Welcome (Again!) to
MATH 4100/COMP 5360– Introduction to Data Science

Network Analysis

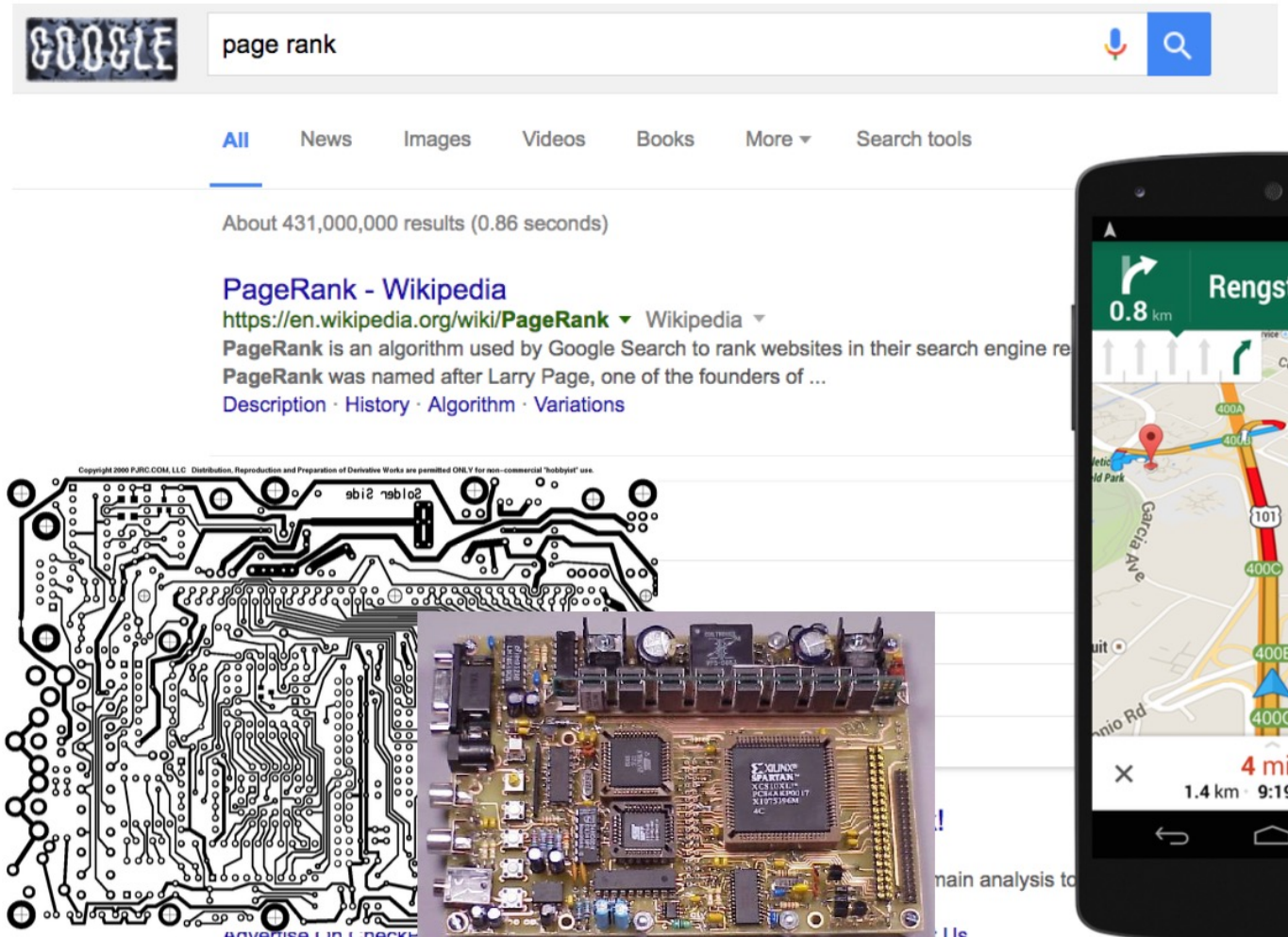
April 18, 2024

Based in part on prior lectures from
Alex Lex and Bei Wang Phillips

+ others where noted

Applications of Graphs

Without graphs, there would be none of these:`




facebook

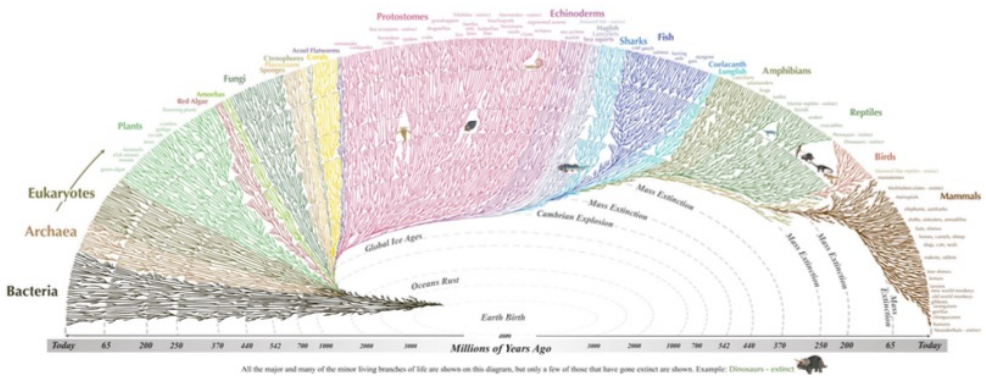
Biological Networks

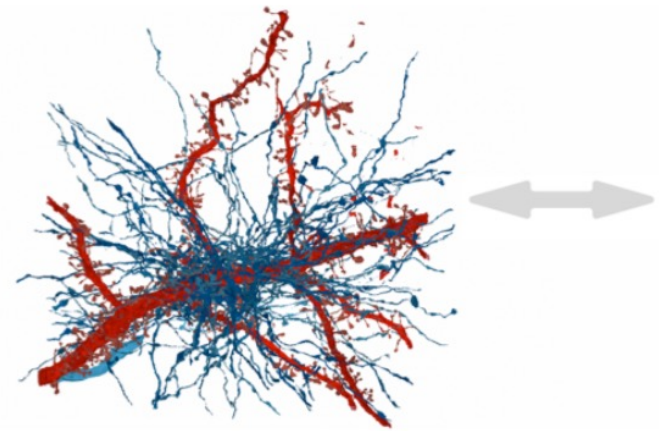
Interaction between genes, proteins and chemical products

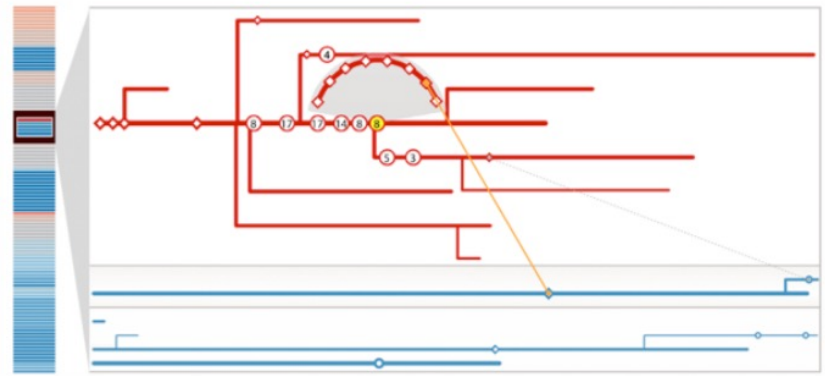
The brain: connections between neurons

Your ancestry: the relations between you and your family

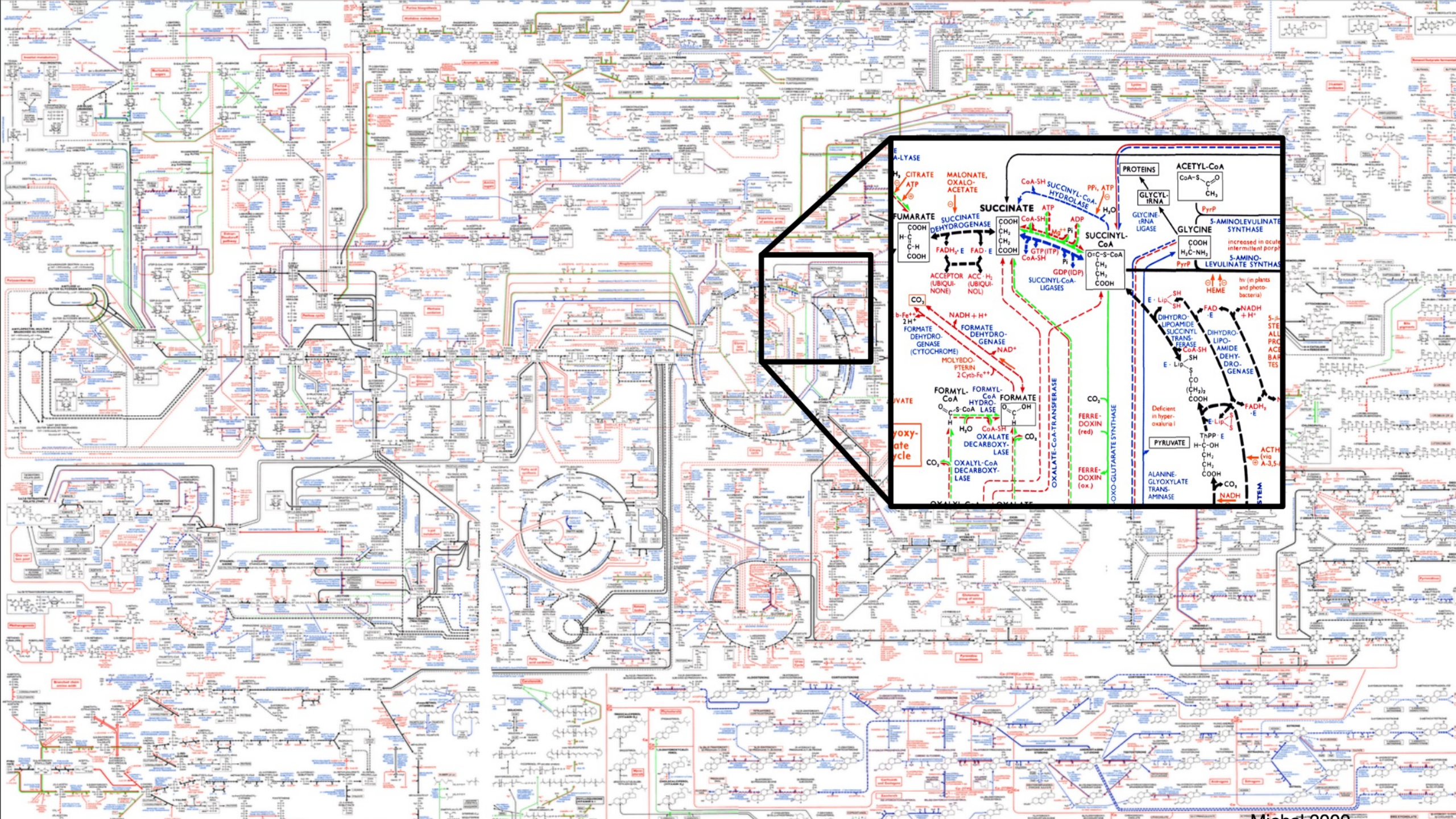
Phylogeny: the evolutionary relationships of life







[Beyer 2014]


SUCCINATE
FUMARATE
SUCCINATE DEHYDROGENASE
SUCCINYL-CoA HYDROLASE
SUCCINYL-CoA LIGASES
SUCCINYL-CoA
MALONATE, OXALO-ACETATE
CITRATE
ATP
ACCEPTOR (UBIQUI-NONE)
ACC·H2 (UBIQUI-NOL)
FORMATE DEHYDRO-GENASE
FORMATE DEHYDRO-GENASE (CYTOCHROME)
MOLYBDO-PTERIN
FORMYL-CoA
FORMYL-CoA HYDRO-LASE
FORMATE
OXALATE DECARBOXY-LASE
OXALYL-CoA DECARBOXY-LASE
OXALATE-CoA-TRANSFERASE
FERRE-DOXIN (red)
FERRE-DOXIN (ox.)
OXO-GLUTARATE SYNTHASE
PROTEINS
GLYCYL-tRNA
GLYCINE-tRNA LIGASE
GLYCINE
ACETYL-CoA
5-AMINOLEVULINATE SYNTHASE
5-AMINO-LEVULINATE SYNTHAS
HEME
hν (in plants and photo-bacteria)
DIHYDRO-LIPOAMIDE SUCCINYL TRANS-FERASE
DIHYDRO-LIPO-AMIDE DEHY-DRO-GENASE
Deficient in hyper-oxaluria I
PYRUVATE
ALANINE-GLYOXYLATE TRANS-AMINASE
ACTH (via A-3,5-
NADH + H+
NAD+
FADH2·E
FAD·E
GTP(ITP)
GDP(IDP)
ADP
PPi, ATP
H2O
CO2

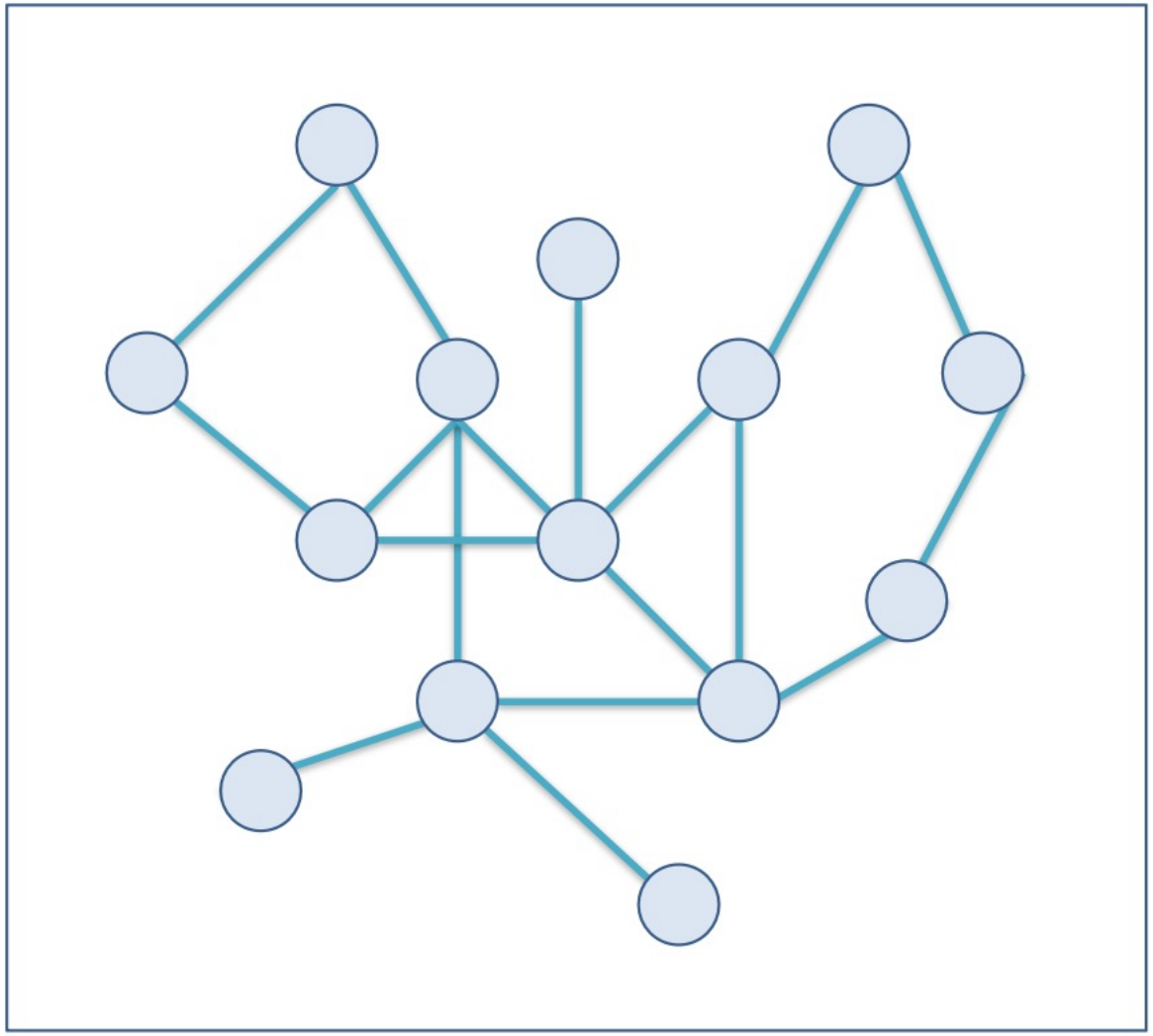
Michal2000

Network (Graph) Terminology

Mathematics Terminology

A graph **G(V,E)** denotes a graph with vertices **V** and edges **E**.

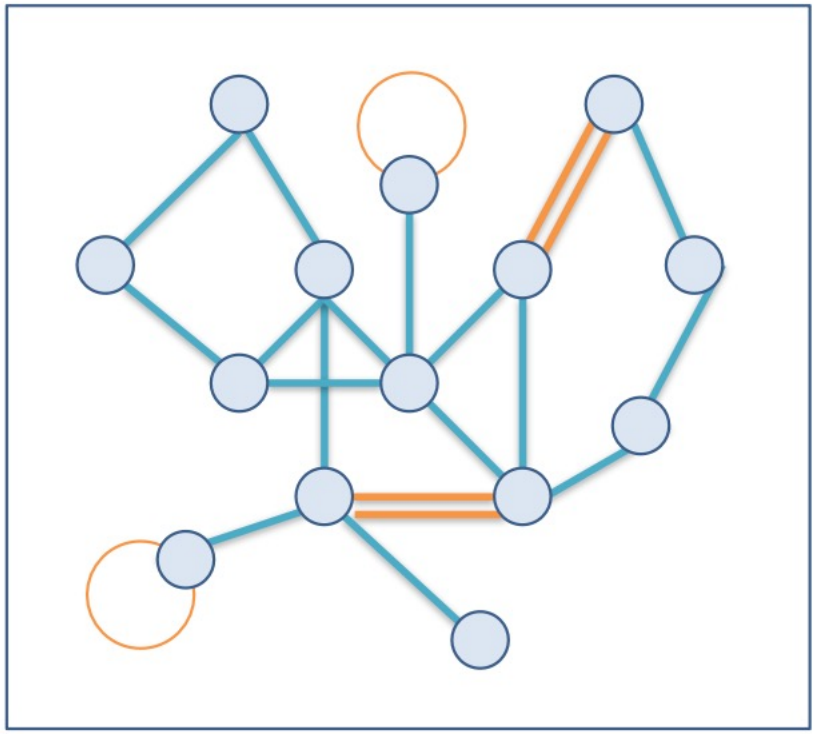
- Vertices and edges are sometimes referred to as **nodes** and **links**.
- Graphs are sometimes referred to as **networks**.



Mathematics Terminology

Properties of a ***simple*** graph:

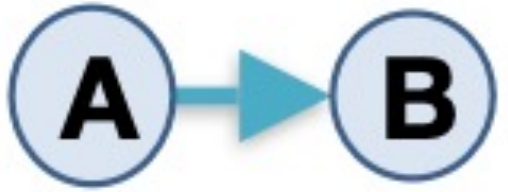
- No self-loops
- No multi-edges

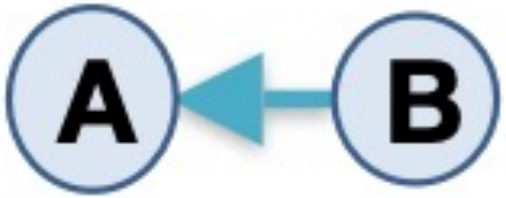


Not a simple graph!
→ A ***general graph***

Mathematics Terminology

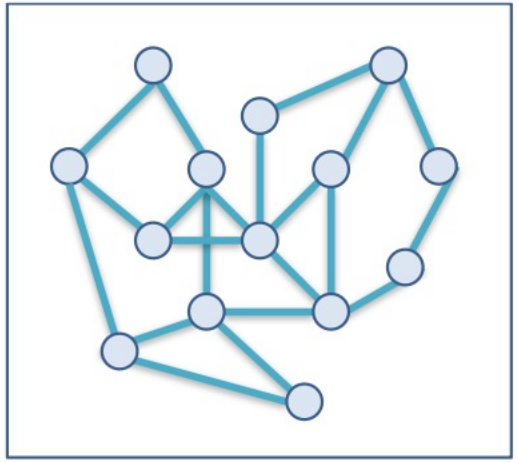
A ***directed*** graph specifies the order of the source and target of each edge.





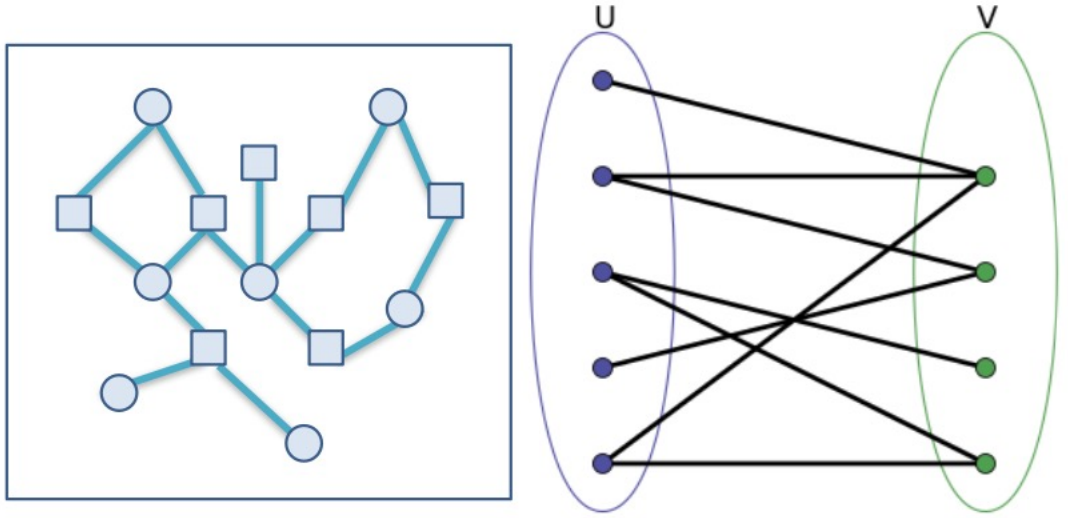
Mathematics Terminology

A ***biconnected*** graph is one that is still connected when you remove any one node.



Biconnected Graph

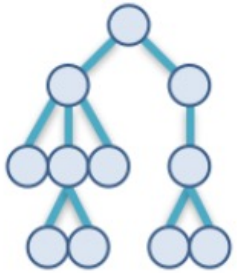
A ***bipartite*** graph can be partitioned into two sets of unconnected vertices.

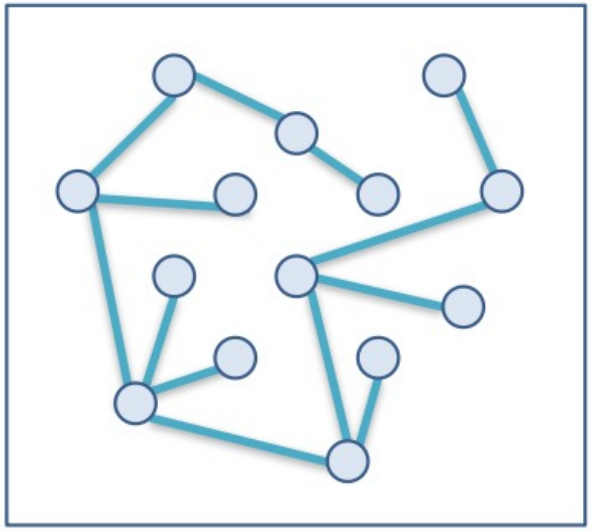


Bipartite Graph

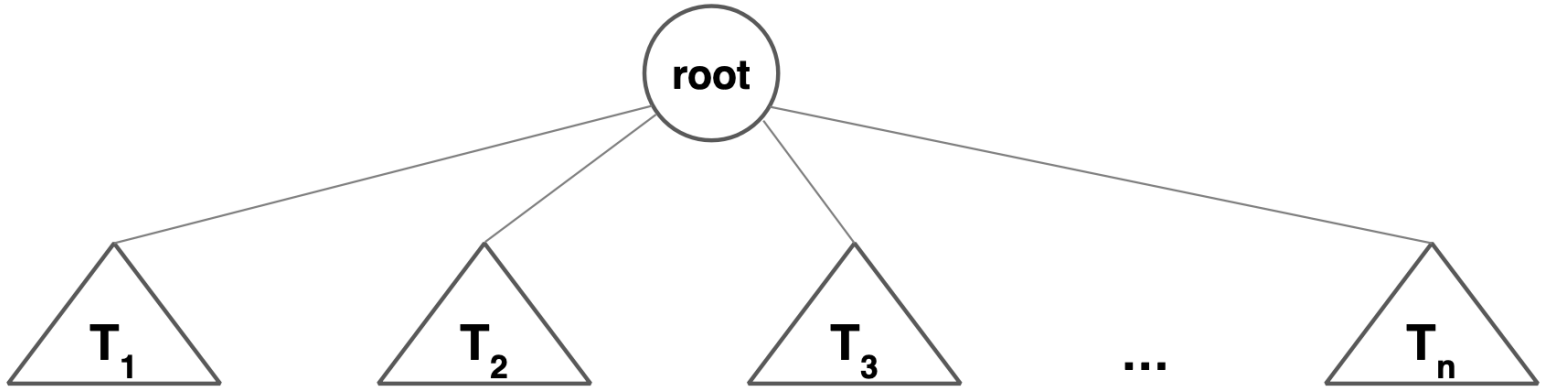
Mathematics Terminology

A ***tree*** is a connected graph without any cycles.



