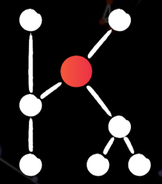


GRAPH VISUALIZATION

KIRELL BENZI, PH.D

@KirellBenzi

Inspired by Lex

www.kirellbenzi.com

BIO

- Ph.D. in Data Science from EPFL
- CEO and founder of Kirelion, creative data science studio
- Data visualization lecturer @ EPFL
- Art exhibitions “Singular Networks” running in France, Switzerland, Brazil, etc.

@kirellb

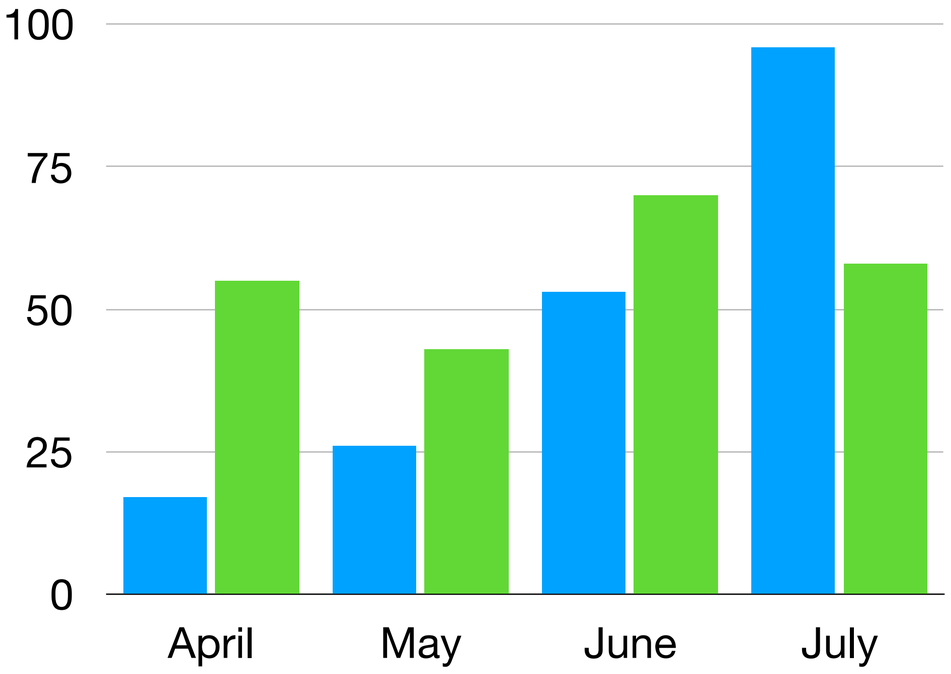
@KirellBenzi

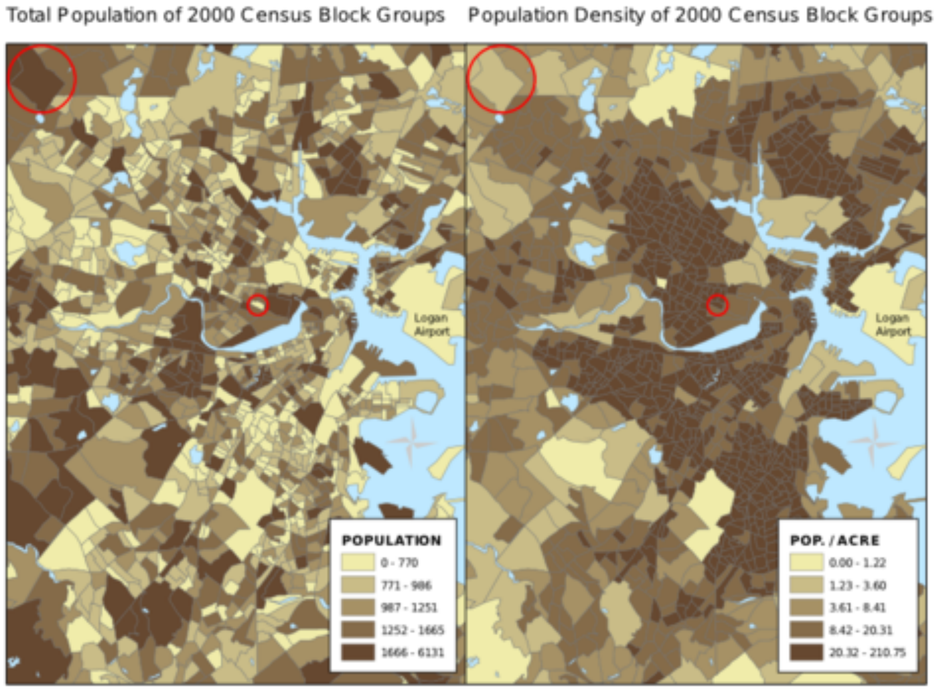
“The purpose of visualization is insight, not pictures.”

–Ben Shneiderman

Defining data visualization

Efficiently communicate information from ~~statistical~~ data using visual objects (such as bars, lines, points, etc.)





Wikipedia

Why visualize?

Communication

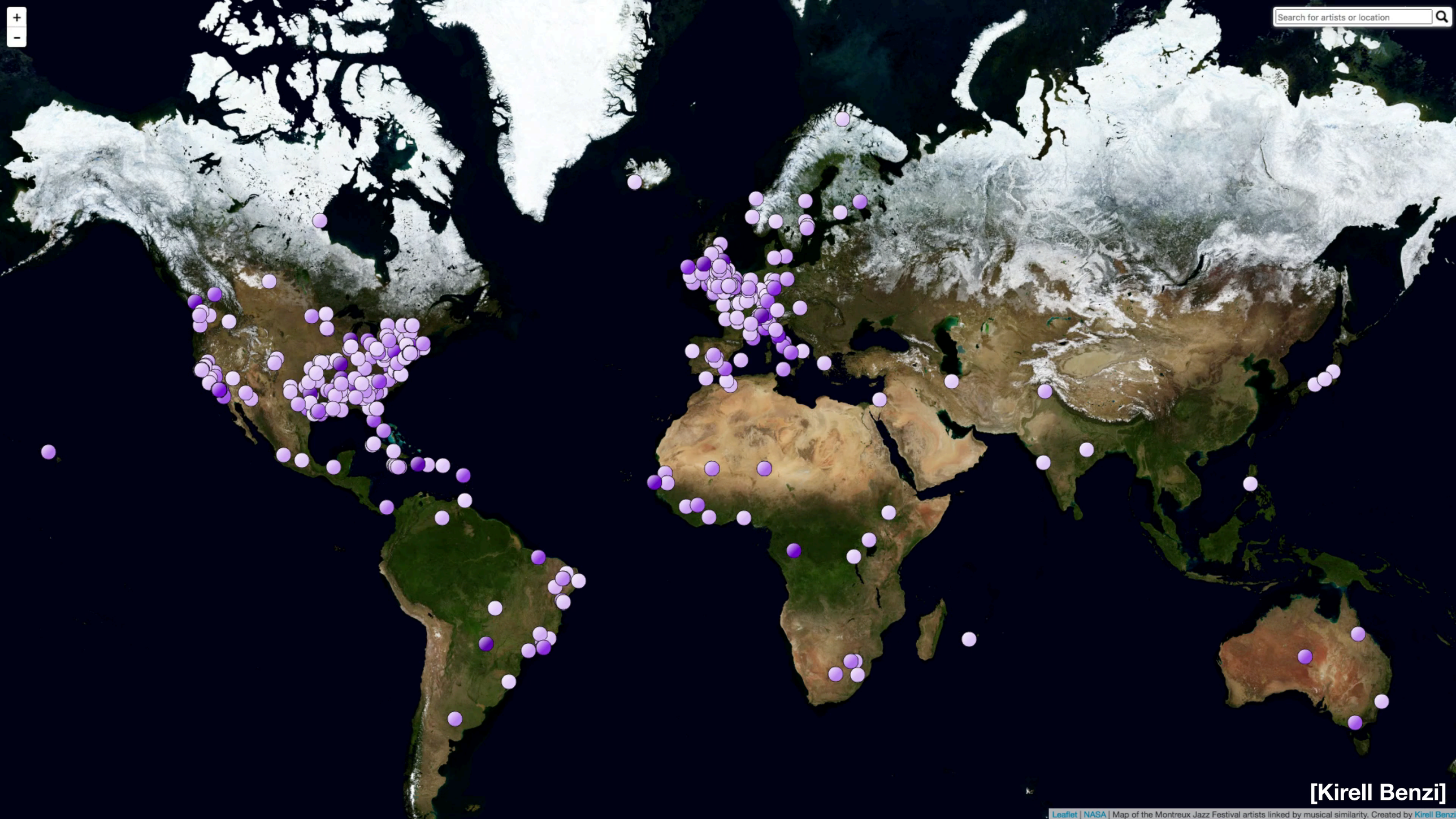
Transmit information to others

Exploration

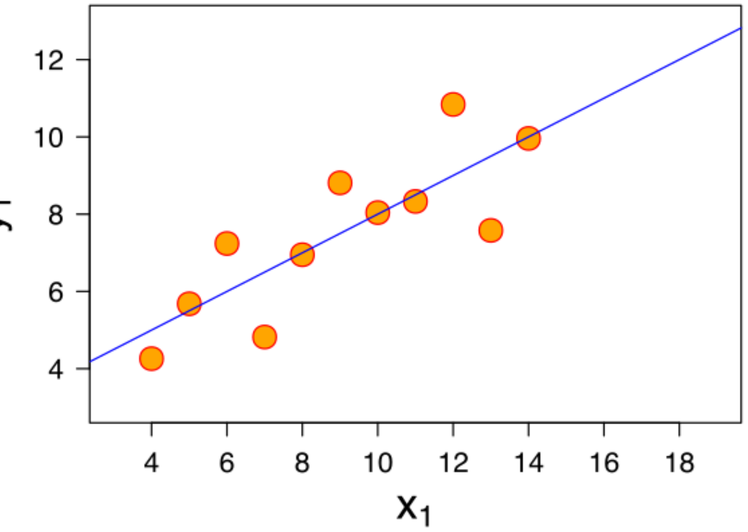
Reveal previously hidden insights from the data

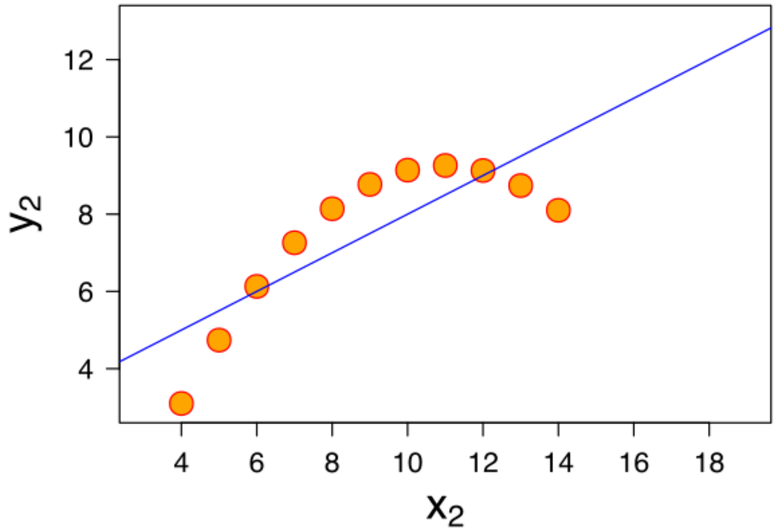
Cognition

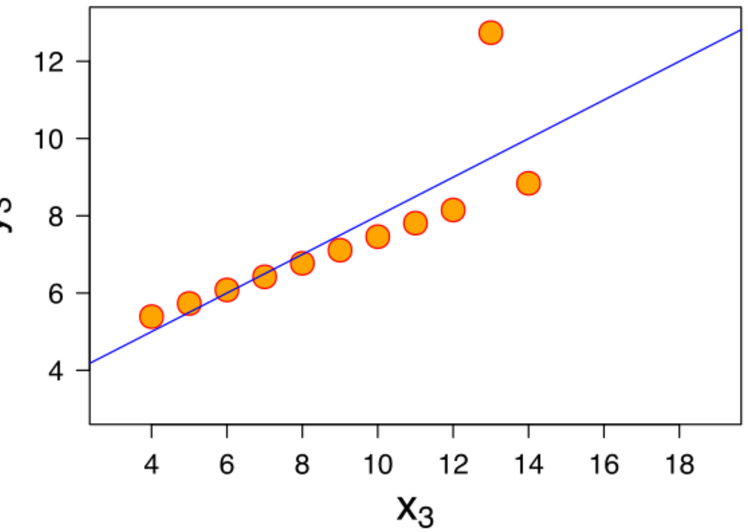
Understand and reason about the data

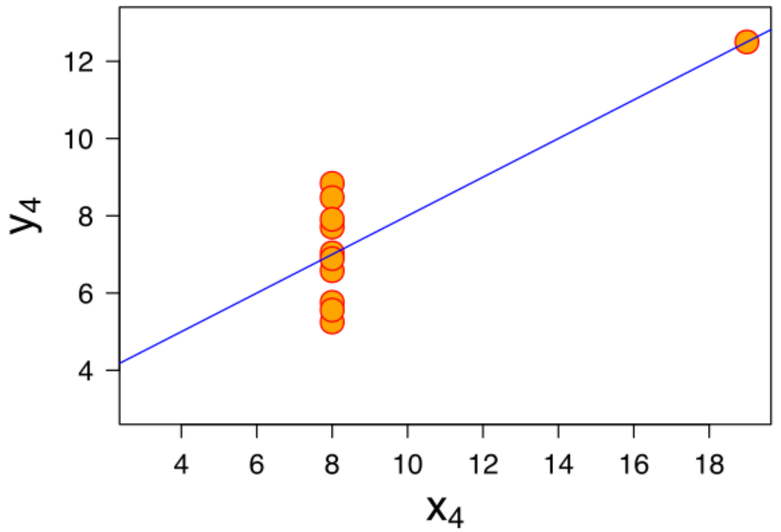

+
-
Search for artists or location
[Kirell Benzi]
Leaflet | NASA | Map of the Montreux Jazz Festival artists linked by musical similarity. Created by Kirell Benzi

Why not only use statistics?









Identical statistical properties

Property	
Mean of *x*	9
Sample variance of *x*	11
Mean of *y*	7.50
Sample variance of *y*	4.125
Correlation between *x* and *y*	0.816
Linear regression line	*y* = 3.0
Coefficient of determination of the linear regression	0.67

[Anscombe's quartet]

Using a graphical representation

Figures are richer; provide more information with less clutter and in less space.

Figures provide the ***gestalt*** effect: they give an overview; make structure more visible.

Figures are more accessible, easier to understand, faster to grasp, more comprehensible, more memorable, more fun, and less formal.

Alexander Lex

Tor Nørretranders

Human visual system

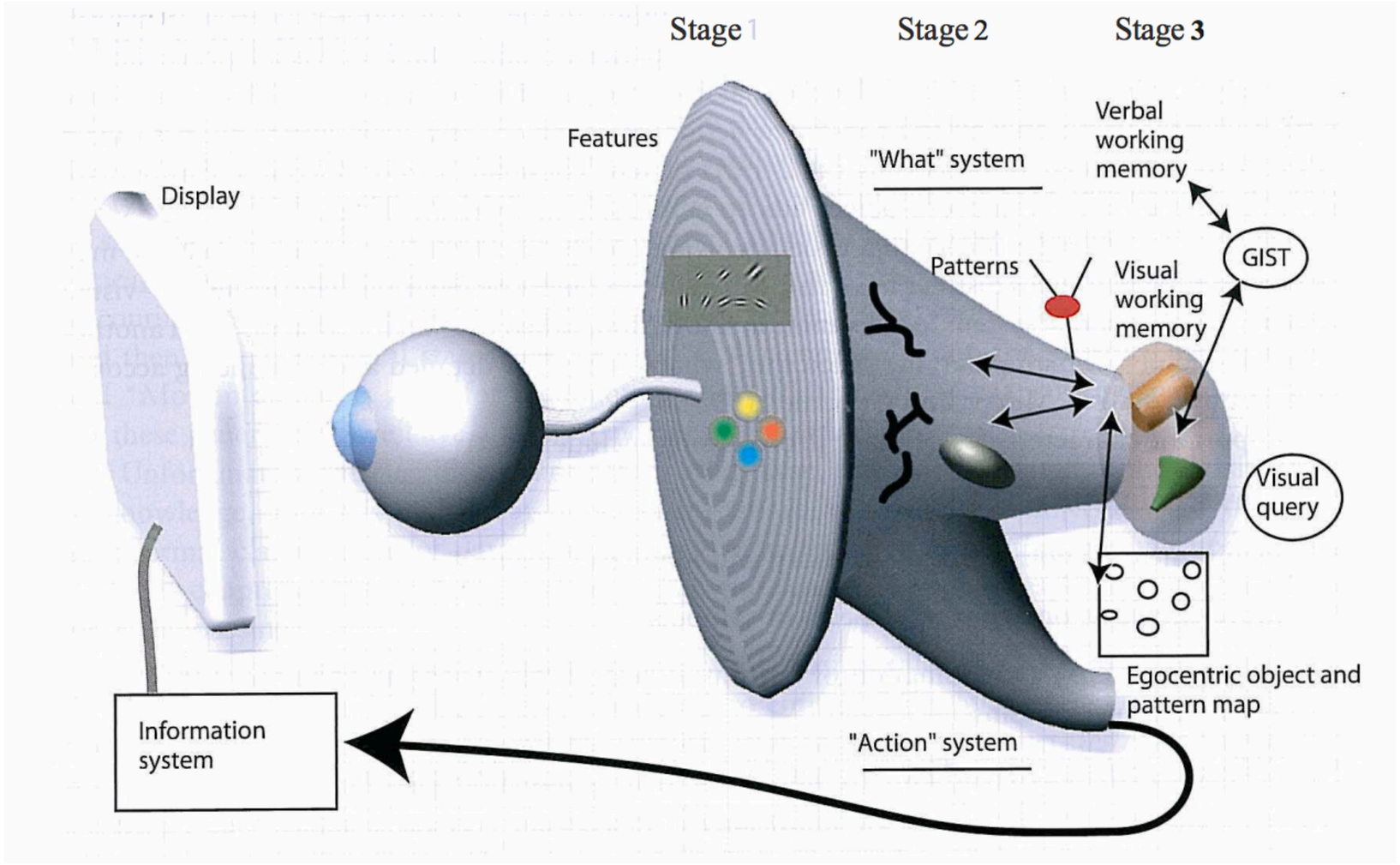
Vision works as a sequence of rapid eye movements: **fixations** and **saccades**

Fixation: maintening gaze on a location during 200-600 ms

Saccade: quickly switching between different locations 20-100ms

Gaze plot example

Visual information processing model



A three-stage model of human visual information processing.

[Ware]

Visual popout

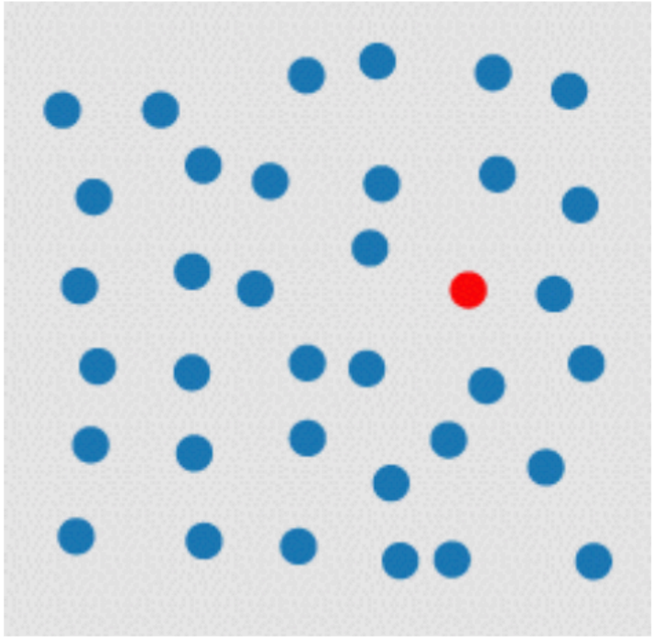
Properties detected by the low-level visual system:

very fast and very accurate (200-250 milliseconds)

processed in parallel

Happens before attention

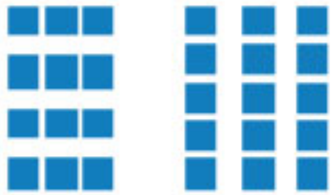
Also called: preattentive processing features



Difference in hue

Gestalt effect

PROXIMITY



Most people see rows on the left and columns on the right.

SIMILARITY

Most people see a larger group containing a sub-group.

FIGURE GROUND

Most people first see a vase or a face then flip between the two.

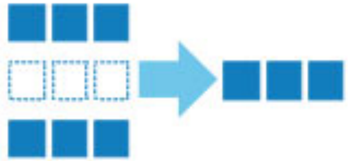
CONTINUATION

Most people see 2 rows crossing rather than for lines meeting at a single point.

CLOSURE

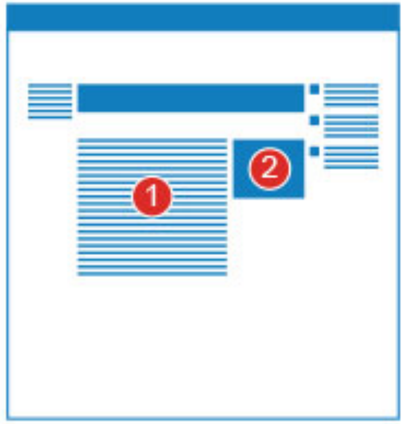
Most people see the white triangle not the 3 circles with segments missing.

COMMON FATE

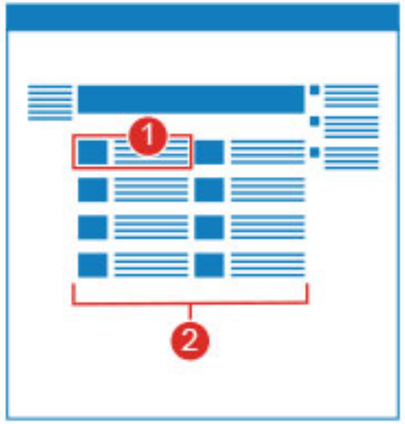


Objects moving in a similar direction are percieved as belonging together.

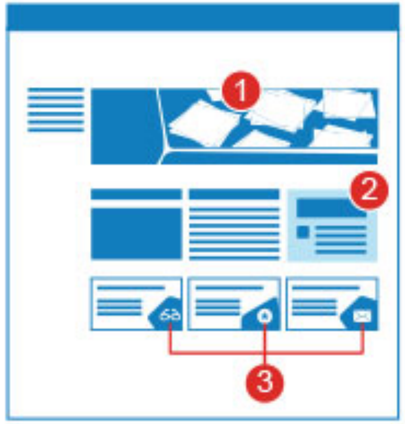
Examples:



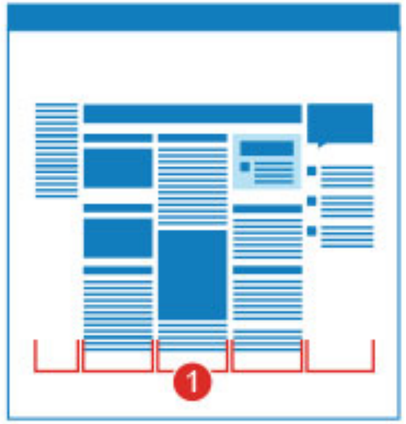
Elements that are closer together are perceived as being more related compared to those spaced further apart. In this



Web parts that look similar are perceived as being grouped together or related.



