lorenzodallamico.github.io

lorenzo.dallamico@unito.it

di.unito.it

Lorenzo Dall'Amico - Università degli Studi di Torino (Italy)

Lecture 03.ns02

Recap on graphs

Course: **Complex Networks Analysis and Visualization**
Sub-Module: **NetSci**

-

Lecture 2 - Agenda

- Basic definitions on graphs
- Paths and Connectivity
- Small world phenomenon (mention)

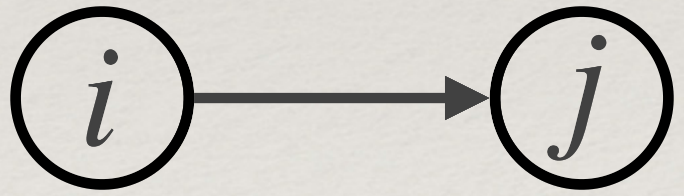
Why graphs?

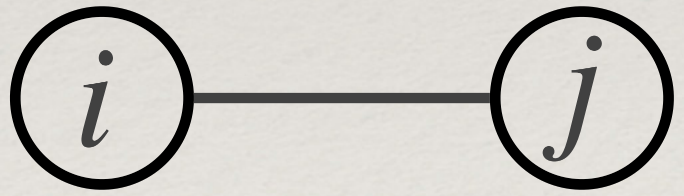
- **Graphs** are a **mathematical model** to represent interacting systems
- We need **a language** to understand the basic **elements of a network**
- With a language, we will be able to talk properly about:
    - properties that characterize **structure** and **behavior** of networks
    - roles of networks in **affecting processes** occurring on network structures

Basic definitions

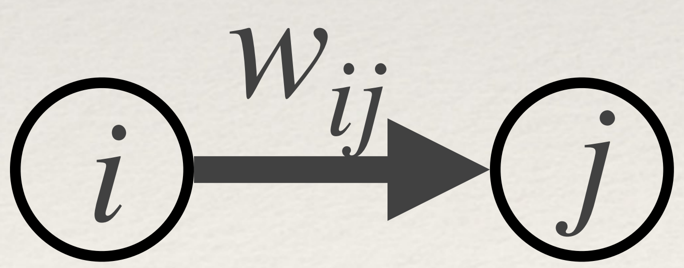
Basic Definitions

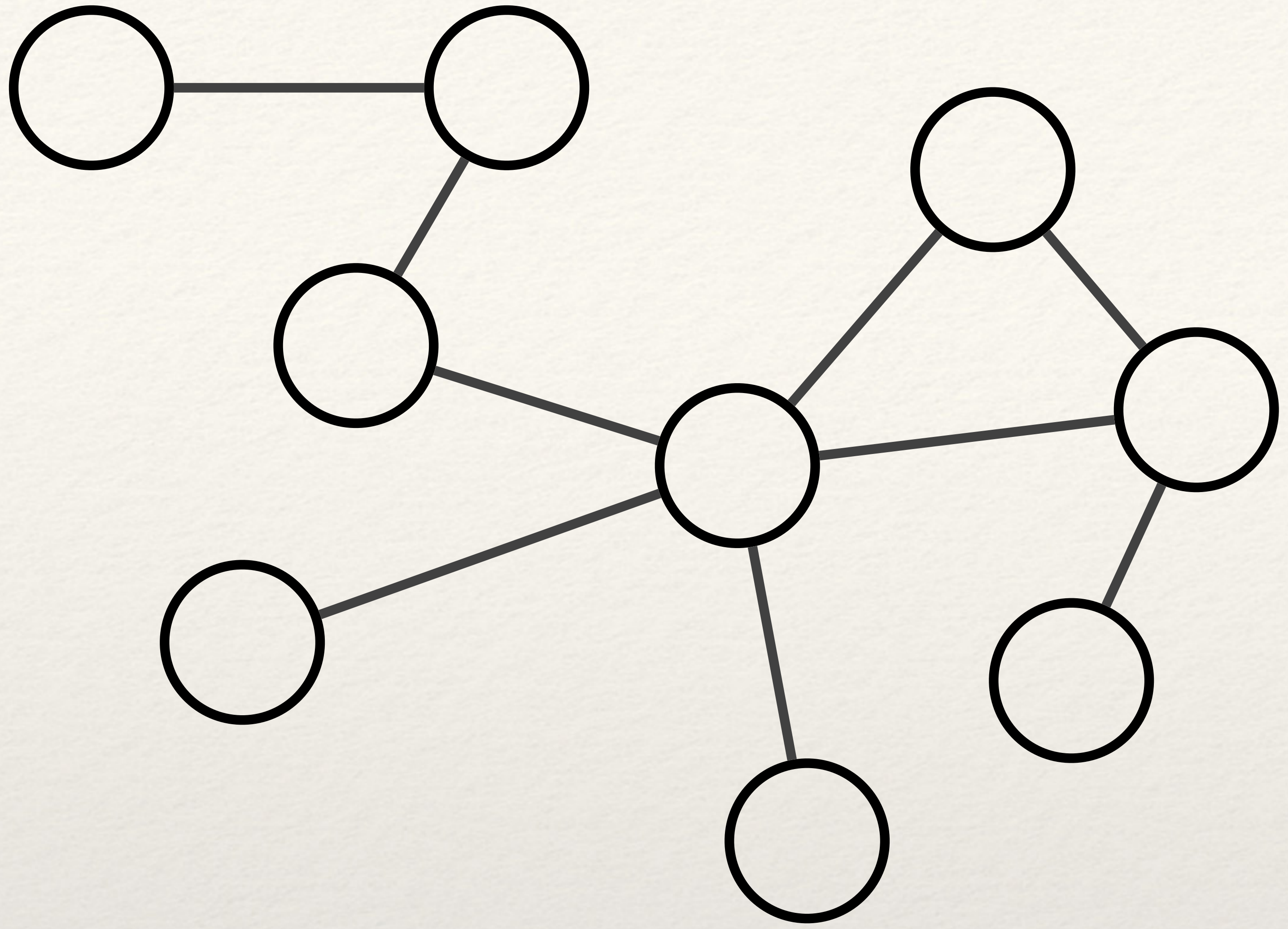
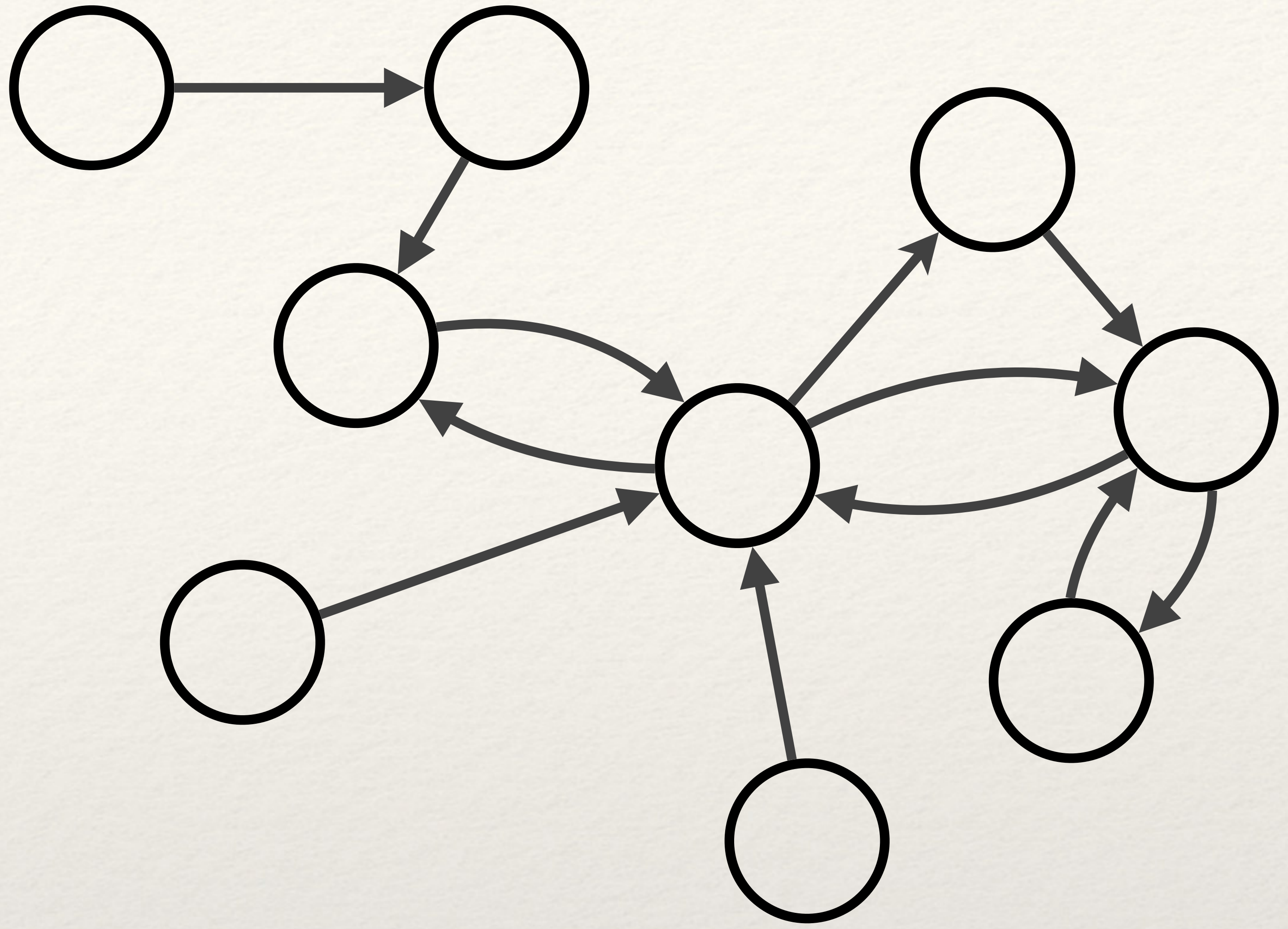
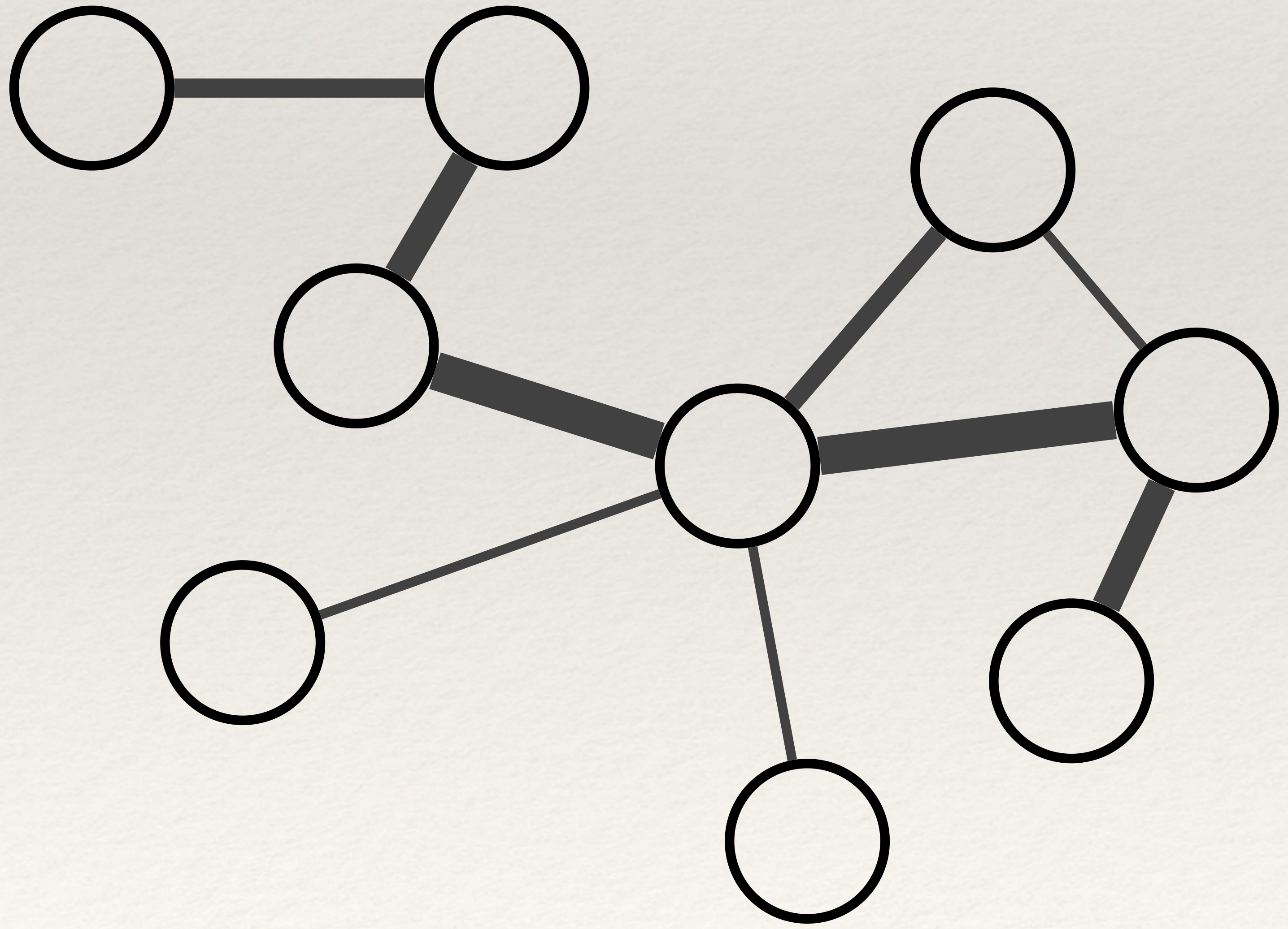
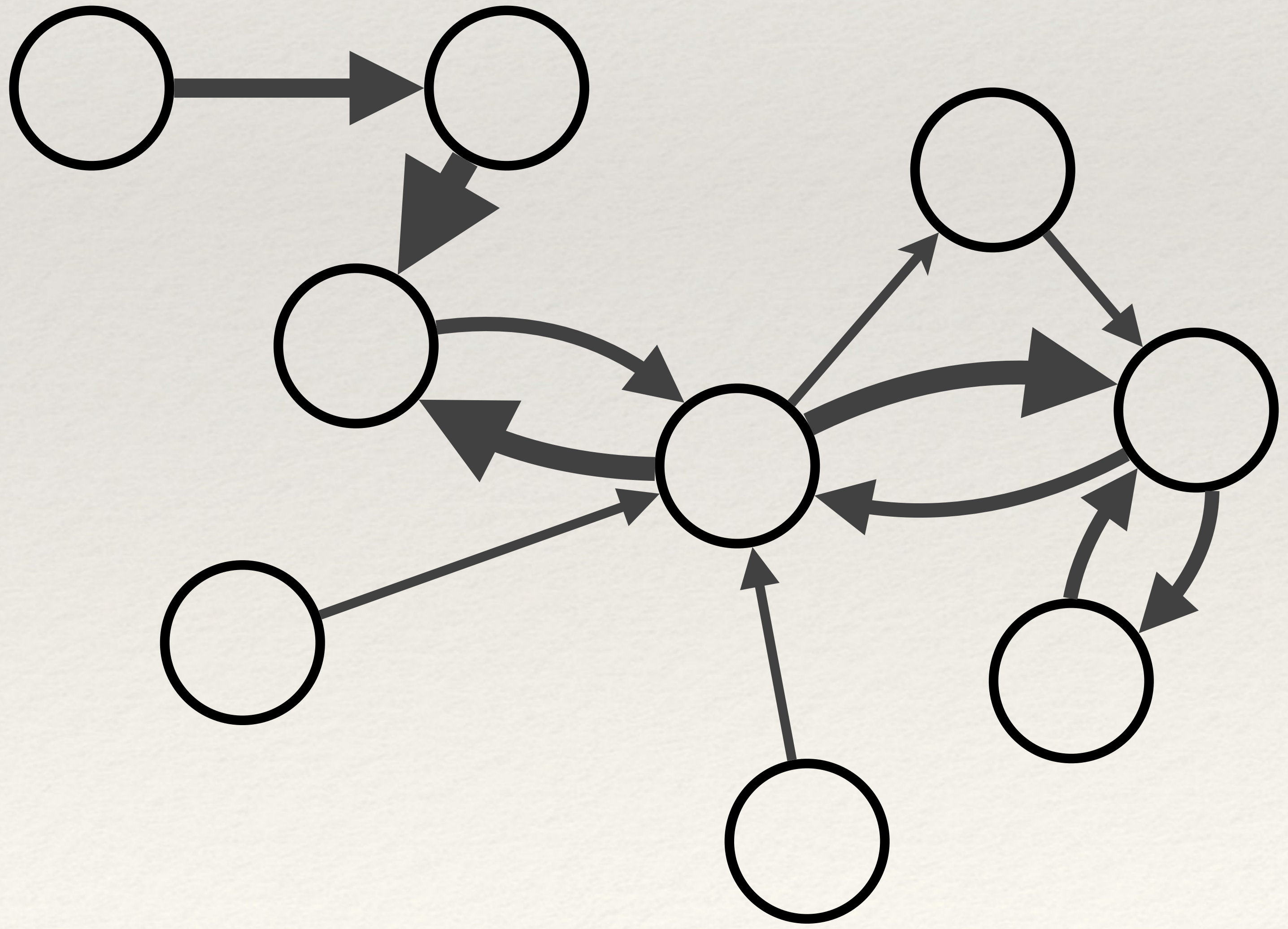
- A **graph** is made of nodes and links $G = (N, L)$
- **nodes** (or vertices) $N = \{n_1, n_2, \ldots, n_l\} = \{1,2,\ldots,l\}$
- **links** (or edges, or arcs) $L = \{(i,j) : i,j \in N\}$
- Graphs can be **directed** or **undirected** (i,j)



