Evaluation

There will be no exam in this course. Grading is done entirely through reports on various tasks throughout the course.

- You are expected to hand in 2 reports (proposals)
- Count for 50% of the final grade
- Two representations about your projects
- Count for 50% of the final grade

Contents

1. Complex Network and Social Network Analysis

☐ Lectures 1-6

☐ Includes: small-world, degree distribution, finding communities, and other advanced metrics, random growth, preferential attachment, and other growth models

2. Natural Language Processing

☐Lectures 8-12

☐Includes: language model, text classification, and sentiment analysis, data mining

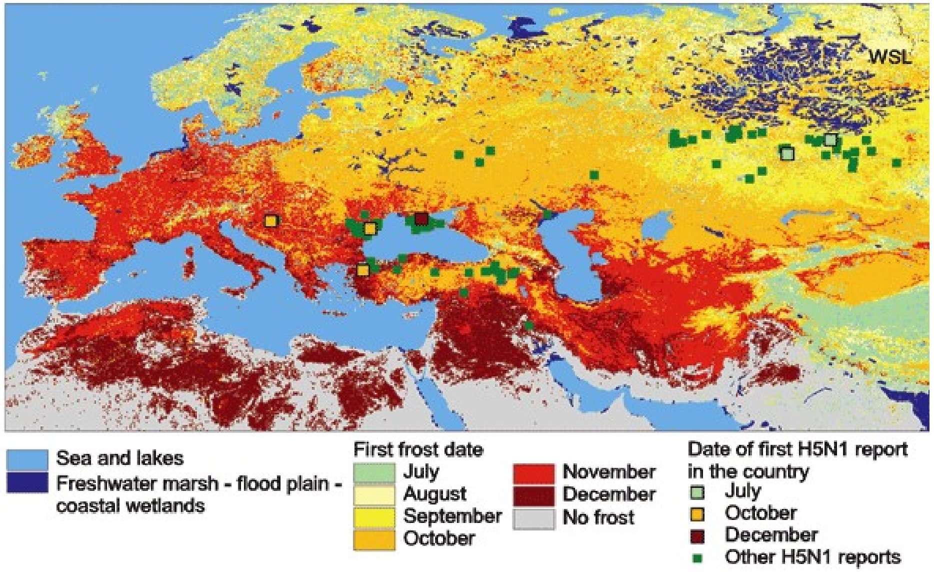
Computational Social Science

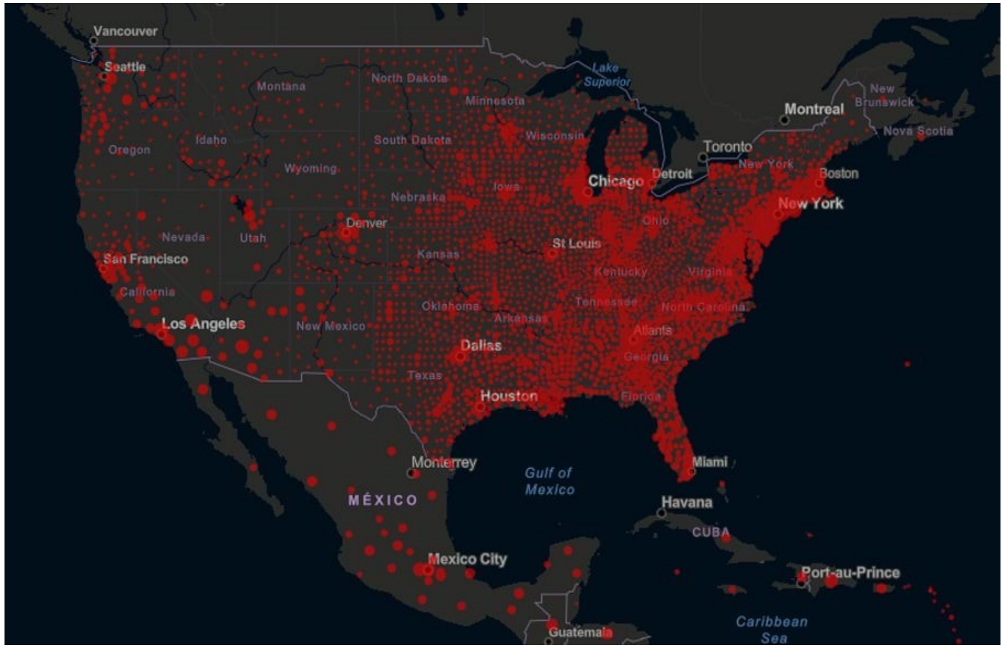
Social sciences are a group of academic disciplines dedicated to examining society. This branch of science studies how people interact with each other, behave, develop as a culture, and influence the world.

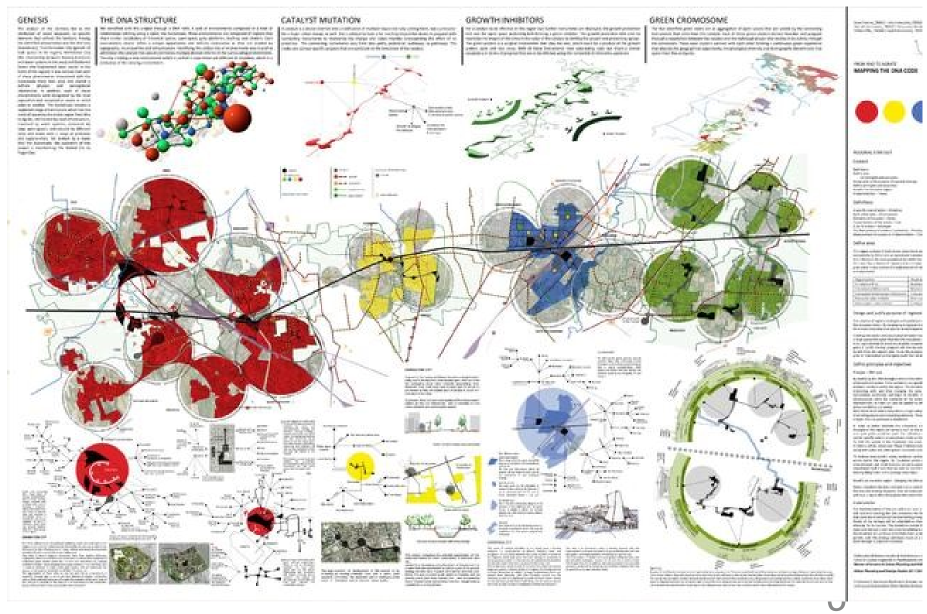
Social science is, in its broadest sense, the study of society and the manner in which people behave and influence the world around us.

Social science tells us about the world beyond our immediate experience, and can help explain how our own society works - from the causes of unemployment or what helps economic growth, to how and why people vote, or what makes people happy. It provides vital information for governments and policymakers, local authorities, non-governmental organisations and others.

Computational Social Science







Computational Social Science

Data | **Methodology** | **Conclusions**

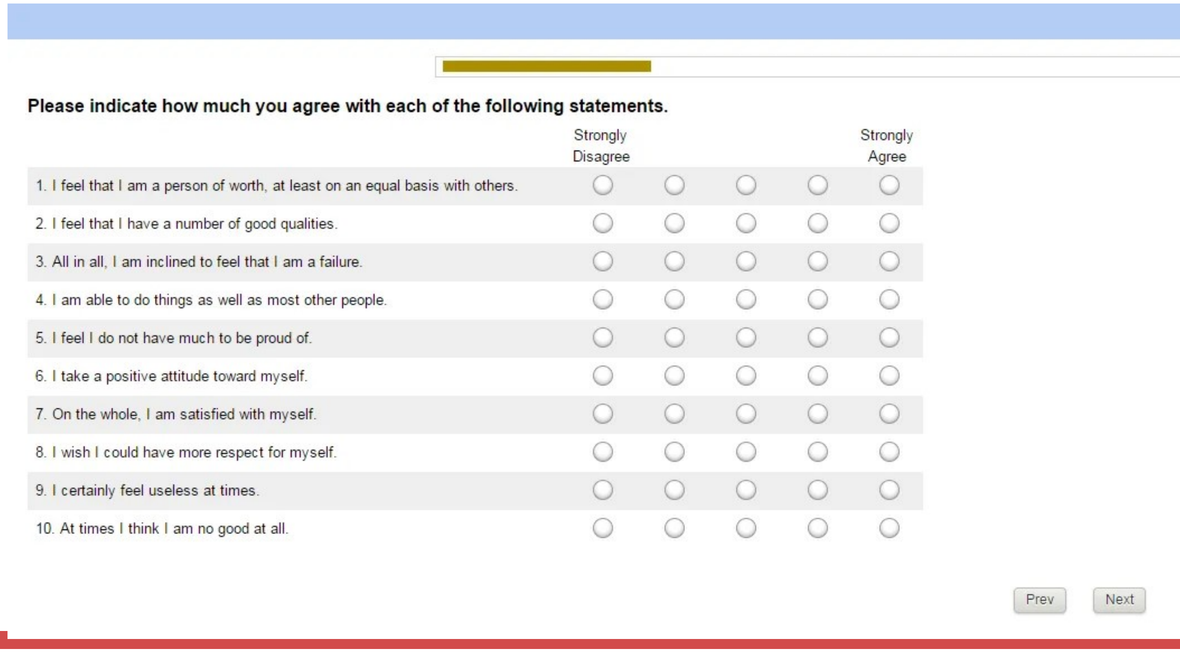
How to collect data?

Computational Social Science

Data | **Methodology** | **Conclusions**

How to collect data?

- ☐ Questionnaire
- ☐ Phone Questionnaire
- ☐ Online Questionnaire



Please indicate how much you agree with each of the following statements.

	Strongly Disagree				Strongly Agree
1. I feel that I am a person of worth, at least on an equal basis with others.	○	○	○	○	○
2. I feel that I have a number of good qualities.	○	○	○	○	○
3. All in all, I am inclined to feel that I am a failure.	○	○	○	○	○
4. I am able to do things as well as most other people.	○	○	○	○	○
5. I feel I do not have much to be proud of.	○	○	○	○	○
6. I take a positive attitude toward myself.	○	○	○	○	○
7. On the whole, I am satisfied with myself.	○	○	○	○	○
8. I wish I could have more respect for myself.	○	○	○	○	○
9. I certainly feel useless at times.	○	○	○	○	○
10. At times I think I am no good at all.	○	○	○	○	○

Prev Next

Computational Social Science

Data | **Methodology** | **Conclusions**

1. It is generally regarded as dependable when used to obtain statement of facts.

2. Information of a personal nature often may be obtained more readily by means of questionnaires.

3. It places less pressure on the subject for immediate response.

4. It helps in focusing the respondent's attention on all the significant items. It does not permit much of variation.

5. It may be used as a preliminary tool for conducting indepth study later on by any other method.

6. The responses given by the subjects are available in their own language and version.

Computational Social Science

Data | **Methodology** | **Conclusions**

- ☐Its reliability and validity is low. It seeks secondary information concerning facts when primary evidence is not at hand.
- ☐It gives a biased sample. The matter of non-response is always a big question mark.
- ☐The respondents who return the questionnaires may not constitute a representative section of the entire group.
- ☐If the subject misinterprets a question or gives an incomplete or indefinite response, nothing can be done.
- ☐Some respondents may not like to put their views on controversial issues in writing.
- ☐The behaviours, gestures, reactions, emotions of the respondent remain unnoticed.
- ☐Some of the research areas are so delicate, sensitive and confidential in nature that it becomes difficult to frame questions on them.

Computational Social Science

Data | **Methodology** | **Conclusions**

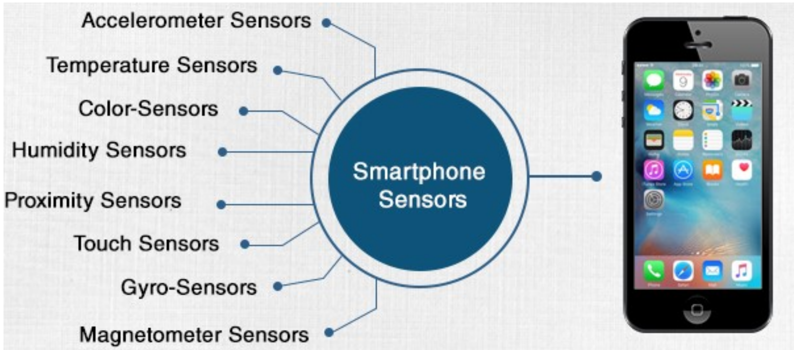
How to analysis the data?

