Graphs and Social Networks

Roshni Iyer
CS Department
UCLA
roshnigiyer@cs.ucla.edu

Debarnab Mita
ECE Department
UCLA
debarnabucla@ucla.edu

Swapnil Sayan Saha
ECE Department
UCLA
swapnilsayan@g.ucla.edu

July 2021

Note: Slides adapted from original LACC slides from Dr. Luis Garcia & Saptadeep Pal
Some materials adapted from UCLA ECE 232E course by Prof. Vwani Roychowdhury

Variability Expedition

Instructors

Roshni Iyer

- Vice President, CS-GSA
- CS PhD student @ UCLA
  & CS Major from UC Berkeley
- Graduate Researcher, ScAi Lab @ UCLA
- Research Interests:
  Graph neural networks, knowledge graphs
  for various applications

Instructors

Swapnil Sayan Saha

- ECE PhD @ UCLA,
  - ECE MS from UCLA
  - EEE Major from UofDhaka
- GSR @ NESL, UCLA
- Entering Event's Director, EGSA
- Research Interests:
  Applied AI, Embedded Systems

Instructors

Debarnab Mitra

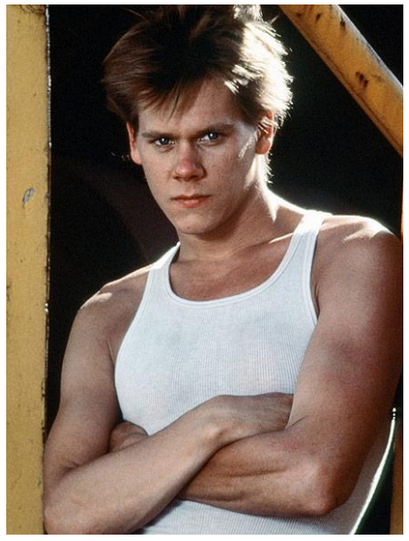
- ECE PhD @ UCLA,
  - ECE MS from UCLA
  - EE Major from IIT Bombay

- GSR @ LORIS, UCLA

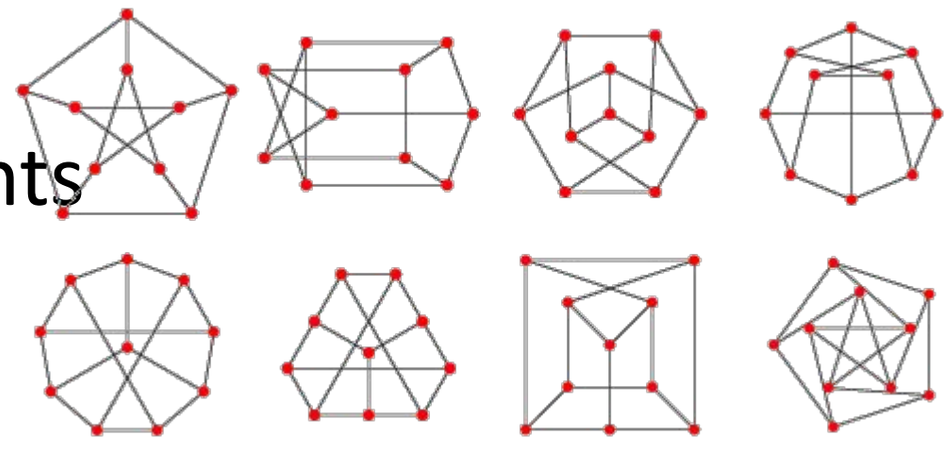
- Research Interests:
  Coding Theory, Blockchains

In This Lecture…

- We are going to:
  - Introduce the concepts of a social network and graph theory
  - Play ‘Six Degrees of Kevin Bacon’
  - Learn some basic graph theory

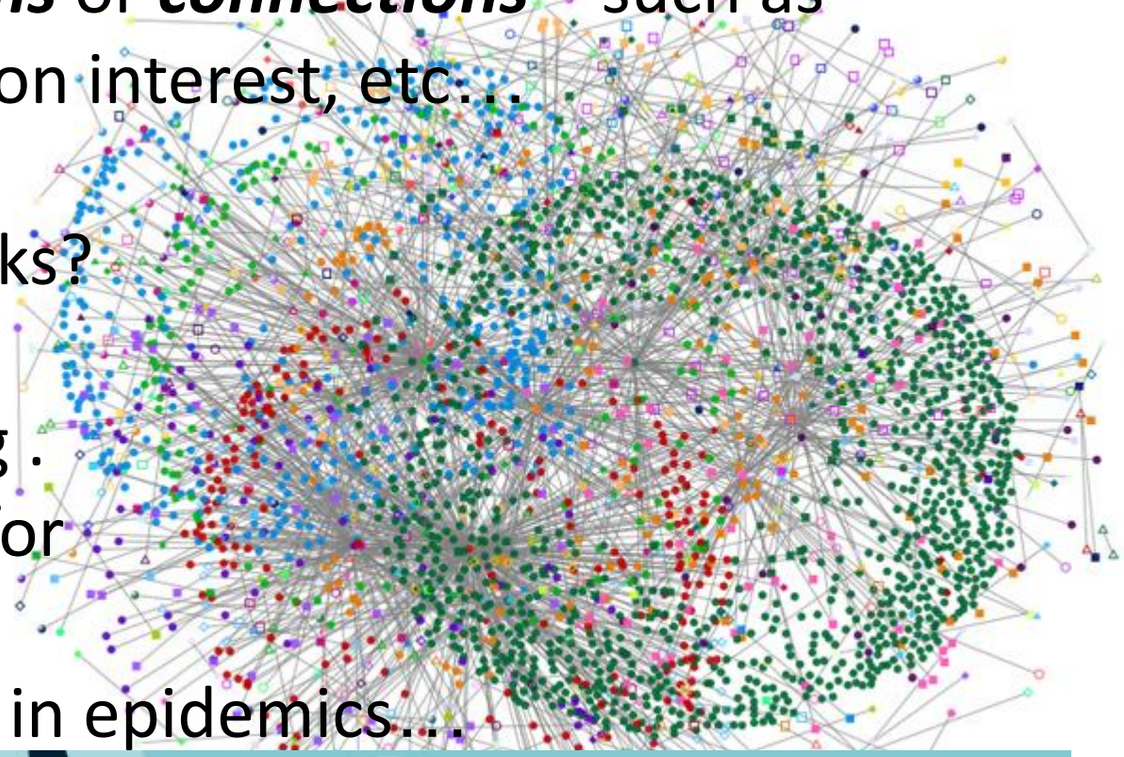


- Then we will do programming exercises:
  - **Easier**: some simple graph theory
  - **Harder**: path and cycle detection
    find connected components



Social Networks

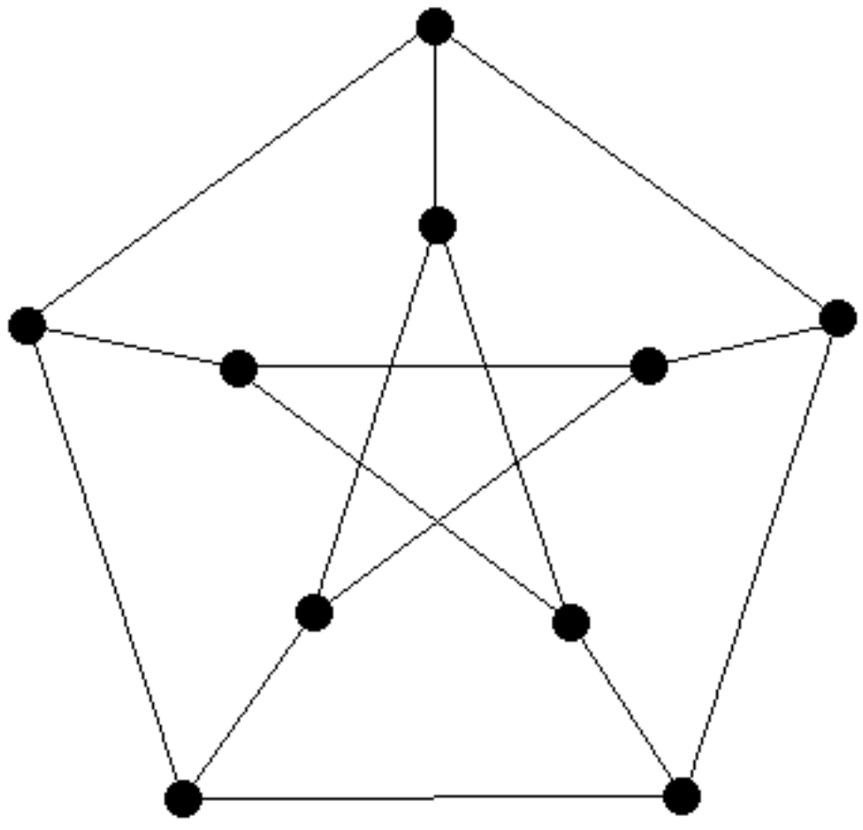
- Q: What's a social network?
- A: It is a structure made up of ***individuals*** (or organizations, or agents, or anything else) tied together by ***relations*** or ***connections*** – such as friendship, alliance, financial or common interest, etc…



- Q: Why should we study social networks?
  - It's useful! Tons of applications:
    - Finding friends and matchmaking .
    - Locating appropriate customers for companies (marketing).
    - Preventing spreading of diseases in epidemics…

Graph Theory

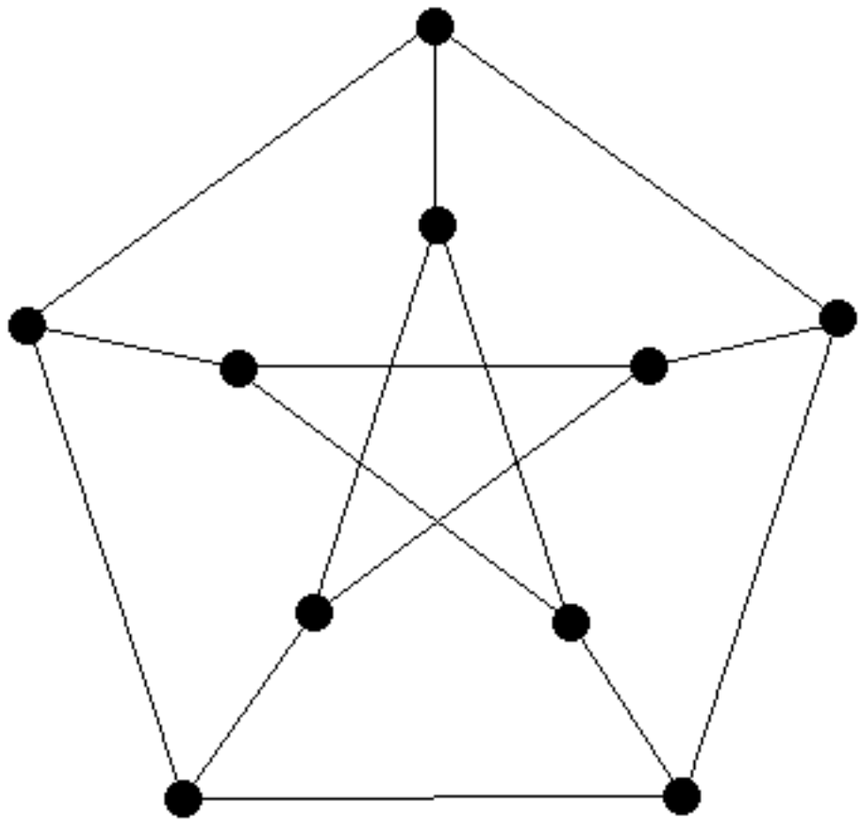
- The main tool used to understand social networks is graph theory.
  - An example of a graph:



Petersen graph

Graph Theory

- A graph is made up of nodes (also called vertices) and edges:
- Which are the nodes and which are the edges in the graph below?

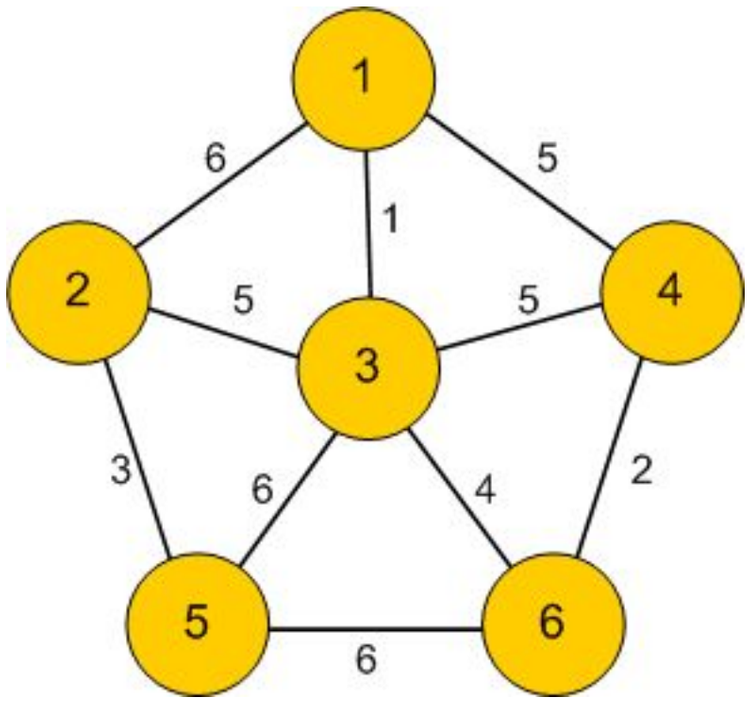


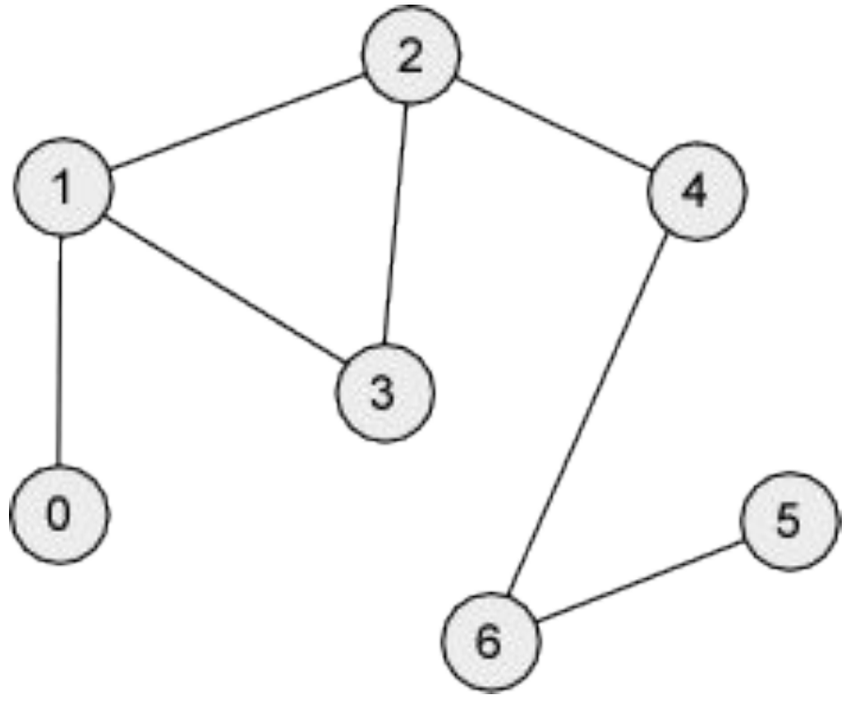
Petersen graph

Graph Theory

- Edges can be **weighted** or **unweighted (= uniformly weighted graph).**
- Weights: allow us to express the importance of a connection.

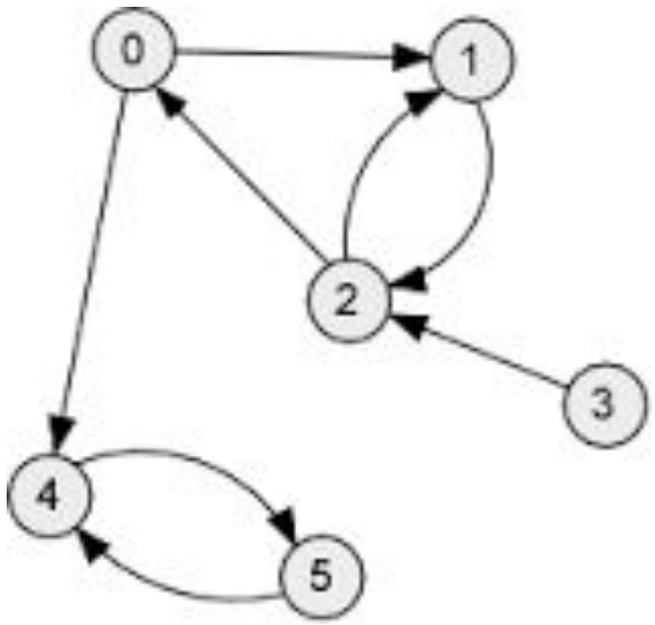
Q: Can you think of some applications where we need edge weights?

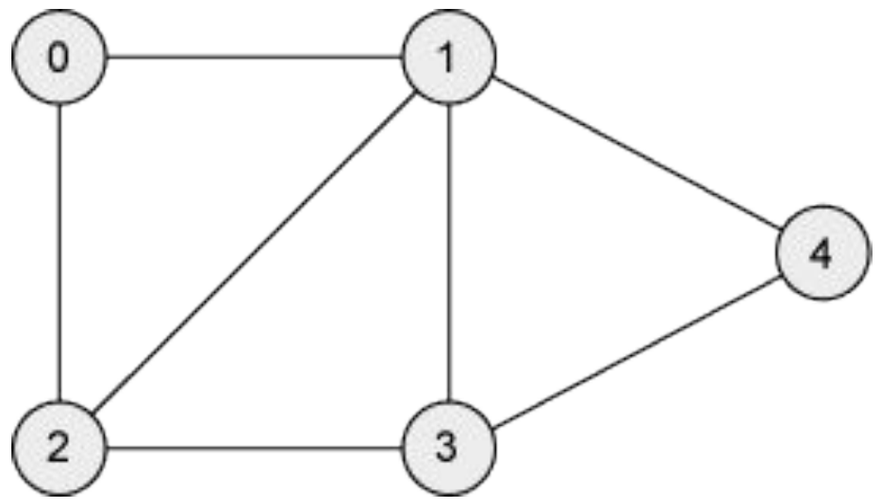




Graph Theory

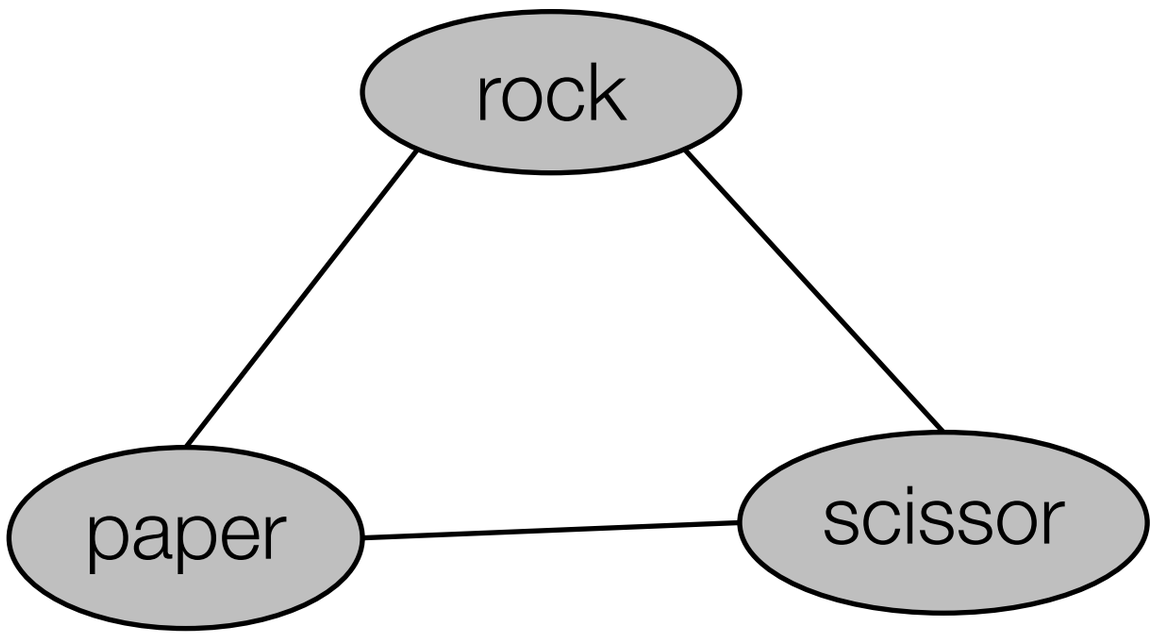
- Edges can be **directed** or **undirected**.
- Directions allow us to have influence go one way (rather than mutual).
  - The type of graph we use is dependent on

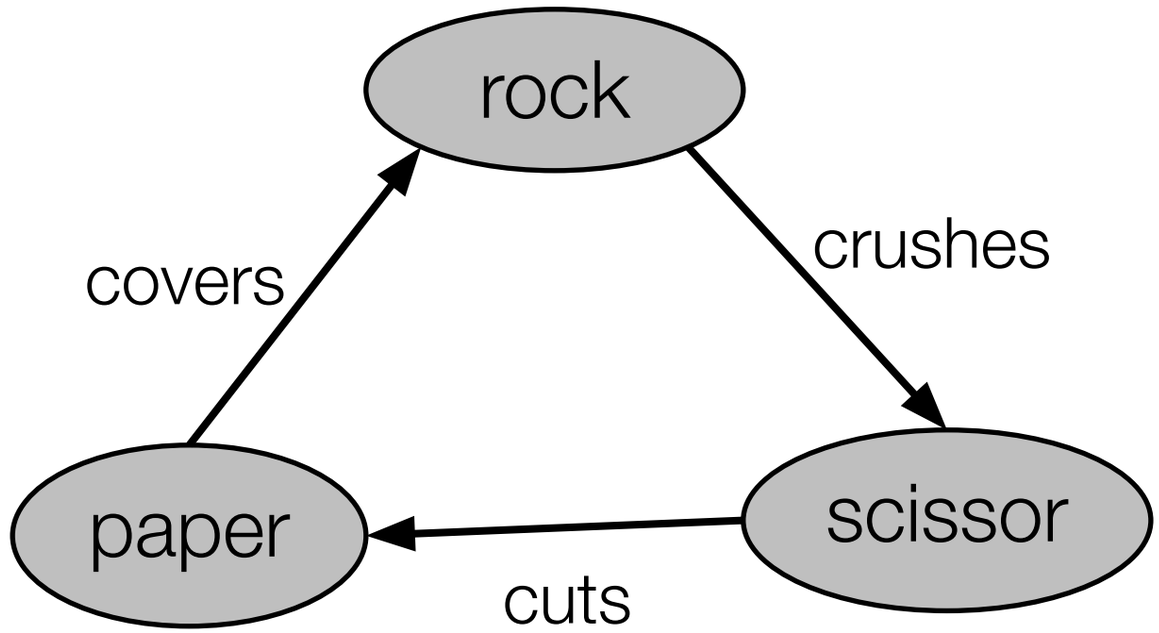




Example of directed edges in a graph

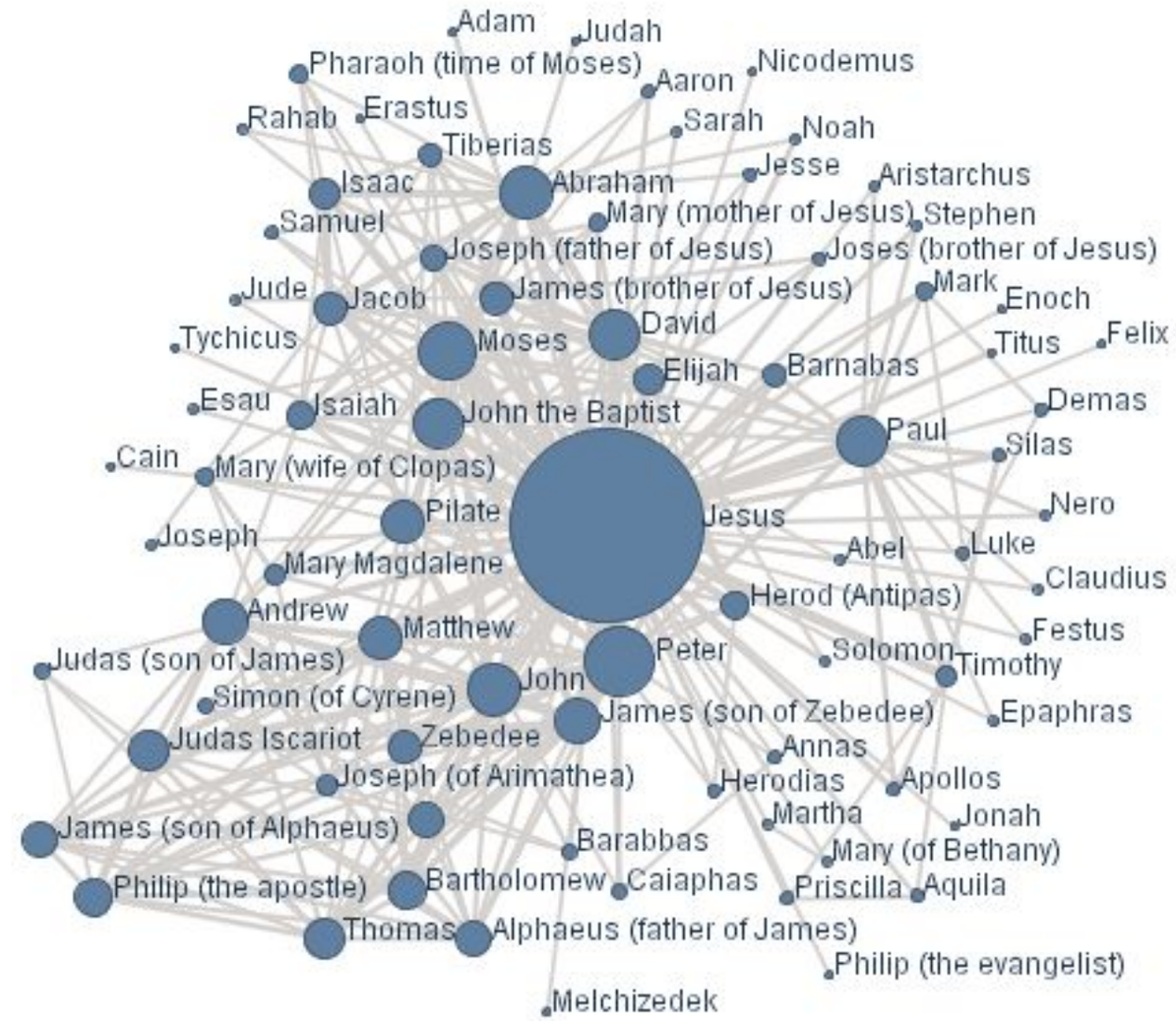
- Rock – Paper – Scissor Game





Graph Theory

- Q: Why do we care about graphs?
- A: We can use them to represent a social network:



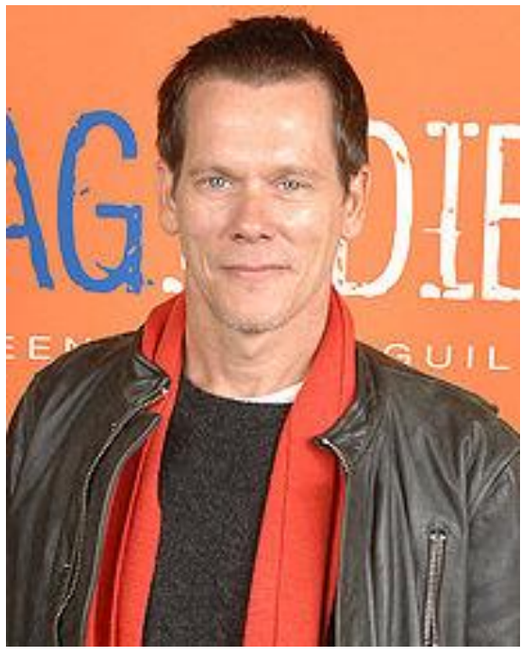
Acting Networks

- A good example of a network is formed by our coworkers, their coworkers, etc…
- We'll look at Hollywood actors.
- We'll say that two actors are connected if they played in the same film.

