks where nodes can be separated into groups easily and where these groups are central to the analysis

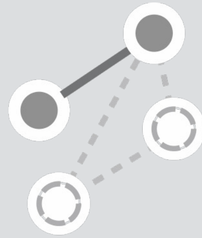
Attribute-Driven Positioning



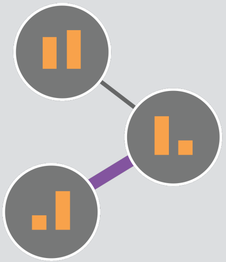

ANCHORAGE
VANCOUVER
EDMONTON
SEATTLE
PORTLAND
SAN FRANCISCO
DENVER
MINNEAPOLIS / ST. PAUL
CHICAGO
KANSAS CITY
TORONTO
DALLAS
CLEVELAND
MONTREAL
BALTIMORE
HALIFAX
WASHINGTON D.C.
BOSTON
PHILADELPHIA
NEW YORK
JFK & NEWARK
TAMPA BAY
ORLANDO
ICELAND
HELSINKI
STOCKHOLM
OSLO
GOTHENBURG
BERGEN
COPENHAGEN
BILLUND
BERLIN
HAMBURG
FRANKFURT
MUNICH
AMSTERDAM
GLASGOW
BRUSSELS
ZURICH
MANCHESTER
MILAN
DUBLIN
LONDON
HEATHROW
& GATWICK
PARIS
ORLY & CDG
GENEVA
MADRID

Edge Map *Dork et al. 2011*

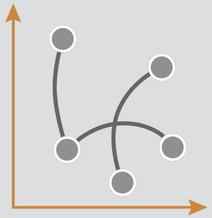




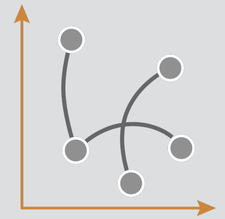
Querying and Filtering



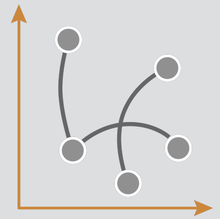
On-Node / On-Edge Encoding



Attribute-Driven Positioning



Attribute-Driven
Positioning



Attribute-Driven Positioning

Well suited for quantitative attributes

Does not lend itself well to visualizing the topology of the network.

Recommended for smaller, sparse networks where relationships between node attributes are paramount to the analysis task, and topological features only provide context

Tools and Applications

For graphic designer and developers

developer

JS

Observable Search Teams Demo Fork Sign in

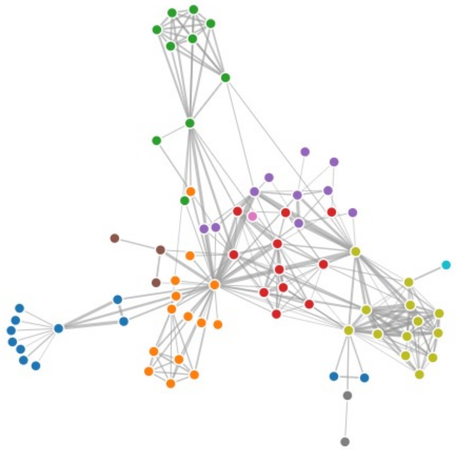
Welcome. This is live code! Click the left margin to view or edit.

D3 · Nov 15, 2017
Bring your data to life.
By Mike Bostock

Listed in d3-drag, d3-force, and Visualization 178 forks

Force-Directed Graph

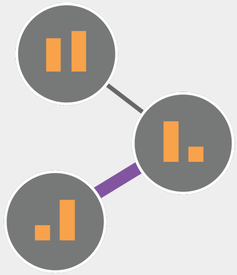
This network of character co-occurence in *Les Misérables* is positioned by simulated forces using d3-force. See also a disconnected graph, and compare to WebCoLa.

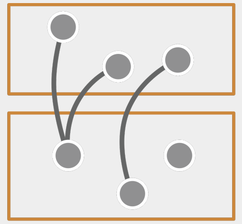


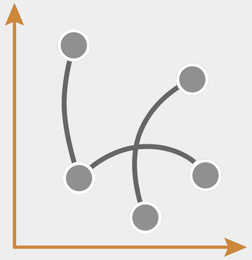
```
chart = {
  const links = data.links.map(d => Object.create(d));
  const nodes = data.nodes.map(d => Object.create(d));

  const simulation = d3.forceSimulation(nodes)
      .force("link", d3.forceLink(links).id(d => d.id))
      .force("charge", d3.forceManyBody())
      .force("center", d3.forceCenter(width / 2, height / 2));

  const svg = d3.create("svg")
```



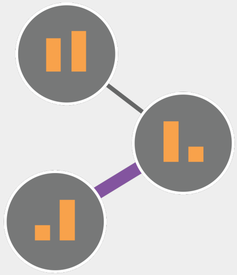




developer

JS

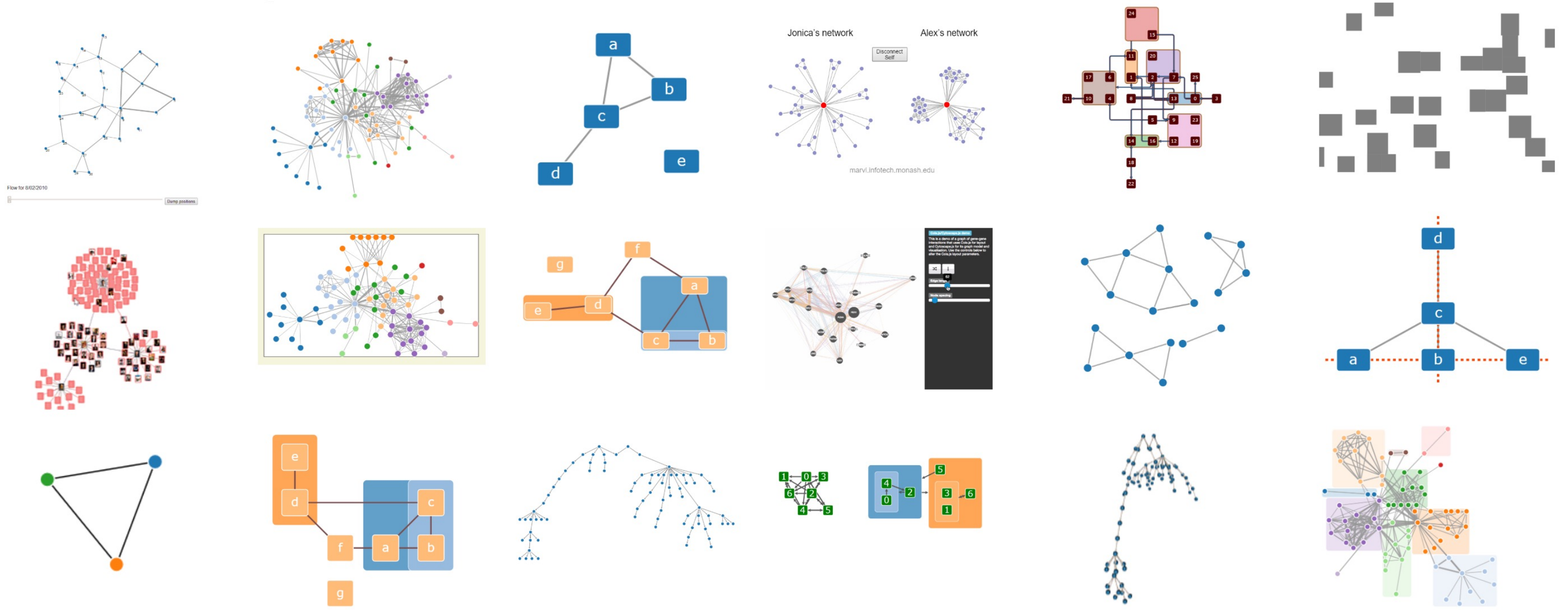
Cola.js (A.K.A. "WebCoLa") is an open-source JavaScript library for arranging your HTML5 documents and diagrams using constraint-based optimization techniques.

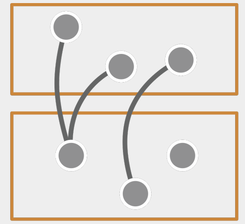


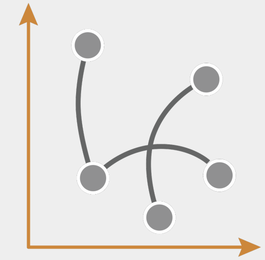
Overview Wiki API Source

cola.js

Constraint-Based Layout in the Browser







developer

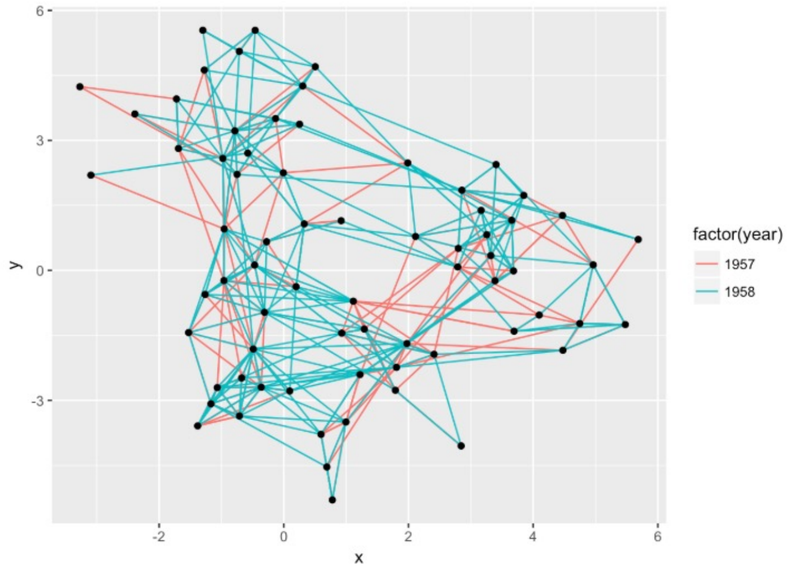
GGRAPH 1.0.2.9999 | Reference | Getting Started | Articles | News

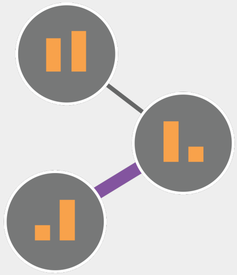
ggraph

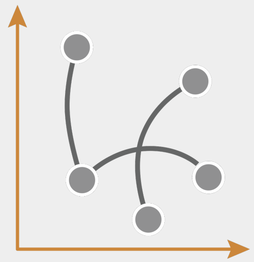
/dʒiː.dʒɪˈrɑːf/ (or g-giraffe)

A grammar of graphics for relational data

ggraph is an extension of `ggplot2` aimed at supporting relational data structures such as networks, graphs, and trees. While it builds upon the foundation of `ggplot2` and its API it comes with its own self-contained set of geoms, facets, etc., as well as adding the concept of *layouts* to the grammar.







developer

plotly | Graphing Libraries

DEMO DASH

Help | Open Source Graphing Libraries | Python | Scientific | Network Graphs

Edit this page on GitHub

Navigation

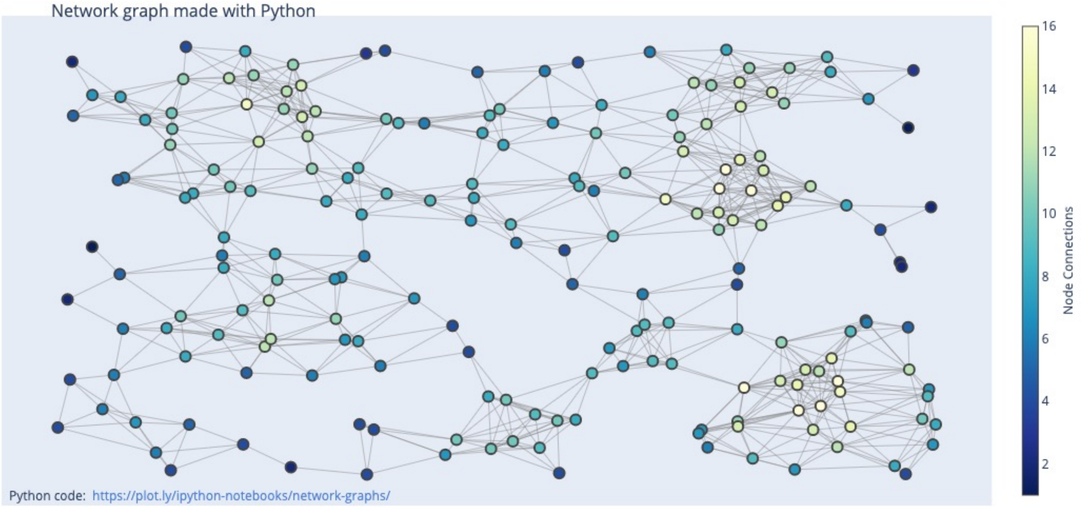
- Create random graph
- Create Edges
- Color Node Points
- Create Network Graph
- Dash Example
- Reference