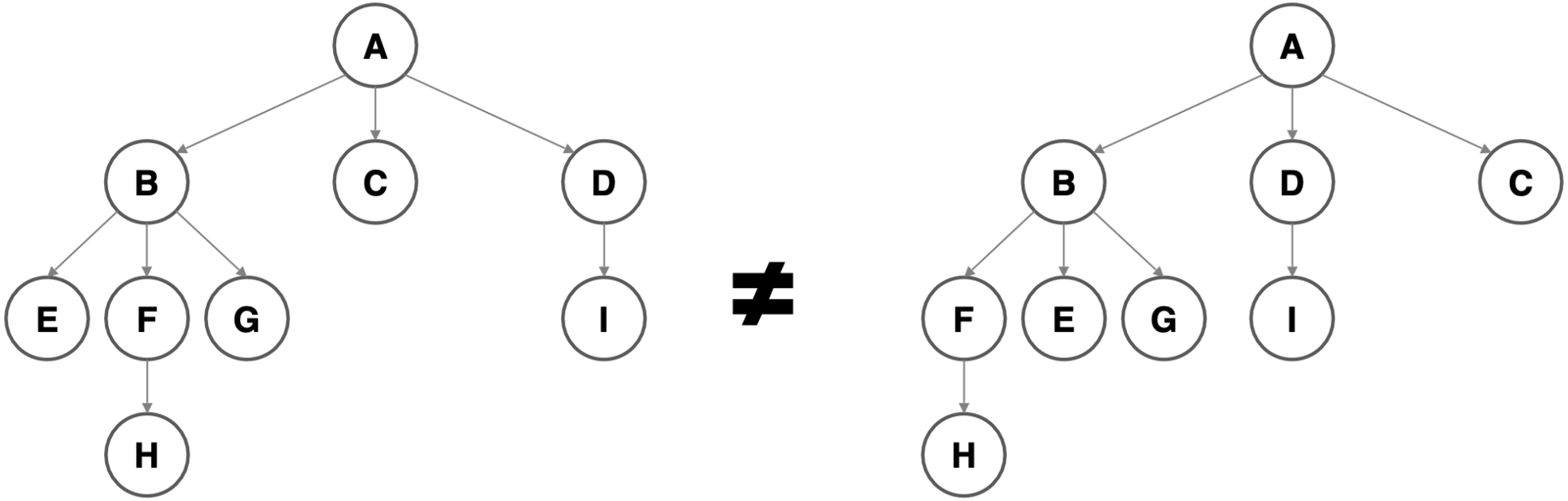


You may choose a single vertex as the ***root*** of a tree. Often it is helpful to consider sub-trees of the the rooted tree.



Mathematics Terminology

An ***ordered tree*** defines an order to nodes coming from the same parent.

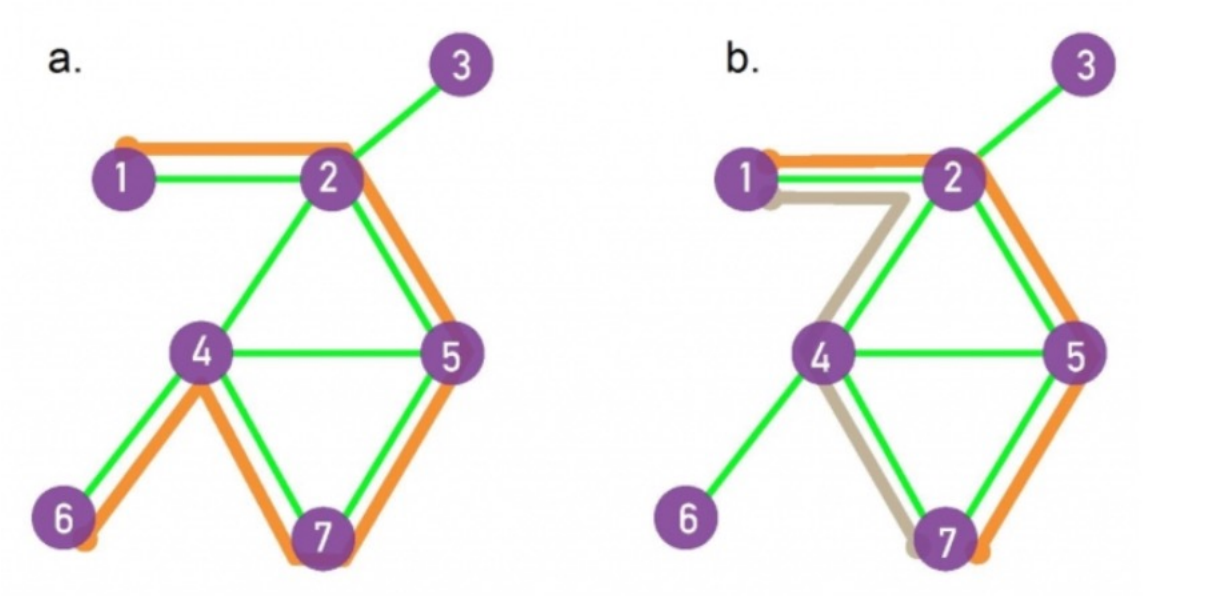


Mathematics Terminology

An ***path*** is an ordered list of unrepeated, connected vertices from one vertex to another.

The ***length*** of the path is its number edges.

This is often used in routing.



A path from 1 to 6

Shortest paths (two) from 1 to 7

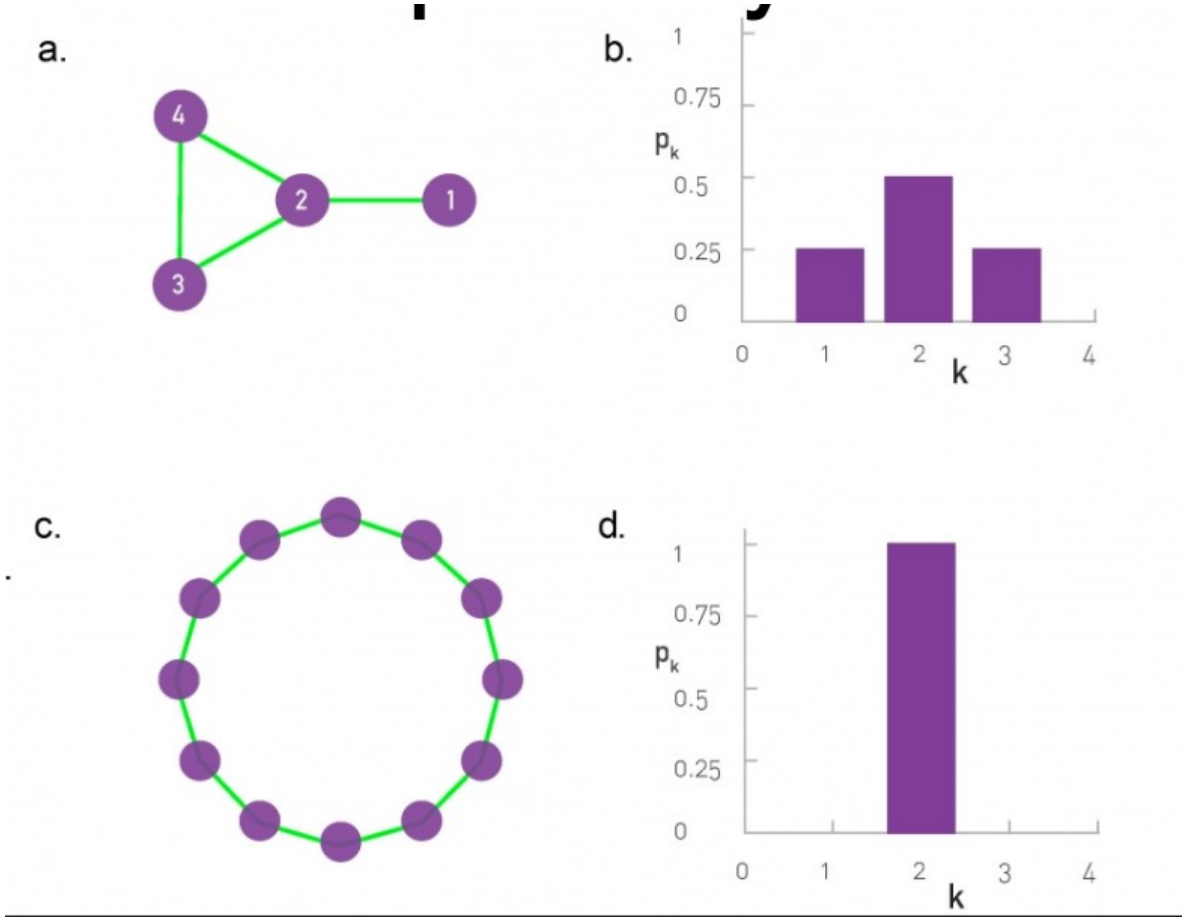
Mathematics Terminology

The ***degree*** of a node is the number of edges connected to it.

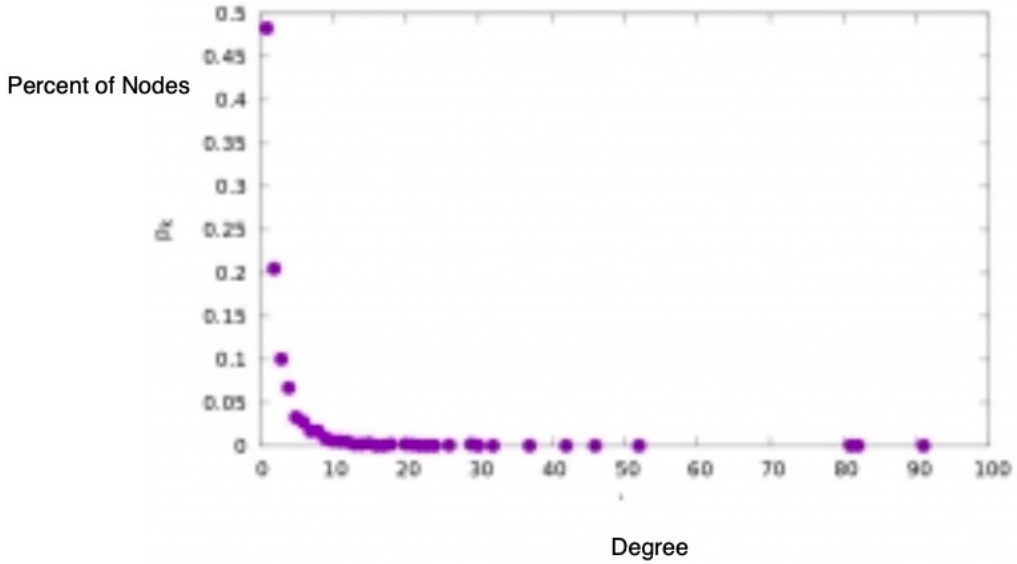
Some analyses involve average degree:

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

Others examine the degree distribution.



Degree distribution example



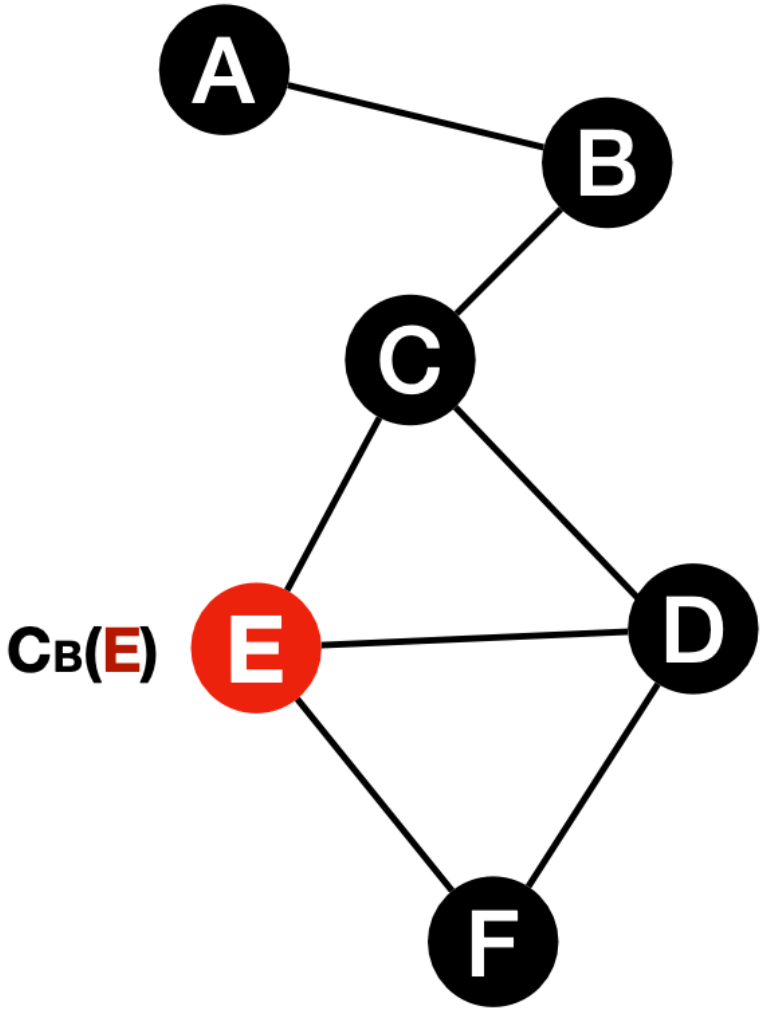
Protein Interaction Network

Betweenness Centrality

$$c_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}$$

$\sigma(s,t|v)$: Number of shortest paths

$\sigma(s,t)$: Total number of paths



Betweenness centrality measures how many shortest paths pass through a given node.

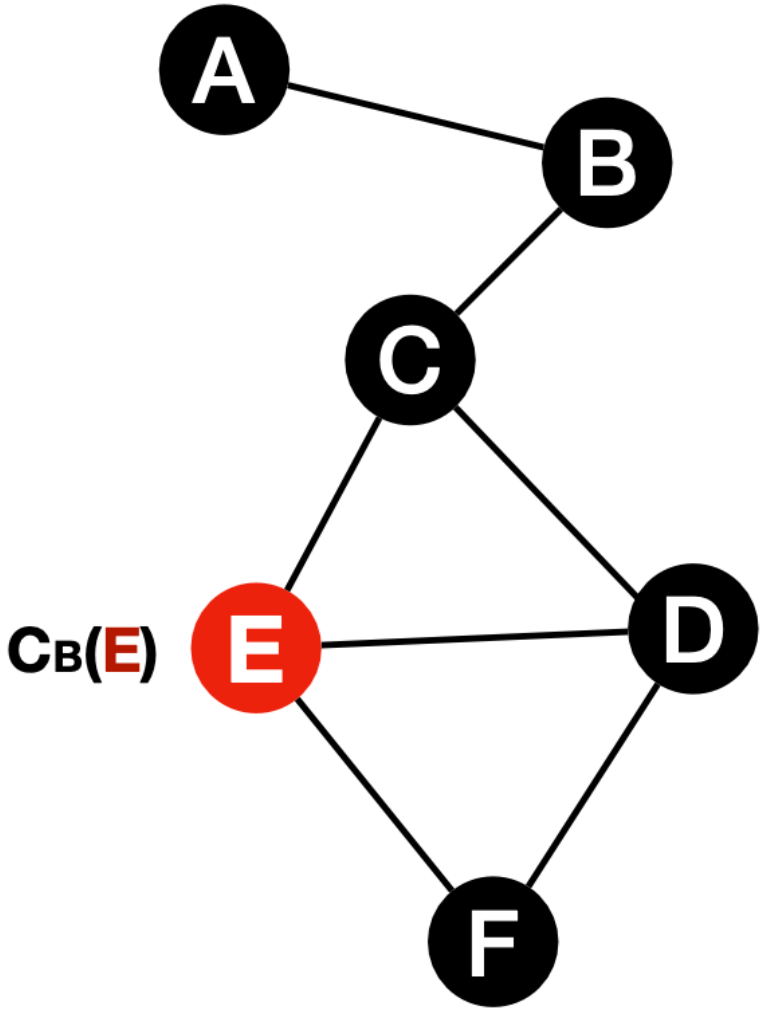
It is often analyzed to determine a node's importance.

Betweenness Centrality

$$c_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}$$

$\sigma(s,t|v)$: Number of shortest paths

$\sigma(s,t)$: Total number of paths



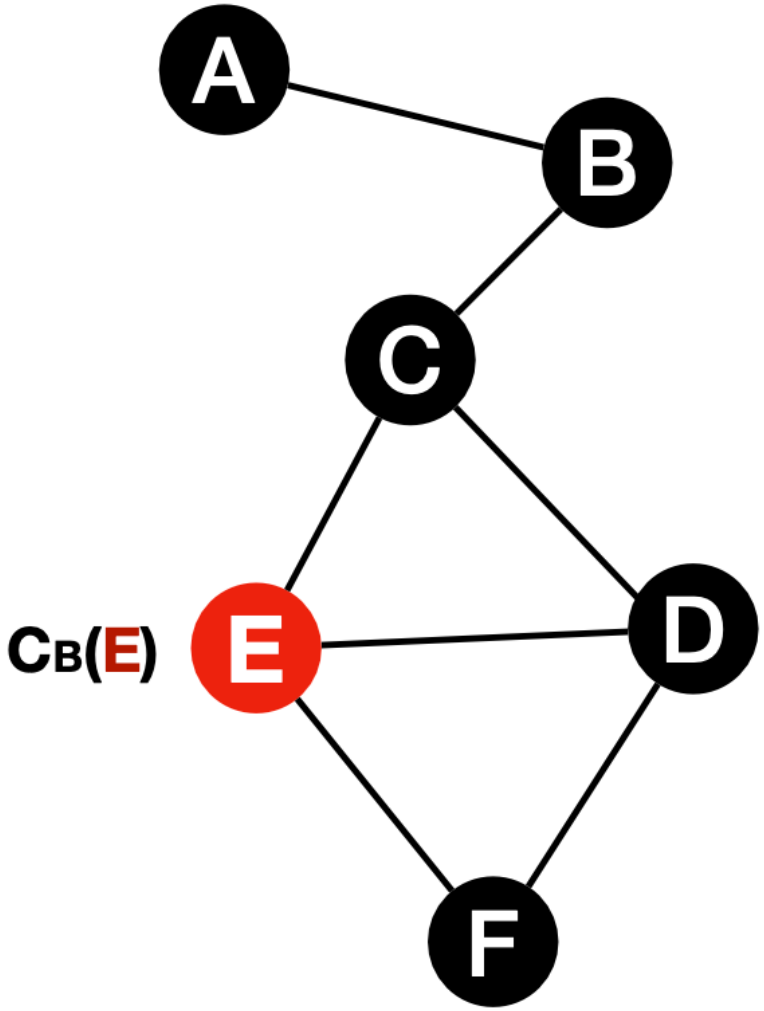
	Shortest Path through E?	Number of shortest paths
A, B	0	1
A, C	0	1
A, D	0	1
A, F	1	2
B, C	0	1
B, D	0	1
B, F	1	2
C, D	0	1
C, F	1	2
D, F	0	1

Betweenness Centrality

$$c_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}$$

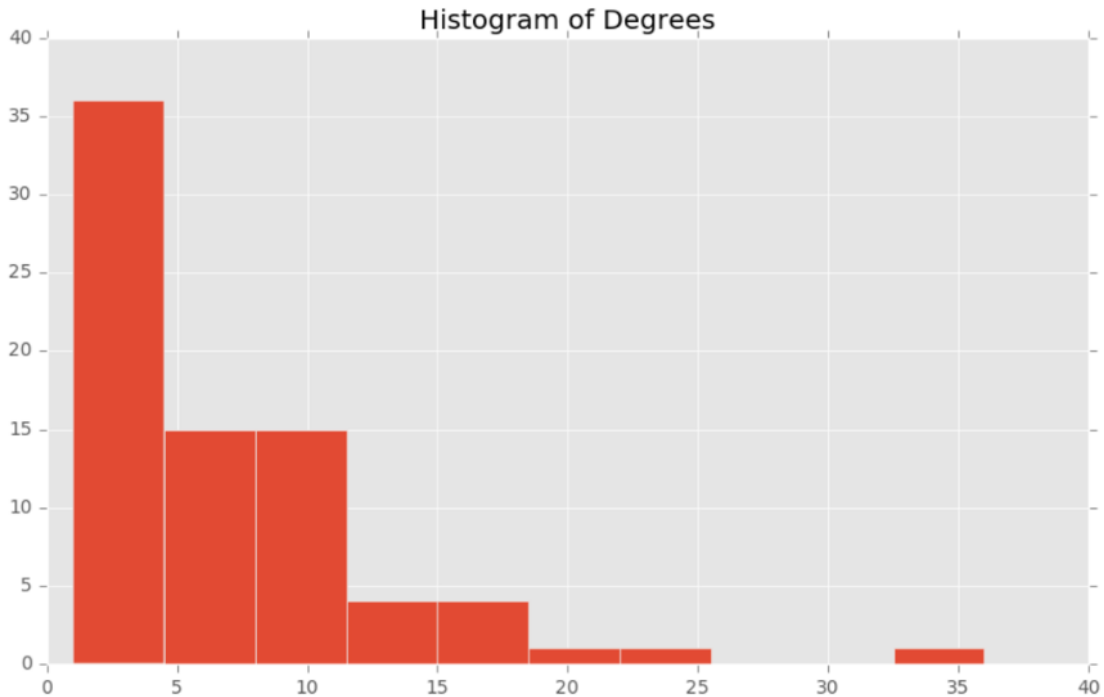
$\sigma(s,t|v)$ → Number of shortest paths

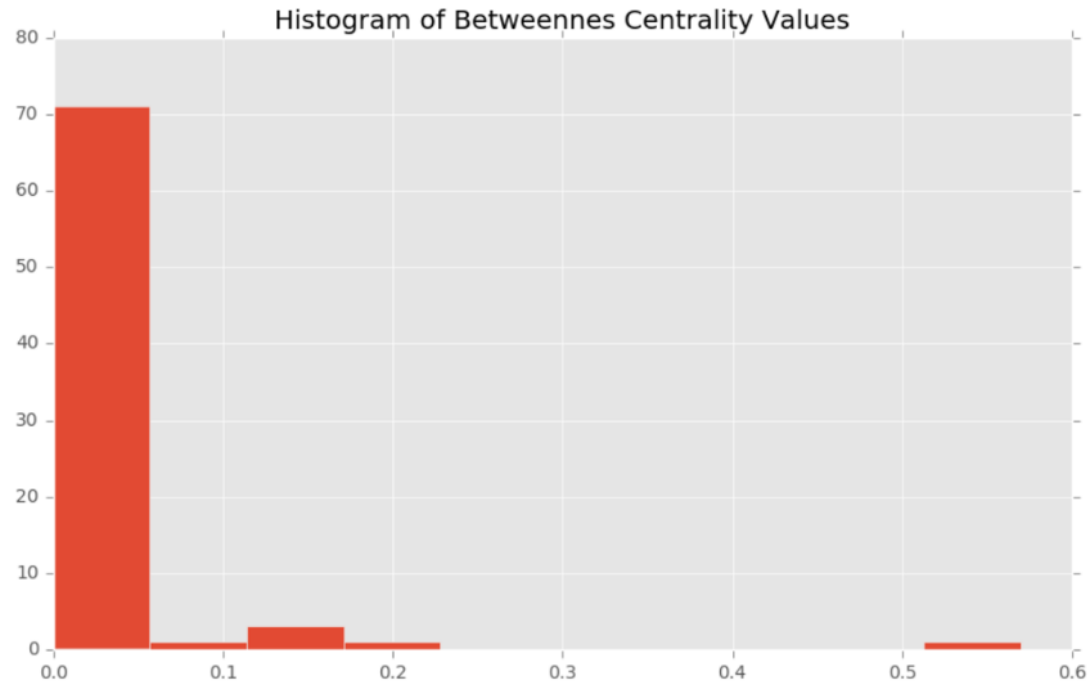
$\sigma(s,t)$ → Total number of paths



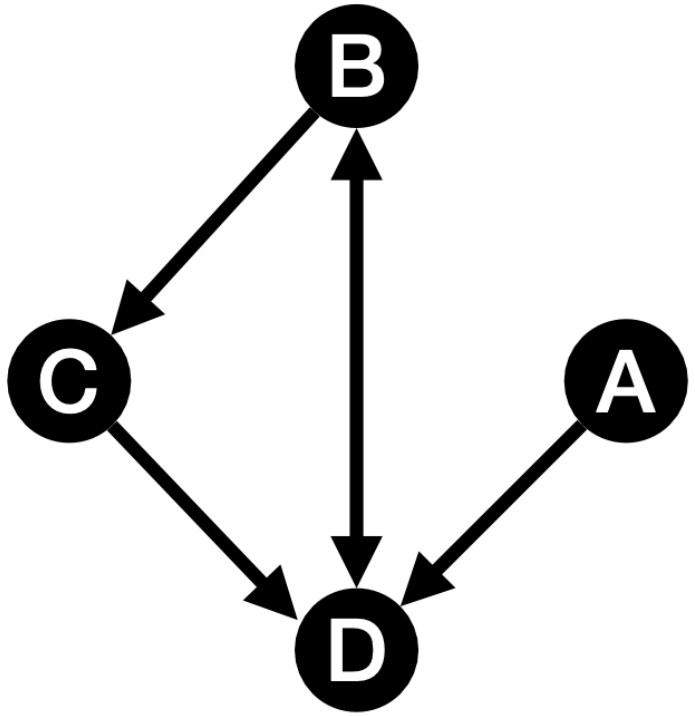
	Shortest Path through E?	Number of shortest paths	
A, B	0	1	
A, C	0	1	
A, D	0	1	
A, F	1	2	0.5
B, C	0	1	
B, D	0	1	
B, F	1	2	0.5
C, D	0	1	
C, F	1	2	
D, F	0	1	0.5

= 1.5


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


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

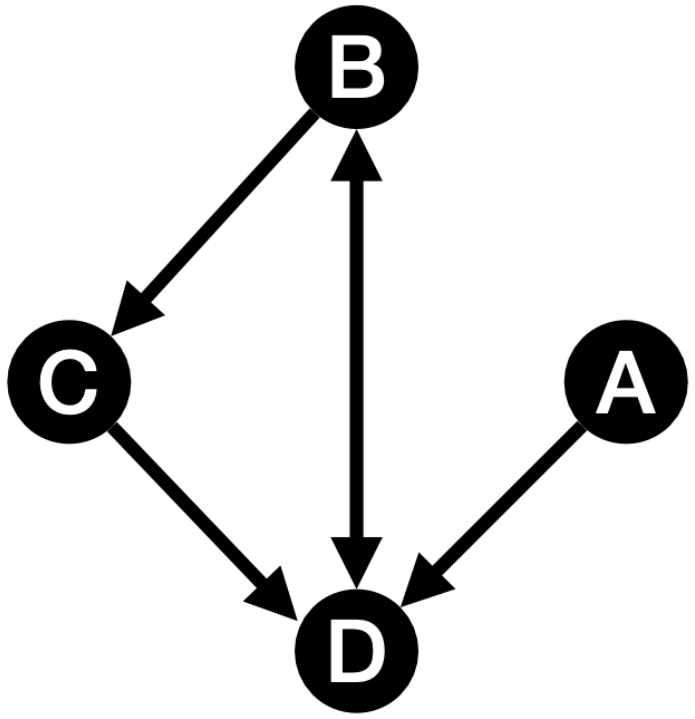
PageRank



PageRank is the algorithm that gave Google early dominance in search engines in the late 1990s.