Kevin Bacon

- Q: Who is the “center” of the movie universe?

- A: Kevin Bacon. Have you heard of him? If not – it’s because he’s known for playing a supporting role in a huge number of films:



Kevin Bacon

Kevin Bacon

- We can play a game called ‘Six Degrees of Kevin Bacon.’

- The rules:
  - Pick an actor or an actress.
  - If they have played in a film with Kevin Bacon, their Bacon number is 1.
  - If they have played in a film with someone with a Bacon number of 1, they have a Bacon number of 2.
  - Etc…

- Example: Bradley Cooper:
  - Bradley Cooper was in “Silver Linings Playbook” (2012) with Jennifer Lawrence.
  - Jennifer Lawrence was in “X-Men: First Class” (2011) with Kevin Bacon.

  - So Bradley Cooper has a Bacon number of 2.

“Six Degrees” Game

- **Exercise:**
- Using IMDB, find the Bacon number (and path) for:
  - Julia Roberts
  - Selena Gomez
  - Christian Bale

“Six Degrees” Game

- It’s a lot easier with an automated tool that searches IMDB:

http://oracleofbacon.org/cgi-bin/movielinks

- For example, it produces this output for Christian Bale:

“Six Degrees” Game

- **Exercise:** Try out the Oracle tool with your favorite actors.
- What’s the largest Bacon number you can get?

“Six Degrees” Game

- We saw a lot of actors with Bacon numbers 1 and 2.
- What about the extremes?
  - What’s the smallest possible Bacon number? Who has it?
  - What’s the largest possible Bacon number?
    - What if someone is not connected to Kevin Bacon at all?
    - Among those who are, what’s the largest Bacon number?

“Six Degrees” Theorem

- The “Six Degrees” idea states that any two people selected at random are connected by at most 6 degrees.
- Of course, there are some counterexamples, but they’re rare.
- Same applies to Hollywood – which is why the game is called “Six Degrees of Kevin Bacon.”

Back to Kevin Bacon

- Here's a statistic about Kevin Bacon numbers for other actors:

Bacon Number	# of People
0	1
1	3434
2	398768
3	1478688
4	382332
5	33473
6	4101
7	679
8	144
9	16
10	7

Back to Kevin Bacon

- What can we see from the table?
- Most people have a Bacon number of 3.
- Slightly less have a Bacon number of 2 and 4.
- Bacon numbers higher than 4 are rare.
- The maximum is 10.

Back to Kevin Bacon

- You can replace Bacon with other actors to see who makes a good “center” for connections.

- Try it out:

http://oracleofbacon.org/cgi-bin/center-cgi

Back to Kevin Bacon

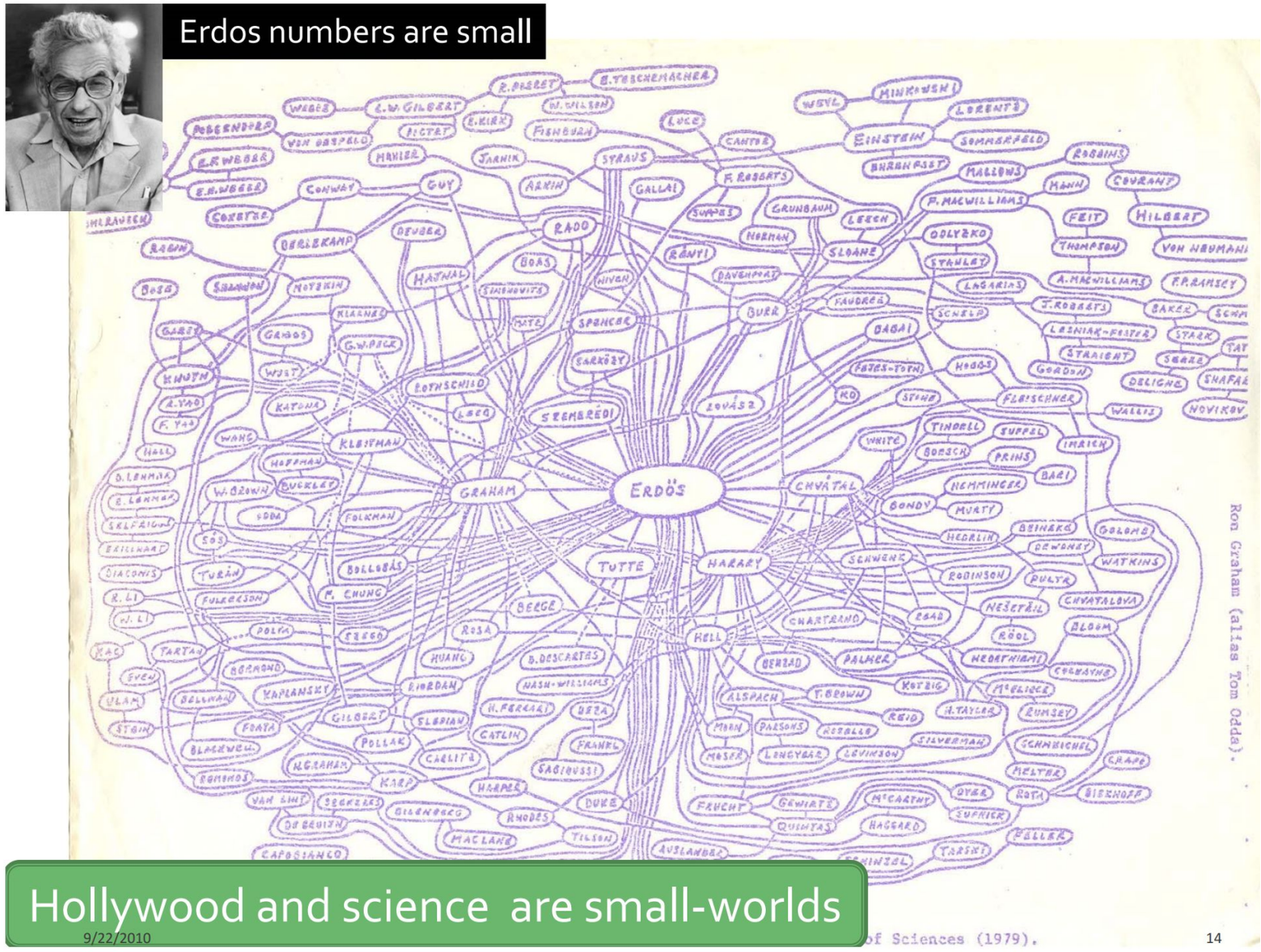
- What does it mean to be a good “center?”
- For example, we see that Tom Cruise has an average number of 2.976 while Bacon has an average of 3.026. Who’s better?
- **Exercise:** Try out your favorite actor. Who picked the best center?

Rachel McAdams Number	# of People
0	1
1	1897
2	300058
3	1505279
4	451047
5	38285
6	4249
7	641
8	163
9	16
10	7

Total number of linkable actors: 2301643
Weighted total of linkable actors: 7144962
Average Rachel McAdams number: 3.104

What about the world of science?

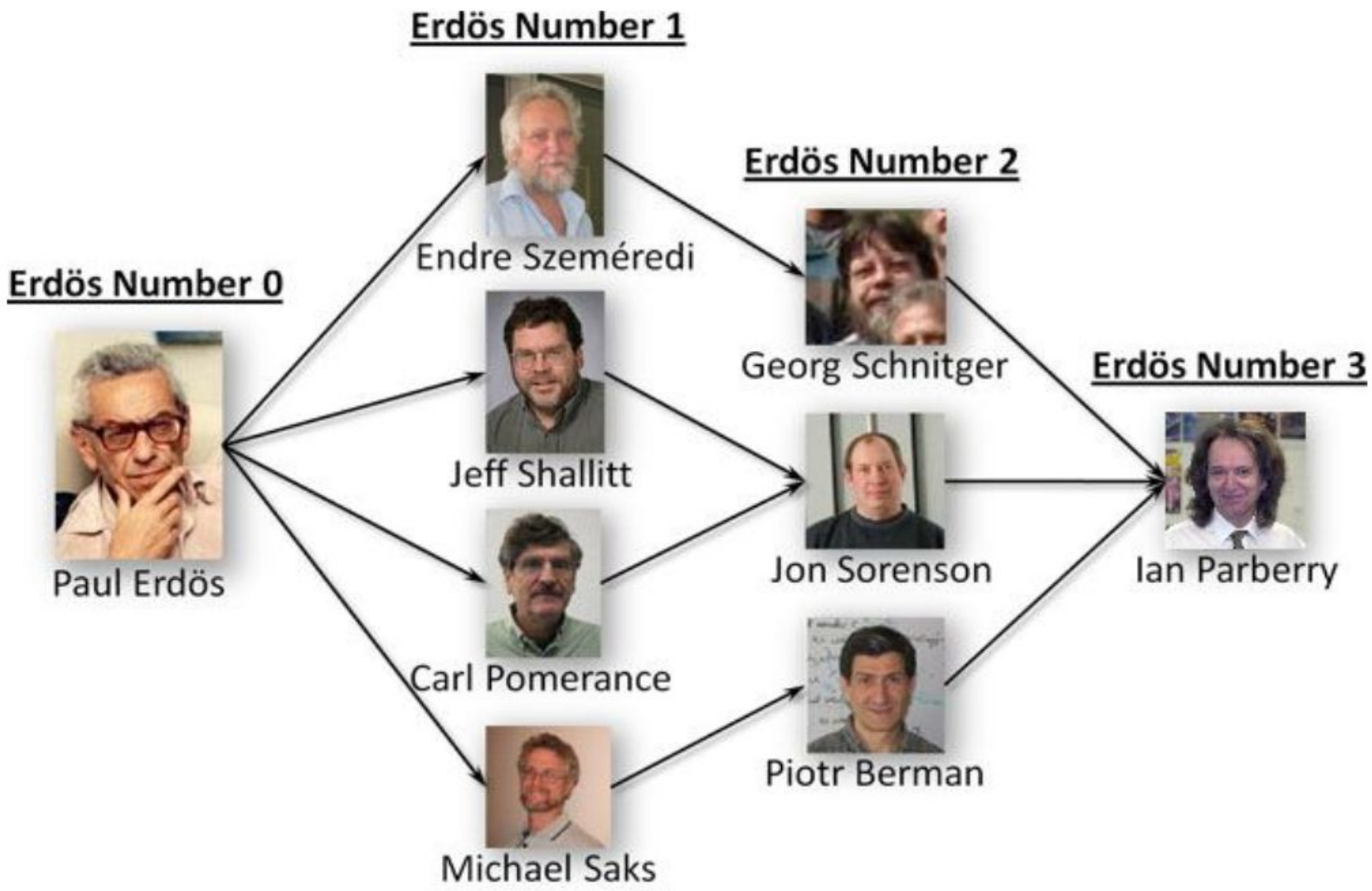
- Erdős numbers:



Hollywood and science are small-worlds

Erdős numbers

- Describes the "collaborative distance" between mathematician Paul **Erdős** and another person, as measured by authorship of mathematical papers.



What’s your Erdos number?

- Roshni Iyer (instructor of this module): 2

Paul Erdős

Satish Rao

Roshni Iyer

Compute:
https://oakland.edu/enp/compute/

Larger networks

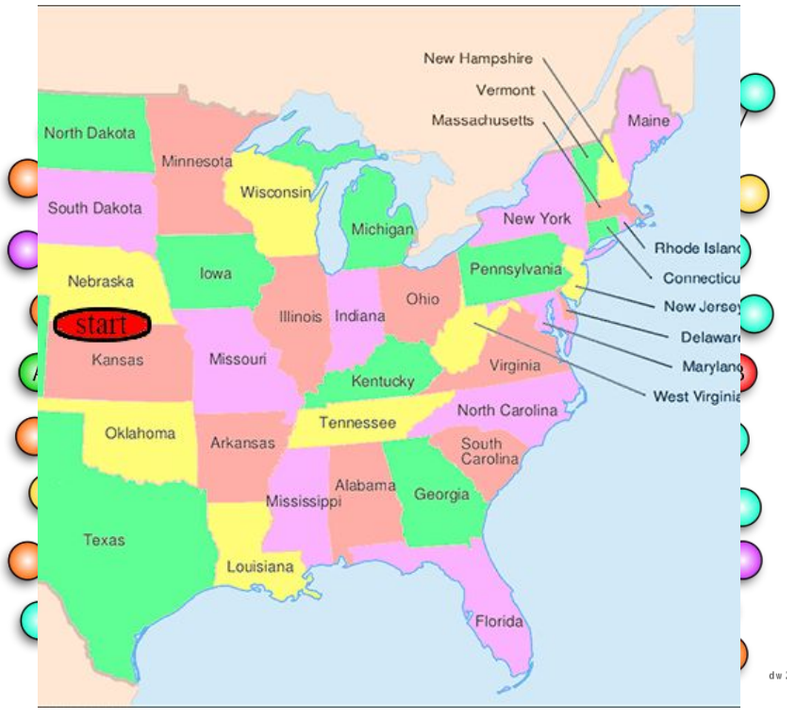
- We saw, from the Kevin Bacon game, that
  - 1. Hollywood has about 1 million actors.
  - 2. Most of them can be related to each other within 4 connections.

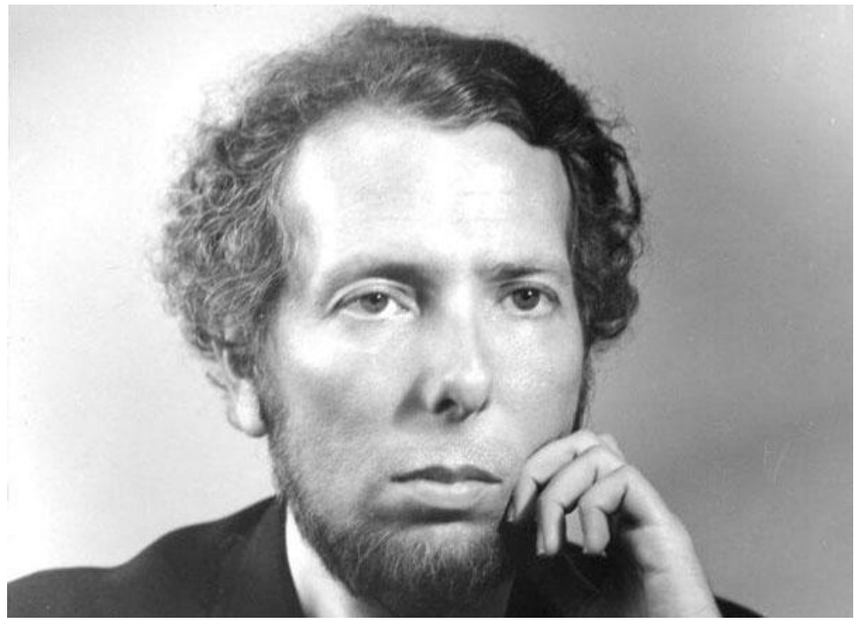
- What about all the people in the US?
  - Population is about 300 million.

- Q: About how many connections on average between two random people?
  - (Note: define connection here as “being acquainted with.”)

Larger Networks

- "Six Degrees" Theorem still applies!
- So, if we pick two random people in the US, using a chain of "friends of friends", we should be able to connect them within 6 connections.
- Studied by the famous psychologist Stanley Milgram in his "Small world experiment" in the 1960's.
- Easily recreated today using Facebook!





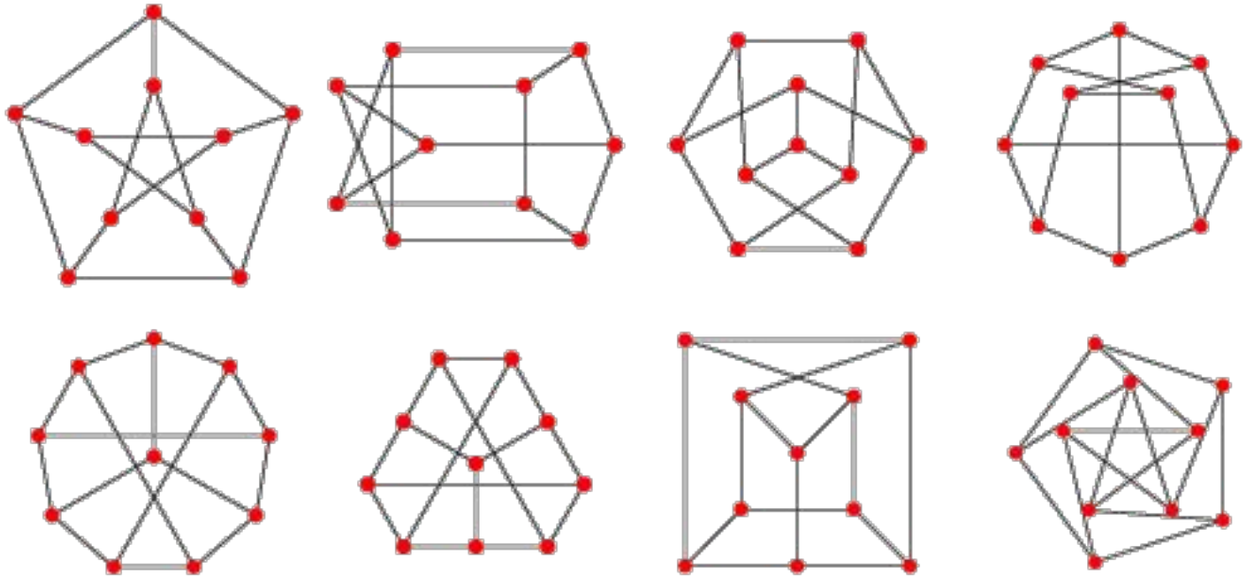
Stanley Milgram

“Six degrees” Theorem

- So what’s useful about it? Applications?
- One example: Insurance companies request customers to provide 5 referrals in order to expand their business.
- **Exercise:** Come up with another example.

Graph Theory

- We've talked about networks, connections, "distances", etc…
- Now it's time to describe the math behind this.
- This domain is called "graph theory."

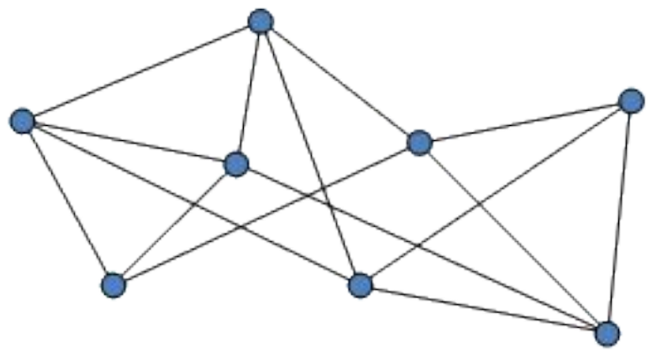


Graphs

- A graph is made up of “nodes” and “edges.”

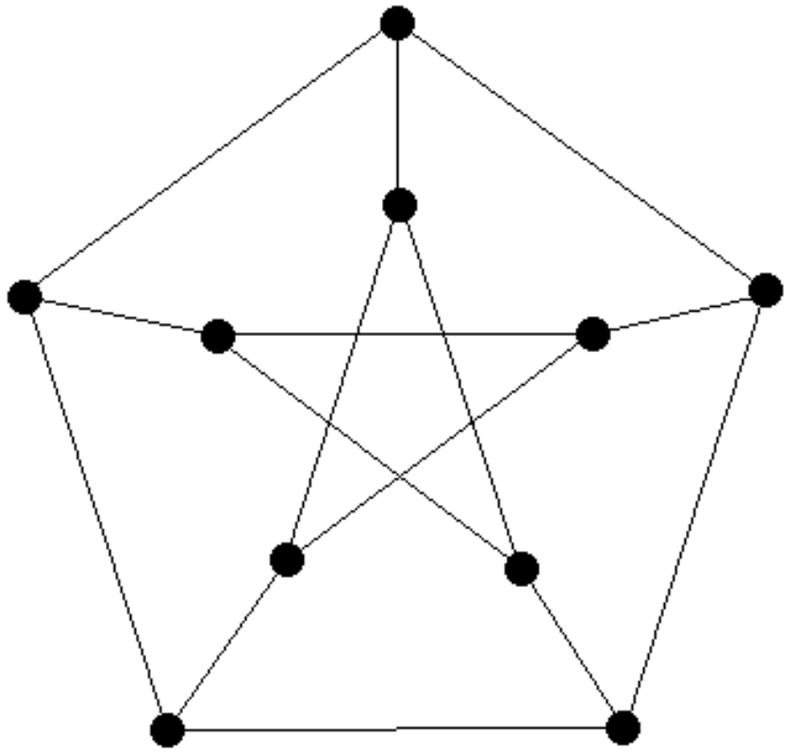
- Nodes are denoted by a dot/vertex.
- Edges connect two nodes.

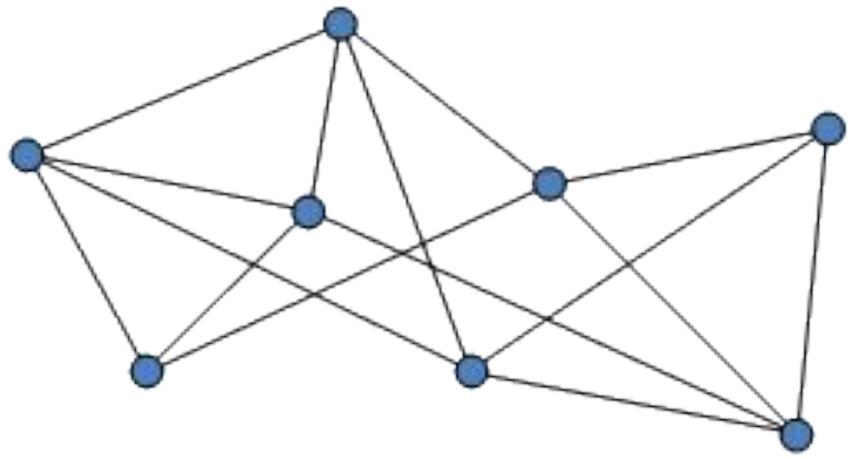
- If two nodes have an edge between them, they are “adjacent” or “neighbors.”



Degrees

- The number of edges coming out of a node is called its “degree.”
  - Sometimes degree is specified to be ***out-degree*** (outgoing edges) or ***in-degree*** (incoming edges).
- What are the in-degrees & out-degrees of the nodes in the graph below?
- All nodes have same degree: **regular** graph. If not, **irregular**.
- Q: Are complete graphs (maximally connected graphs) regular or irregular?

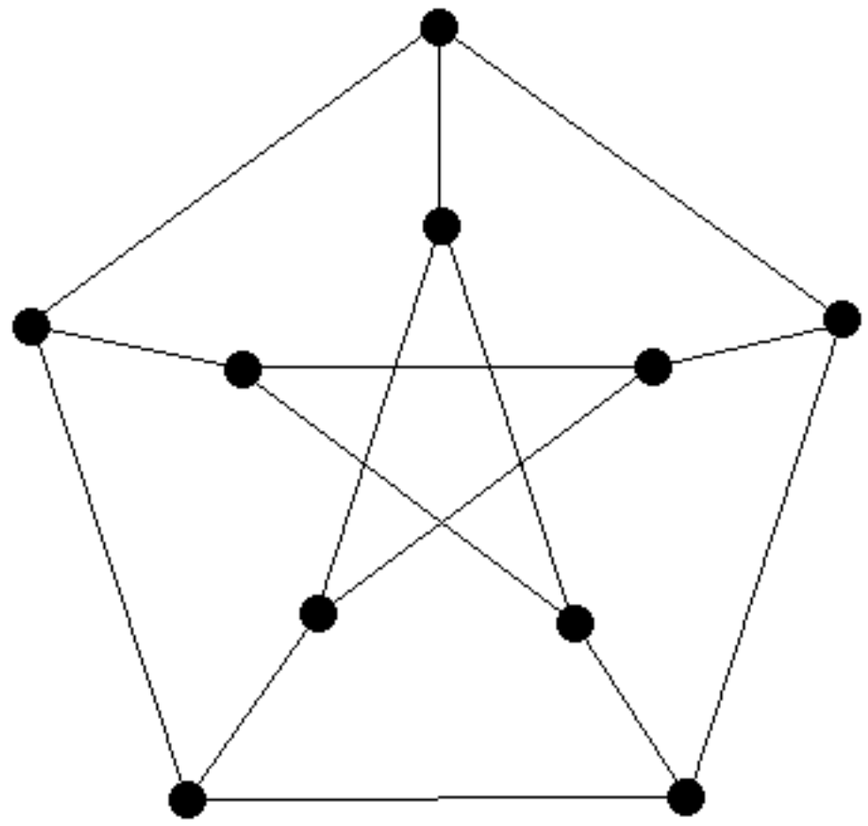




Degrees

- What's the relationship between **vertex degrees** and **edges**?