Back To Python

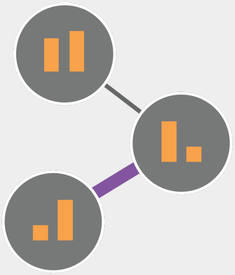
Create Network Graph

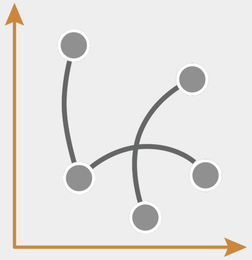
```
fig = go.Figure(data=[edge_trace, node_trace],
             layout=go.Layout(
                title='<br>Network graph made with Python',
                titlefont_size=16,
                showlegend=False,
                hovermode='closest',
                margin=dict(b=20,l=5,r=5,t=40),
                annotations=[ dict(
                    text="Python code: <a href='https://plot.ly/ipython-notebooks/network-graphs/'> https://plot.ly/ipython-notebooks/network-graphs/</a>",
                    showarrow=False,
                    xref="paper", yref="paper",
                    x=0.005, y=-0.002 ) ],
                xaxis=dict(showgrid=False, zeroline=False, showticklabels=False),
                yaxis=dict(showgrid=False, zeroline=False, showticklabels=False))
                )
fig.show()
```

Network graph made with Python



Python code: https://plot.ly/ipython-notebooks/network-graphs/





developer

NetworkX

Stable (notes)

2.3 — April 2019
download | doc | pdf

Latest (notes)

2.4 development
github | doc | pdf

Archive

Contact

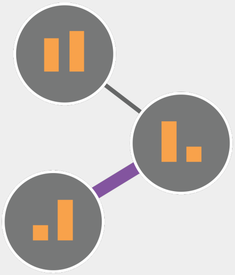
Mailing list
Issue tracker

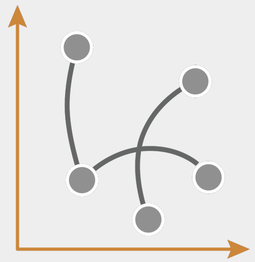
Software for complex networks

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

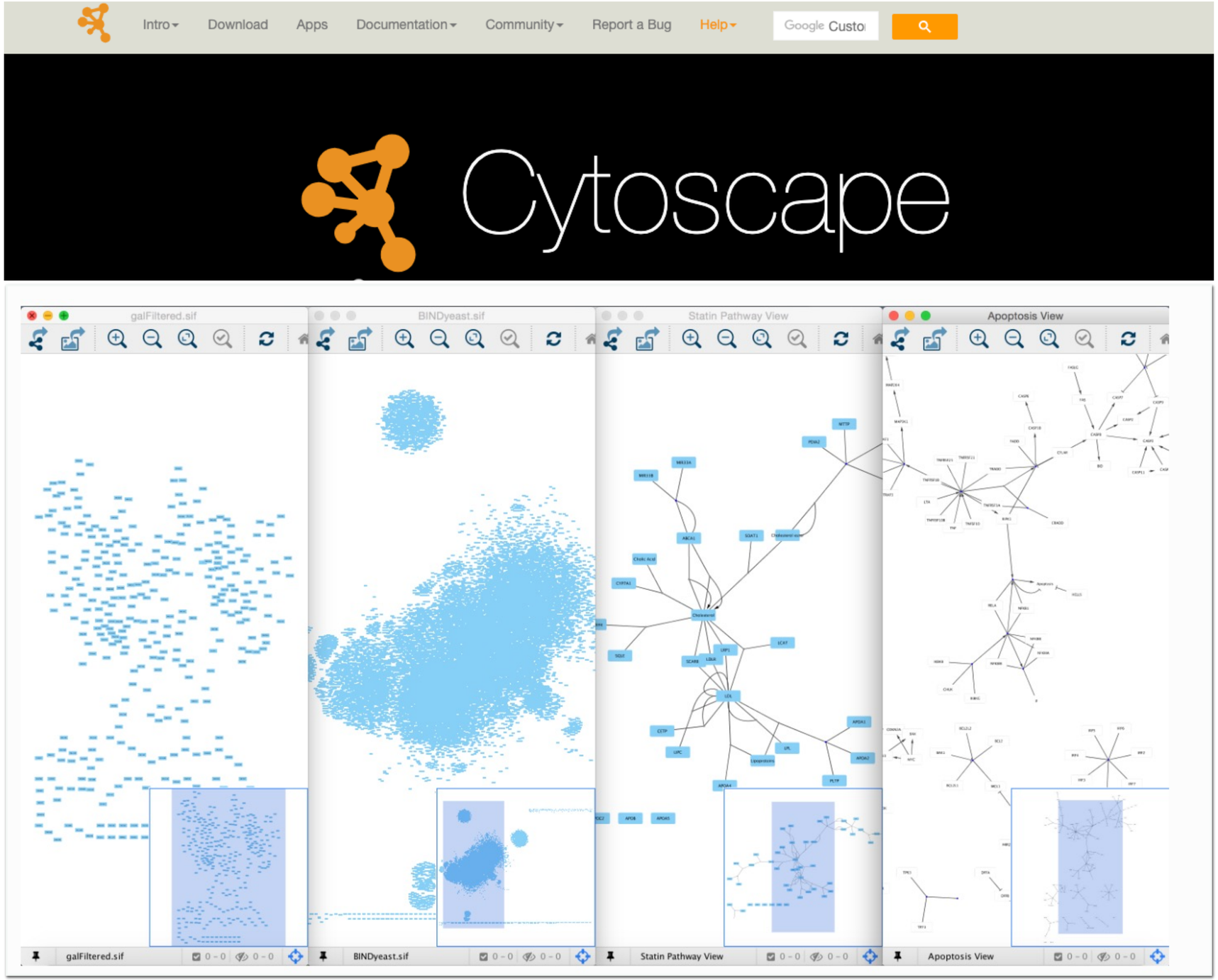
Features

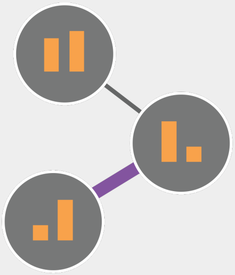
- Data structures for graphs, digraphs, and multigraphs
- Many standard graph algorithms
- Network structure and analysis measures
- Generators for classic graphs, random graphs, and synthetic networks
- Nodes can be "anything" (e.g., text, images, XML records)
- Edges can hold arbitrary data (e.g., weights, time-series)
- Open source 3-clause BSD license
- Well tested with over 90% code coverage
- Additional benefits from Python include fast prototyping, easy to teach, and multi-platform

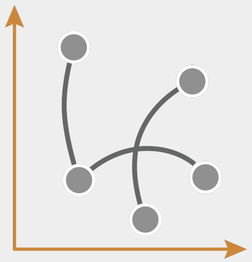




graphic designer







graphic designer

Gephi
makes graphs handy

Download | Blog | Wiki | Forum | Support | Bug tracker

Home | Features | Learn | Develop | Plugins | Services | Consortium

The Open Graph Viz Platform

Gephi is the leading visualization and exploration software for all kinds of graphs and networks. Gephi is open-source and free.

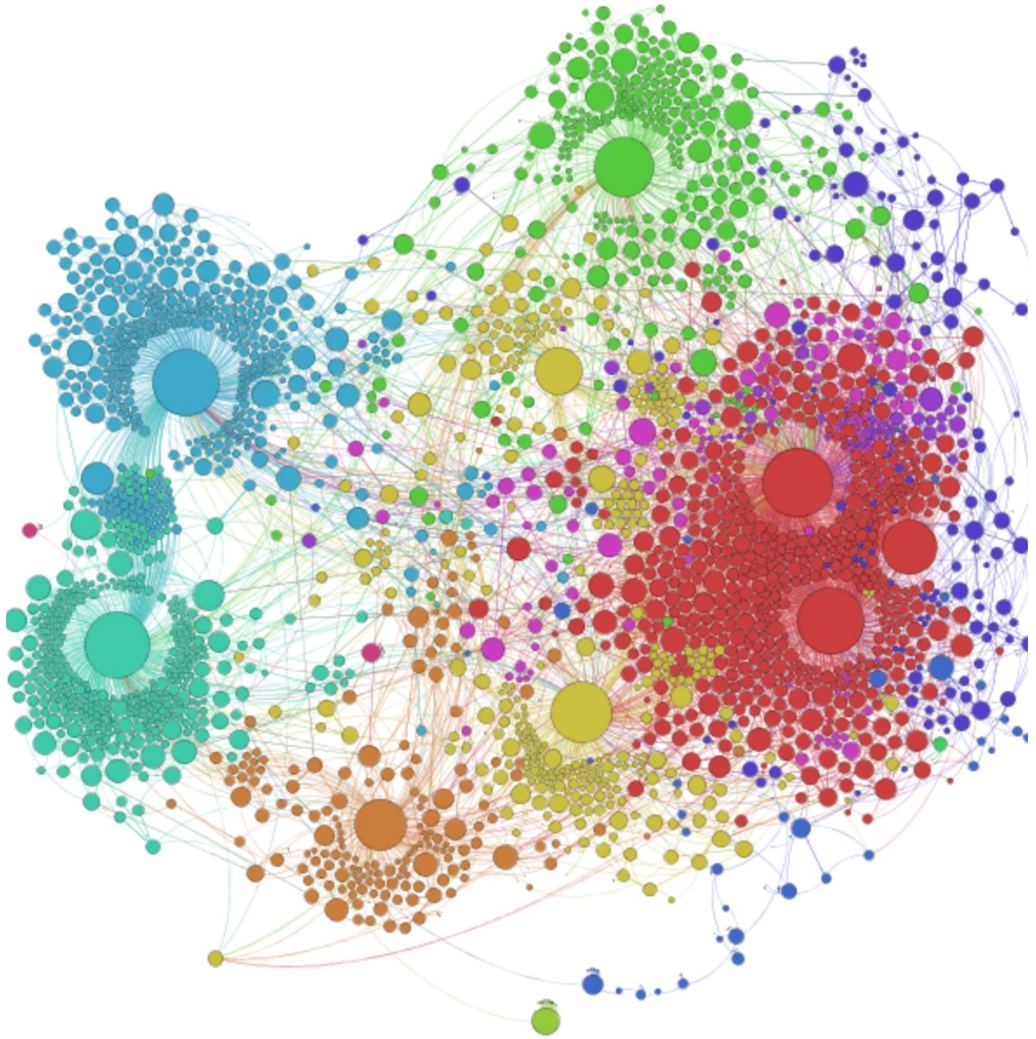
Runs on Windows, Mac OS X and Linux.

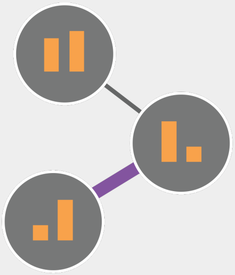
Learn More on Gephi Platform »

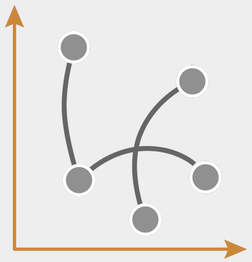
Download FREE
Gephi 0.9.2

Release Notes | System Requirements

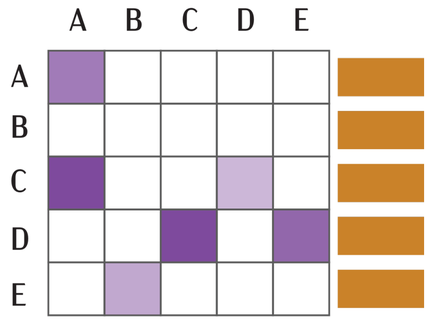
- **Features**
- **Quick start**
- **Screenshots**
- **Videos**