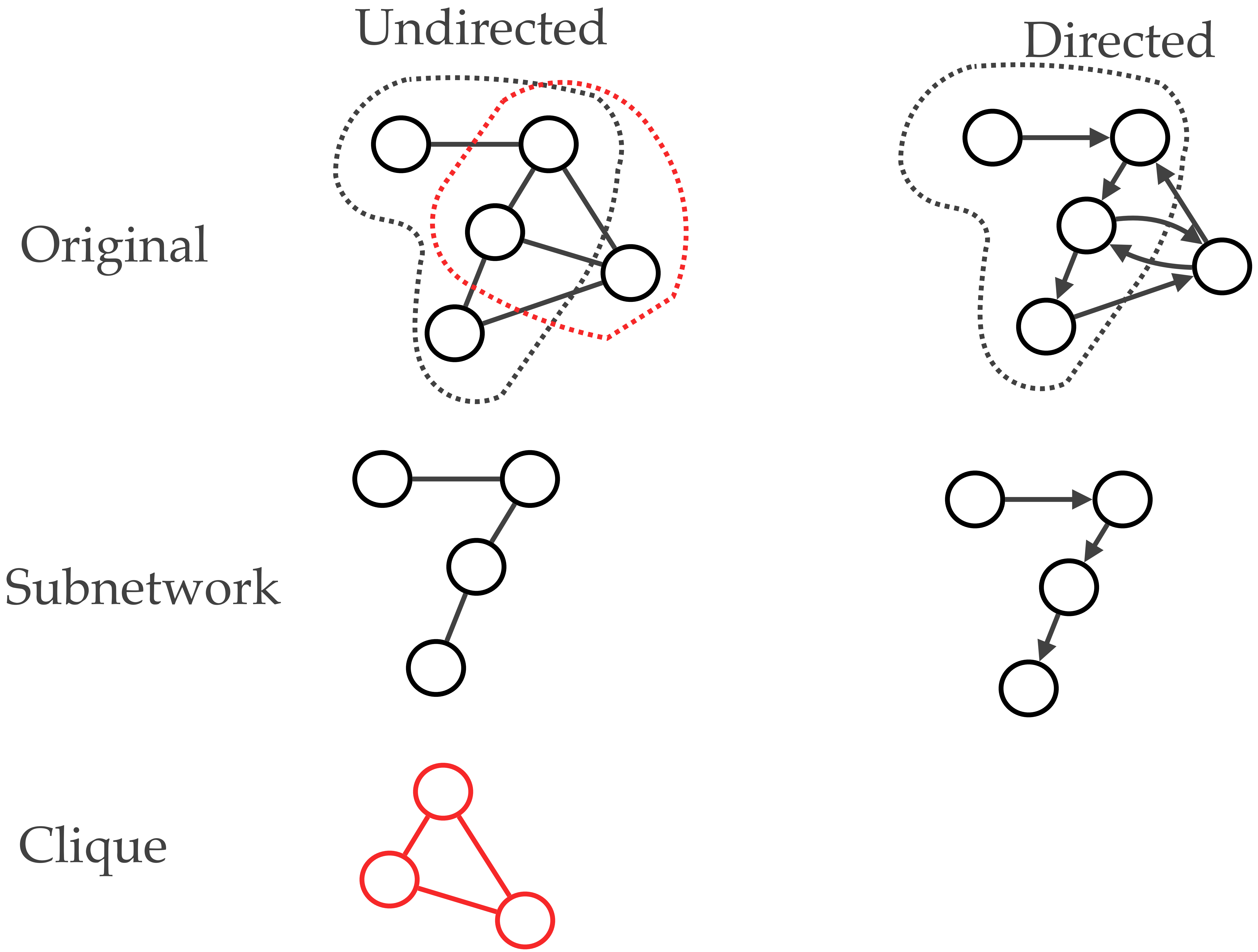


- Graphs can be **weighted** or **unweighted** w_{ij} (i, j, w)