Like Betweeness Centrality, it calculates the importance of nodes based on their links.

Page Rank



…and continues until settled…

A	0.25	0	0	0
B	0.25	0 + 0.25	0 + 0.625	0 + 0.1875
C	0.25	0 + 0.125	0 + 0.125	0 + 0.3125
D	0.25	0 + 0.25 + 0.125 + 0.25	0 + + 0.125 + 0.0625	0 + + 0.3125 + 0.0625

Exploratory (Visual) Analysis of Networks

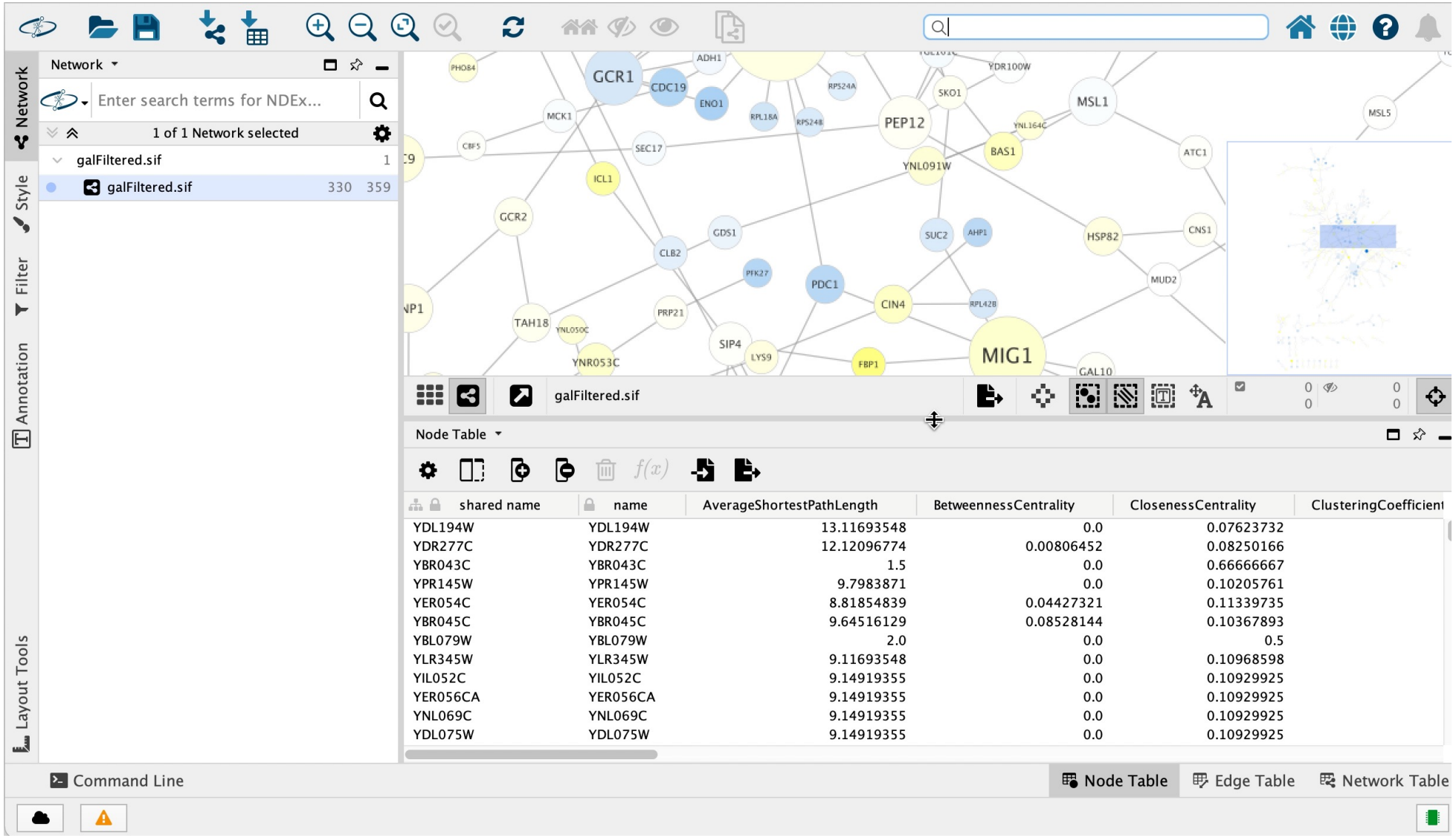
A good place to start for network-centric analysis and visualization is a graph drawing framework.

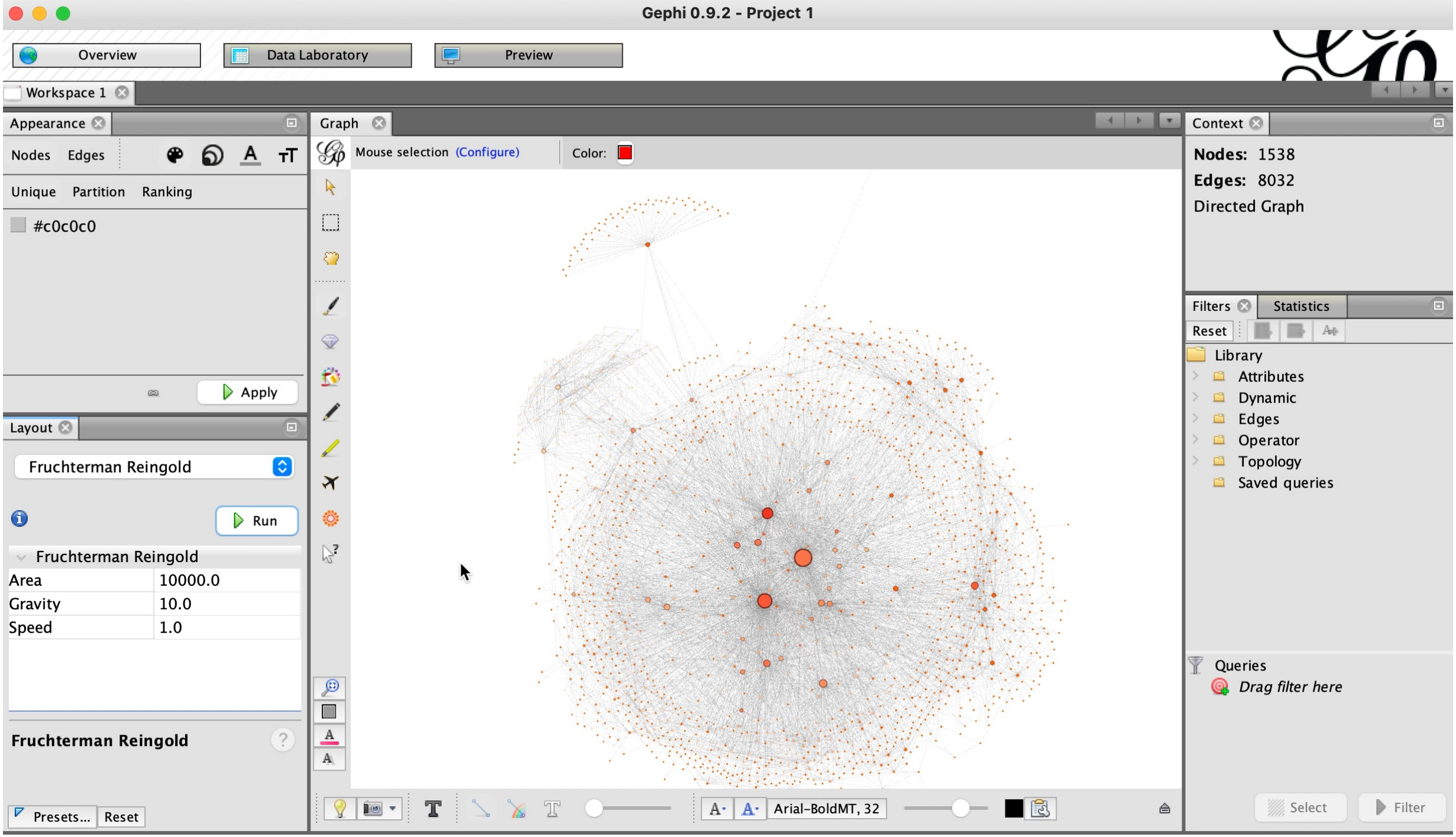
This slide:

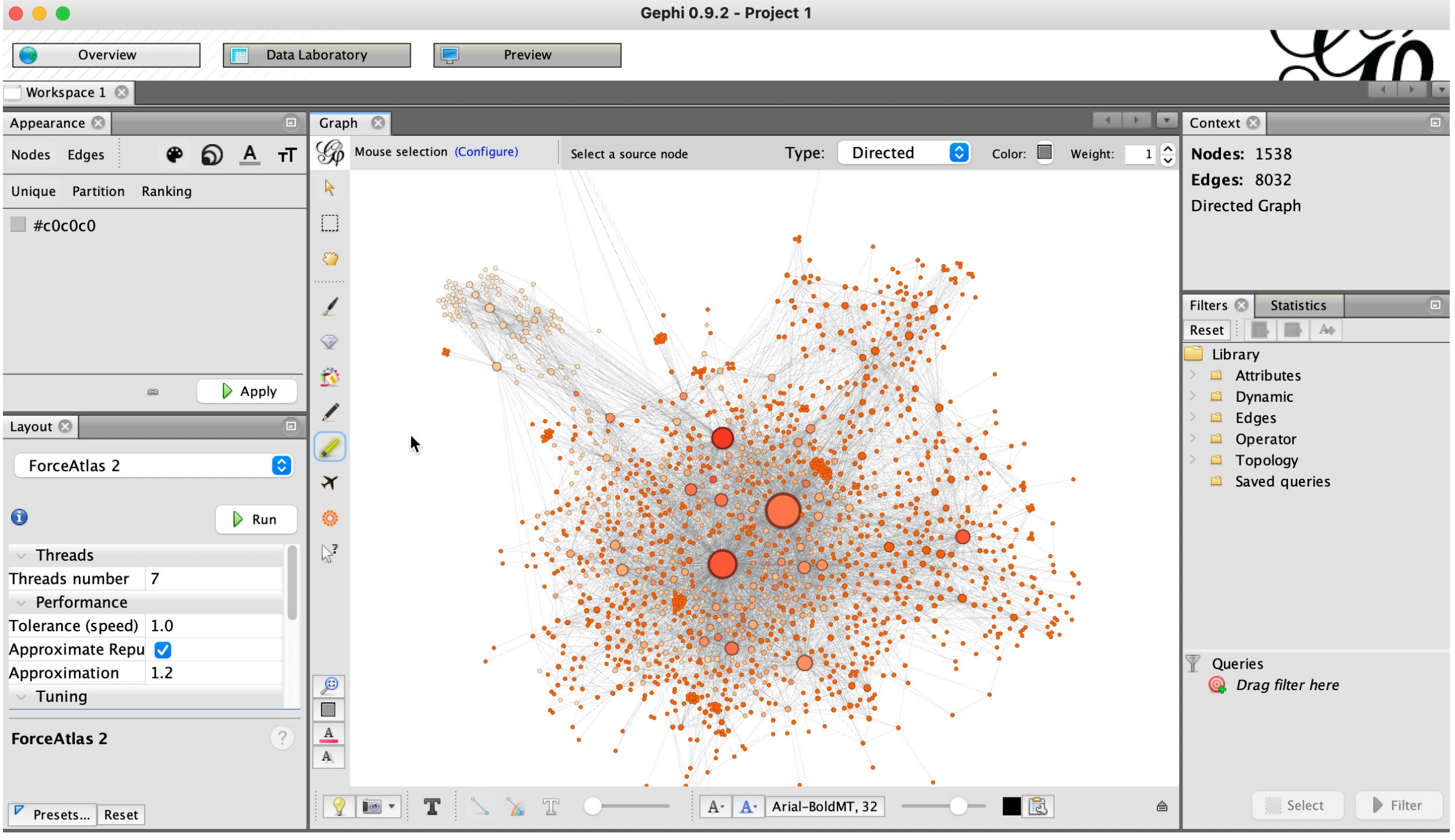
- Cytoscape

Next slide:

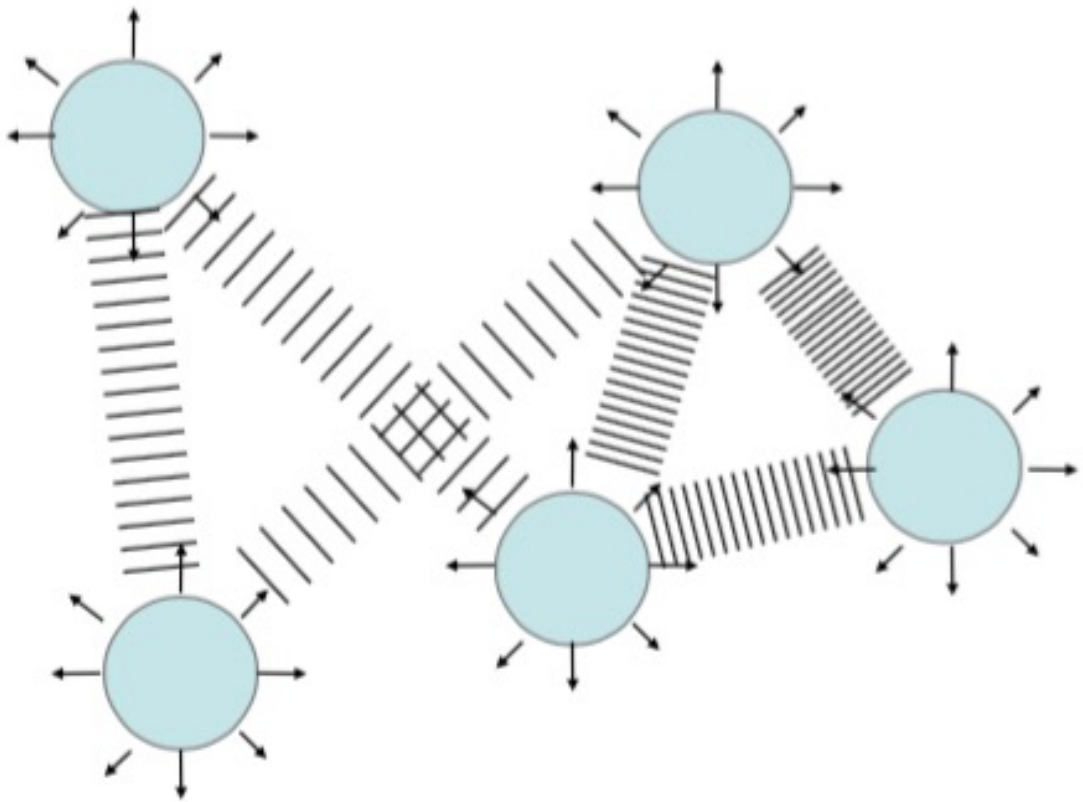
- Gephi




Gephi 0.9.2 - Project 1
Overview
Data Laboratory
Preview
Workspace 1
Appearance
Nodes
Edges
Unique
Partition
Ranking
#c0c0c0
Apply
Layout
Fruchterman Reingold
Run
Fruchterman Reingold
Area
10000.0
Gravity
10.0
Speed
1.0
Fruchterman Reingold
Presets...
Reset
Graph
Mouse selection (Configure)
Color:
Arial-BoldMT, 32
Context
Nodes: 1538
Edges: 8032
Directed Graph
Filters
Statistics
Reset
Library
Attributes
Dynamic
Edges
Operator
Topology
Saved queries
Queries
Drag filter here
Select
Filter


Gephi 0.9.2 - Project 1
Overview
Data Laboratory
Preview
Workspace 1
Appearance
Nodes
Edges
Unique
Partition
Ranking
#c0c0c0
Apply
Layout
ForceAtlas 2
Run
Threads
Threads number
7
Performance
Tolerance (speed)
1.0
Approximate Repu
Approximation
1.2
Tuning
ForceAtlas 2
Presets...
Reset
Graph
Mouse selection (Configure)
Select a source node
Type:
Directed
Color:
Weight:
1
Arial-BoldMT, 32
Context
Nodes: 1538
Edges: 8032
Directed Graph
Filters
Statistics
Reset
Library
Attributes
Dynamic
Edges
Operator
Topology
Saved queries
Queries
Drag filter here
Select
Filter

Force-directed layouts model networks as a physical system where nodes repel but edges attract



These layouts “work” on a wide variety of graphs.

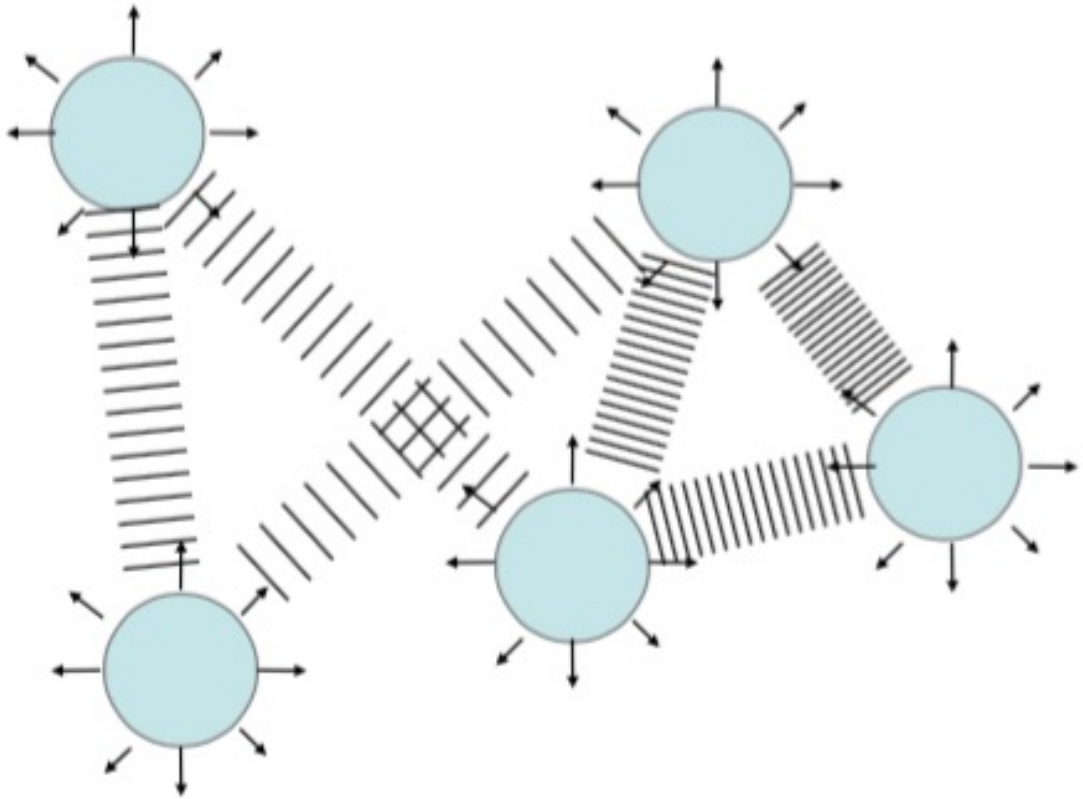
They also scale better than other approaches.

Issues:

- Lots of tweakable settings
- Are unstable - hay not be repeatable

Image from Josh Levine & Miriah Meyer, slides

Force-directed layout animation for Les Miserable graph



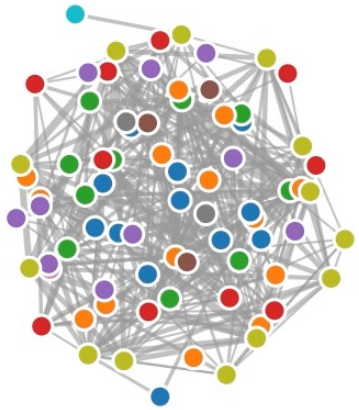


Image from Josh Levine & Miriah Meyer, slides

https://beta.observablehq.com/@mbostock/d3-force-directed-graph

If you know more about your network, another layout may perform “better.”

Hierarchical networks assume data has a layered, ordered, or otherwise hierarchical structure.

Examples:

- Workflows
- Computing
- Geneaology

Well-known hierarchical layout:

- `dot` in GraphViz (pygraphviz)

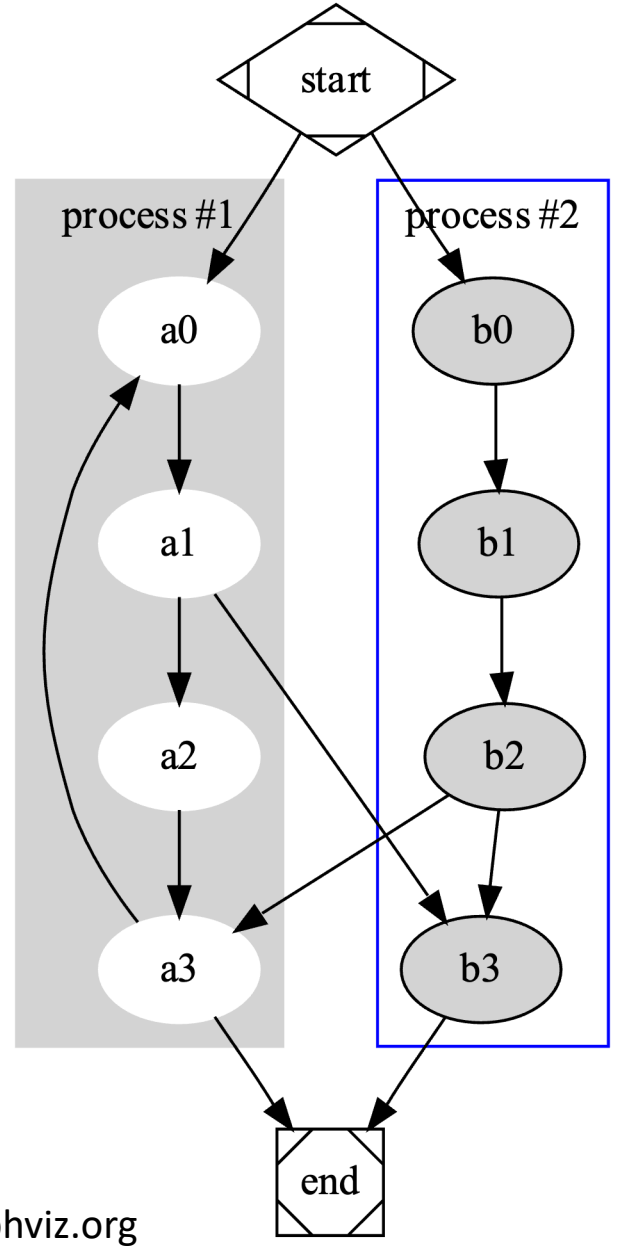


Image from graphviz.org

How to cite this site?