We automatically perceive objects as being in the foreground, (figure) or in the background (ground). Anything

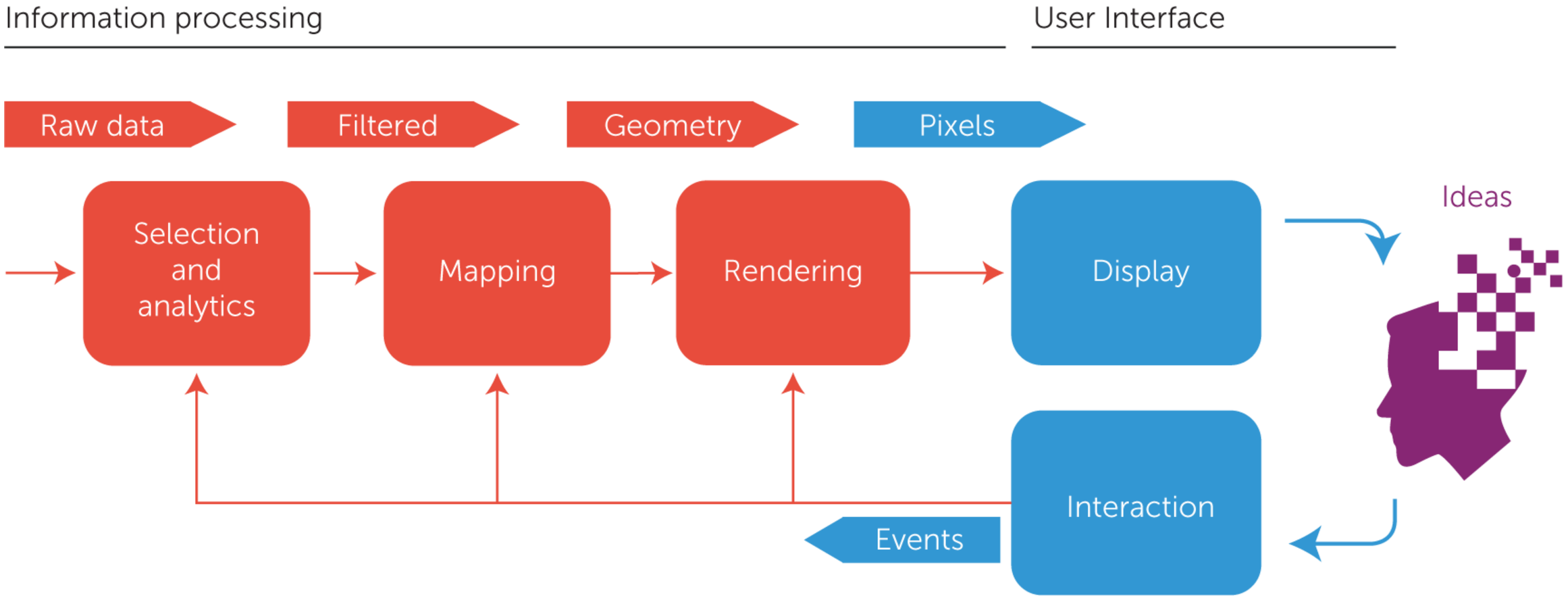


We tend to perceive contours as objects. In this way we perceive lines continue in an established direction (even when they don't) .

When we perceive a pattern the gaps (1) between the objects (negative or white space) are just as important as the objects

When objects move in the same direction we perceive them to to be related and moving on an invisible path. Even if objects

Visualization Pipeline

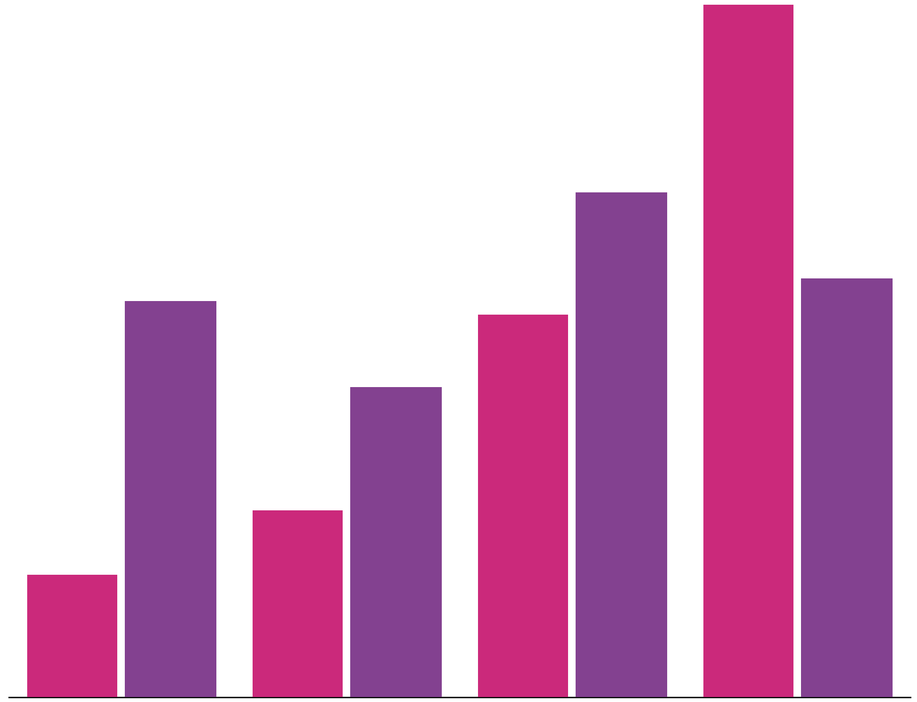


[Holliman 2015]

Definitions

Marks: basic geometric elements to represent items or links

Channels: visual variable, change the appearance of marks based on attributes



Marks for items/nodes

Points

0D

sense of place

Lines

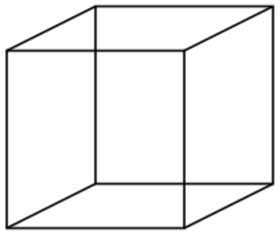
1D

sense of length and direction

Areas

2D

sense of space and scale



3D

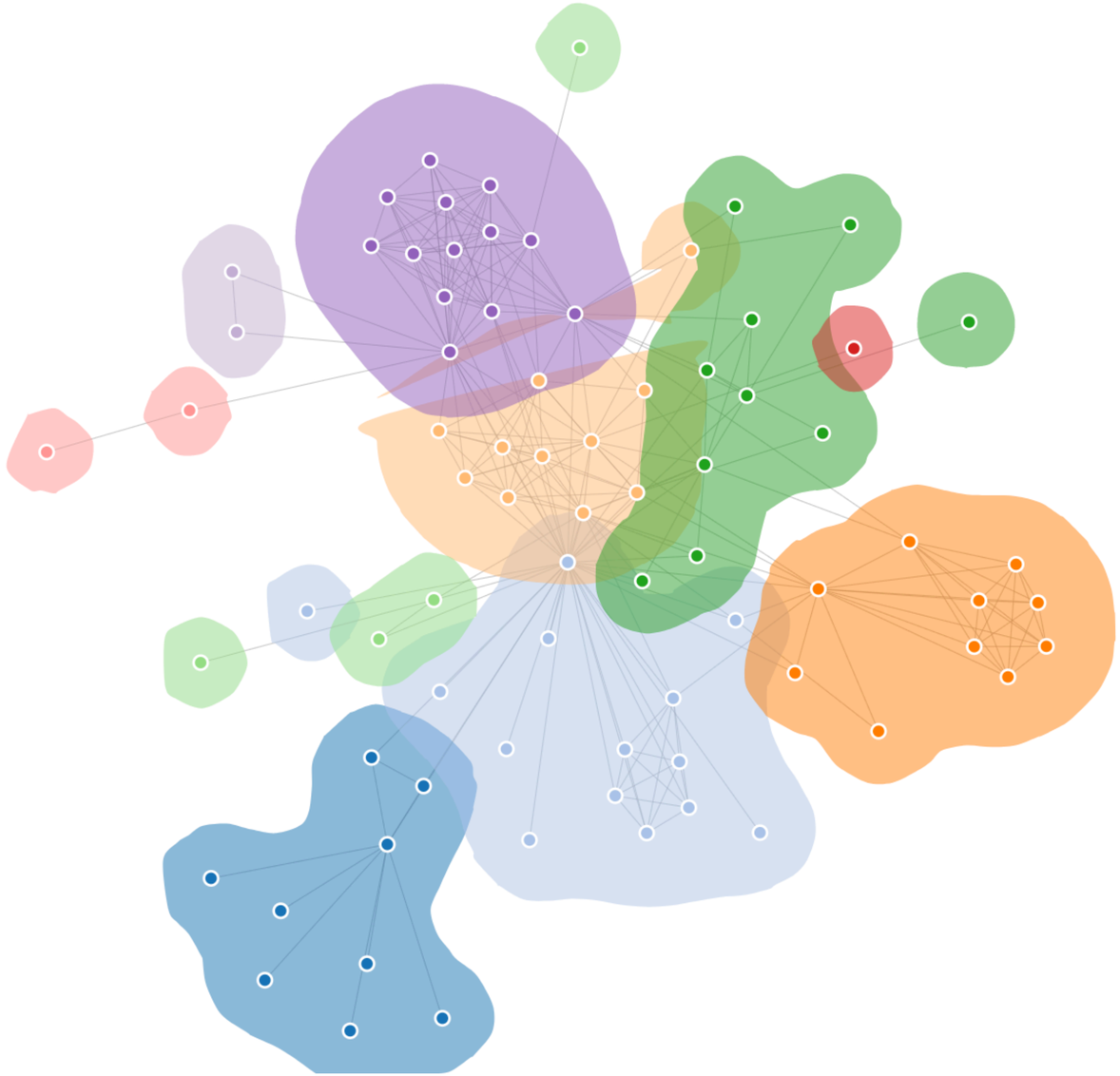
sense of volume

Marks for links

(Enclosure)

Containment

Connection



Channels (visual attributes)

Position

- Horizontal
- Vertical
- Both

Color

Shape

Tilt

Size

- Length
- Area
- Volume

Control appearence of marks

[T. Munzner, Visualization Analysis and Design, 2014]

Channel effectiveness

Accuracy: How well can a user read the information in the channel?

Discriminability: How easily can we perceive differences between attribute levels?

Separability: Can we use one channel independently of another? Do they interfere?

Popout: How can a channel trigger a visual popout when processing data?

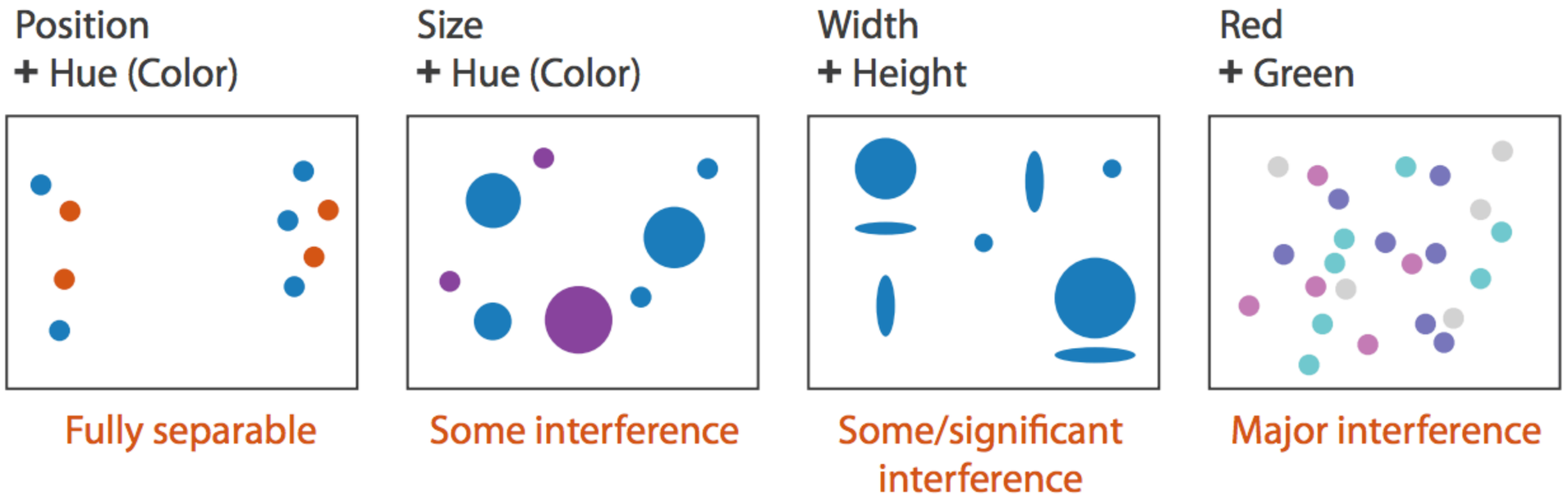
Grouping: How can a channel trigger some of the Gestalt principles?

Test of separability

Low-level preattentive processing features

+

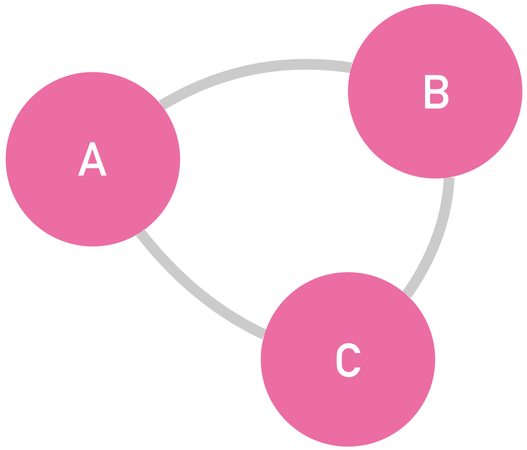
Gestalt principles



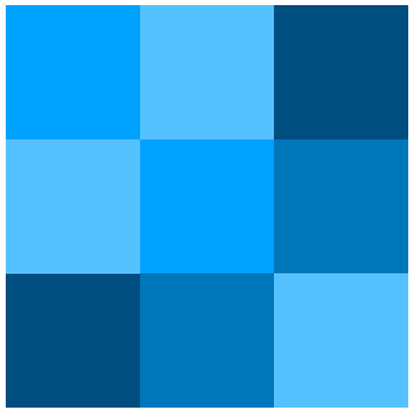
[T. Munzner, Visualization Analysis and Design, 2014]

Graph visualizations

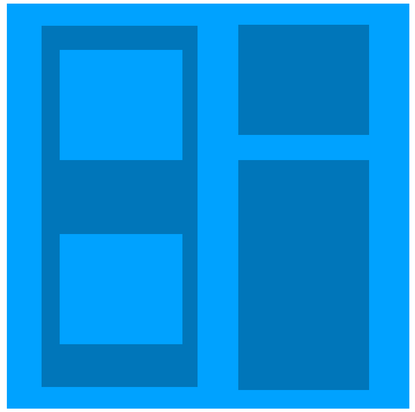
Graph visual encodings



Node-link diagram



Matrix form



Implicit form

Choosing the right representation?

Depends on the task, **attribute-based (ABT)** and/or **topology-based (TBT)**

Localize – find a single or multiple nodes/edges

- ABT: Find edges of a certain type
- TBT: Get the neighborhood of a given node

Quantify – count or estimate a numerical property of the graph

- ABT: Get number of edges
- TBT: Get the number of outboud links from a node

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

- ABT: Sort all edges by distance (or weight)
- TBT: Get the shortest paths from a given node

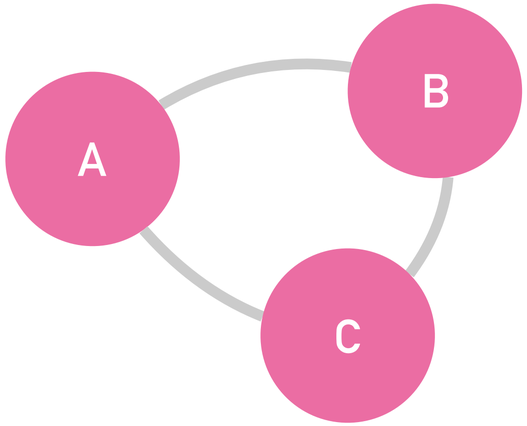
[Lex]

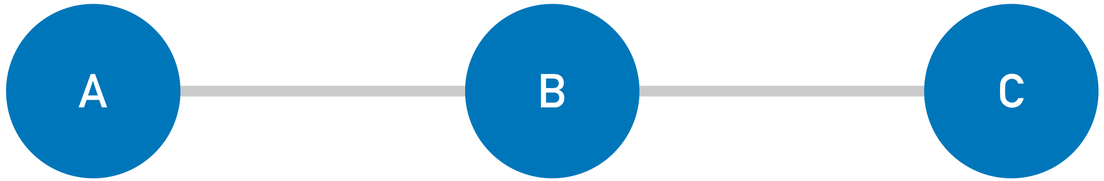
Node-link diagram

Explicit graph representation

Vertex = point

Edge = line or arc



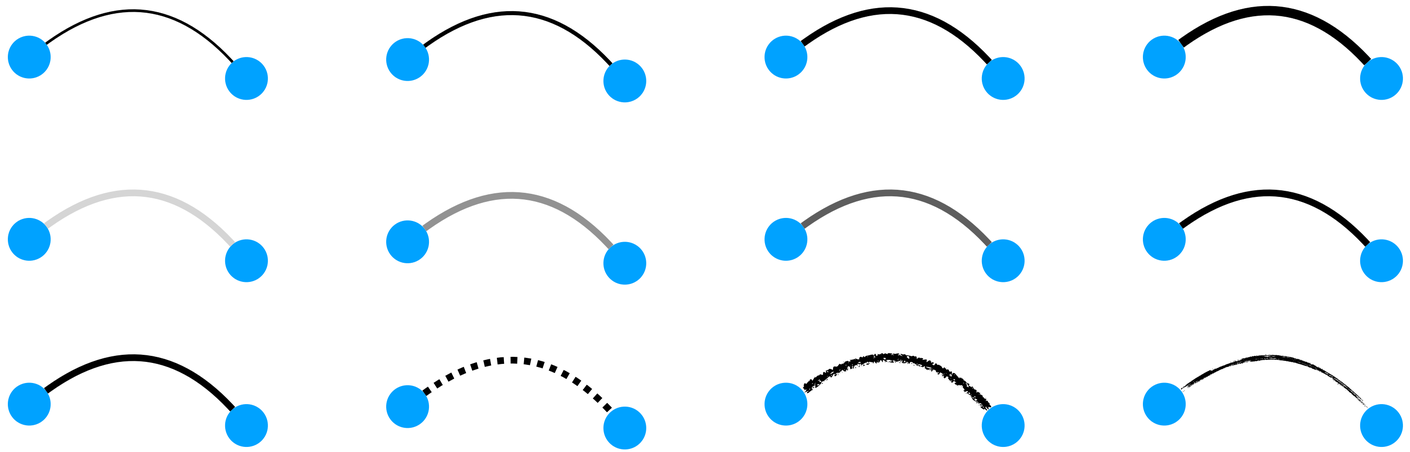


Edge attribute encoding

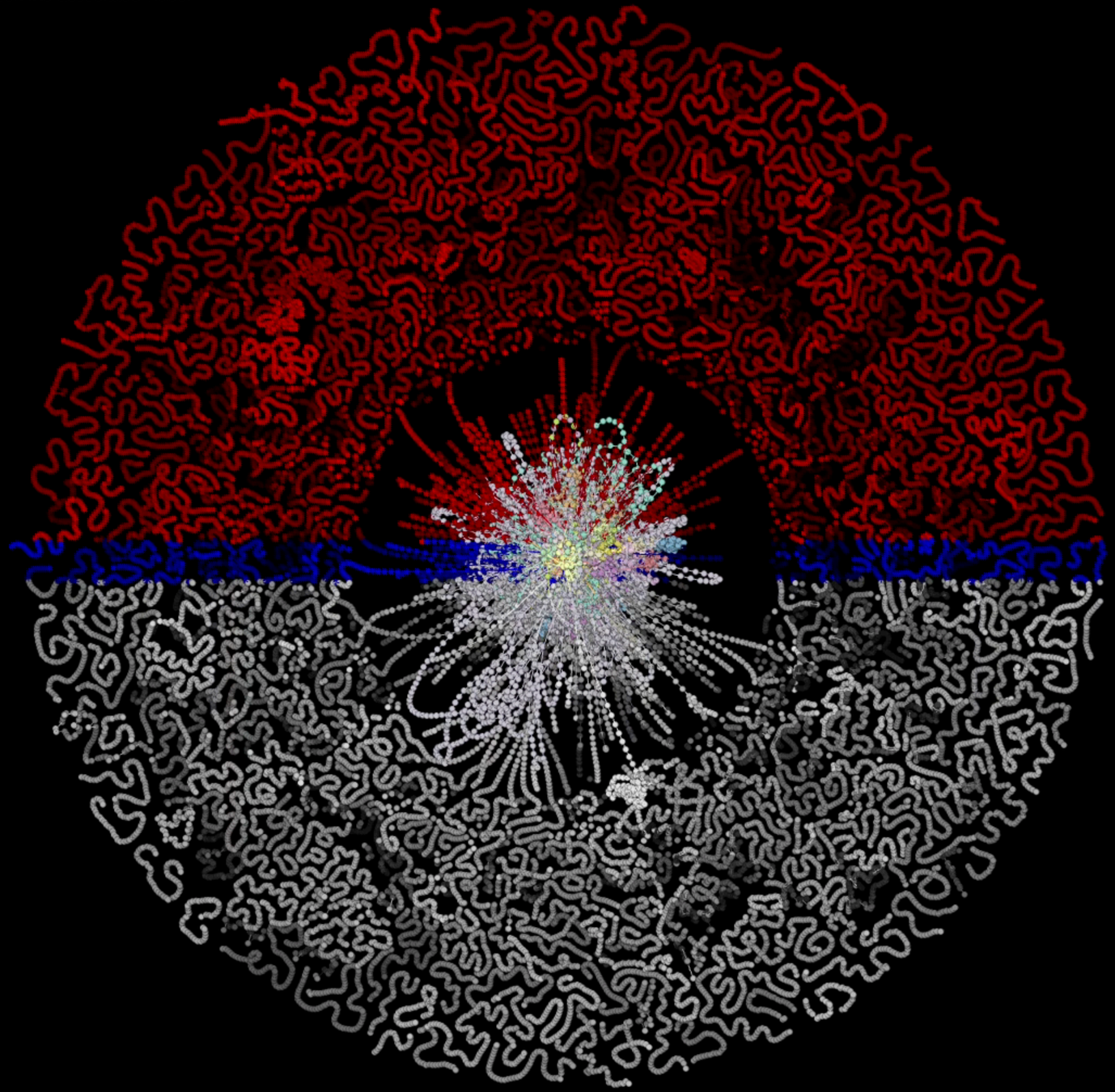
Width: quantitative

Saturation: ordinal

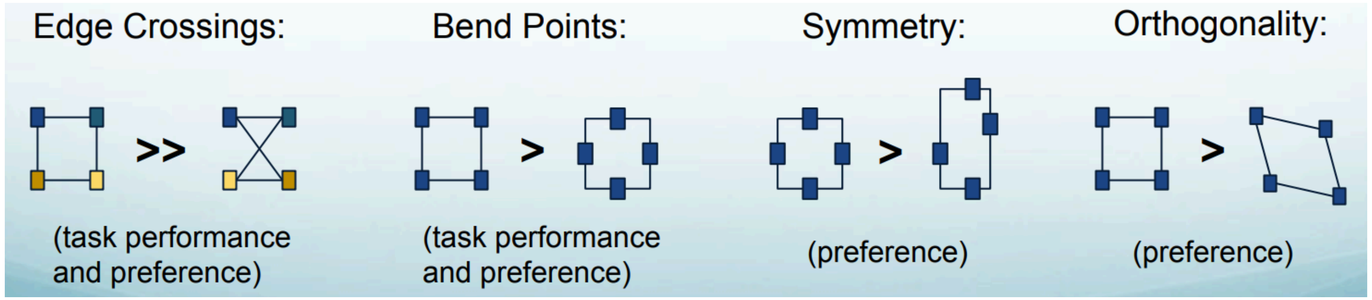
Style: nominal



Navigate using your mouse and double click on a node to open the associated video.