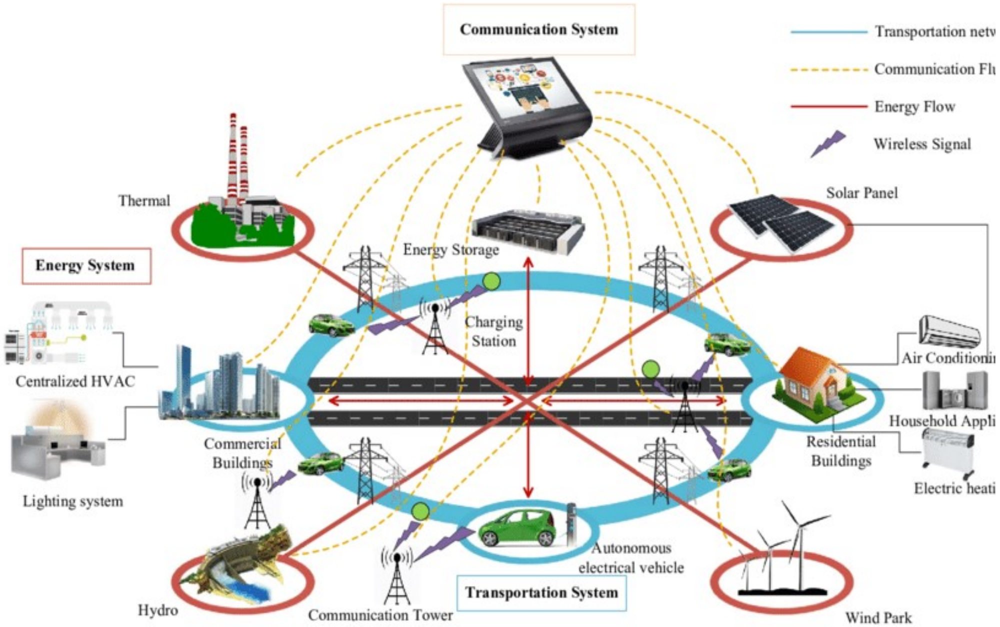
- Statistical analysis

the science of collecting, exploring and presenting large amounts of data to discover underlying patterns and trends.

Computational Social Science

Why Computational Social Science?

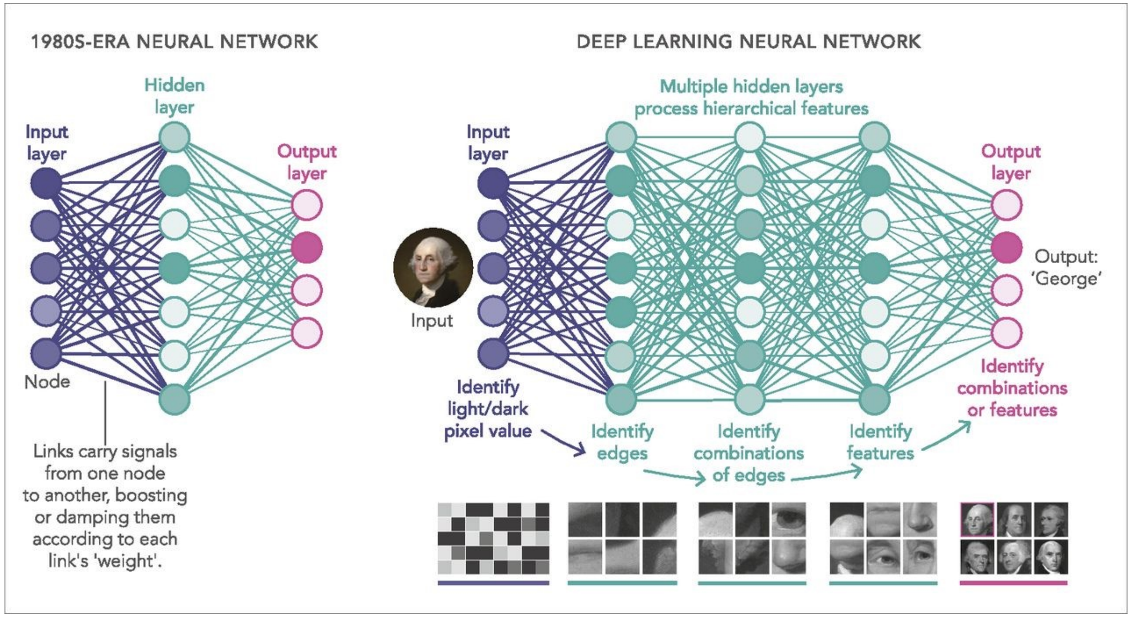


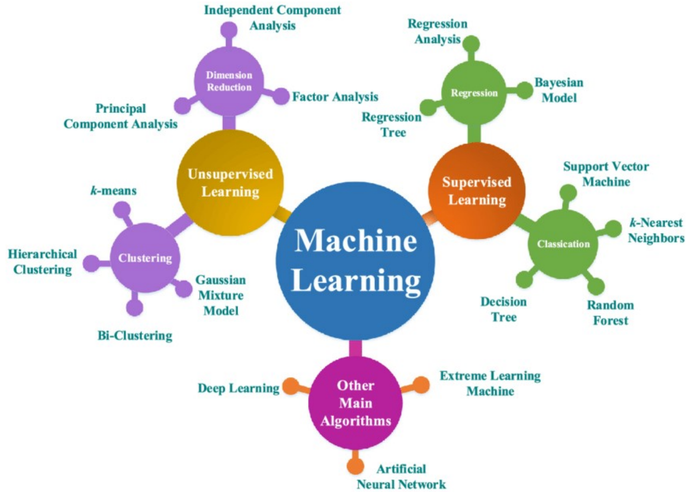


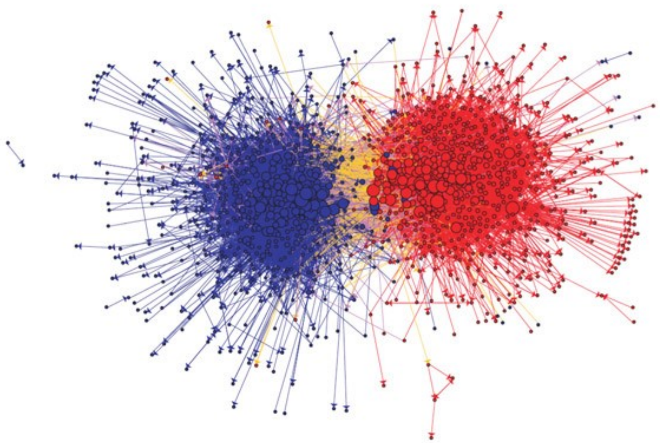
Social Signals

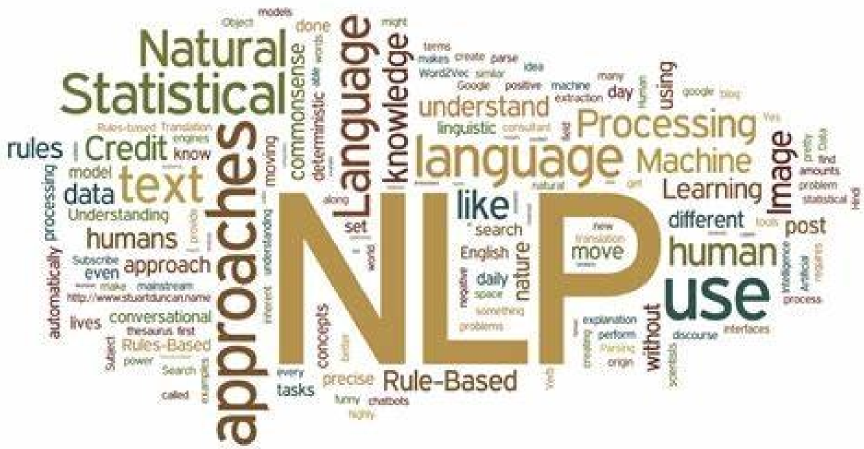
Computational Social Science

Why Computational Social Science?









Complex Network and Social Network Analysis

Introduction

Yunji Liang

2021-3-22

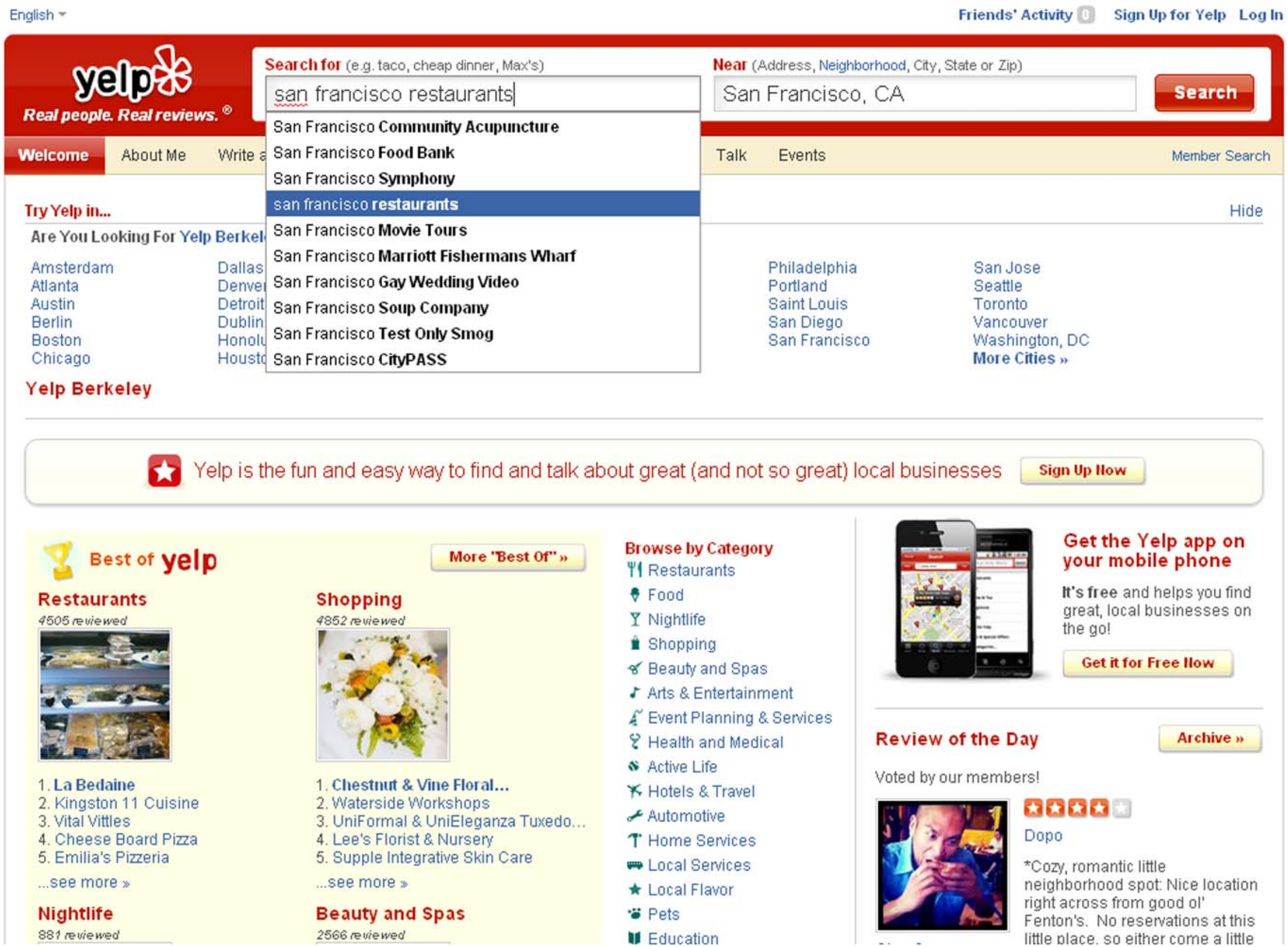
Facebook

- **How does Facebook use your data?**
- **Where do you think Facebook can use your data?**

Amazon

Amazon.com: Online Shoppi
www.amazon.com
amazon
Your Amazon.com
Today's Deals
Gift Cards
Help
FREE Two-Day Shipping
Join Amazon Prime Today
Shop by Department
Search
All
Go
Hello. Sign in Your Account
0 Cart
Wish List
Amazon Instant Video: The latest TV shows, available instantly.
EXPAND TO LEARN MORE
Instant Video
MP3 Store
Cloud Player
Kindle
Cloud Drive
Appstore for Android
Digital Games & Software
Audible Audiobooks
The All-New kindle fire HD
The ultimate HD experience
From $199 Shop now
Introducing kindle paperwhite
The world's most advanced e-reader
From $119 Shop now
Bikes with Street Cred
Clothing Trends
Amazon Prime
THE AMAZON CLOTHING STORE
DRESS SHOP
The season's biggest trends from Calvin Klein, BCBGMAXAZRIA, Tracy Reese, and more.
Shop Dresses
Shop All Clothing
Friends & Family Gifting
Gift Connections Made Easy
Learn more
$0
Hundreds of Free Songs
Play on the Web, Android, and Kindle Fire
Shop now
amazon MP3
Advertisement
Gold Box Deal of the Day
$1.99 Kindle Inspirational Memoirs and More