- **Exercise**: Give the relationship described above.
  (hint: think about the sum of all the degrees and the number of edges.)



Idea

- Enumerate ordered pairs (i,j) in the graph, where i and j is connected by an edge.

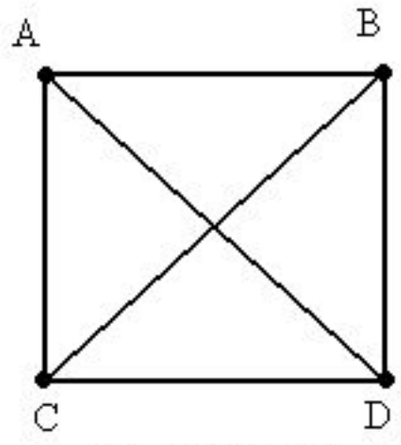
- Count by vertices:
  - If node i has degree di, then there are di ordered pairs of of (i,j)
  - The number of ordered pairs=sum of degrees of all nodes

- Count by edges:
  - If (i,j) is an edge, then both (i,j) and (j,i) are ordered pairs
  - The number of ordered pairs=2 times number of edges

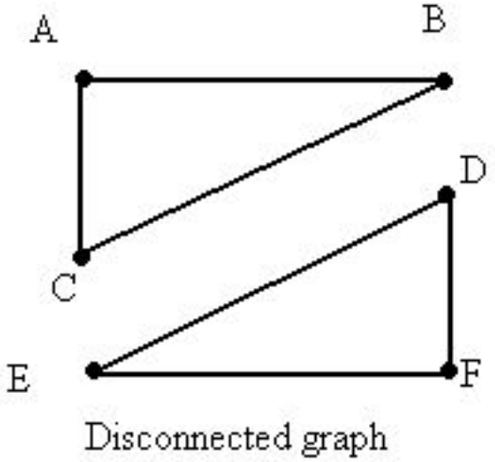
- Sum of degrees of all nodes=2 times number of edges

Connectivity

- A series of edges is called a “path.”
- If there exists a path between two nodes, the two nodes are “connected.”
- If every pair of nodes in a graph is connected, the entire graph is “connected”, otherwise it is “disconnected”.
  - A disconnected graph can be divided into multiple connected components
  - Connected graph has only 1 connected component

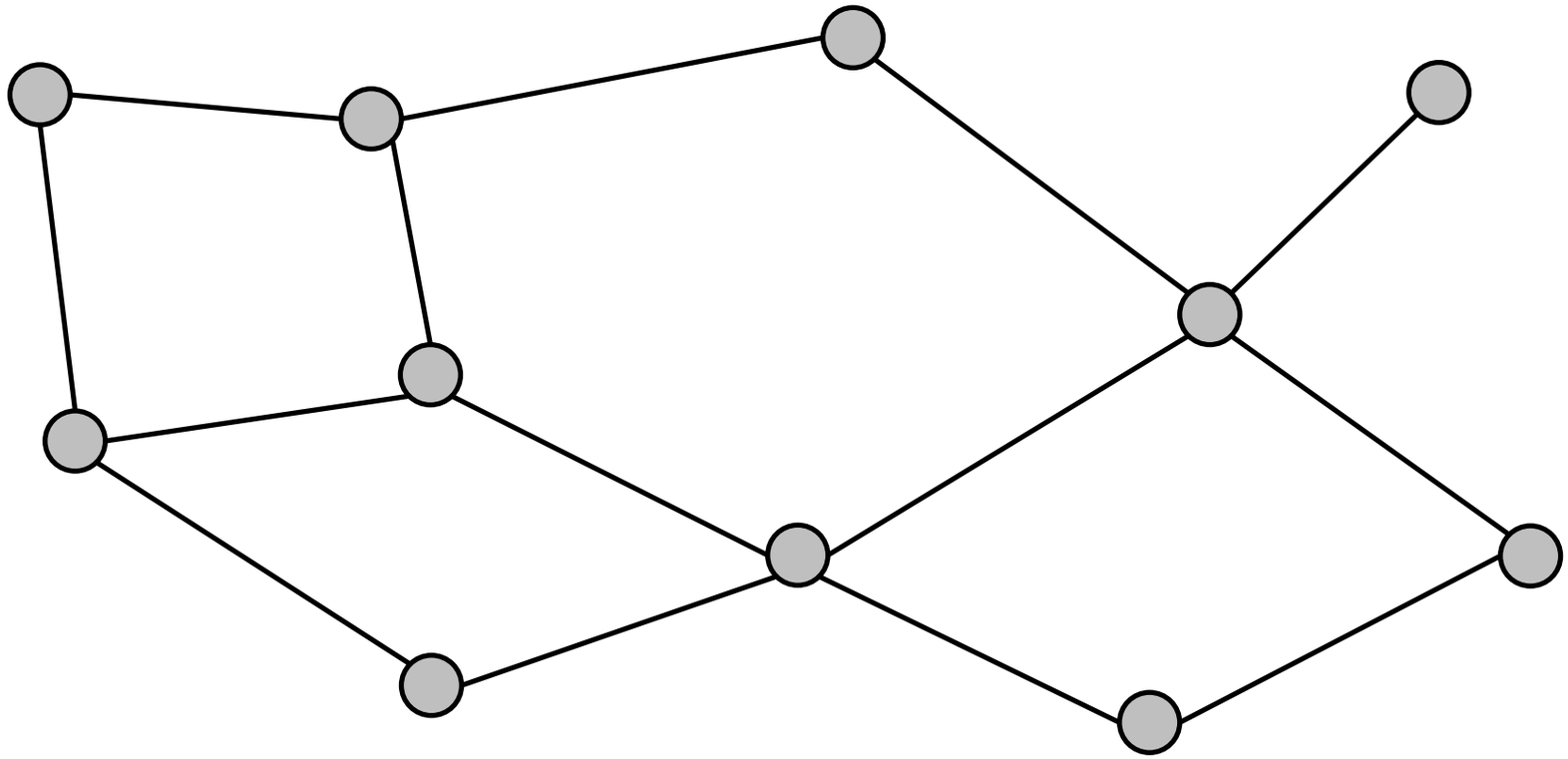


Connected graph

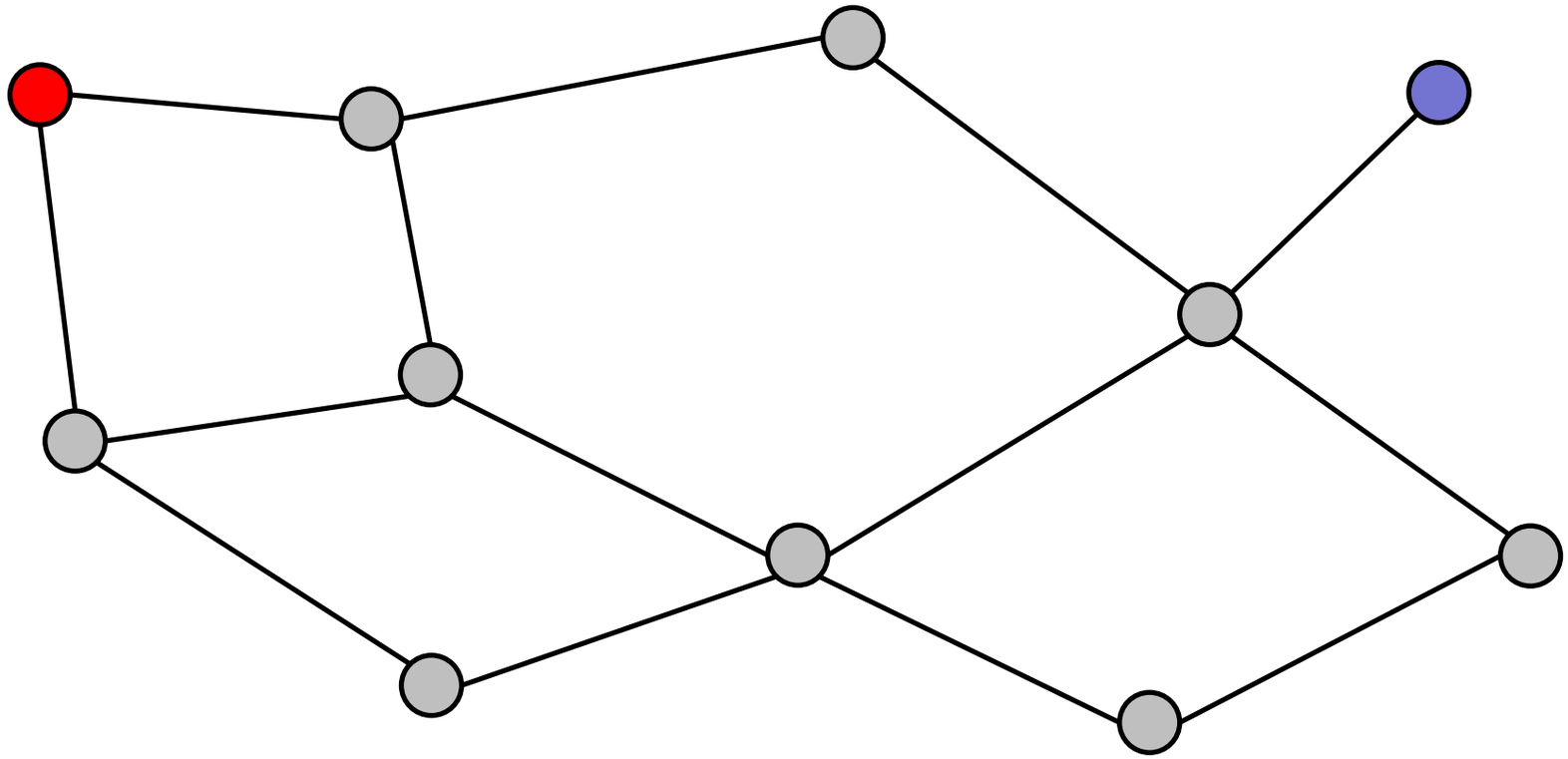


Disconnected graph

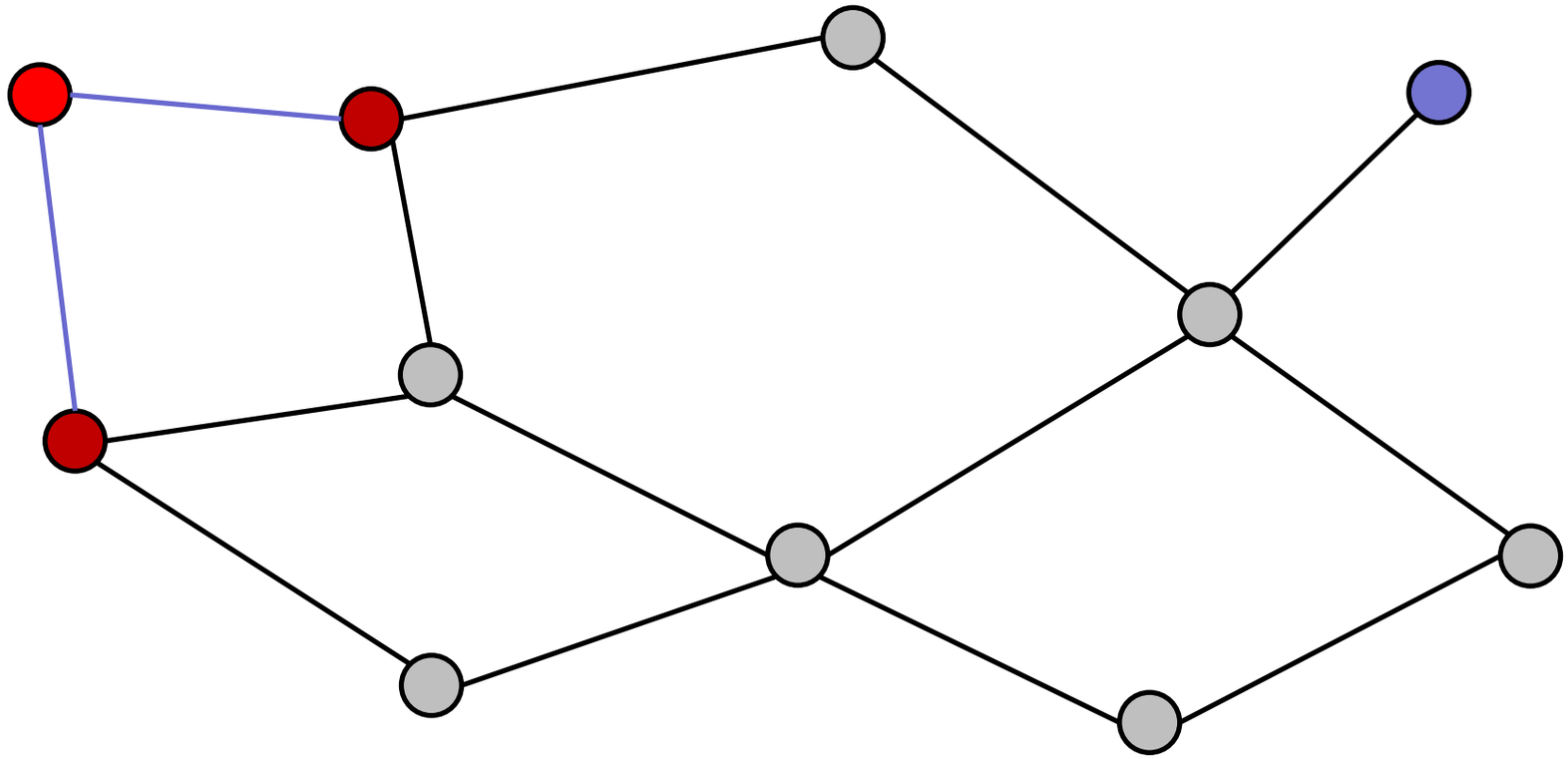
How to find the shortest path between two nodes?



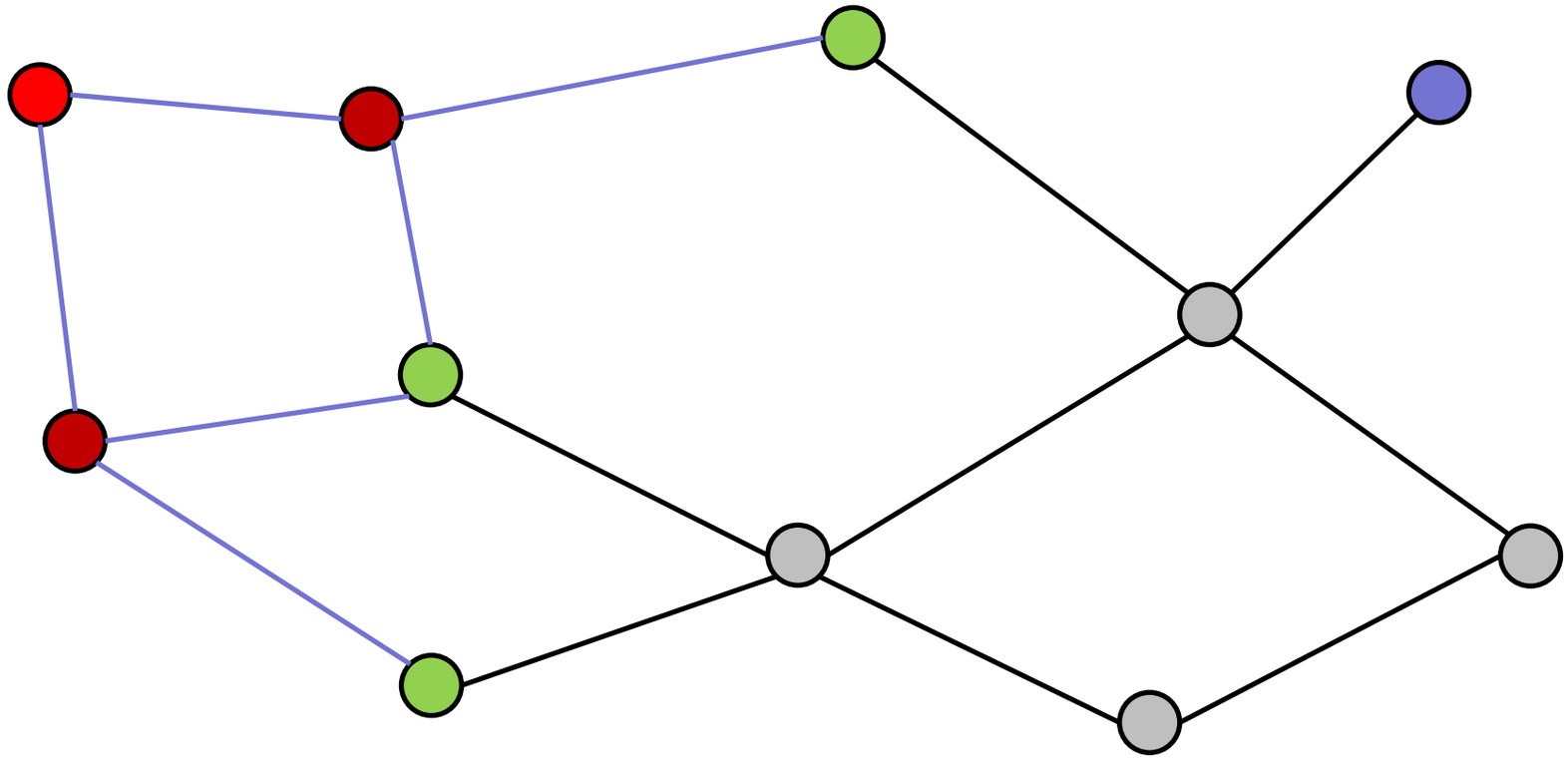
How to find the shortest path between two nodes?



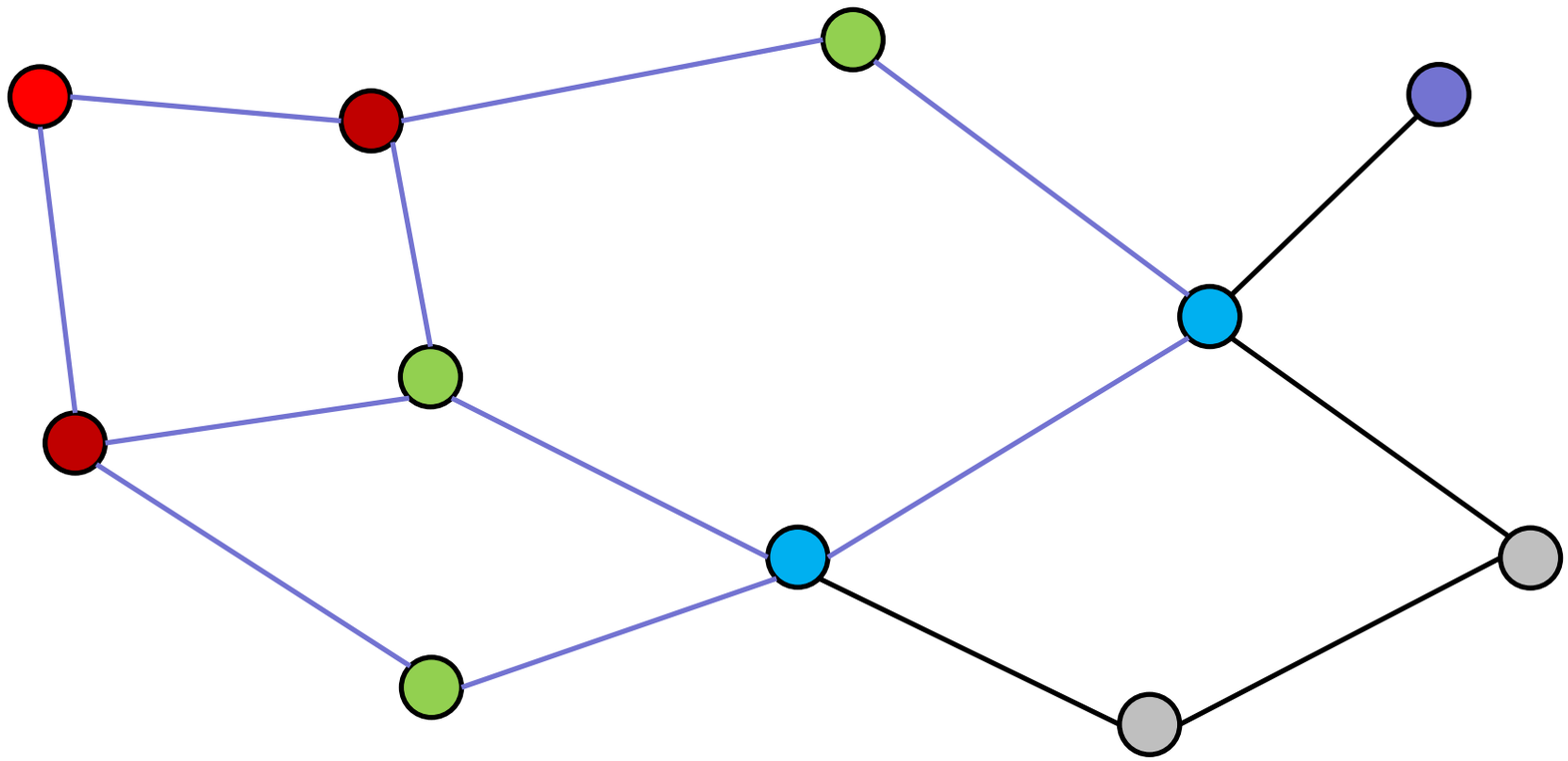
How to find the shortest path between two nodes?



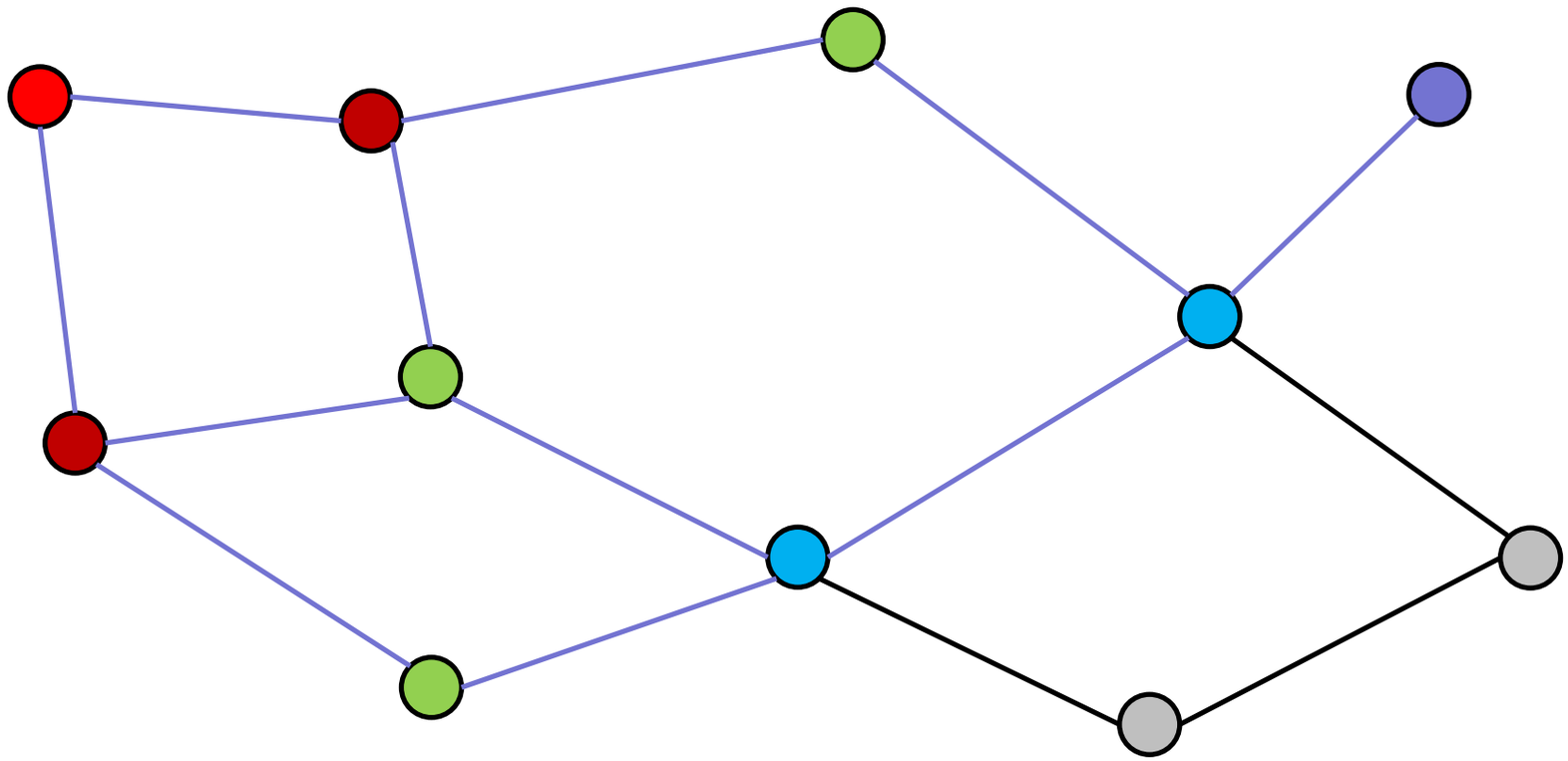
How to find the shortest path between two nodes?