Check out other surveys!

treevis.net - A Visual Bibliography of Tree Visualization 2.0 by Hans-Jörg Schulz

v.18-NOV-2022

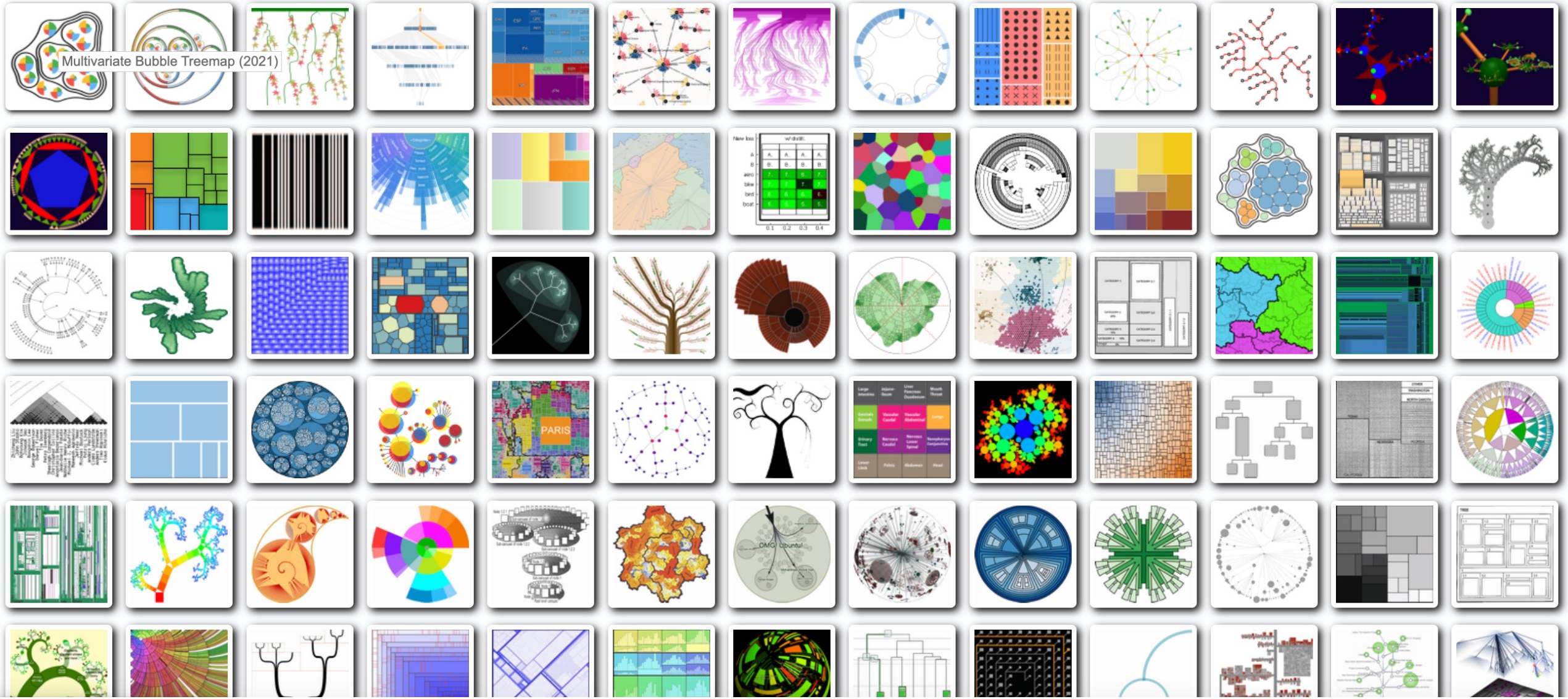
Dimensionality: All

Representation: All

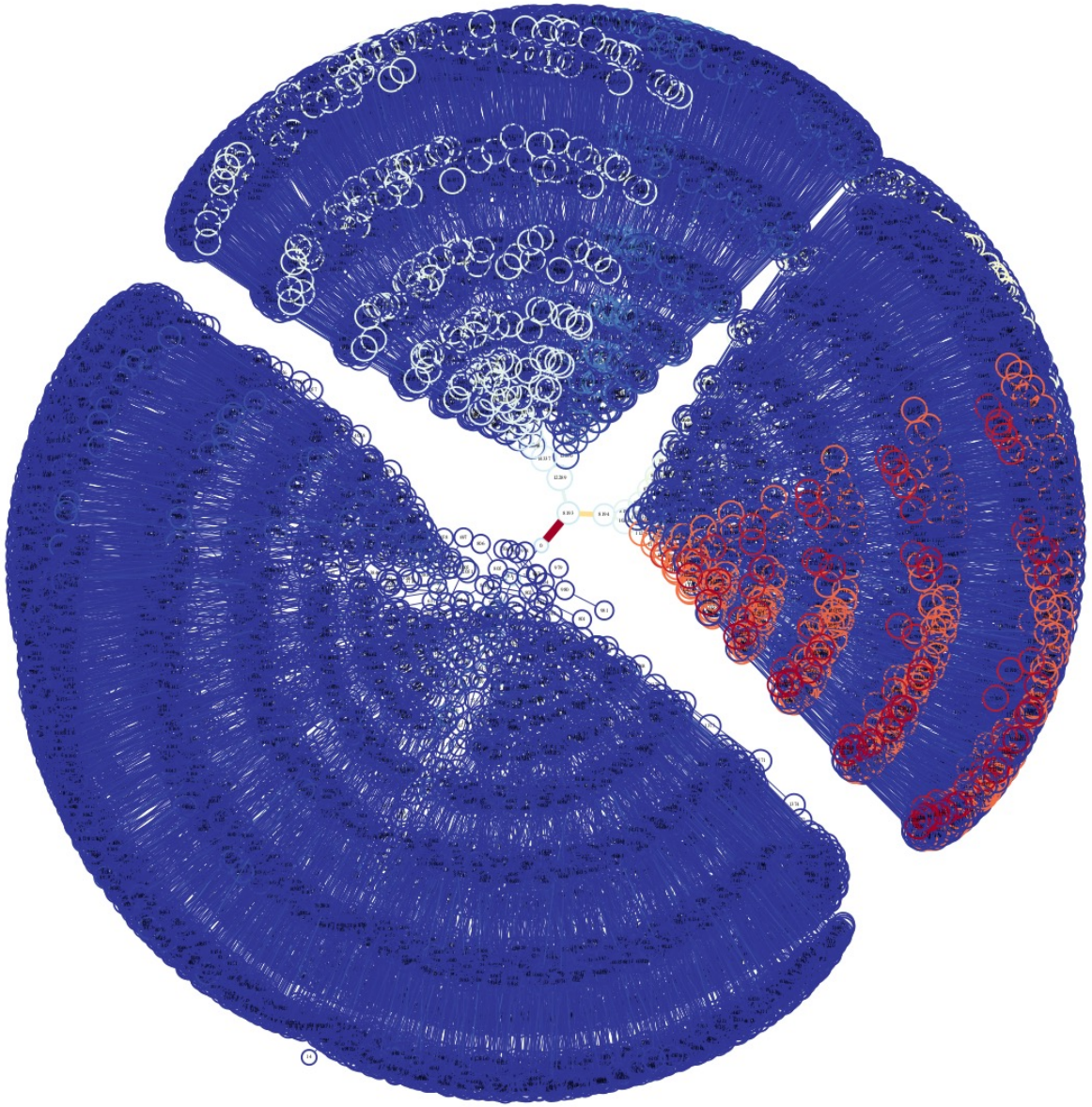
Alignment: All

Fulltext Search

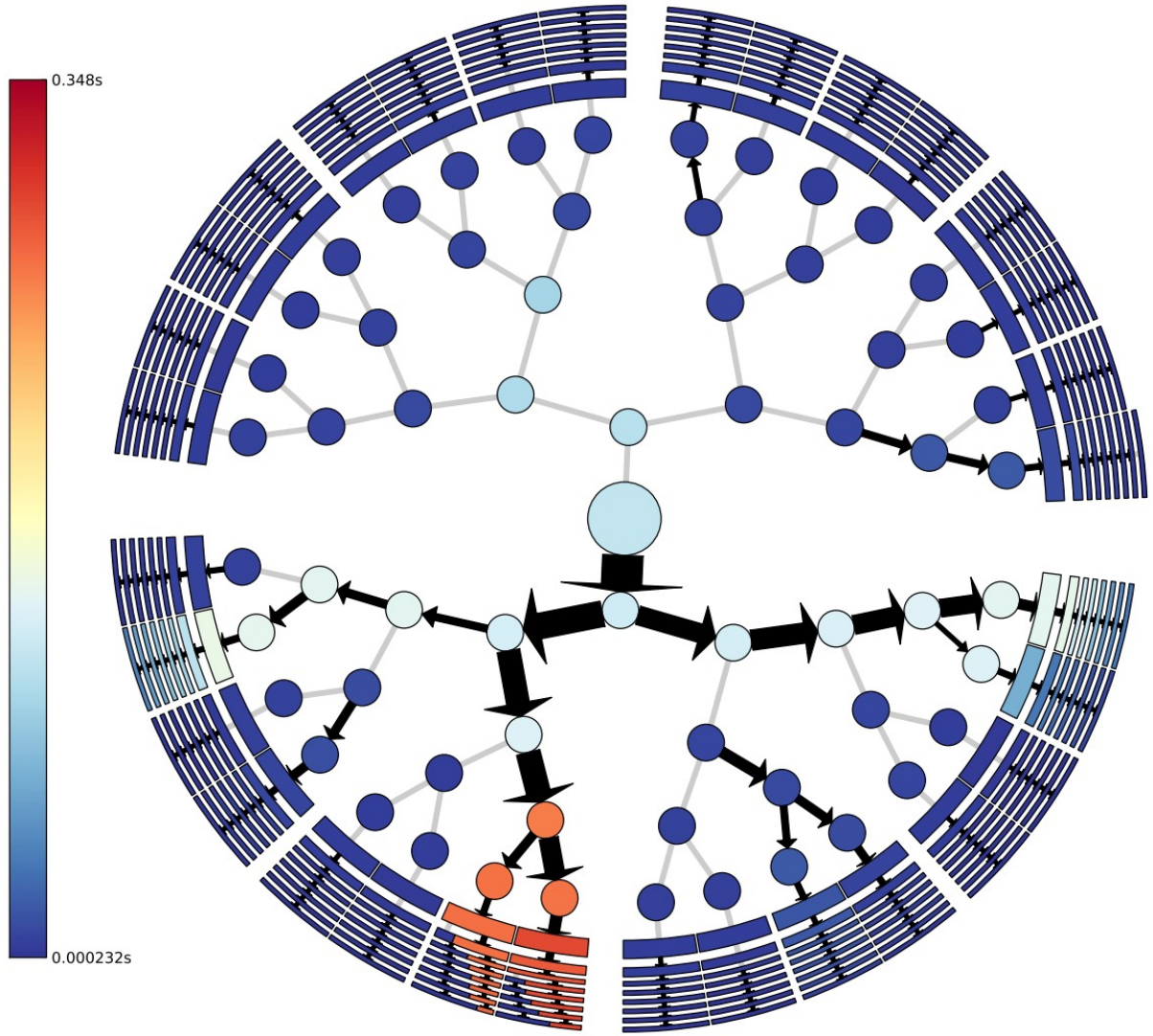
Techniques Shown: 333



When networks get large, aggregation can help.

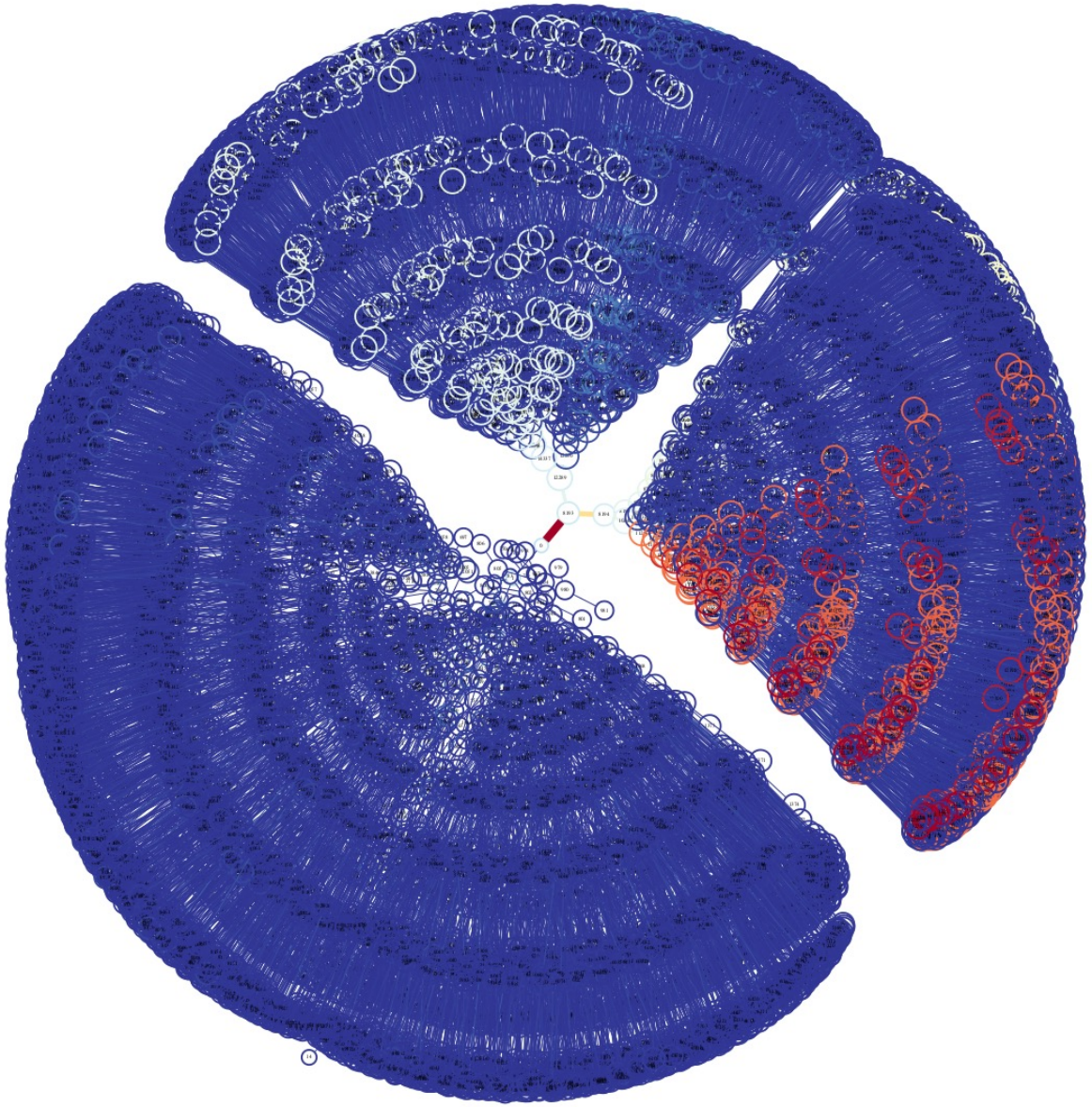


16,384 nodes, force-directed, GraphViz

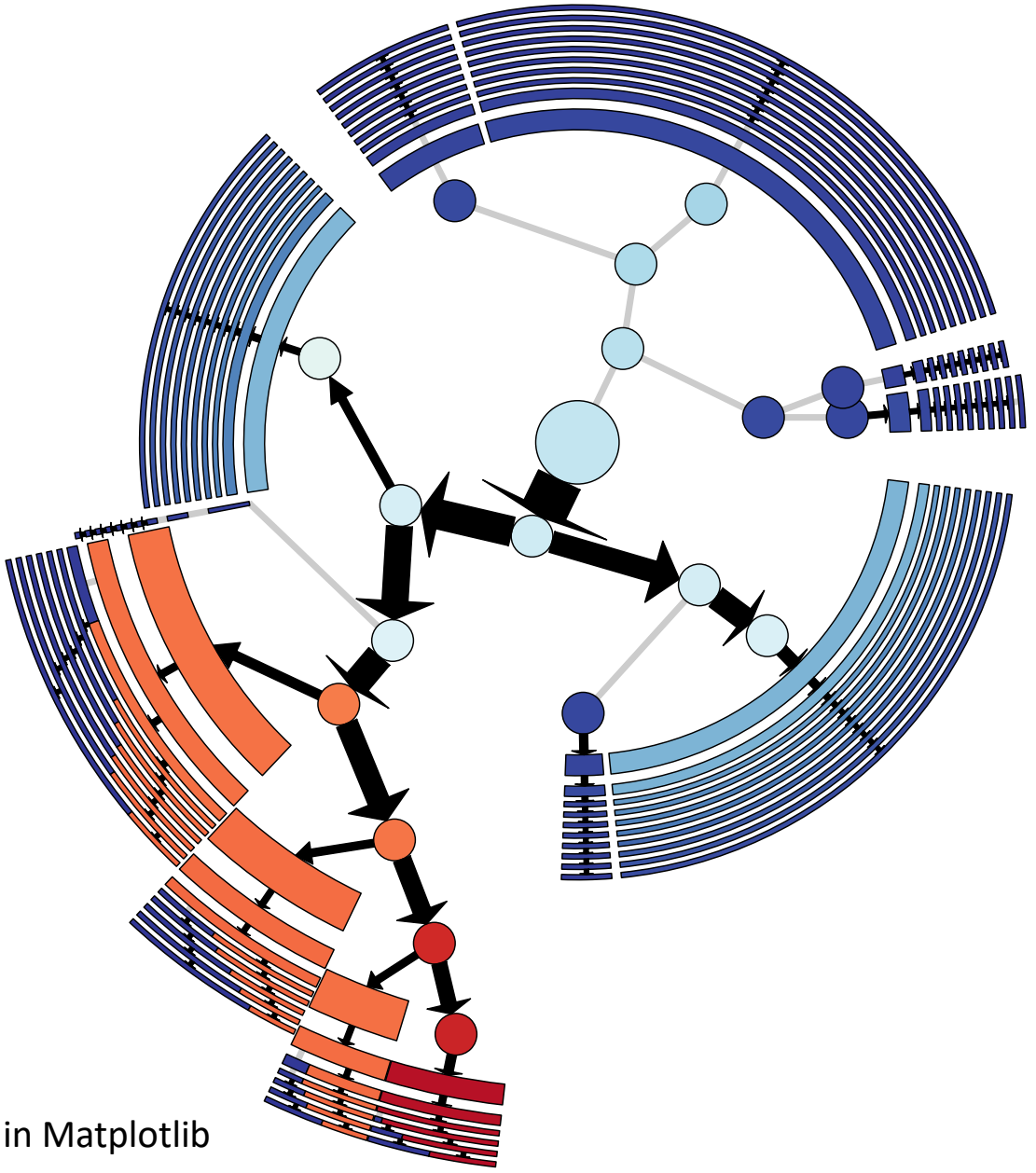


Drawn in Matplotlib

When networks get large, aggregation can help.



16,384 nodes, force-directed, GraphViz



Drawn in Matplotlib

Home Page - Select or cre | Workflow Example - Jupy

localhost:8888/notebooks/Workflow Example.ipynb#

jupyter Workflow Example Last Checkpoint: 16 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [1]:
```
from IPython.display import HTML, display
import hatchet as ht


display(HTML("<style>.container { width:60% !important; }</style>"))
%load_ext hatchet.vis.loader
```

Identifying Optimization Targets

In [2]:
```
gf_64 = ht.GraphFrame.from_hpctoolkit('cct-vis-eval/datasets/kripke-scaling/hpctoolkit-kripke-64-cores/')
gf_128 = ht.GraphFrame.from_hpctoolkit('cct-vis-eval/datasets/kripke-scaling/hpctoolkit-kripke-128-cores/')
```

In [3]:
```
%cct gf_64
```

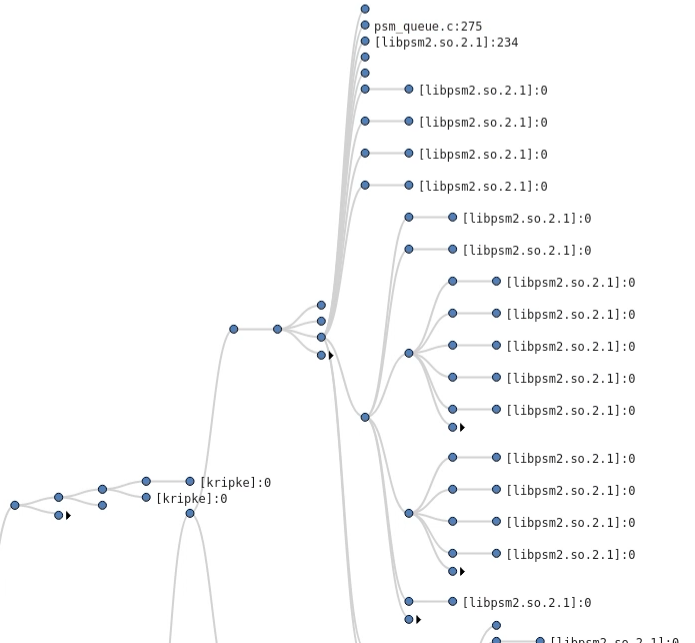
Metrics | Display | Query | Interactive Calling Context Tree

Legend for metric: time

64.32M - 77.19M
51.46M - 64.32M
38.59M - 51.46M
25.73M - 38.59M
12.86M - 25.73M
0.00 - 12.86M

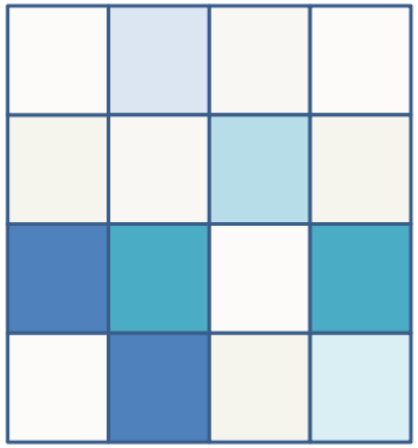
Legend for metric: time (inc)

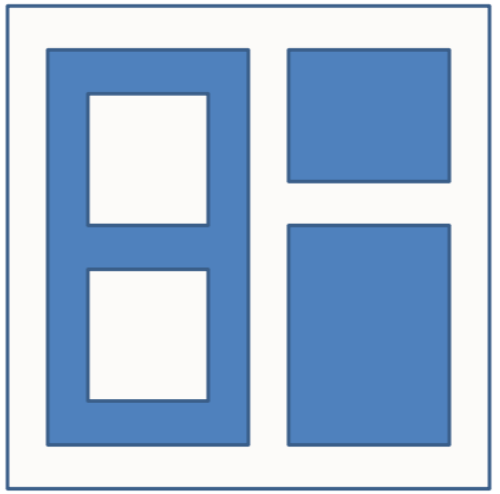
159.89M - 191.87M
127.91M - 159.89M
95.94M - 127.91M
63.96M - 95.94M
31.98M - 63.96M
0.00 - 31.98M



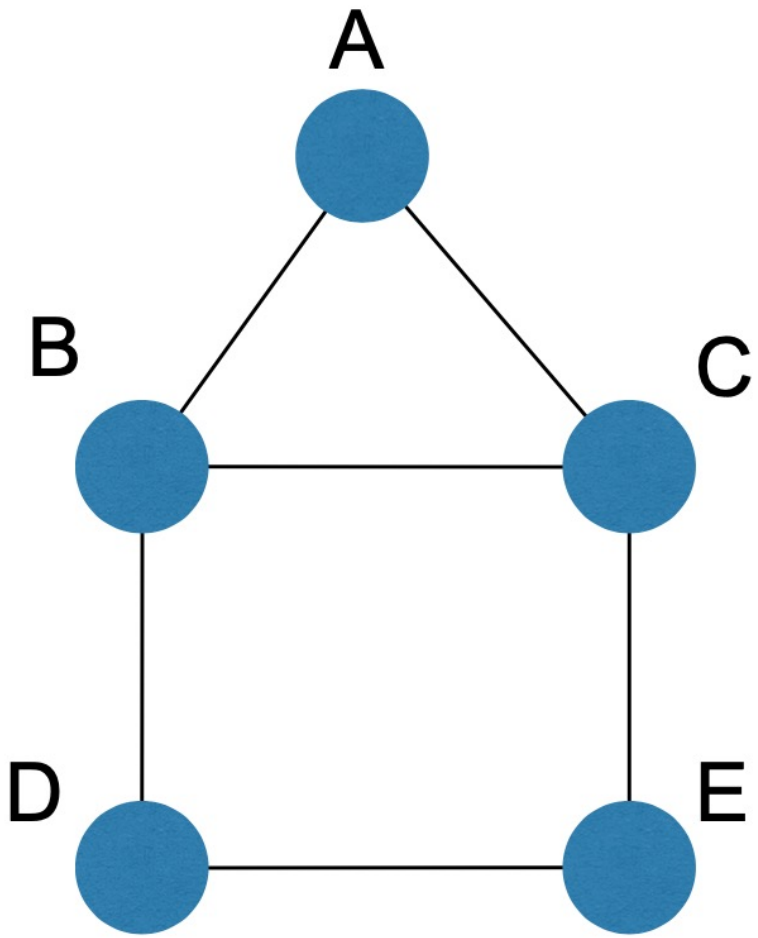
Eliding or filtering can help as well.

Networks don’t have to be drawn as node-link diagrams.



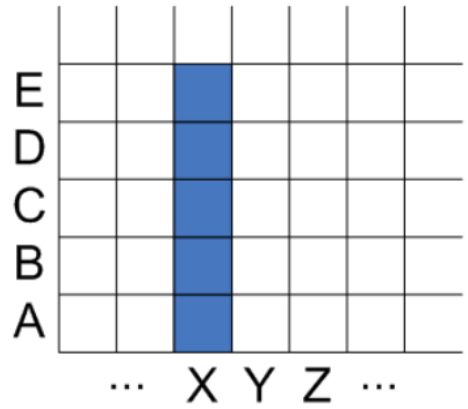


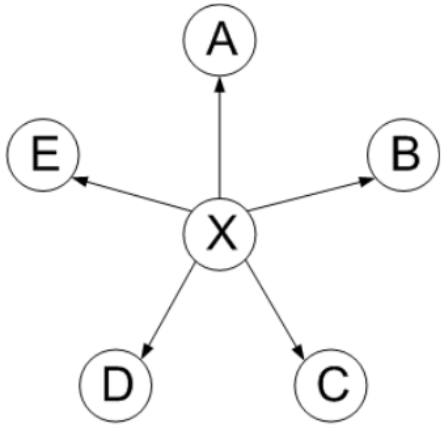
Graphs can be represented as adjacency matrices.