	Undirected	Directed
Unweighted		
Weighted		

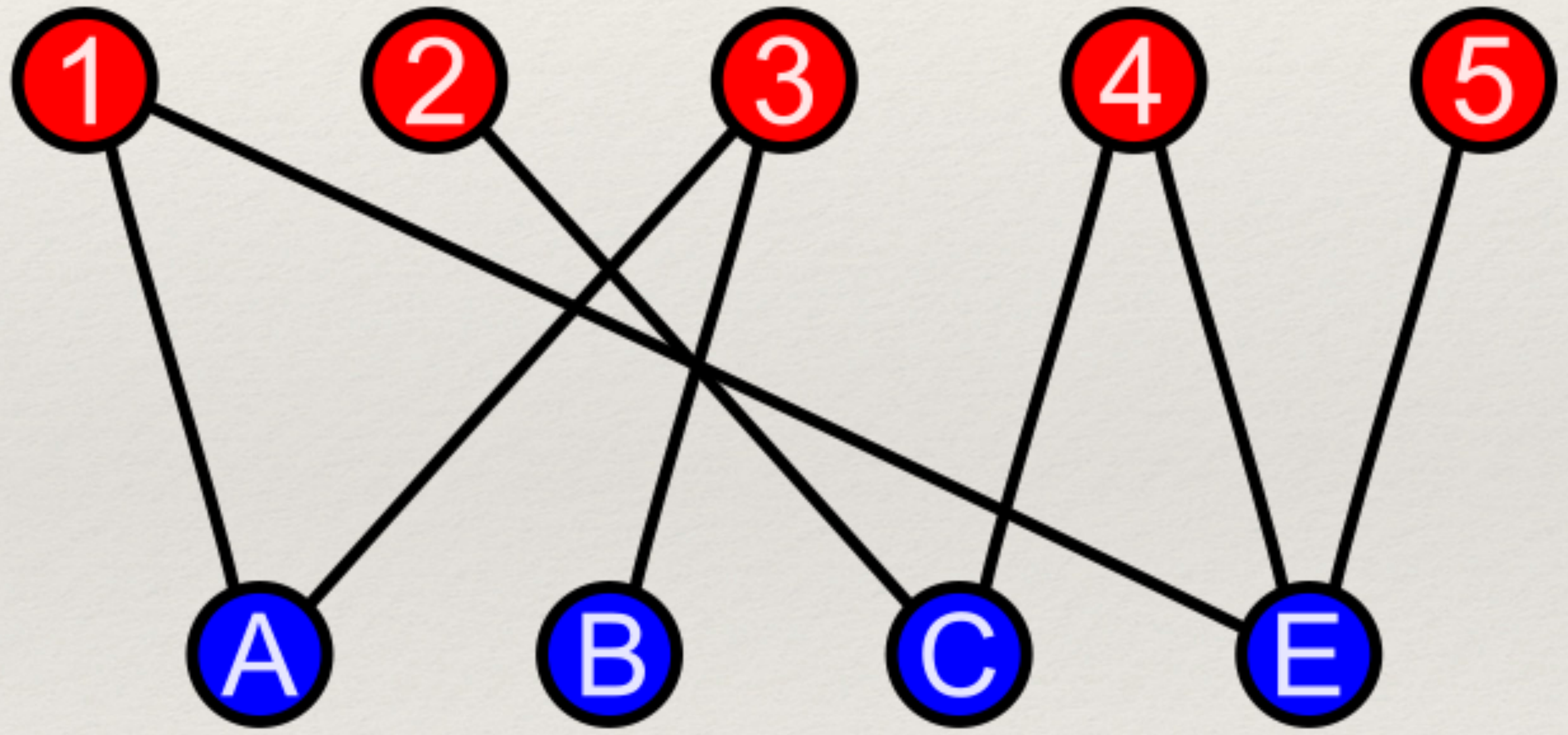
Subnetworks



- a **subnetwork** is a network obtained by selecting a subset of the nodes and all of the links among these nodes
- A **clique** is a complete subnetwork

Bipartite graph

- Two types of nodes
- Connections only happen between nodes of opposite type
- Example: actor, movie dataset



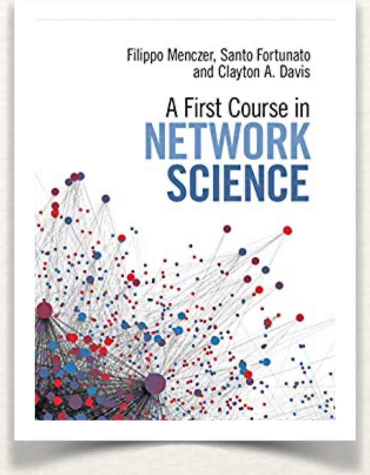
Multilayer networks

- A network can have multiple **layers**, each with its own nodes and edges
    - Example: air transportation networks of distinct airlines, with some but not complete overlap of airport nodes
- **Intralayer links** among nodes in the same layer, interlayer links across layers
- If the sets of nodes in the different layers are identical, we call the network a **multiplex**; interlayer links are **couplings** linking the same node across layers
    - Example: layers to represent different types of relationship in a social network, such as friendship, family ties, coworkers, etc.

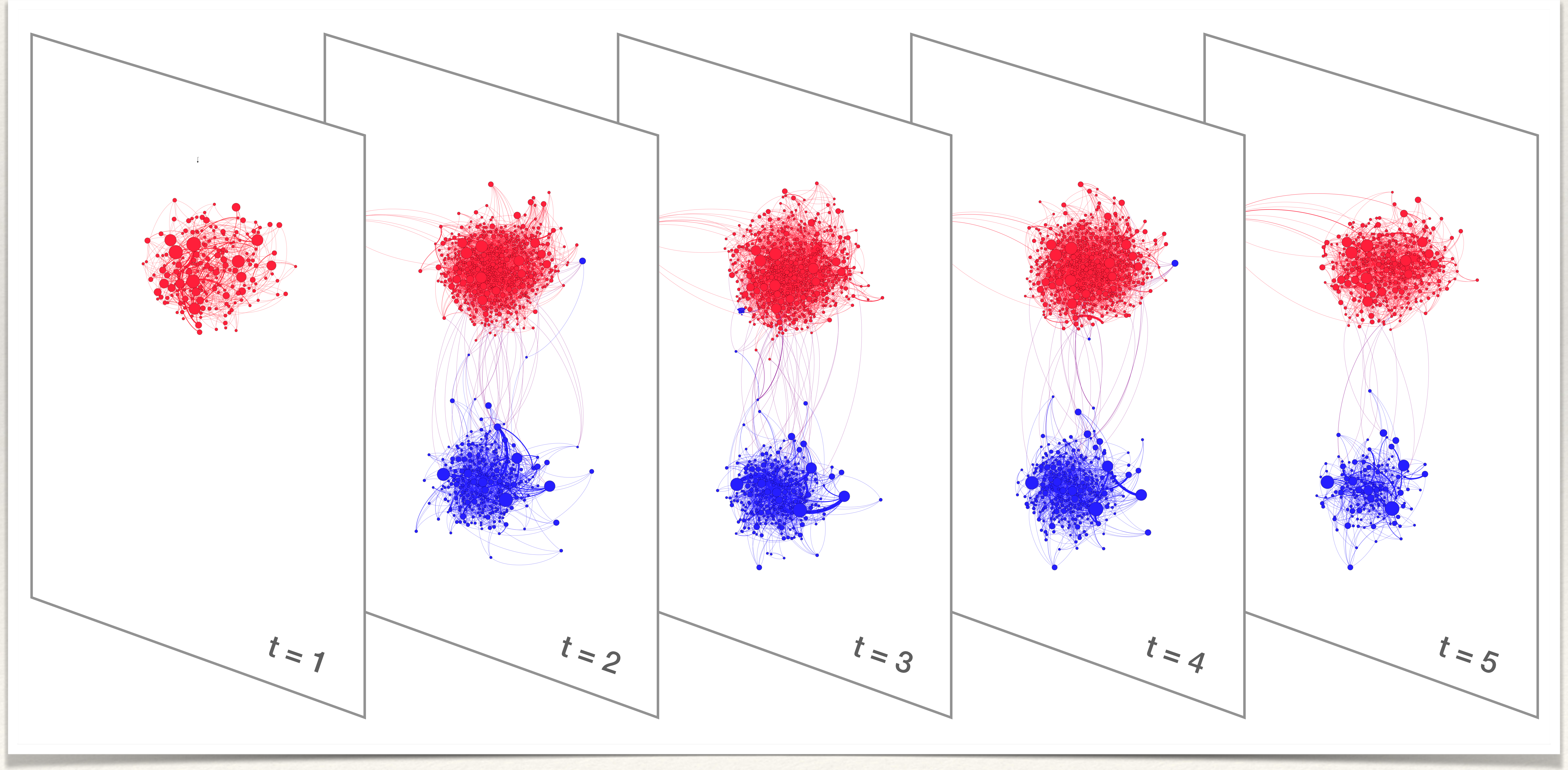
Networks of networks

- In general, each layer in a multilayer network can have its own nodes and edges. We call this a **network of networks**
  - Examples: the electrical power grid and the Internet; also the Internet itself!
- The interactions between networks in the different layers are captured by the interlayer links
  - Example: power stations communicate via the Internet, Internet routers are powered by the power grid
  - **Cascading failures** are one type of unpredictable vulnerabilities that arise in networks of networks