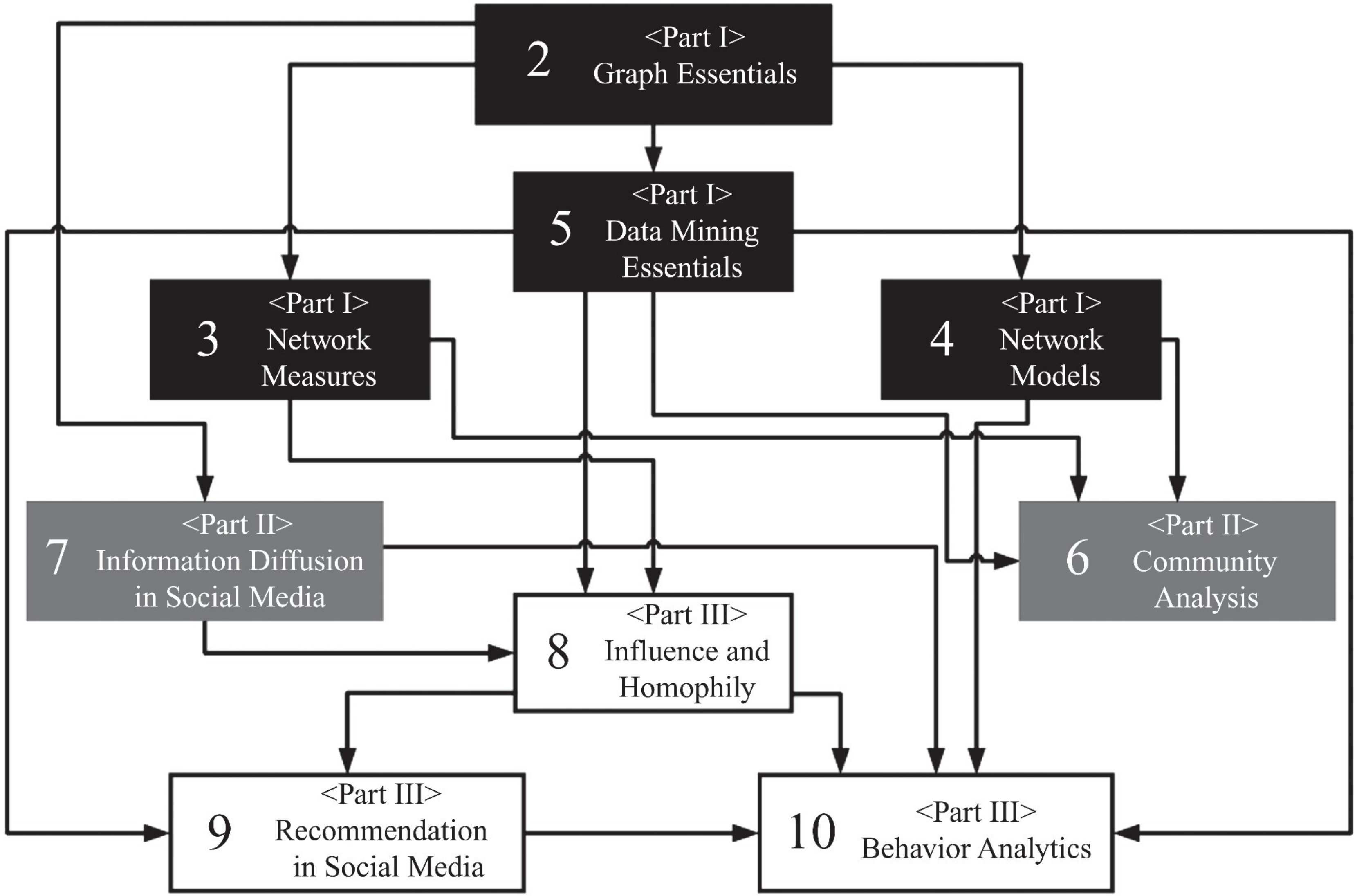
Yelp


English
Friends' Activity 0
Sign Up for Yelp
Log In
yelp
Real people. Real reviews.
Search for (e.g. taco, cheap dinner, Max's)
san francisco restaurants
Near (Address, Neighborhood, City, State or Zip)
San Francisco, CA
Search
San Francisco Community Acupuncture
San Francisco Food Bank
San Francisco Symphony
san francisco restaurants
San Francisco Movie Tours
San Francisco Marriott Fishermans Wharf
San Francisco Gay Wedding Video
San Francisco Soup Company
San Francisco Test Only Smog
San Francisco CityPASS
Welcome
About Me
Talk
Events
Member Search
Try Yelp in...
Hide
Amsterdam
Atlanta
Austin
Berlin
Boston
Chicago
Philadelphia
Portland
Saint Louis
San Diego
San Francisco
San Jose
Seattle
Toronto
Vancouver
Washington, DC
More Cities »
Yelp Berkeley
Yelp is the fun and easy way to find and talk about great (and not so great) local businesses
Sign Up Now
Best of yelp
More "Best Of" »
Restaurants
4505 reviewed
1. La Bedaine
2. Kingston 11 Cuisine
3. Vital Vittles
4. Cheese Board Pizza
5. Emilia's Pizzeria
...see more »
Shopping
4852 reviewed
1. Chestnut & Vine Floral...
2. Waterside Workshops
3. UniFormal & UniEleganza Tuxedo...
4. Lee's Florist & Nursery
5. Supple Integrative Skin Care
...see more »
Nightlife
881 reviewed
Beauty and Spas
2566 reviewed
Browse by Category
Restaurants
Food
Nightlife
Shopping
Beauty and Spas
Arts & Entertainment
Event Planning & Services
Health and Medical
Active Life
Hotels & Travel
Automotive
Home Services
Local Services
Local Flavor
Pets
Education
Get the Yelp app on your mobile phone
It's free and helps you find great, local businesses on the go!
Get it for Free Now
Review of the Day
Archive »
Voted by our members!
Dopo
"Cozy, romantic little neighborhood spot: Nice location right across from good ol' Fenton's. No reservations at this little place, so either come a little

Objectives of Our Course

- Understand social aspects of the Web
  - Social Theories + Social media + Mining
  - Learn how to collect, clean, and represent social media data
  - How to measure important properties of social media and simulate social media models
  - Find and analyze communities in social media
  - Understanding friendships in social media, perform recommendations, and analyze behavior
- Study or ask interesting research issues
  - e.g., start-up ideas
- Learn representative algorithms and tools

Overview – Dependency Graph



Social Media

Definition

Social Media is the use of electronic and Internet tools for the purpose of sharing and discussing information and experiences with other human beings in more efficient ways.

Social Media Landscape

Social Media Mining *is the process of representing, analyzing, and extracting meaningful patterns from social media data*

Social Media Mining Challenges

1. Big Data Paradox
   1. Social media data is big, yet not evenly distributed.
   2. Often little data is available for an individual

2. Obtaining Sufficient Samples
   1. Are our samples reliable representatives of the full data?

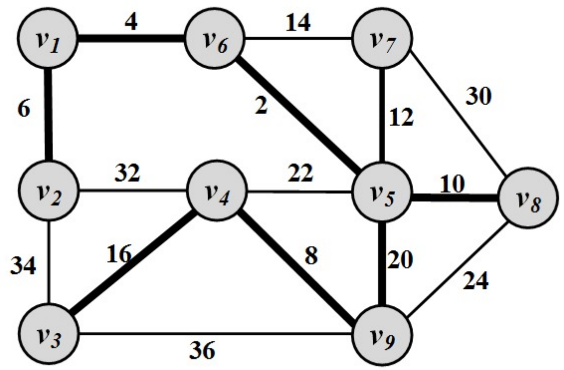
3. Noise Removal Fallacy
   1. Too much removal makes data more sparse
   2. Noise definition is relative and complicated and is task-dependent
4. Evaluation Dilemma
   1. When there is no ground truth, how can you evaluate?

Social Media Mining

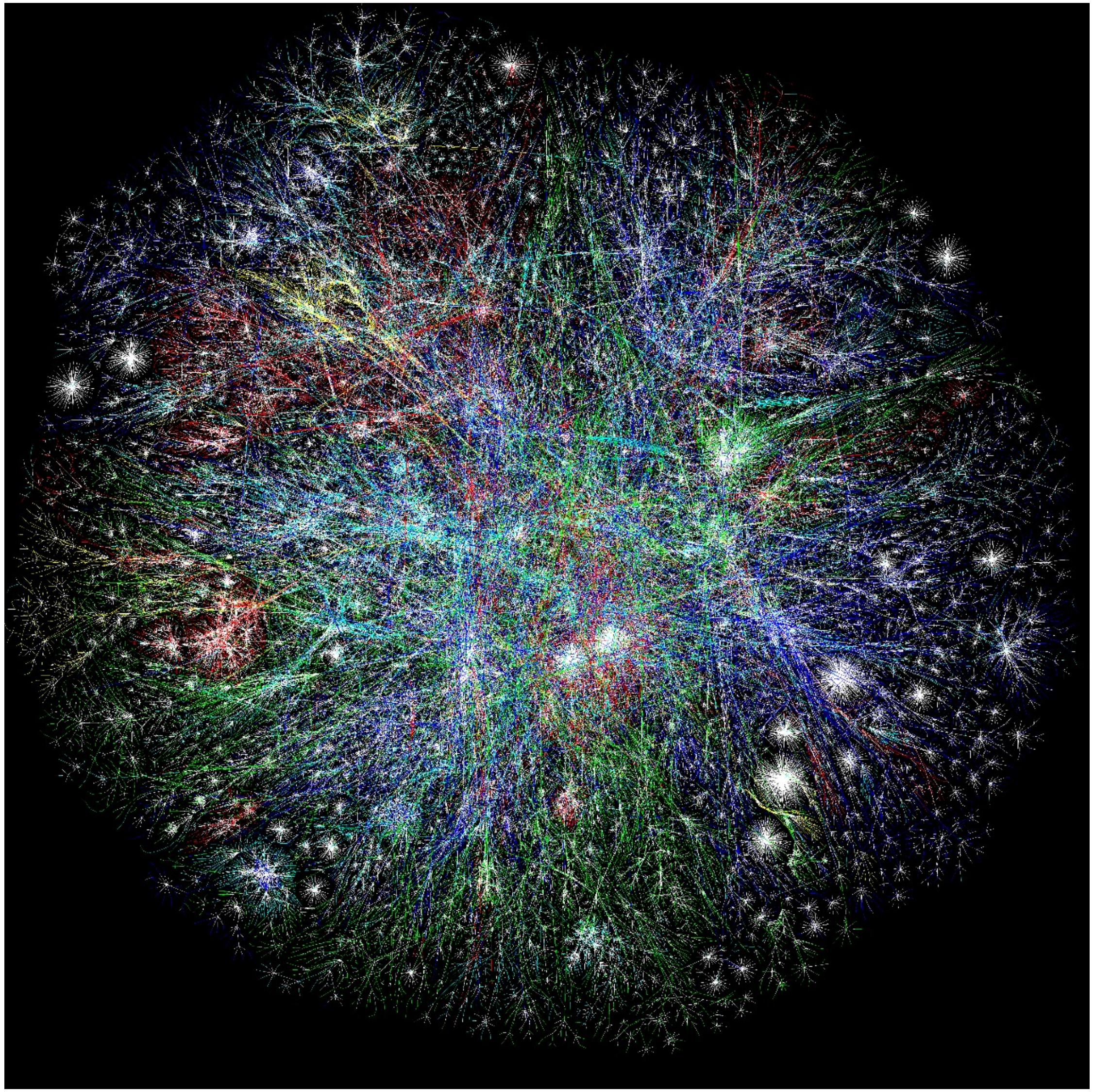
Graph Essentials

Networks

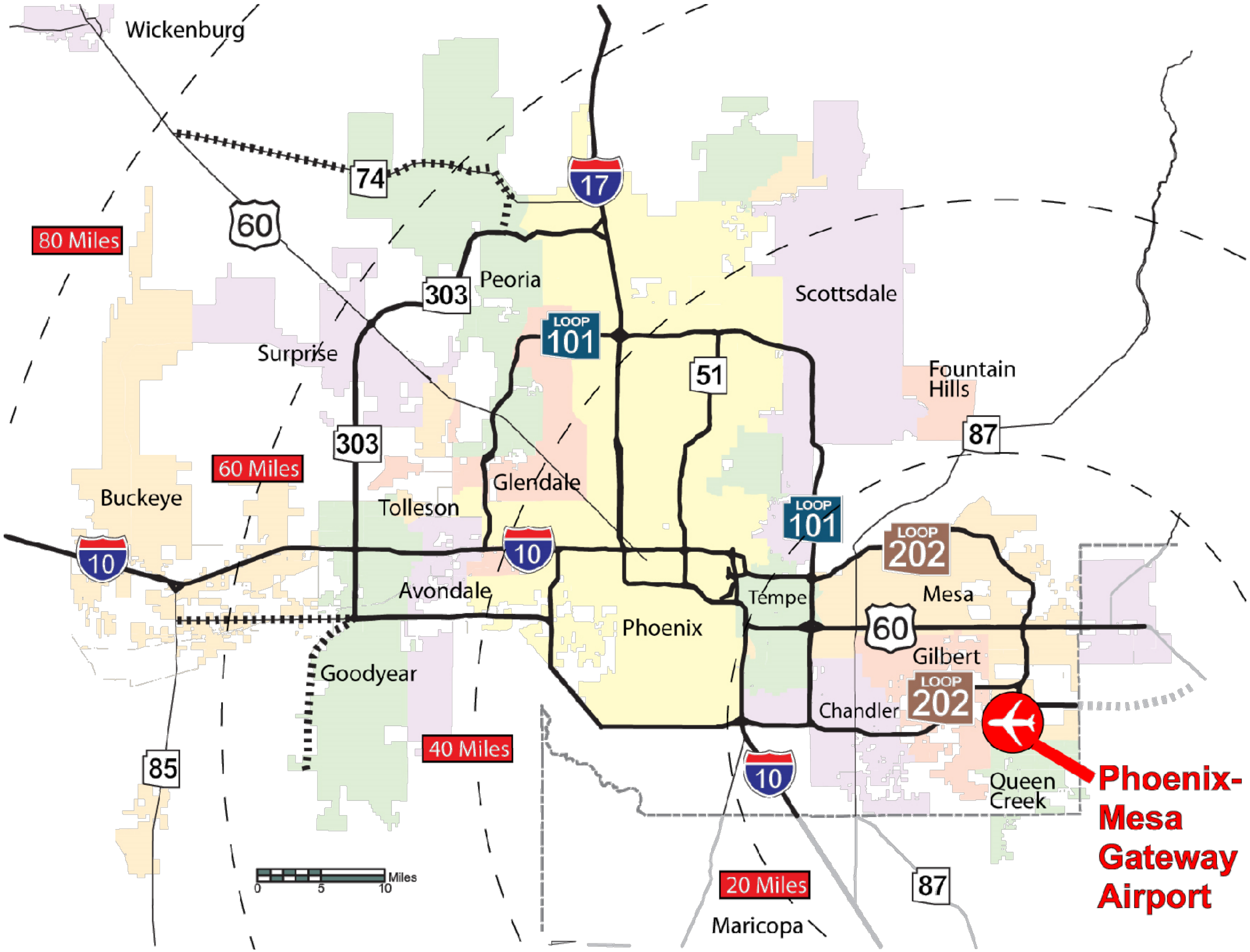
- A network is a graph.
  - Elements of the network have meanings
- Network problems can usually be represented in terms of graph theory
- Twitter example:
  - Given a piece of information, a network of individuals, and the cost to propagate information among any connected pair, find the minimum cost to disseminate the information to all individuals.