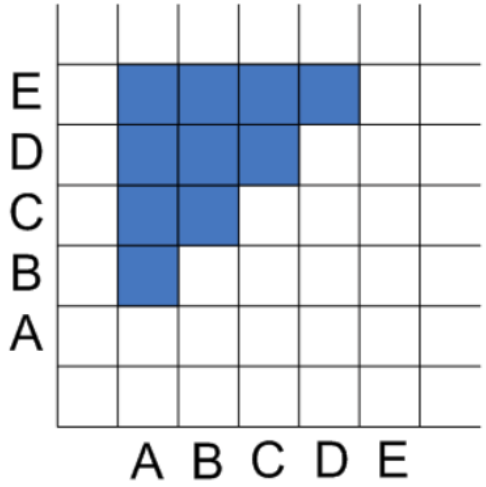


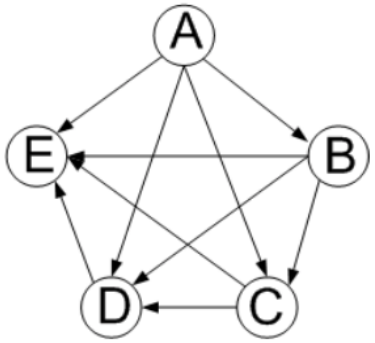
	A	B	C	D	E
A		■	■		
B	■		■	■	
C	■	■			■
D		■			■
E			■	■	

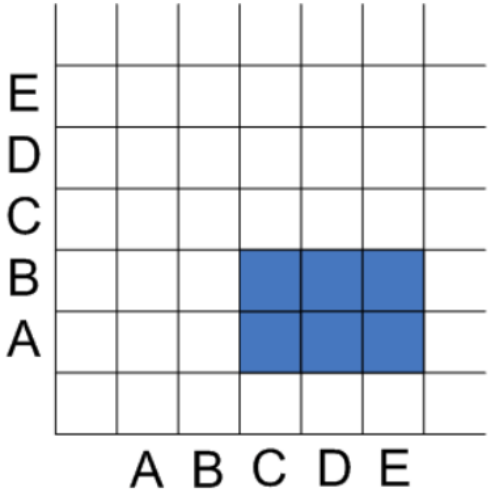
Patterns in directed adjacency matrices

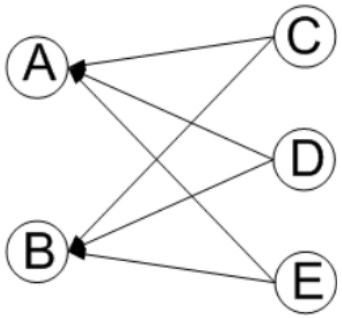


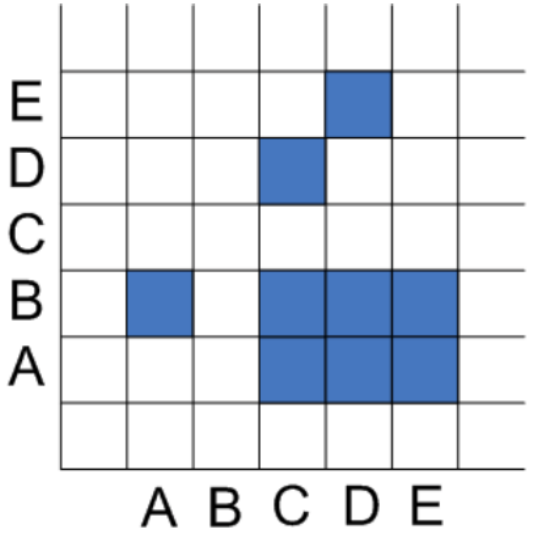


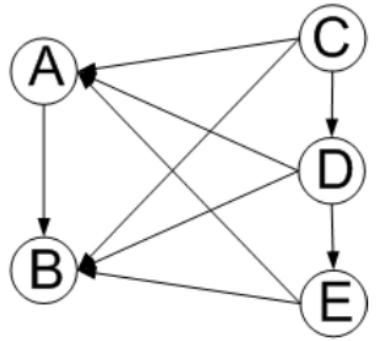






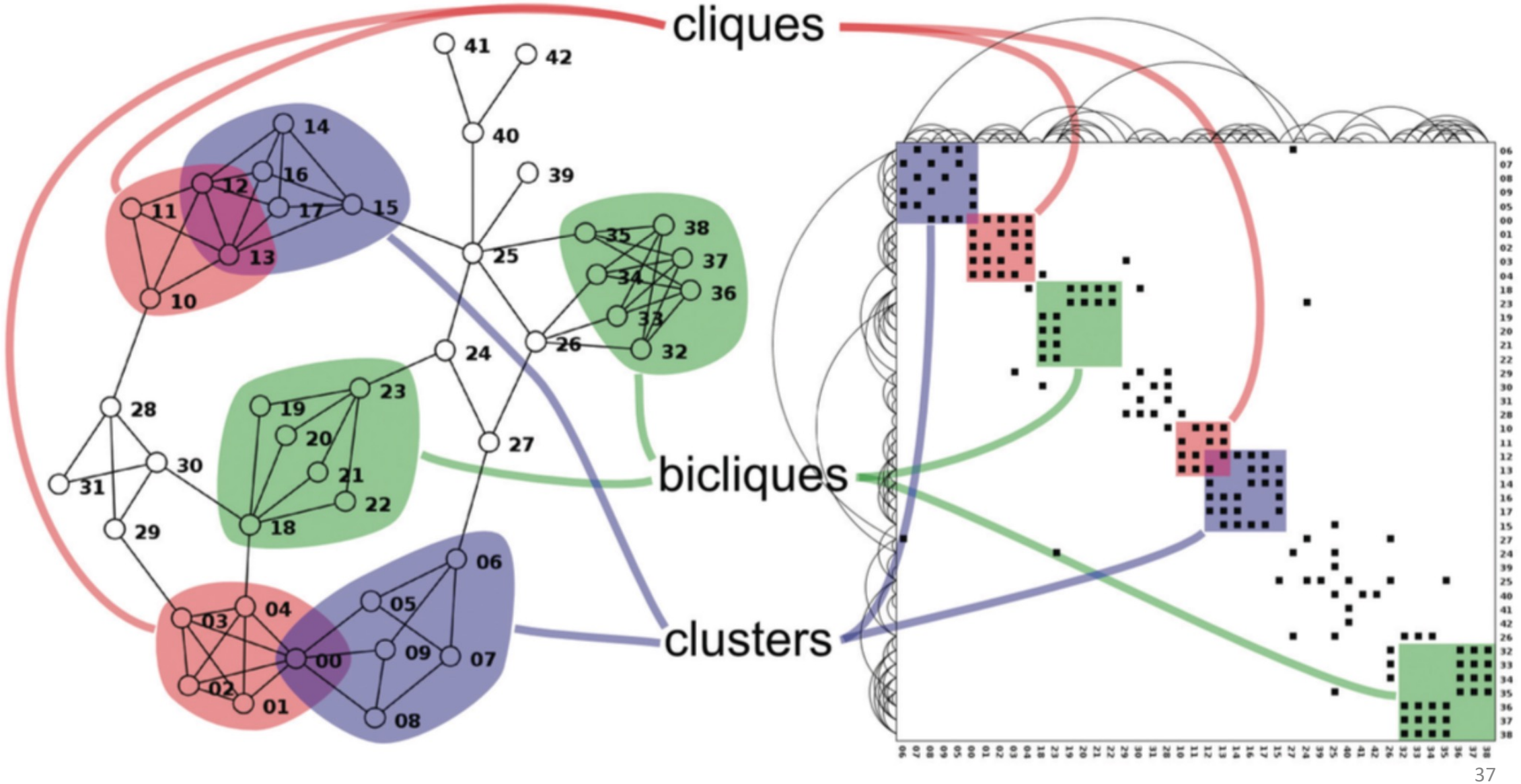






HJ Schulz 2007

Patterns in directed adjacency matrices



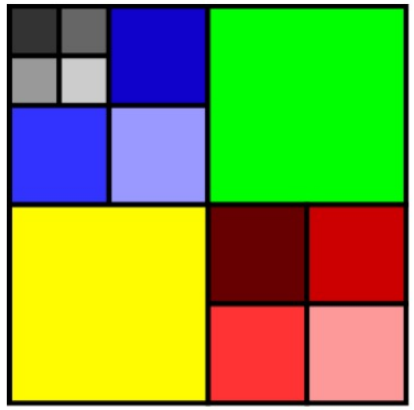
McGuffin 2012

But patterns are only revealed with order…

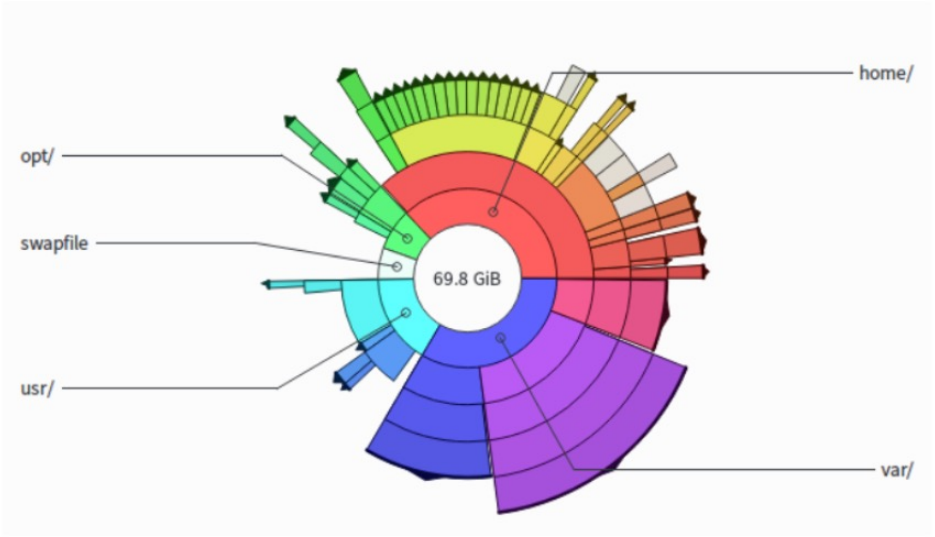
Les Miserable graph

Trees have additional options

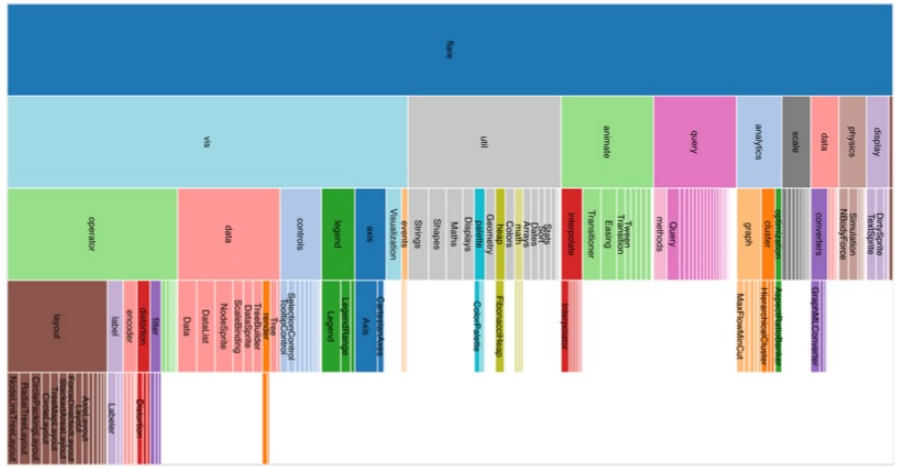
Treemap



Sunburst



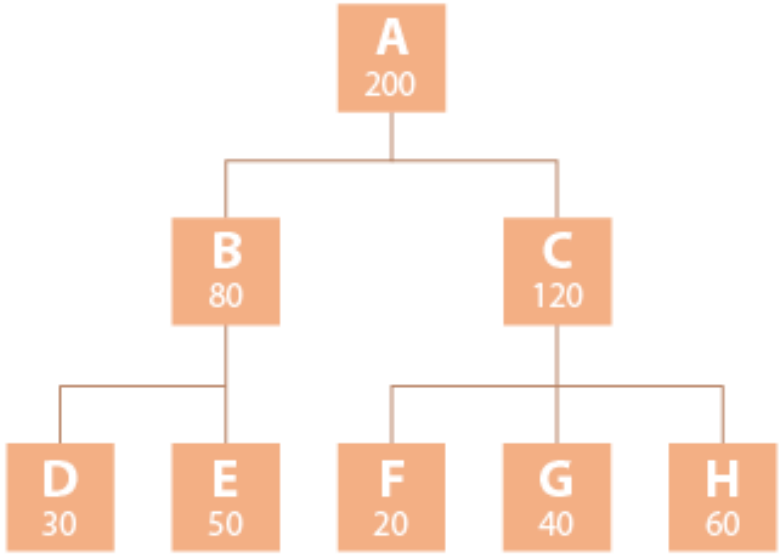
Icicle Plot



Treemaps

Recursively divide space based on some size metric of the nodes.

- What metrics?
  - Size of subtree
  - Attribute of node
- How to divide space?
  - Cluster
  - Pivot
  - Slice & Dice
  - Squarified
  - Strip
  - More…



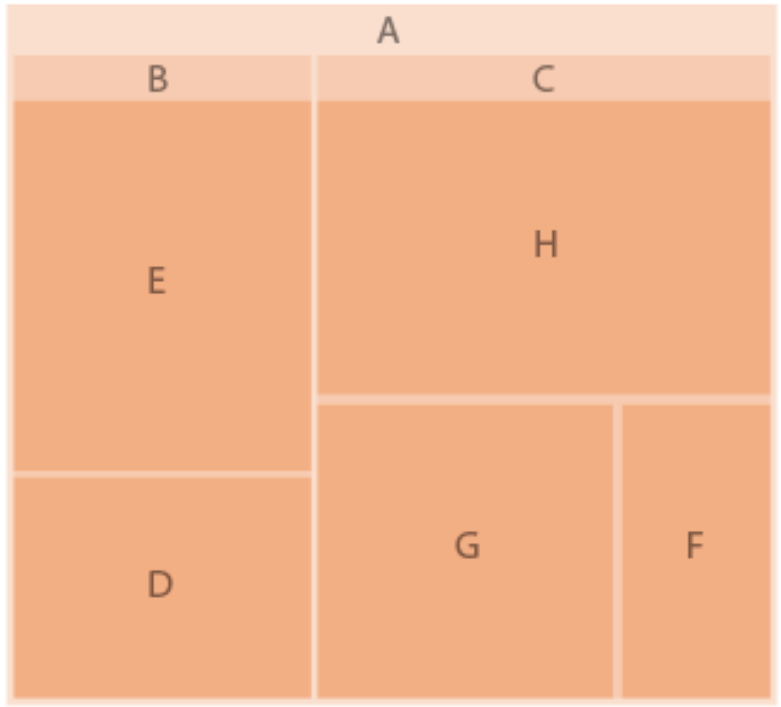
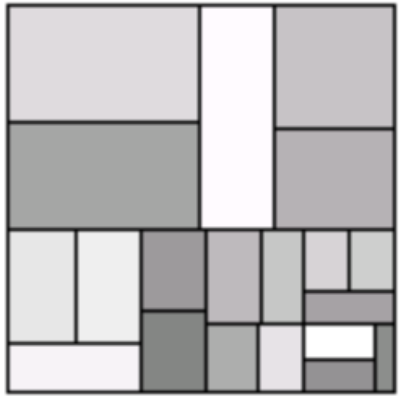
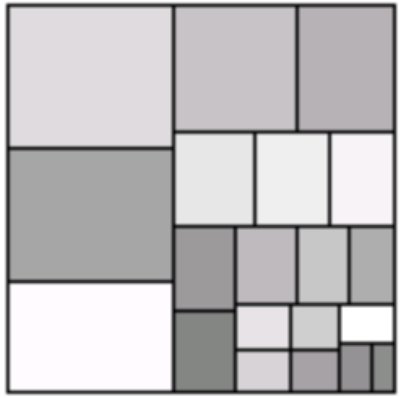


Image from DataVizCatalogue.com

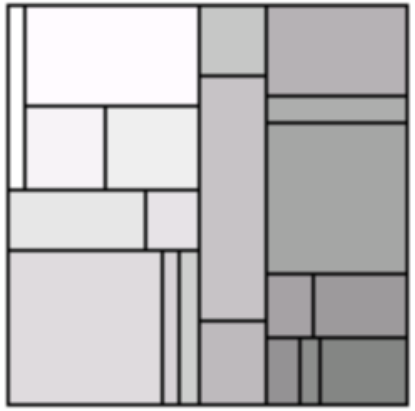
Treemap Layout Algorithms



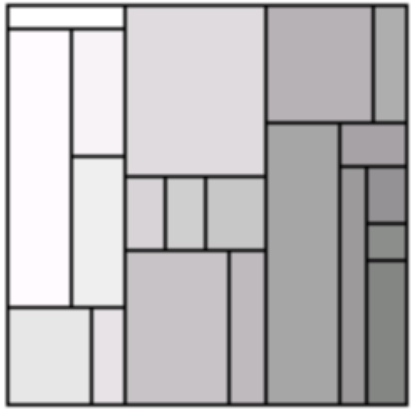
Cluster



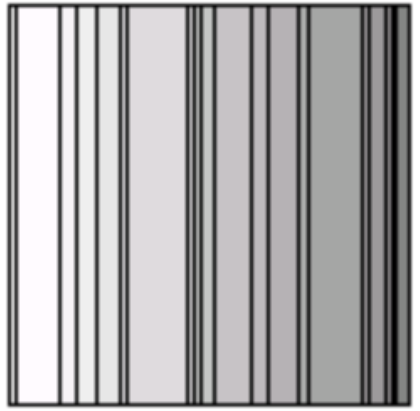
Squarified



Pivot-by-Middle

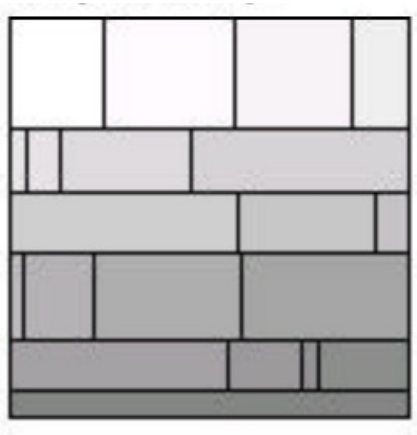


Pivot-by-Size



Slice-and-Dice

Grayscale indicates index order of nodes.

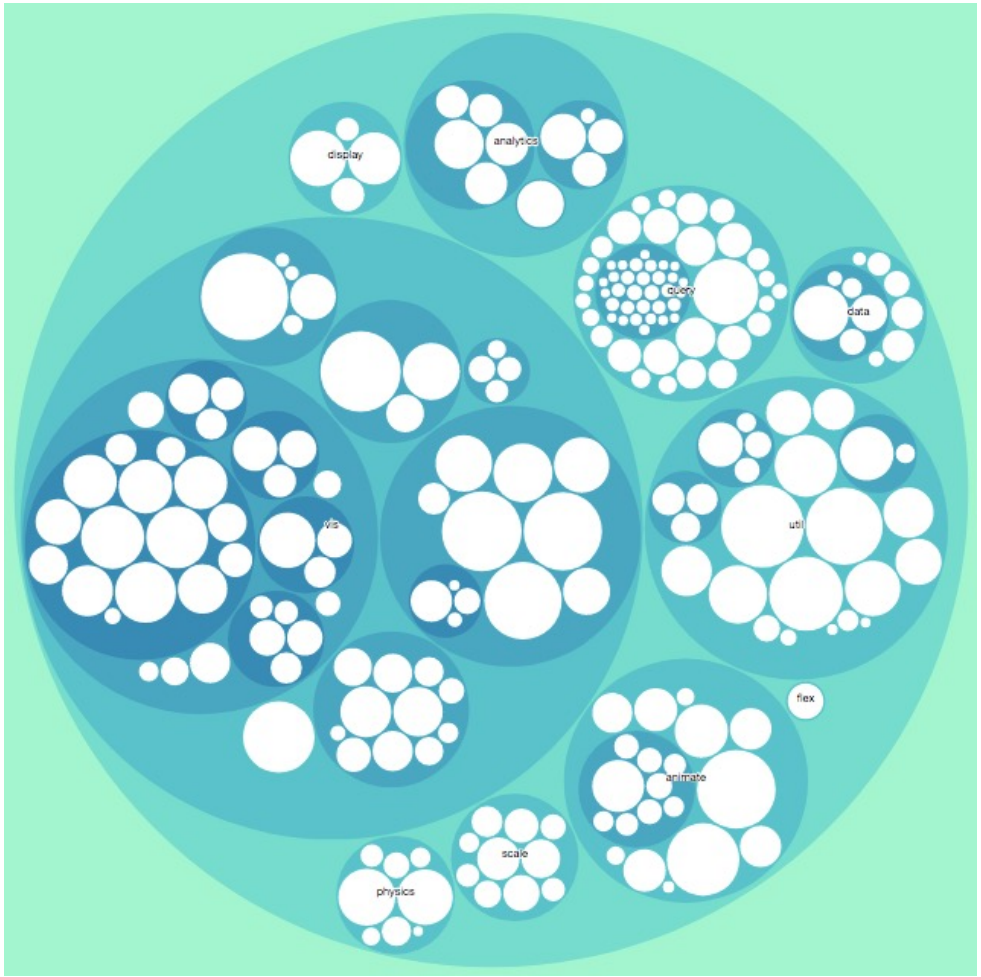


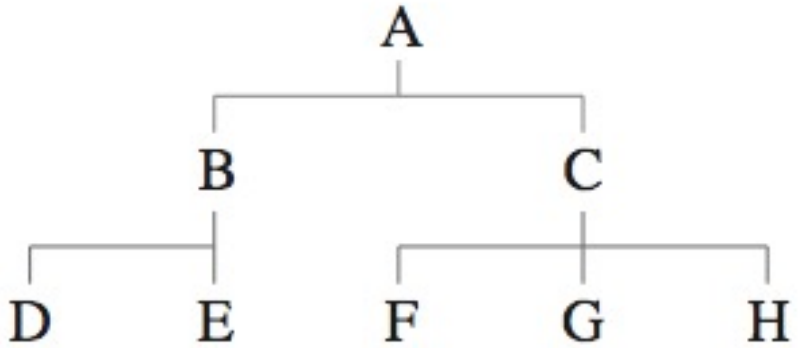
Strip

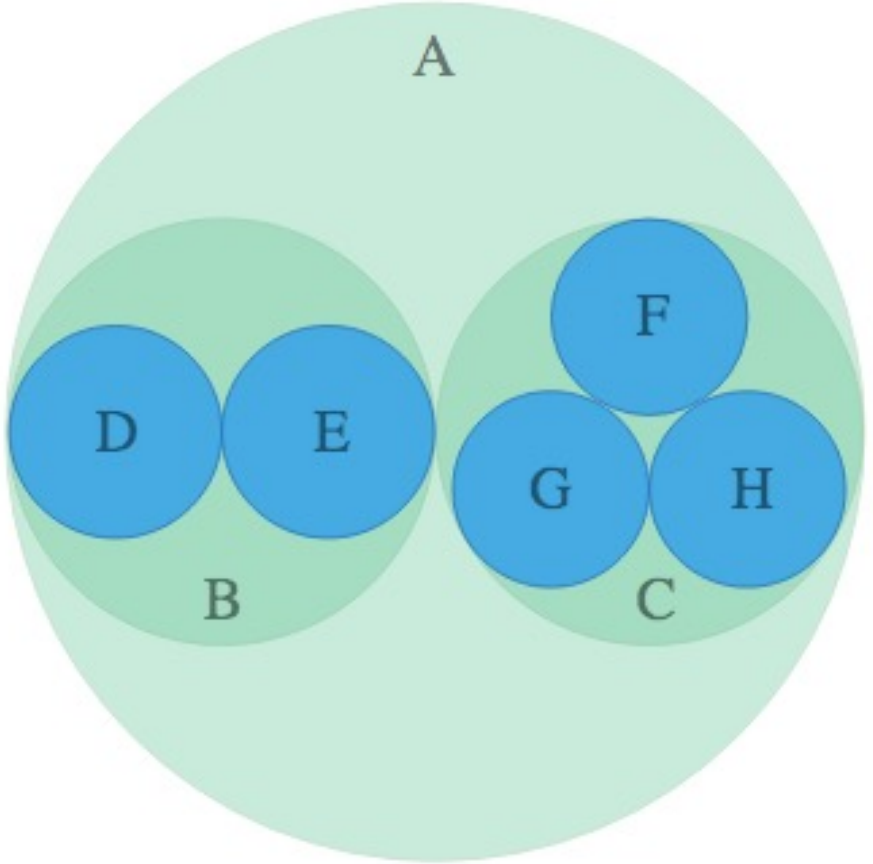
Images from Bederson, Shneiderman, & Wattenberg, “Ordered and Quantum Treemaps”