Criteria for good layout



[Hindalong]

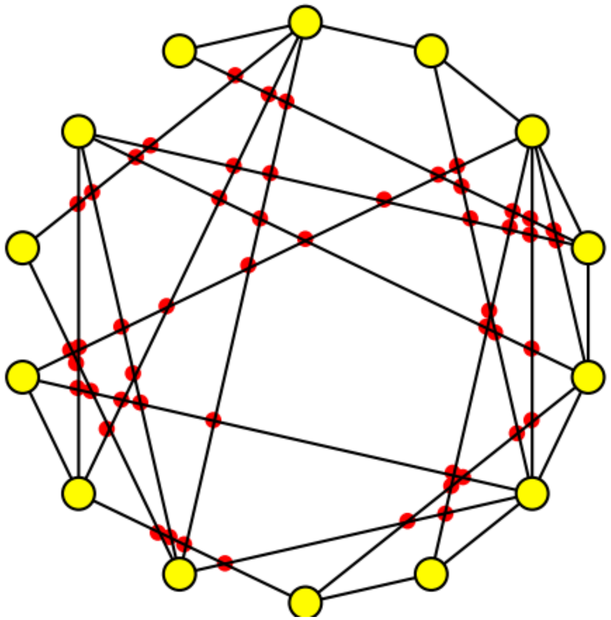
Minimized distance of neighboring nodes

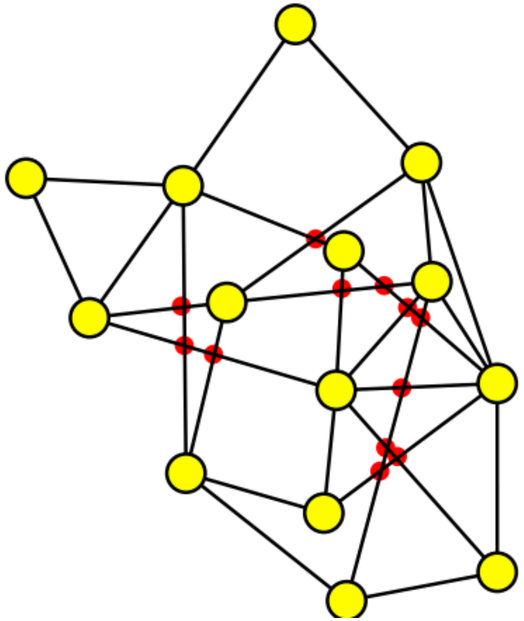
Uniform edge length

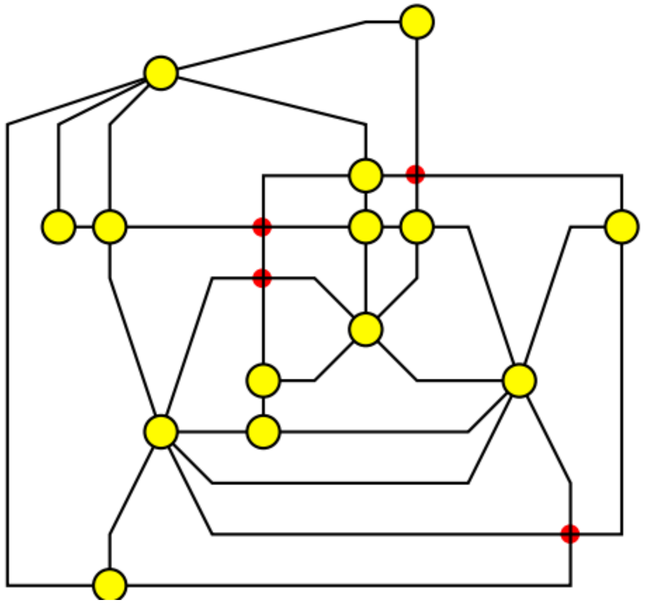
Aspect ratio about 1 (not too long and not too wide)

Minimized drawing area

Conflicting criteria



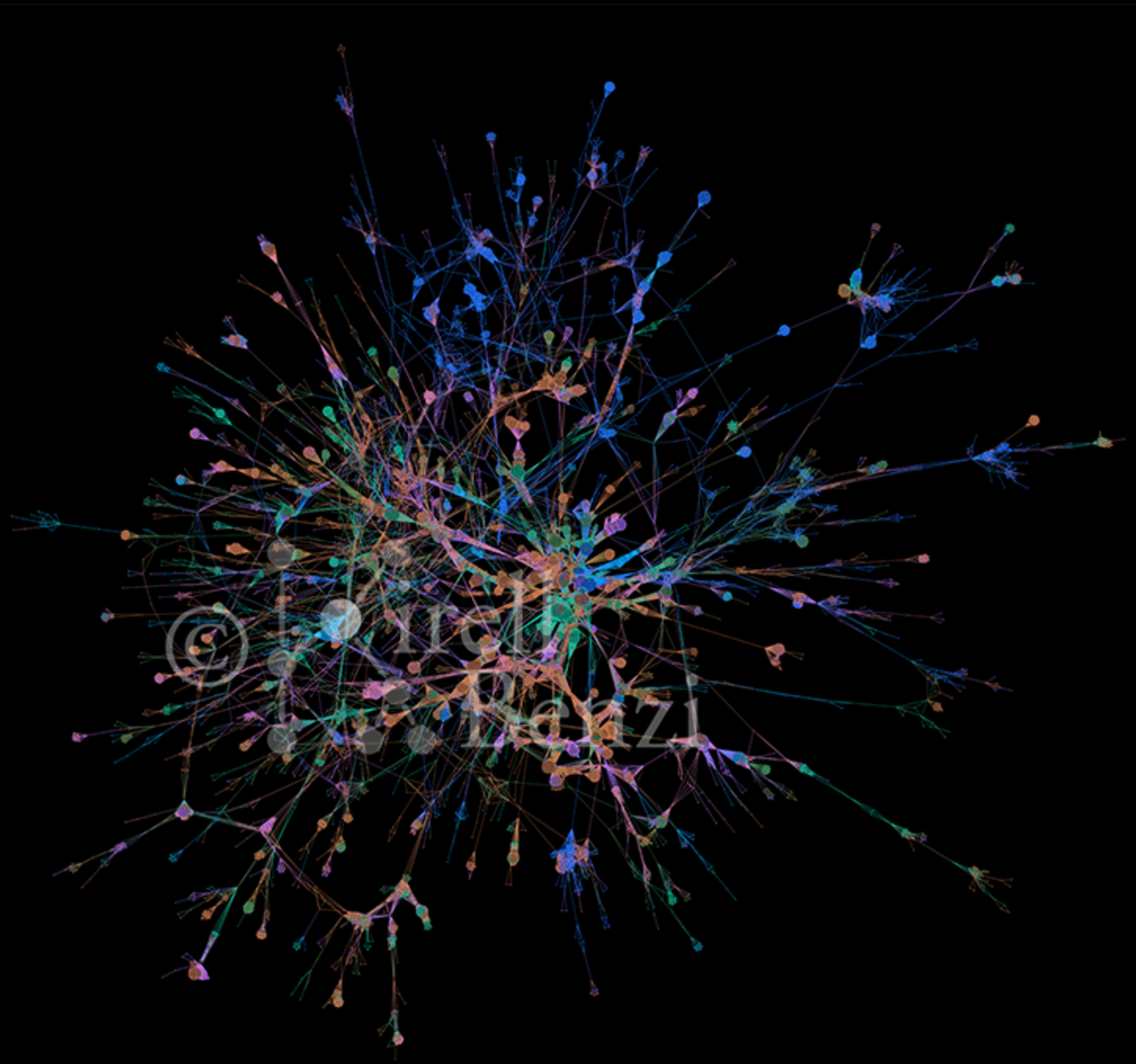




[Buchheim et al.]

Design critique

Kirell Benzi



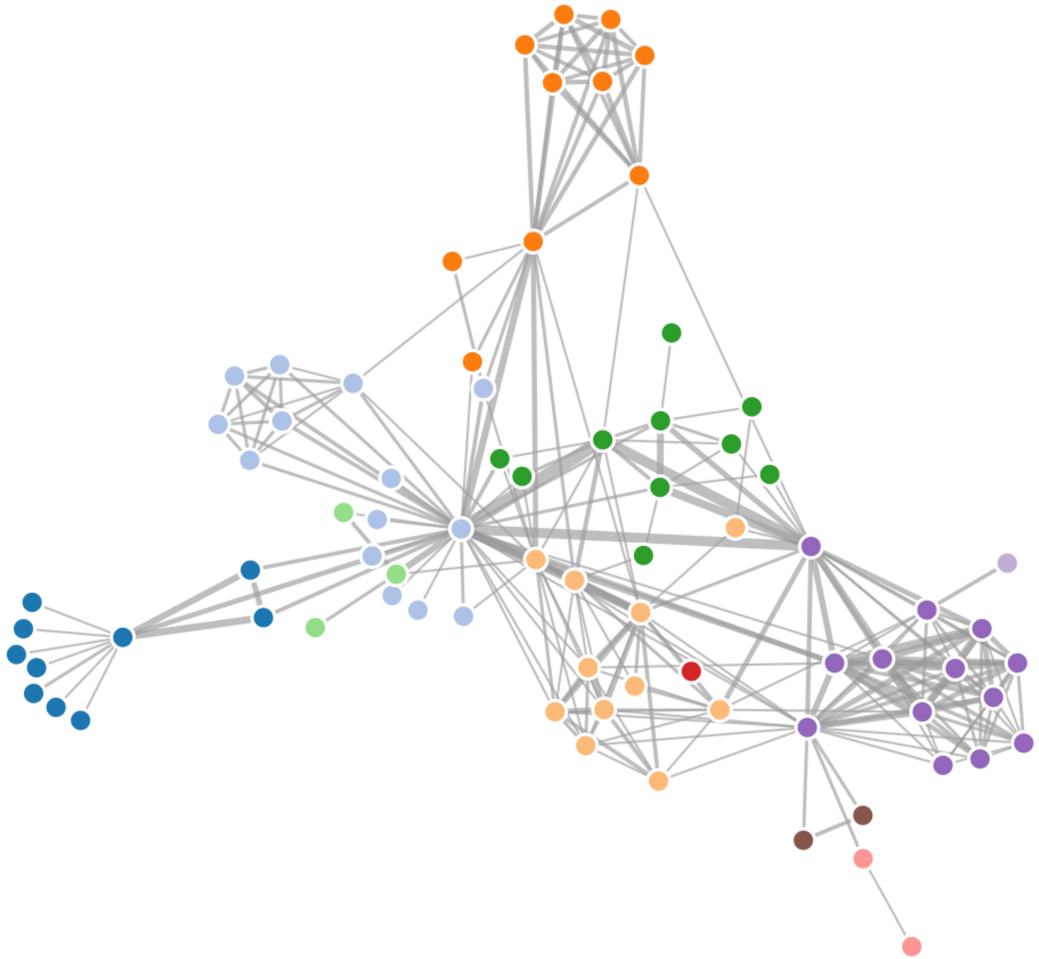
Force-directed layouts

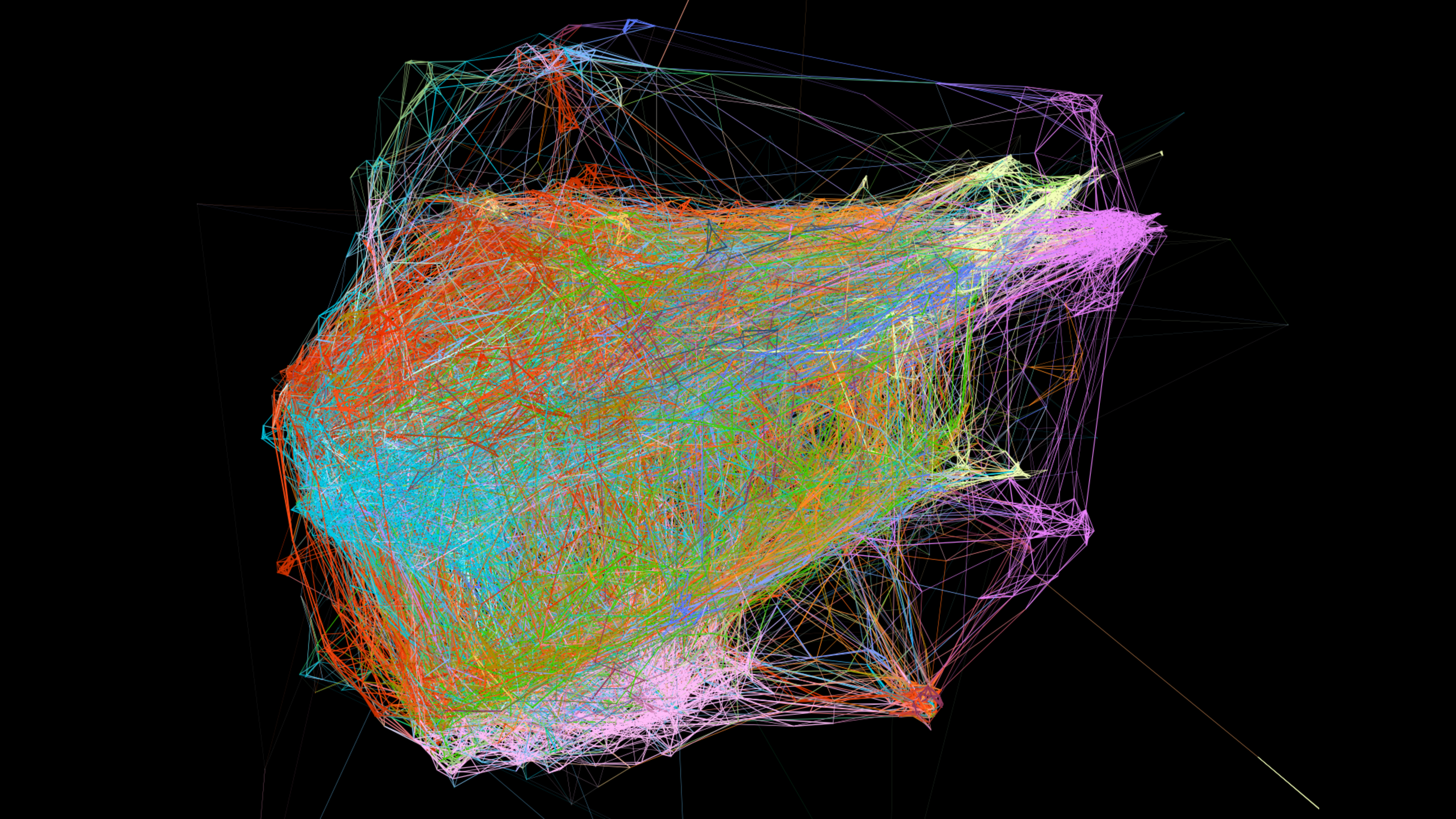
Most common layouts

Adapted from Physics, with vertices as repulsive magnets and edges as springs

Computationally expensive: O(n^3)

Hard to scale for interactivity





Force-directed issues

Giant hairball depending on the topology of the graph

Computing an optimal layout lies in NP

Try to solve as an optimization problem, not necessarily convex

Complexity is still high O(n^2)

In each optimization step, all vertices are compared to all others

[Lex]

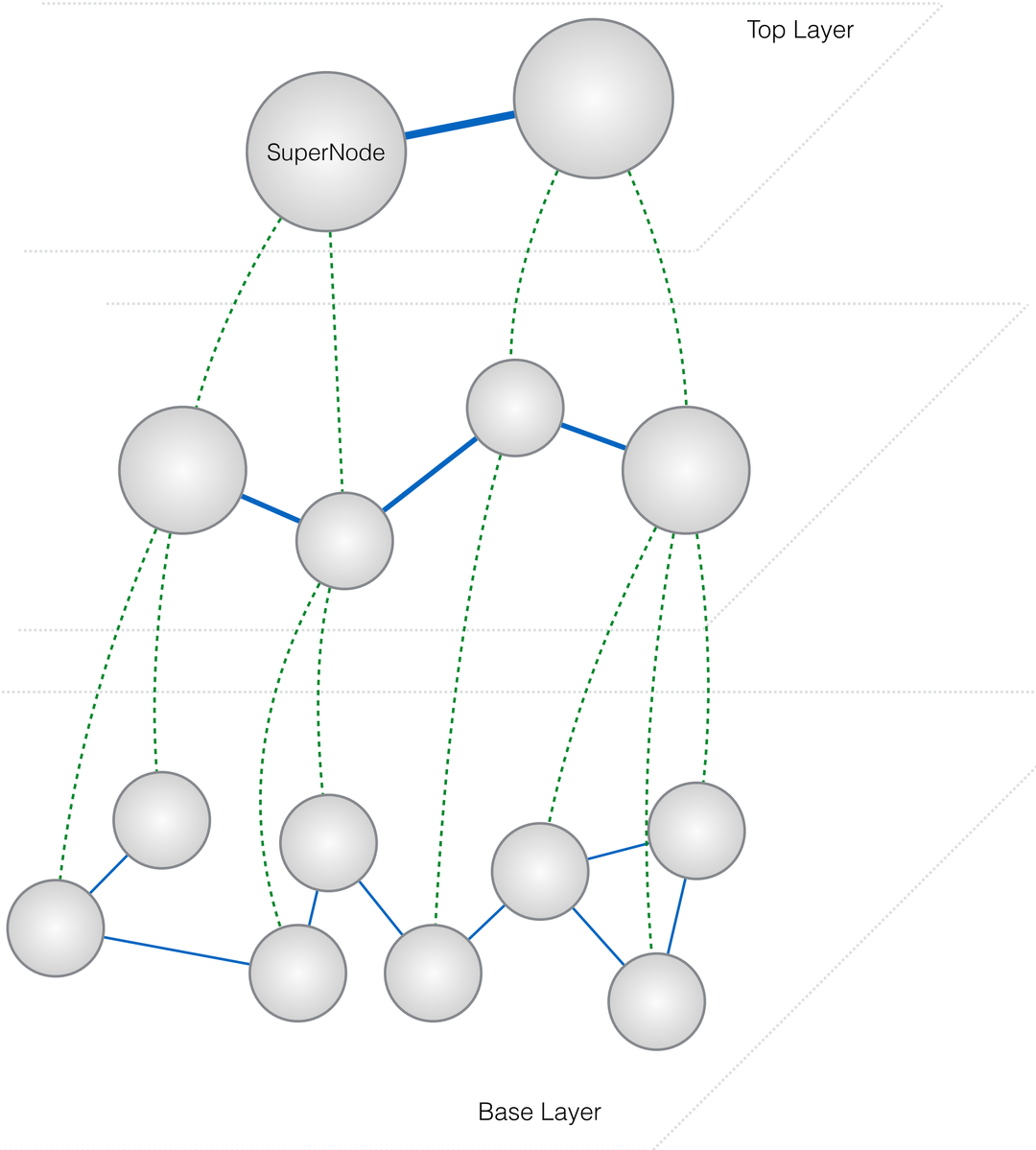
Multilevel approaches

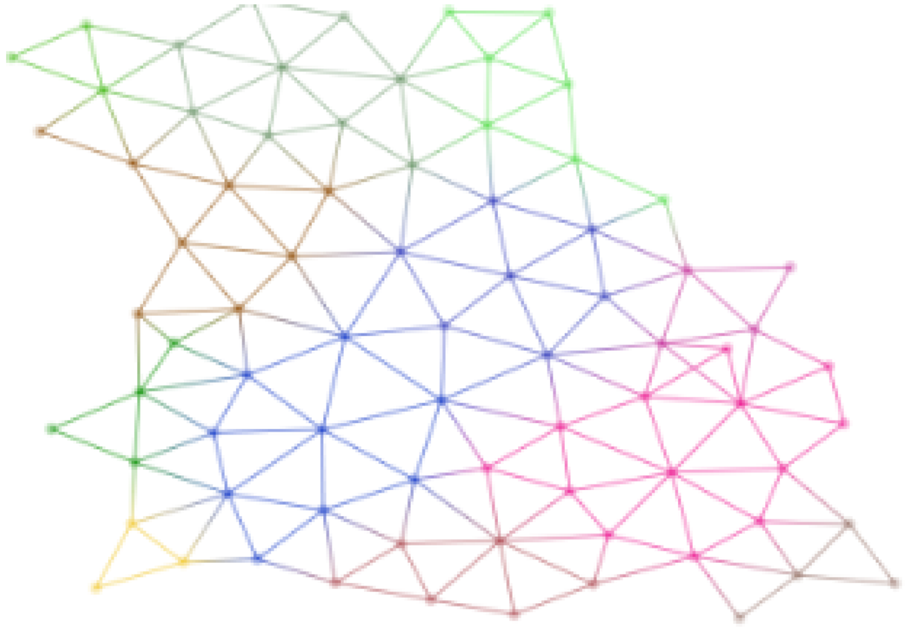
Also known as graph coarsening

Different strategies to collapse node or edges

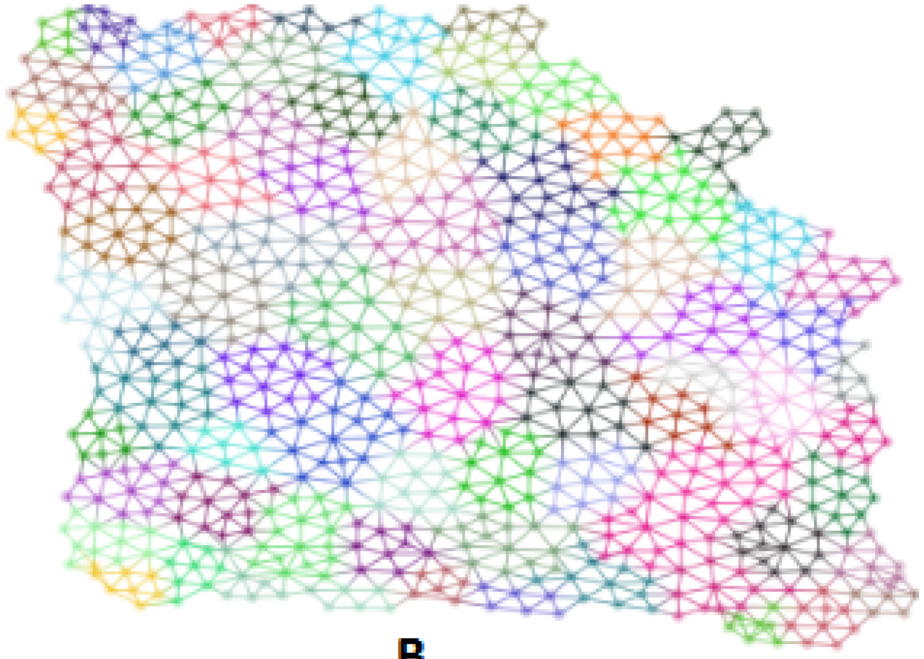
Recursive graph layout from top to bottom

Used also in graph partitioning, shortest path, etc.