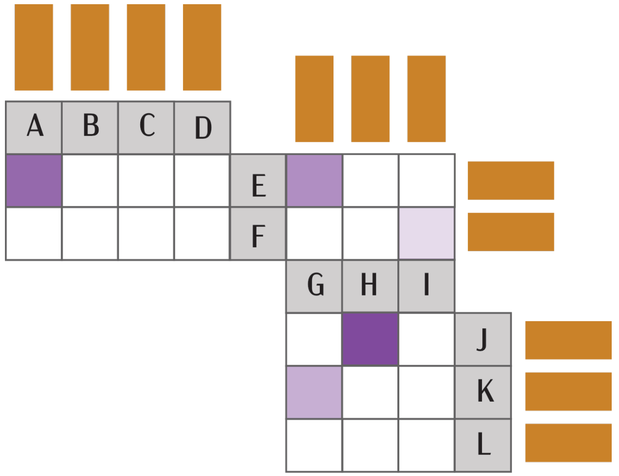




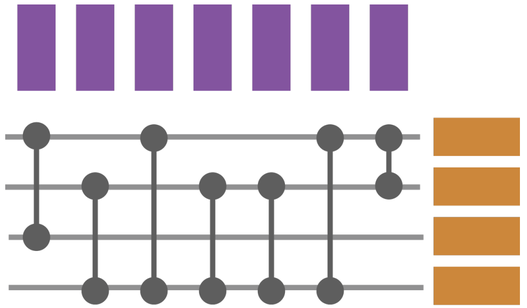
Tabular Layouts



Adjacency Matrix

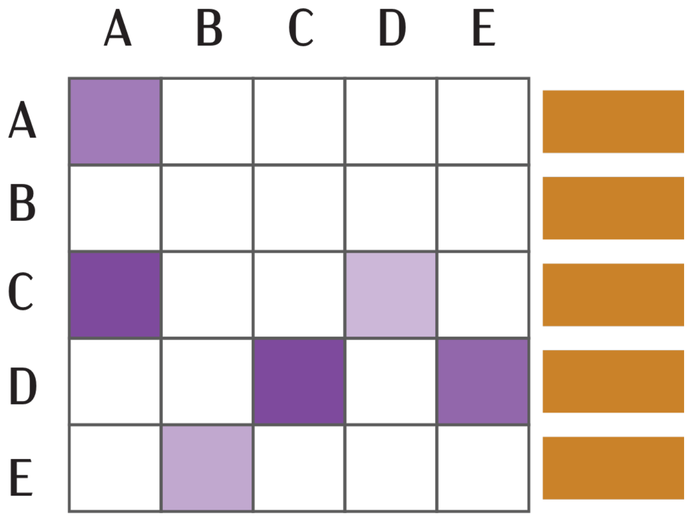


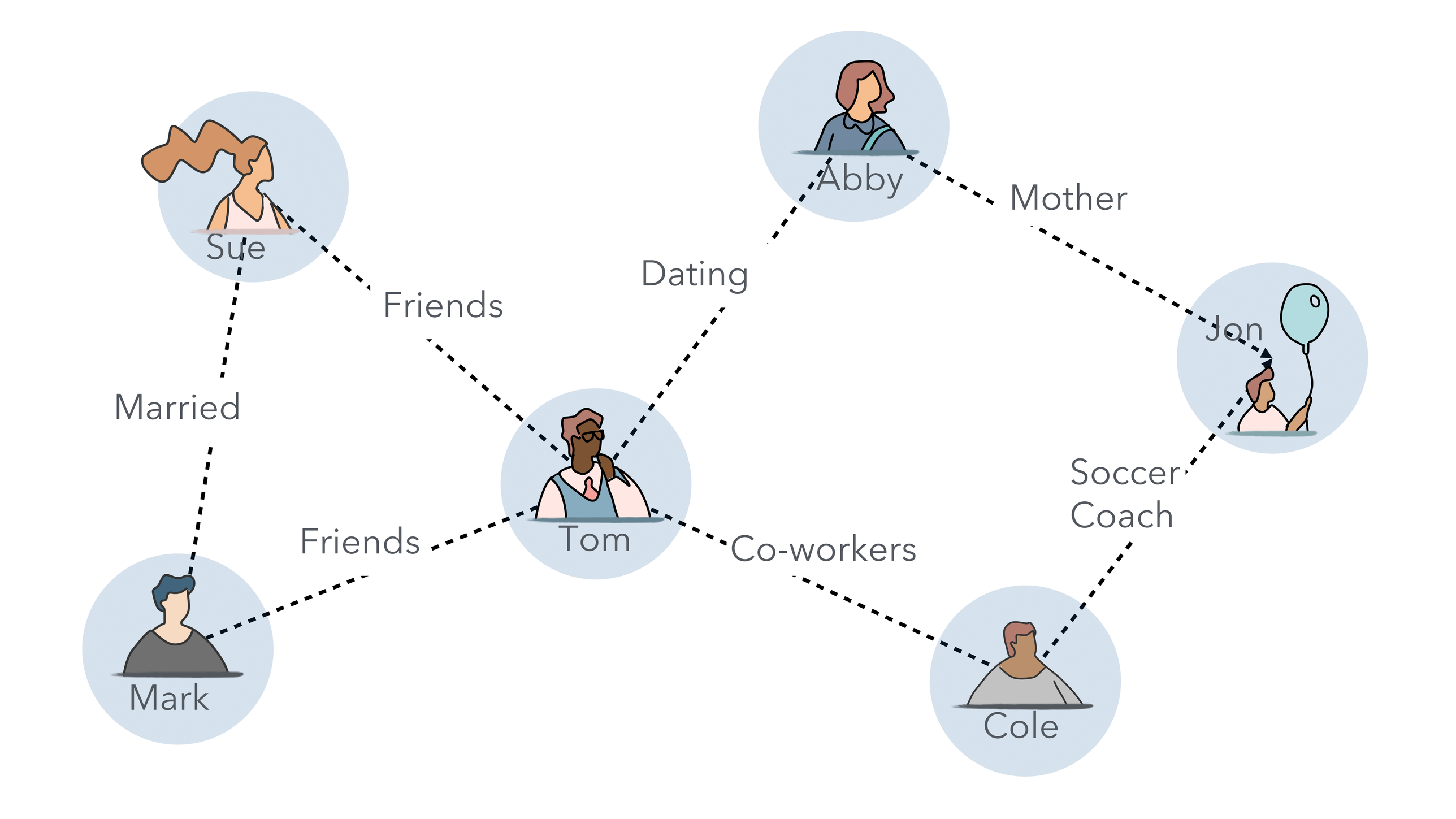
Quilts

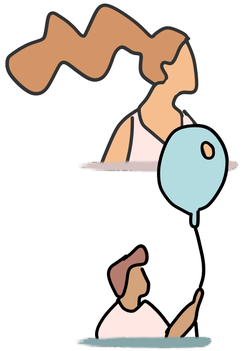


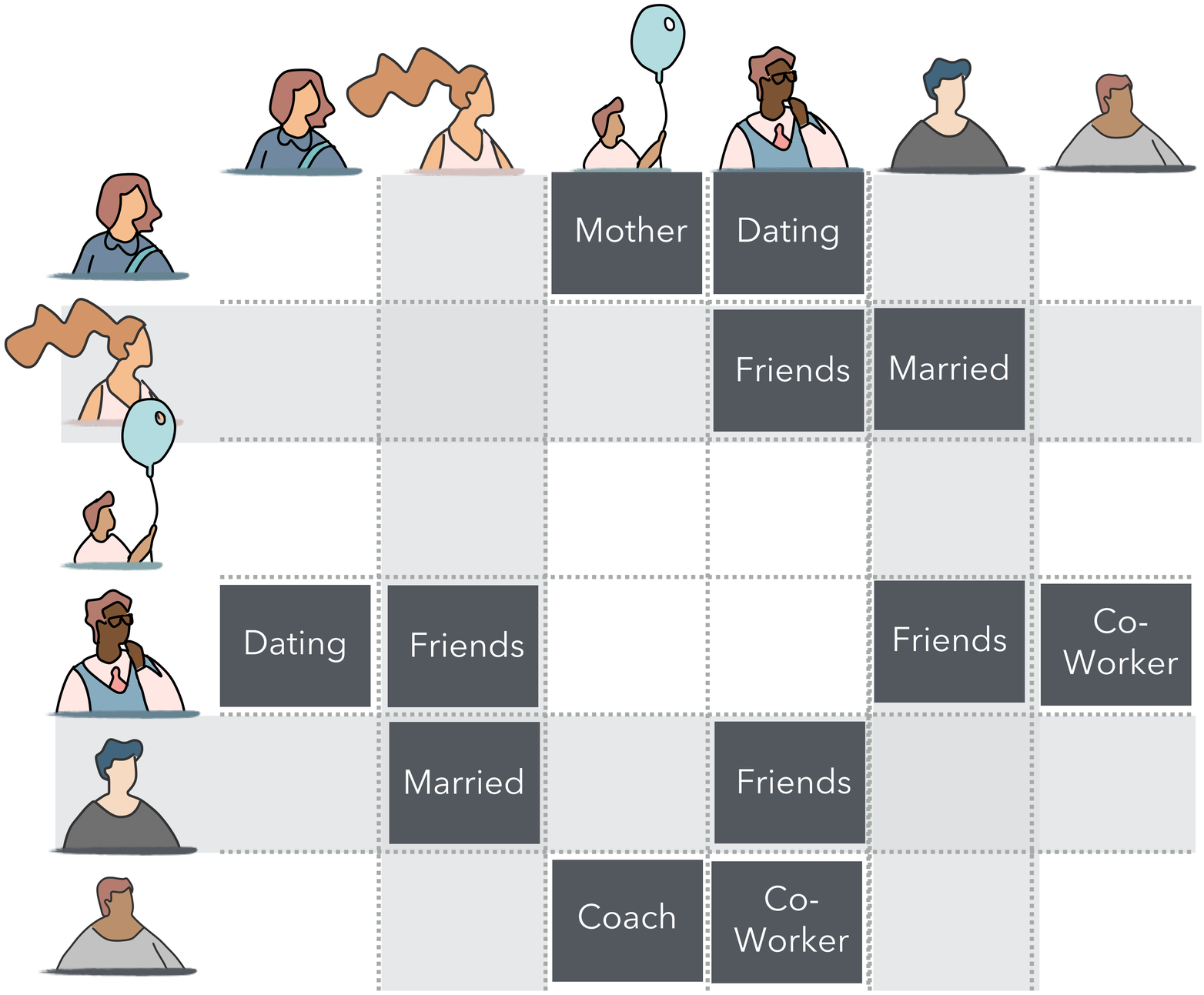
BioFabric

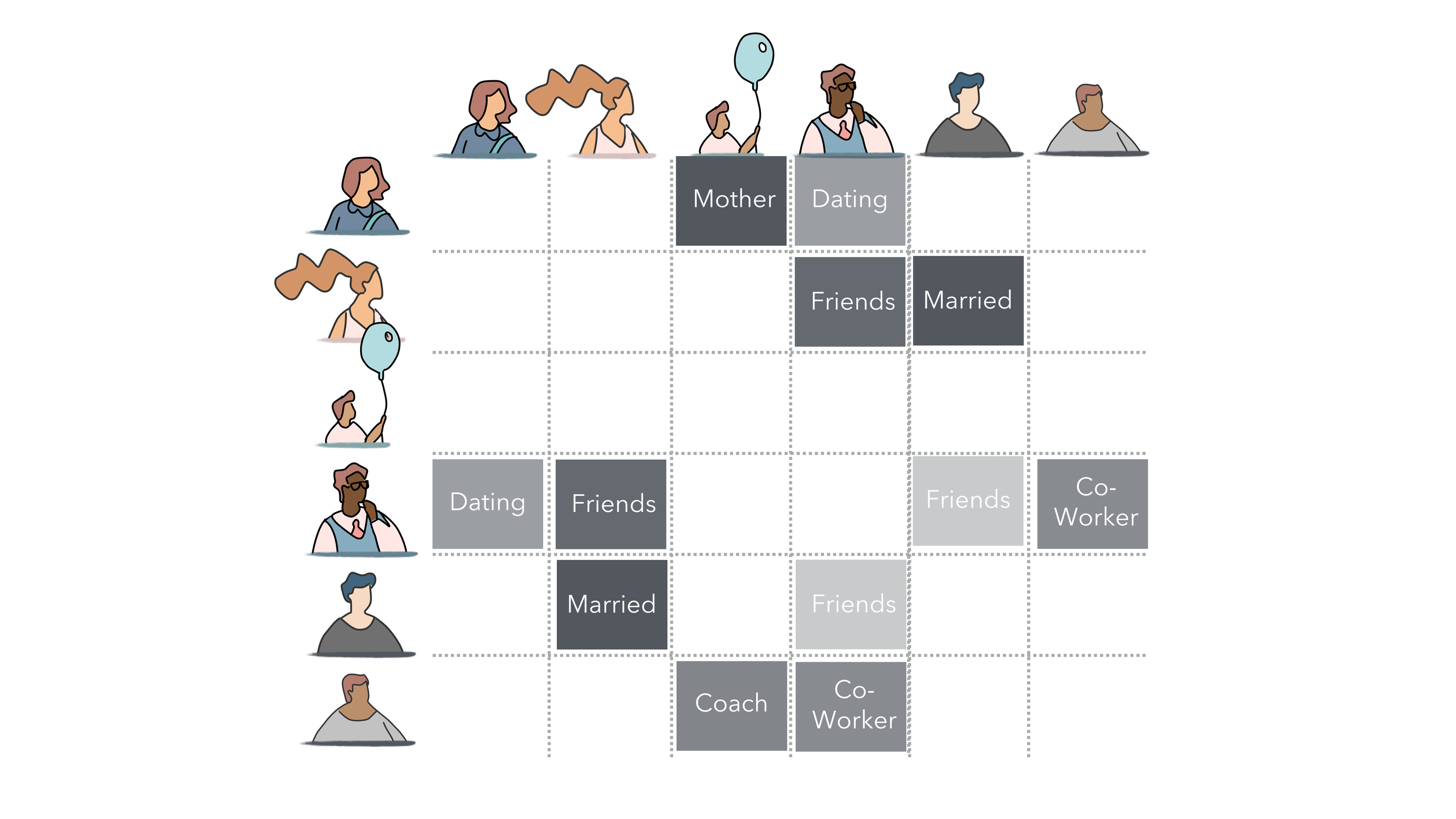
Adjacency Matrix

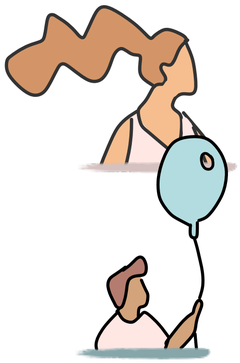



Sue
Abby
Mother
Friends
Dating
Jon
Married
Soccer
Coach
Tom
Friends
Co-workers
Mark
Cole

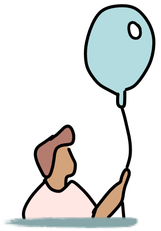



Mother
Dating
Friends
Married
Dating
Friends
Friends
Co-
Worker
Married
Friends
Coach
Co-
Worker


Mother
Dating
Friends
Married
Dating
Friends
Friends
Co-Worker
Married
Friends
Coach
Co-Worker

			Mother	Dating		
				Friends	Married	
	Dating	Friends			Friends	Co-Worker
		Married		Friends		
			Coach	Co-Worker		