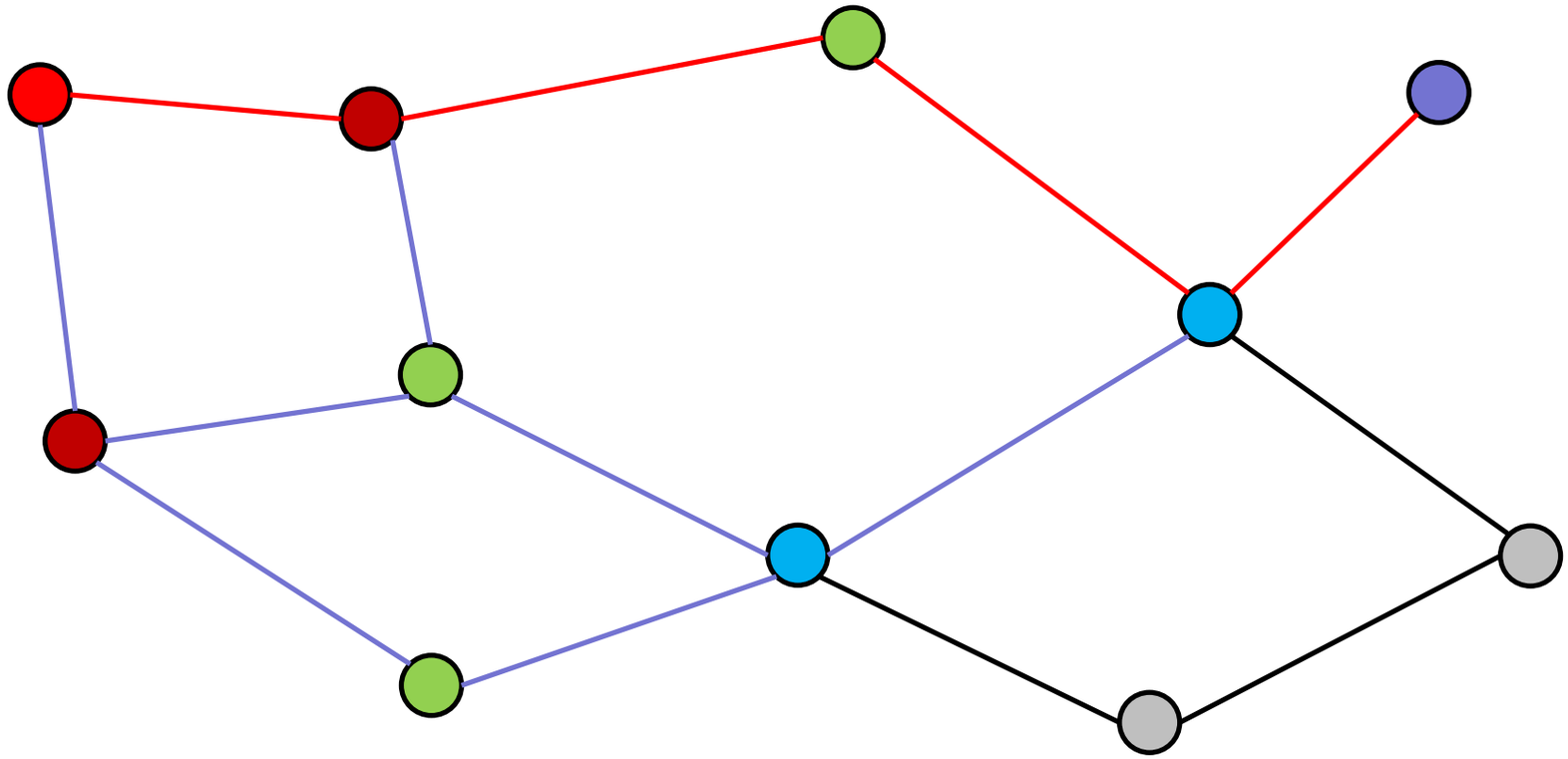
How to find the shortest path between two nodes?



How to find the shortest path between two nodes?

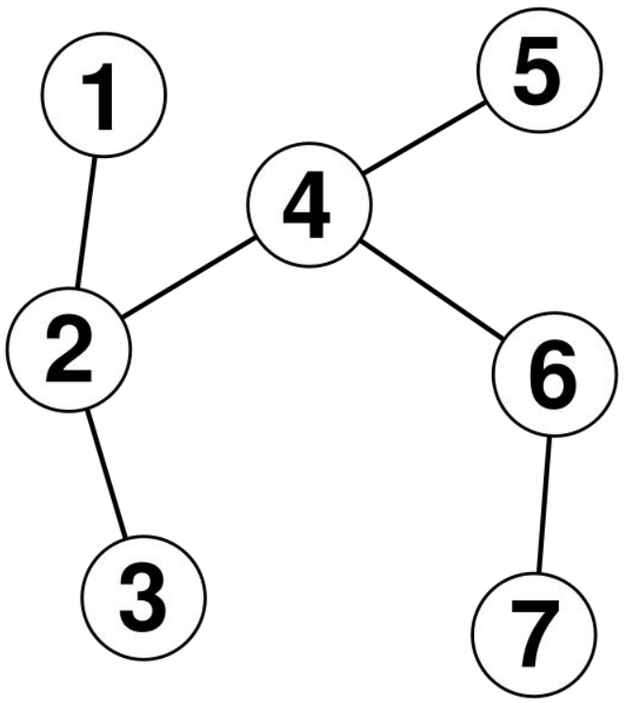


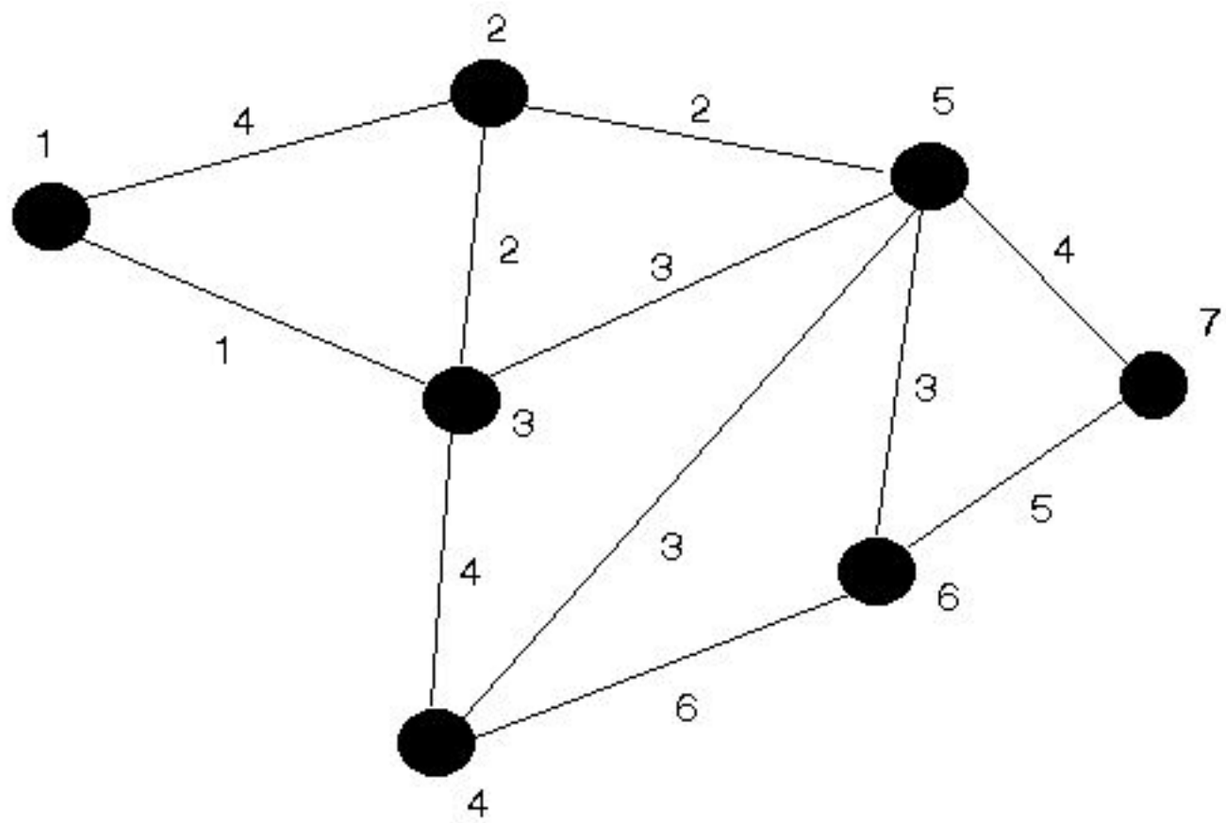
How to find the shortest path between two nodes?



Cycles

- A series of edges (a path) starting and ending in the same node is called a “cycle.”
- Q: Which graphs have cycles? If so, how many cycles are there?

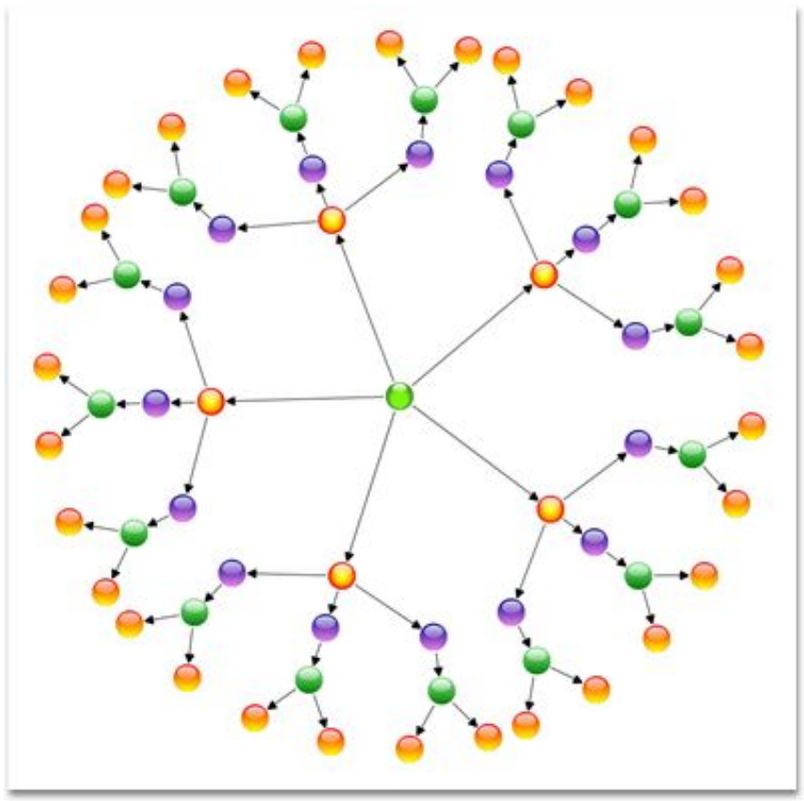




Trees

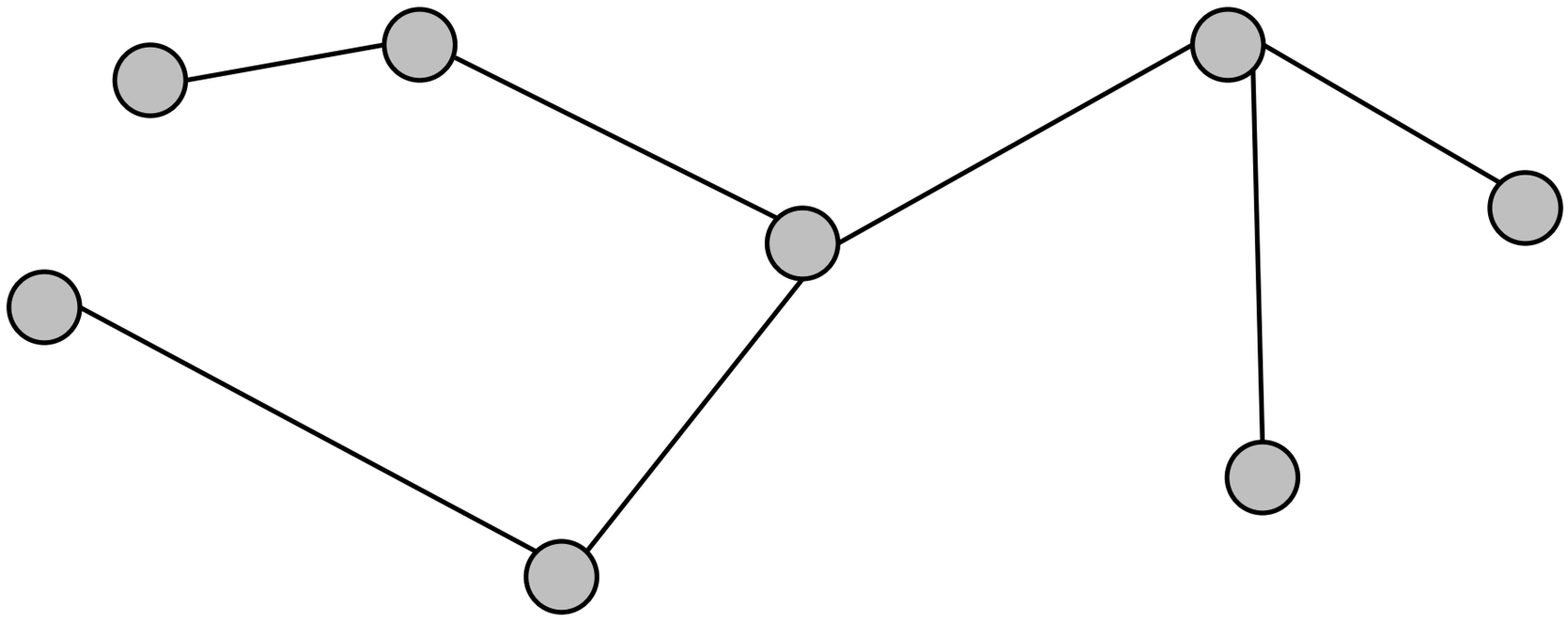
- A connected graph with no cycles is called a “tree”.
  - If we remove “connected” from this definition, the graph will be called a “forest”.

- **Exercise**: If we have a tree with n nodes, how many edges will it have?
  - (hint: draw some small examples of trees and see what you get.)



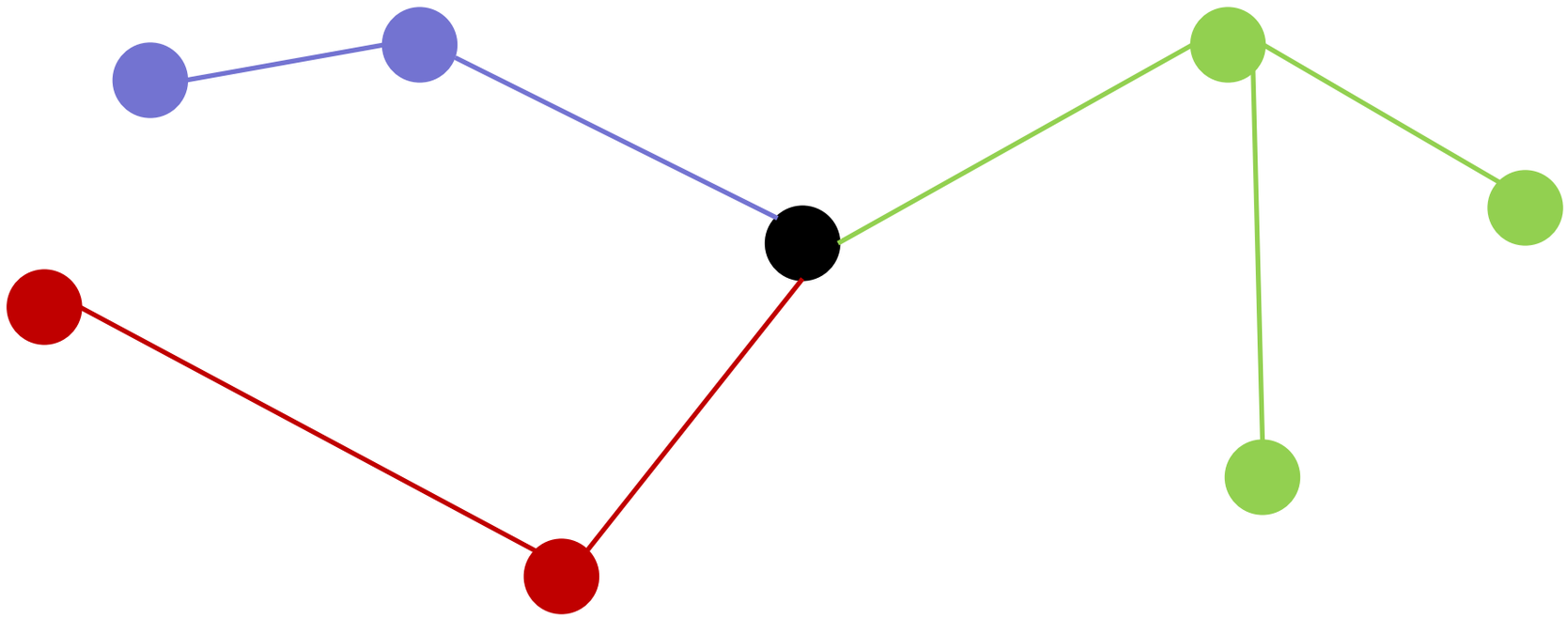
Idea

- Leaves in a tree: a node with degree 1 is called a leaf of the tree
  - Every tree has at least two leaves



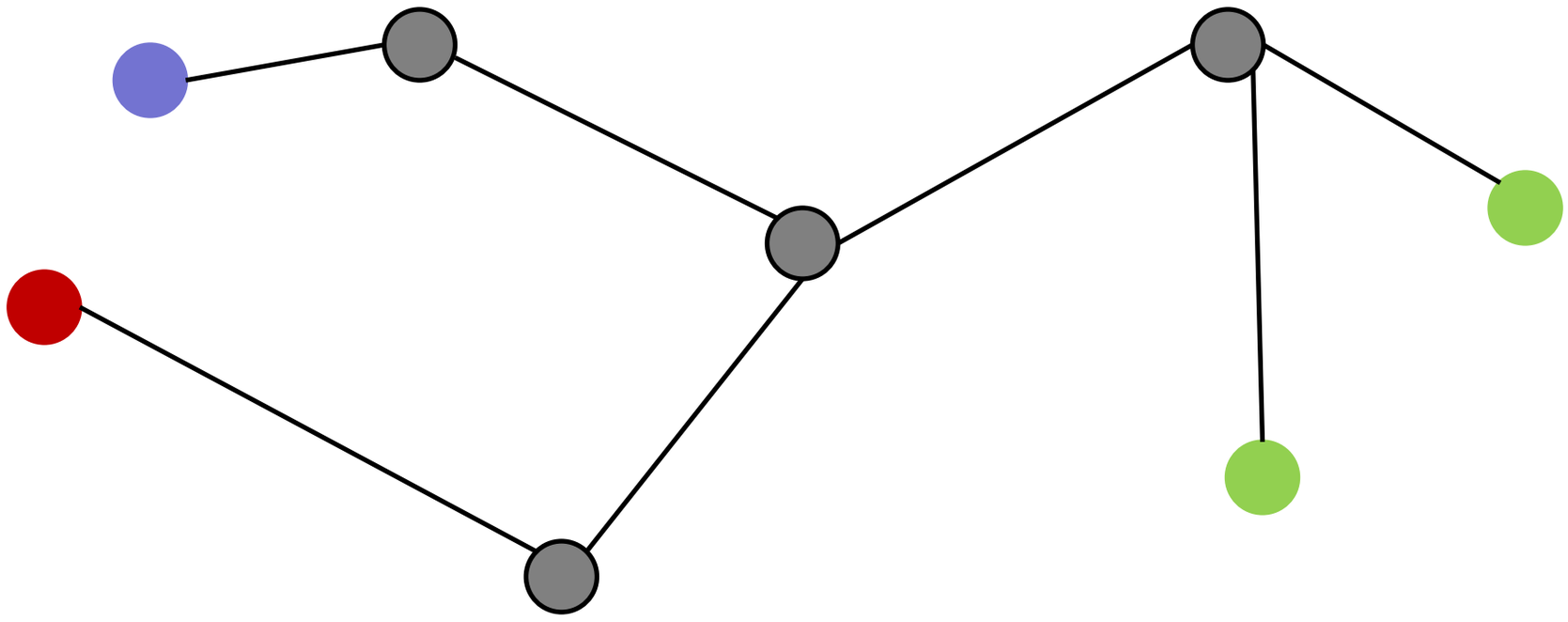
Idea

- Leaves in a tree: a node with degree 1 is called a leaf of the tree
  - Every tree has at least two leaves



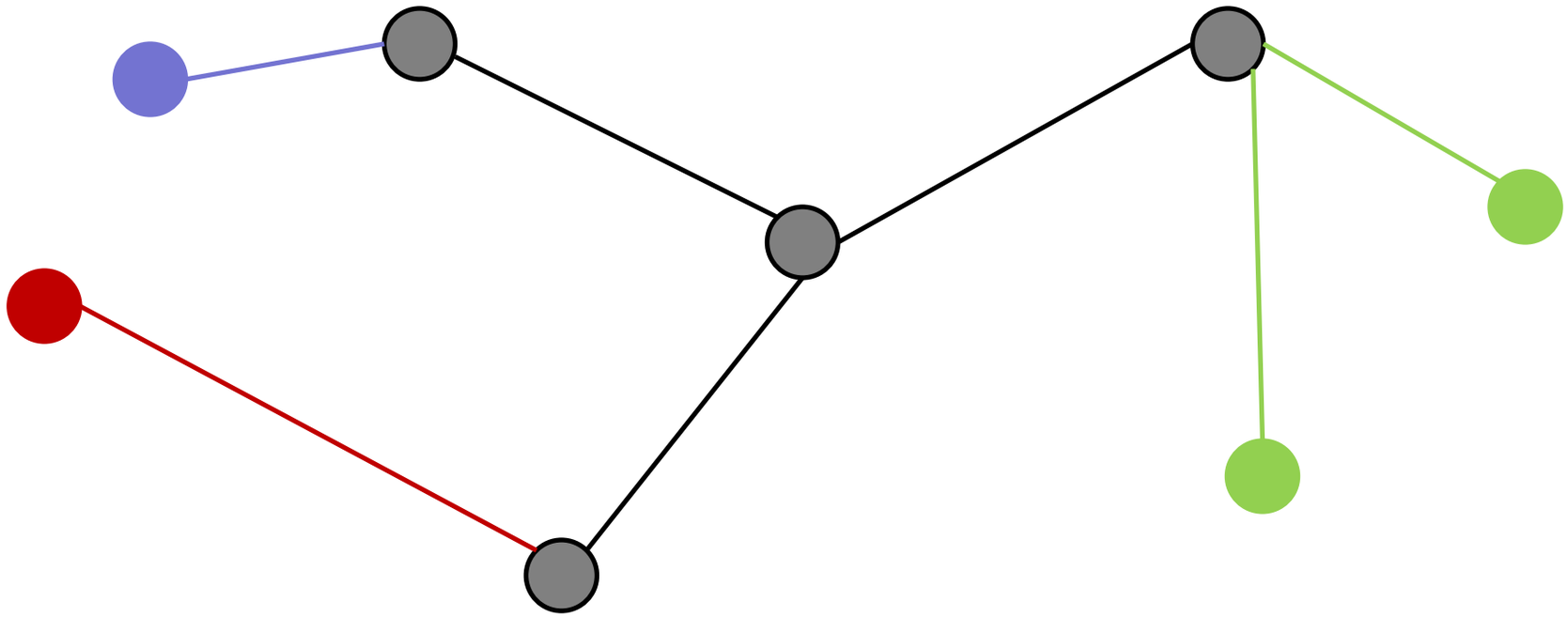
Idea

- Leaves in a tree: a node with degree 1 is called a leaf of the tree
  - Every tree has at least two leaves



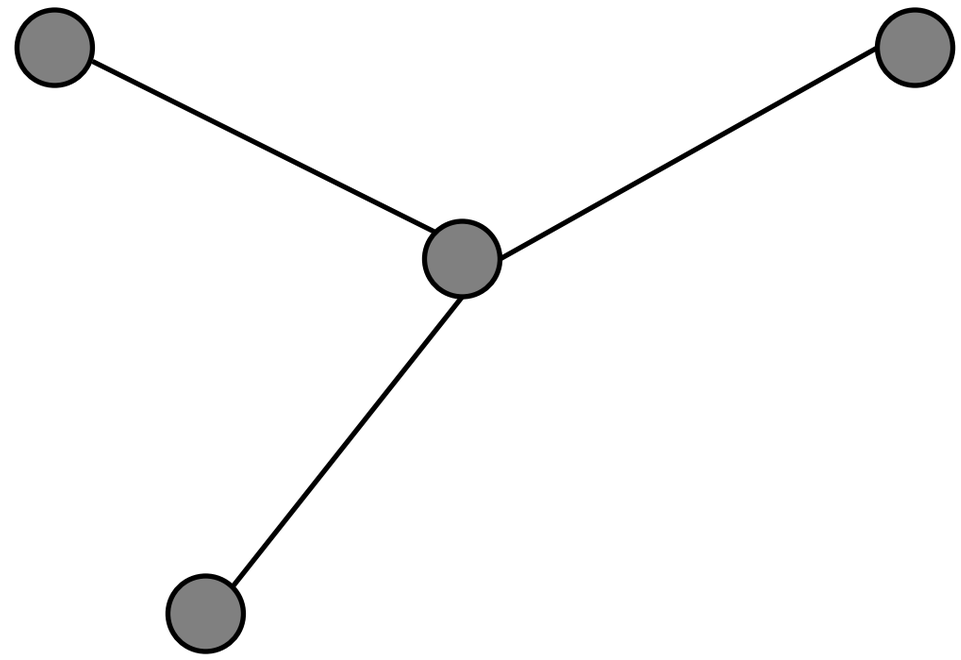
Idea

- Leaves in a tree: a node with degree 1 is called a leaf of the tree
  - Every tree has at least two leaves
  - Remove leaves and its connected edges will not change the difference between the number of nodes and the number of edges