Internet



Phoenix Road Network



Social Networks and Social Network Analysis

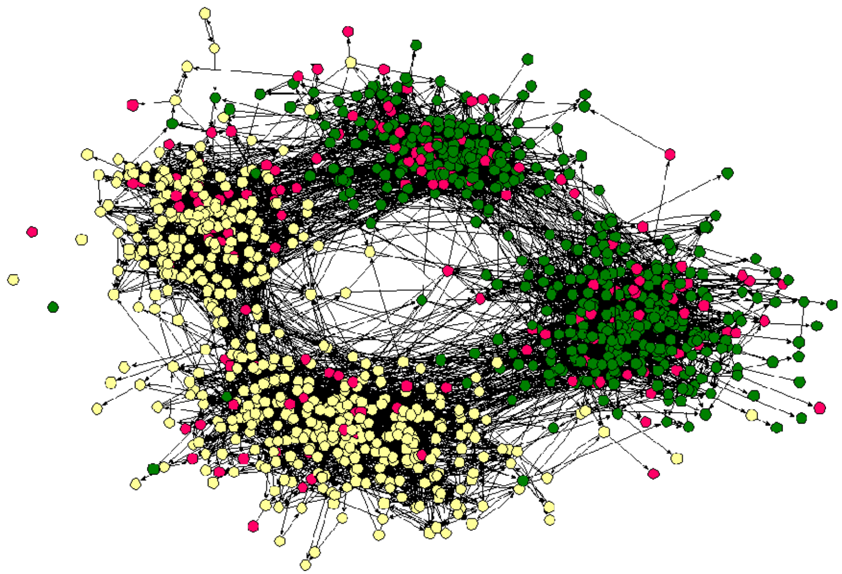
- A social network
  - A network where elements have a social structure
    - A set of actors (such as individuals or organizations)
    - A set of ties (connections between individuals)

- Social networks examples:
  - your family network, your friend network, your colleagues ,etc.

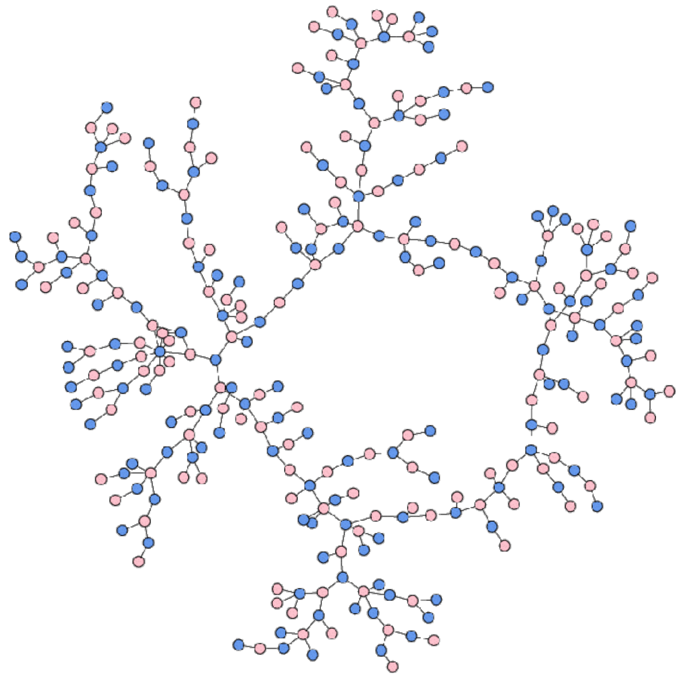
- To analyze these networks we can use Social Network Analysis (SNA)

- Social Network Analysis is an interdisciplinary field from social sciences, statistics, graph theory, complex networks, and now computer science

Social Networks: Examples



High school friendship



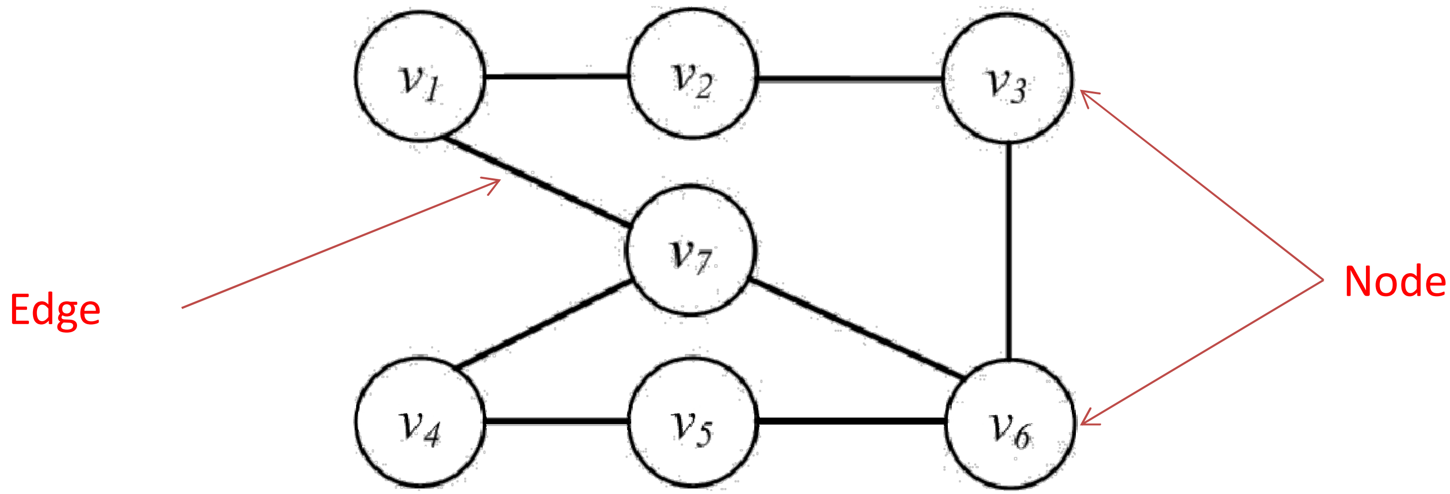
High school dating

Graph Basics

Nodes and Edges

A network is a graph, or a collection of points connected by lines

- Points are referred to as **nodes, actors,** or **vertices** (plural of **vertex**)
- Connections are referred to as **edges** or **ties**



Nodes or Actors

- In a friendship social graph, nodes are people and any pair of people connected denotes the friendship between them
- Depending on the context, these nodes are called nodes, or actors
  - In a web graph, “nodes” represent sites and the connection between nodes indicates web-links between them
  - In a social setting, these nodes are called actors

$$V = \{v_1, v_2, \ldots, v_n\}$$

  - The size of the graph is $|V| = \mathbf{n}$

Edges

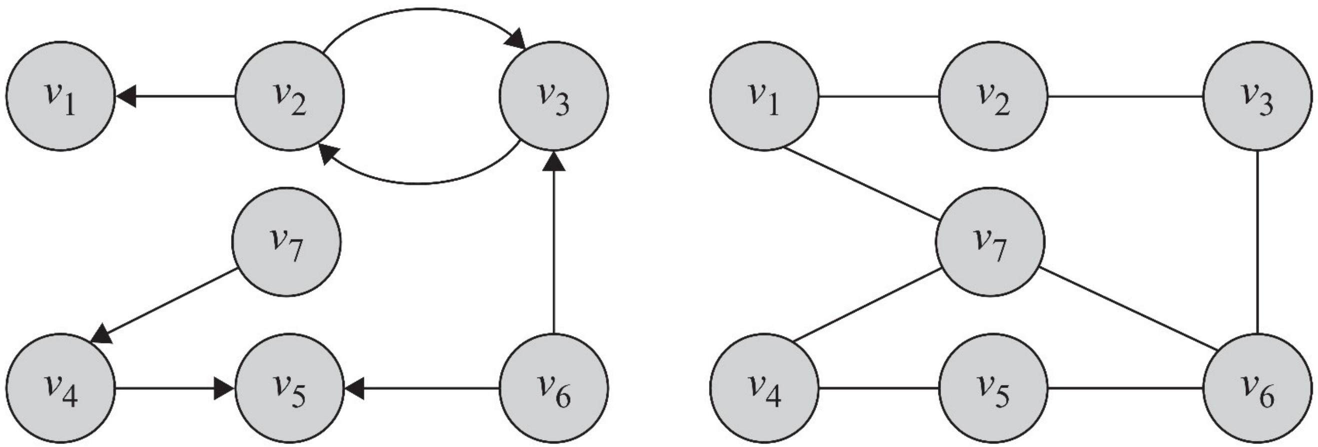
- Edges connect nodes and are also known as **ties** or **relationships**
- In a social setting, where nodes represent social entities such as people, edges indicate internode relationships and are therefore known as relationships or (social) ties

$$E = \{e_1, e_2, \ldots, e_m\}$$

- Number is edges (size of the edge-set) is denoted as |E|=**m**

Directed Edges and Directed Graphs

- Edges can have directions. A directed edge is sometimes called an arc



(a) Directed Graph (b) Undirected Graph

- Edges are represented using their end-points e(v2,v1). In undirected graphs both representations are the same

Neighborhood and Degree (In-degree, out-degree)

- For any node *v*, the set of nodes it is connected to via an edge is called its neighborhood and is represented as *N(v)*

- The number of edges connected to one node is the degree of that node (the size of its neighborhood)
  - Degree of a node *i* is usually presented using notation d_i
  - In case of directed graphs
    - d_i^{in} In-degrees is the number of edges pointing towards a node
    - d_i^{out} Out-degree is the number of edges pointing away from a node

Degree and Degree Distribution

- **Theorem 1.** The summation of degrees in an undirected graph is twice the number of edges

$$\sum_i d_i = 2|E|$$

- **Lemma 1.** The number of nodes with odd degree is even
- **Lemma 2.** In any directed graph, the summation of in-degrees is equal to the summation of out-degrees,

$$\sum_i d_i^{out} = \sum_j d_j^{in}$$

Degree Distribution

When dealing with very large graphs, how nodes' degrees are distributed is an important concept to analyze and is called ***Degree Distribution*** p_d

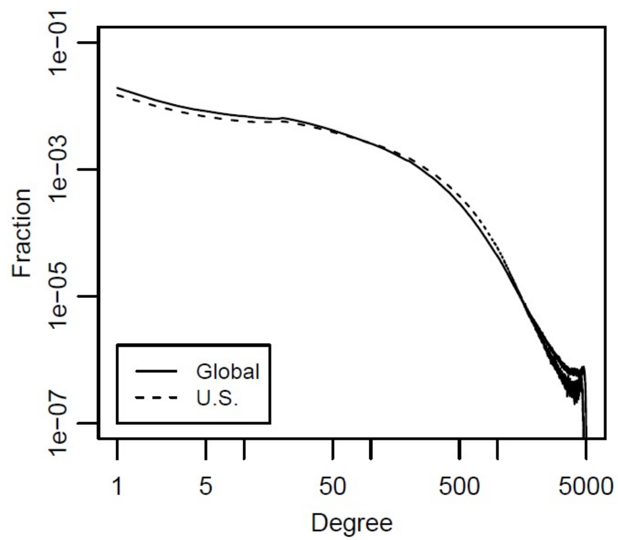
$$p_d = \frac{n_d}{n}$$

- Where n_d is the number of nodes with degree d
- Degree distribution can be computed from **degree sequence:**

$$\pi(d) = \{d_1, d_2, \ldots, d_n\}$$

Degree distribution histogram

– The x-axis represents the degree and the y-axis represents the number of nodes (frequency) having that degree