Name	Beverage	Day 1
Abby	Port	1
Sue	Coke	0
Jon	Coke	4
Tom	Beer	5
Mark	Beer	2
Cole	Port	3

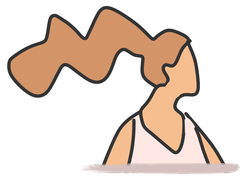





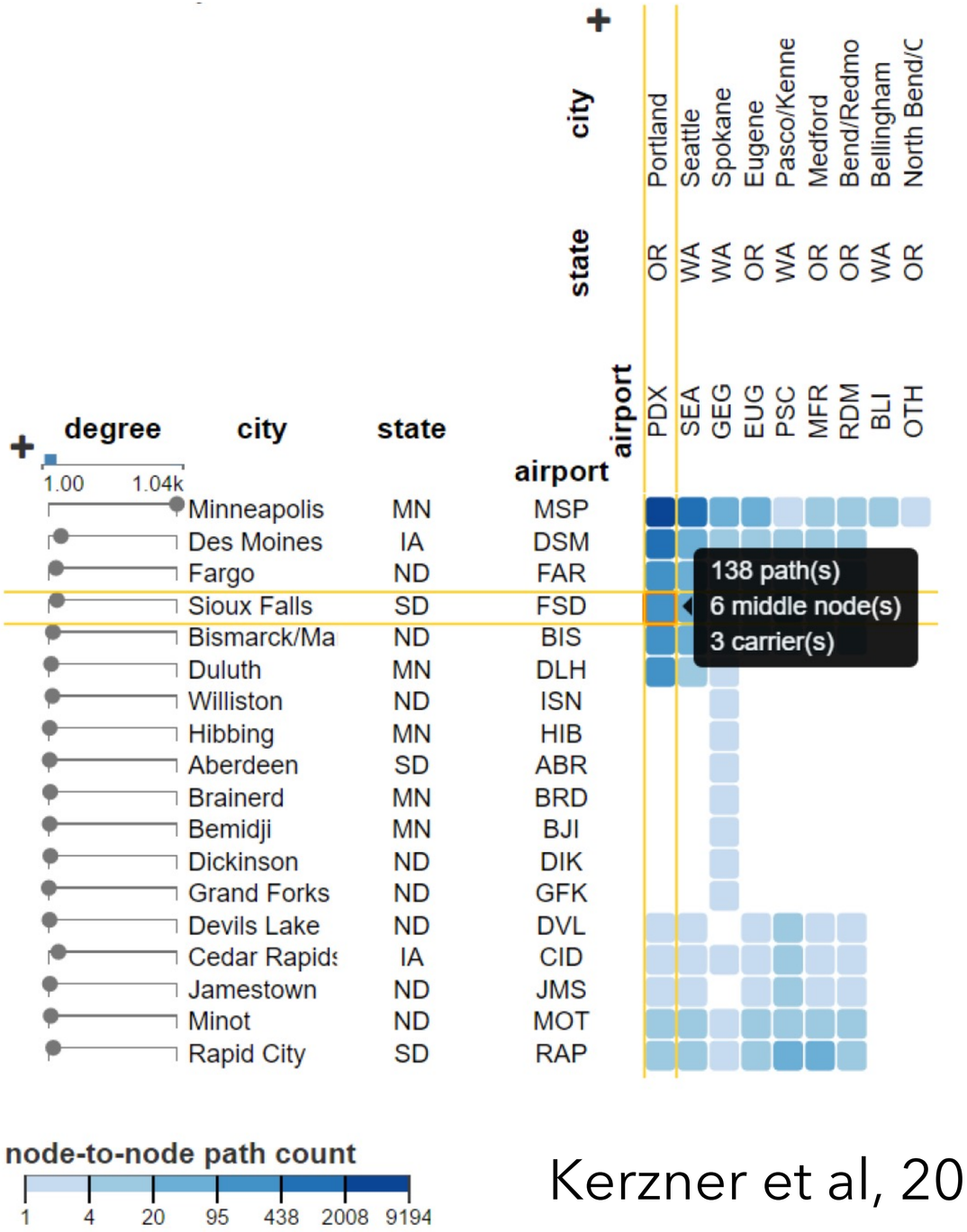
		Co-Worker	Friends	Dating	Friends
Co-Worker	Coach				
Friends					Married
Dating	Mother				
Friends			Married		



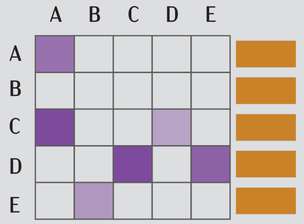




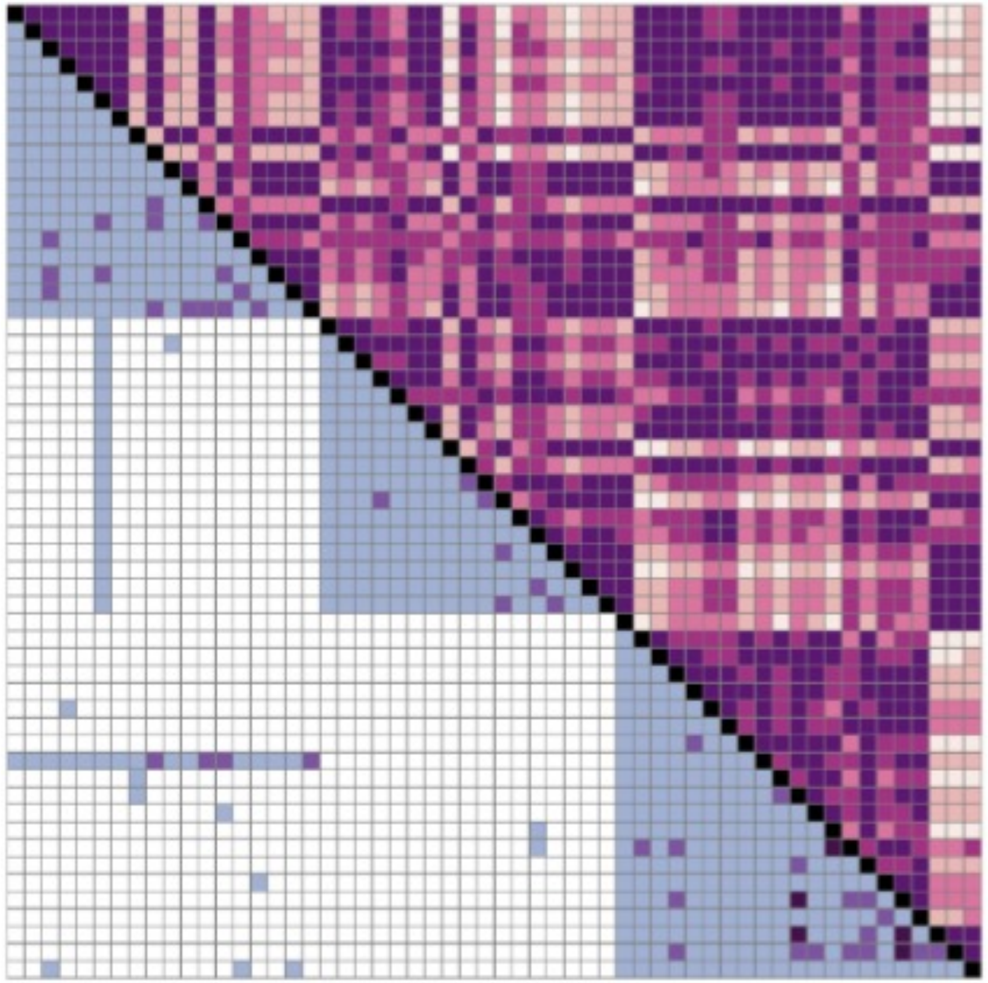
Name	Beverage	Day 1
Tom	Beer	5
Jon	Coke	4
Cole	Port	3
Mark	Beer	2
Abby	Port	1
Sue	Coke	0



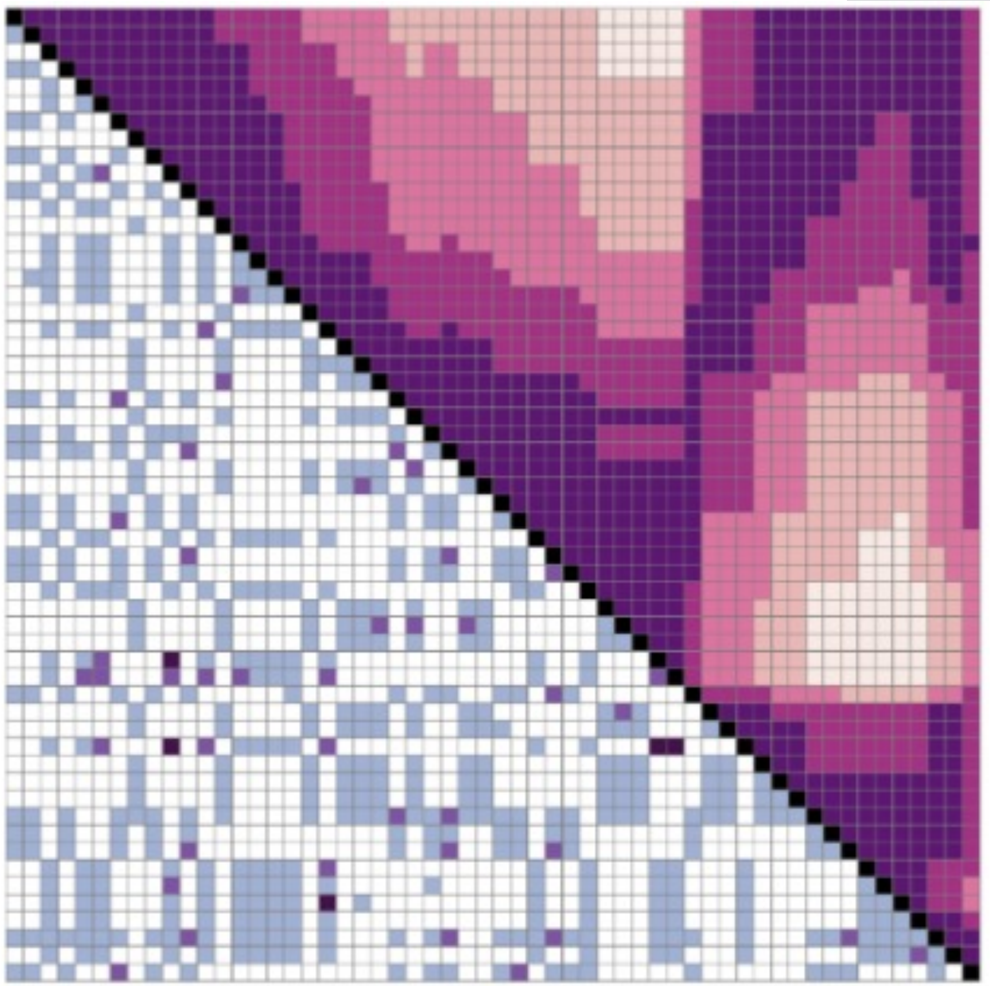
Kerzner et al, 2017



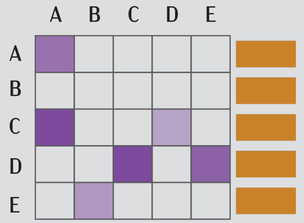
Adjacency Matrix



(a) Sorted by structure.



(b) Sorted by attribute similarity.