Temporal networks

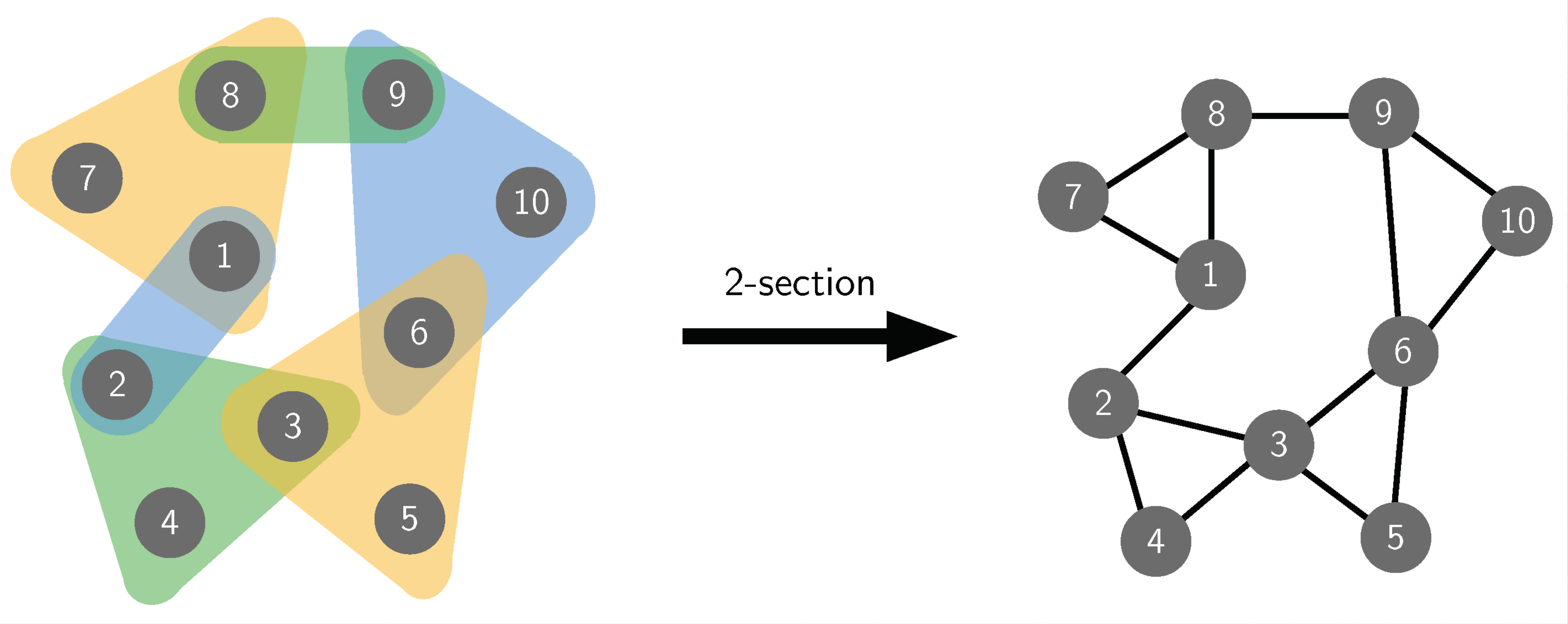
Figure from 

- A **temporal network** is a multiplex in which the layers represent links at different times (temporal snapshots)
  - Example: a Twitter retweet network



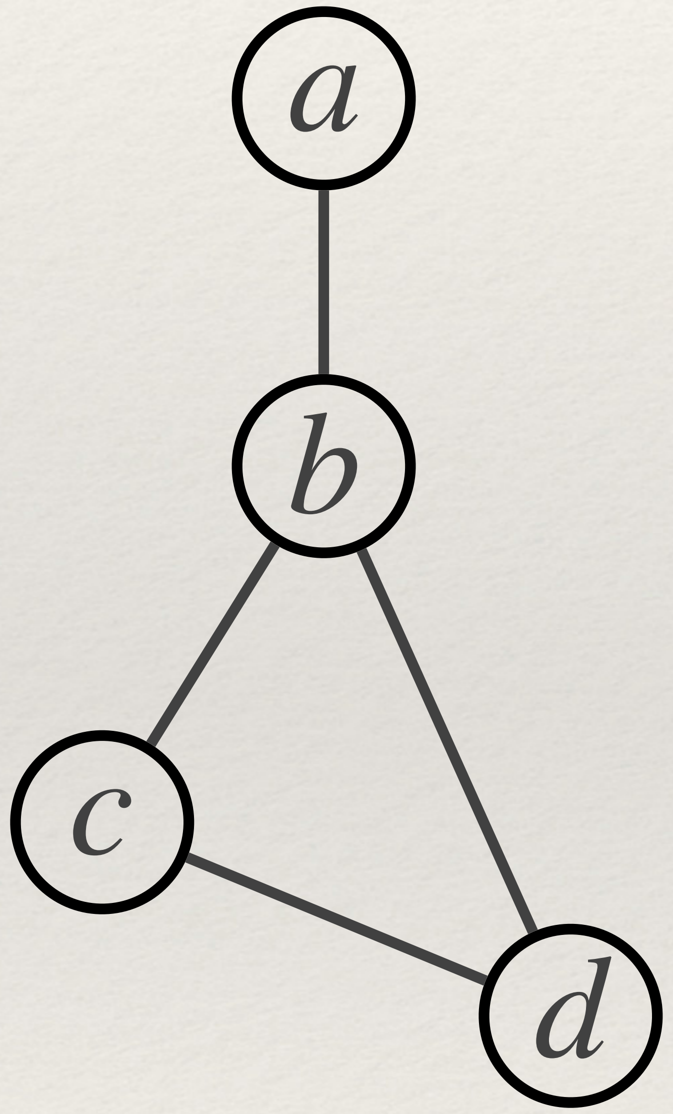
Hypergraphs

- Interaction may occur between multiple nodes simultaneously
- The hypergraph is represented by a sequence of hyperedges of size greater or equal to 2
- Ex: {(1,2), (1,7,8), (8,9), (9,10,6), (3,6,5), (2,3,4)}



Neighbors

Neighbors



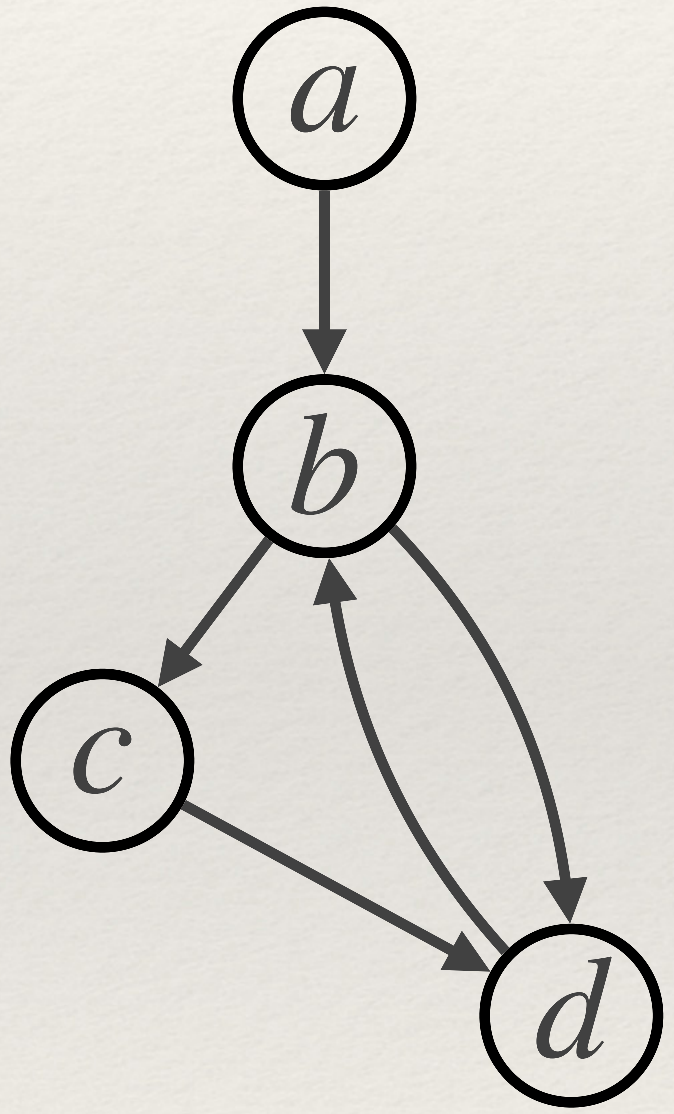
$$N_a = \{b\}$$

$$N_b = \{a, c, d\}$$

$$N_c = \{b, d\}$$

$$N_d = \{b, c\}$$

Neighbors, Successors, Predecessors



$$S_a = \{b\}, P_a = \{\}$$

$$S_b = \{c, d\}, P_b = \{a, d\}$$

$$S_c = \{d\}, P_c = \{b\}$$

$$S_d = \{b\}, P_d = \{b, c\}$$

Degree

Number of links (or neighbors)

$i \to N_i \qquad k_i = |N_i|$ degree

Singleton: a node whose degree is zero

$N_i = \{\}, k_i = 0$

In directed networks

$k_i^{in} = |P_i|$ in-degree

$k_i^{out} = |S_i|$ out-degree

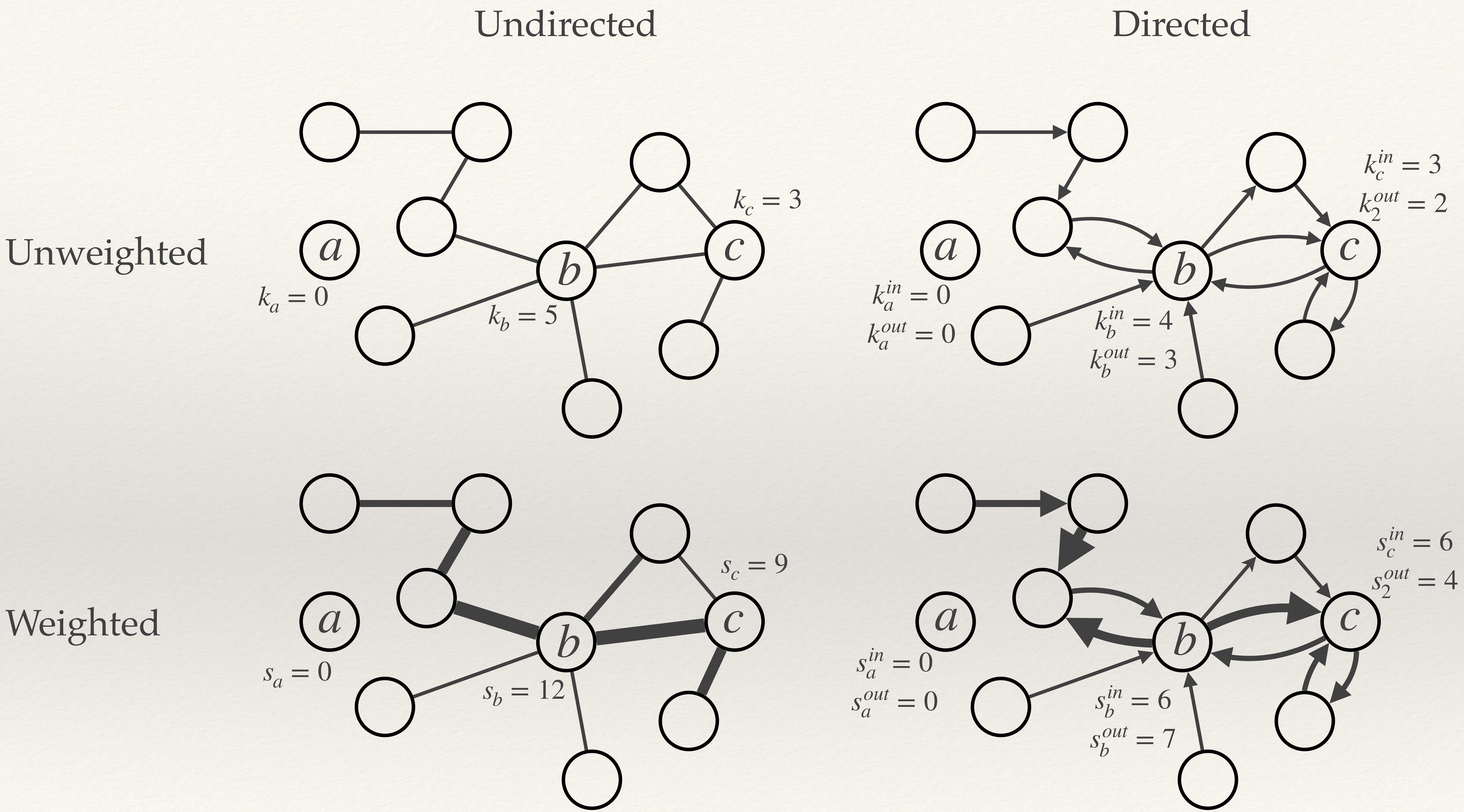
$$k_i = k_i^{in} + k_i^{out}$$

Strength

Strength: Weighted degree $s_i = \sum_{j \in N_i} w_{ij}$

in-strength $s_i^{in} = \sum_{j \in P_i} w_{ji}$

out-strength $s_i^{out} = \sum_{j \in S_i} w_{ij}$


Undirected
Directed
Unweighted
Weighted
a
b
c
$k_a = 0$
$k_b = 5$
$k_c = 3$
$k_a^{in} = 0$
$k_a^{out} = 0$
$k_b^{in} = 4$
$k_b^{out} = 3$
$k_c^{in} = 3$
$k_2^{out} = 2$
$s_a = 0$
$s_b = 12$
$s_c = 9$
$s_a^{in} = 0$
$s_a^{out} = 0$
$s_b^{in} = 6$
$s_b^{out} = 7$
$s_c^{in} = 6$
$s_2^{out} = 4$