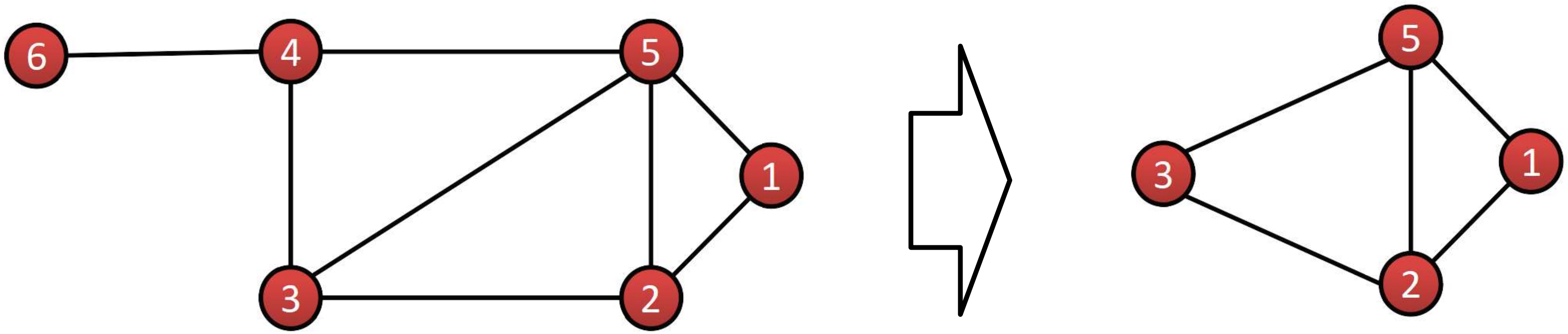
Subgraph

- Graph G can be represented as a pair G(V; E), where V is the node set and E is the edge set

- G'(V',E') is a subgraph of G(V,E)

$$V' \subseteq V,$$

$$E' \subseteq (V' \times V') \cap E$$



Graph Representation

- Adjacency Matrix
- Adjacency List
- Edge List

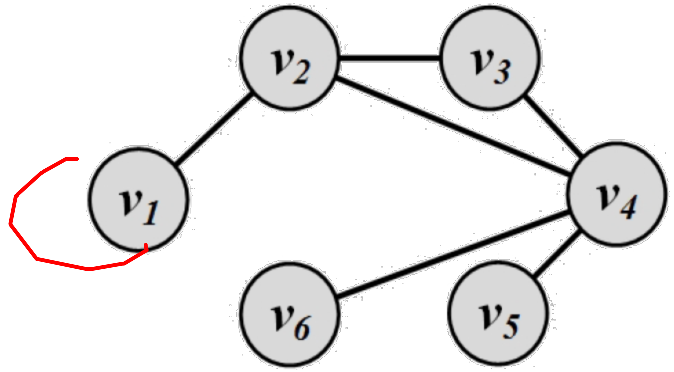
Graph Representation

- Graph representation is straightforward and intuitive, but it cannot be effectively manipulated using mathematical and computational tools

- We are seeking representations that can store these two sets in a way such that
  - Does not lose information
  - Can be manipulated easily by computers
  - Can have mathematical methods applied easily

Adjacency Matrix

$$A_{ij} = \begin{cases} 1, \text{ if there is an edge between nodes } vi \text{ and } vj \\ 0, \text{ otherwise} \end{cases}$$



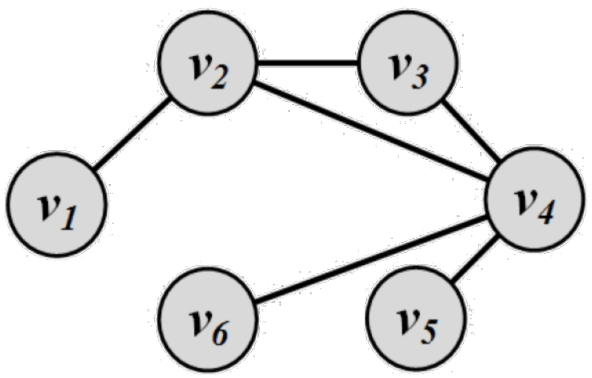
	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	1	0	0	0	0
v_2	1	0	1	1	0	0
v_3	0	1	0	1	0	0
v_4	0	1	1	0	1	1
v_5	0	0	0	1	0	0
v_6	0	0	0	1	0	0

- Diagonal Entries are self-links or loops

Social media networks have very sparse Adjacency matrices

Adjacency List

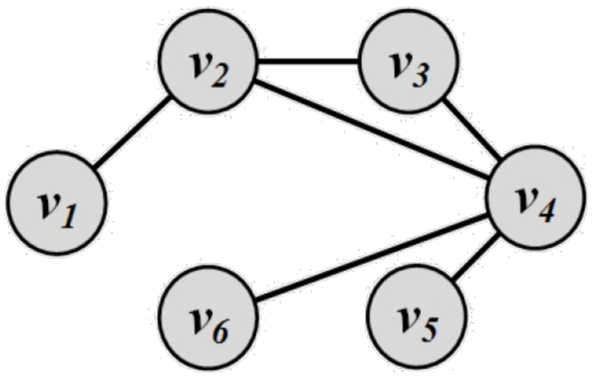
- In an adjacency list for every node, we maintain a list of all the nodes that it is connected to
- The list is usually sorted based on the node order or other preferences



Node	Connected To
v_1	v_2
v_2	v_1, v_3, v_4
v_3	v_2, v_4
v_4	v_2, v_3, v_5, v_6
v_5	v_4
v_6	v_4

Edge List

- In this representation, each element is an edge and is usually represented as *(u, v)*, denoting that node *u* is connected to node *v* via an edge



(v_1, v_2)
(v_2, v_3)
(v_2, v_4)
(v_3, v_4)
(v_4, v_5)
(v_4, v_6)

Types of Graphs

- Null, Empty, Directed/Undirected/Mixed, Simple/Multigraph, Weighted, Webgraph, Signed Graph

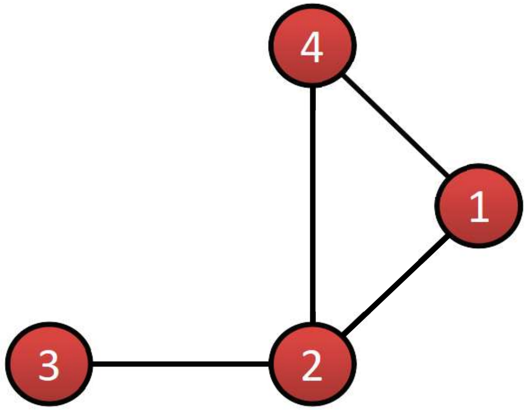
Null Graph and Empty Graph

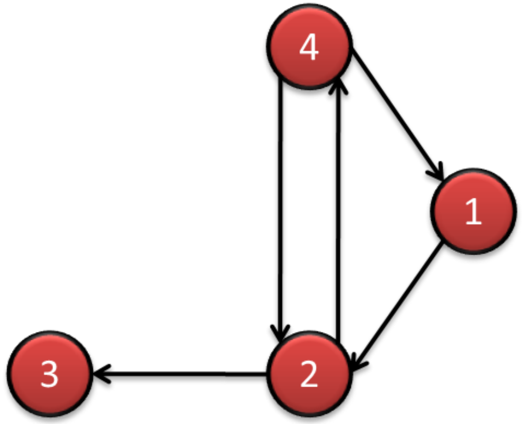
- A **null graph** is one where the node set is empty (there are no nodes)
  - Since there are no nodes, there are also no edges

$$G(V, E),\ \ V = E = \varnothing$$

- An **empty graph** or **edge-less graph** is one where the edge set is empty, $E = \varnothing$
  - The node set can be non-empty.
  - A null-graph is an empty graph.

Directed/Undirected/Mixed Graphs

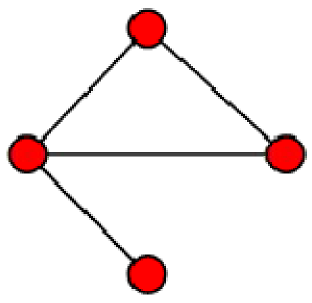




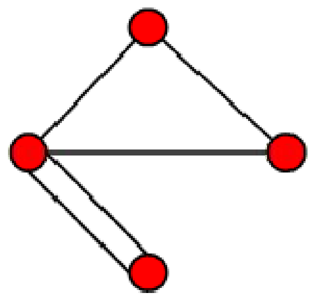
- The adjacency matrix for directed graphs is not symmetric ($A \neq A^T$)
  - ($A_{ij} \neq A_{ji}$)
- The adjacency matrix for undirected graphs is symmetric ($A = A^T$)

Simple Graphs and Multigraphs

- Simple graphs are graphs where only a single edge can be between any pair of nodes
- Multigraphs are graphs where you can have multiple edges between two nodes and loops



Simple graph



Multigraph

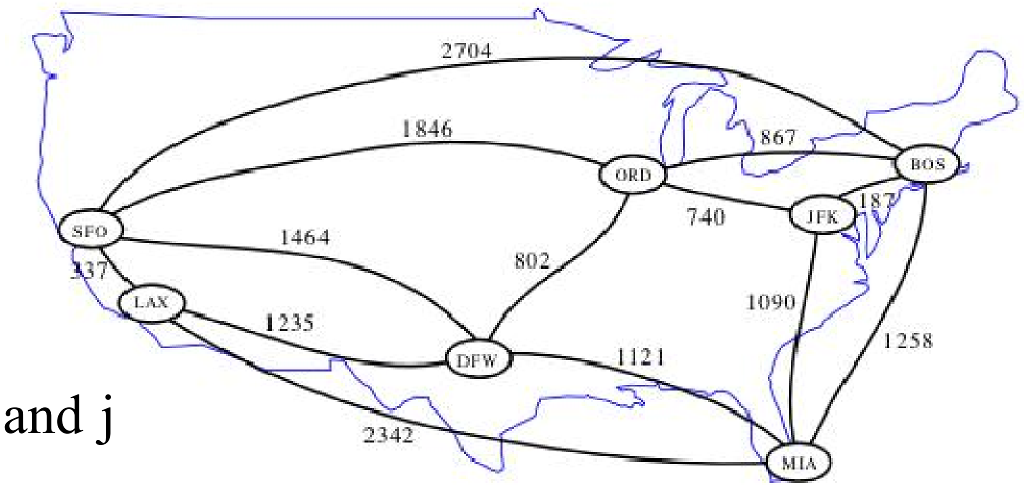
- The adjacency matrix for multigraphs can include numbers larger than one, indicating multiple edges between nodes

Weighted Graph

- A weighted graph is one where edges are associated with weights
  - For example, a graph could represent a map where nodes are cities and edges are routes between them
    - The weight associated with each edge could represent the distance between these cities

G(V, E, W)

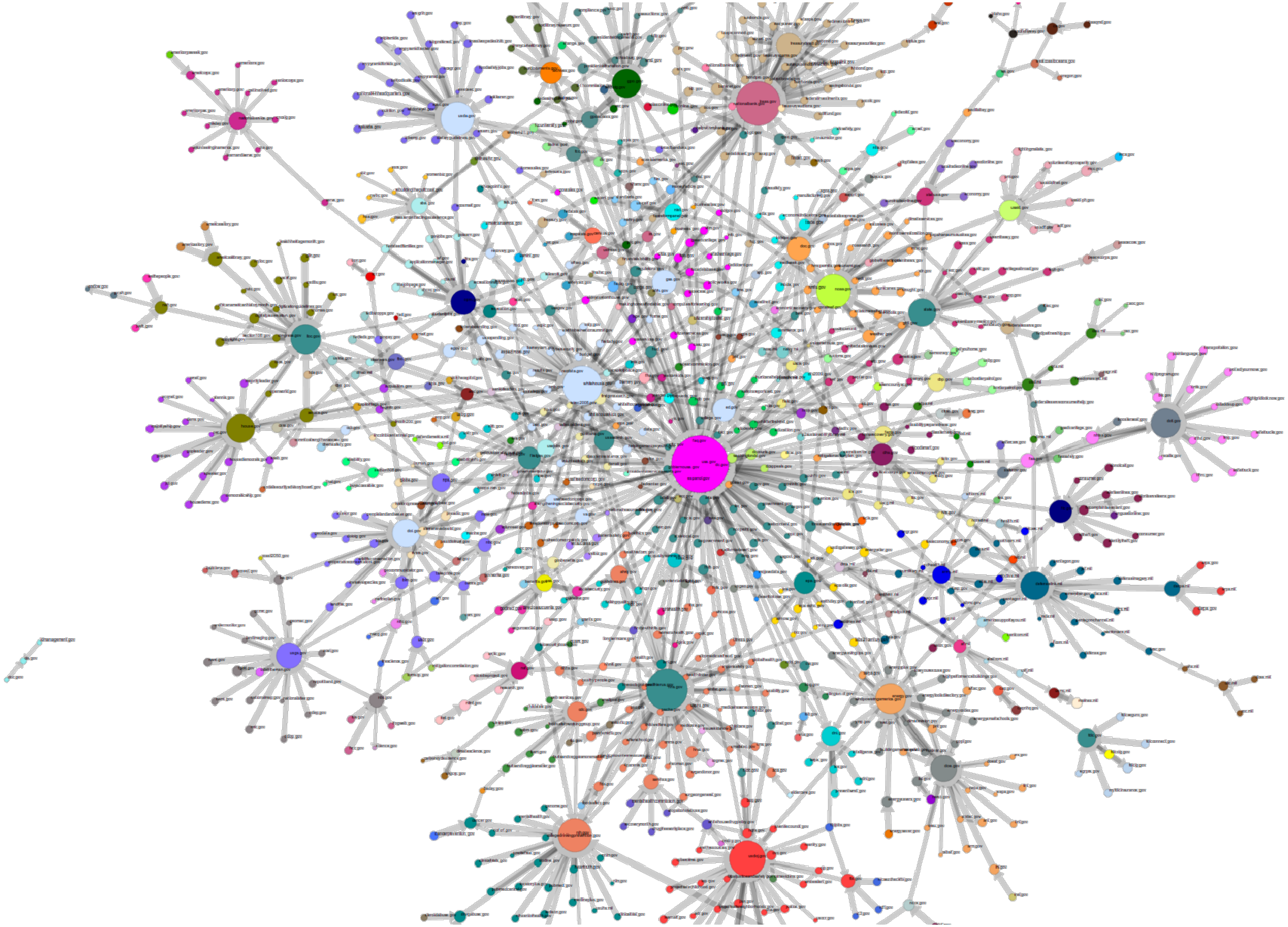
$$A_{ij} = \begin{cases} \mathrm{w}, \mathrm{w} \in \mathrm{R} \\ 0, \text{There is no edge between i and j} \end{cases}$$