Idea

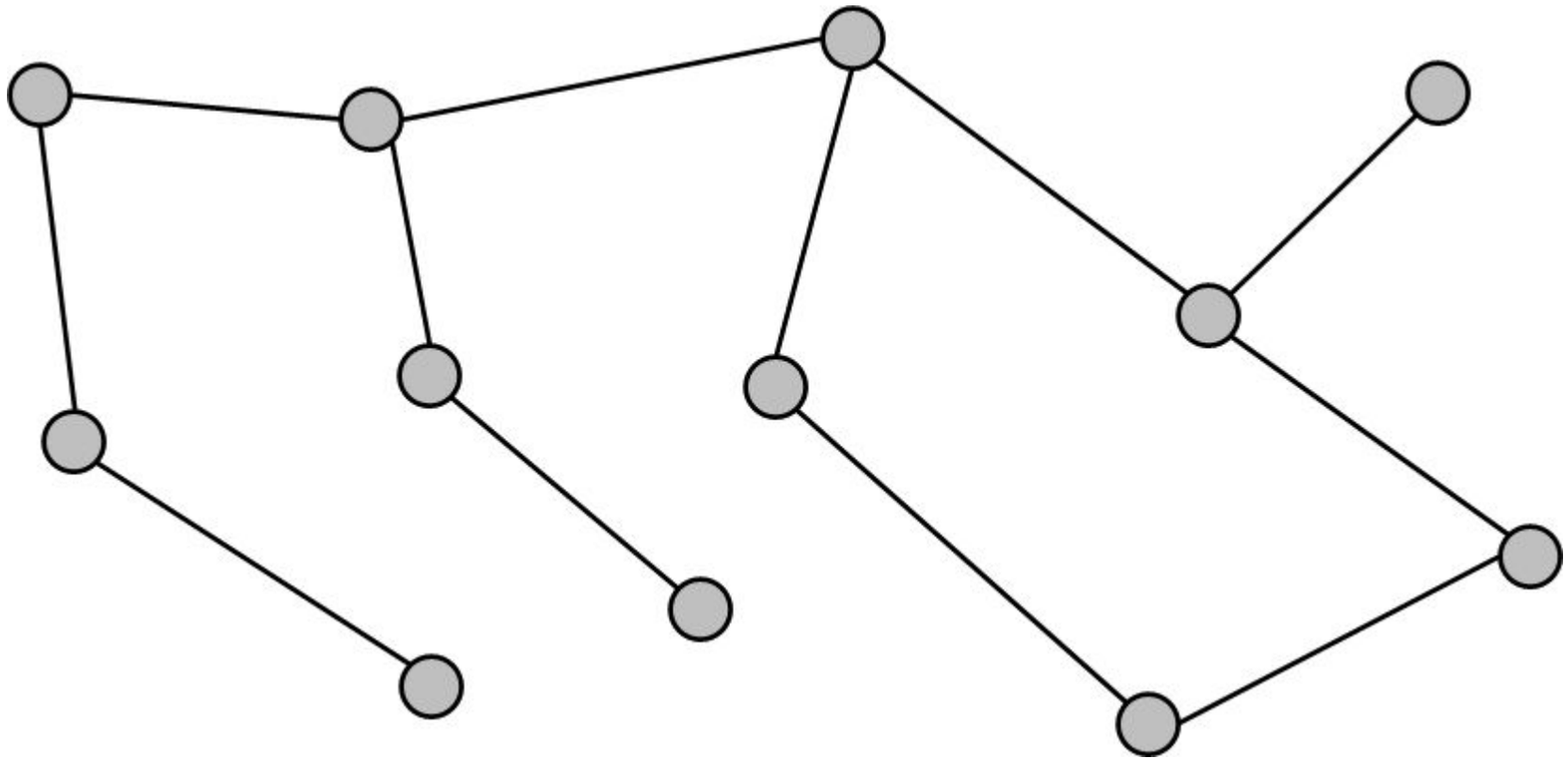
- Leaves in a tree: a node with degree 1 is called a leaf of the tree
  - Every tree has at least two leaves
  - Remove leaves and its connected edges will not change the difference between the number of nodes and the number of edges



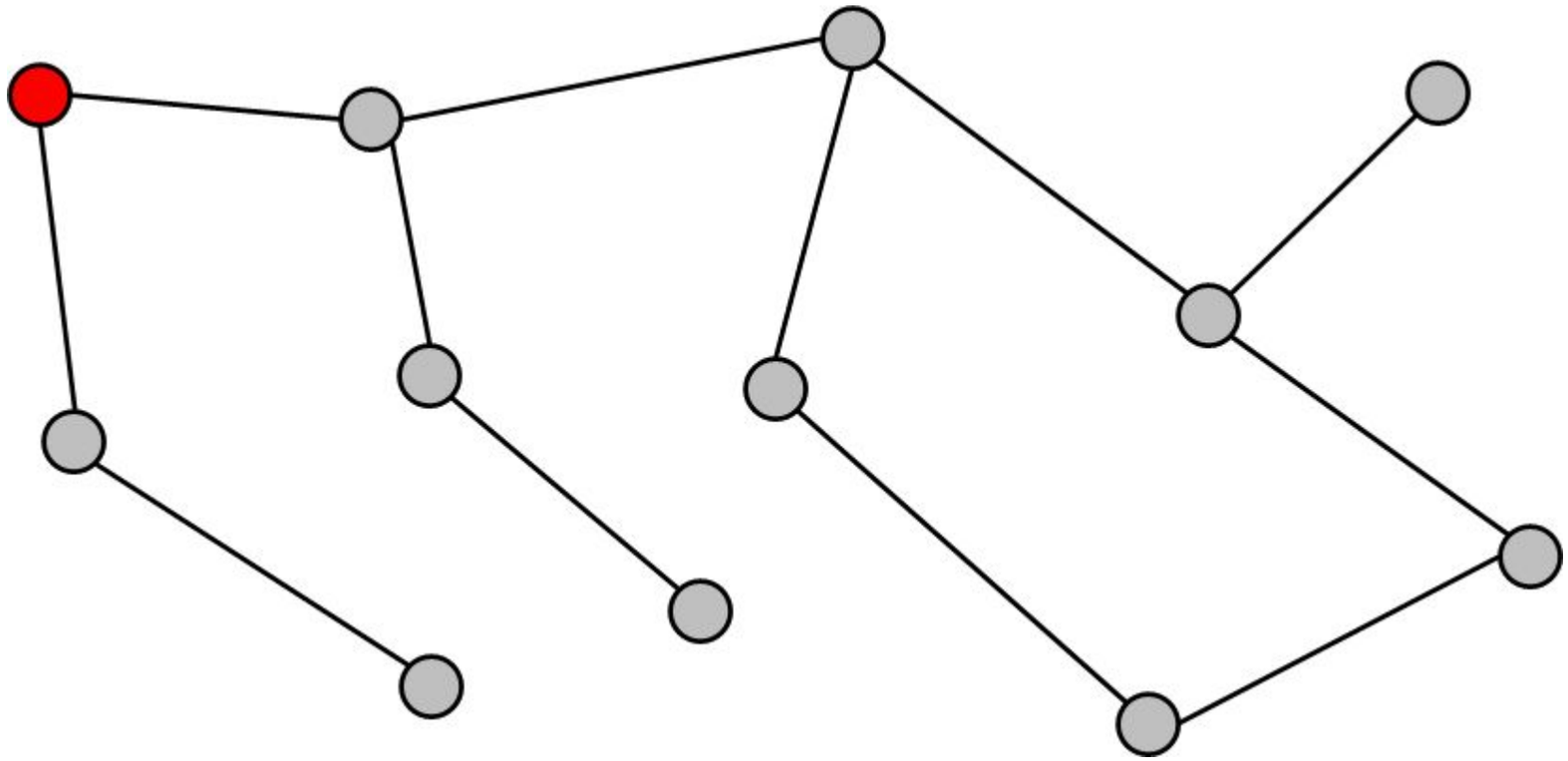
Idea

- Leaves in a tree: a node with degree 1 is called a leaf of the tree
  - Every tree has at least two leaves
  - Remove leaves and its connected edges will not change the difference between the number of nodes and the number of edges
  - Kept removing leaves from the tree, we finally obtain a graph of a single node
  - #nodes-#edges=1 in a tree

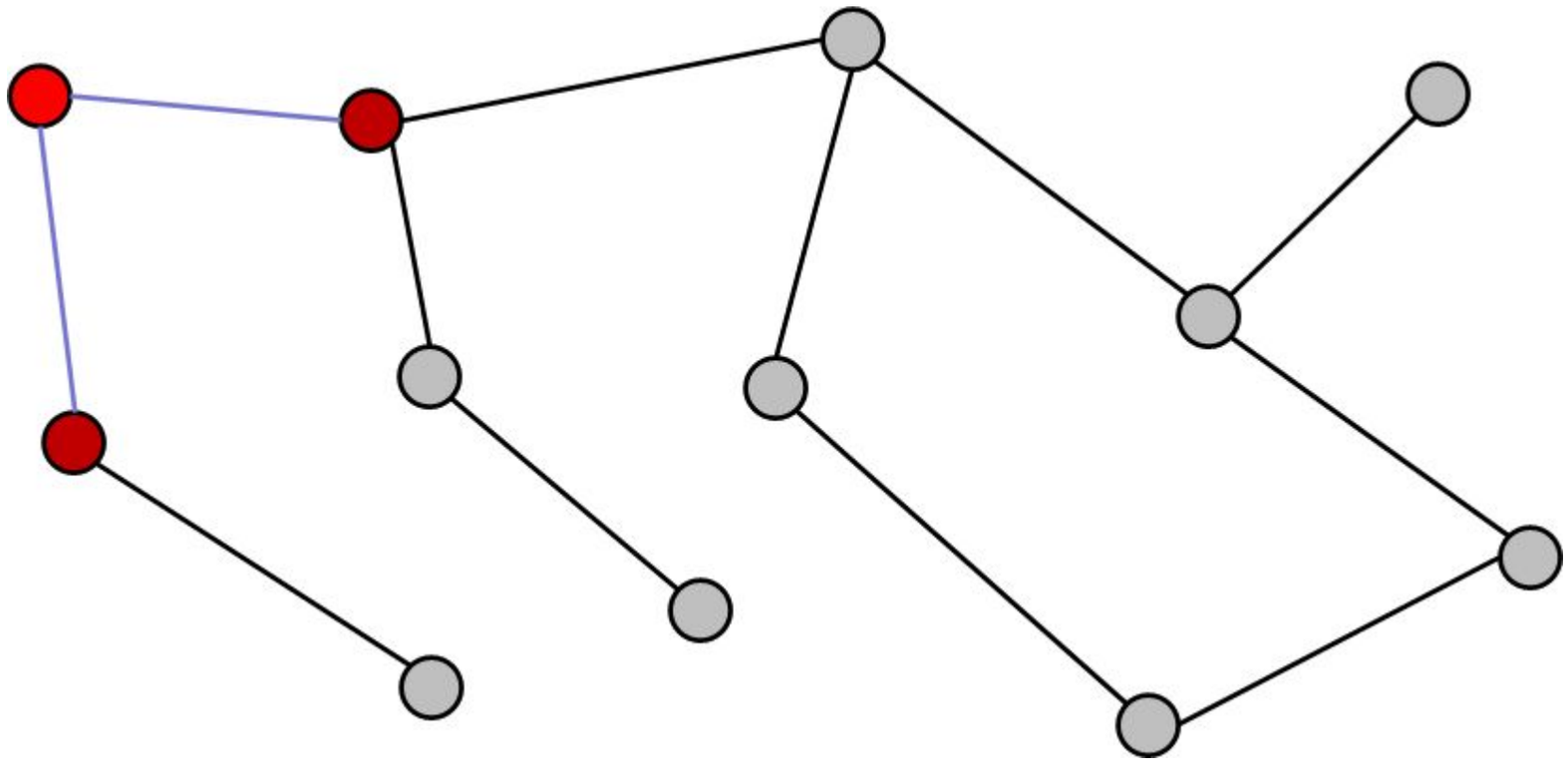
How to find a cycle in a graph?



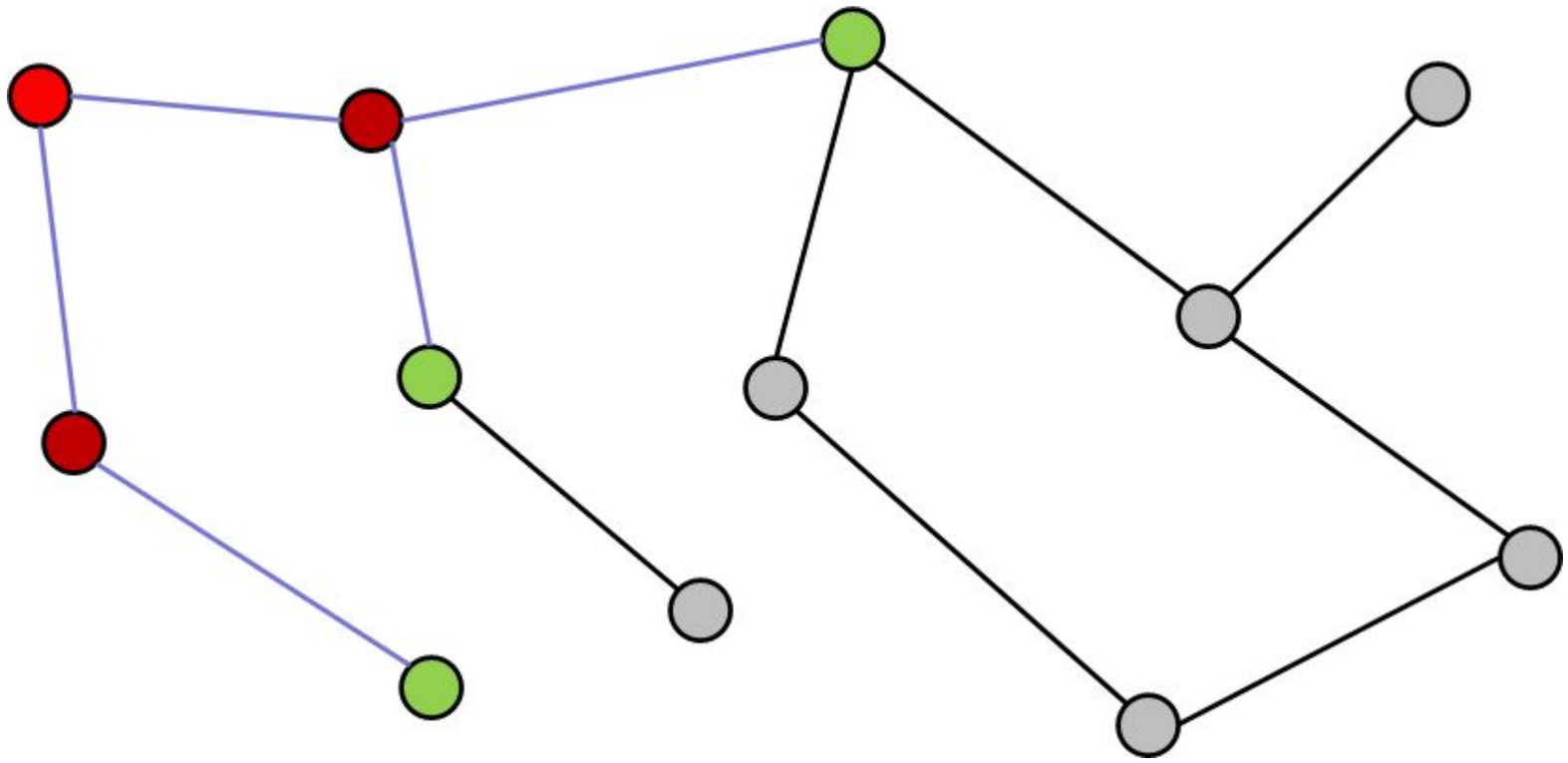
How to find a cycle in a graph?



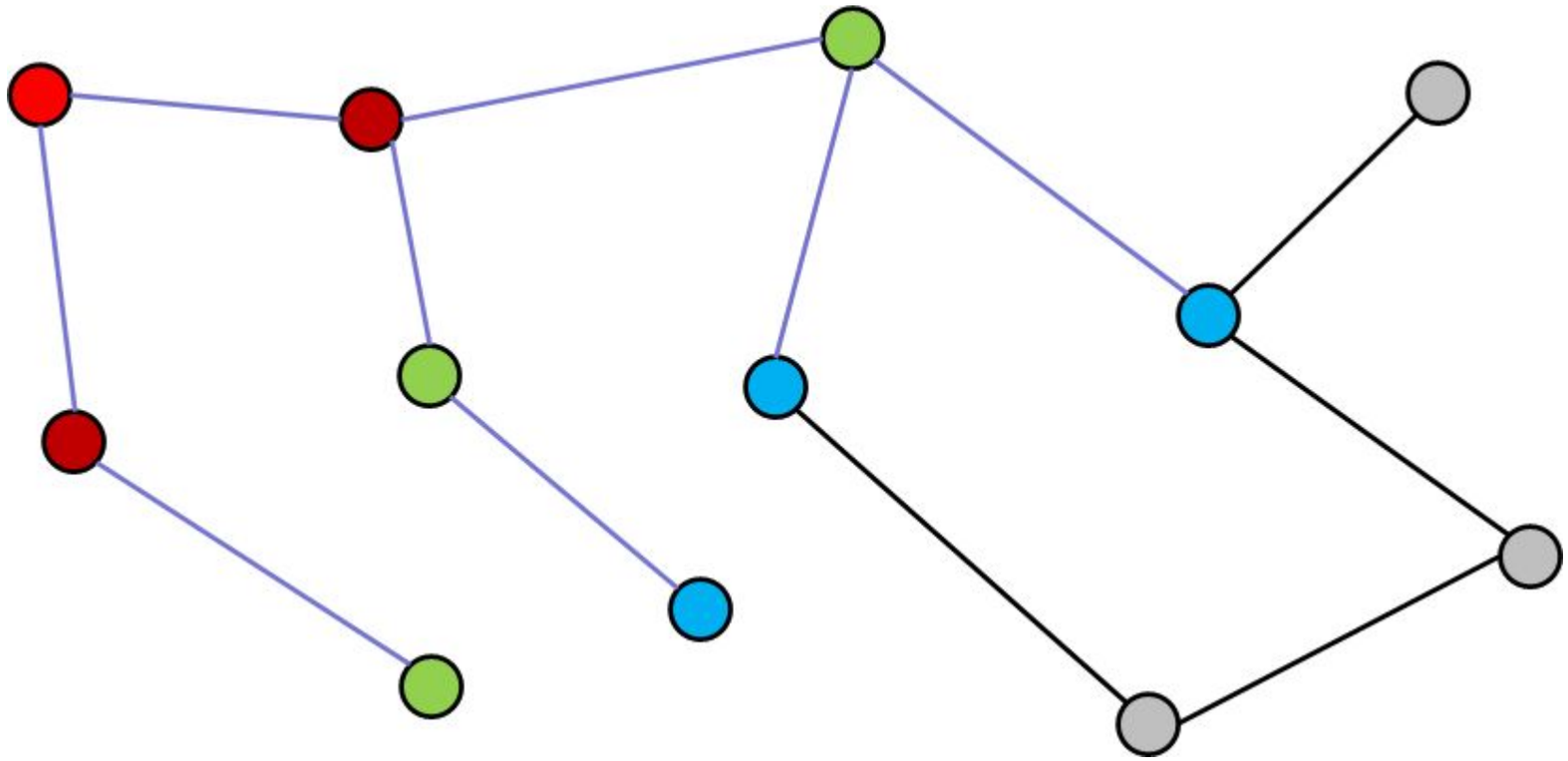
How to find a cycle in a graph?



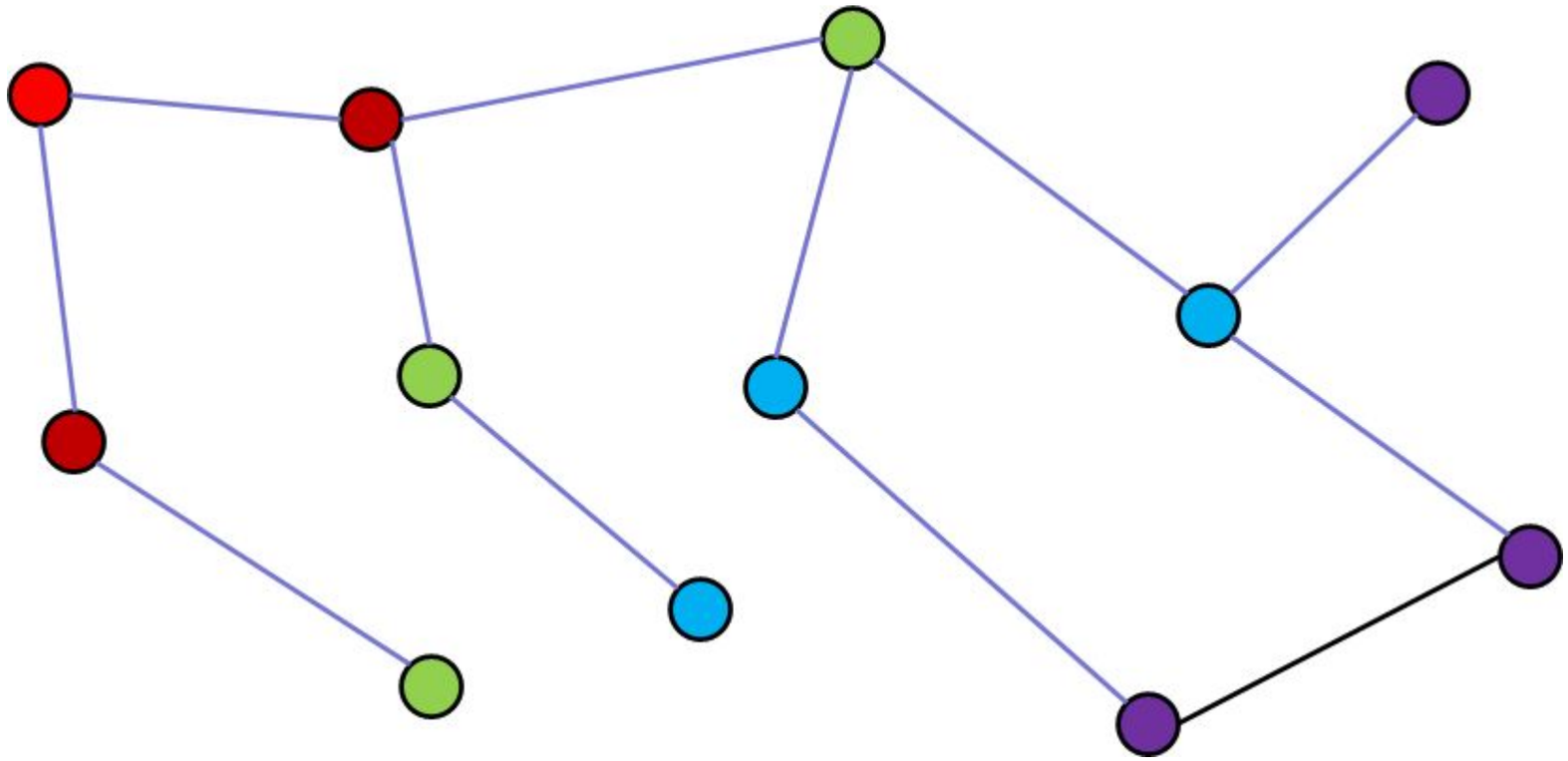
How to find a cycle in a graph?



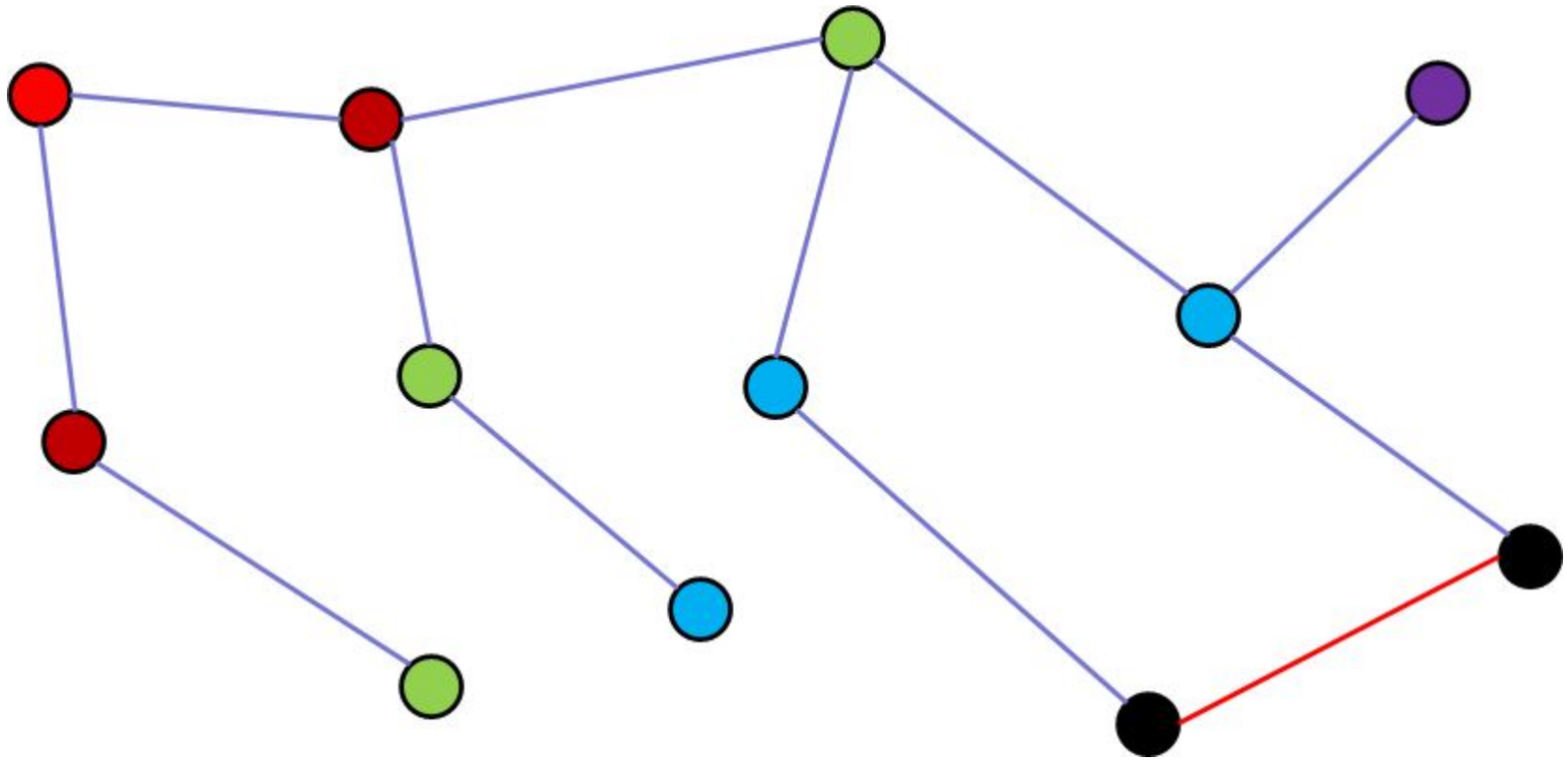
How to find a cycle in a graph?