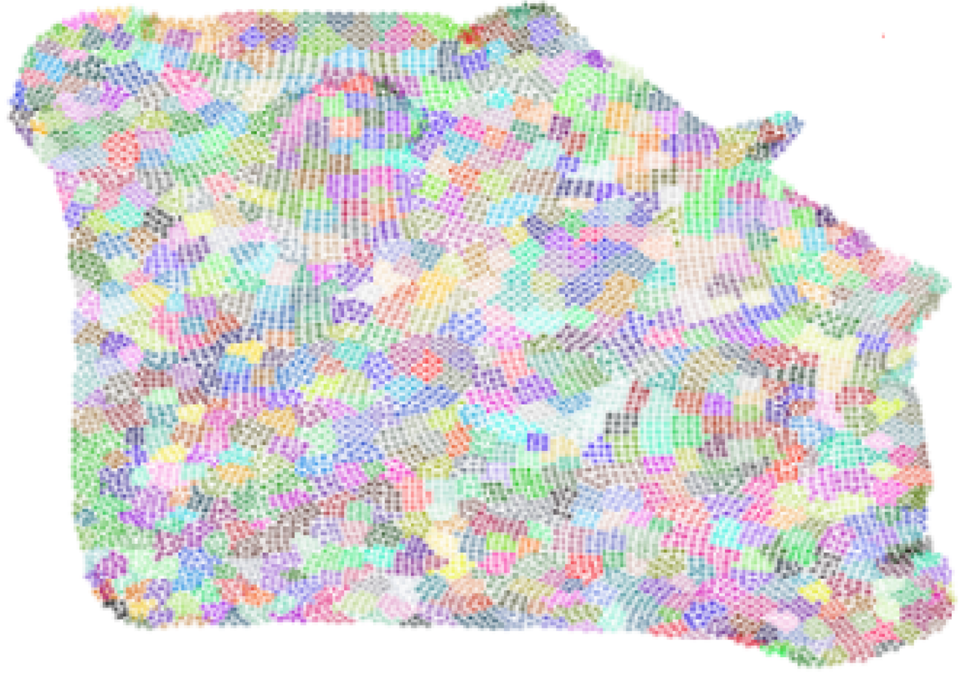




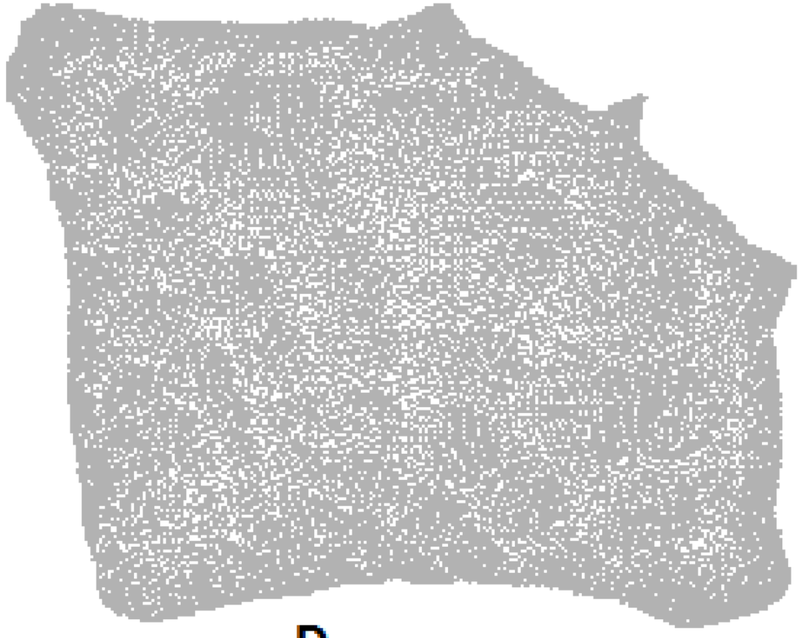
A



B



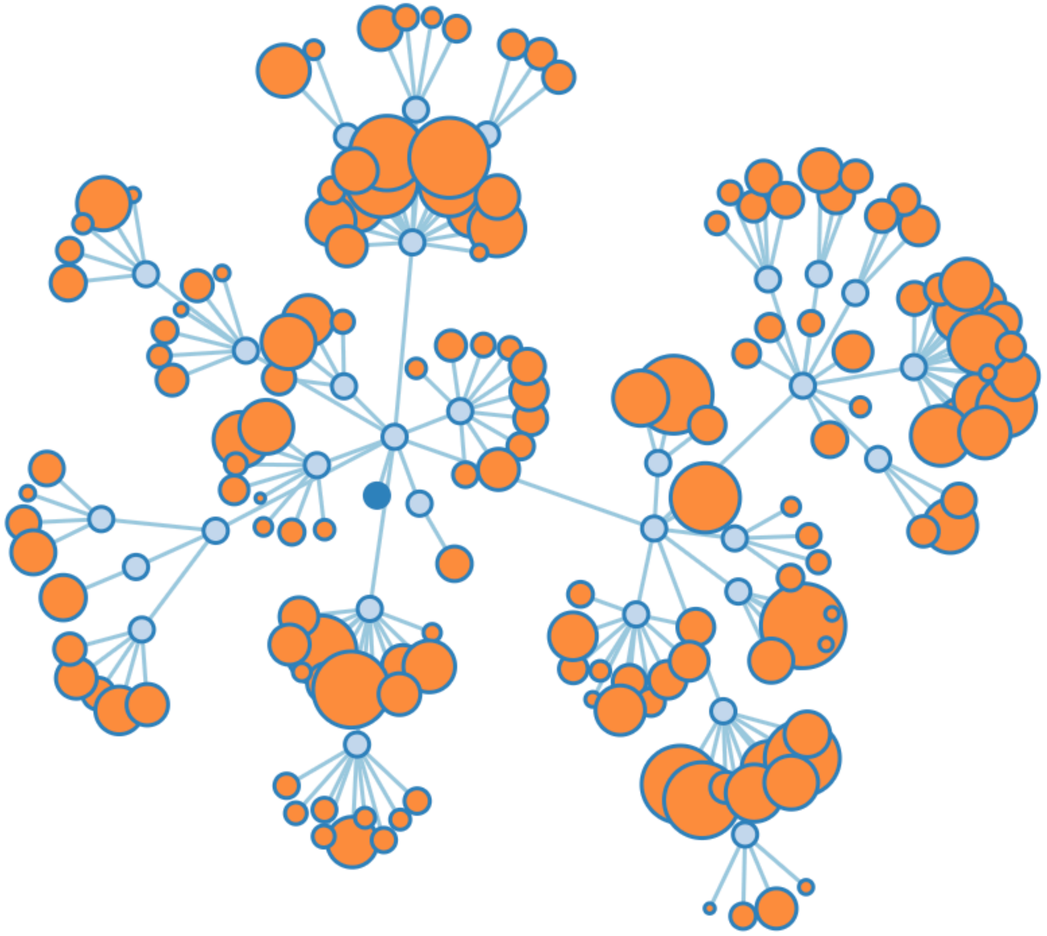
C



D

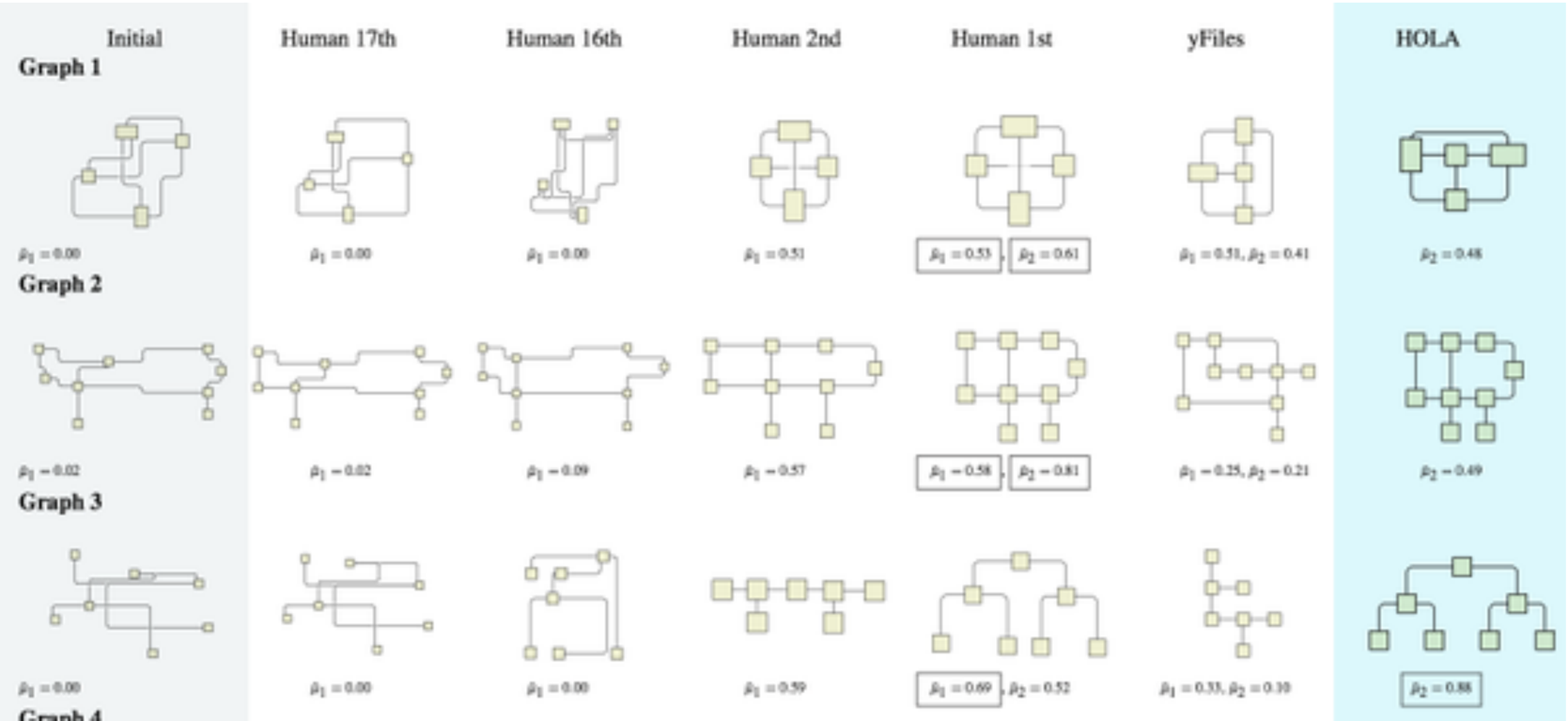
[Mi et al.]

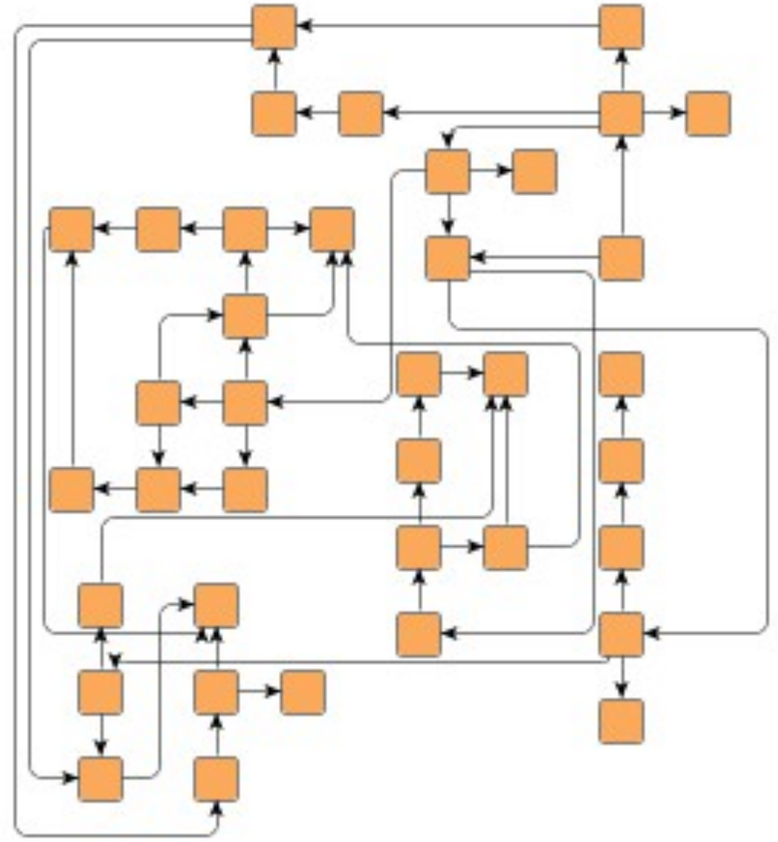
Collapsible Force Layout



Orthogonal layout

Used primarily in electrical engineering and software dev



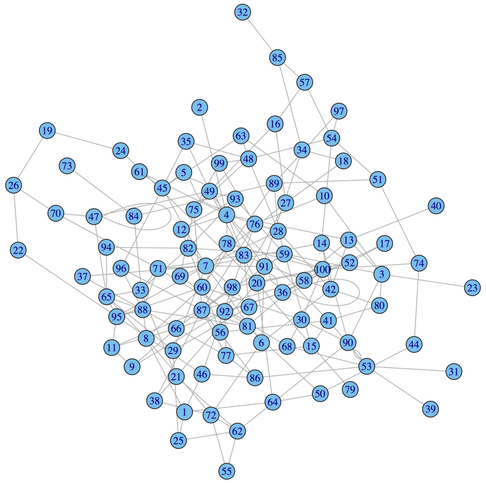


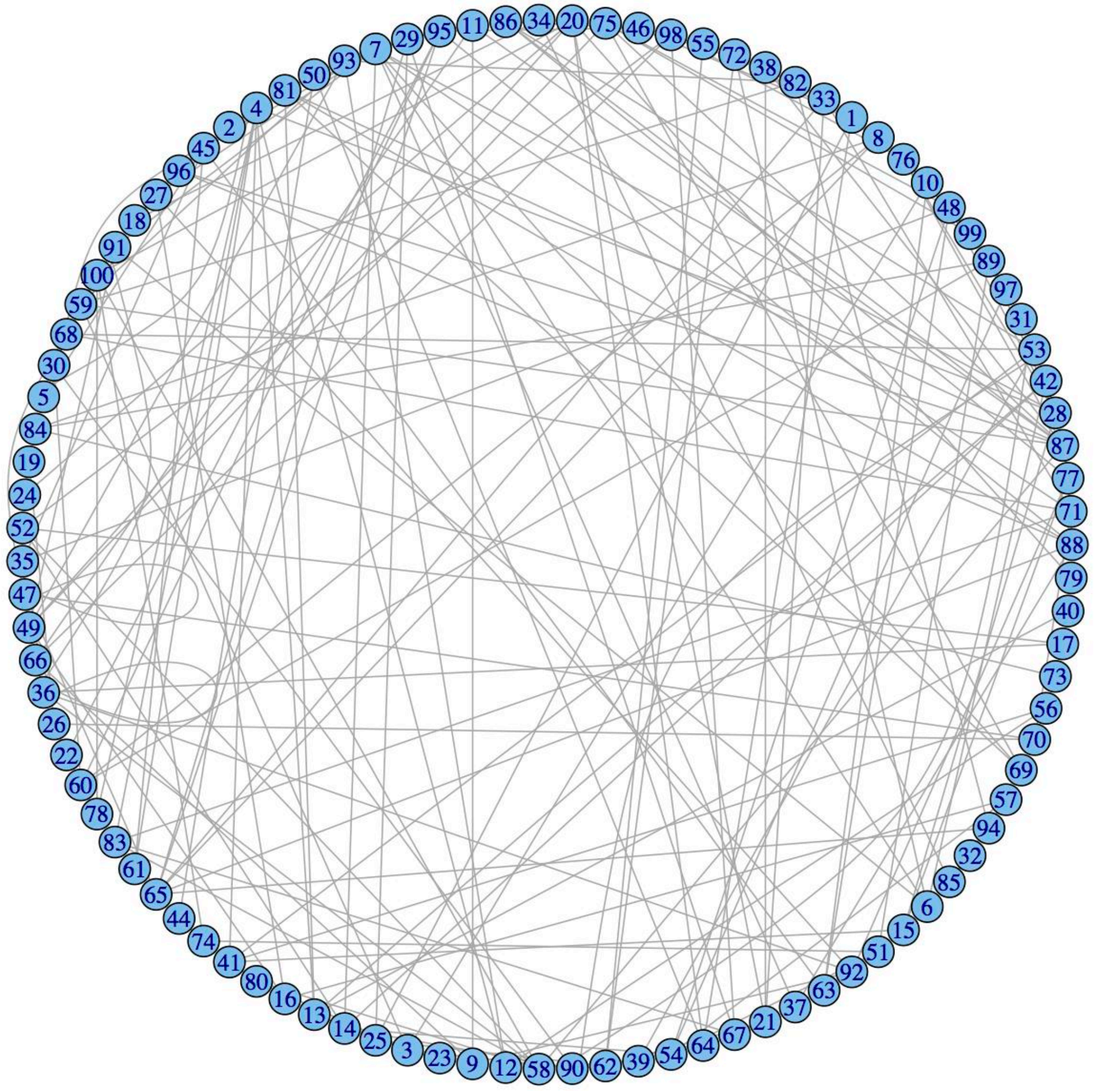
Circle layout

Node positions are constrained on the edge of the circle

Clutter is important

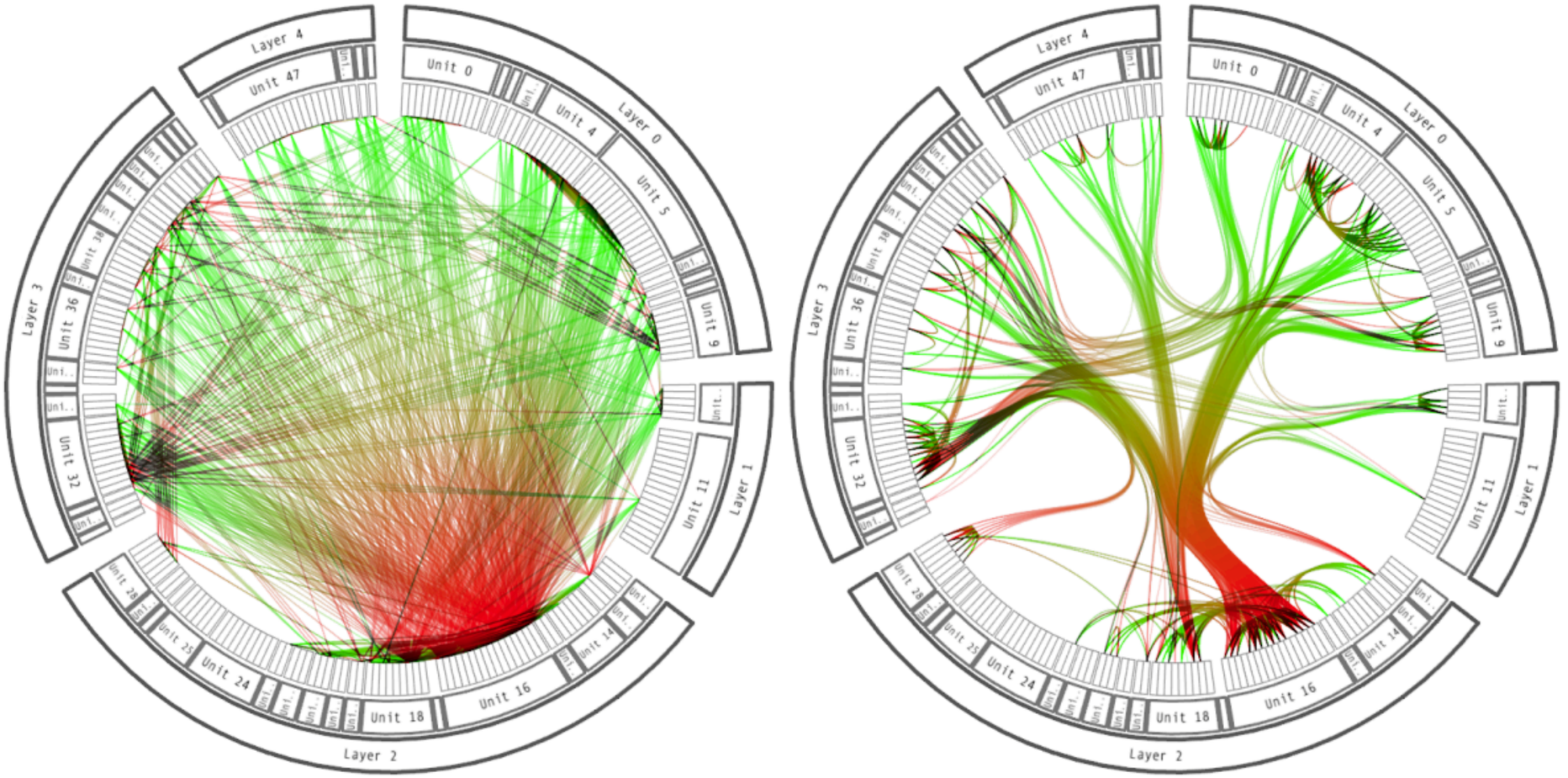
Node ordering is paramount



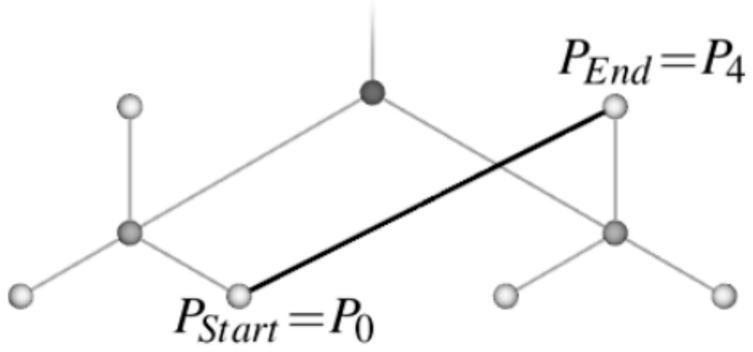


Edge bundling

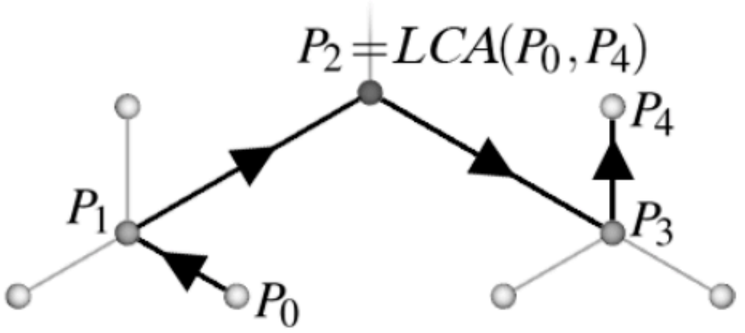
Reduce clutter by visually grouping edges



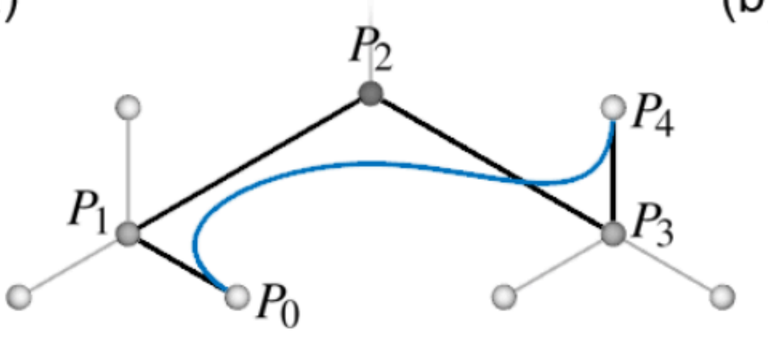
[Holten]



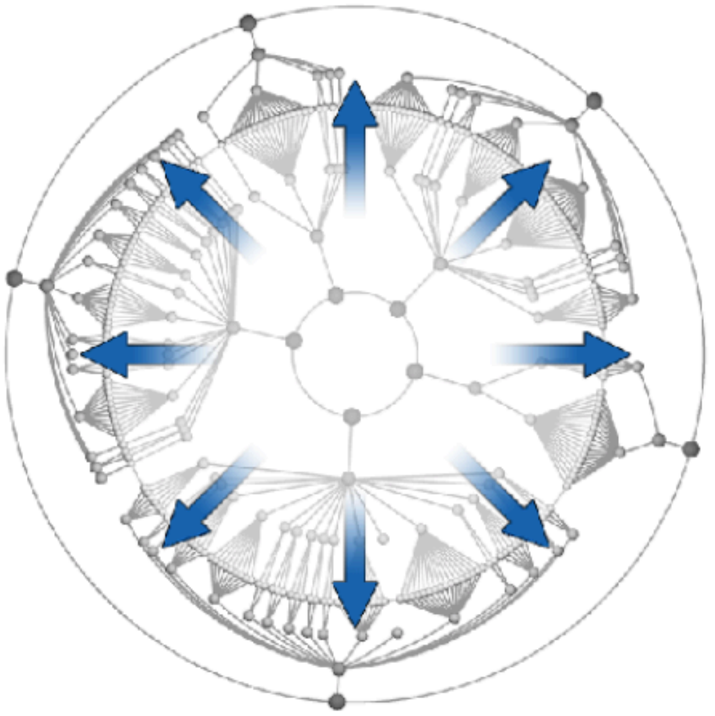
(a)