TECHNOLOGY
INTERNET INFORMATION
GOOGL -0.34%
FB +0.80%
APPLICATION S
ORCL -0.14
ADBE +0.04
CRM -0.13
INTU
SEMICONDUCT
INTC -0.45%
TXN
AVGO +0.97
BUSINESS SOFTWARE & S
MSFT +1.33%
TELECOM S
T
VZ
INFORMATIO
IBM -0.72
ACN
FIS
NVDA -0.28
AMAT
DIVER
CCI
MU
COMMUNIC
CSCO +0.12%
ATVI
EA
SERVICES
CATALOG & MAIL O
AMZN +0.72%
DISCOUN
WMT +2.38%
COST -0.46
TGT
ENTERTAI
DIS
CMCSA +1.38%
FOXA
BUSINESS
BKNG -0.21%
CATV SYS
NFLX +1.11%
RESTAU
MCD +1.47%
MAR
RAILROA
UNP -0.78
CSX
NSC
TJX
SBUX +2.03
DAL
HOME I
HD
LOW
AIR DELI
UPS -0.81
FDX
LUV
BASIC MATERIALS
MAJOR INT
XOM +0.38%
CVX
INDEPE
COP +1.62
EOG
OXY
SPECIAL
PX
LYB
CHEMIC
DWDP -2.04%
SLB +0.12%
HAL
KMI
PSX
VLO
HEALTHCARE
DRUG MANUFACTURE
JNJ -0.38%
MRK +1.01%
ABBV -0.19%
PFE
LLY -0.36
BMY +0.51
BIOTECHN
AMGN +0.67%
GILD +0.78
BIIB
VRTX
CELG
MEDICAL
MDT +0.22
ABT -0.18
SYK
BSX
HEALTH CARE PLANS
UNH +0.42%
CVS -0.48
AET
CI
MEDICA
BDX +0.17
ISRG
BAX
MEDIC
TMO
AGN
ZTS
HCA
FINANCIAL
MONEY CENTER BANKS
JPM -0.56%
WFC +0.12%
BAC -0.68%
C
PNC -0.12
STI
PROPERTY & CASUAL
BRK-B +0.39%
AIG
INVESTM
MS -1.23
GS -0.69
CME
CREDIT SERVICES
V
MA +0.54%
PYPL -0.76
AXP -0.47
DFS
SYF
ASSET
BLK
BK
STT
REIT -
LIFE I
REIT -
AMT +0.74
ICE
REGION
USB
INSUR
AFL
CONSUMER GOODS
ELECTRONIC
AAPL +0.81%
BEVERAGES -
KO -0.10%
PEP +0.23%
PERSONAL
PG +0.04%
CL
EL
KMB
CIGARE
PM
MO
K
F
IP
KHC
STZ
JCI
TEXTILE
NKE
INDUSTRIAL GOODS
DIVERSIFIE
MMM -1.04
HON -1.88
GE +0.44%
DHR
ITW
ETN
IR
AEROSPACE/DEFEN
BA -0.12%
UTX -2.22%
GD
RTN
LMT
CAT -2.37
DE
UTILITIES
ELECTRIC U
NEE -0.13
SO
D
DIVERSI
Use mouse wheel to zoom in and out. Drag zoomed map to pan it.
Double-click a ticker to display detailed information in a new window.
Hover mouse cursor over a ticker to see its main competitors in a stacked view with a 3-month history graph.
-3%
-2%
-1%
0%
+1%
+2%
+3%

Treemaps need not be rectangular







Also known as circle packing, Images from DataVizCatalogue.com and https://bl.ocks.org/mbostock/7607535

Indented Trees

- Parent-child relationships shown via indention
- Trade-off between breadth (one level) and depth (to the leaves)

```
root
├── dir1
├── dir2
│   └── file1
├── dir3
│   ├── file2
│   ├── file3
│   └── dir4
│       └── file4
└── file5
```

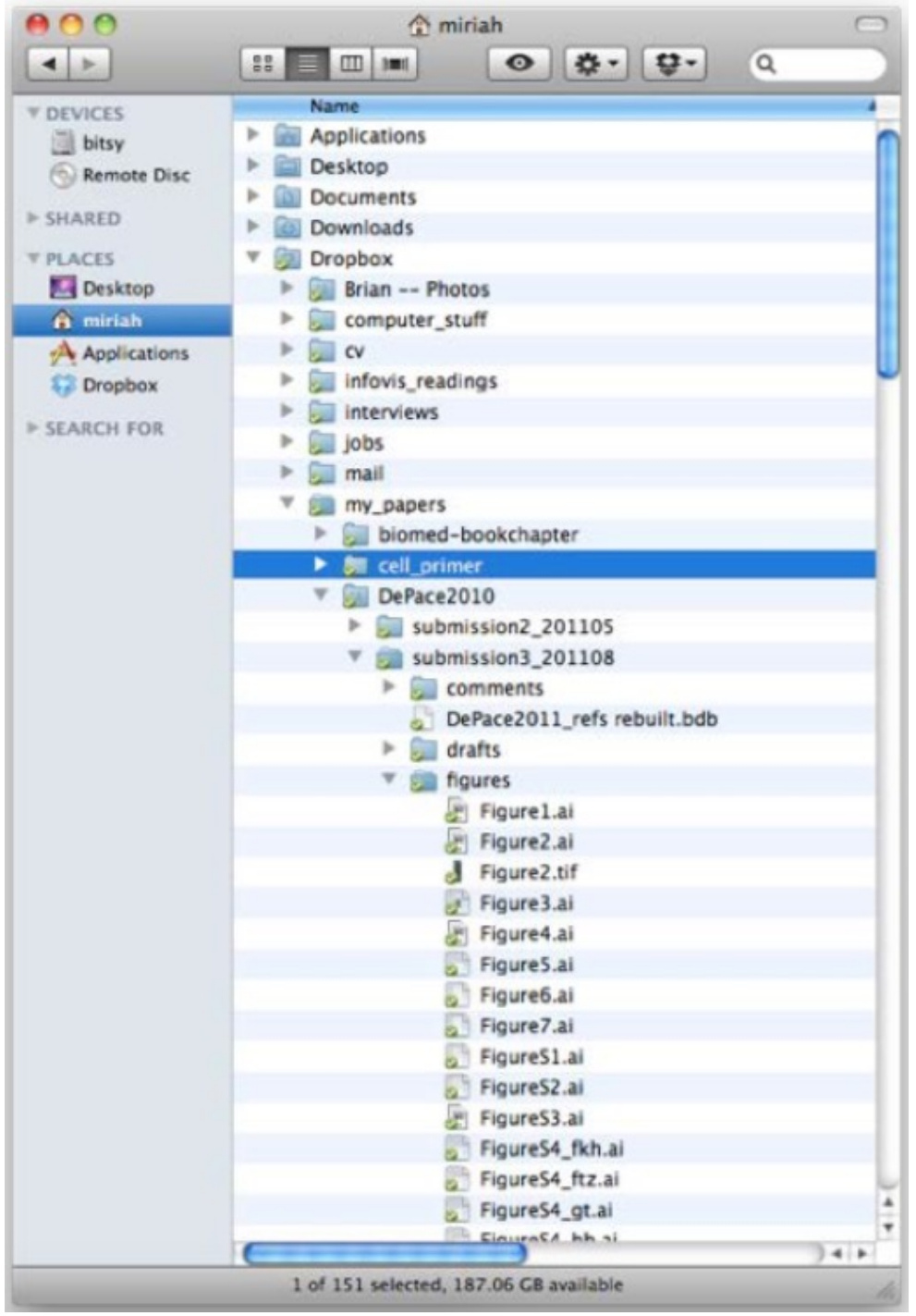


Image from Miriah Meyer & Josh Levine, slides and https://atom.io/packages/ascii-tree

Layered Trees

Parent-child relationships shown via layering, adjacency, & alignment

Layout similar to node-link without edges

Extent of parent (e.g., length, angle) constrains extent of children, similar to enclosure

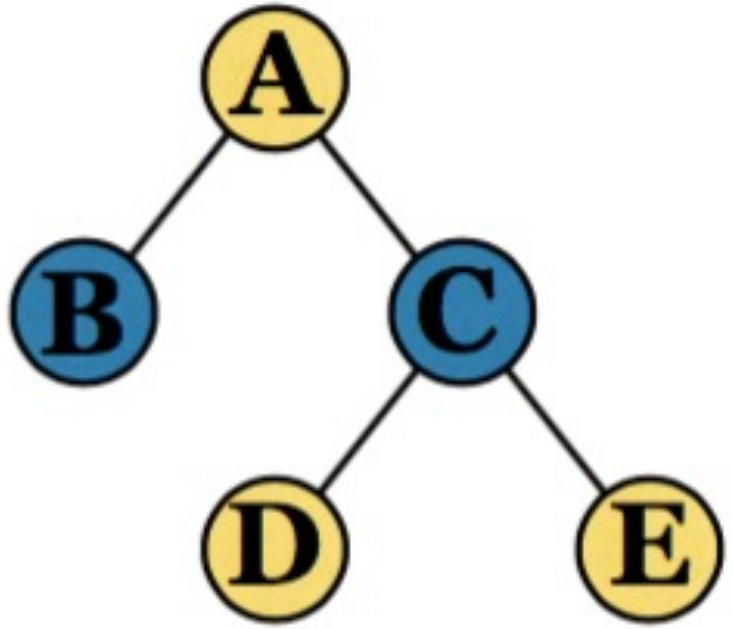
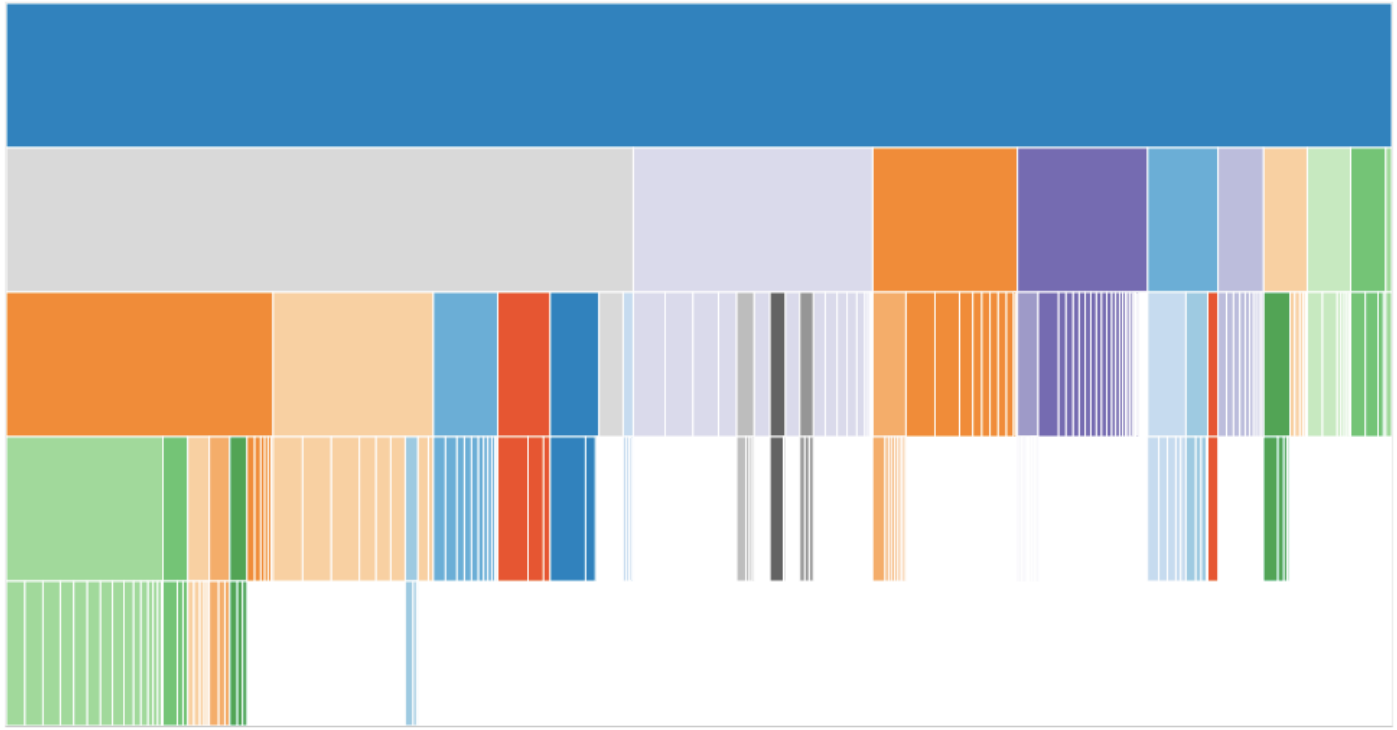
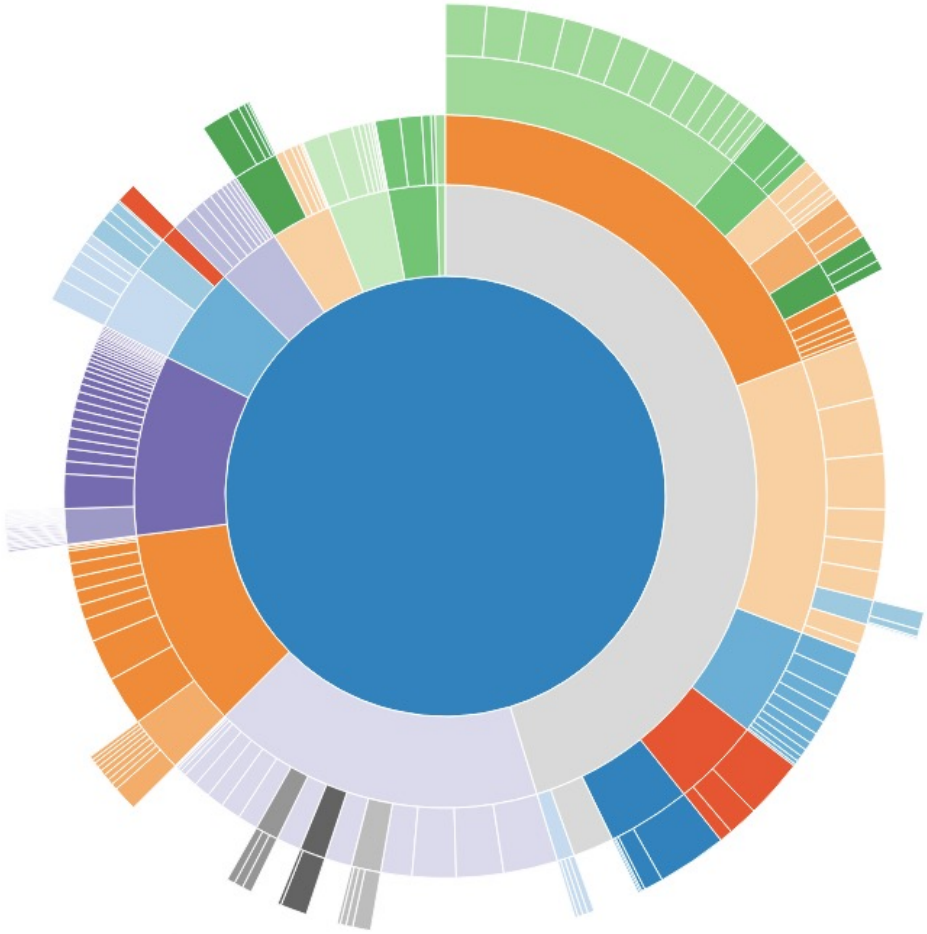


Image from Miriah Meyer & Josh Levine, slides and https://atom.io/packages/ascii-tree

Icicle charts and sunbursts use layering

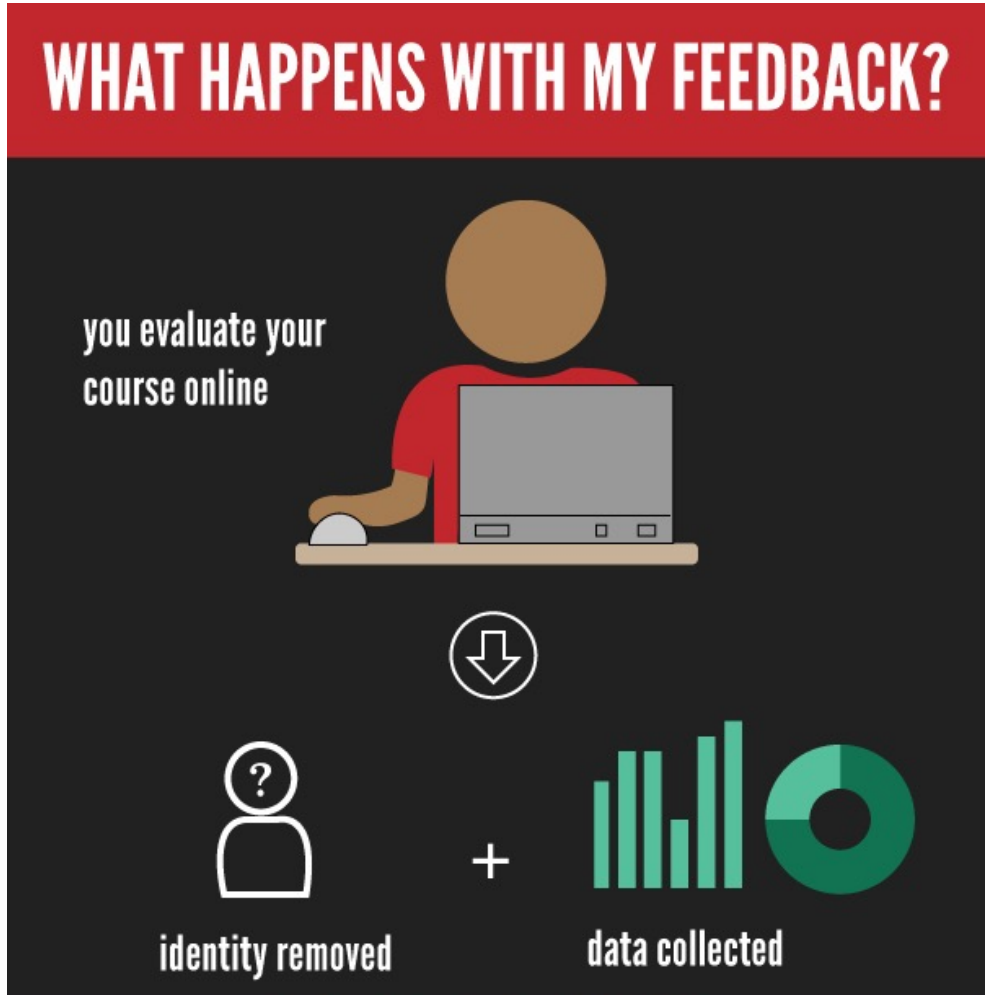


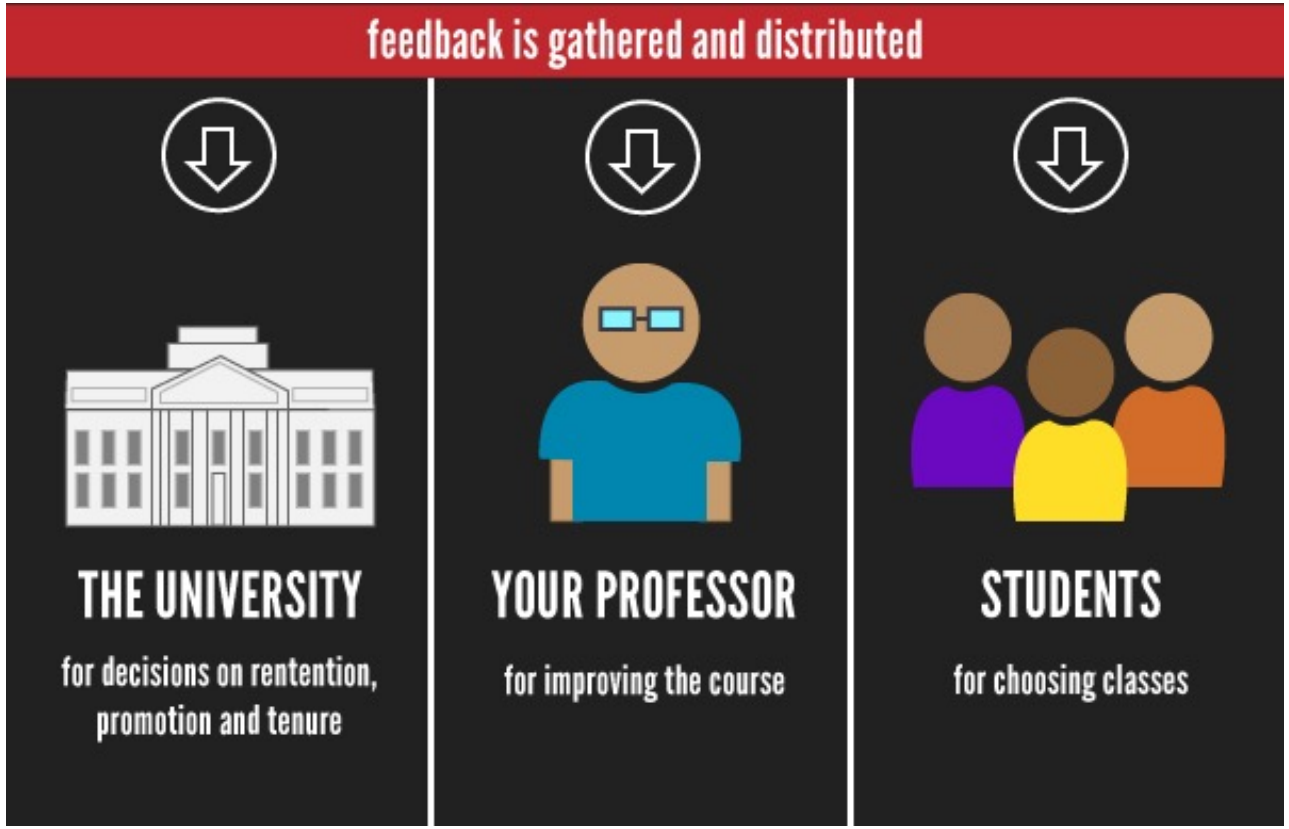


Images fromhttps://bl.ocks.org/mbostock/1005873,
https://bl.ocks.org/mbostock/4348373

Student Course Feedback

https://scf.utah.edu

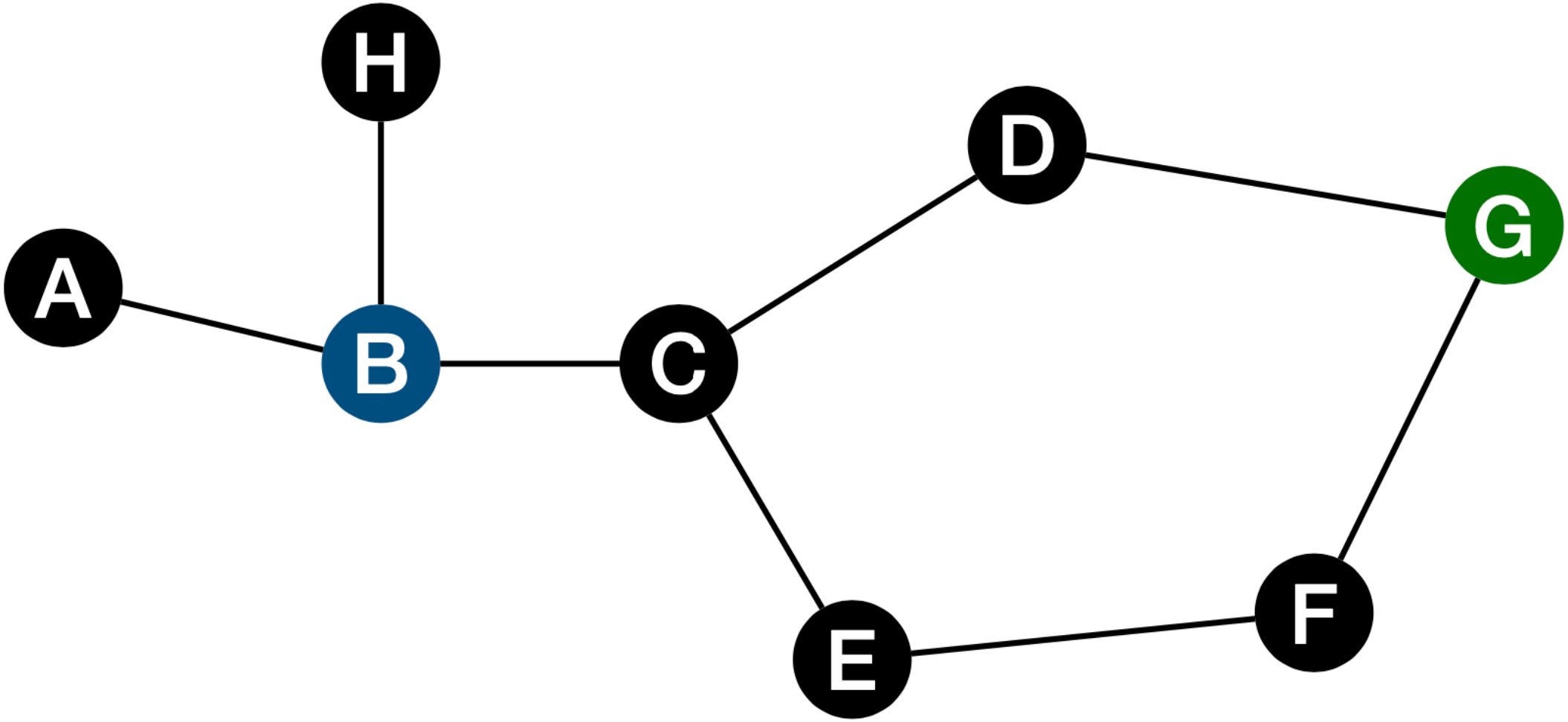




Shortest Path Algorithms

Breadth-First Search Example

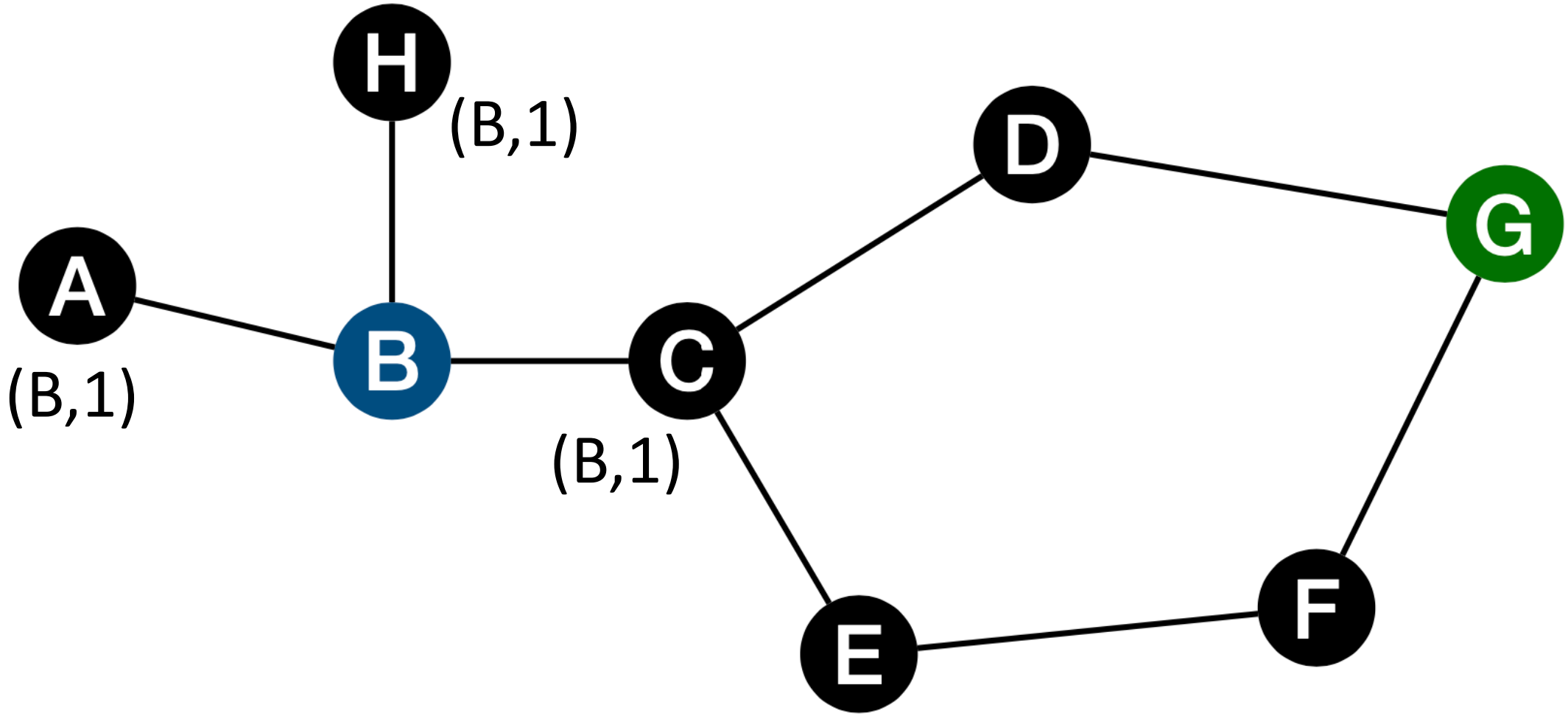
Shortest path from B to G



[B]