Webgraph

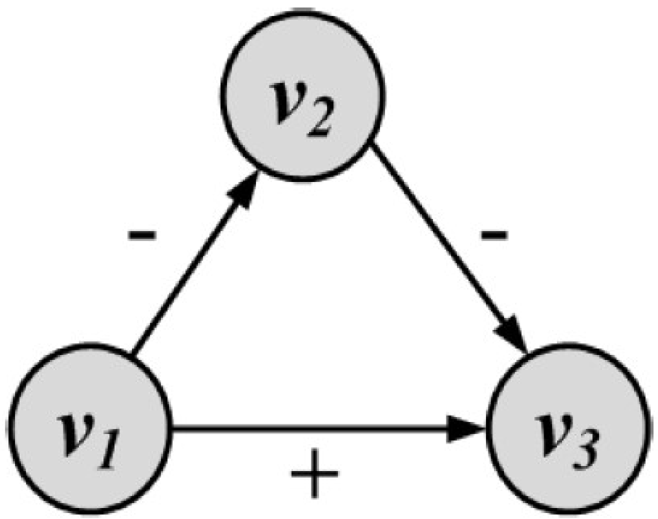
- A webgraph is a way of representing how internet sites are connected on the web
- In general, a web graph is a directed multigraph
- Nodes represent sites and edges represent links between sites.
- Two sites can have multiple links pointing to each other and can have loops (links pointing to themselves)

Webgraph: Government Agencies



Signed Graph

- When weights are binary (0/1, -1/1, +/-) we have a signed graph



- It is used to represent friends or foes
- It is also used to represent social status

Quiz

- Graph
- Graph Representation
- Degree
- Degree Distribution

哈尔滨
乌鲁木齐
长春
沈阳
呼和浩特
北京
朝鲜
平壤
天津
渤海
银川
太原
石家庄
首尔
西宁
济南
兰州
韩国
黄海
西安
州
南京
合肥
海
成都
武汉
拉萨
重庆
杭州
尔
南昌
长沙
东海
贵阳
福州
台北
昆明
孟加拉国
南宁
广州
澳门
香港
缅甸
河内

Quiz

City	# of Nodes
Xi’an	1
Beijing	2
ZhengZhou	3
Chengdu	4
GuangZhou	5
WuHan	6
Shanghai	7
XinJiang	8

	1	2	3	4	5	6	7	8
1	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
2	1	0	1	1	1	1	1	1
3	1	1	0	1	1	0	0	1
4	0	1	1	0	0	0	1	0
5	0	1	1	0	0	0	1	1
6	1	1	0	0	0	0	1	0
7	0	1	0	1	1	1	0	1
8	0	1	1	0	1	0	1	0

Quiz

	1	2	3	4	5	6	7	8
1	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
2	1	0	1	1	1	1	1	1
3	1	1	0	1	1	0	0	1
4	0	1	1	0	0	0	1	0
5	0	1	1	0	0	0	1	1
6	1	1	0	0	0	0	1	0
7	0	1	0	1	1	1	0	1
8	0	1	1	0	1	0	1	0

City	# of Nodes	Degree
Xi'an	1	3
Beijing	2	7
ZhengZhou	3	5
Chengdu	4	3
GuangZhou	5	3
WuHan	6	4
Shanghai	7	5
XinJiang	8	4

Quiz

City	# of Nodes	Degree
Xi'an	1	3
Beijing	2	7
ZhengZhou	3	5
Chengdu	4	3
GuangZhou	5	3
WuHan	6	4
Shanghai	7	5
XinJiang	8	4

$\{3, 4, 5, 7\}$

$\{3, 2, 2, 1\}$

$\{3/8, 2/8, 2/8, 1/8\}$

Connectivity in Graphs

- **Adjacent nodes/Edges, Walk/Path/Trail/Tour/Cycle,**

Adjacent nodes and Incident Edges

Two nodes are adjacent if they are connected via an edge.

Two edges are incident, if they share on end-point

When the graph is directed, edge directions must match for edges to be incident

An edge in a graph can be traversed when one starts

at one of its end-nodes, moves along the edge, and stops at its other end-node.

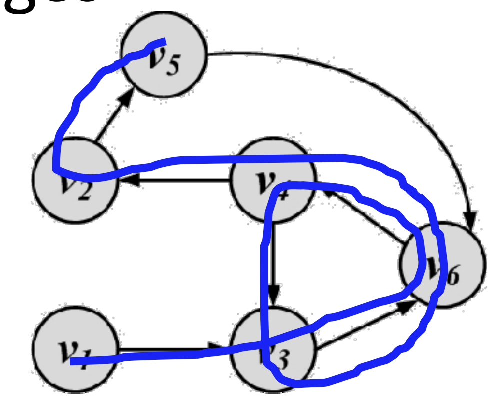
Walk, Path, Trail, Tour, and Cycle

Walk: A walk is a sequence of incident edges visited one after another

- **Open walk**: A walk does not end where it starts
- **Close walk**: A walk returns to where it starts

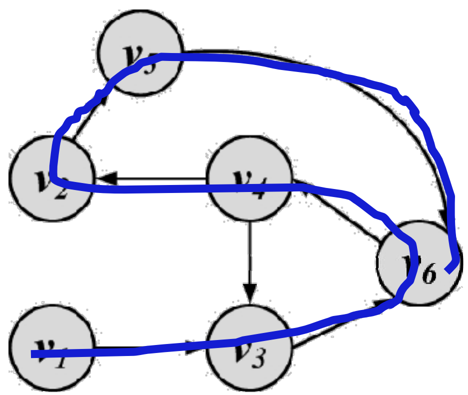
- Representing a walk:
  - A sequence of edges: $e_1, e_2, \ldots, e_n$
  - A sequence of nodes: $v_1, v_2, \ldots, v_n$
- Length of walk: the number of visited edges

Length of walk= 8



Trail

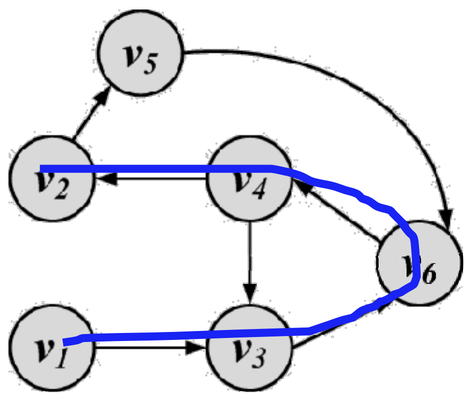
- A trail is a walk where **no edge** **is visited more than once** and all walk edges are distinct
- A closed trail (one that ends where it starts) is called a **tour** or **circuit**



Path

- A walk where **nodes and edges are distinct** is called a **path** and a closed path is called a **cycle**
- The length of a path or cycle is the number of edges visited in the path or cycle

Length of path= 4

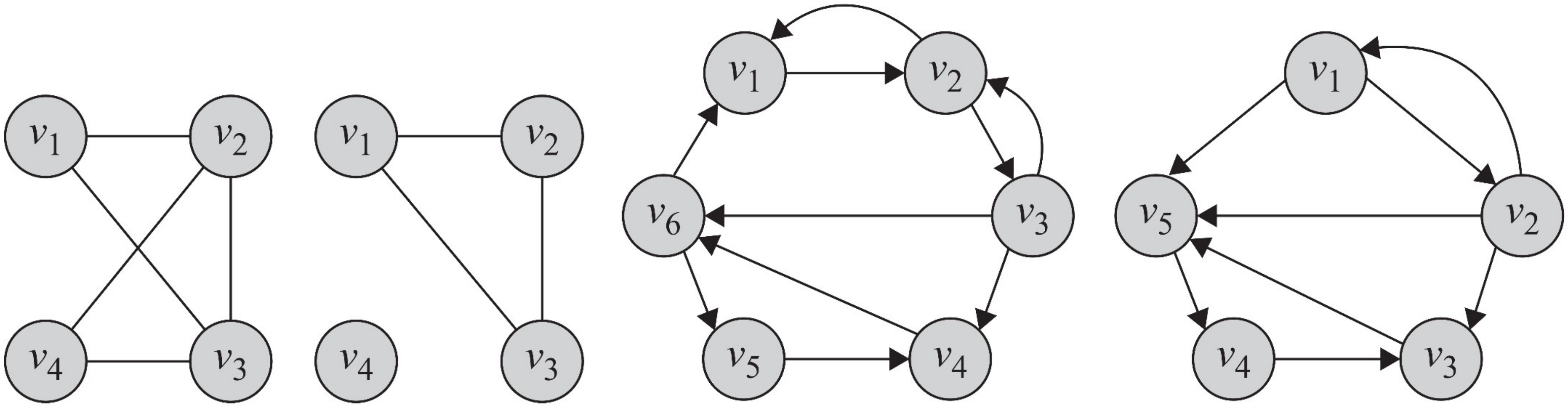


Connectivity

- **A node v_i is connected** to node v_j (or reachable from v_j) if it is adjacent to it or there exists a path from v_i to $v_{j.}$

- **A graph is connected**, if there exists a path between any pair of nodes in it
  - In a directed graph, **a graph is strongly connected** if there exists a directed path between any pair of nodes
  - In a directed graph, **a graph is weakly connected** if there exists a path between any pair of nodes, without following the edge directions
- A graph is **disconnected,** if it not connected.

Connectivity: Example



(a) Connected (b) Disconnected (c) Strongly connected (d) Weakly connected

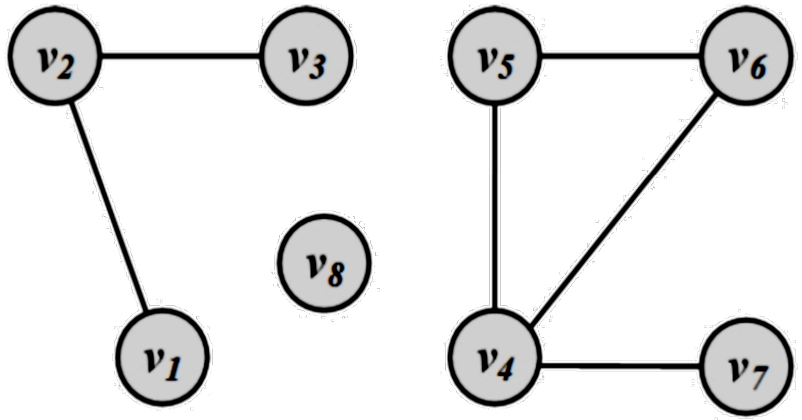
Component

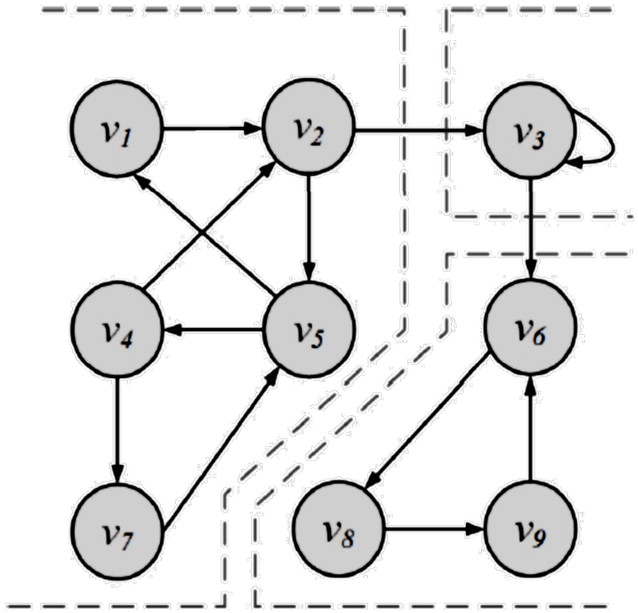
- A **component** in an undirected graph is a connected **subgraph**, i.e., there is a path between every pair of nodes inside the component

- In directed graphs, we have a **strongly connected** components when there is a path from *u* to *v* and one from *v* to *u* for every pair *(u,v).*

- The component is **weakly connected** if replacing directed edges with undirected edges results in a connected component

Component Examples:





3 components

3 Strongly-connected components

Shortest Path

- **Shortest Path** is the path between two nodes that has the shortest length.
  - We denote the length of the shortest path between nodes v_i and v_j as $l_{i,j}$

- The concept of the neighborhood of a node can be generalized using shortest paths. An **n-hop neighborhood** of a node is the set of nodes that are within n hops distance from the node.