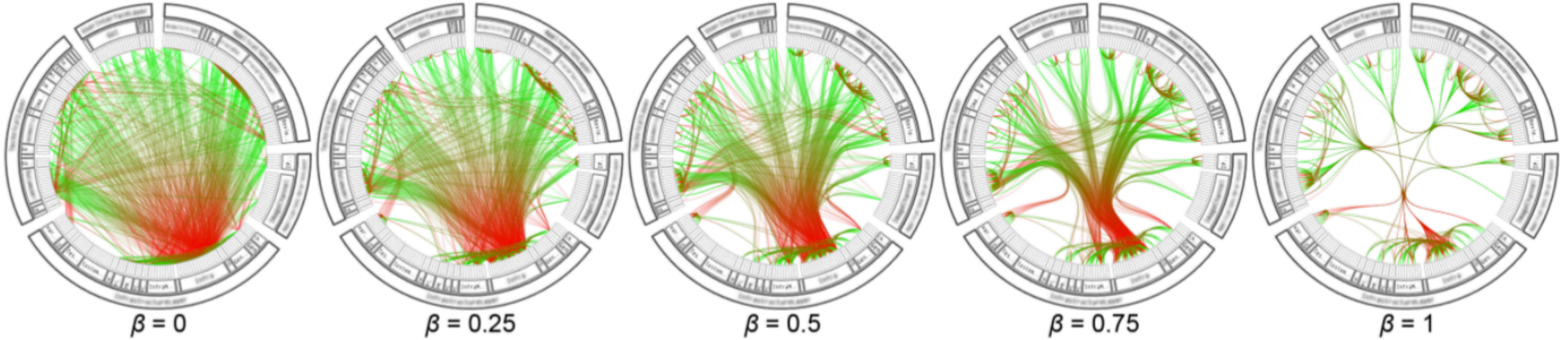


(b)

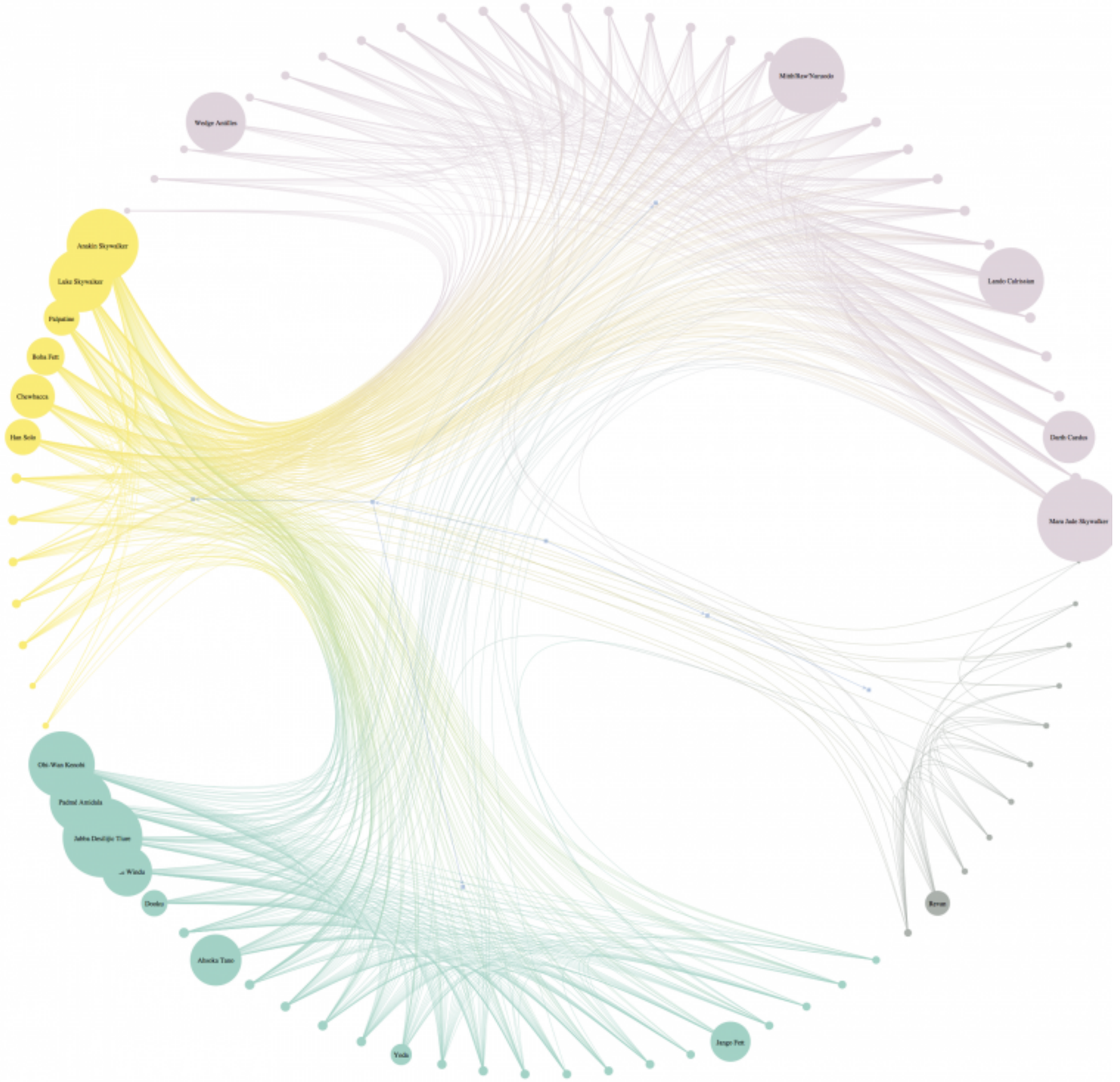


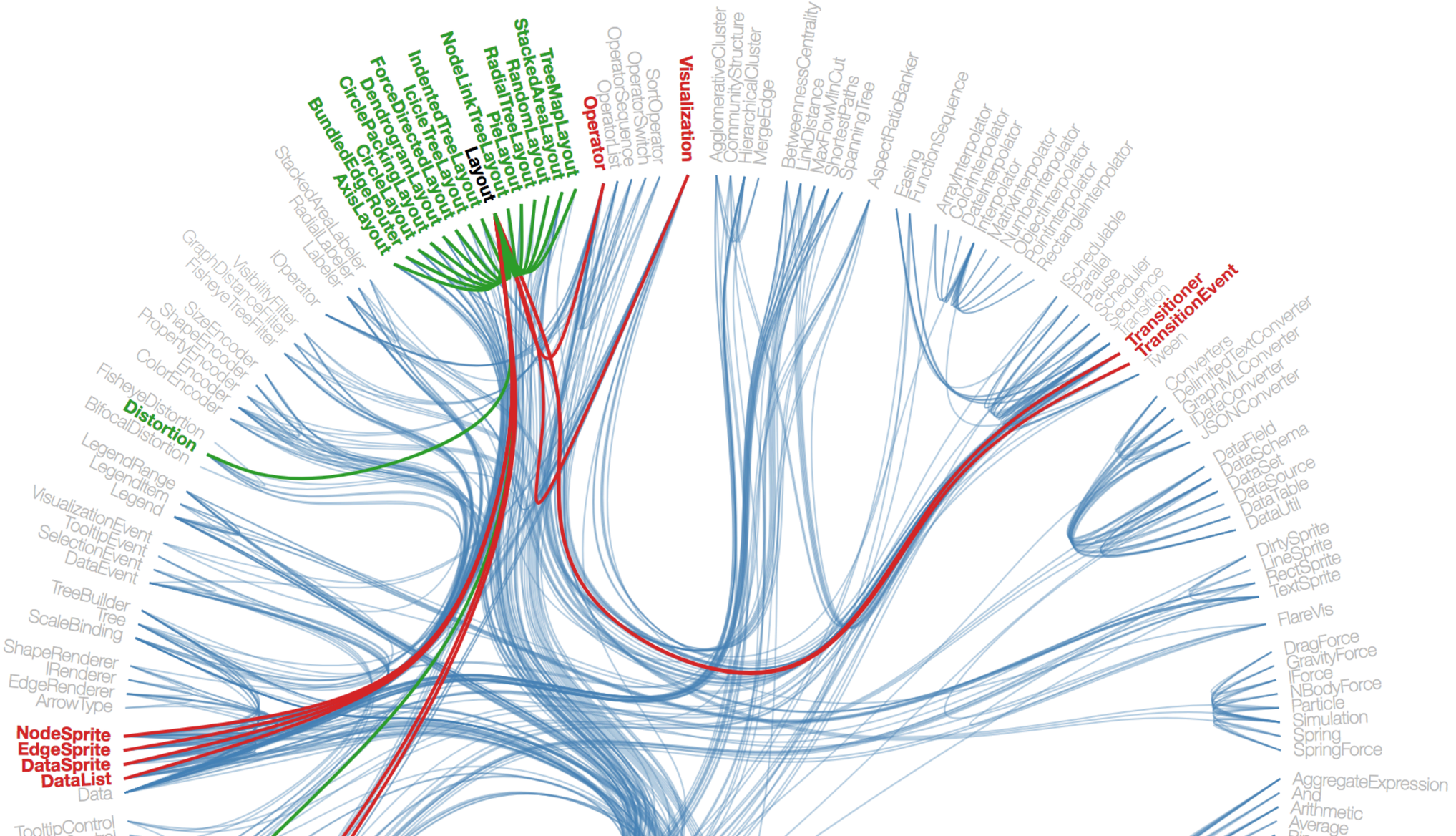
(c)



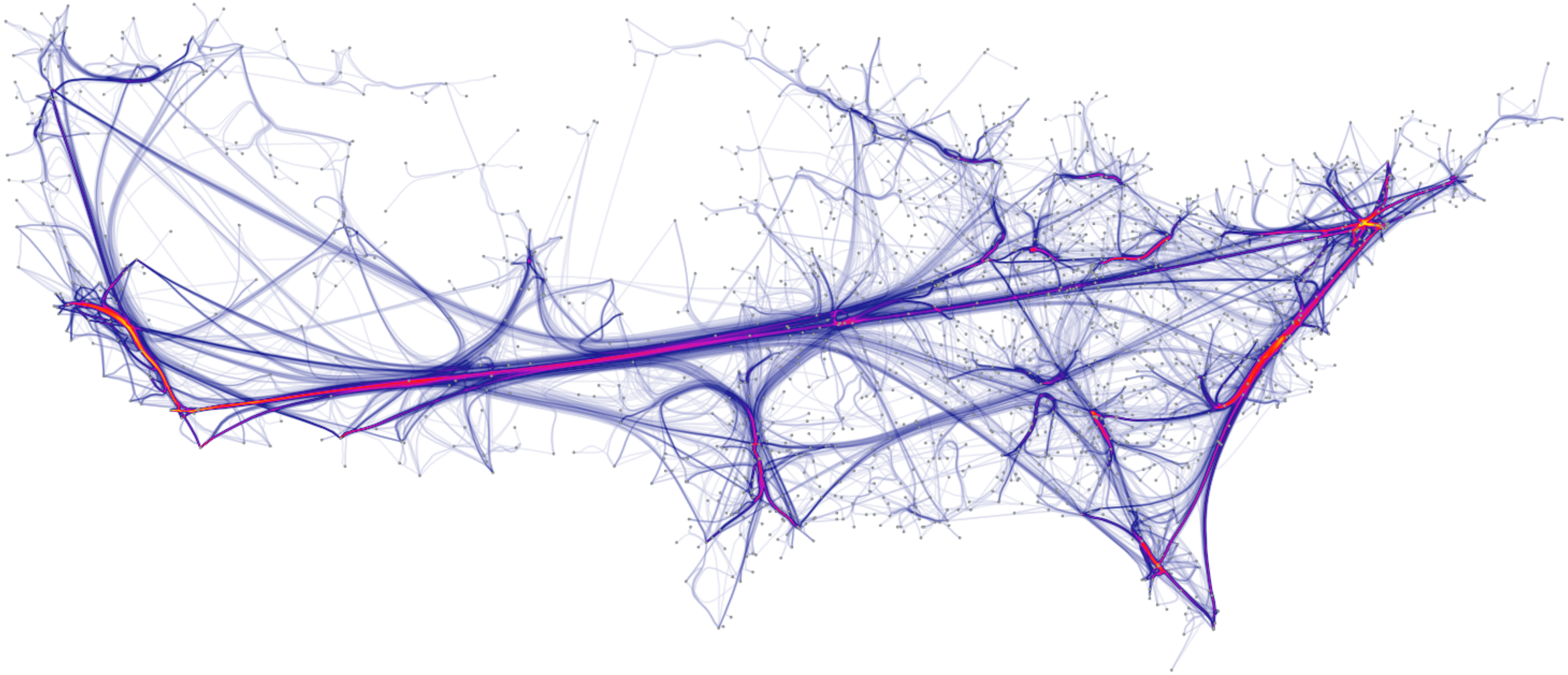


[Taken from Lex]


Mitth'Raw'Nuruodo
Wedge Antilles
Anakin Skywalker
Luke Skywalker
Palpatine
Boba Fett
Chewbacca
Han Solo
Lando Calrissian
Darth Caedus
Mara Jade Skywalker
Obi-Wan Kenobi
Padmé Amidala
Jaden Korr
Winda
Dooku
Ahsoka Tano
Yoda
Jango Fett
Revan


TreeMapLayout
StackedAreaLayout
RandomLayout
RadialTreeLayout
PieLayout
NodeLinkTreeLayout
Layout
IndentedTreeLayout
IcicleTreeLayout
ForceDirectedLayout
DendrogramLayout
CirclePackingLayout
CircleLayout
BundledEdgeRouter
AxisLayout
Visualization
Operator
OpSort
OpOperatorSequence
OpOperatorSwitch
SortOperator
OperatorSequence
OperatorSwitch
OperatorList
AgglomerativeCluster
CommunityStructure
HierarchicalCluster
MergeEdge
BetweennessCentrality
LinkDistance
MaxFlowMinCut
ShortestPaths
SpanningTree
AspectRatioBanker
Easing
FunctionSequence
ArrayInterpolator
ColorInterpolator
DateInterpolator
Interpolator
MatrixInterpolator
NumberInterpolator
ObjectInterpolator
PointInterpolator
RectangleInterpolator
ISchedulable
Parallel
Pause
Scheduler
Sequence
Transition
Transitioner
TransitionEvent
Tween
StackedAreaLabeler
RadialLabeler
Labeler
IOperator
VisibilityFilter
GraphDistanceFilter
FisheyeTreeFilter
SizeEncoder
ShapeEncoder
PropertyEncoder
Encoder
ColorEncoder
FisheyeDistortion
BifocalDistortion
Distortion
LegendRange
LegendItem
Legend
VisualizationEvent
TooltipEvent
SelectionEvent
DataEvent
TreeBuilder
Tree
ScaleBinding
ShapeRenderer
IRenderer
EdgeRenderer
ArrowType
NodeSprite
EdgeSprite
DataSprite
DataList
Data
TooltipControl
Converters
DelimitedTextConverter
GraphMLConverter
IDataConverter
JSONConverter
DataField
DataSchema
DataSet
DataSource
DataTable
DataUtil
DirtySprite
LineSprite
RectSprite
TextSprite
FlareVis
DragForce
GravityForce
IForce
NBodyForce
Particle
Simulation
Spring
SpringForce
AggregateExpression
And
Arithmetic
Average

Fixed layout



[Holten]


facebook
December 2010

Node-link diagrams

Pros

- Visualize all graph classes
- Very flexible in the layout
- Good for topology related queries (if good layout)

Cons

- Tendency to clutter
- Hard to find a good layout (lies in NP)
- Even heuristics are still slow/complex to run

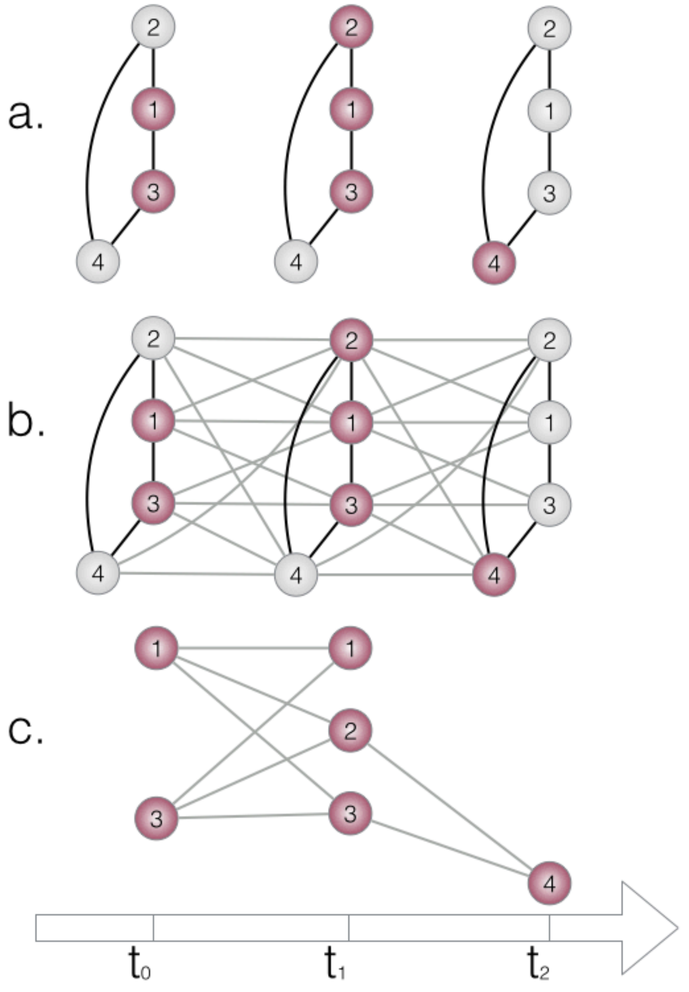
Multivariate graphs

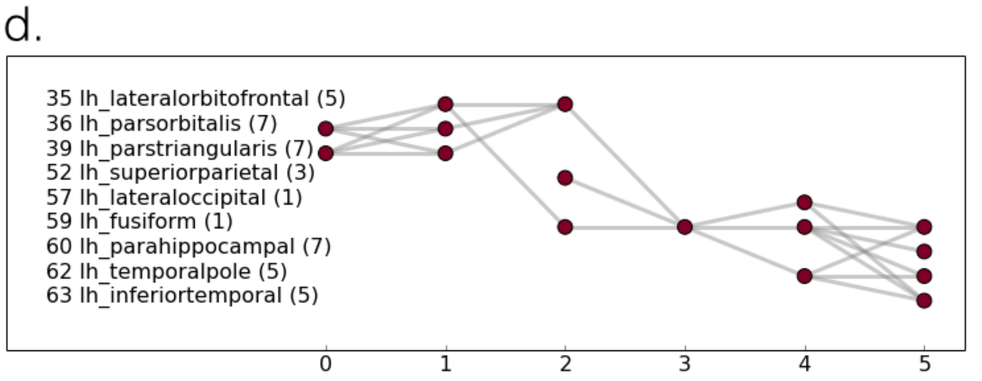
Multivariate Graphs

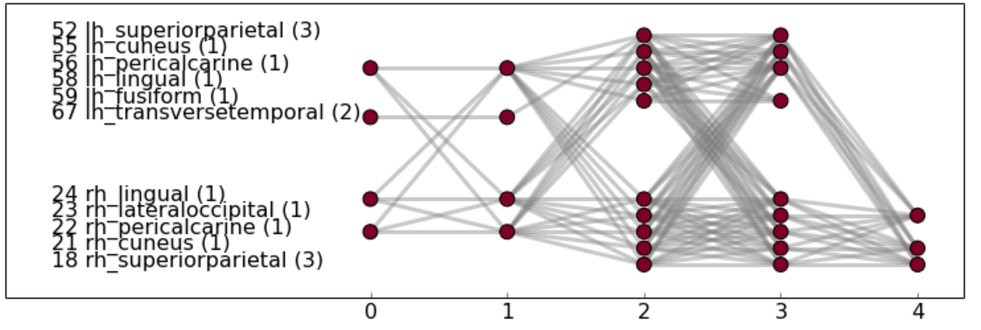
How to represent both the topology and attributes on the graph?

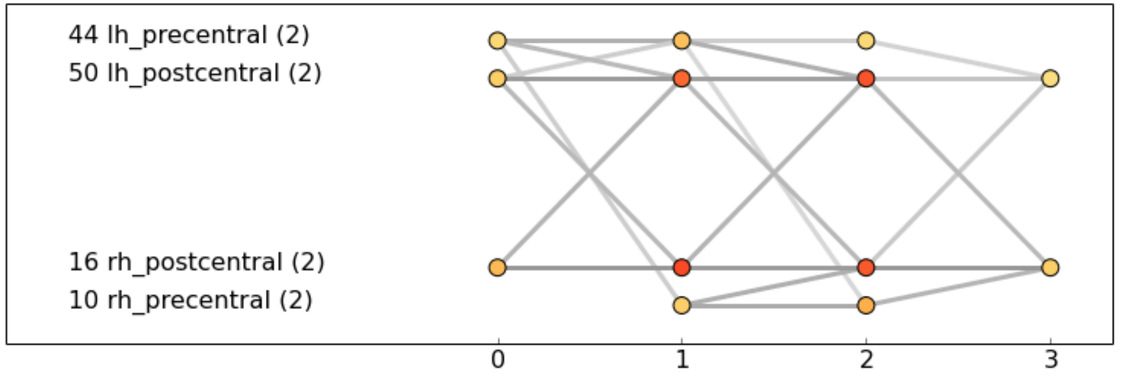
Attributes can also influence topology

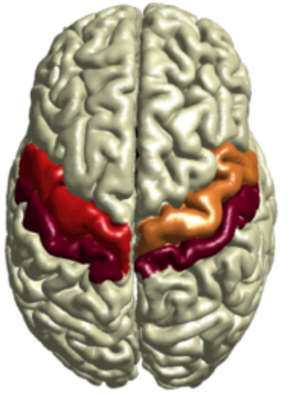
Show/hide edges or paths


a.
b.
c.
t0
t1
t2


d.
35 lh_lateralorbitofrontal (5)
36 lh_parsorbitalis (7)
39 lh_parstriangularis (7)
52 lh_superiorparietal (3)
57 lh_lateraloccipital (1)
59 lh_fusiform (1)
60 lh_parahippocampal (7)
62 lh_temporalpole (5)
63 lh_inferiortemporal (5)
0
1
2
3
4
5


e.
52 lh_superiorparietal (3)
55 lh_cuneus (1)
56 lh_pericalcarine (1)
58 lh_lingual (1)
59 lh_fusiform (1)
67 lh_transversetemporal (2)
24 rh_lingual (1)
23 rh_lateraloccipital (1)
22 rh_pericalcarine (1)
21 rh_cuneus (1)
18 rh_superiorparietal (3)
0
1
2
3
4


44 lh_precentral (2)
50 lh_postcentral (2)
16 rh_postcentral (2)
10 rh_precentral (2)
0
1
2
3

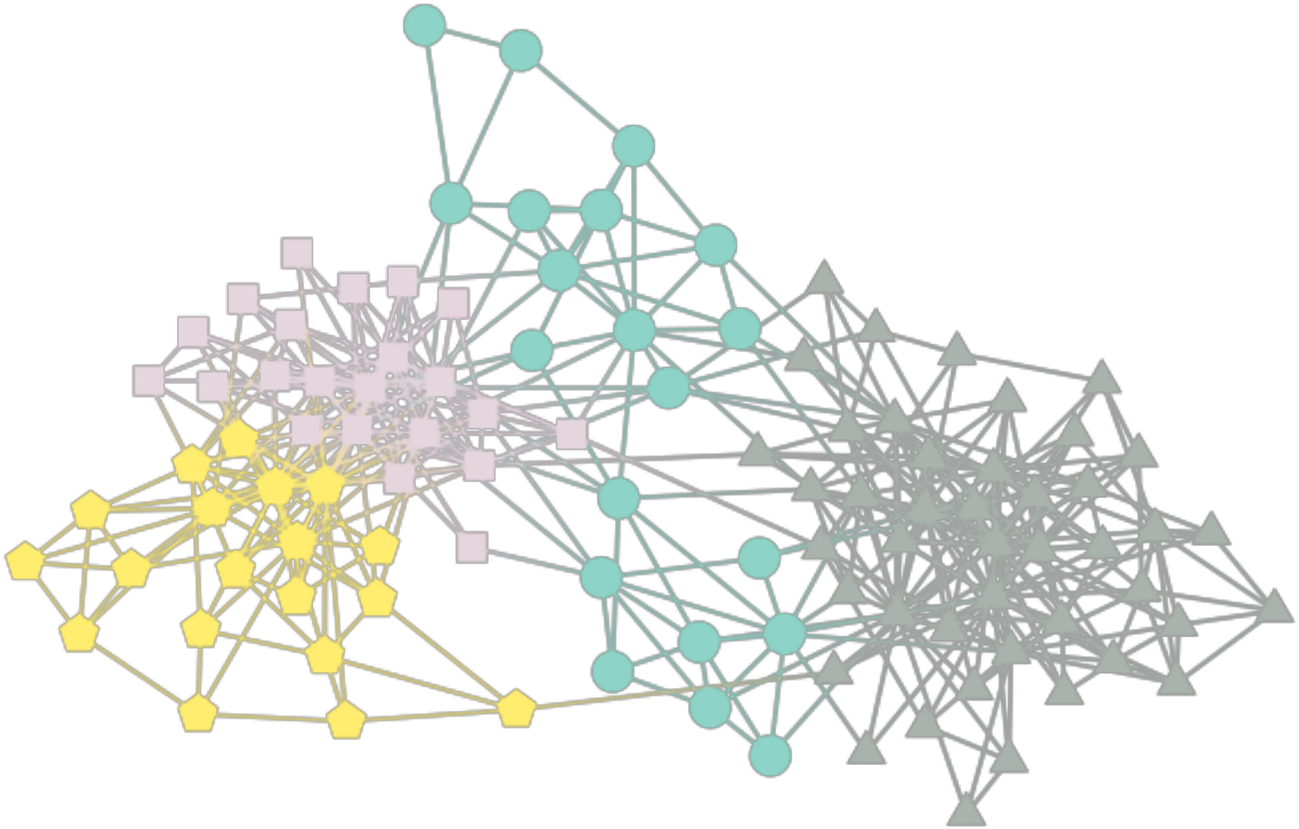


[Benzi et al.]