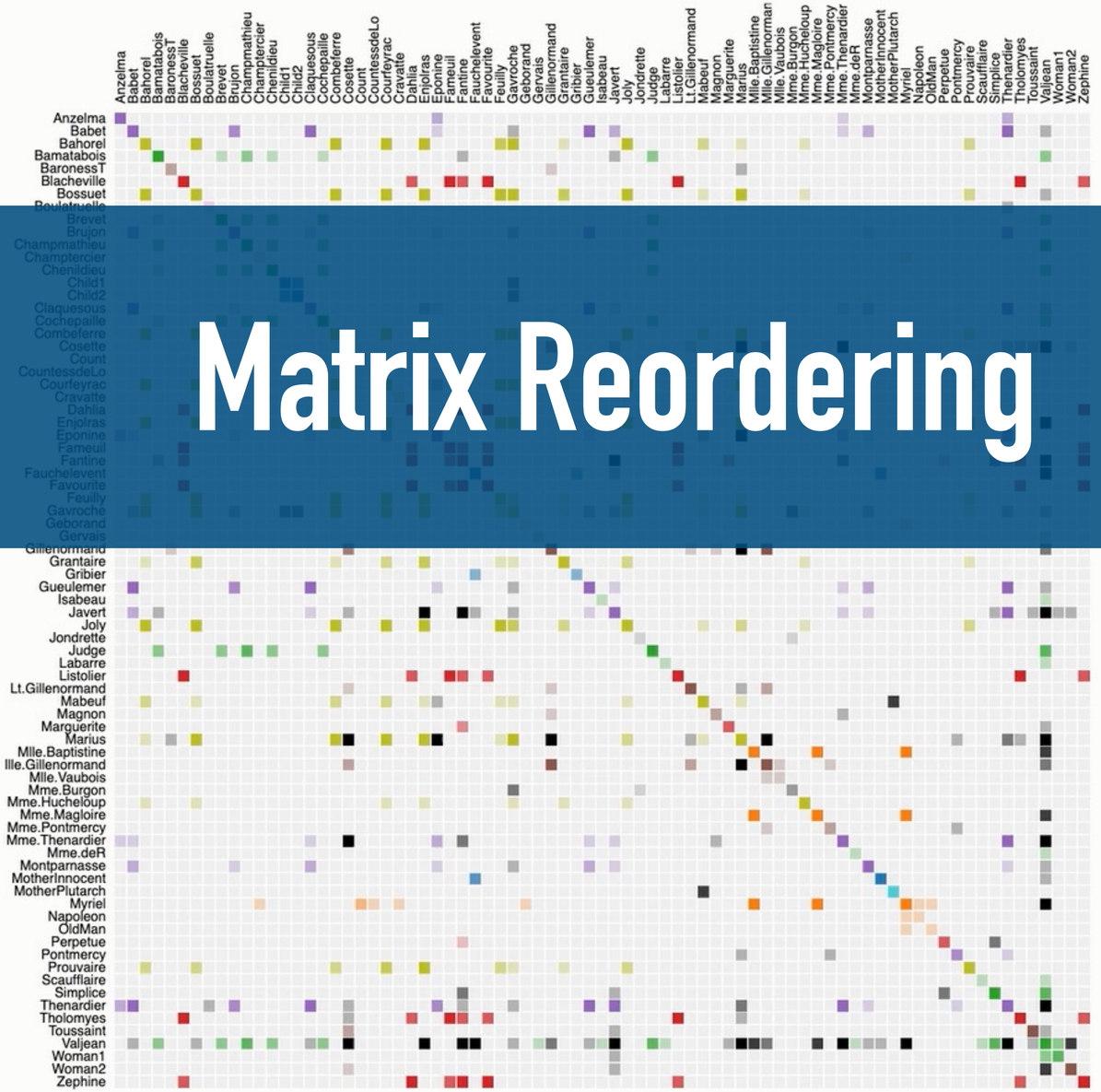


Adjacency Matrix

Berger et al, 2019

April 10, 2012 / Mike Bostock

Les Misérables Co-occurrence



Order: by Name

This matrix diagram visualizes character co-occurrences in Victor Hugo's *Les Misérables*.

Each colored cell represents two characters that appeared in the same chapter; darker cells indicate characters that co-occurred more frequently.

Use the drop-down menu to reorder the matrix and explore the data.

Built with d3.js.

Source: The Stanford GraphBase.

Matrix Reordering

Code | Issues 2 | Pull requests 1 | Projects 0 | Wiki | Security | Insights

Home

Edit | New Page

Jean-Daniel Fekete edited this page on Apr 23, 2015 · 2 revisions

Reorder.js is a library to reorder tables and graph/networks.

Resources

- Introduction
- API Reference

Browser / Platform Support

Reorder.js is mainly developed on Chrome and Node.js. Use `npm install reorder.js` to install, and `require("reorder")` to load.

Installing

Download the latest version here:

- https://github.com/jdfekete/reorder.js/releases

+ Add a custom footer

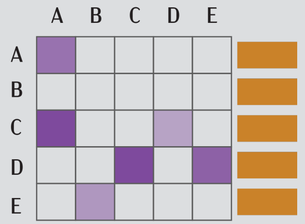
Pages 12

Find a Page...

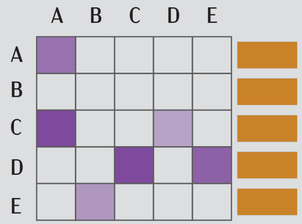
- Home
- API Reference
- Conversion
- Core
- Gallery
- Graph
- Introduction
- LinearAlgebra
- Matrix
- Measure
- Permutation
- Reordering

+ Add a custom sidebar

Reorder.js



Adjacency Matrix



Adjacency Matrix

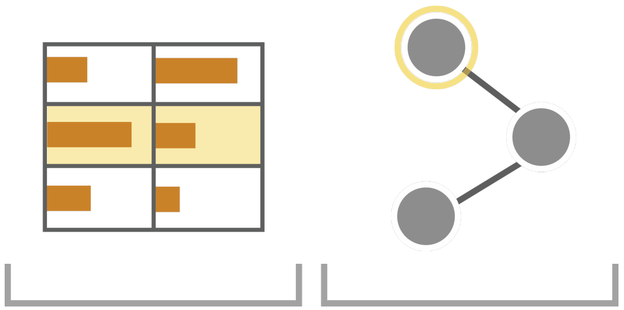
Ideal for dense and completely connected networks

Requires quadratic space with respect to the number of nodes.

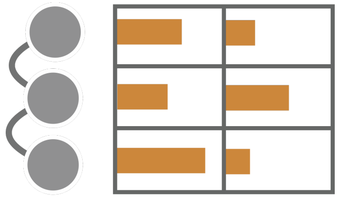
Complexity of choosing the right reordering algorithm

Recommended for smaller, complex and dense networks with rich node and/or edge attributes, for all tasks except for those involving paths