Density and sparsity

- Network **size** = number of nodes — $|N|$ or, simply N
- Number of links — $|L|$ or, simply L
- Maximum possible number of links — $L_{\max} = \binom{N}{2} = \dfrac{N(N-1)}{2}$
- Density — $d = \dfrac{L}{L_{max}} = \dfrac{2L}{N(N-1)}$
- Sparsity — if $d \ll 1 \Rightarrow$ sparse

$$\begin{cases} L \text{ in the order of } N & \Rightarrow \text{sparse} \\ L \text{ in the order of } N^2 & \Rightarrow \text{dense} \end{cases}$$

Average degree

- average degree

$$< k > = \frac{\sum_{i \in N} k_i}{N}$$

(Undirected Net)

$$= \frac{2L}{N}$$

$$d = \frac{2L}{N(N-1)} \Rightarrow L = \frac{dN(N-1)}{2}$$

$$= \frac{dN(N-1)}{N} \qquad = d(N-1)$$

- Average degree is also connected to density

$$d = \frac{< k >}{N-1} = \frac{< k >}{k_{max}}$$

Example: facebook

- Rough orders-of-magnitude approximations:
- $N \approx 10^9$
- $L \approx 10^3 \times N$
- $d \approx \frac{L}{N^2} = \frac{10^3 N}{N^2} = \frac{10^3}{10^9} = 10^{-6} \ll 1$
- Most (but not all) real-world networks are similarly sparse because the number of links scales proportionally to N, whereas the maximum scales with N^2

Real networks

Table 1.1 Basic statistics of network examples. Network types can be (D)irected and/or (W)eighted. When there is no label the network is undirected and unweighted. For directed networks, we provide the average in-degree (which coincides with the average out-degree).

Network	Type	Nodes (N)	Links (L)	Density (d)	Average degree ($\langle k \rangle$)
Facebook Northwestern Univ.		10,567	488,337	0.009	92.4
IMDB movies and stars		563,443	921,160	0.000006	3.3
IMDB co-stars	W	252,999	1,015,187	0.00003	8.0
Twitter US politics	DW	18,470	48,365	0.0001	2.6
Enron Email	DW	87,273	321,918	0.00004	3.7
Wikipedia math	D	15,220	194,103	0.0008	12.8
Internet routers		190,914	607,610	0.00003	6.4
US air transportation		546	2,781	0.02	10.2
World air transportation		3,179	18,617	0.004	11.7
Yeast protein interactions		1,870	2,277	0.001	2.4
C. elegans brain	DW	297	2,345	0.03	7.9
Everglades ecological food web	DW	69	916	0.2	13.3

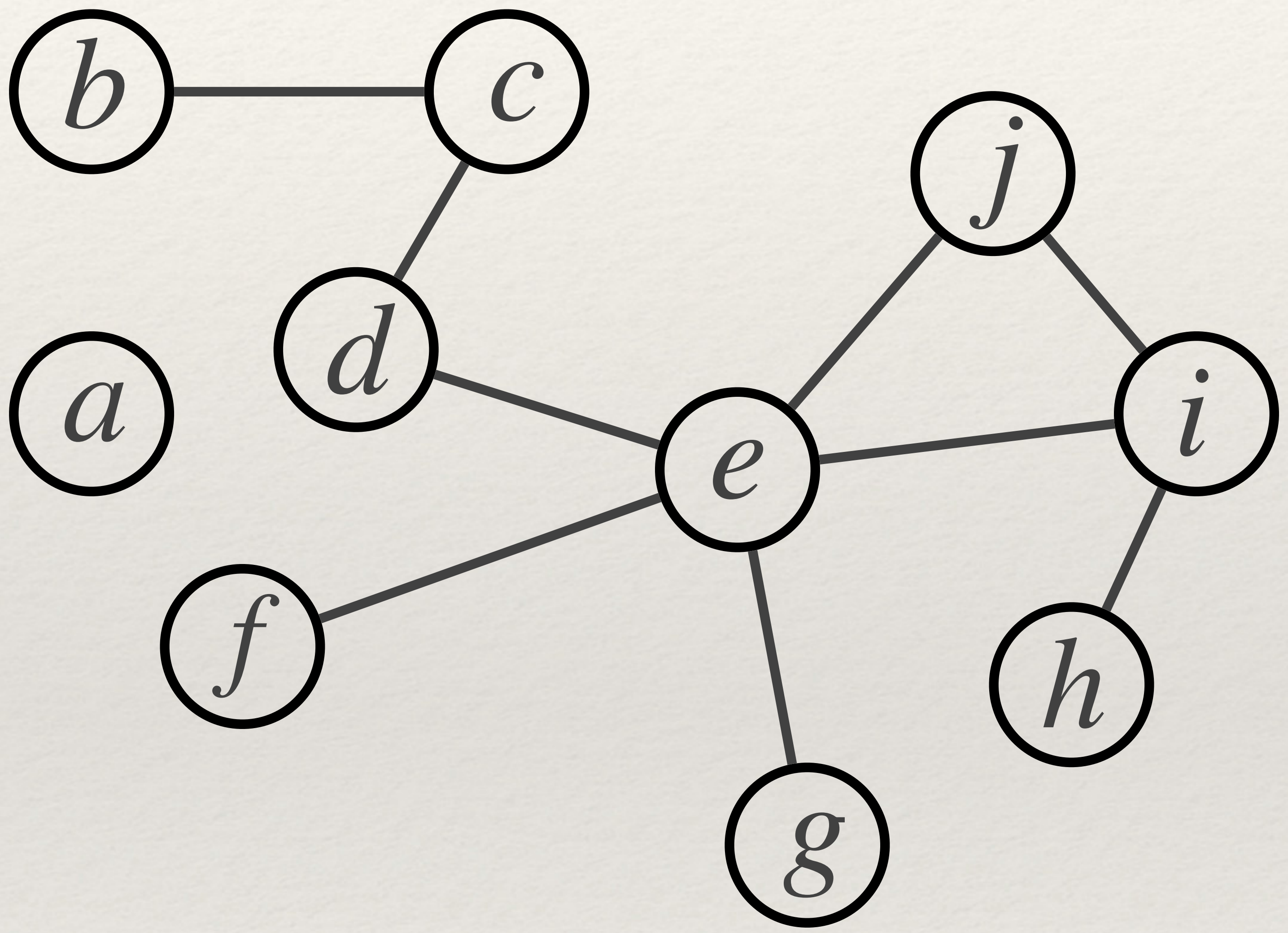
Network representations

Adjacency Matrix

$N x N$ matrix

$$a_{ij} = \begin{cases} 0 & \text{no edge} \\ 1 & (i,j) \in L \end{cases}$$

Undirected network: $a_{ij} = a_{ji}$



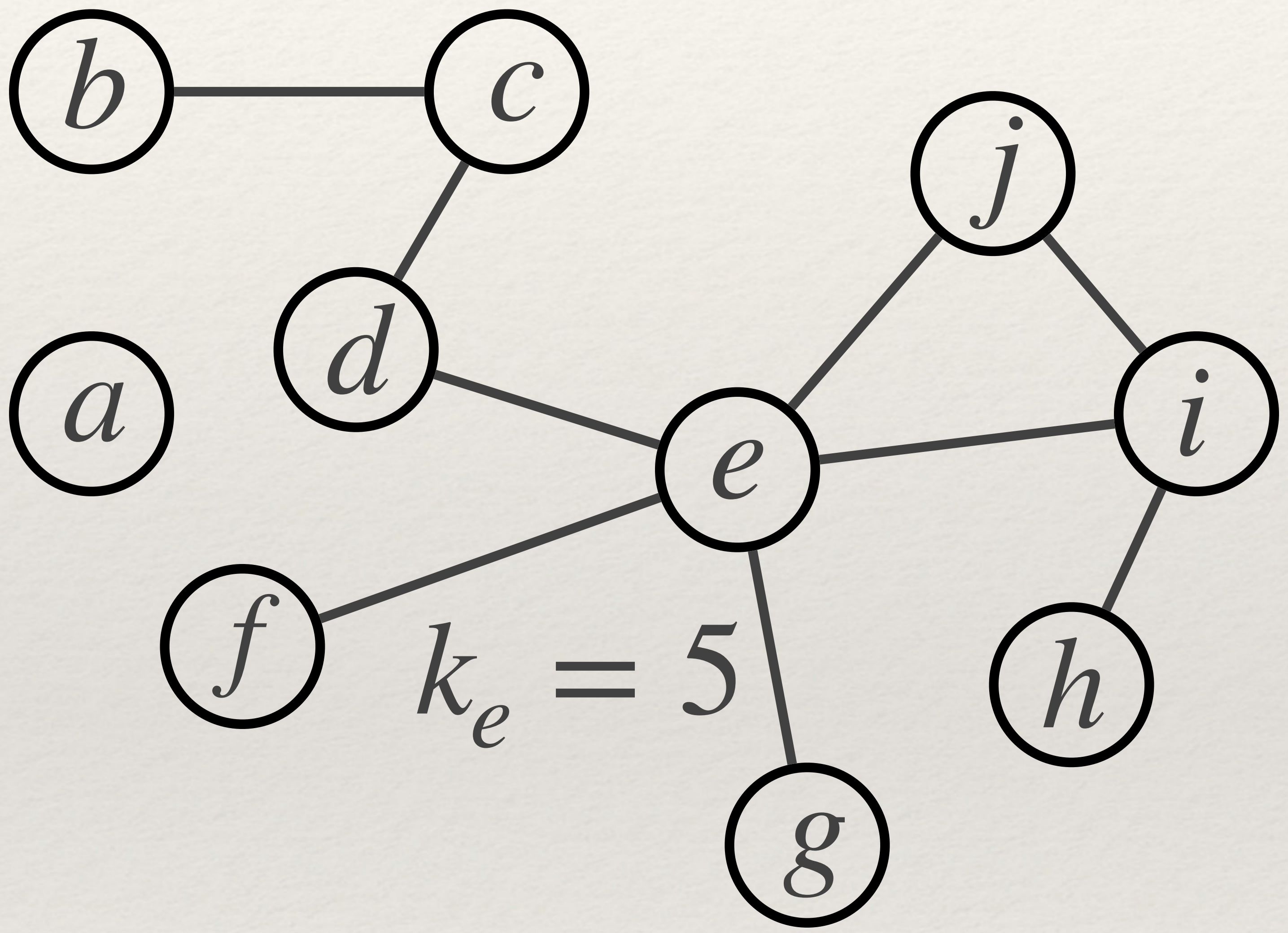
$$\begin{array}{c c} & \begin{array}{cccccccccc} \mathtt{a} & \mathtt{b} & \mathtt{c} & \mathtt{d} & \mathtt{e} & \mathtt{f} & \mathtt{g} & \mathtt{h} & \mathtt{i} & \mathtt{j} \end{array} \\ \begin{array}{c} \mathtt{a} \\ \mathtt{b} \\ \mathtt{c} \\ \mathtt{d} \\ \mathtt{e} \\ \mathtt{f} \\ \mathtt{g} \\ \mathtt{h} \\ \mathtt{i} \\ \mathtt{j} \end{array} & \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Network representations