Node attributes

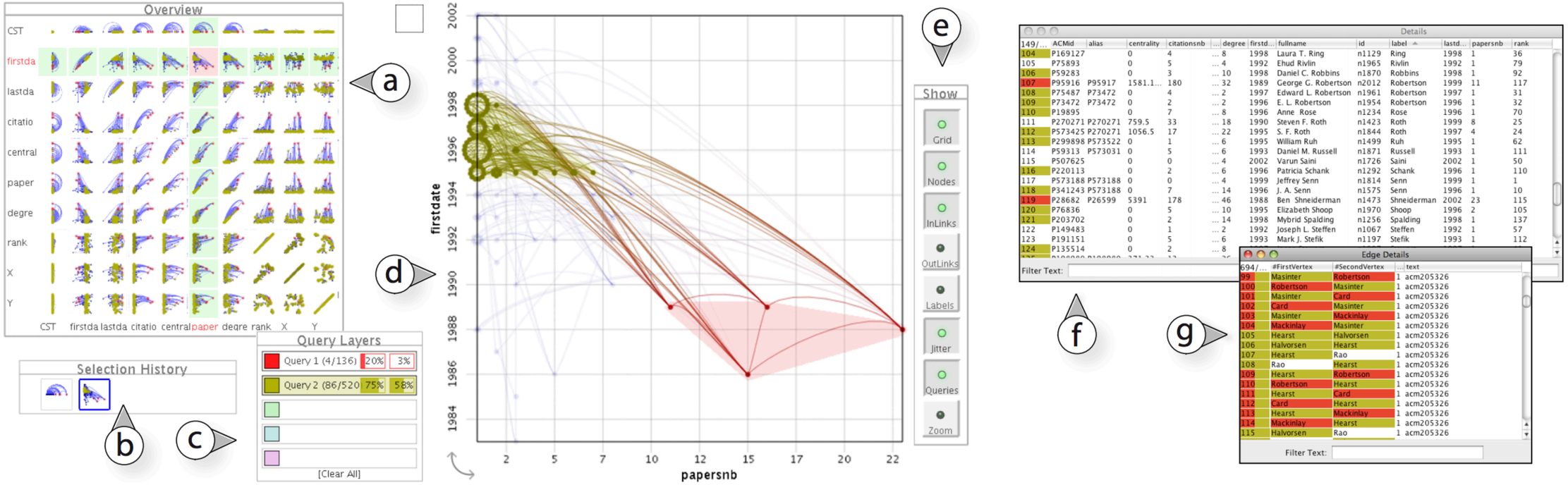
Natural choice is color or glyphs

Limited in scalability



GraphDice

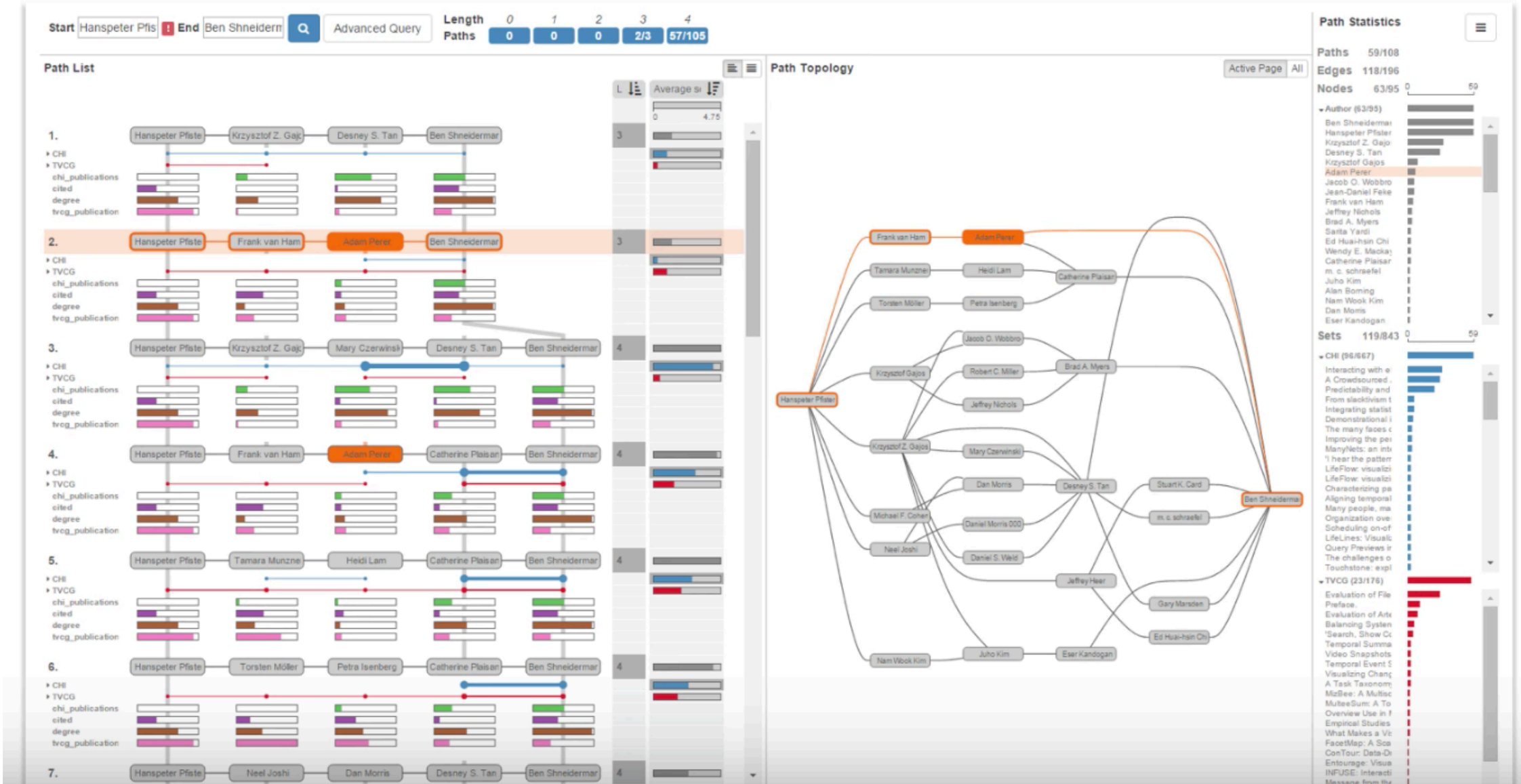
Nodes are positionned in space according to attribute values



[Bezerianos et al, 2010]

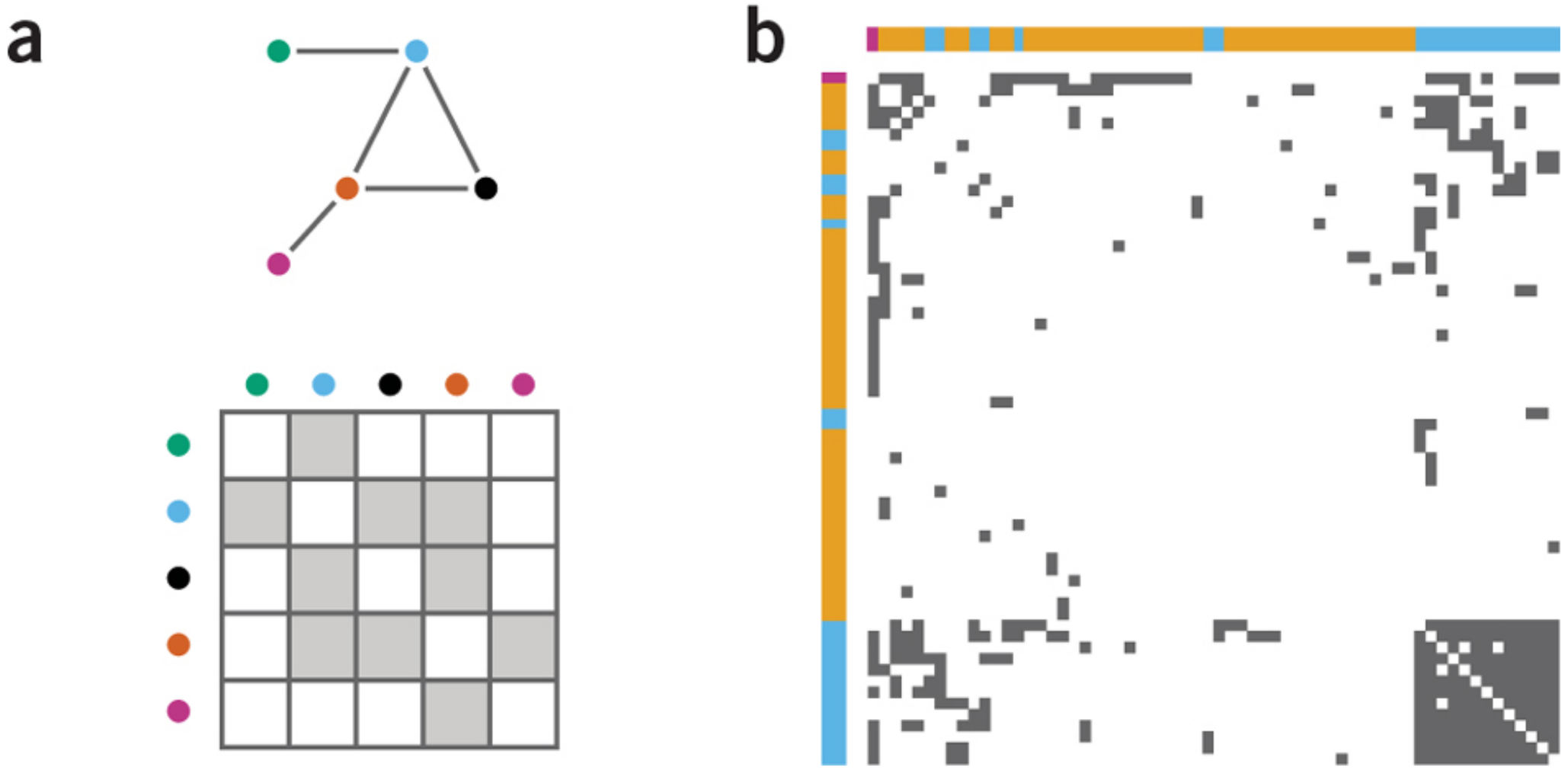
PathFinder

[Partl et al.]

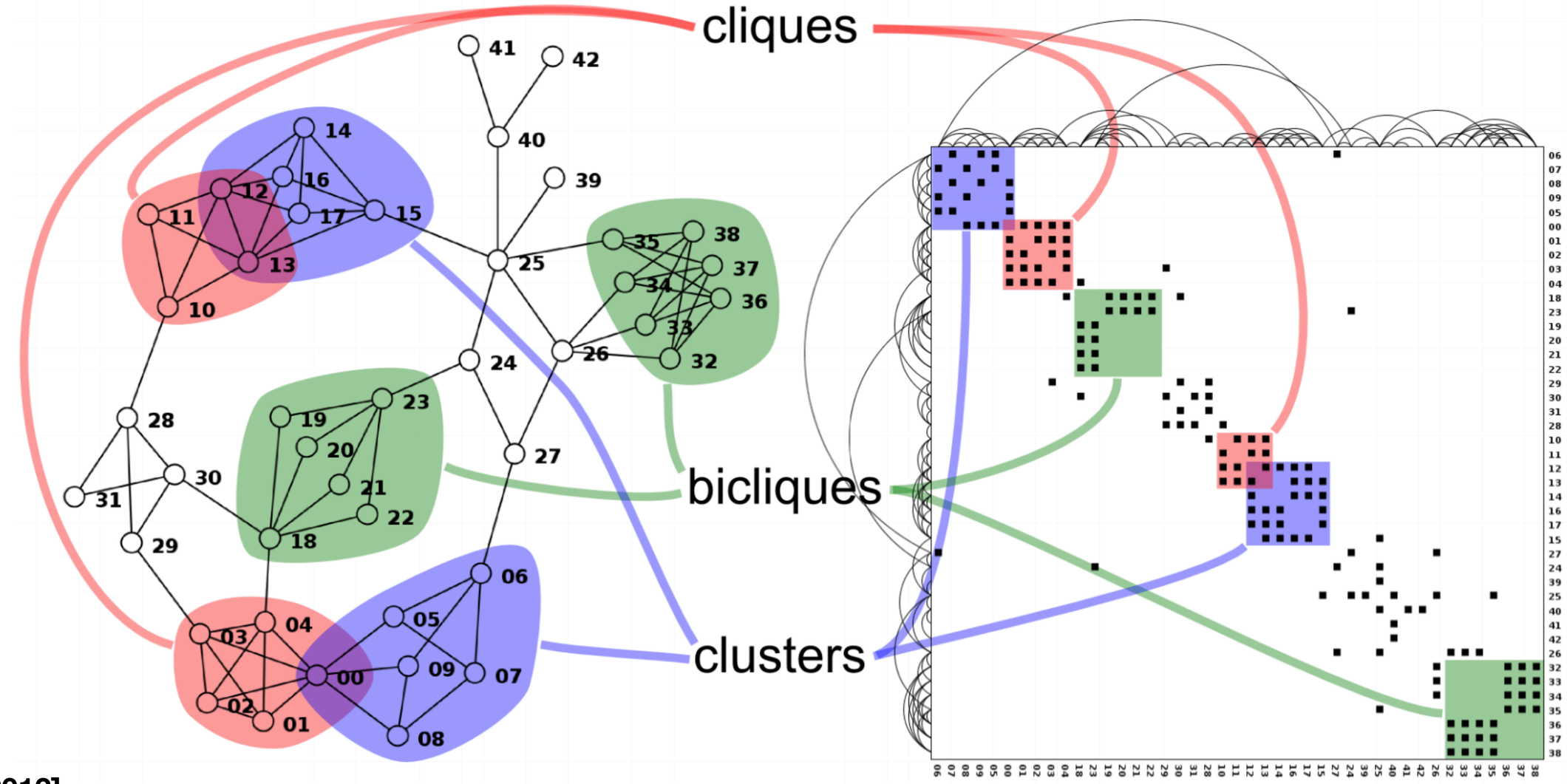


Matrix representation

Matrix representation

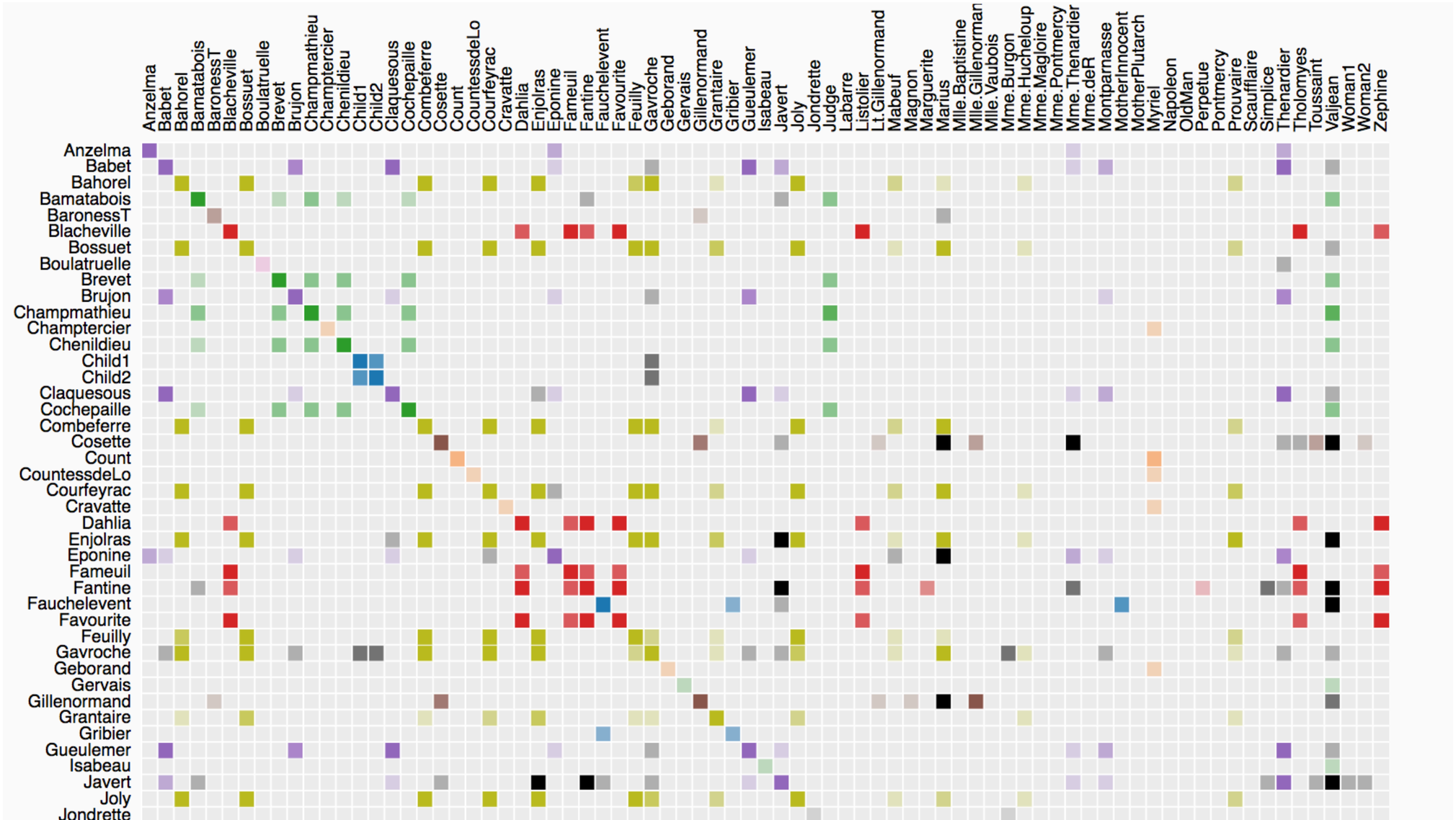


Good for neighborhood tasks

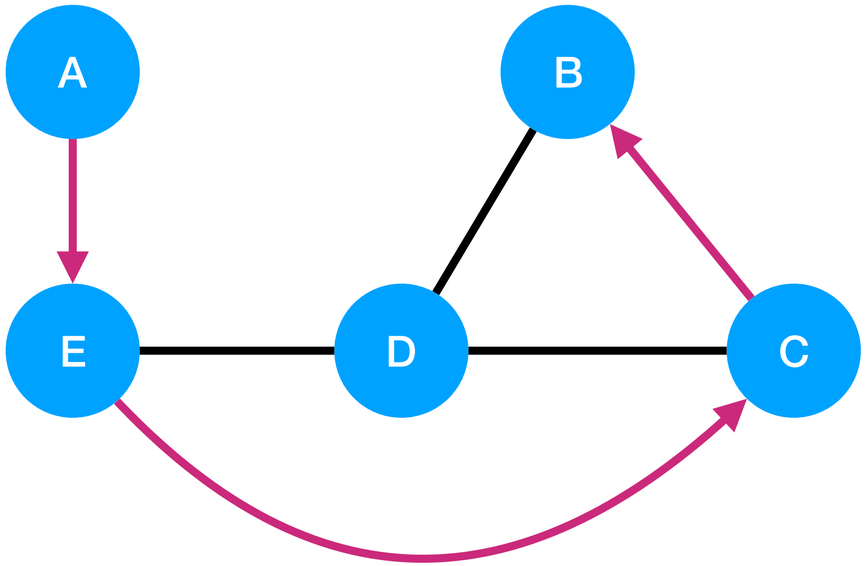


[McGuffin 2012]

Node ordering is paramount!



Don’t use for paths!