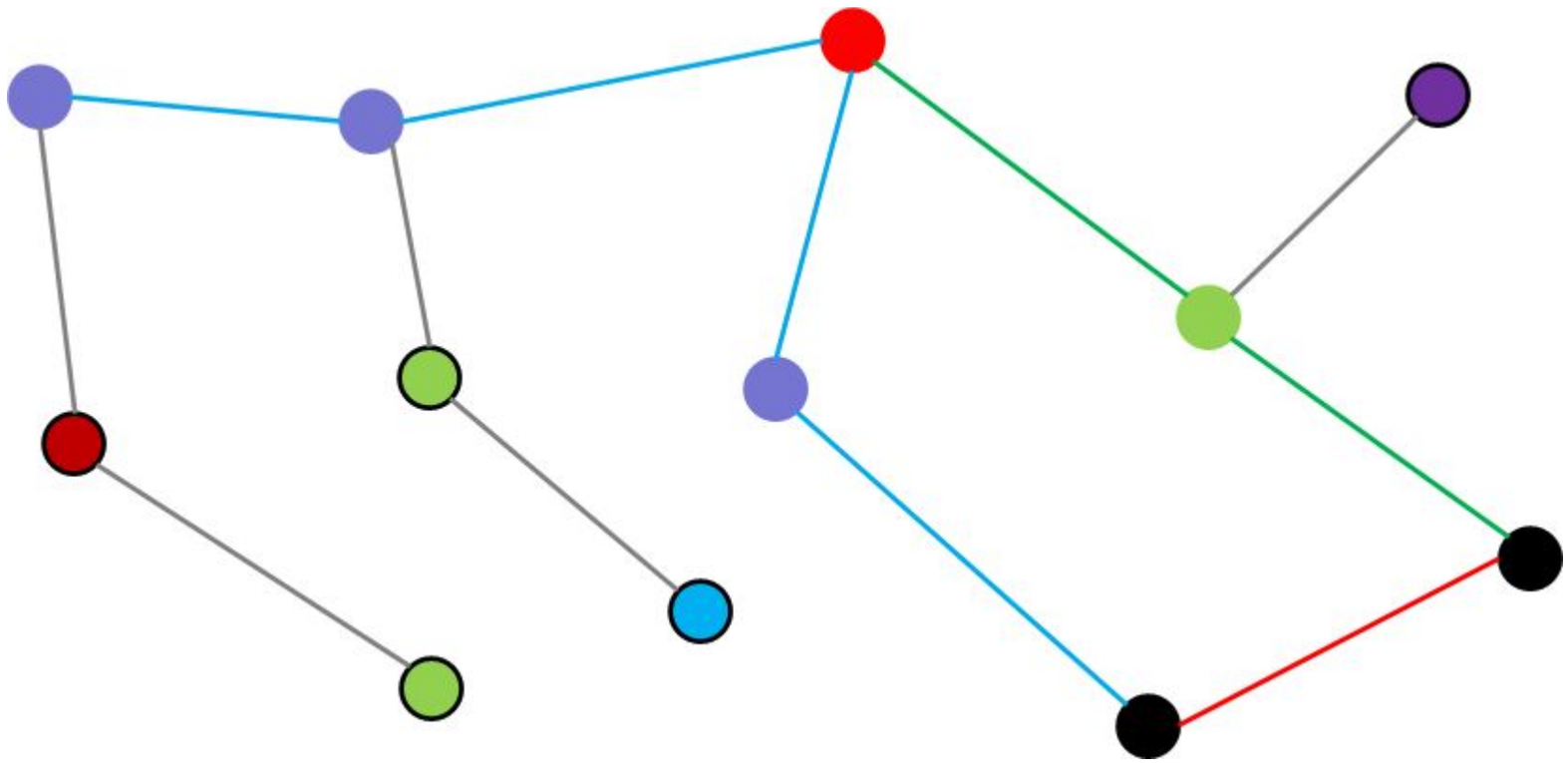
How to find a cycle in a graph?



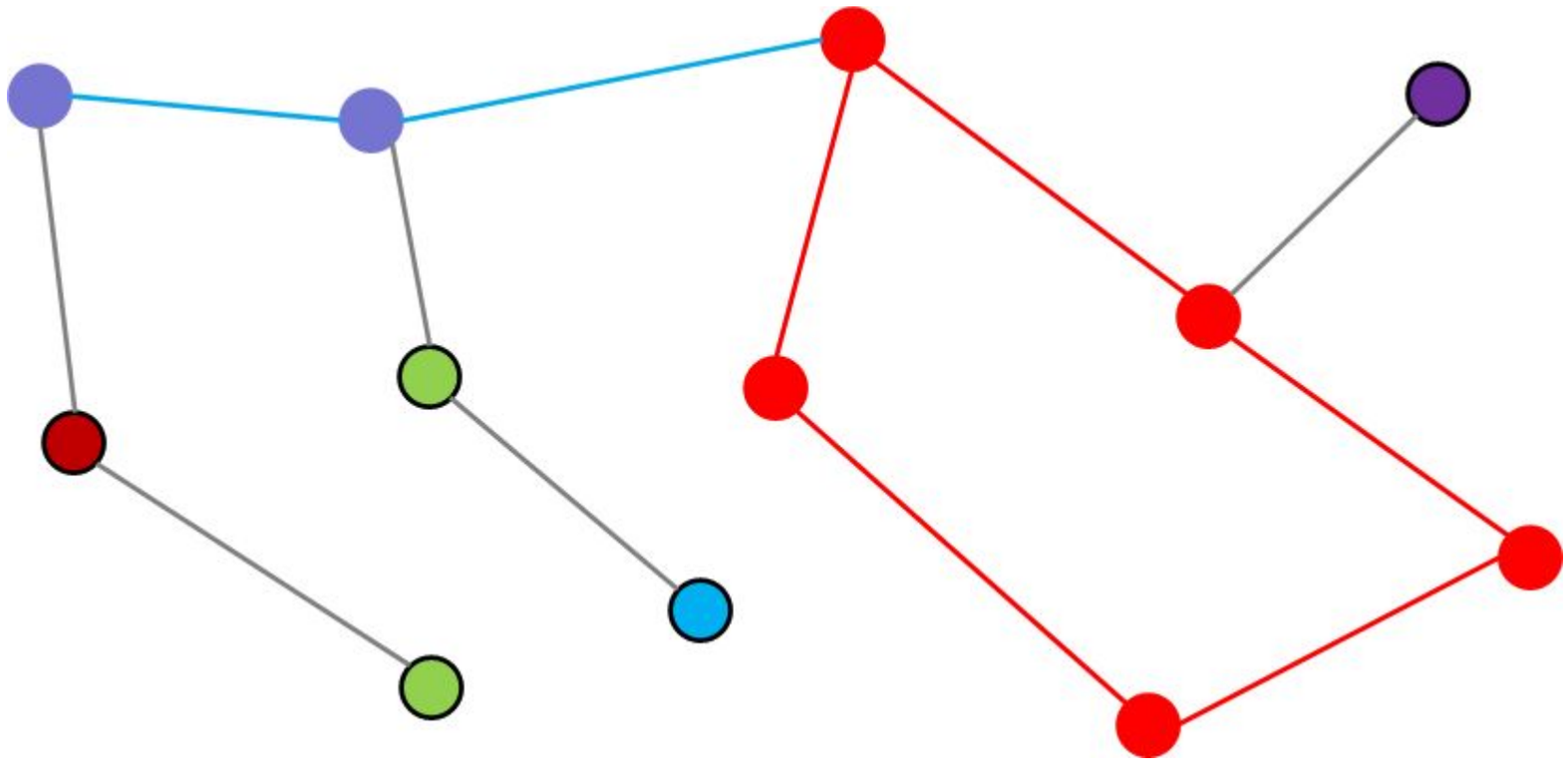
How to find a cycle in a graph?



How to find a cycle in a graph?



How to find a cycle in a graph?



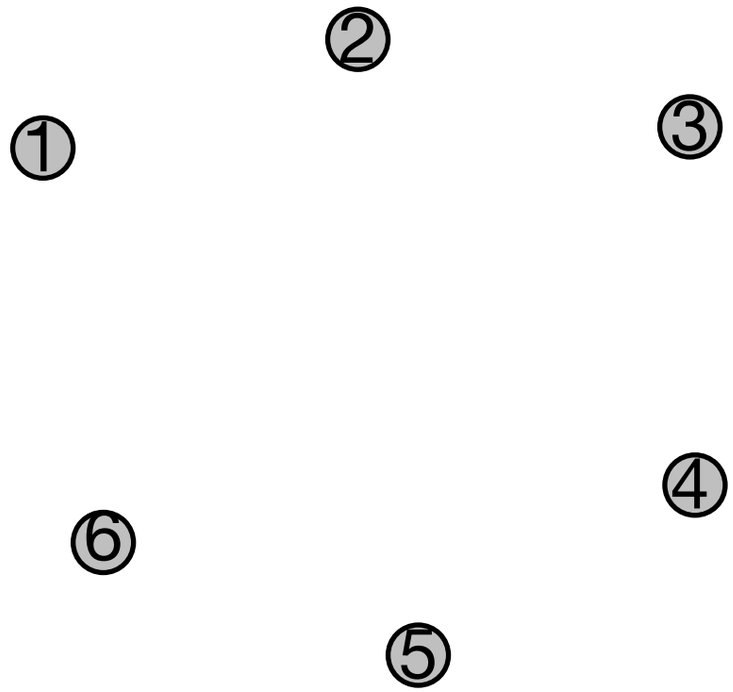
Adjacency Matrix

- When we can't draw a graph, we can represent it with a matrix.
- If row i, column j contains a 1, there is an edge between nodes i and j. If there's a 0, there's no edge.
- **Exercise**: Draw the graph for this adjacency matrix. What's the degree of each node? Is there something special about node 1? Are there any cycles? What are their lengths?

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

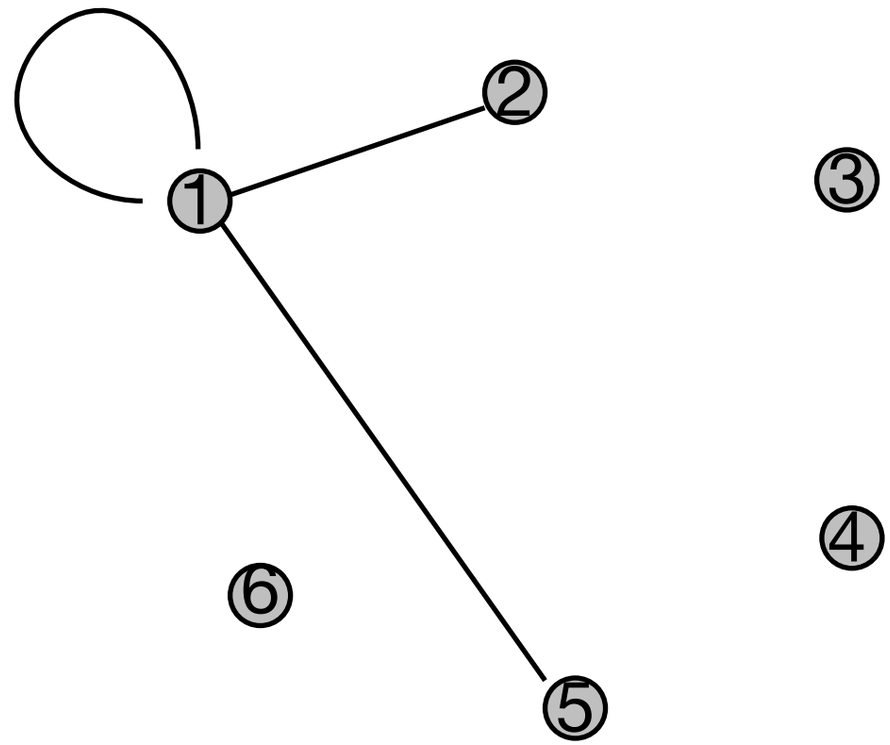
Adjacency Matrix

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



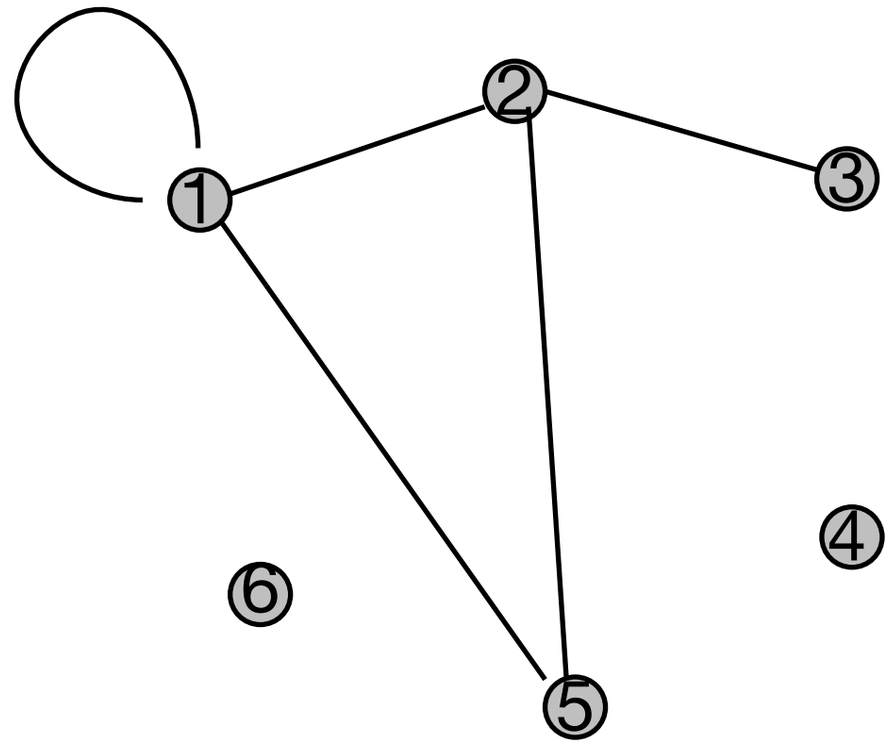
Adjacency Matrix

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



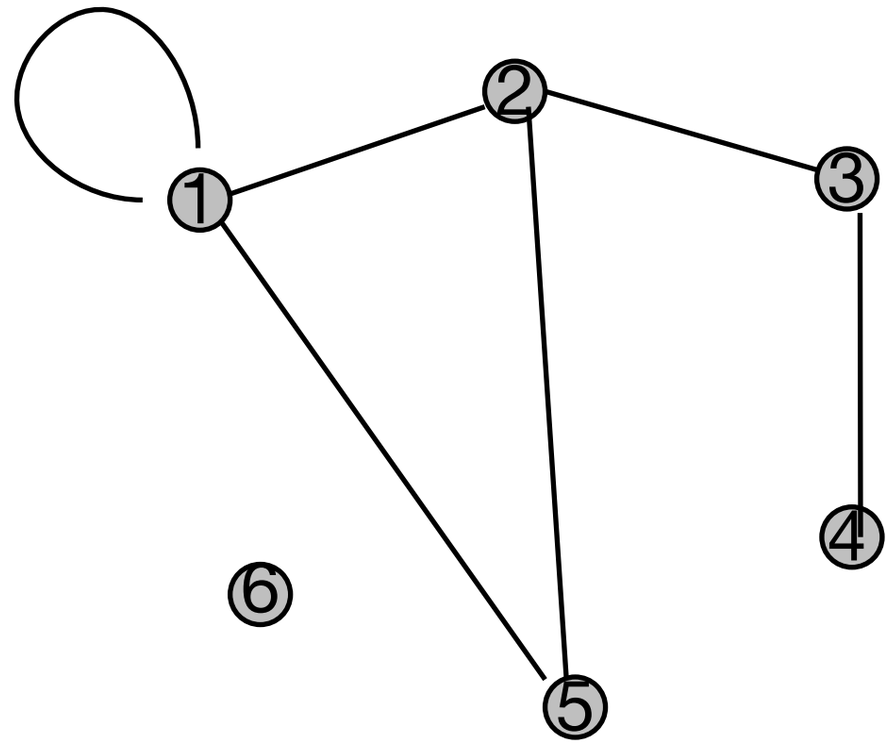
Adjacency Matrix

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



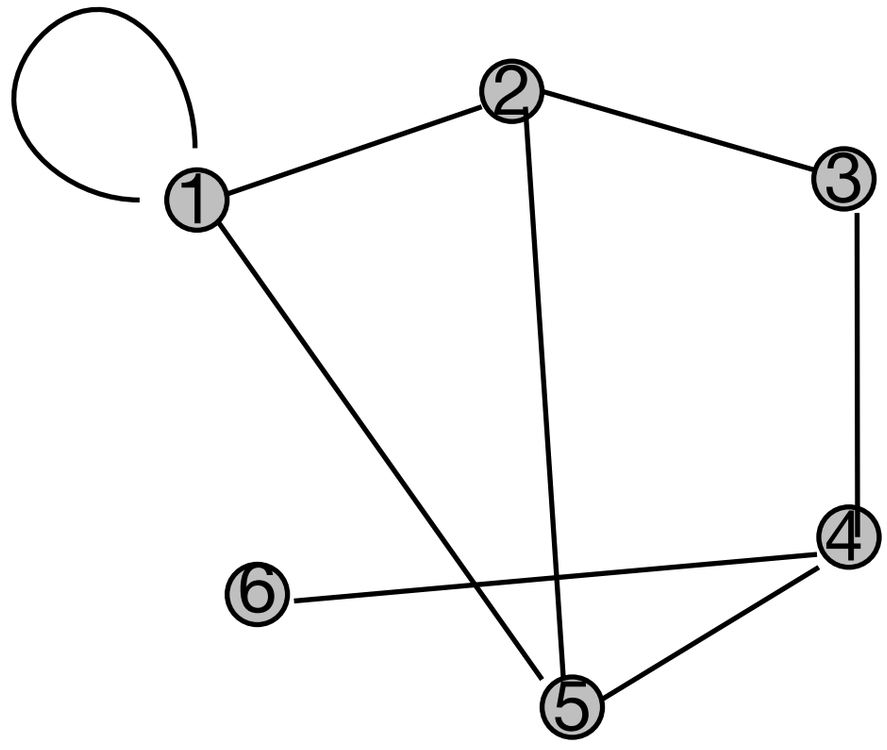
Adjacency Matrix

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



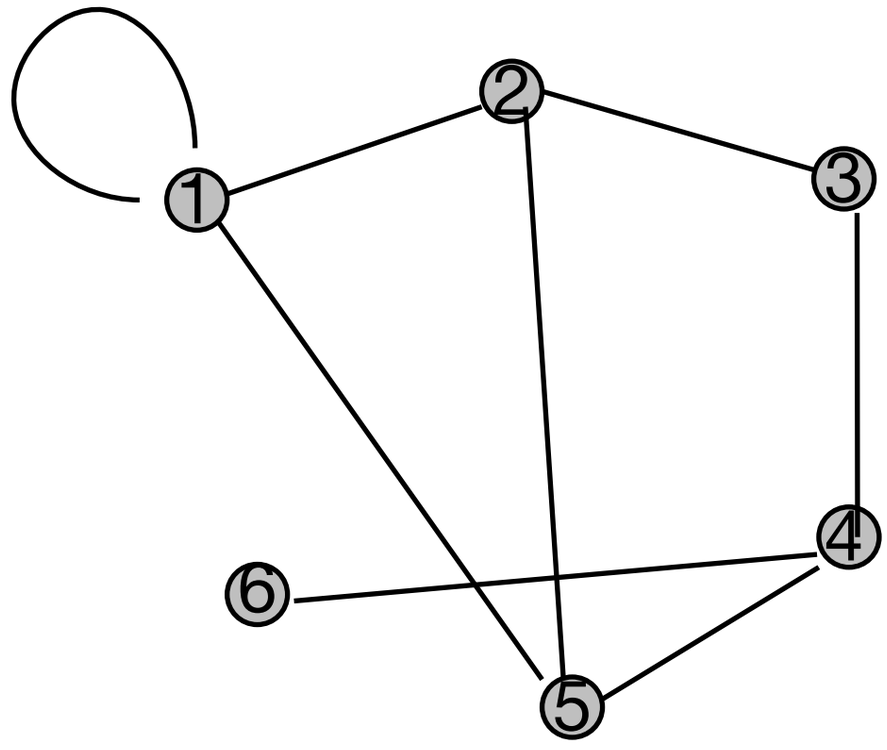
Adjacency Matrix

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



Adjacency Matrix

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



Exercise:

- What's the degree of each node? How is it related to the adjacency matrix?
- Is there something special about node 1?
- Are there any cycles? What are their lengths?
- Is the Adjacency matrix of a graph symmetric?

Further topics:

- Social Network:
  - Social Network Visualization

    http://www.science.smith.edu/dftwiki/index.php/Visualizations:_Social_Networks

- Classical Graph Theoretical Problems:
  - Seven Bridges of Konigsberg
  - Traveling Salesman Problem
  - Four Color Theorem
  - Algorithms on graphs

- What's more?
  - Algebraic graph theory: Complexity theory
  - Extremal graph theory: Cryptography, Coding theory
  - Random graph theory: Social networks, Network navigation
  - Machine learning

Exercise 1: Easy graph theory

- First, take a look at matrix0.txt in data folder
- It’s an adjacency matrix
- Now, look at LACC2021_graph_theory_day1_Exc.ipynb
  - All we’ll do is implement two simple functions:
  - matrix_load() reads the matrix.txt file and loads it into a matrix (an array of arrays)
  - print_degrees() prints the degrees of each node
  - Comments / hints are provided above each function.

- Optional: is_connected() returns 1 if the graph is connected, 0 otherwise.

Mini-Project: Shortest Path

- Next we'll do something harder.
  - Implement a function that finds the shortest path between arbitrarily two nodes:

    def shortest_path (mat,node1,node2)

    - Print out 'No such path' and return [] if node1 and node2 are not connected
    - Print out the shortest path by the edges on the path and return a list containing the nodes on the graph otherwise.

  - It's not obvious how to do this at first – try out some ideas on paper, then code it up only when you have a working algorithm that you can do manually.

  - Generate multiple random graphs to test on your program

Mini-Project: Cycle Detection

- Next we'll do something harder.
  - Implement a function that detects whether or not there is a cycle in the graph. The definition will be:

def detect_cycle(mat)

- Print out an arbitrary edge on the cycle and return a list containing all nodes on the cycle if there is one
- Print out 'No cycle in the graph' and return [] otherwise
- In the first step, assume the graph is connected and solve the problem.

- Generate multiple graphs to test on your program
- Question: Does it make any difference if the graph is directed?

Mini-Project: Cycle Detection for General Graphs

- Next we'll do something harder.
  - Bonus: solve it for more general graphs that contain multiple connected components
  - The definition will be:

def detect_cycle_gen(mat)

- The requirements are specified in CycleGen_E.py
- Generate multiple graphs to test on your program
- Question: Does it make any difference if the graph is directed?