Adjacency Matrix

degree

$$k_i = \sum_j a_{ij} = \sum_j a_{ji}$$



$k_e = 5$

$$\begin{array}{c c}
 & \begin{array}{cccccccccc} \mathtt{a} & \mathtt{b} & \mathtt{c} & \mathtt{d} & \mathtt{e} & \mathtt{f} & \mathtt{g} & \mathtt{h} & \mathtt{i} & \mathtt{j} \end{array} \\
\begin{array}{c} \mathtt{a} \\ \mathtt{b} \\ \mathtt{c} \\ \mathtt{d} \\ \mathtt{e} \\ \mathtt{f} \\ \mathtt{g} \\ \mathtt{h} \\ \mathtt{i} \\ \mathtt{j} \end{array} &
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}
\end{array}$$

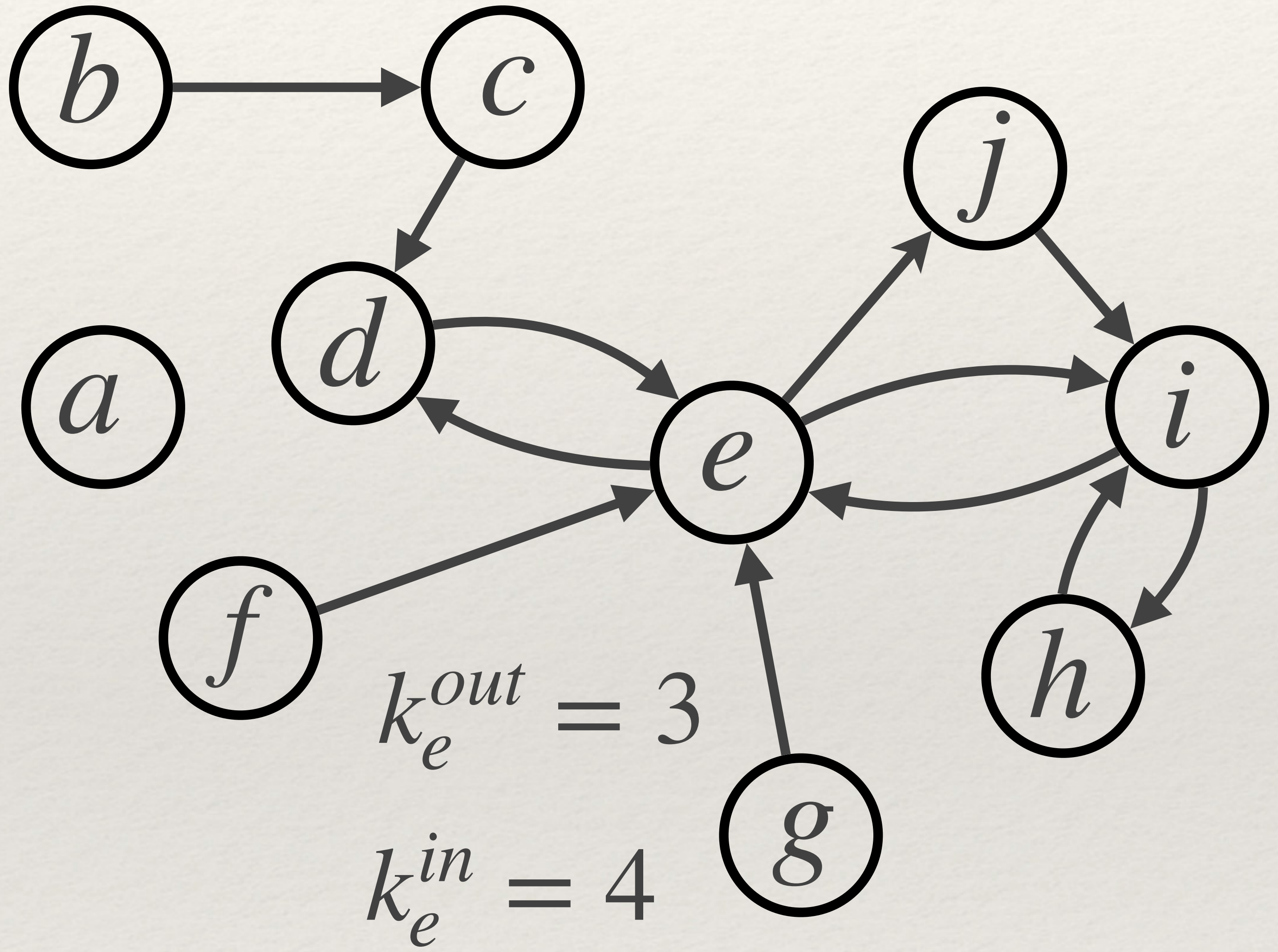
Network representations

Adjacency Matrix
directed

Matrix is not symmetric

$$k_i^{out} = \sum_j a_{ij}$$

$$k_i^{in} = \sum_j a_{ji}$$



$$\begin{array}{c} \\ \mathtt{a} \\ \mathtt{b} \\ \mathtt{c} \\ \mathtt{d} \\ \mathtt{e} \\ \mathtt{f} \\ \mathtt{g} \\ \mathtt{h} \\ \mathtt{i} \\ \mathtt{j} \end{array}
\begin{array}{c} \begin{array}{cccccccccc} \mathtt{a} & \mathtt{b} & \mathtt{c} & \mathtt{d} & \mathtt{e} & \mathtt{f} & \mathtt{g} & \mathtt{h} & \mathtt{i} & \mathtt{j} \end{array} \\
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix} \end{array}$$