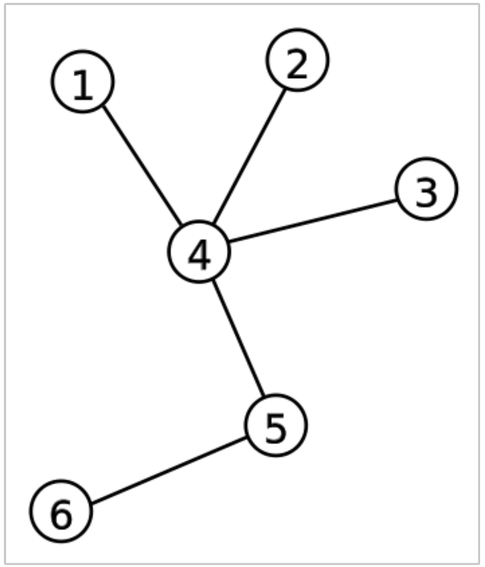
	A	B	C	D	E
A					1
B					
C		3			
D					
E			2		

	Matrix-based representations	Node-link diagrams
+	No node overlapping No edge crossing Readable for dense graphs Fast navigation Fast manipulation More readable for some tasks	Intuitive Compact More readable for path following More effective for small graphs More effective for sparse graphs
–	Less intuitive Use more space Weak for path following tasks	Useless without layout Node overlapping Edge crossing Not readable for dense graphs Manipulation requires layout computation

[Heymann 2014]

Tree visualization

Trees

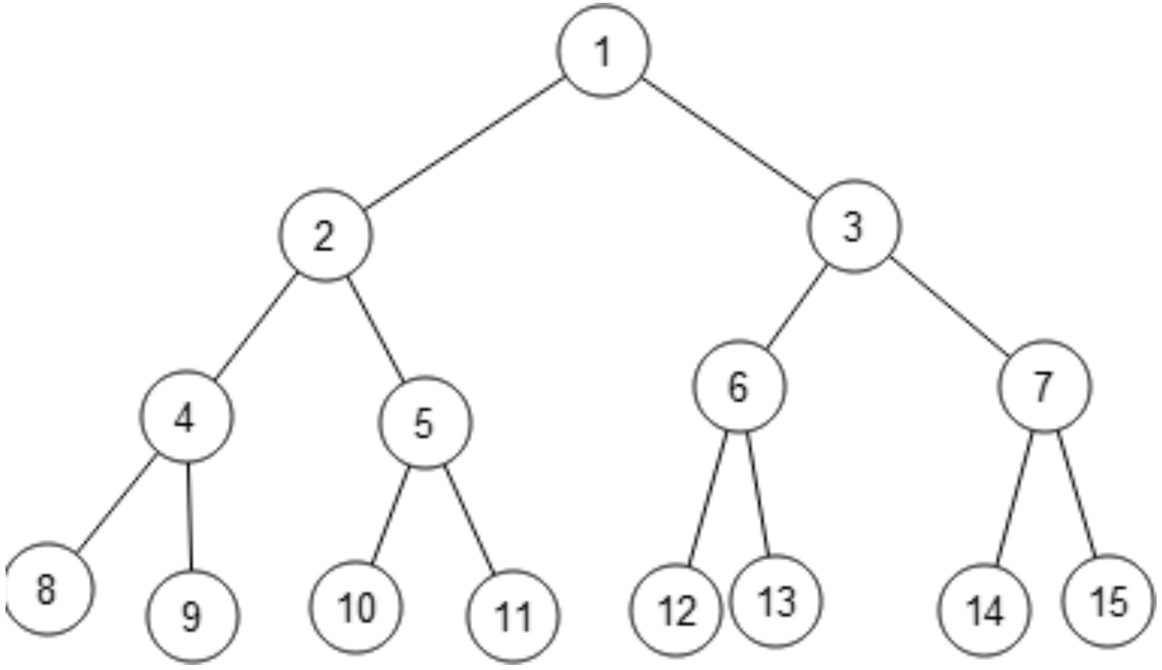


A tree is a graph with constraints:

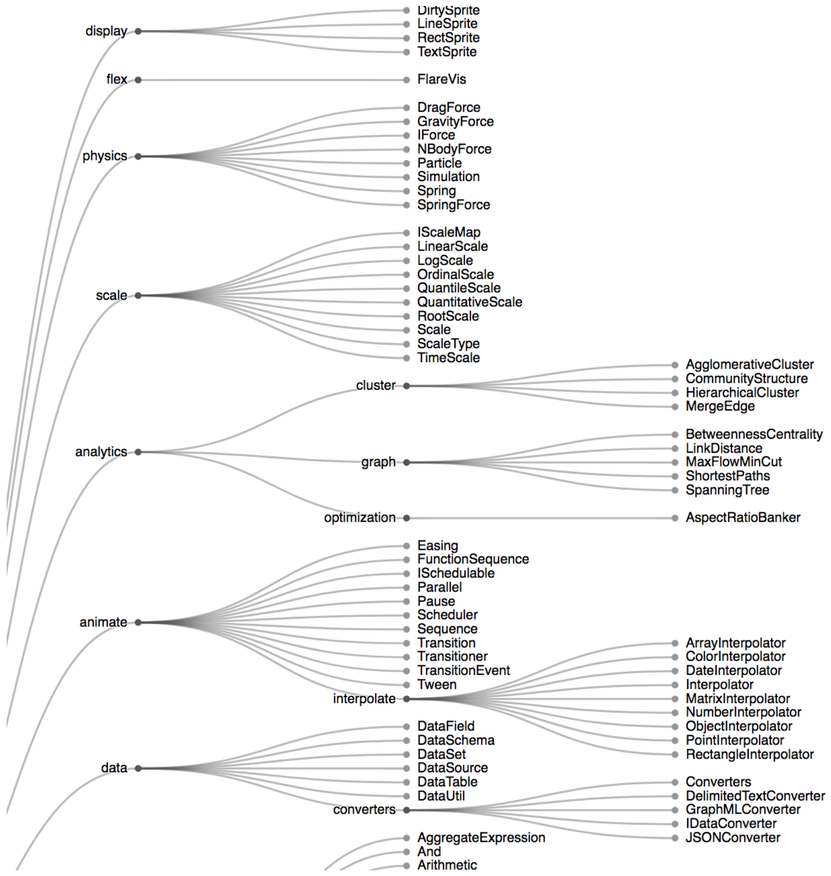
no cycles

can have a root node and subtrees

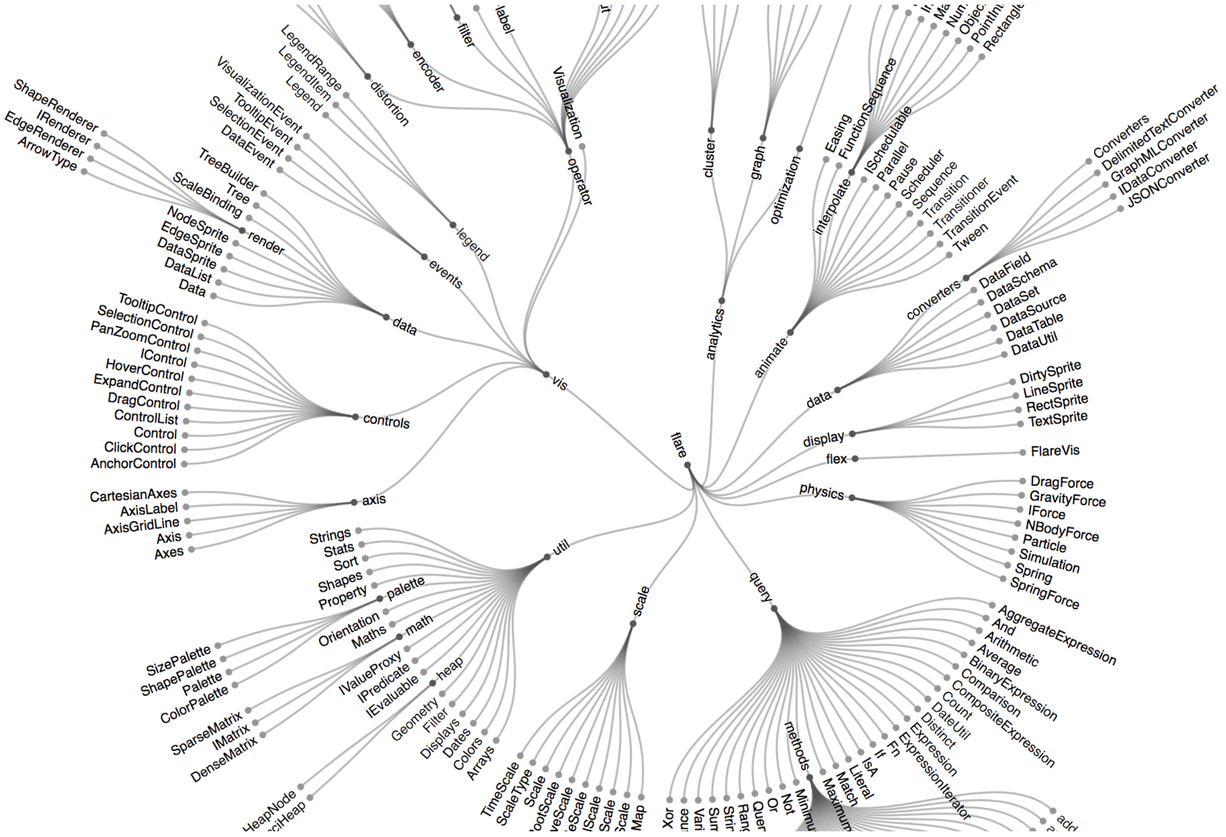
can be ordered



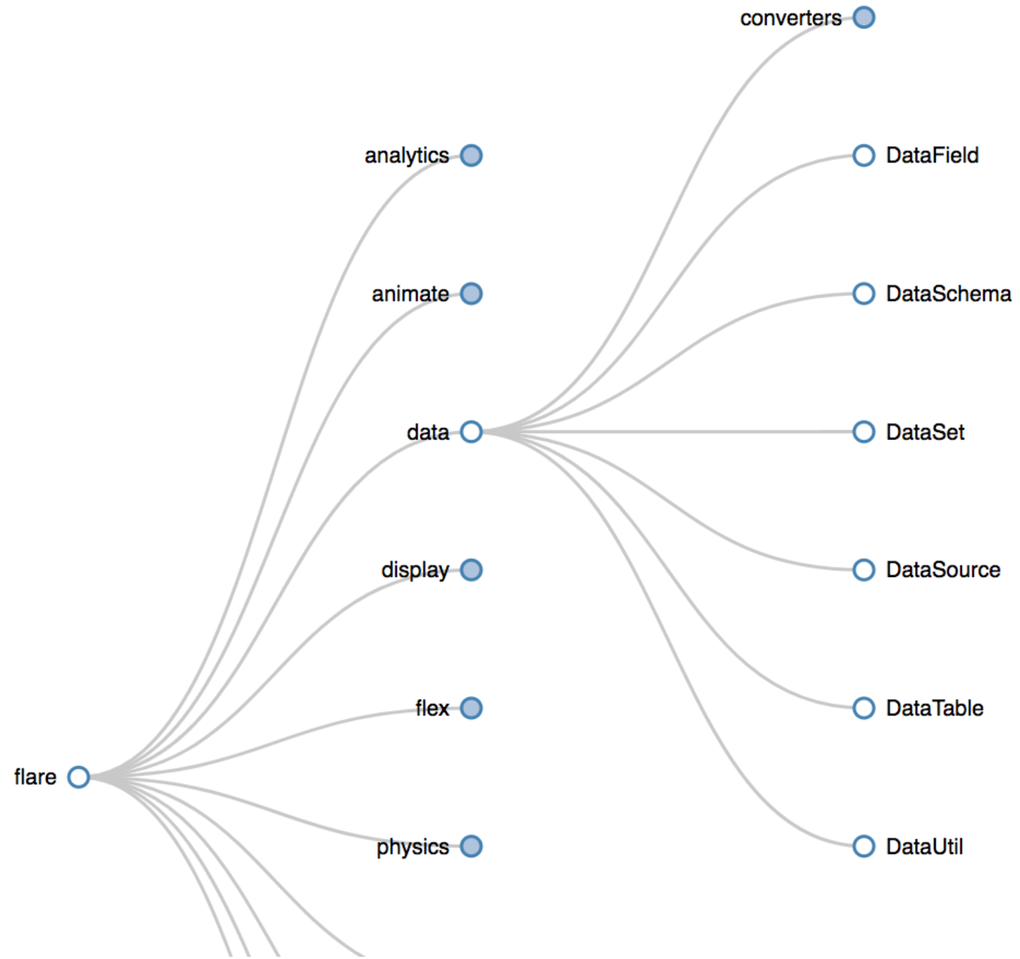
Node-link diagram



Reingold–Tilford layout

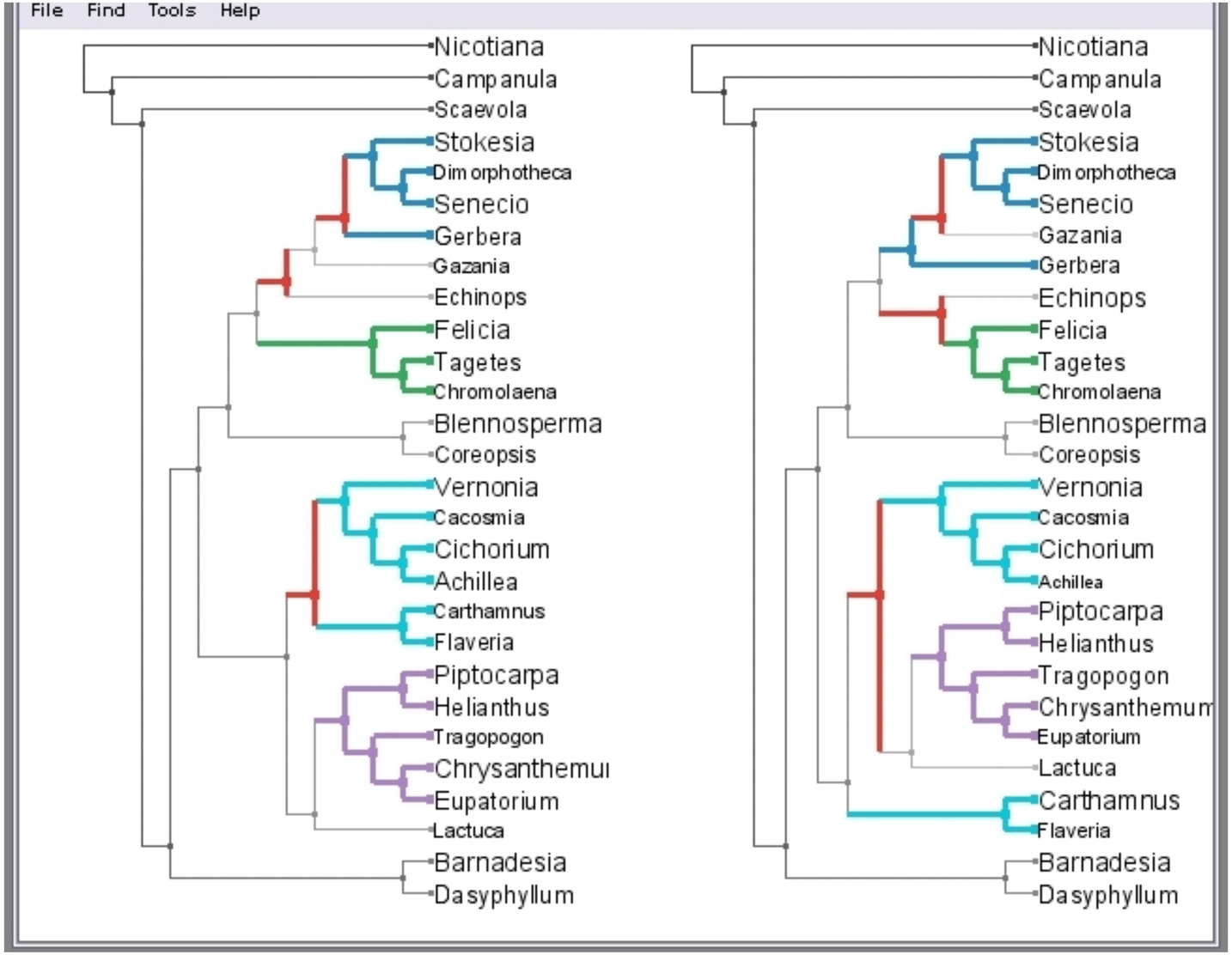


Collapsible Tree



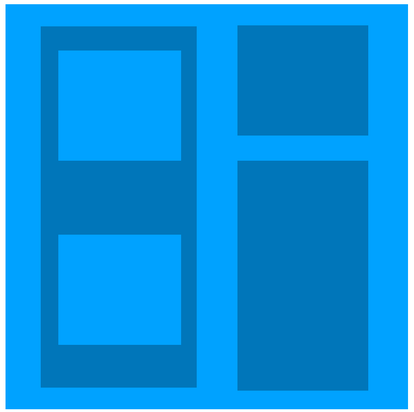
[Heer]

TreeJuxtaposer



[Munzner et al.]

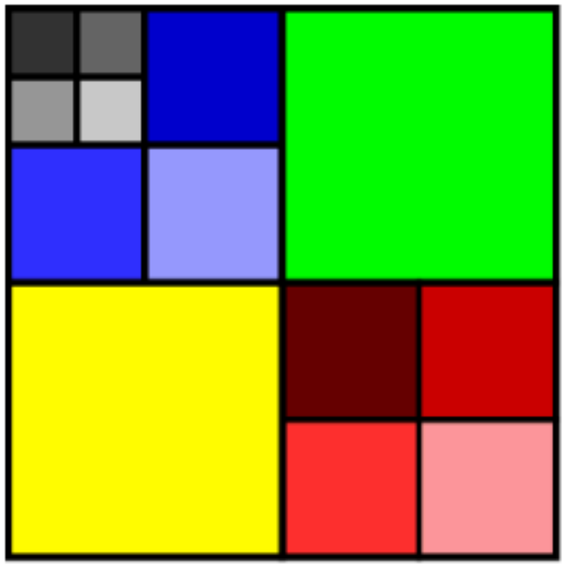
TreeMap

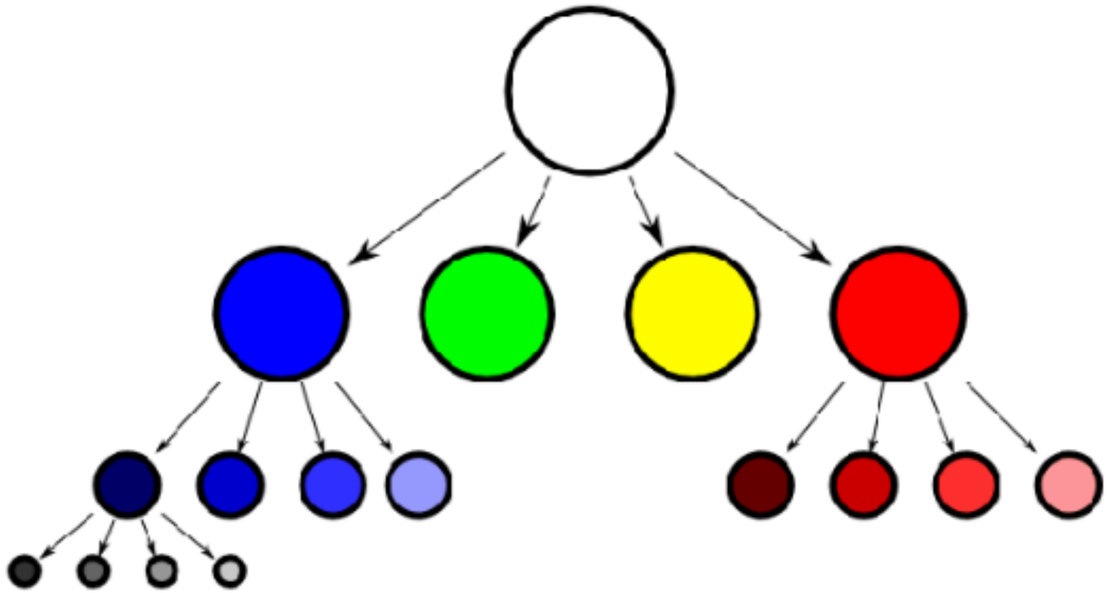


Implicit layout for tree

Display hierarchical (tree-structured) data as a set of nested rectangles

Efficient use of space



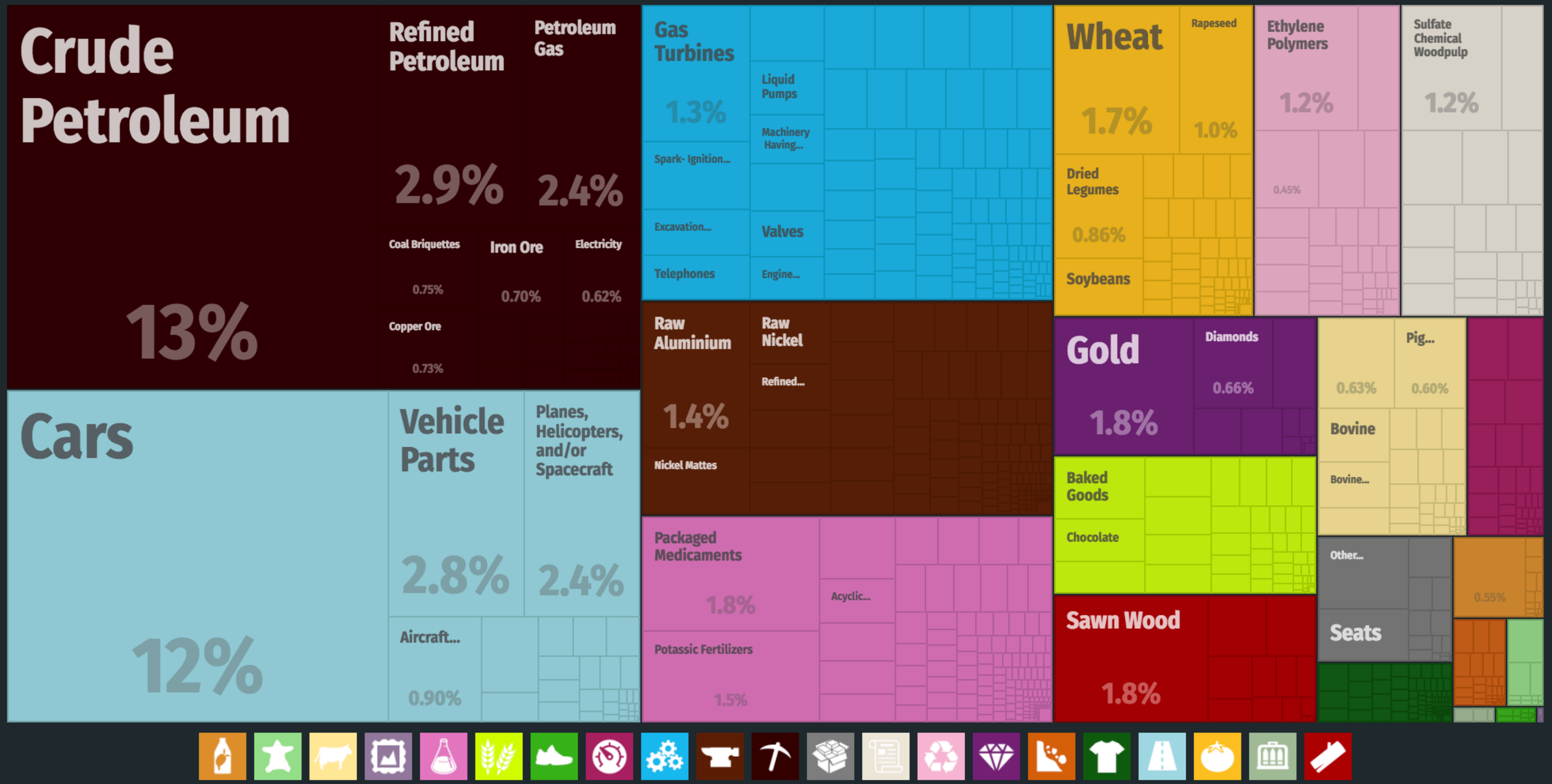


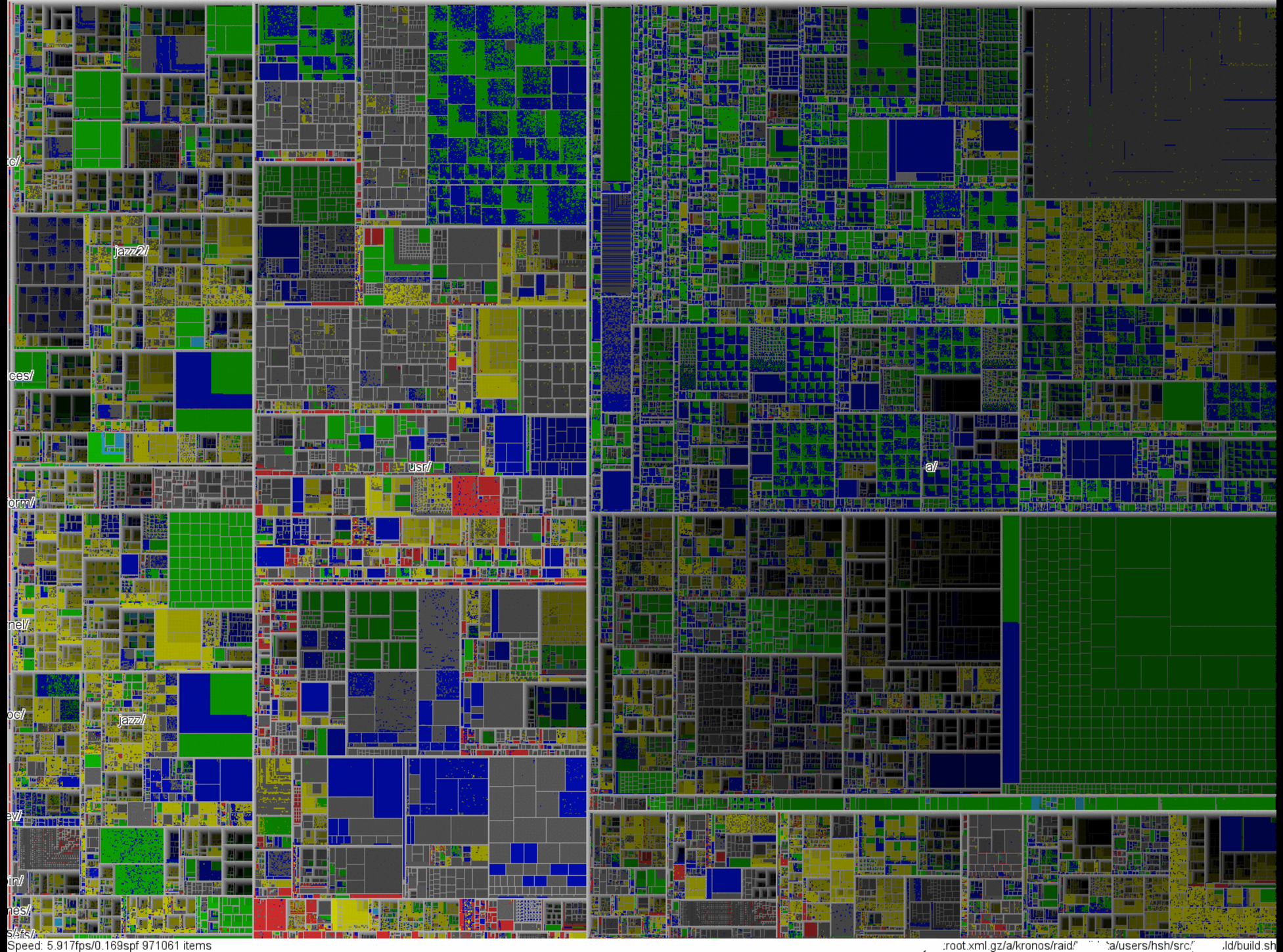
[Johnson and Shneiderman 1991]