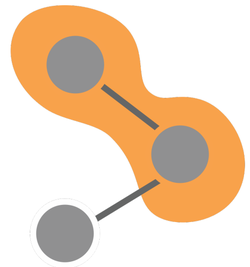
View Operations



Juxtaposed

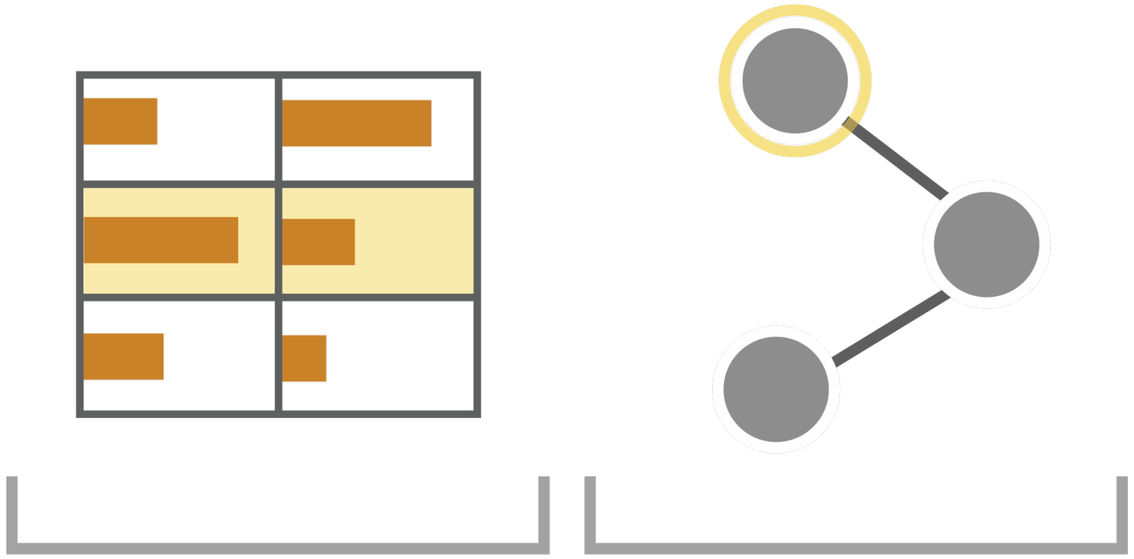


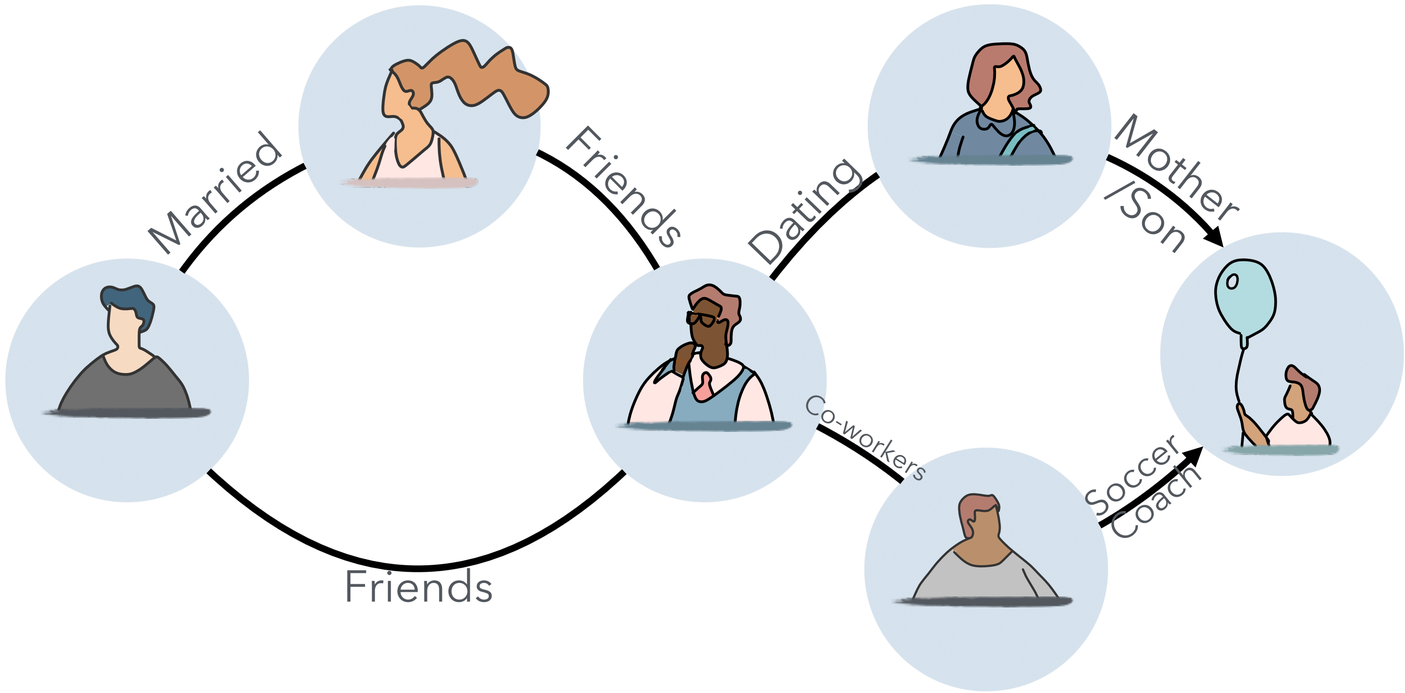
Integrated

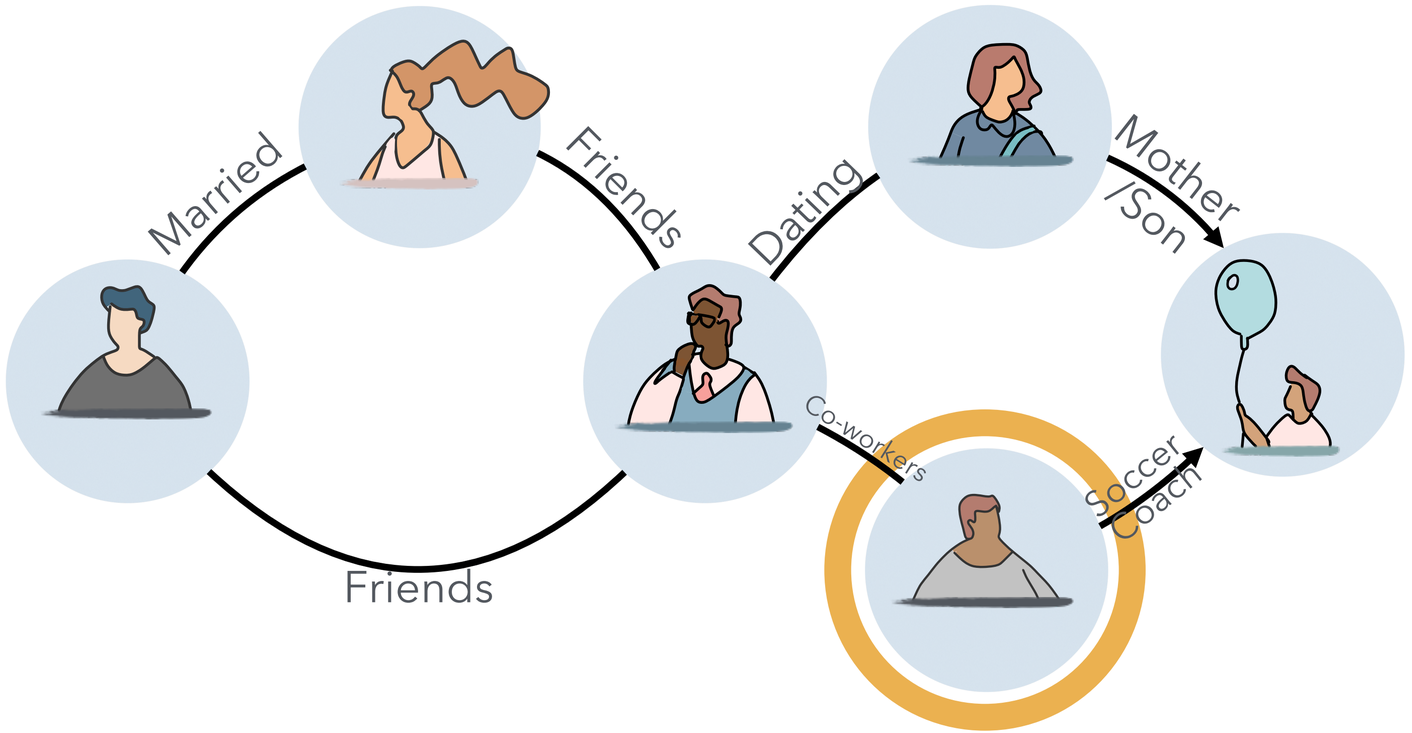


Overloaded

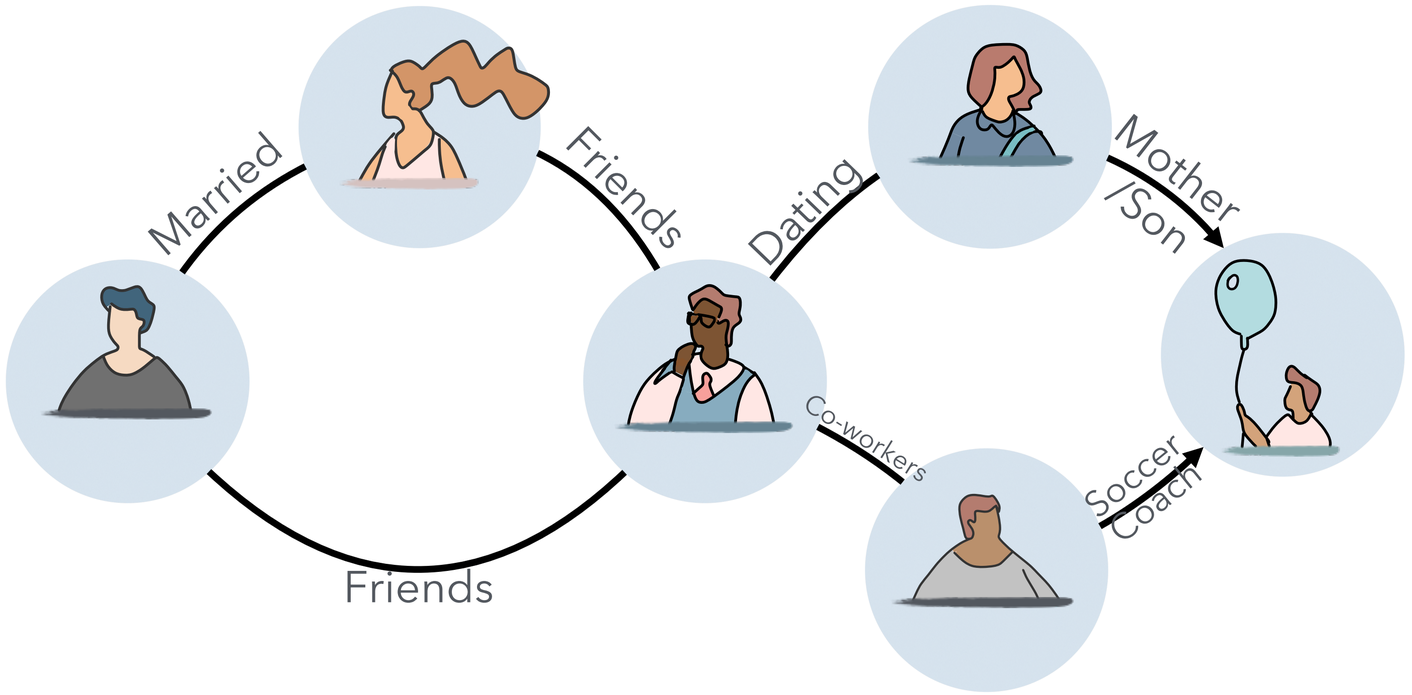
Juxtaposed




Married
Friends
Dating
Mother
/Son
Co-workers
Soccer
Coach
Friends

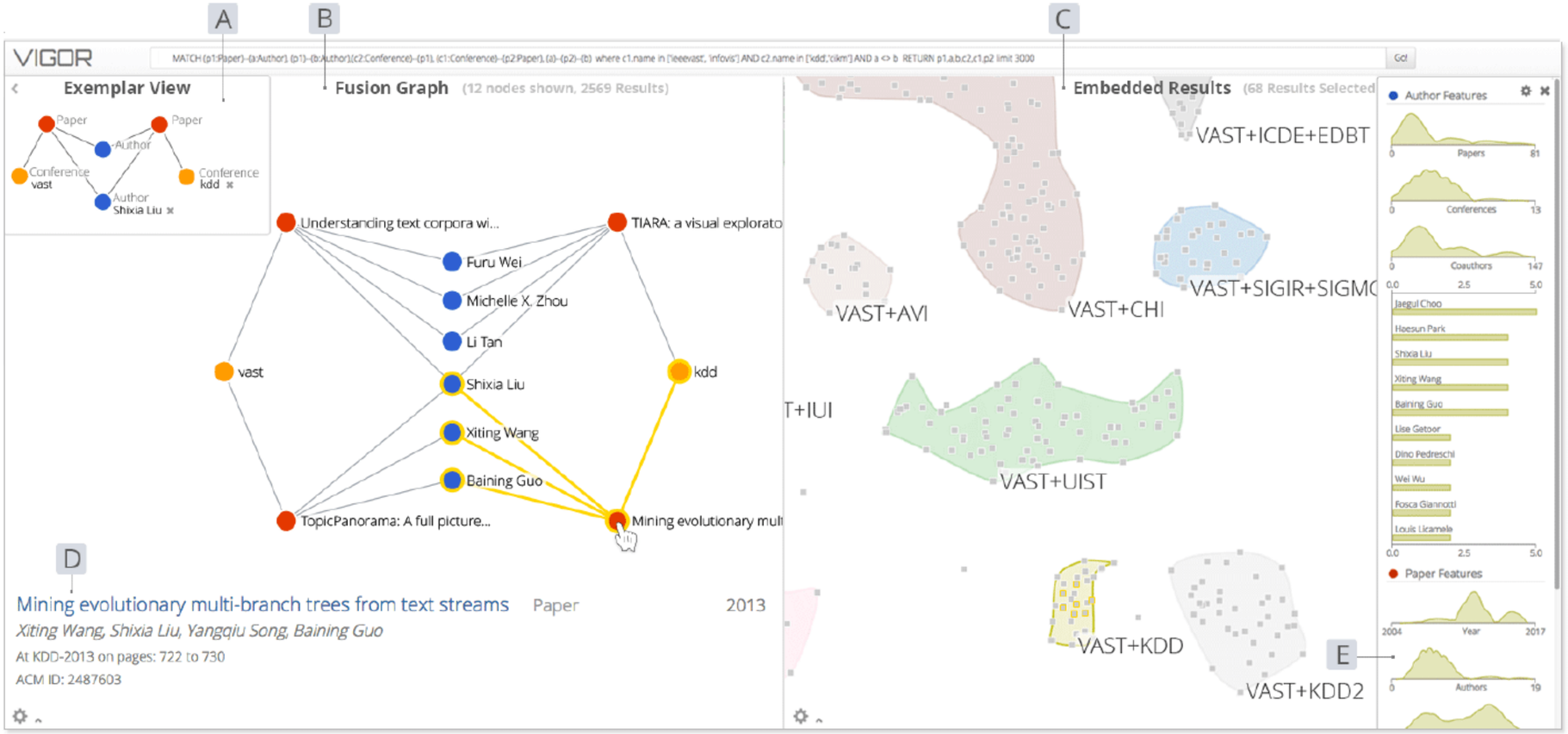


Name	Beverage	Day 1
Mark	Beer	1
Sue	Coke	0
Cole	Port	4
Jon	Coke	5
Tom	Beer	2
Abby	Port	3

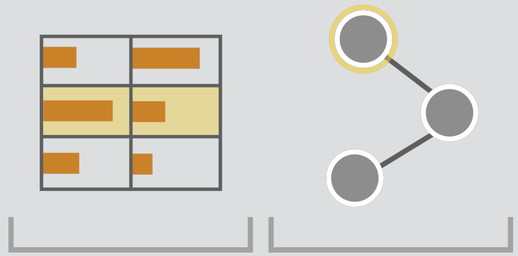


Name	Beverage	Day 1
Mark	Beer	1
Sue	Coke	0
Cole	Port	4
Jon	Coke	5
Tom	Beer	2
Abby	Port	3

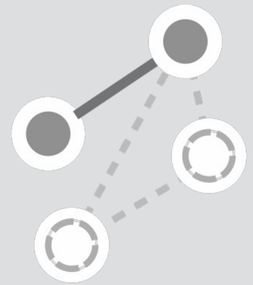
Relationship	Years
Dating	4
Mother / Son	12
Co-workers	3
Soccer Coach	2
Friends	8
Friends	3
Married	4