Network representations

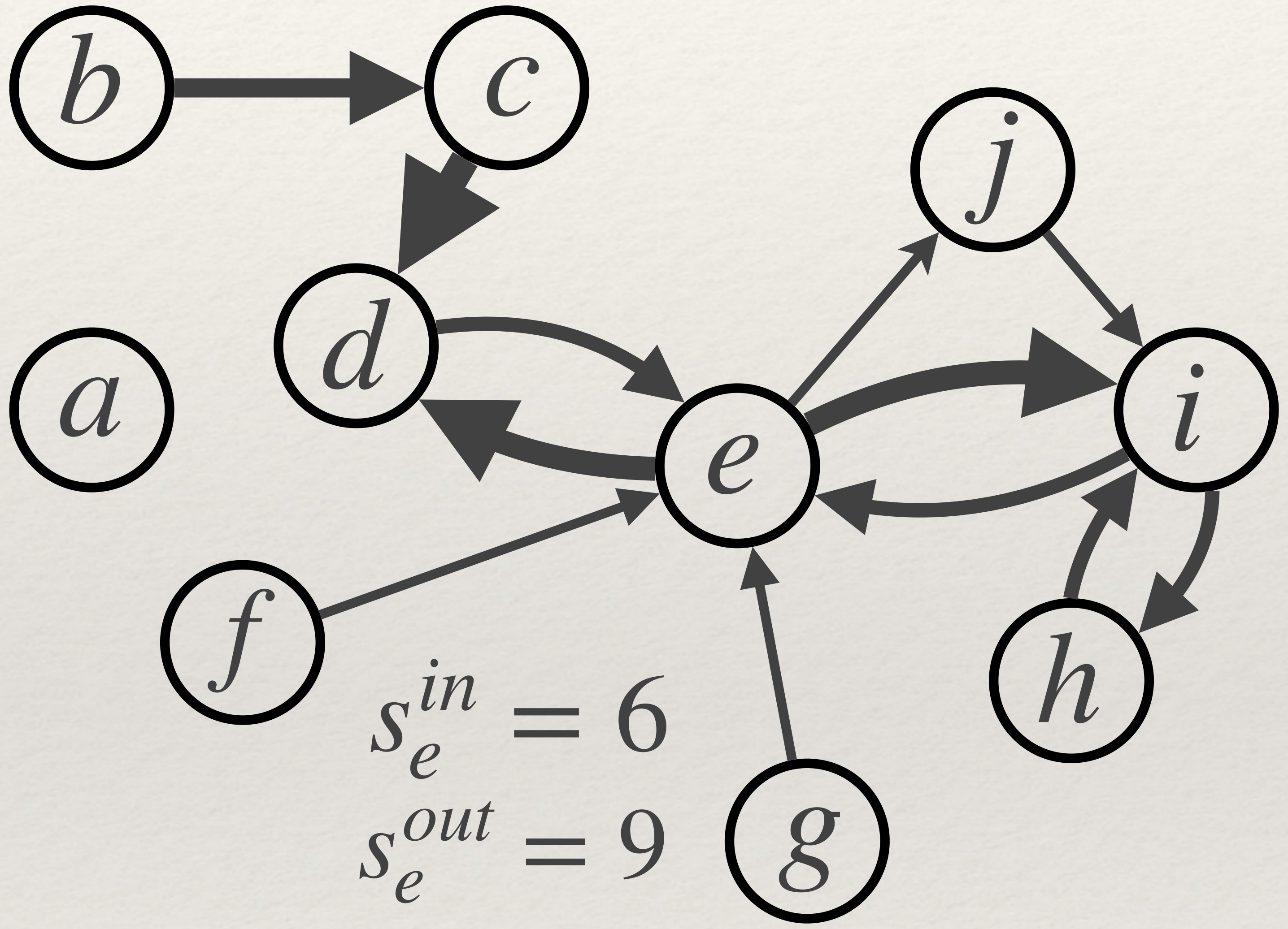
Adjacency Matrix

weighted

$$w_{ij}$$

$$s_i^{out} = \sum_j w_{ij}$$

$$s_i^{in} = \sum_j w_{ji}$$

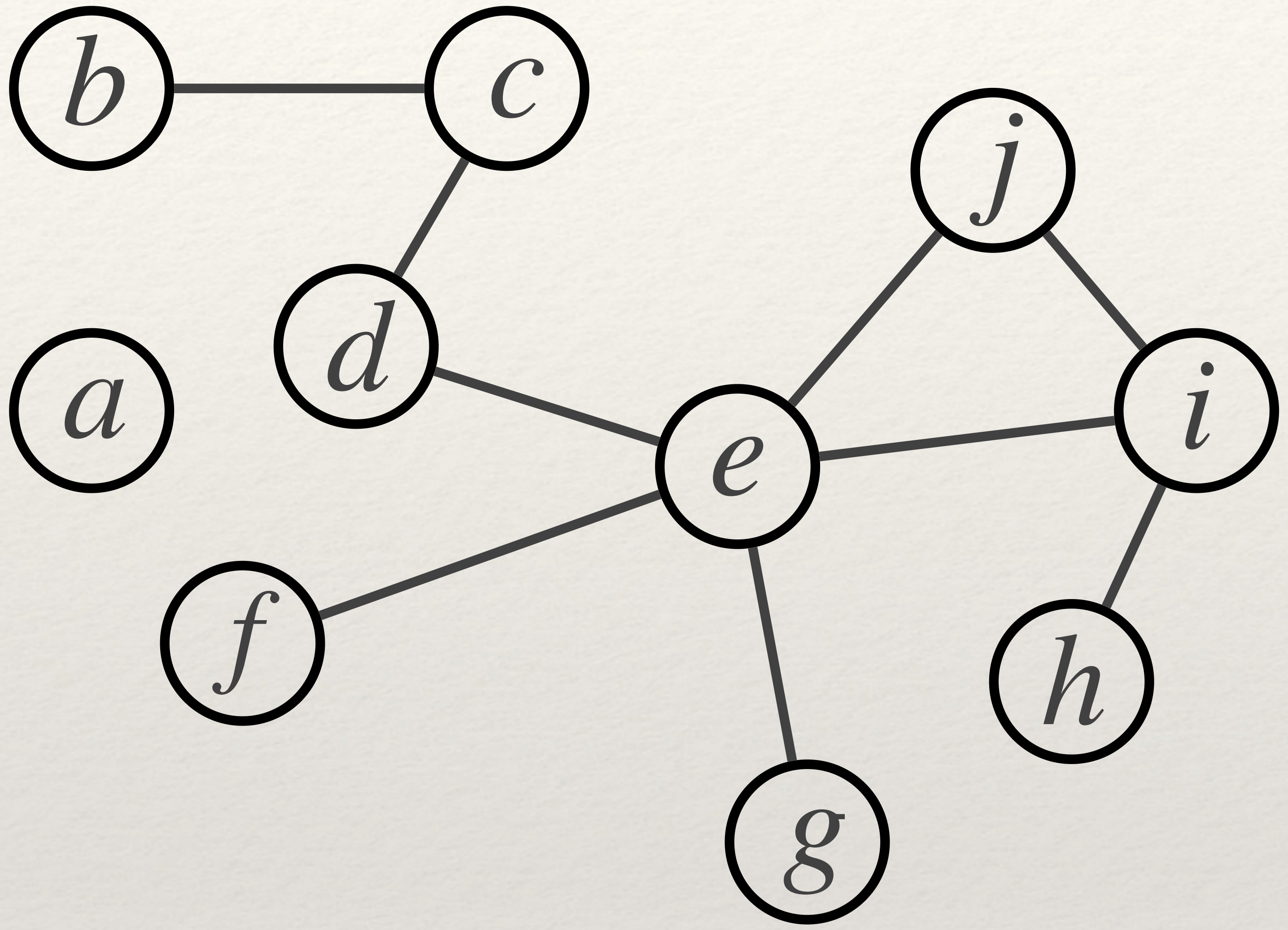


$$
\begin{array}{c}
\\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \\ j
\end{array}
\begin{array}{c}
\begin{array}{cccccccccc} a & b & c & d & e & f & g & h & i & j \end{array} \\
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 4 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0
\end{pmatrix}
\end{array}
$$

Sparse network representations

- The memory/disk storage needed by an adjacency matrix is proportional to N^2
- In sparse networks (most real-world networks), this is terribly inefficient: most of the space is wasted storing zeros (non-links); for very large networks, adjacency matrices are unfeasible
- It is much more efficient, often necessary, to store only the actual links, and assume that if a link is not listed it means it is not present
- There are two commonly used sparse networks representations:
    - **Adjacency list**
    - **Edge list**

Adjacency list

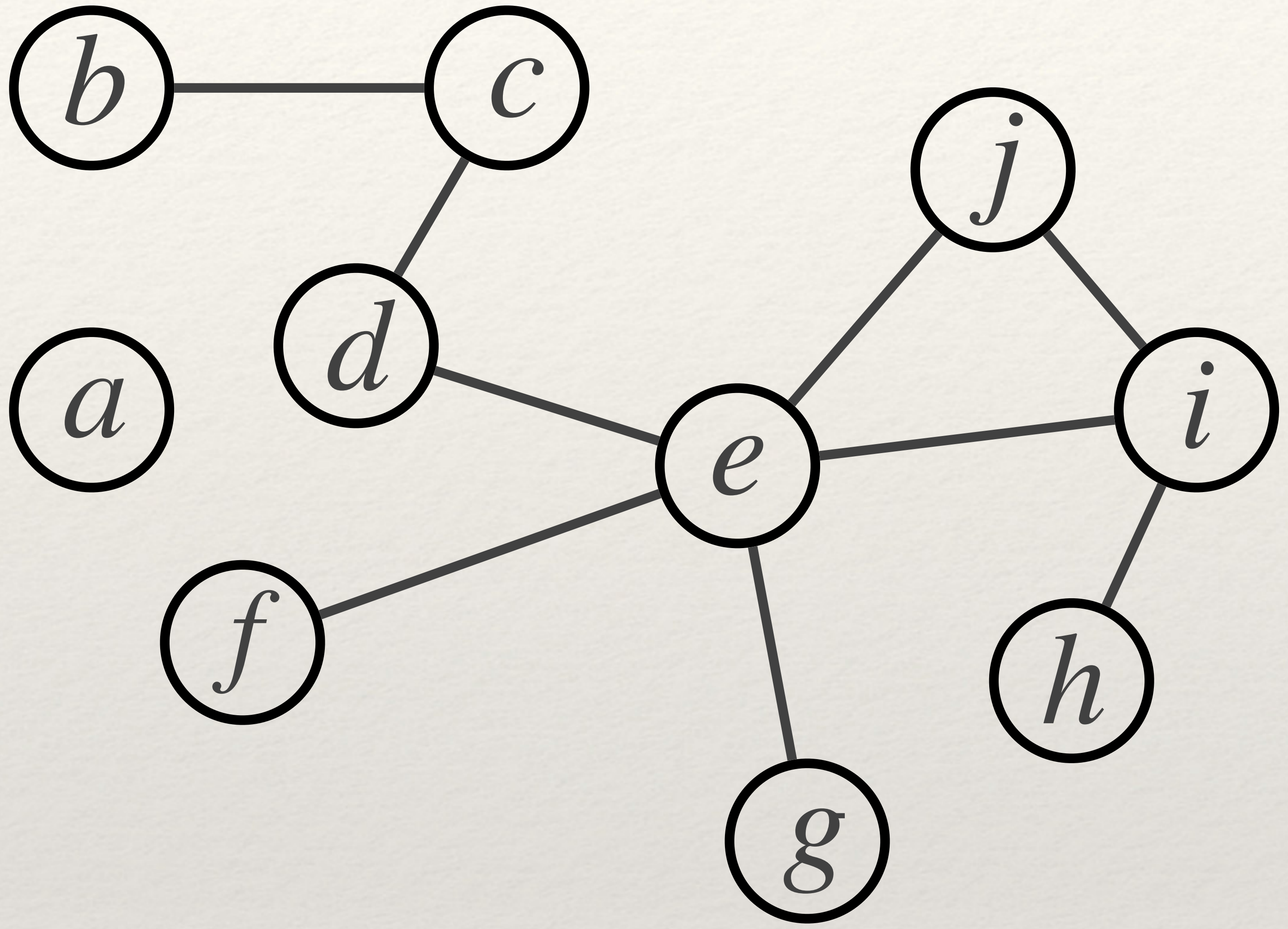



```
a
b    c
c    b d
d    c e
e    d f i j g
f    e
g    e
h    i
i    h j
j    e i
```