< 2014

What does Canada export? (2015)

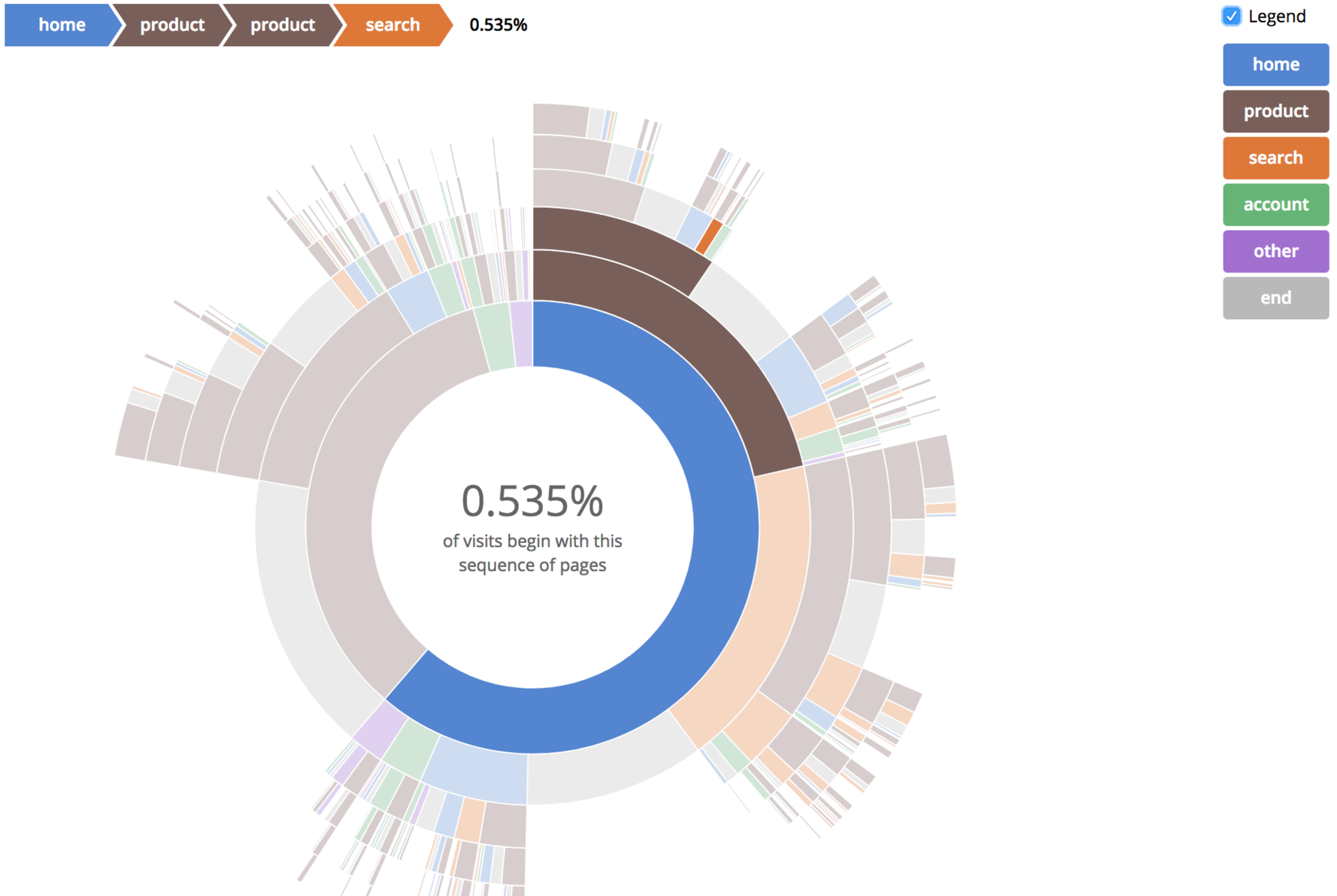
TOTAL: $390B





Speed: 5.917fps/0.169spf 971061 items :root.xml.gz/a/kronos/raid/ a/users/hsh/src/ ld/build.sh

Radial layout



Viz by Rodden

[Stasko et al.]

Implicit visualization

Pros

Space-efficient because there is no need to draw the edges

Scale well up to large graphs

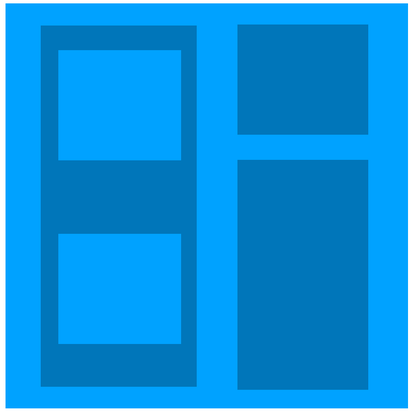
Useful for Attribute based taks on nodes, also useful for simple paths related tasks

Cons

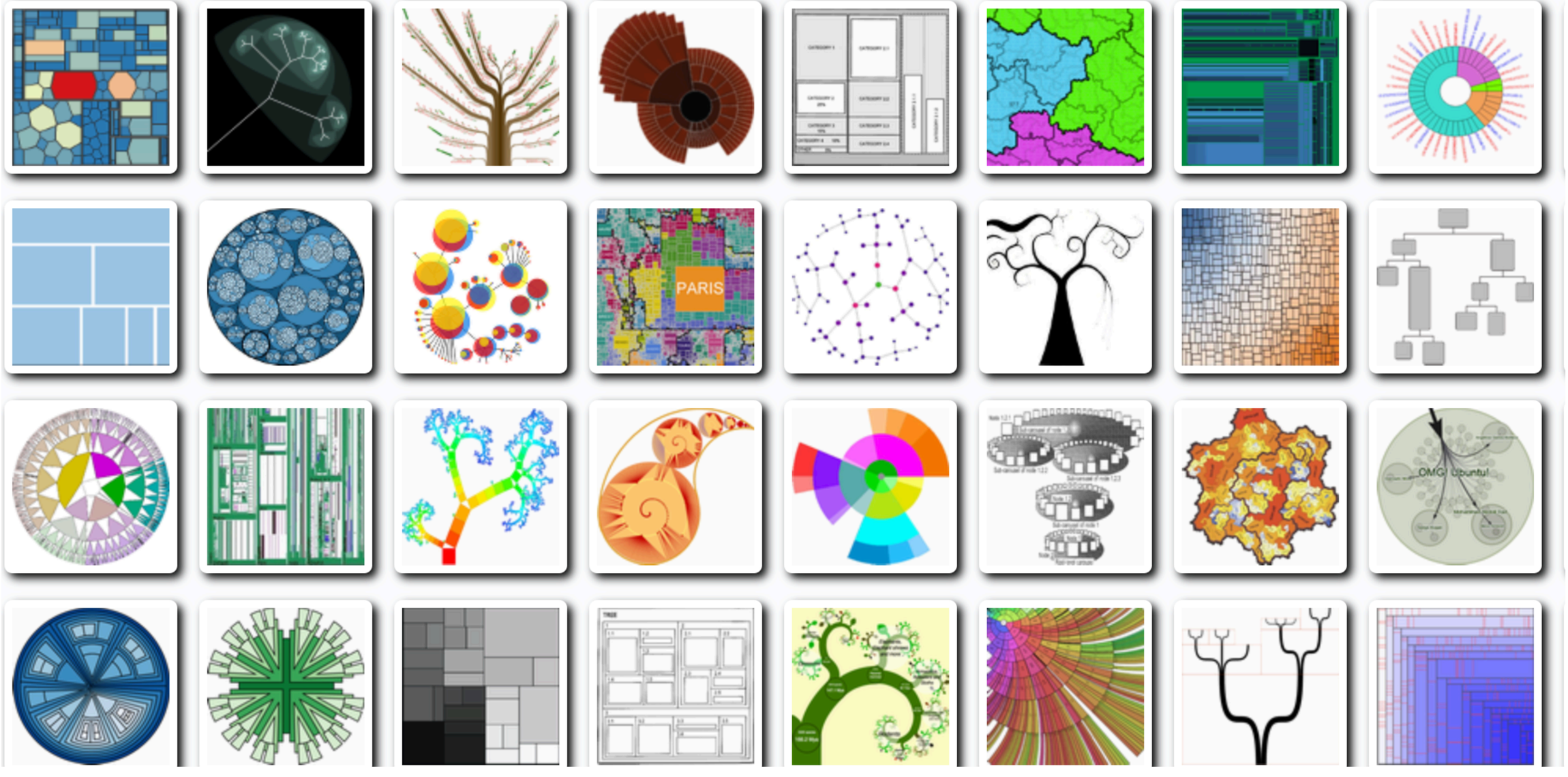
Only works for trees

Node cannot be freely arranged in space, no edges

Spatial relations such as overlap or inclusion lead to occlusion



More on tree viz



Tools, libraries

Cytoscape.js

Graph theory / network library for analysis and visualisation

Supports CommonJS/Node.js/Browserify/Webpack, AMD/Require.js, jQuery, npm, CDNJS, Bower, jspm, Meteor/Atmosphere, and plain JS/JavaScript

GitHub | Download 3.2.3

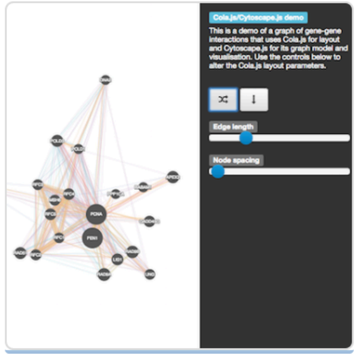
Extensions | News & tutorials | Twitter | Gitter | File a GitHub issue | Contribute | Search previous questions via Stack Overflow (or ask a different question)

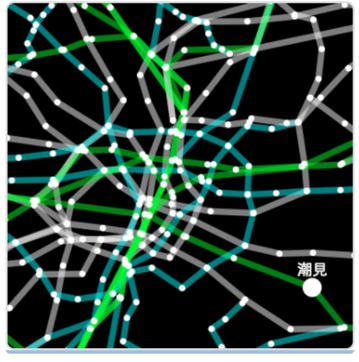
Like Cytoscape.js? Donate | Star it on GitHub | Share it on Twitter | Share it on Google+

Donations are tax deductible to a 501(c)(3) nonprofit organization, The Cytoscape Consoritum, Tax ID: 20-4909879.

Use Cytoscape.js? Let us know

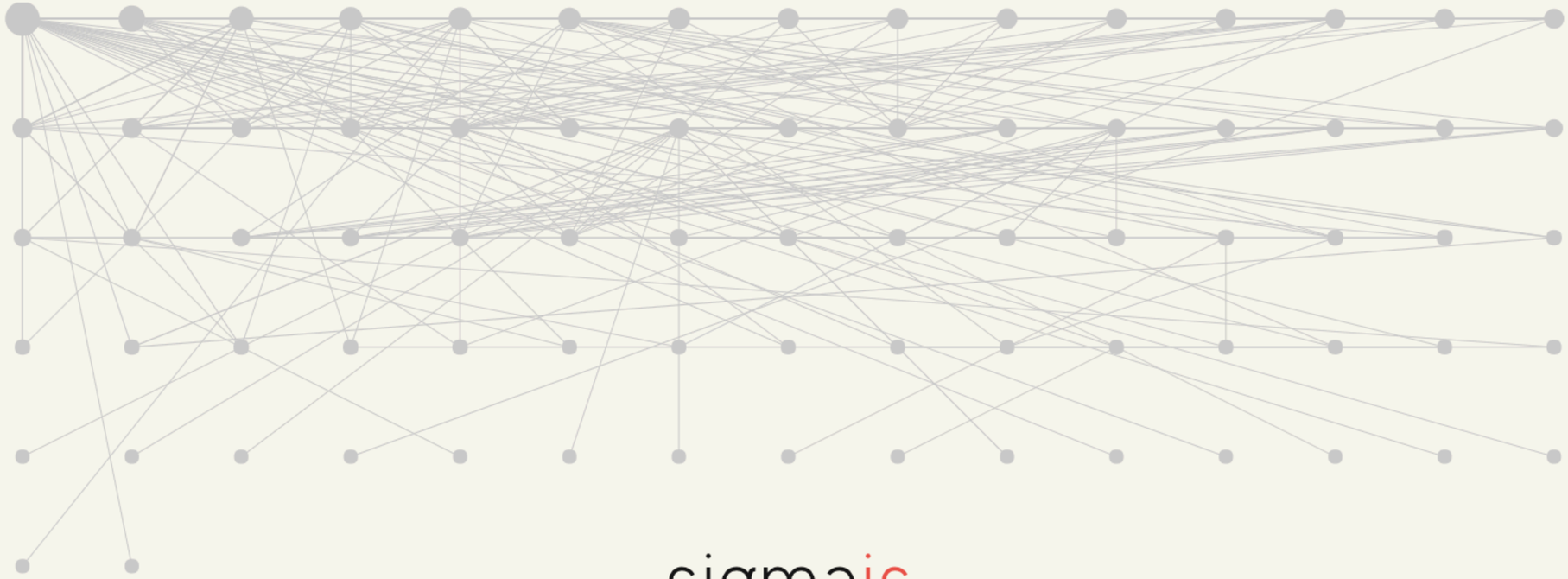
Demos





sigmajs

GET STARTED FEATURES USE CASES TUTORIAL REFERENCES



sigmajs

TUTORIAL v1.2.0 DOWNLOAD

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.

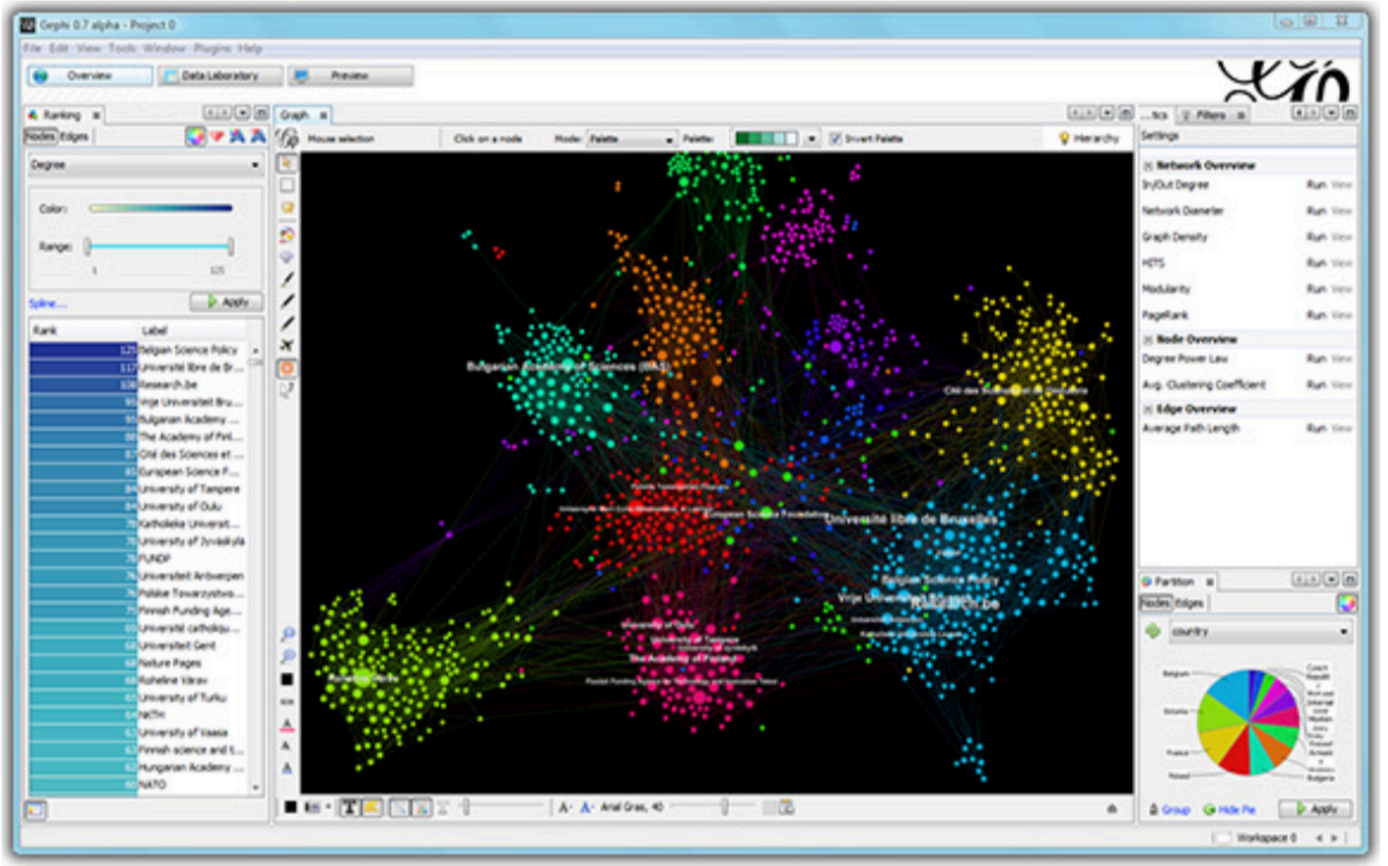
Runs on Windows, Mac OS X and Linux.

Learn More on Gephi Platform »

Download FREE
Gephi 0.9.1

Release Notes | System Requirements

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



graph-tool | Efficient network analysis

Download | Documentation | Mailing List | Git | Issues

What is graph-tool?

Graph-tool is an efficient Python module for manipulation and statistical analysis of graphs (a.k.a. networks). Contrary to most other python modules with similar functionality, the core data structures and algorithms are implemented in C++, making extensive use of template metaprogramming, based heavily on the Boost Graph Library. This confers it a level of performance that is comparable (both in memory usage and computation time) to that of a pure C/C++ library.

Download version 2.22

See Instructions | See Changelog

It is *Fast!*

Extensive Features

Powerful Visualization

NetworkX

Stable (notes)

1.11 — January 2016
download | doc | pdf

Latest (notes)

2.0 development
github | doc | pdf

Archive

Contact

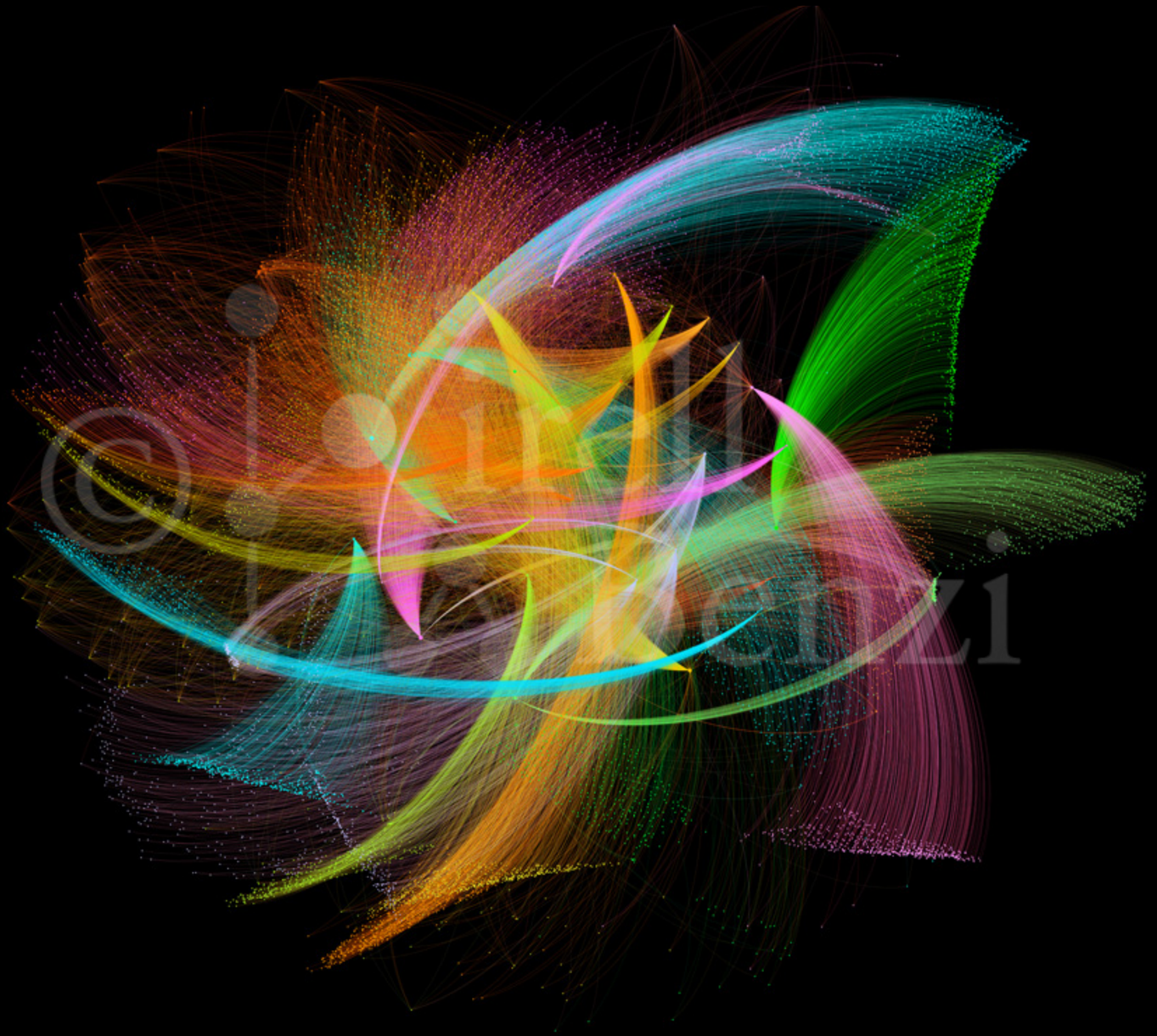
Mailing list
Issue tracker

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



Kirell Benzi

kirellbenzi.com

"ART UPSETS, SCIENCE REASSURES" GEORGES BRAQUE