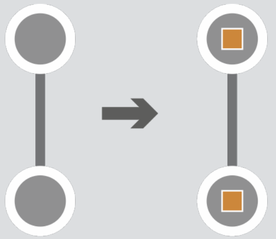
VIGOR *Pienta et al. 2018*



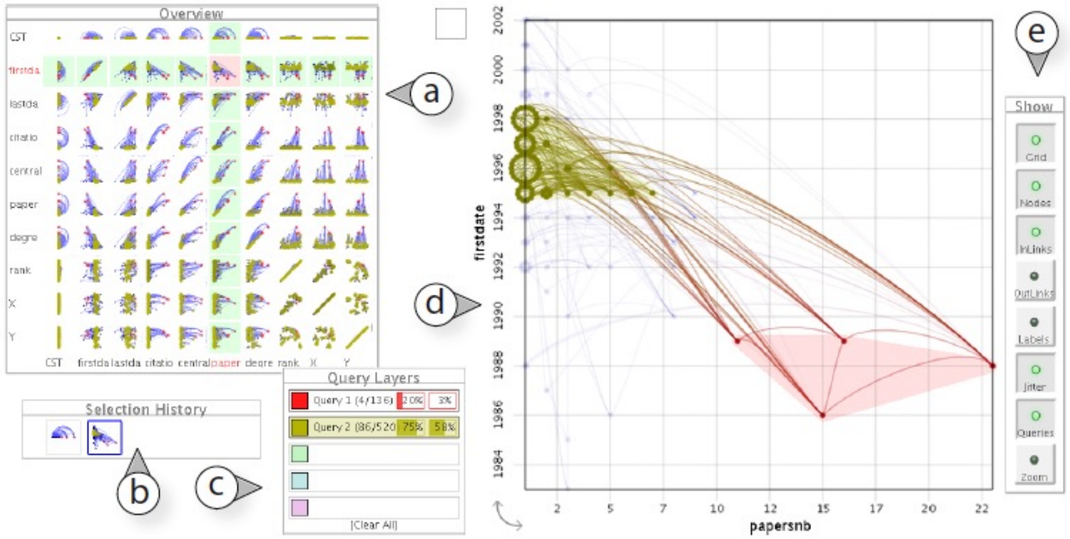
Juxtaposed

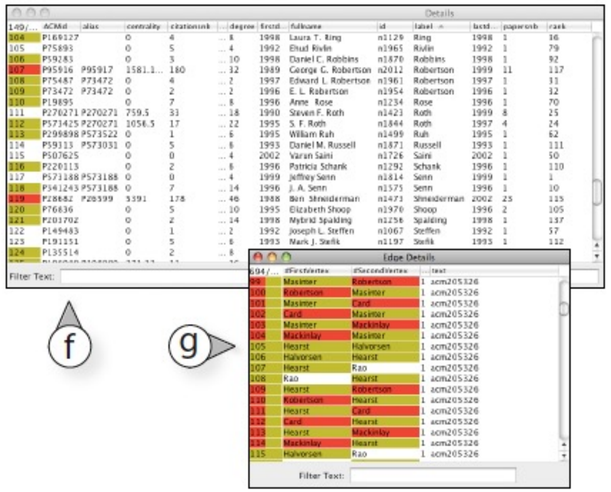


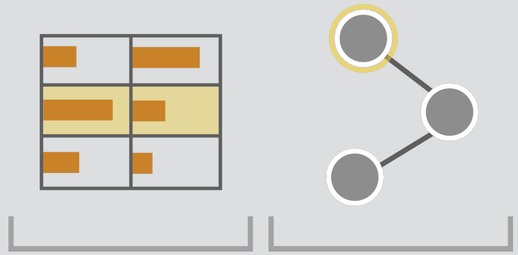
Querying and Filtering



Deriving New Attributes



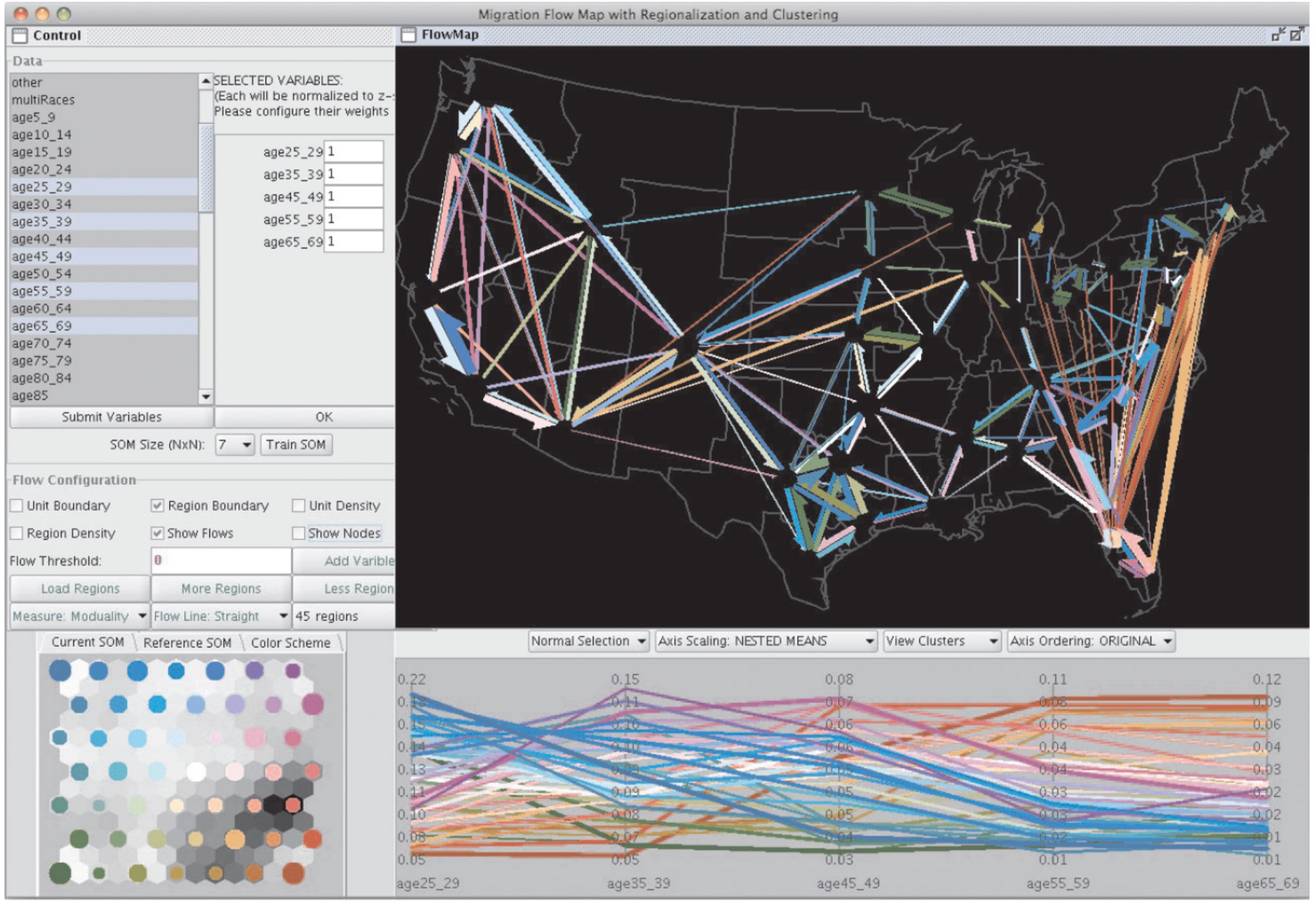


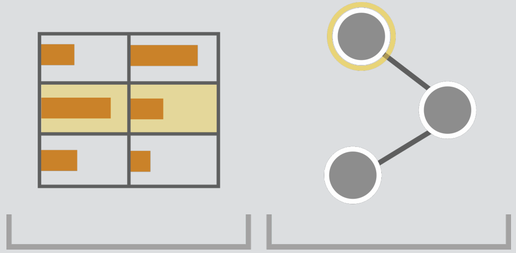


Juxtaposed

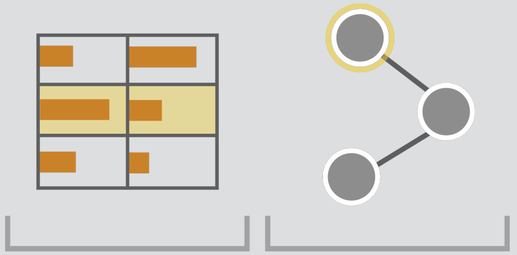
Graph Dice *Bezerianos et al. 2010*

Guo, 2009



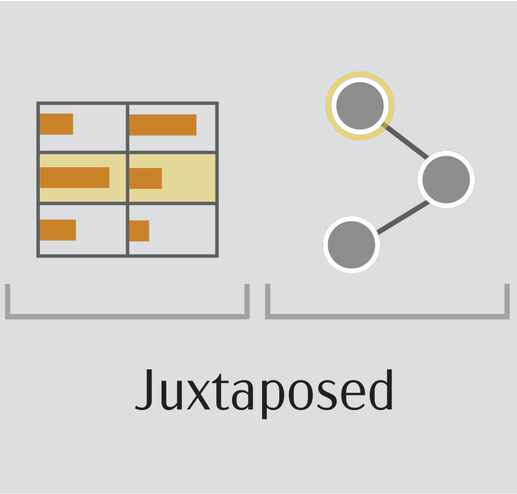


Juxtaposed



Juxtaposed

Independent views can optimize for topology and attribute independently.

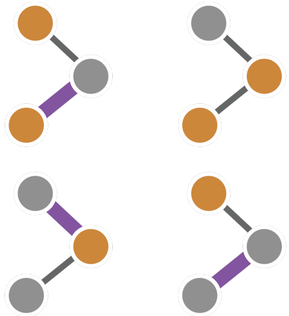


Juxtaposed

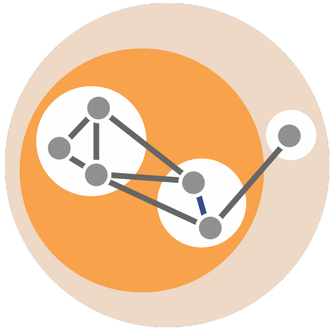
Not great for tasks on topological structures beyond a single node or edge.

Recommended for large networks and/or very large numbers or heterogeneous types of node and link attributes