Diameter

- The diameter of a graph is the length of the longest shortest path between any pair of nodes between any pairs of nodes in the graph

$$diameter_G = \max_{(v_i, v_j) \in V \times V} l_{i,j}.$$

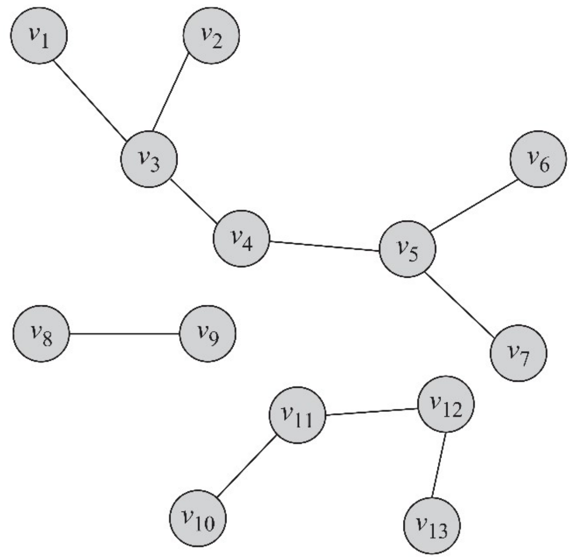
- How big is the diameter of the web?

Special Graphs

Trees and Forests

- **Trees** are special cases of undirected graphs
- A tree is a graph structure that has no cycle in it
- In a tree, there is exactly one path between any pair of nodes
- In a tree: |V| = |E| + 1

- A set of disconnected trees is called a **forest**

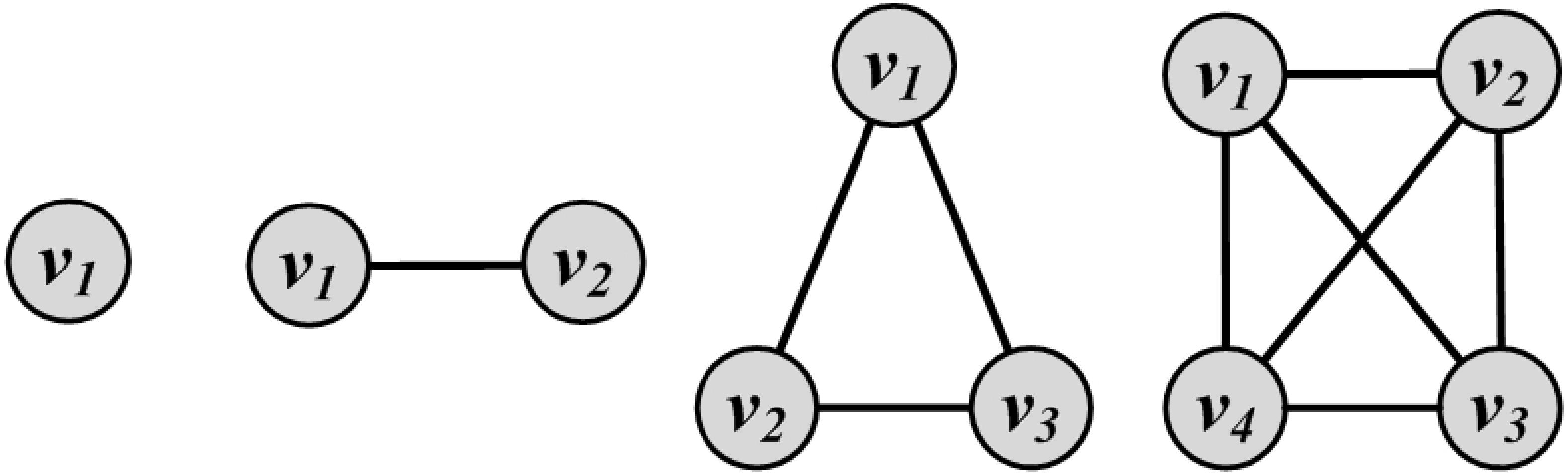


A forest containing 3 trees

Special Subgraphs

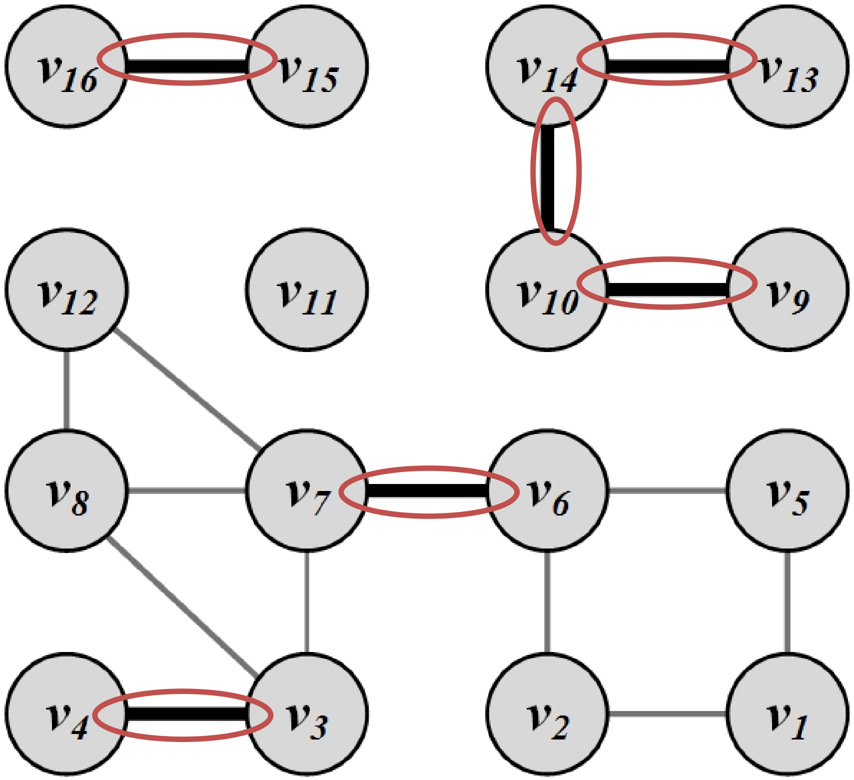
Complete Graphs

- A complete graph is a graph where for a set of nodes V, all possible edges exist in the graph
- In a complete graph, any pair of nodes are connected via an edge



Bridges (cut-edges)

- Bridges are edges whose removal will increase the number of connected components



Graph Algorithms

Graph/Network Traversal Algorithms

Graph/Tree Traversal

- Consider a social media site that has many users and we are interested in surveying the site and computing the average age of its users. The usual technique is to start from one user and employ some traversal technique to browse his friends and then these friends' friends and so on. The traversal technique guarantees that
- 1. All users are visited; and
- 2. No user is visited more than once.

- There are two main techniques:
  - **Depth-First Search (DFS)**
  - **Breadth-First Search (BFS)**

Depth-First Search (DFS)

- Depth-First Search (DFS) starts from a node i, selects one of its neighbors j from N(i) and performs Depth-First Search on j before visiting other neighbors in N(i).

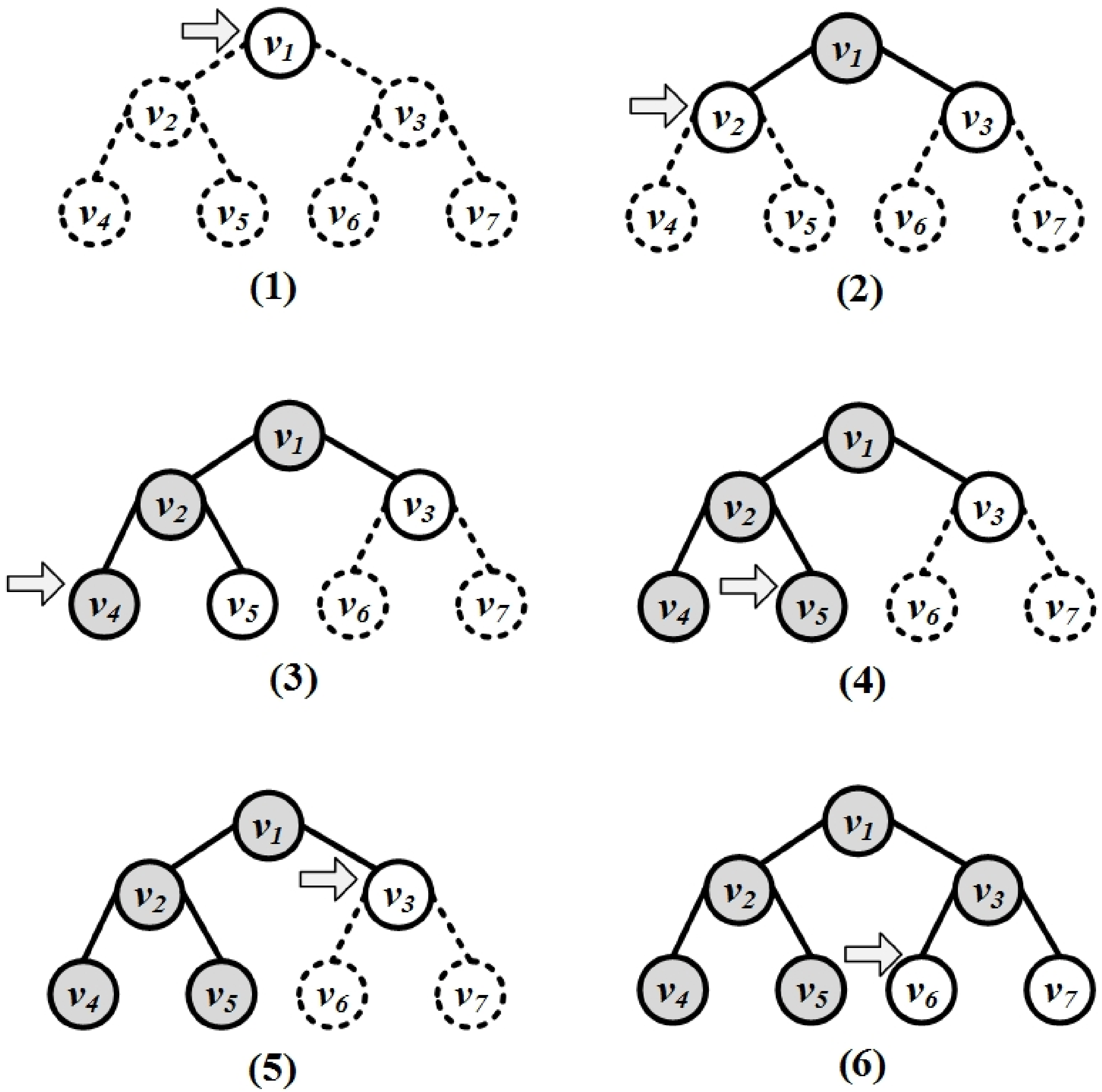
- The algorithm can be used both for trees and graphs
  - The algorithm can be implemented using a stack structure

DFS Algorithm

Algorithm 2.2 Depth-First Search (DFS)

```
Require: Initial node v, graph/tree G:(V, E), stack S
return An ordering on how nodes in G are visited
Push v into S;
visitOrder = 0;
while S not empty do
    node = pop from S;
    if node not visited then
        visitOrder = visitOrder +1;
        Mark node as visited with order visitOrder; //or print node
        Push all neighbors/children of node into S;
    end if
end while
Return all nodes with their visit order.
```

Depth-First Search (DFS): An Example



Breadth-First Search (BFS)

- BFS starts from a node, visits all its immediate neighbors first, and then moves to the second level by traversing their neighbors.
- The algorithm can be used both for trees and graphs
  - The algorithm can be implemented using a queue structure