Breadth-First Search Example

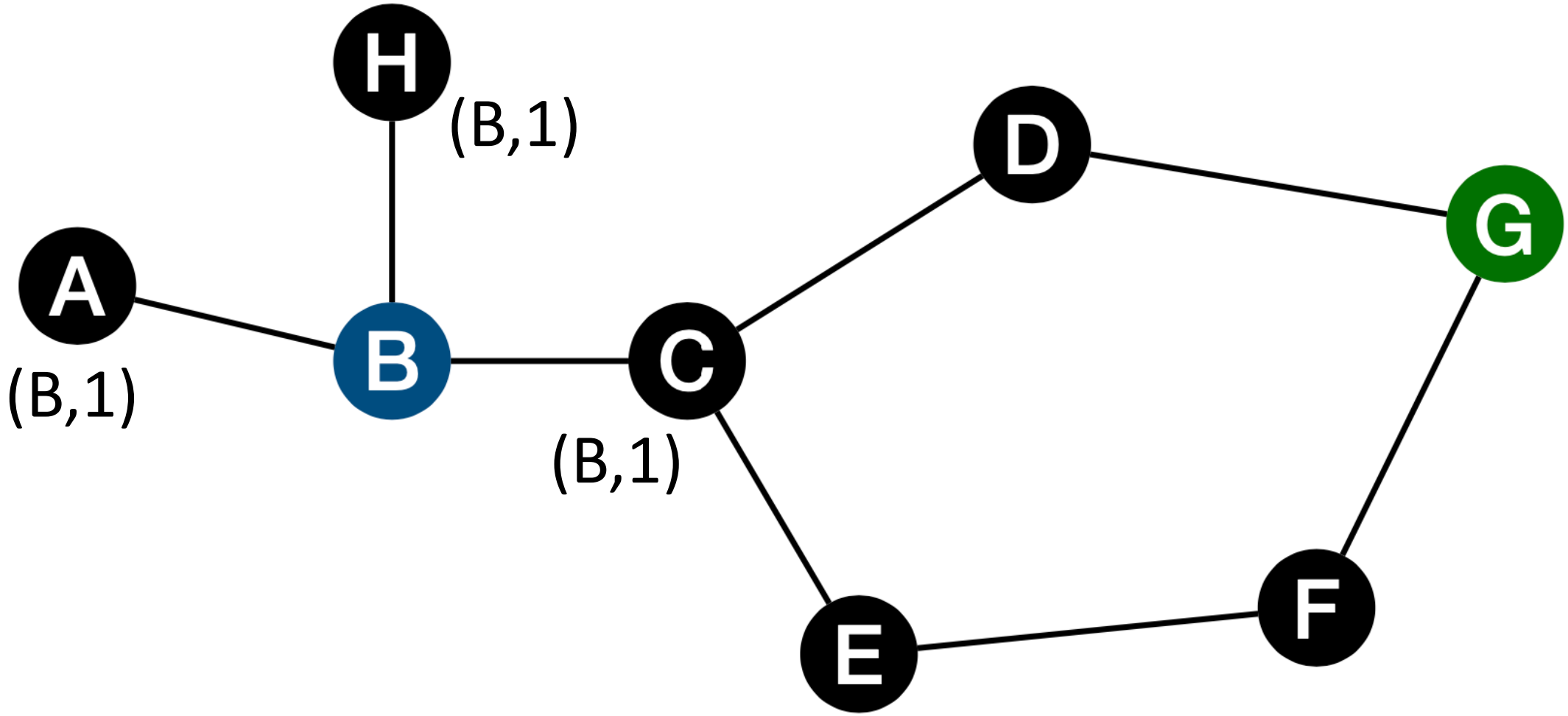
Shortest path from B to G



[~~B~~, A, H, C]

Breadth-First Search Example

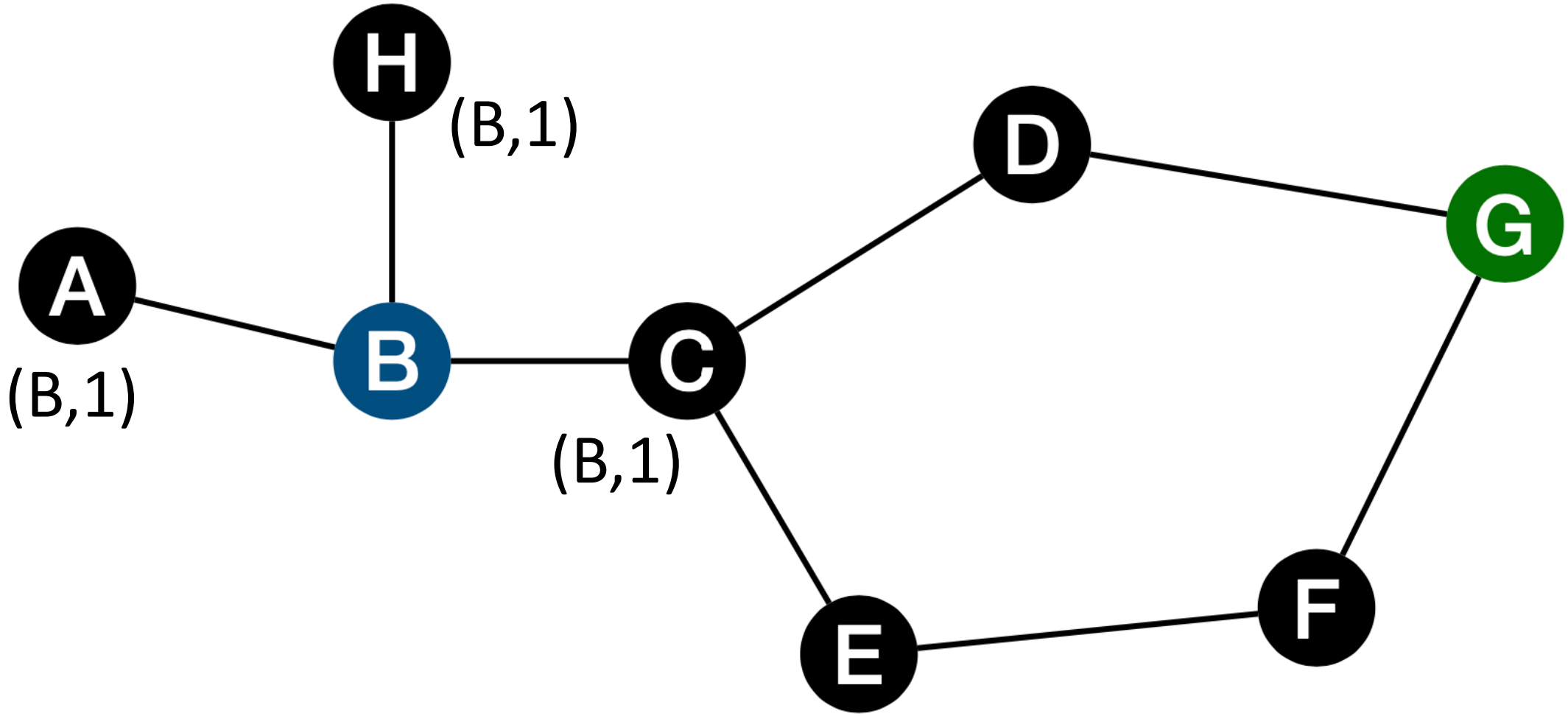
Shortest path from B to G



[~~B~~, ~~A~~, H, C]

Breadth-First Search Example

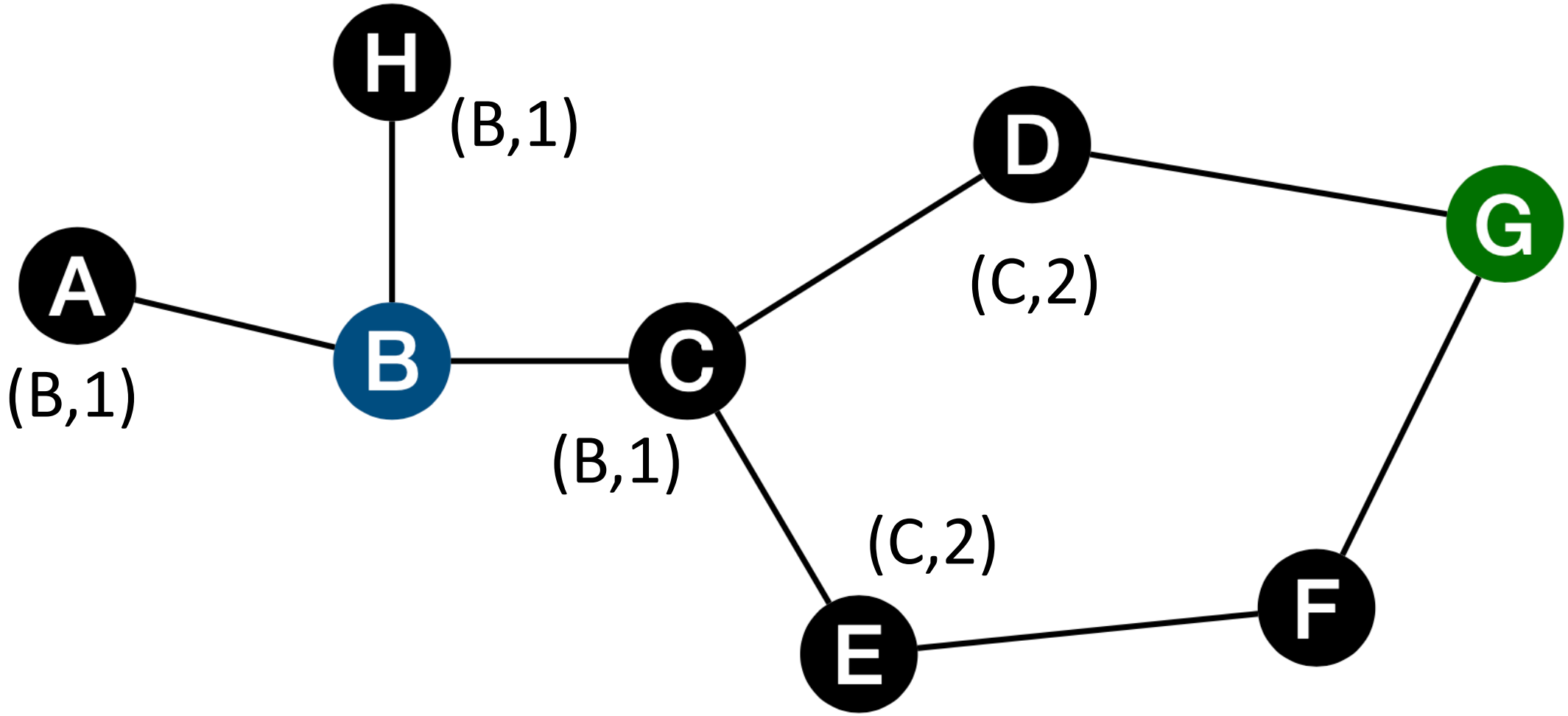
Shortest path from B to G



[~~B~~, ~~A~~, ~~H~~, C]

Breadth-First Search Example

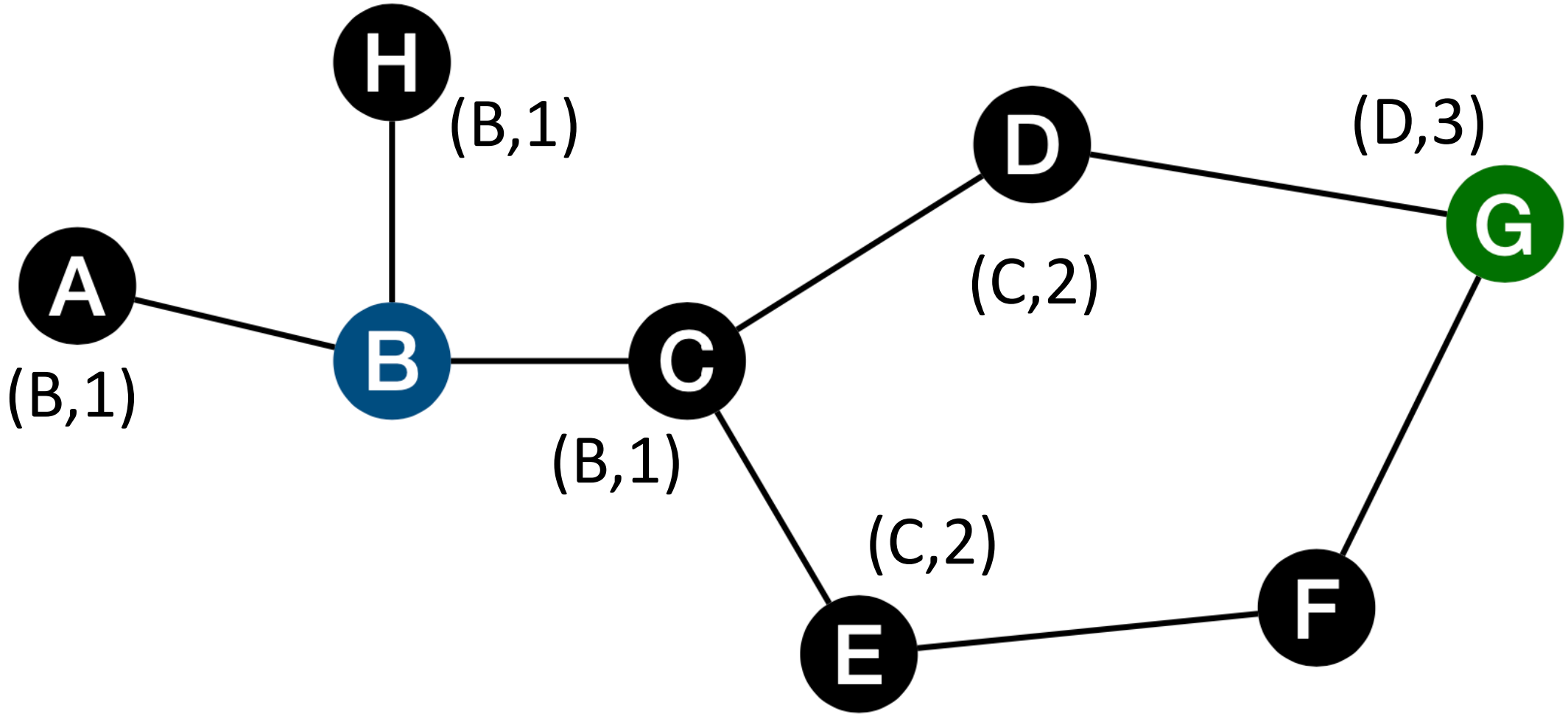
Shortest path from B to G



[~~B~~, ~~A~~, ~~H~~, ~~C~~, D, E]

Breadth-First Search Example

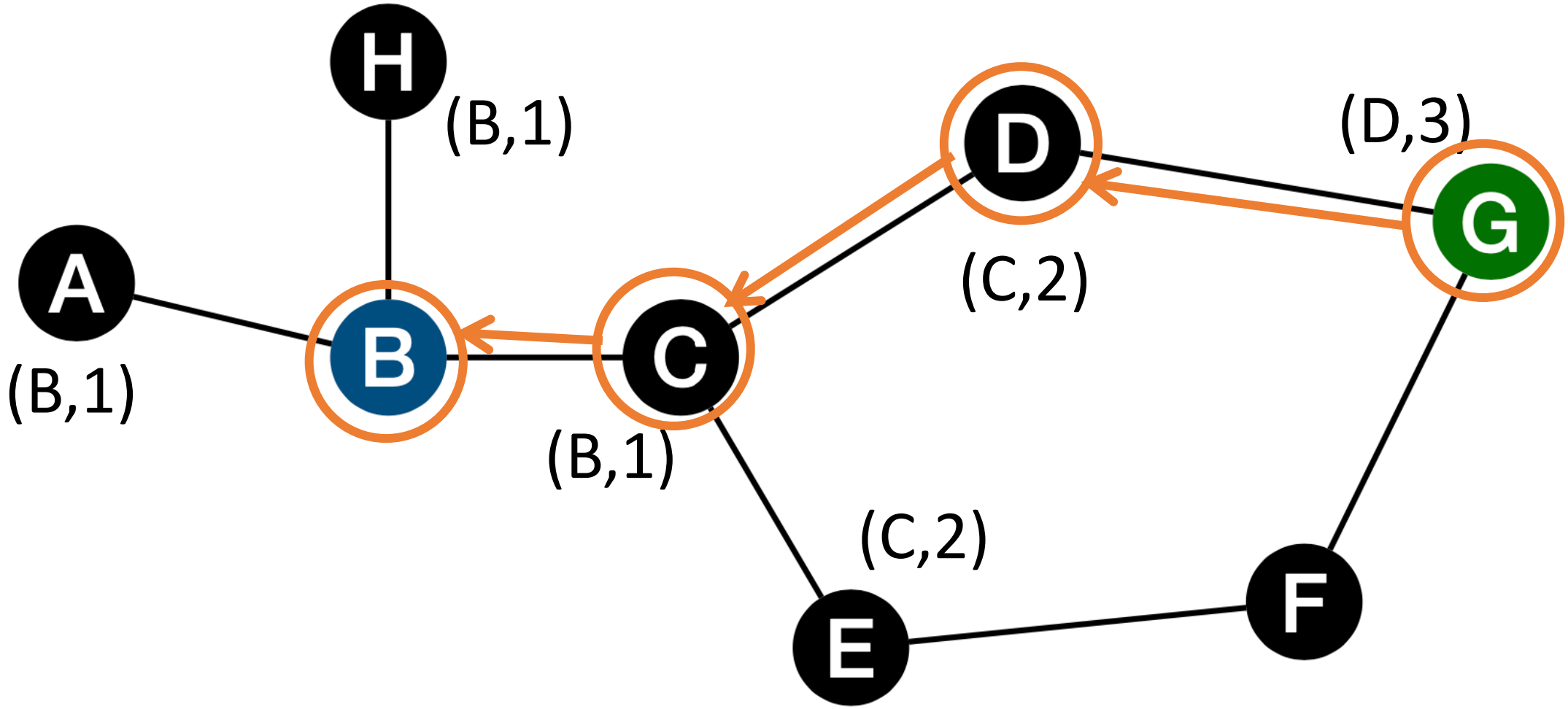
Shortest path from B to G



[~~B~~, ~~A~~, ~~H~~, ~~C~~, ~~D~~, E, **G**]

Breadth-First Search Example

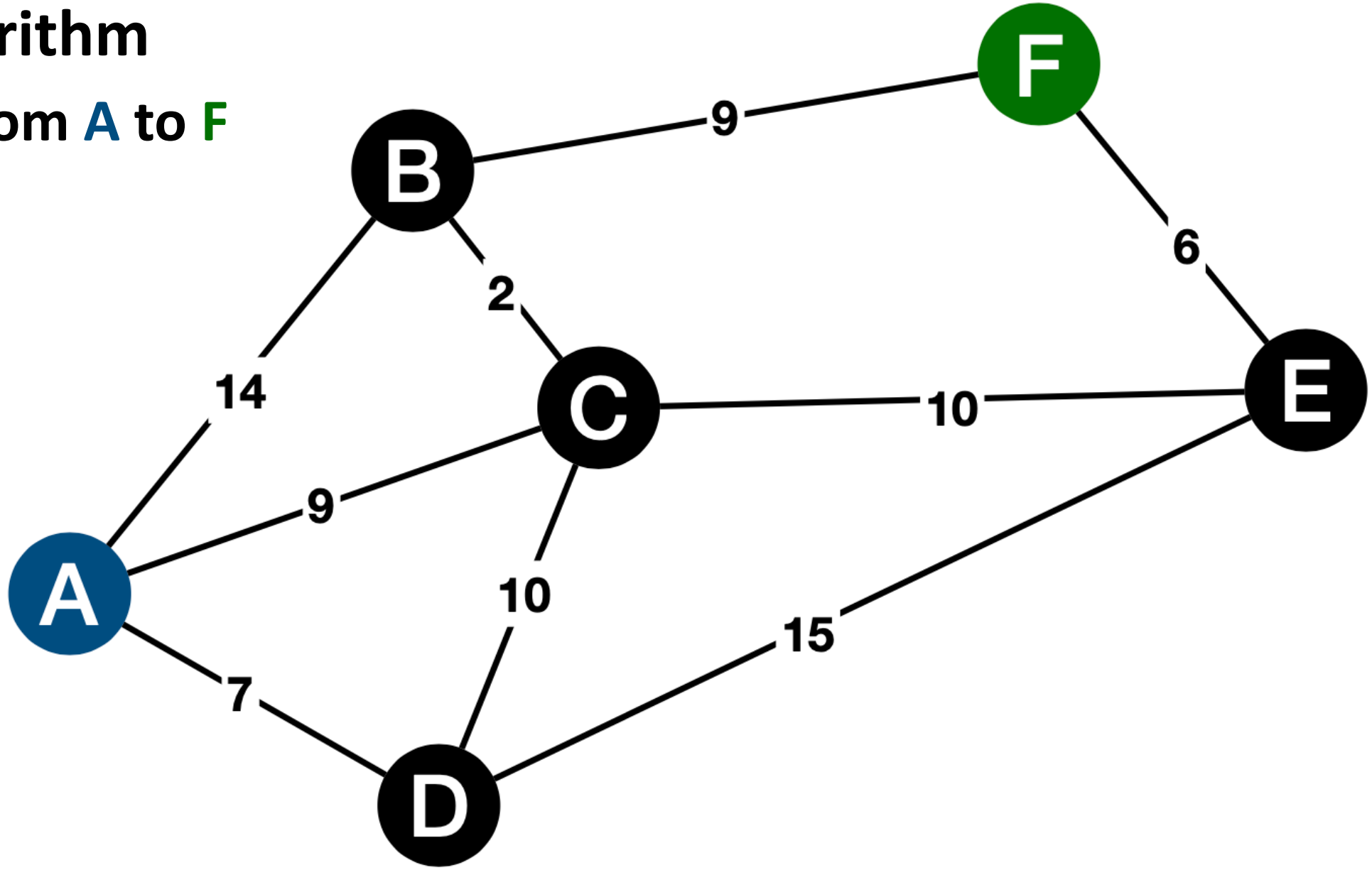
Shortest path from B to G



[~~B~~, ~~A~~, ~~H~~, ~~C~~, ~~D~~, E, **G**]