Layout Operations

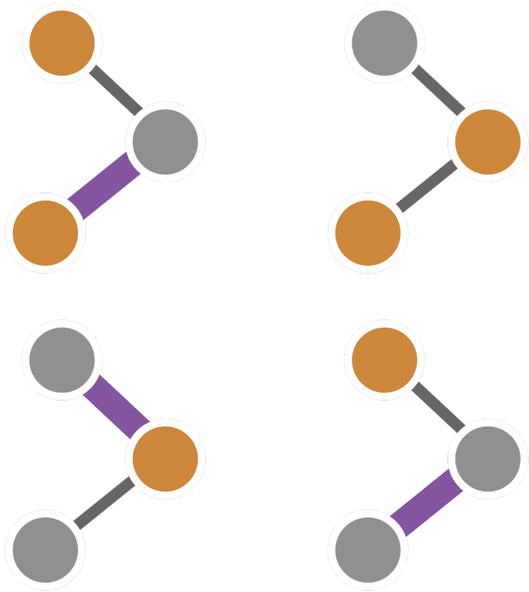


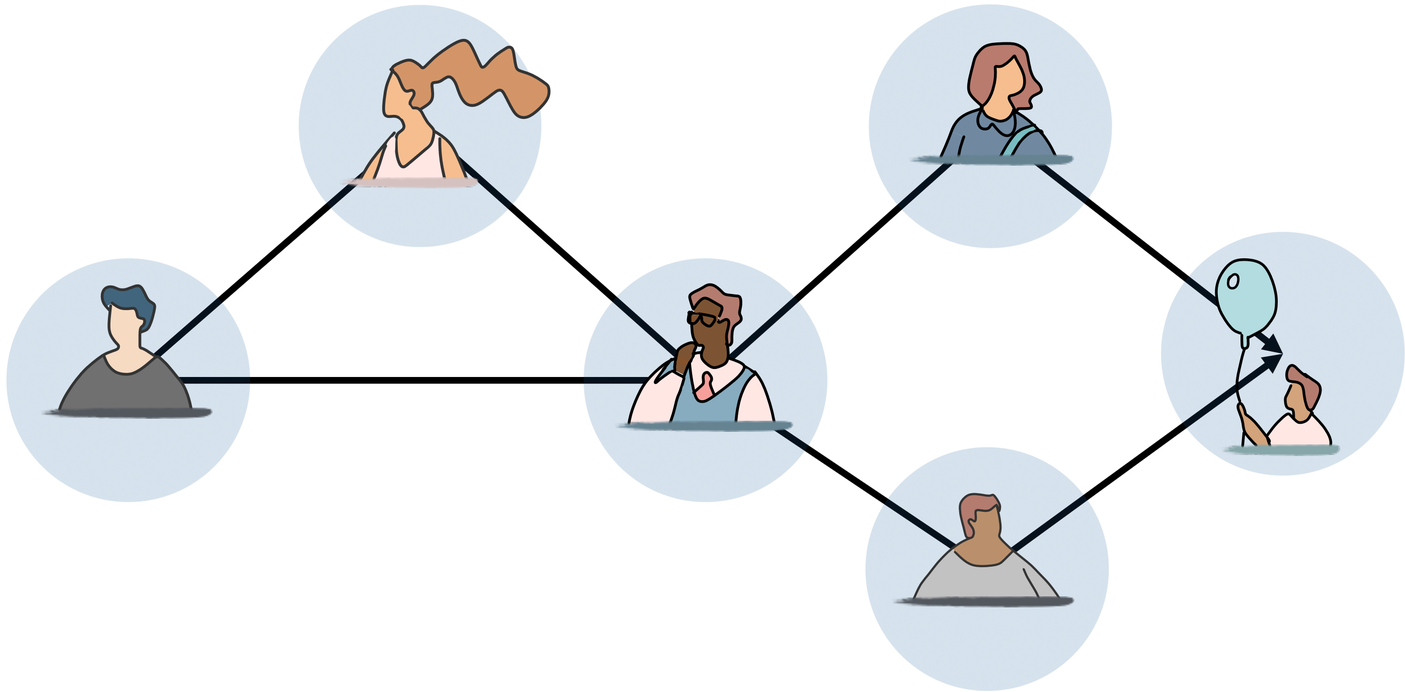
Small Multiples



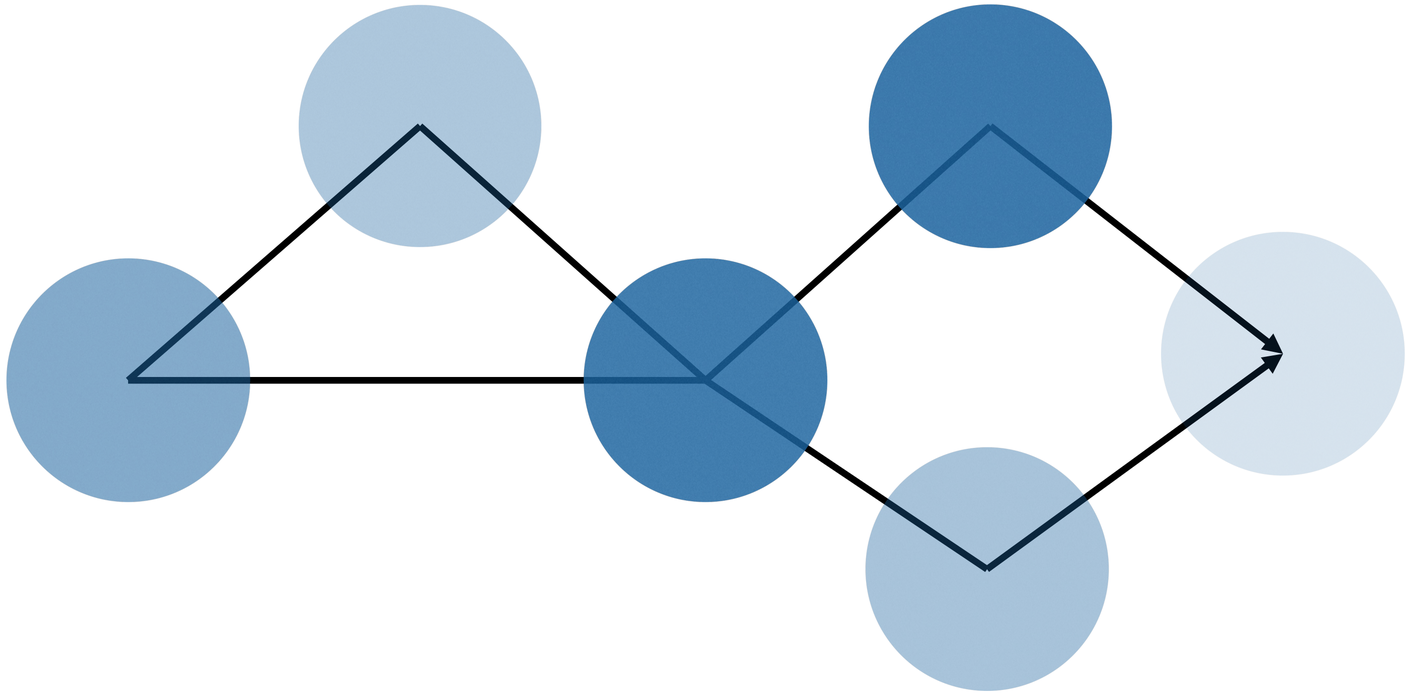
Hybrids

Small Multiples

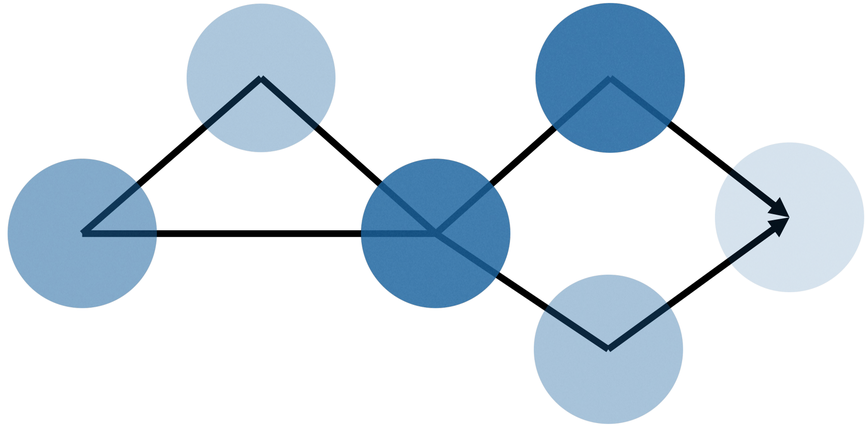




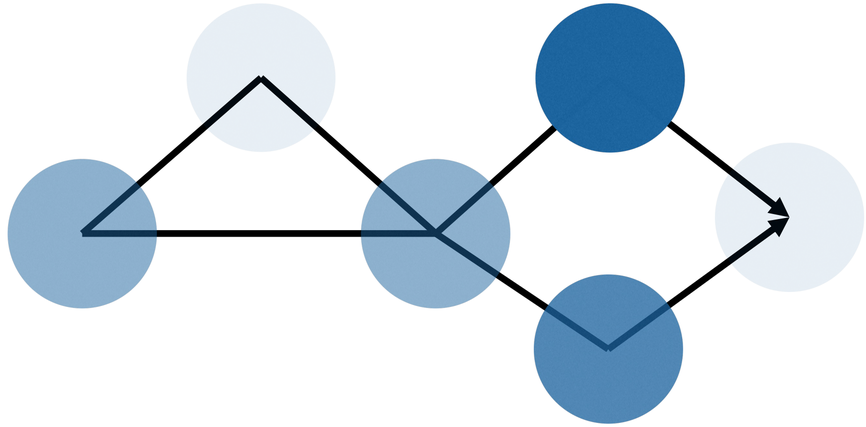
Day 1



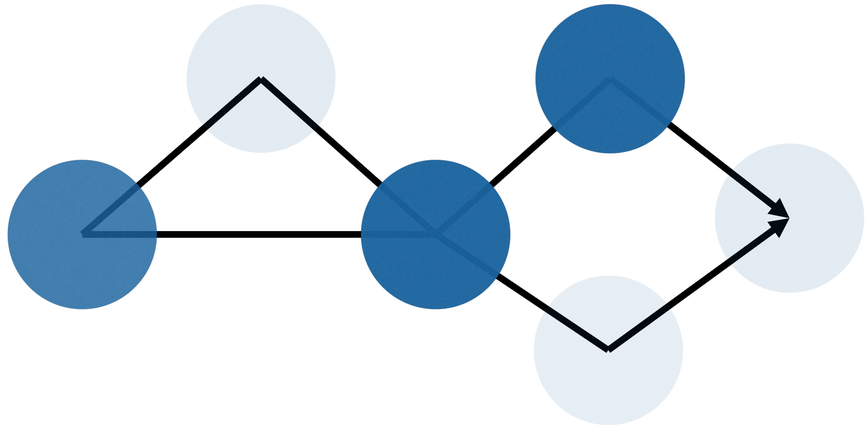
Day 1

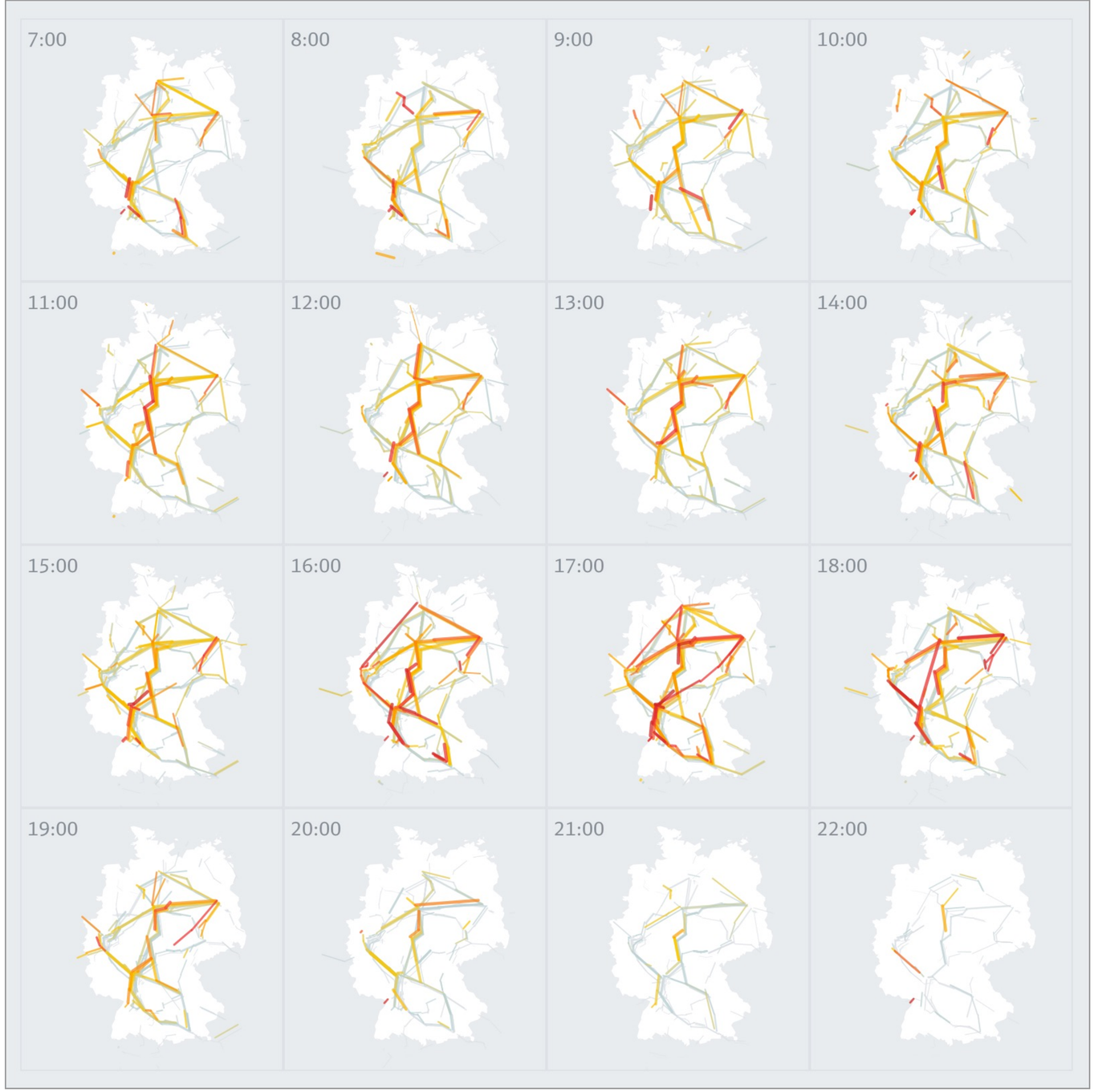


Day 2

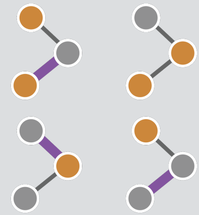


Day 3

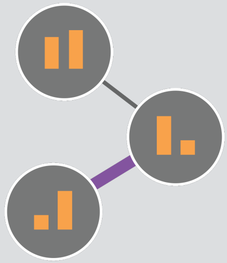




Peakspotting – https://truth-and-beauty.net/projects/peakspotting

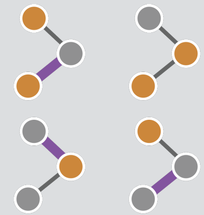


Small Multiples



On-Node / On-Edge Encoding

Small Multiples



Small Multiples

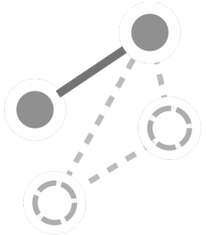
Common layout facilitates attribute comparisons in specific topological features

Not ideal for large networks, or tasks on clusters

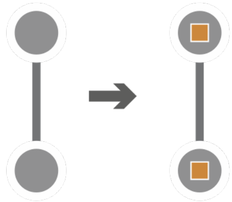
Recommended for small networks where the tasks are focused on attribute comparison

Data Operations

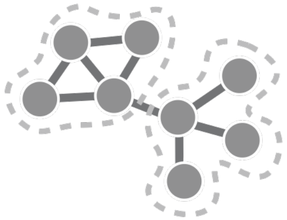
Aggregating Nodes/Edges



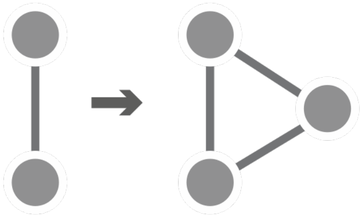
Querying and Filtering



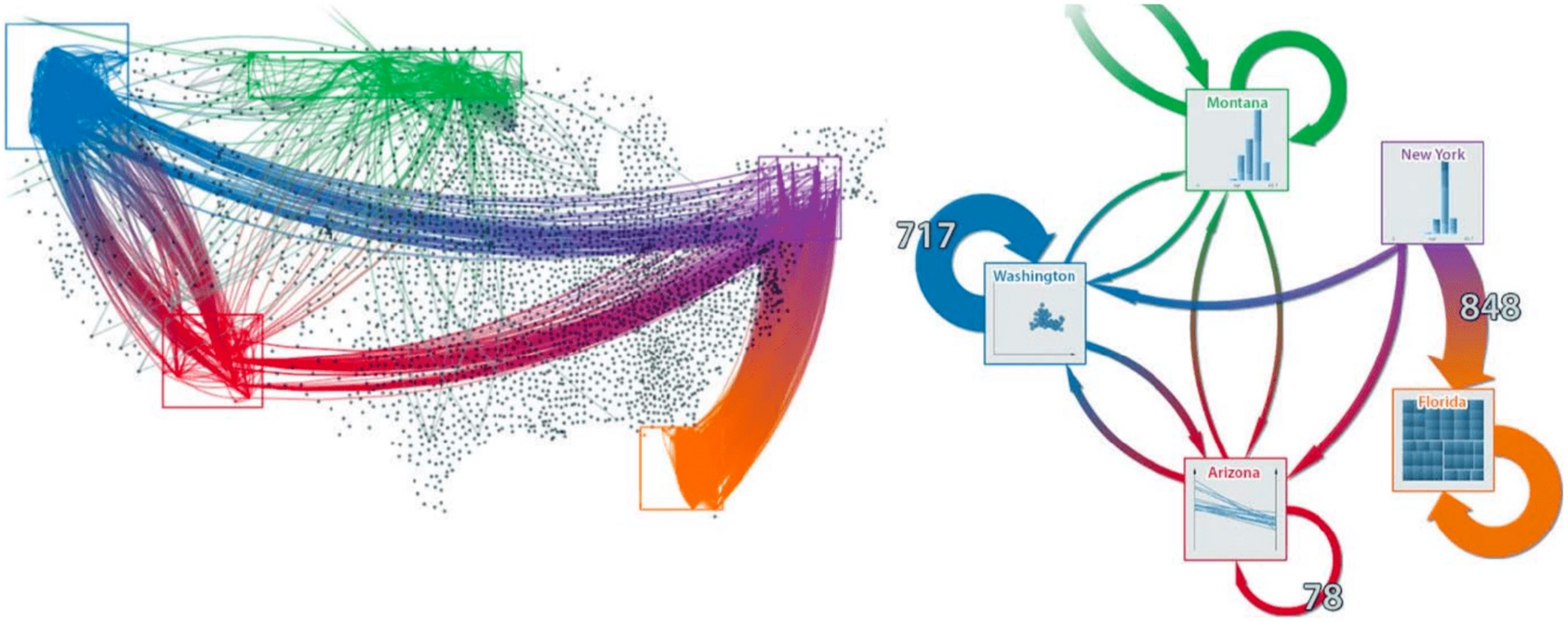
Deriving New Attributes



Clustering



Converting Attributes/Edge to Nodes



Elzen and Wijk, 2014

Aggregating Nodes/Edges

Multivariate Network Visualization Techniques

A companion website for the STAR Report on Multivariate Network Visualization Techniques.

HOME TECHNIQUES WIZARD

About

This is a companion website for a review article on multivariate network visualization techniques.

Multivariate networks are networks where both the structure of the network and the attributes of the nodes and edges matter. It turns out, these are very common. Every person in a social network, for example, has both, relationships and lots of other characteristics, such as their age, the school they went to, or the city they live in. Multivariate network visualization techniques are designed to be able to show both, these attributes and the structure. Using these visualization techniques, we can analyze, for example, if a network of friends predominantly went to the same high school.

The visualization research community has developed many techniques to visualize these kinds of networks, and our review article – and this website – are designed to help you sort through these options.

Browse through the techniques illustrated below, or use our wizard to find the right multivariate network visualization technique for your datasets and tasks!

Get in touch if you have questions or comments.

Use the Wizard

Technique recommendations to fit your needs!

Navigate to the wizard tab and select your specific network characteristics, such as the size of the network and its type, and what tasks are relevant for your analysis and receive technique recommendations that are best suited to your selection.

Read the Review Article

The State of the Art in Visualizing Multivariate Networks
Carolina Nobre, Miriah Meyer, Marc Streit, and Alexander Lex
To appear in Computer Graphics Forum (EuroVis 2019)

vdl.sci.utah.edu/mvnv/