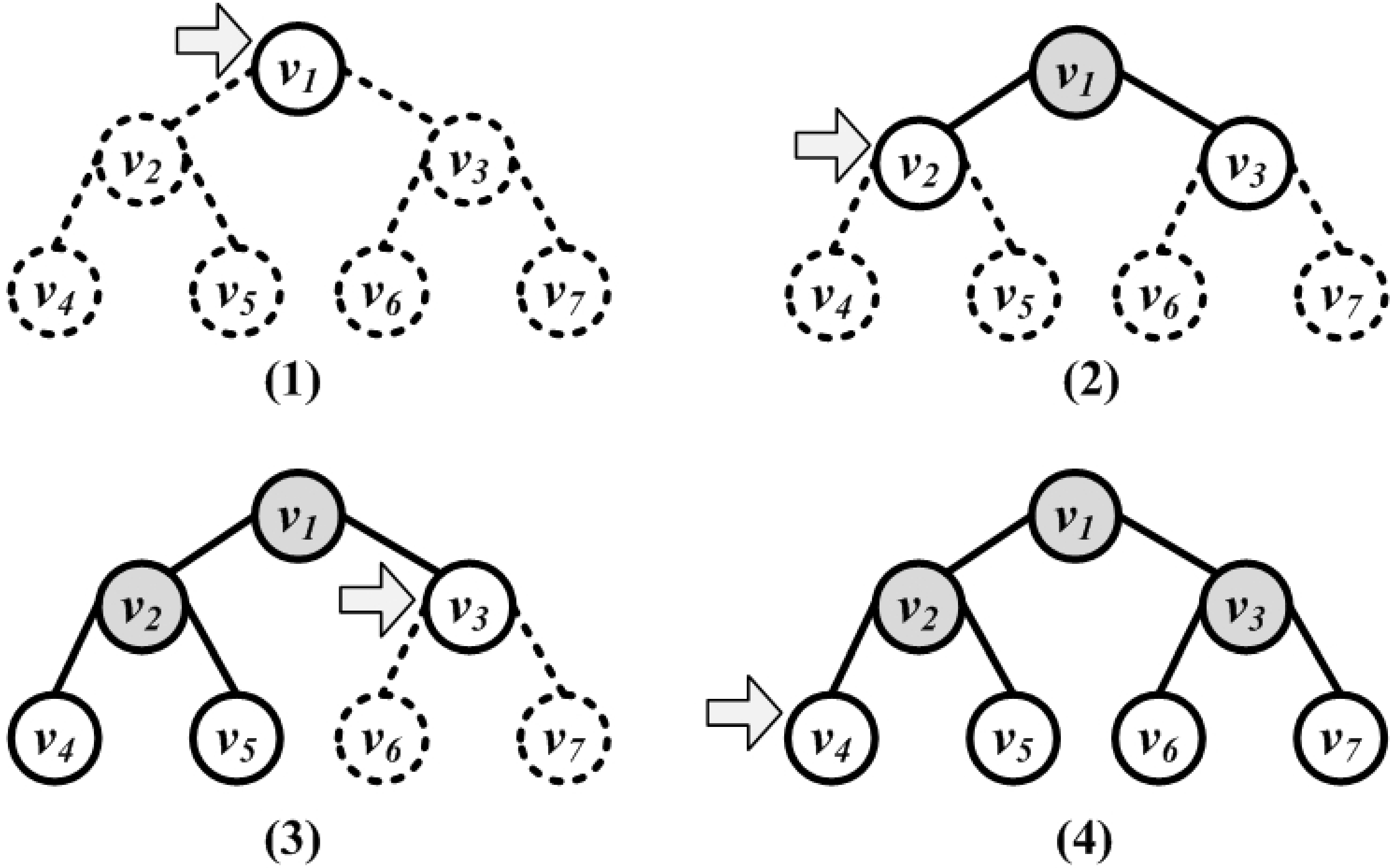
BFS Algorithm

Algorithm 2.3 Breadth-First Search (BFS)

Require: Initial node v, graph/tree $G(V, E)$, queue Q

return An ordering on how nodes are visited
Enqueue v into queue Q;
$visitOrder = 0$;
while Q not empty **do**
  $node$ = dequeue from Q;
  if $node$ not visited **then**
    $visitOrder = visitOrder + 1$;
    Mark $node$ as visited with order $visitOrder$; `//or print` $node$
    Enqueue all neighbors/children of $node$ into Q;
  end if
end while

Breadth-First Search (BFS)



Shortest Path

When a graph is connected, there is a chance that multiple paths exist between any pair of nodes

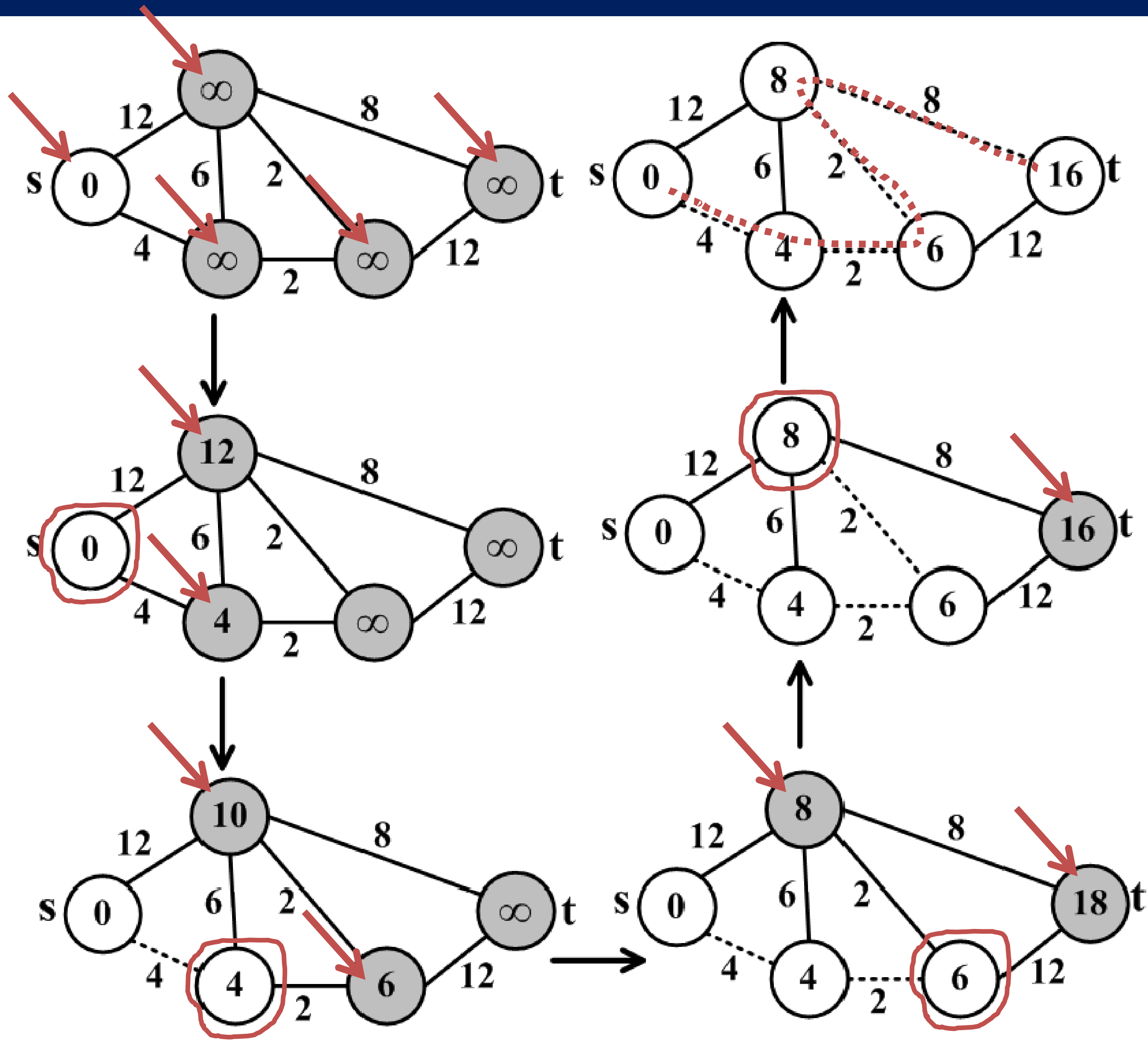
- In many scenarios, we want the shortest path between two nodes in a graph

- **Dijkstra's Algorithm**
  - It is designed for weighted graphs with non-negative edges
  - It finds shortest paths that start from a provided node *s* to all other nodes
  - It finds both shortest paths and their respective lengths

Dijkstra’s Algorithm: Finding the shortest path

1. Initiation:
   – Assign zero to the source node and infinity to all other nodes
   – Mark all nodes unvisited
   – Set the source node as current
2. For the current node, consider all of its unvisited neighbors and calculate their *tentative* distances
   – If tentative distance (current node’s distance + edge weight) is smaller than neighbor’s distance, then Neighbor’s distance = tentative distance
3. After considering all of the neighbors of the current node, mark the current node as visited and remove it from the *unvisited set*
   – A visited node will never be checked again and its distance recorded now is final and minimal
4. If the destination node has been marked visited or if the smallest tentative distance among the nodes in the *unvisited set* is infinity, then stop
5. Set the unvisited node marked with the smallest tentative distance as the next "current node" and go to step 2

Dijkstra's Algorithm Execution Example



Dijkstra’s Algorithm

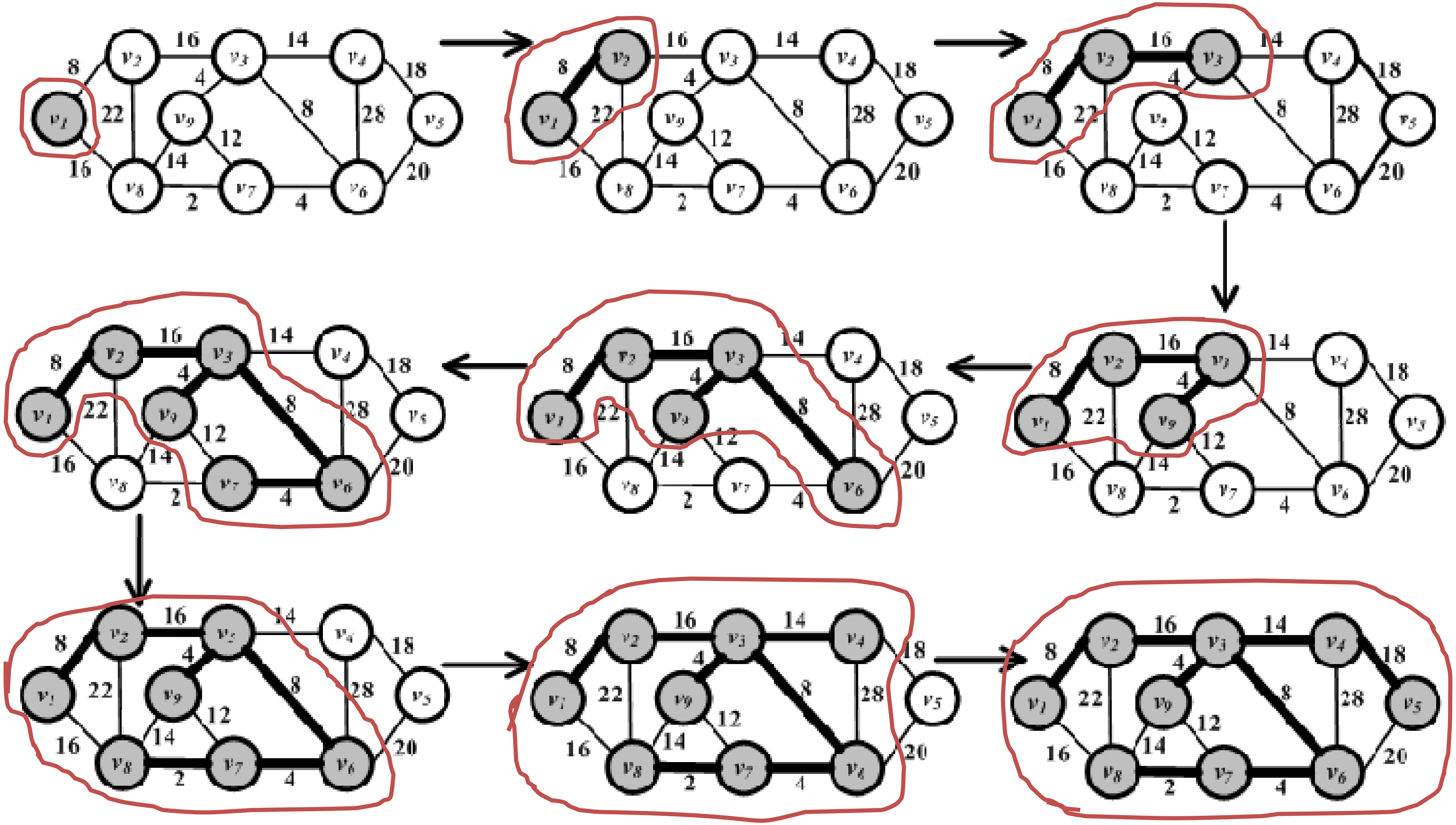
- Dijkstra’s algorithm is source-dependent and finds the shortest paths between the source node and all other nodes. To generate all-pair shortest paths, one can run dijsktra’s algorithm *n* times or use other algorithms such as Floyd-Warshall algorithm.

- If we want to compute the shortest path from source *v* to destination *d*, we can stop the algorithm once the shortest path to the destination node has been determined

Prim’s Algorithm: Finding Minimum Spanning Tree

- It finds minimal spanning trees in a weighted graph
  - It starts by selecting a random node and adding it to the spanning tree.
  - It then grows the spanning tree by selecting edges which have one endpoint in the existing spanning tree and one endpoint among the nodes that are not selected yet. Among the possible edges, the one with the minimum weight is added to the set (along with its end-point).
  - This process is iterated until the graph is fully spanned

Prim's Algorithm Execution Example



Bridge Detection

Algorithm 2.7 Bridge Detection Algorithm

Require: Connected graph $G(V, E)$

return Bridge Edges
$bridgeSet = \{\}$
for $e(u, v) \in E$ **do**
    G' = Remove e from G
    Disconnected = False;
    if BFS in G' starting at u does not visit v **then**
        Disconnected = True;
    end if
    if Disconnected **then**
        $bridgeSet = bridgeSet \cup \{e\}$
    end if
end for
Return $bridgeSet$