Mini-Project: Preliminaries

- Things you might need in Python:
  - Lists
    - Suppose A is a list
    - A.append(x) appends x to A
    - The expression (x in A) returns True if A contains x, otherwise it returns False
    - A[n] returns the (n+1)-th element of A
    - The elements of a list could also be lists
      - A=[2,3,[4,5],6,[7,[8,9]]]
      - A[0]=2,A[1]=3,A[2]=[4,5],A[3]=6,A[4]=[7,[8,9]]
      - A[2][0]=4,A[2][1]=5
      - A[4][0]=7,A[4][1]=[8,9]
      - A[4][1][0]=8,A[4][1][1]=9
  - Reading files
    - Filename “matrix.txt”
    - Fh=open(“matrix.txt”)
    - A=Fh.readlines() will return a list containing all the lines of the file

Mini-Project: Preliminaries

- Things you might need in Python:
  - Reading files
    - Filename “B.txt”
    - Fh=open(“B.txt”)
    - A=Fh.readlines() will return a list containing all the lines of the file
    - Example
    - If the file “B.txt” contains the following text:

Hello world!
010001
Python

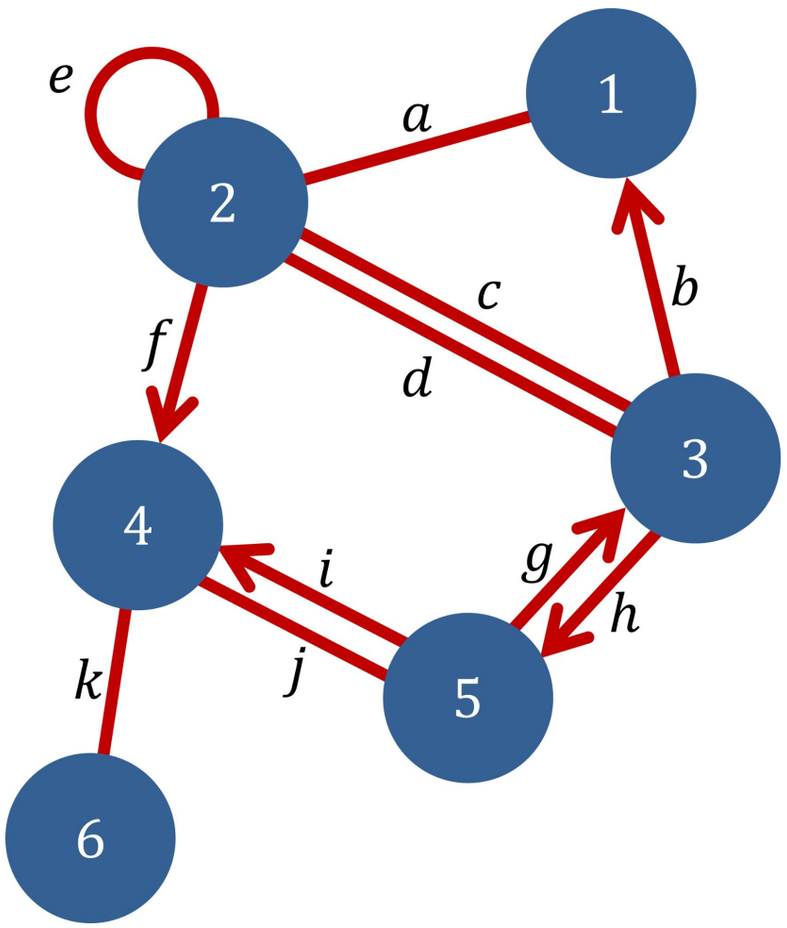
    - Then A=[‘Hello world!’, ‘010001’, ‘Python’]
      - Each component of A is a string

References

- Graphs: http://people.brunel.ac.uk/~mastjjb/jeb/or/graph.html
- Graph Theory: http://en.wikipedia.org/wiki/Graph_theory
- Kevin Bacon: http://oracleofbacon.org/

Quick Recap

- Last day, we learnt about:
  - Graphs: Vertices, Edges, Weights, Directivity
  - Degree, Connectivity
  - Shortest Path, Cycles, Tree
  - Adjacency Matrix



Contents

- Common random graphs:
  - Barabasi-Albert Networks with Preferential Attachment
  - Erdos-Renyi Networks

- Community Structures and Modularity

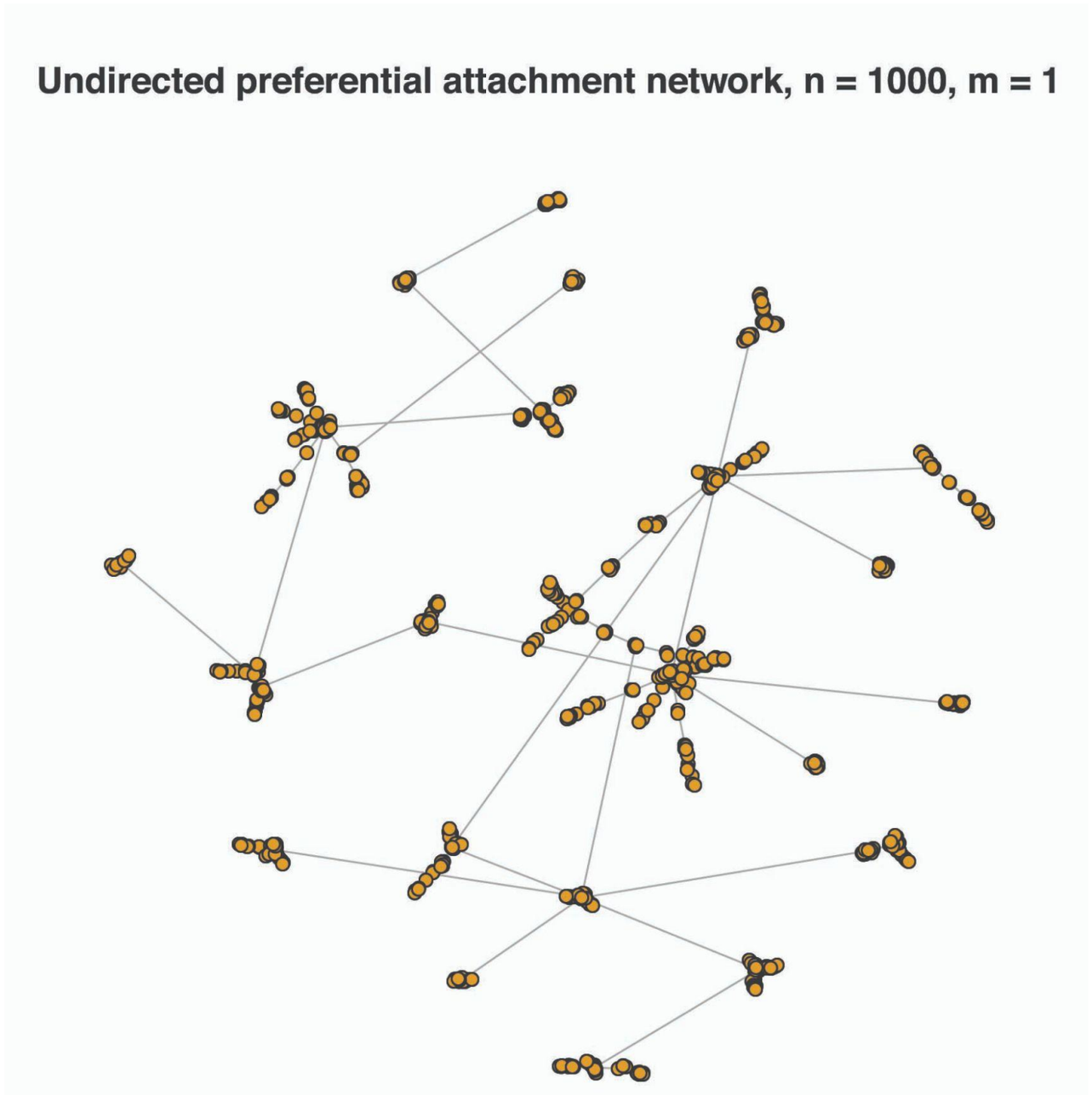
- Minimum Spanning Tree (MST)
- Shortest Path: Dijkstra's Algorithm

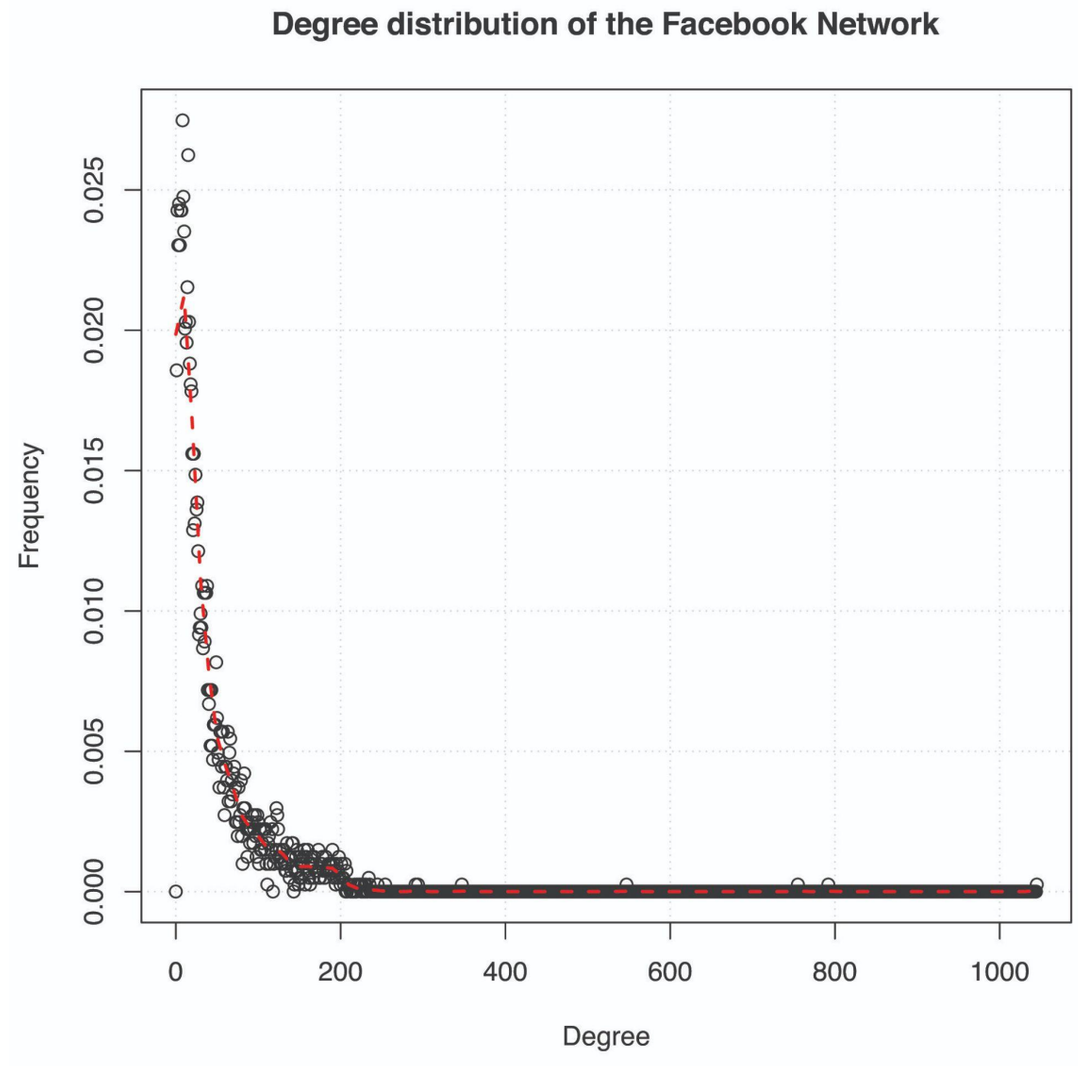
- Traveling Salesman Problem (TSP)

Preferential attachment

- Popular nodes act as bridge between unpopular nodes.
- The more popular a node, the higher is the probability for it to be connected to incoming nodes.
- Popularity is measured by the degree of a node.
- The Barabasi-Albert model creates **scale-free**, **evolving** and **connected** networks with preferential attachment.
- The degree distribution of such networks follow power-law.

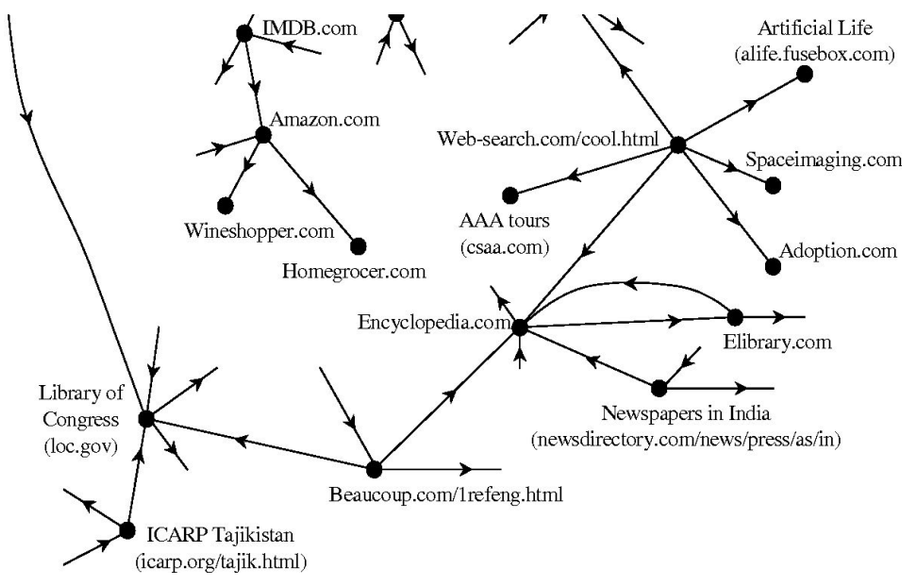
Preferential attachment

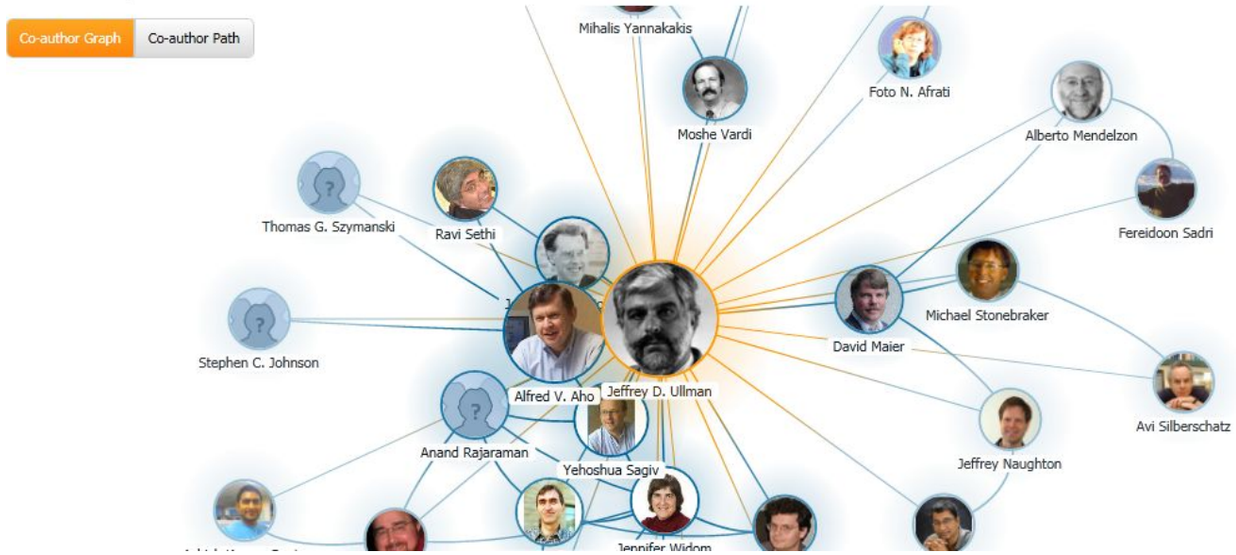




Applications

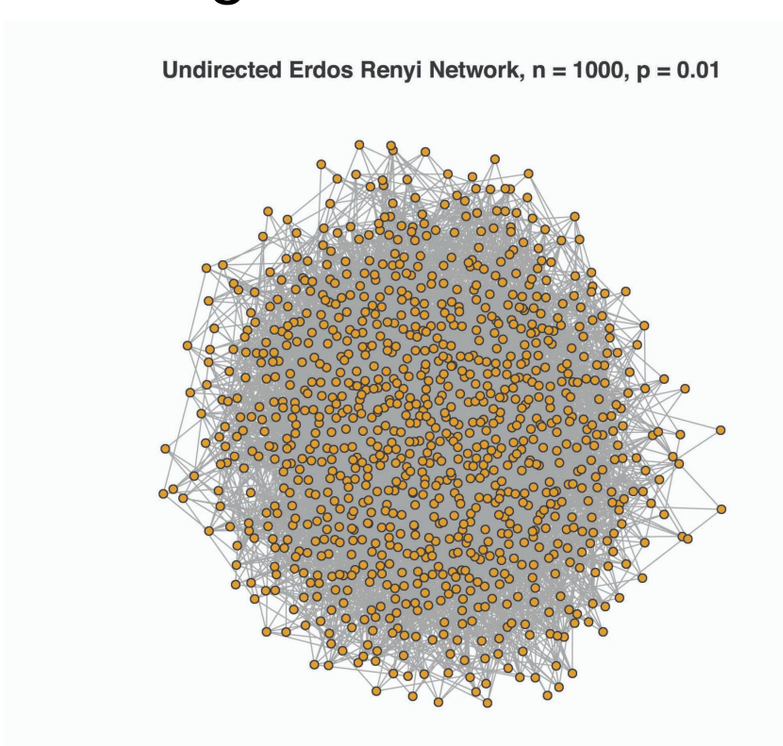
- Social Networks
- Search engines (PageRank)
- World Wide Web
- Citations among researchers
- Internet
- Recommendation Systems
- Family / Erdos Trees

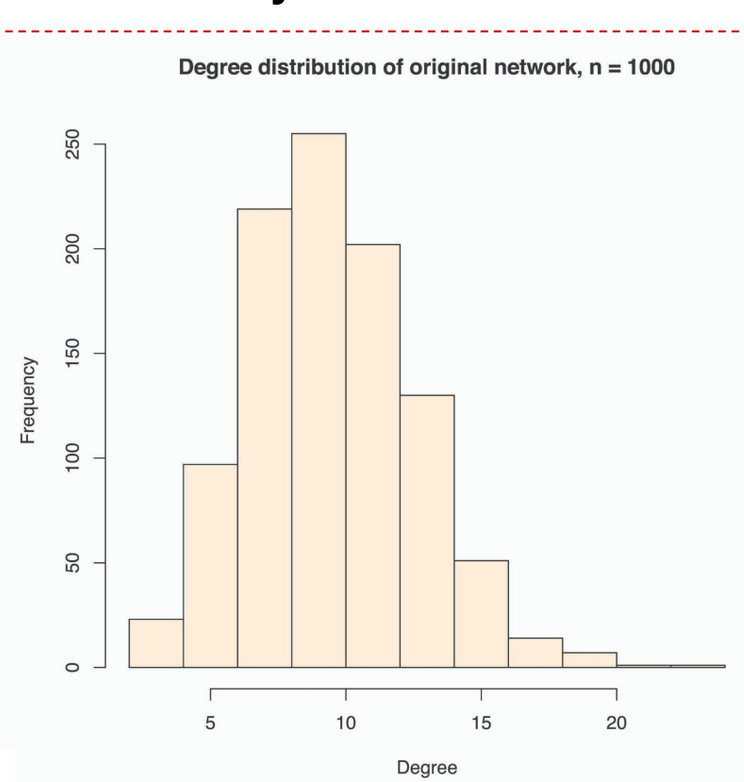


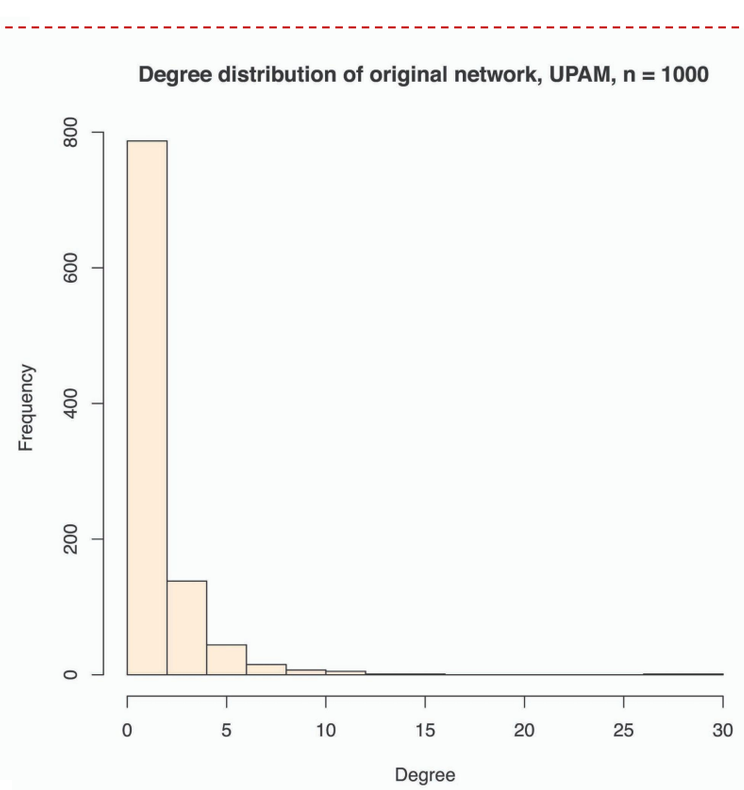


Erdos-Renyi Models

- All edges have a certain probability of being present or absent (probability of connectedness).
- Network may or may not be fully connected; concept of Giant Connected Component (GCC)
- Degree distribution of Erdos-Renyi networks are binomial or Poisson.



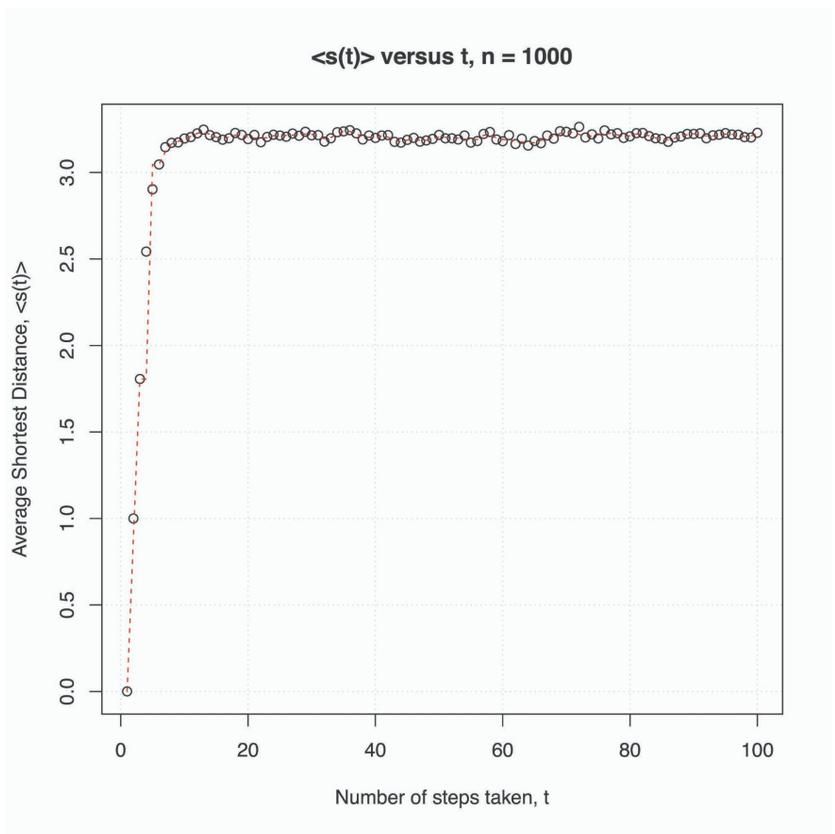


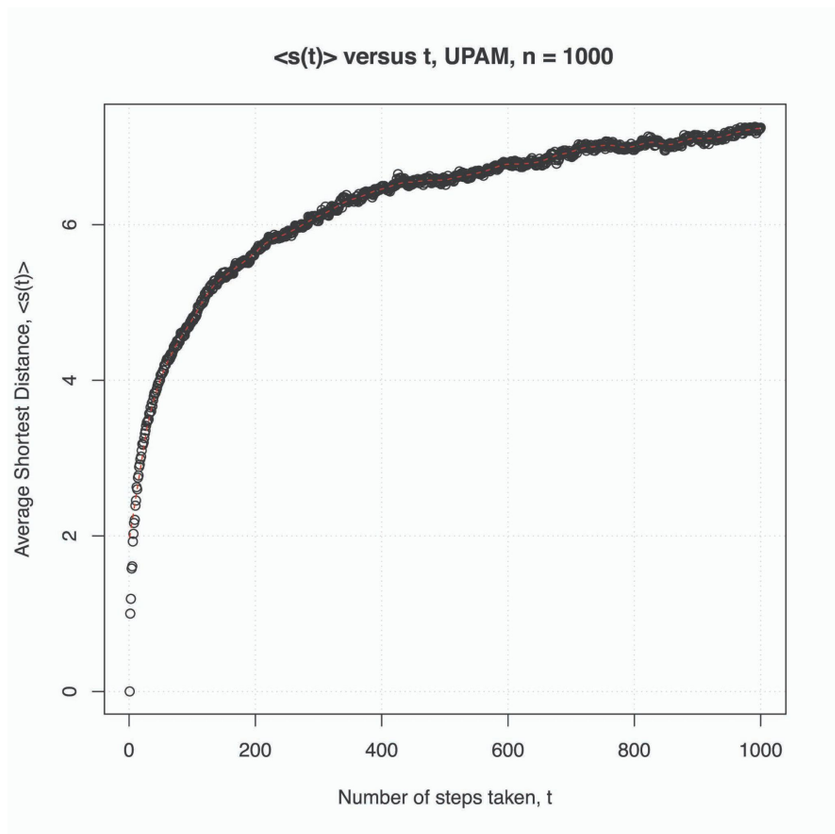


Random Walk on Random Graphs

- If you walk randomly on an Erdos-Renyi graph, you will likely end up within a “circular region” corresponding to the average shortest distance of the nodes in the entire network from an initial node.

- However, for Barabasi-Albert graphs, your average shortest distance will increase non-monotonically with more steps from the initial node.





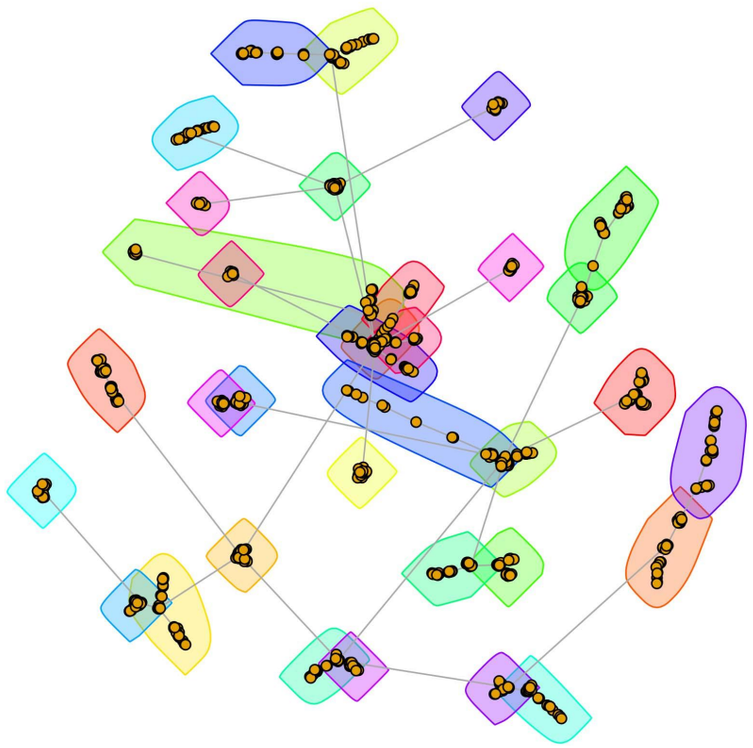
Community Structures and Modularity

- Community structures: clusters or regions of high connectedness in the graph
- We measure the strength of clustering using modularity score.