Dijkstra's Algorithm

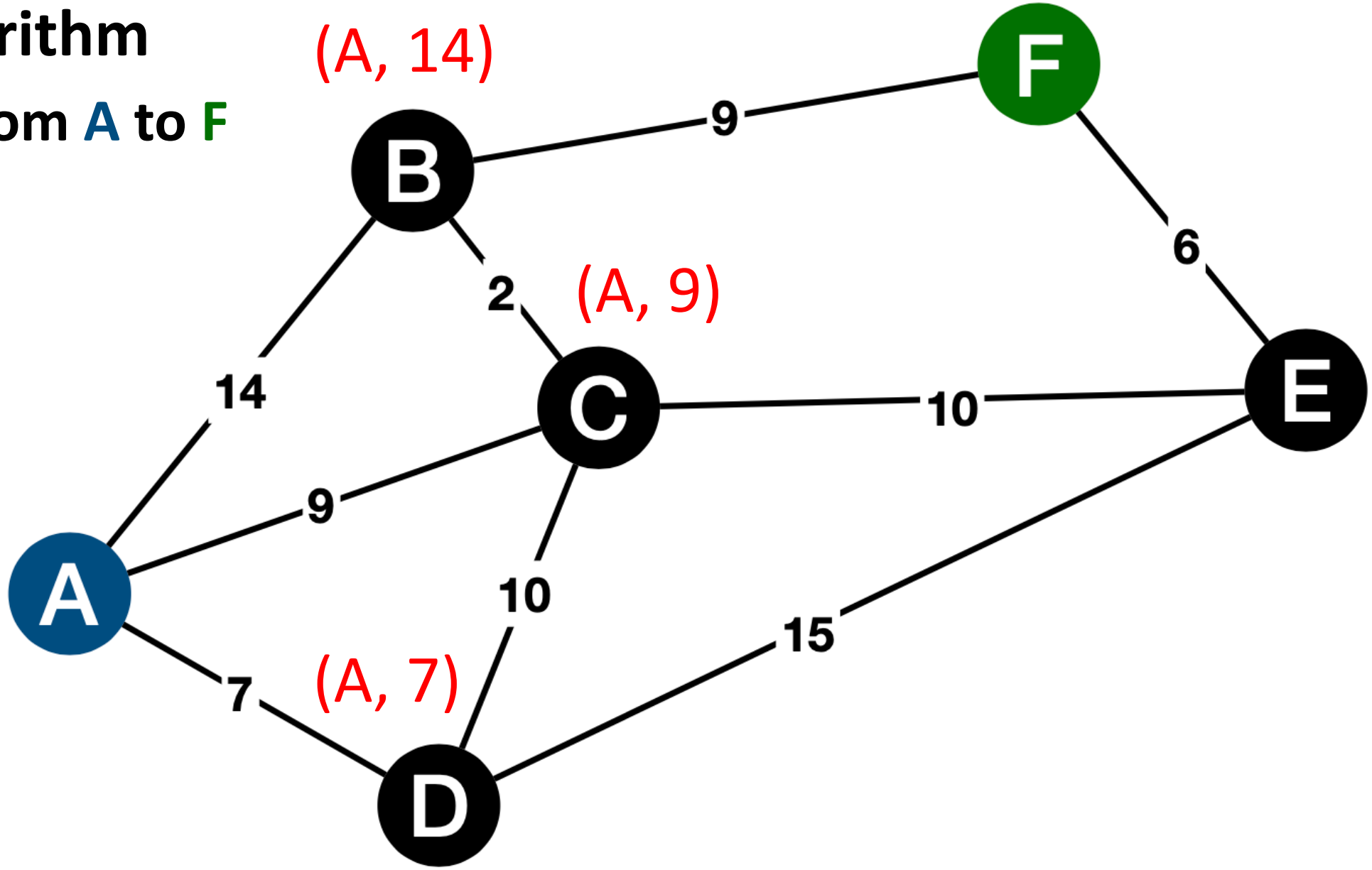
Shortest path from A to F



[A]

Dijkstra's Algorithm

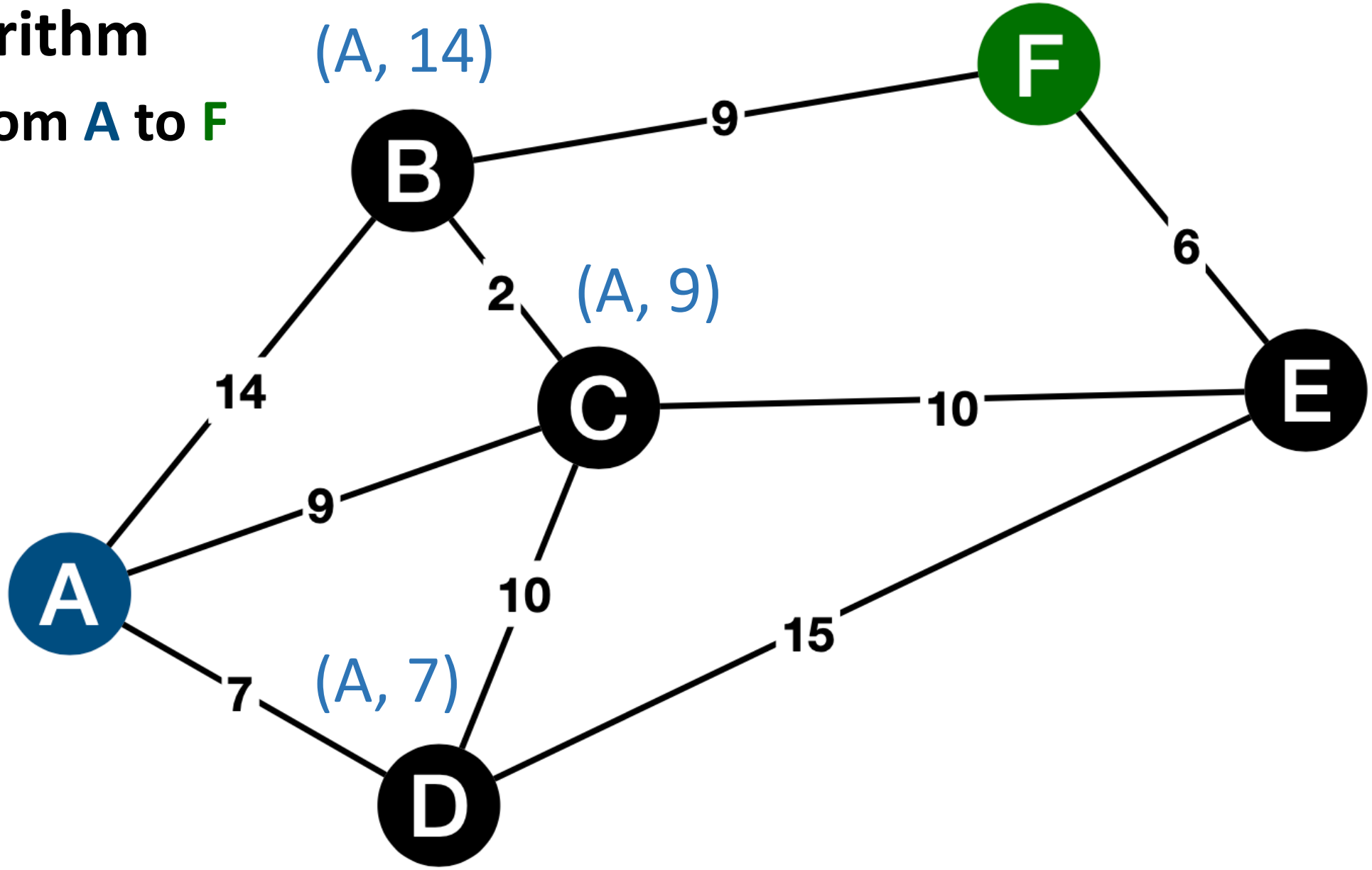
Shortest path from A to F



[**A**, D(7), C(9), B(14)]

Dijkstra's Algorithm

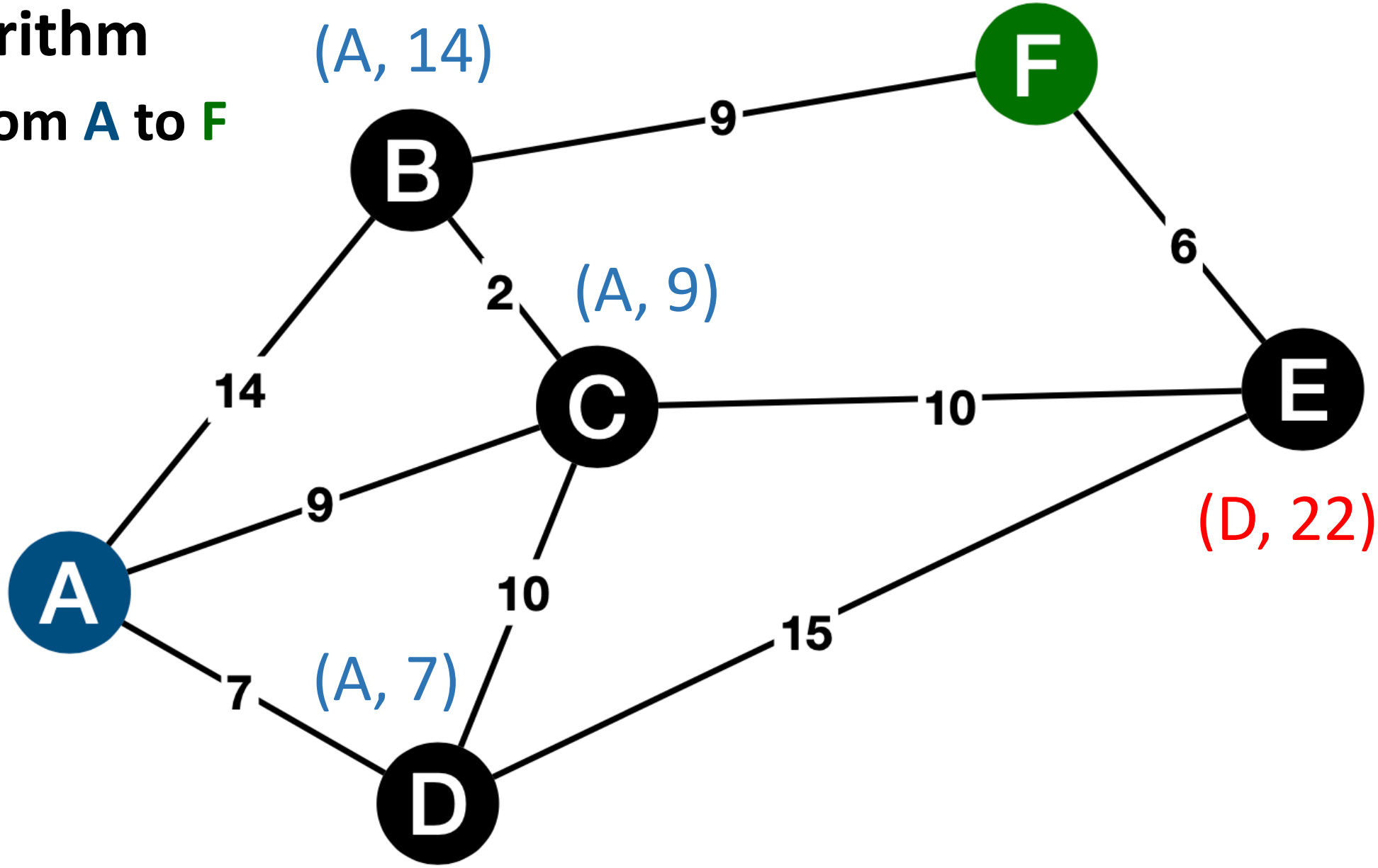
Shortest path from A to F



[~~A~~, D(7), C(9), B(14)]

Dijkstra's Algorithm

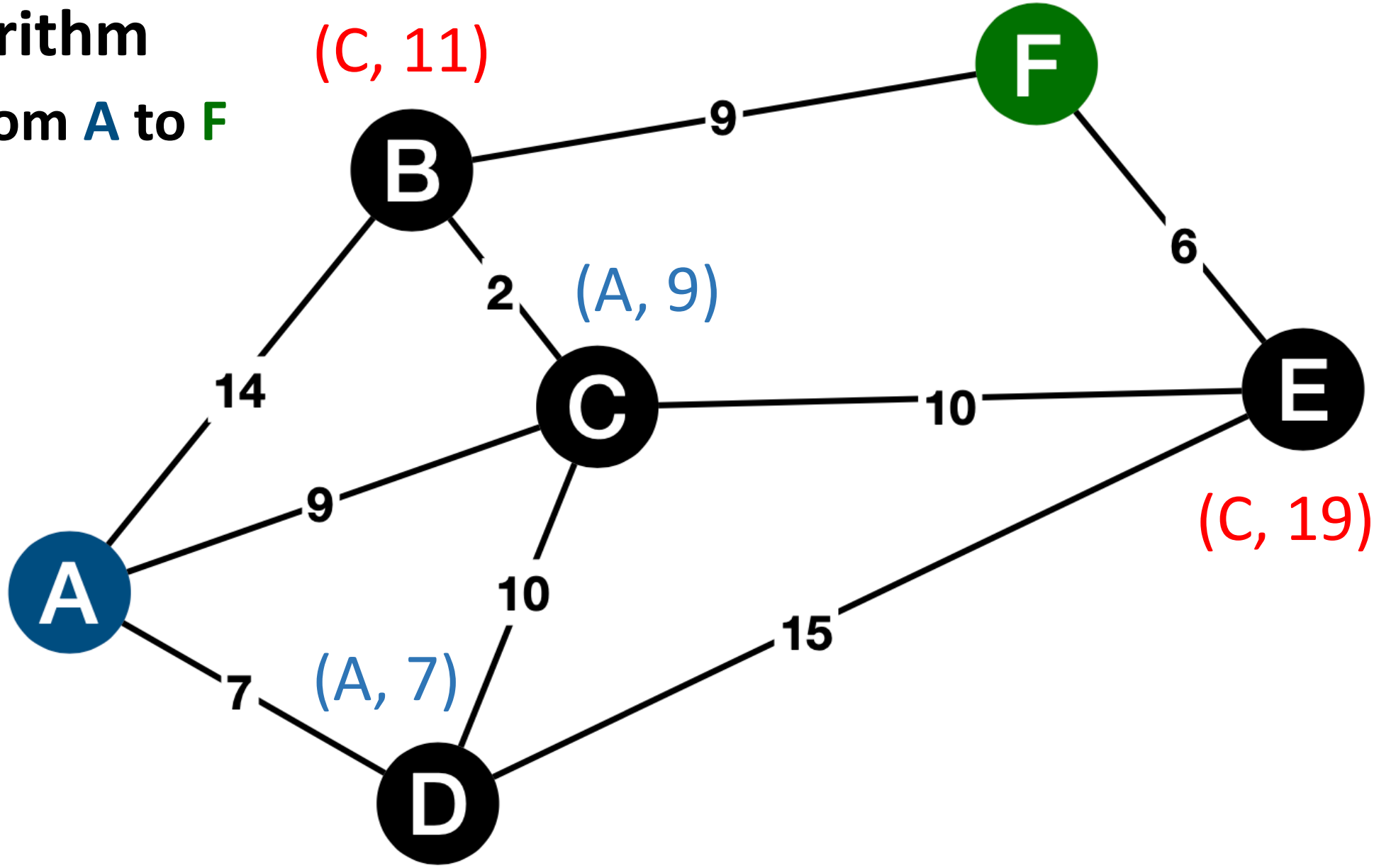
Shortest path from A to F



[~~A~~, **D(7)**, C(9), B(14), E(22)]

Dijkstra's Algorithm

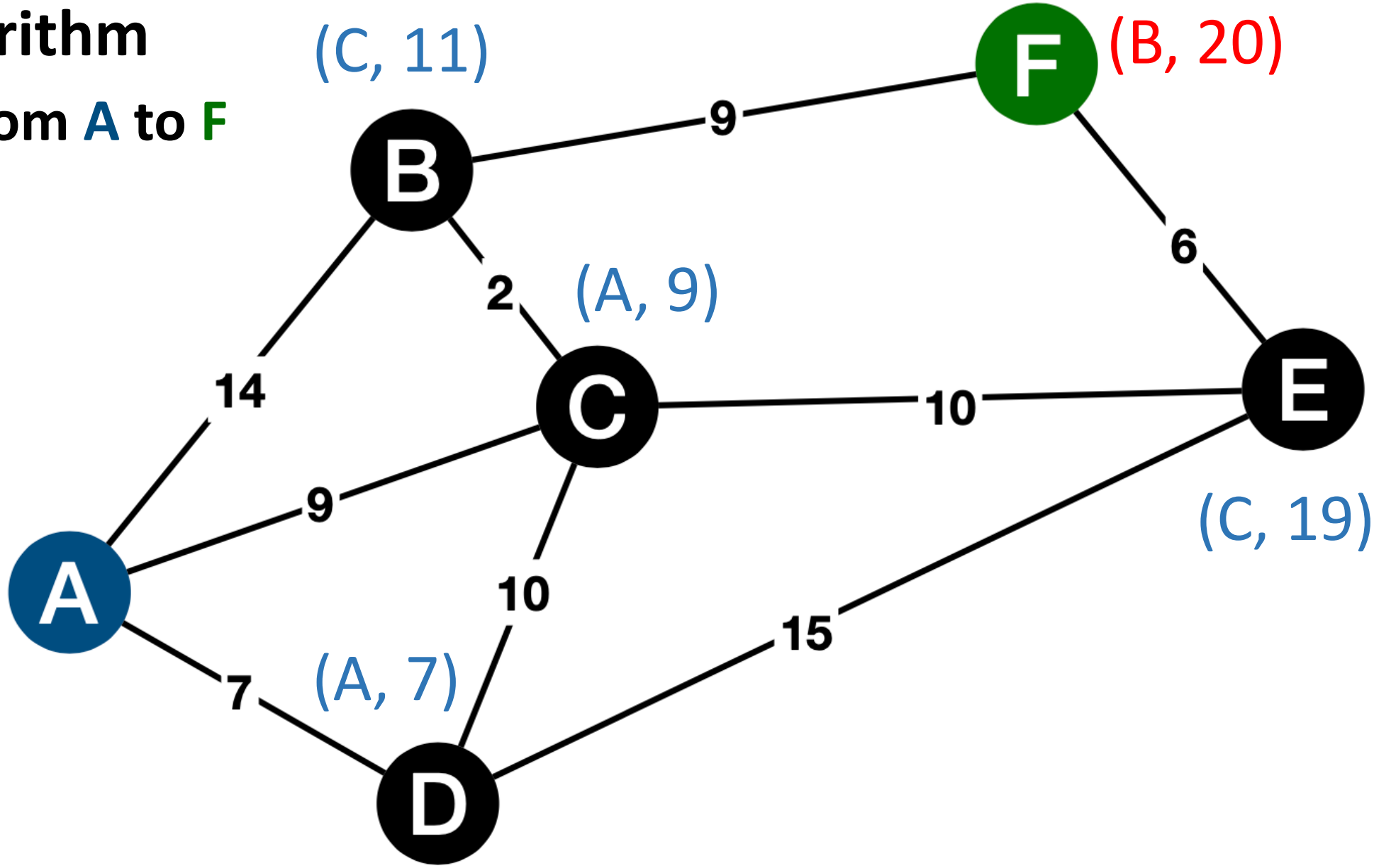
Shortest path from A to F



[~~A~~, ~~D(7)~~, **C(9)**, B(11), E(19)]

Dijkstra's Algorithm

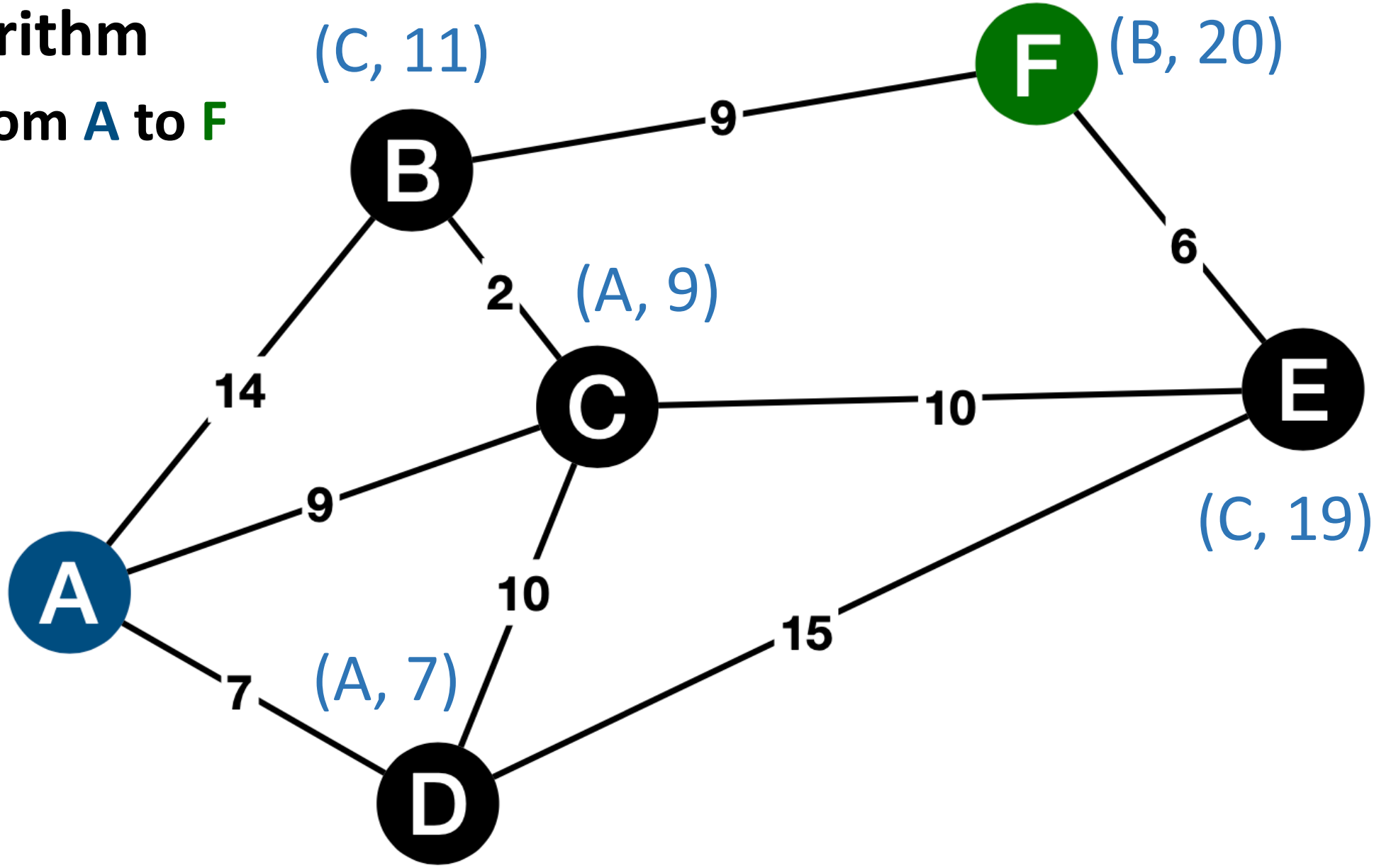
Shortest path from A to F



[~~A~~, ~~D(7)~~, ~~C(9)~~, **B(11)**, E(19), F(20)]

Dijkstra's Algorithm

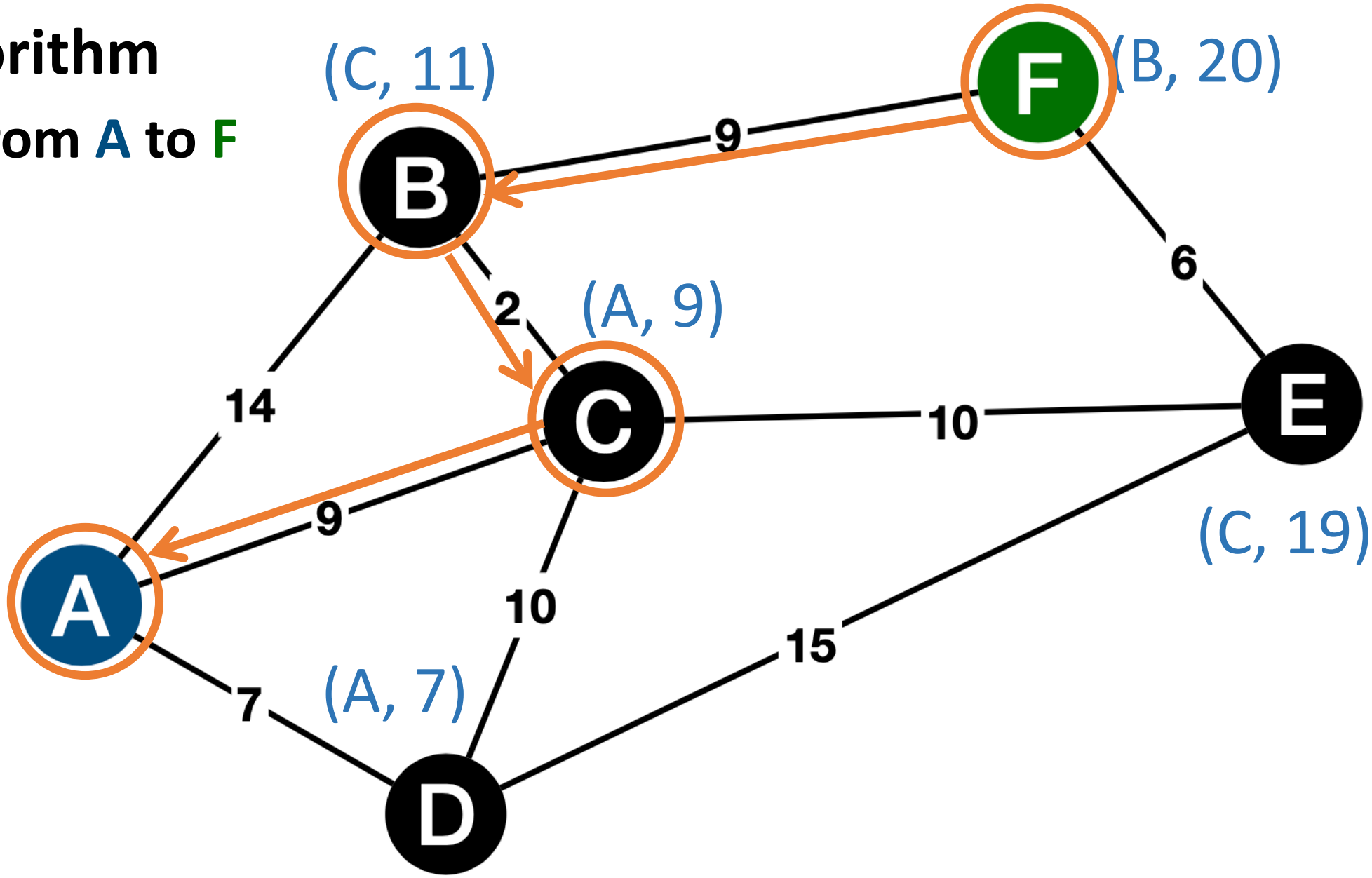
Shortest path from A to F



[~~A~~, ~~D(7)~~, ~~C(9)~~, ~~B(11)~~, **E(19)**, F(20)]

Dijkstra's Algorithm

Shortest path from A to F



[~~A~~, ~~D(7)~~, ~~C(9)~~, ~~B(11)~~, ~~E(19)~~, **F(20)**]