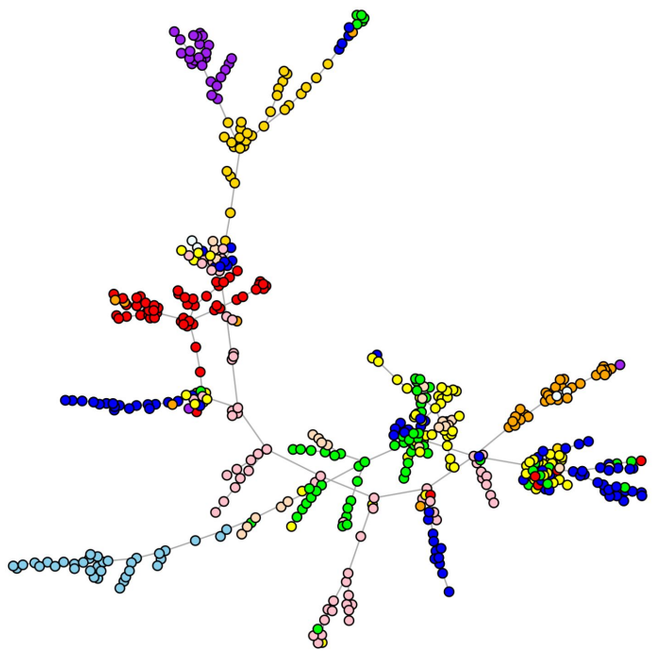
- Application example: Correlation graph of stocks

- Techniques: Fast-Greedy, InfoMap, Edge-Betweenness

Community Structure of UNPA, n = 1000, m = 1



Example of stock market correlation graph MST

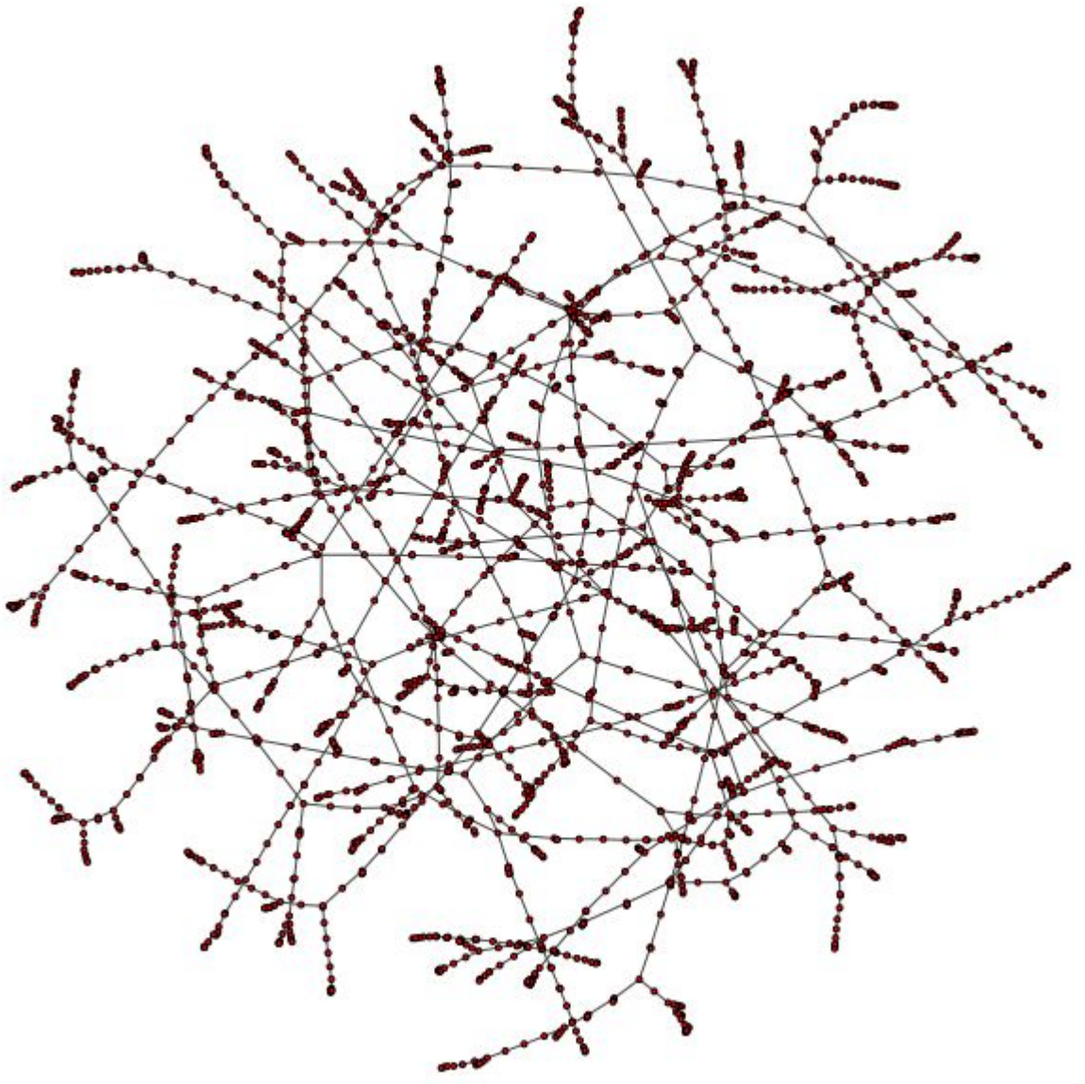


Minimum Spanning Tree (MST)

- MST: Connects all the nodes of a connected and edge-weighted undirected graph:
  - **without any cycles**
  - **with the lowest possible cumulative edge weights (lowest cost).**

- Techniques to find MST:
  - Prim’s Algorithm
  - Kruskal’s Algorithm

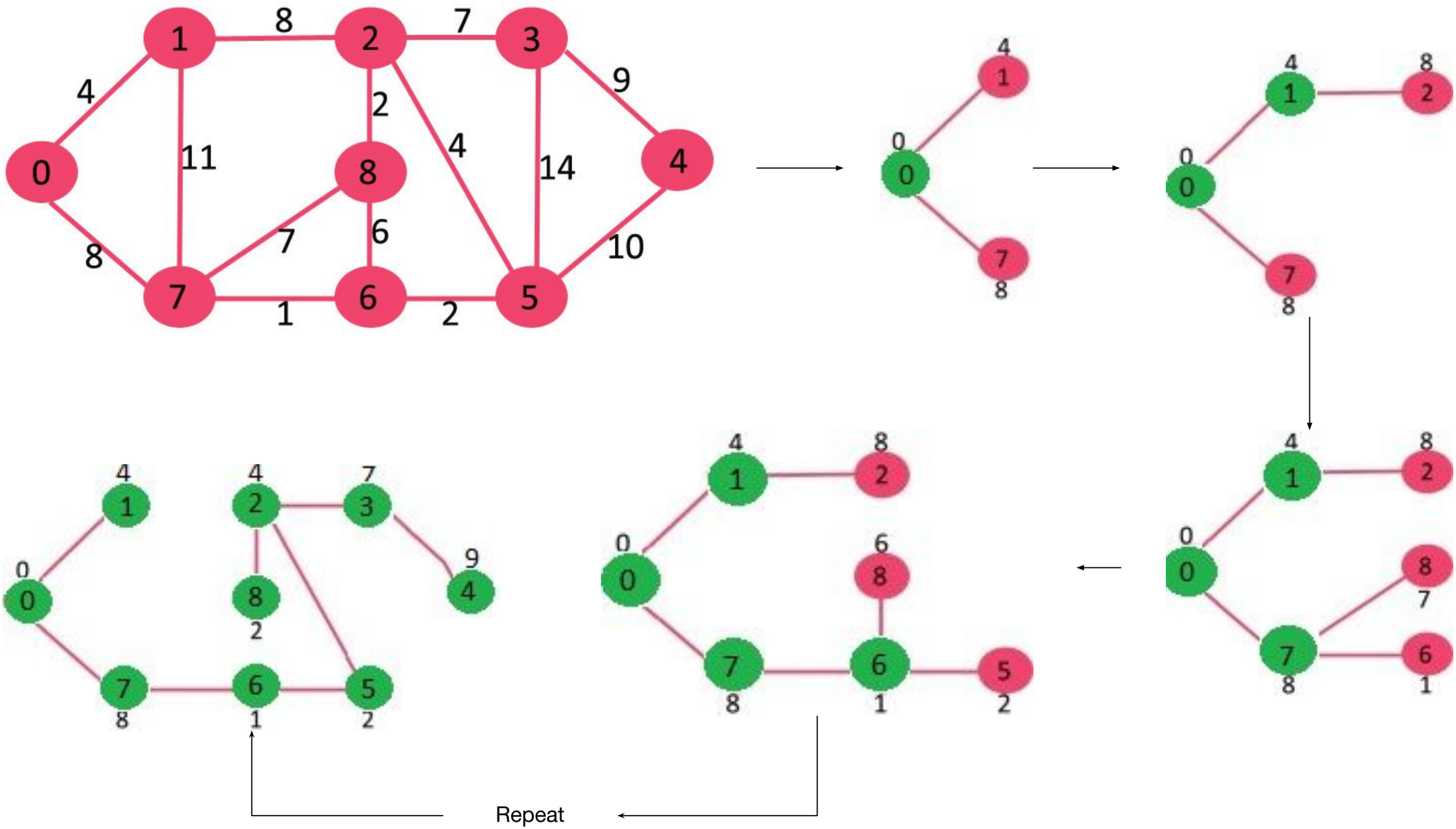
- Prim’s Algorithm (uses two sets):
  - Find a cut
  - Pick the edge with minimum weight.
  - Include the edge in the list of included vertices.



MST of Uber Dataset

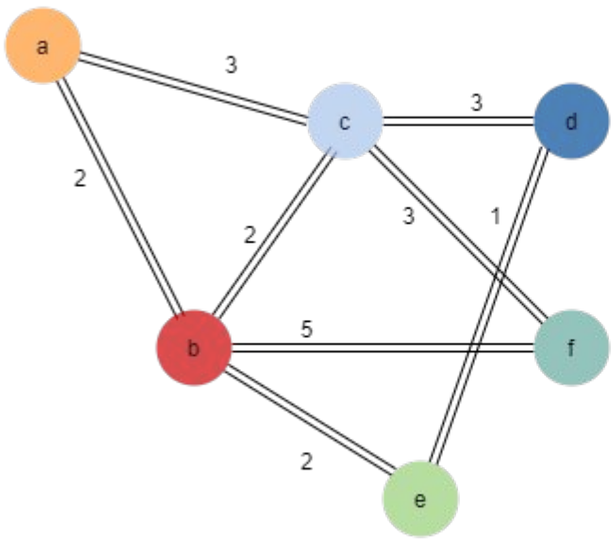
https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/

Prim's Algorithm

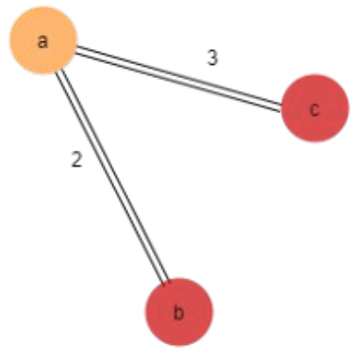


https://medium.com/star-gazers/graph-theory-finding-the-shortest-paths-afb35d28176b

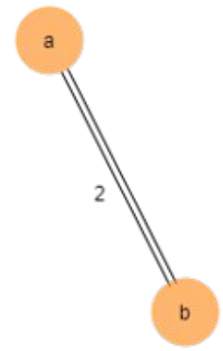
Dijkstra's Shortest Path Algorithm



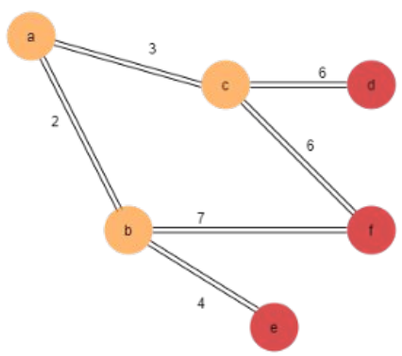
Edge		Weight
a	b	2
a	c	3
b	c	2
b	e	2
b	f	5
c	d	3
c	f	3
d	e	1



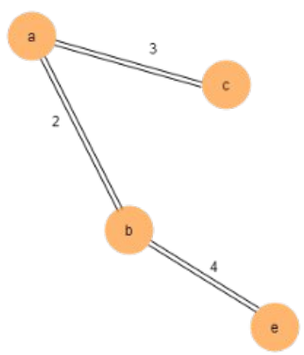
Edge		Weight
a	b	2
a	c	3



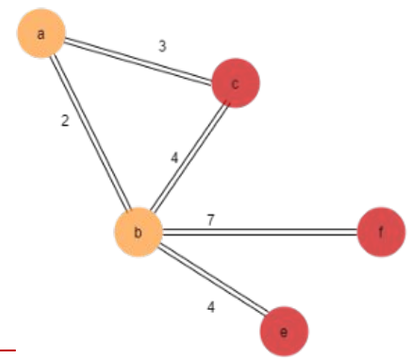
Edge		Weight
a	b	2



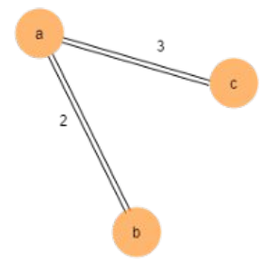
Edge		Weight
c	d	6
c	f	6
b	e	4
b	f	7



Edge		Weight
a	b	2
a	c	3
b	e	4



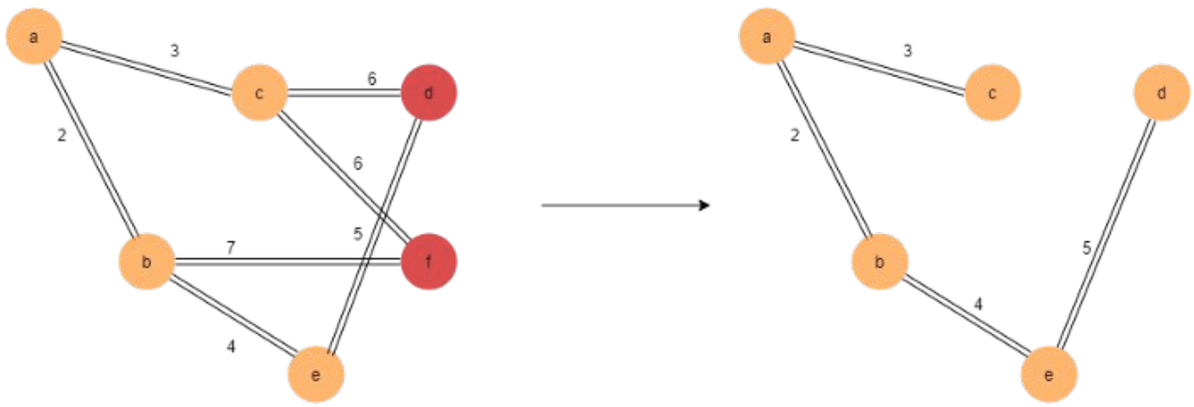
Edge		Weight
a	c	3
b	c	4
b	e	2
b	f	5



Edge		Weight
a	b	2
a	c	3

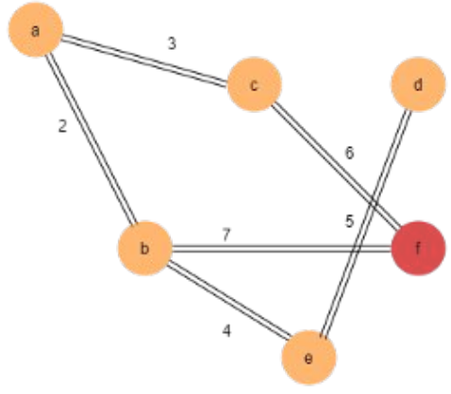
https://medium.com/star-gazers/graph-theory-finding-the-shortest-paths-afb35d28176b

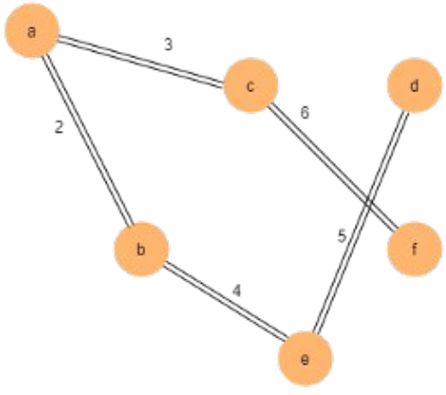
Dijkstra's Shortest Path Algorithm (contd.)



Edge		Weight
c	d	6
c	f	6
d	e	5
b	f	7

Edge		Weight
a	b	2
a	c	3
b	e	4
d	e	5





Edge		Weight
c	f	6
b	f	7

Edge		Weight
a	b	2
a	c	3
b	e	4
d	e	5
c	f	6

https://medium.com/star-gazers/graph-theory-finding-the-shortest-paths-afb35d28176b
https://www.baeldung.com/cs/prim-dijkstra-difference

Prim’s versus Dijkstra's

- Dijkstra’s algorithm finds the shortest path, but Prim’s algorithm finds the MST.
- Dijkstra’s algorithm can work on both directed and undirected graphs, but Prim’s algorithm only works on undirected graphs.
- Prim’s algorithm can handle negative edge weights, but Dijkstra’s algorithm may fail to accurately compute distances if at least one negative edge weight exists.
- Prim’s Algorithm adds the next vertex that is closest to the tree. Dijkstra’s Algorithm, on the other hand, adds the next vertex that is closest to the root.
- Intuitive example:
  - Dijkstra's: Find shortest and lowest cost path in existing road network.
  - Prim’s: How to create the road network itself efficiently.

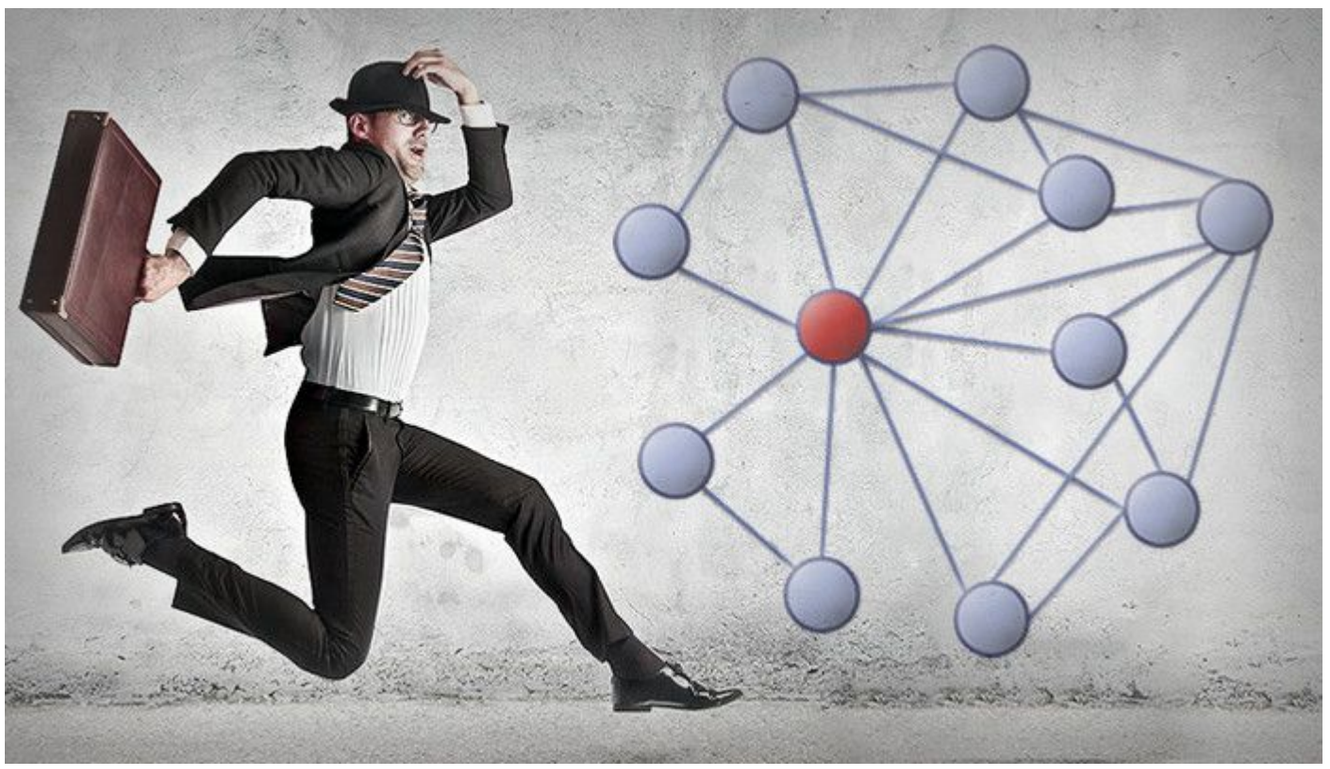
Traveling Salesman Problem (TSP)

- Problem formulation: Find the shortest route from a city such that all other cities are visited only once and the traveller returns to the original city.

- The triangle inequality: sum of any two sides of a triangle must always be greater than the third side: $a + b > c$, $a + c > b$, $b + c > a$.

- The route is also called a Hamiltonian cycle satisfying the triangle inequality

- Graph:
  - Nodes: cities
  - Edges: Roads
  - Weights: Distance, travel time, etc.



https://towardsdatascience.com/how-to-solve-the-traveling-salesman-problem-a-comparative-analysis-39056a916c9f

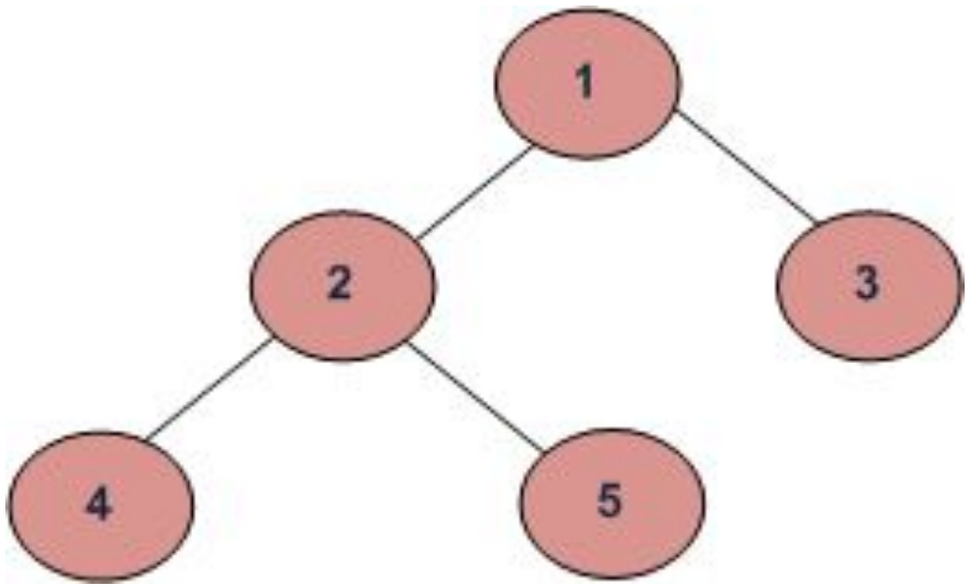
TSP Intuition

- TSP is NP-hard: It is not possible to guarantee you will find the shortest path within some predefined time-limit; n cities can have n! paths (nX(n-1)X(n-2)...1), e.g., 15 cities can result in trillion possible routes.

- We use approximation techniques to select pareto-optimal routes.

www.variability.org

Pre-order traversal

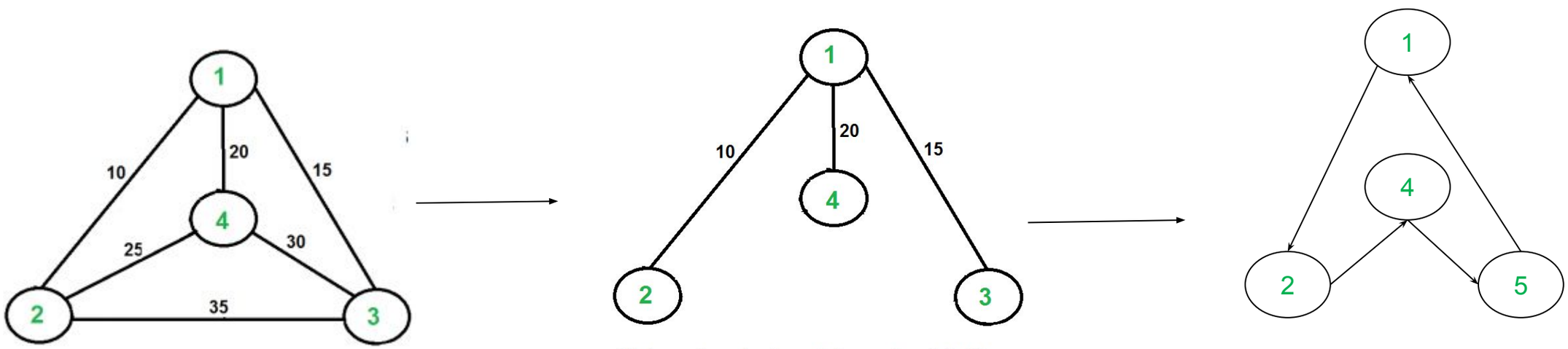


1 - 2 - 4 - 5 - 3

https://www.geeksforgeeks.org/travelling-salesman-problem-set-2-approximate-using-mst/
https://www.geeksforgeeks.org/tree-traversals-inorder-preorder-and-postorder/

2-approximate algorithm for TSP using MST

- The cost of the output is never more than twice the cost of best possible output.
- Algorithm:
  - Use Prim's algorithm to generate MST, using the starting/ending point as the root node.
  - Find depth-first pre-order traversal of the MST (root, left, right)
  - Add 1 at the end of the pre-order walk to get the Hamiltonian cycle.



Minimum Spanning Tree of the graph on left side with 1 as root. The Preorder Traversal of MST is 1-2-4-3. So the output is 1-2-4-3-1

Other TSP solutions

- Exact algorithms: Dynamic programming

- Approximate algorithms:
  - 1-approximate using Eulerian circuits and Dijkstra's algorithm
  - 2-approximate using MST
  - Simulated Annealing (guided local search)
  - Ant-colony Optimization

- Special cases
  - Independent set
  - 2-SAT