Undirected network: list each link twice

Directed network: list only existing links

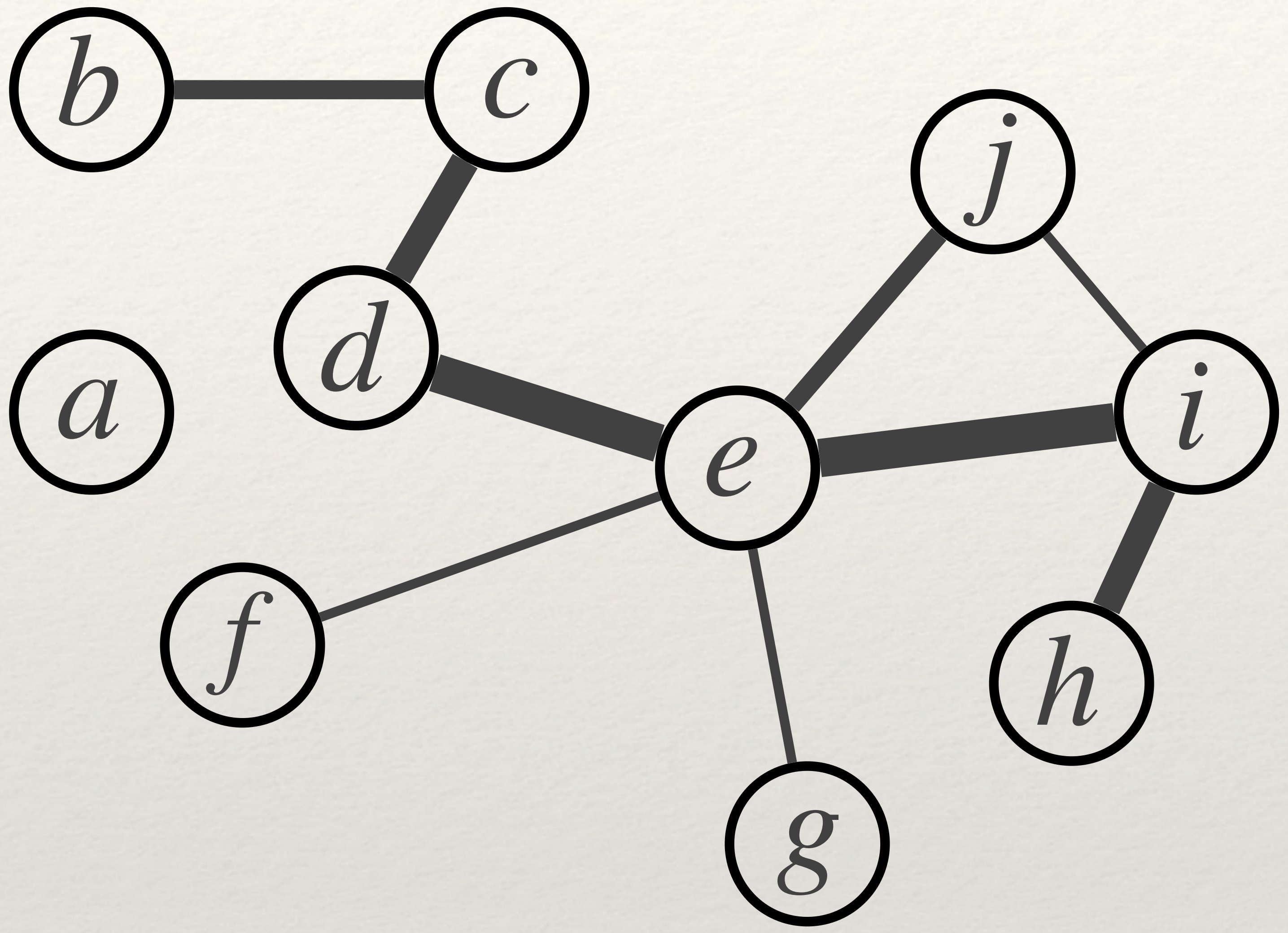
Edge list



b	c
c	d
d	e
e	f
e	g
e	i
e	j
h	i
i	j

L

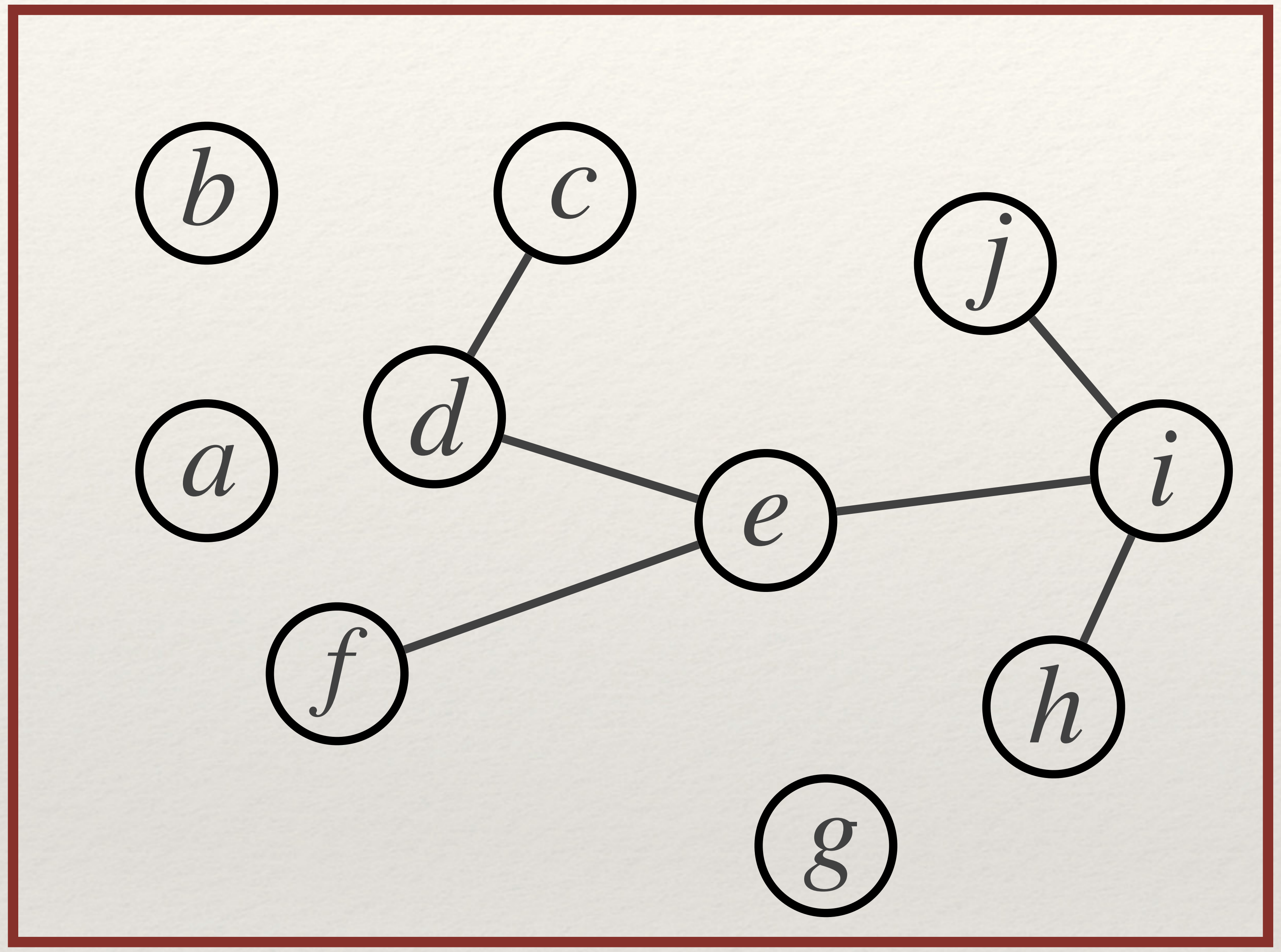
Edge (weighted) list

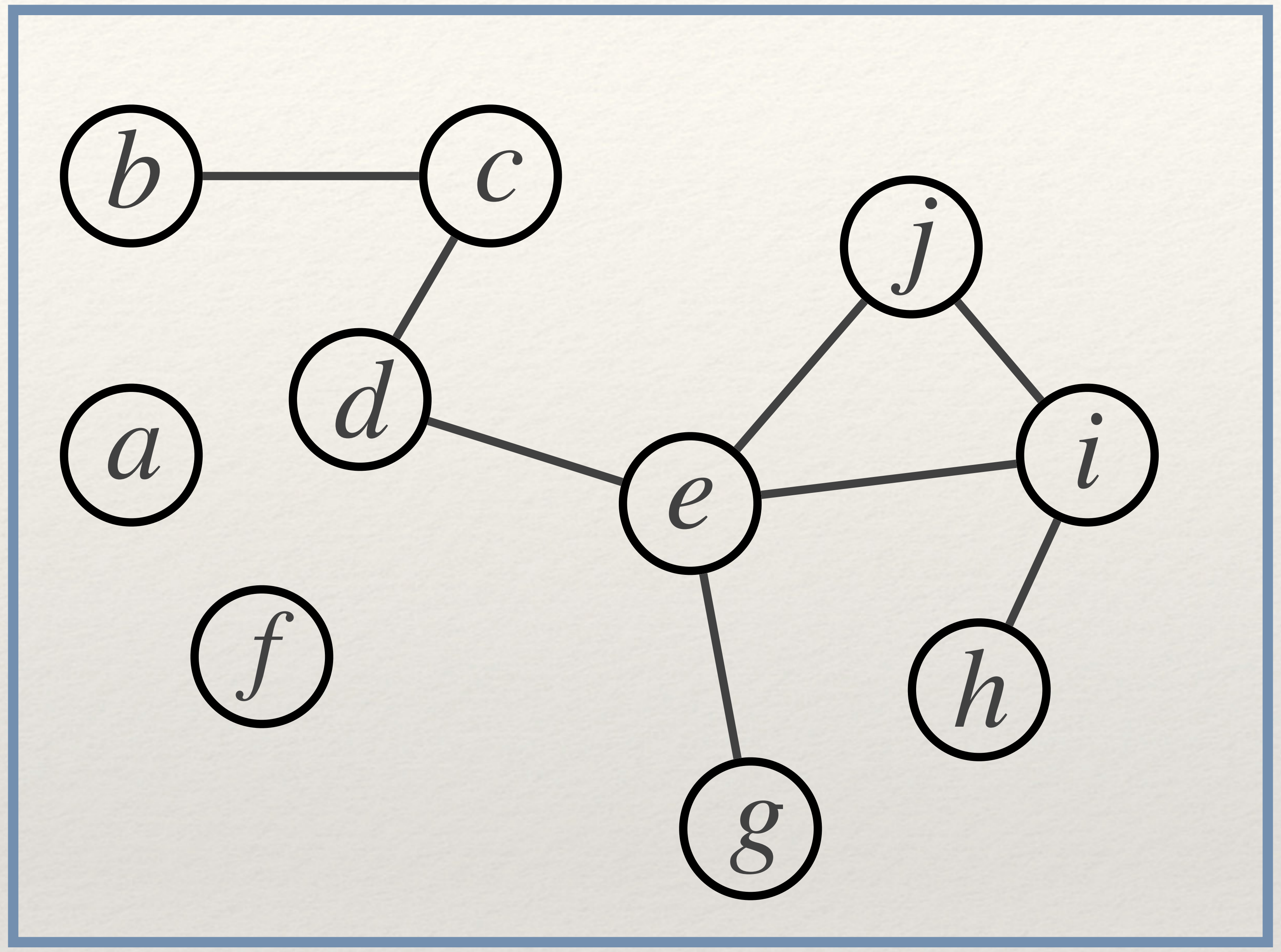


b	c	2
c	d	3
d	e	4
e	f	4
e	g	1
e	i	1
e	j	2
h	i	3
i	j	1

L

Temporal graphs





i	j	t	T
c	d	1	2
d	e	1	2
e	f	1	1
e	i	1	2
i	j	1	2
i	h	1	2
b	c	2	1
e	g	2	1
e	j	2	1

Building a graph in Python

NetworkX (https://networkx.org/)

```
import networkx as nx

# undirected unweighted
G = nx.Graph()
G.add_nodes_from([1, 2, 3])
G.add_edge(1, 2)
```

Building a graph in Python

NetworkX (https://networkx.org/)

```
import networkx as nx

# undirected unweighted
G = nx.Graph()
G.edges_from(edge_list)
```

Building a graph in Python

NetworkX (https://networkx.org/)

```
import networkx as nx

# directed graph
G = nx.DiGraph()

# weighted graph
G = nx.Graph()
G.add_weighted_edges_from([(1, 2, 0.125), (1, 3, 0.75), (2, 4, 1.2), (3, 4, 0.375)])
```

Building a graph in Python

GraphTool (https://graph-tool.skewed.de/static/doc/index.html)

```
from graph_tool.all import *

g = Graph(directed = False)

# add vertices
v1 = g.add_vertex()
v2 = g.add_vertex()

# add the edge
e = g.add_edge(v1, v2)
```

The adjacency matrix

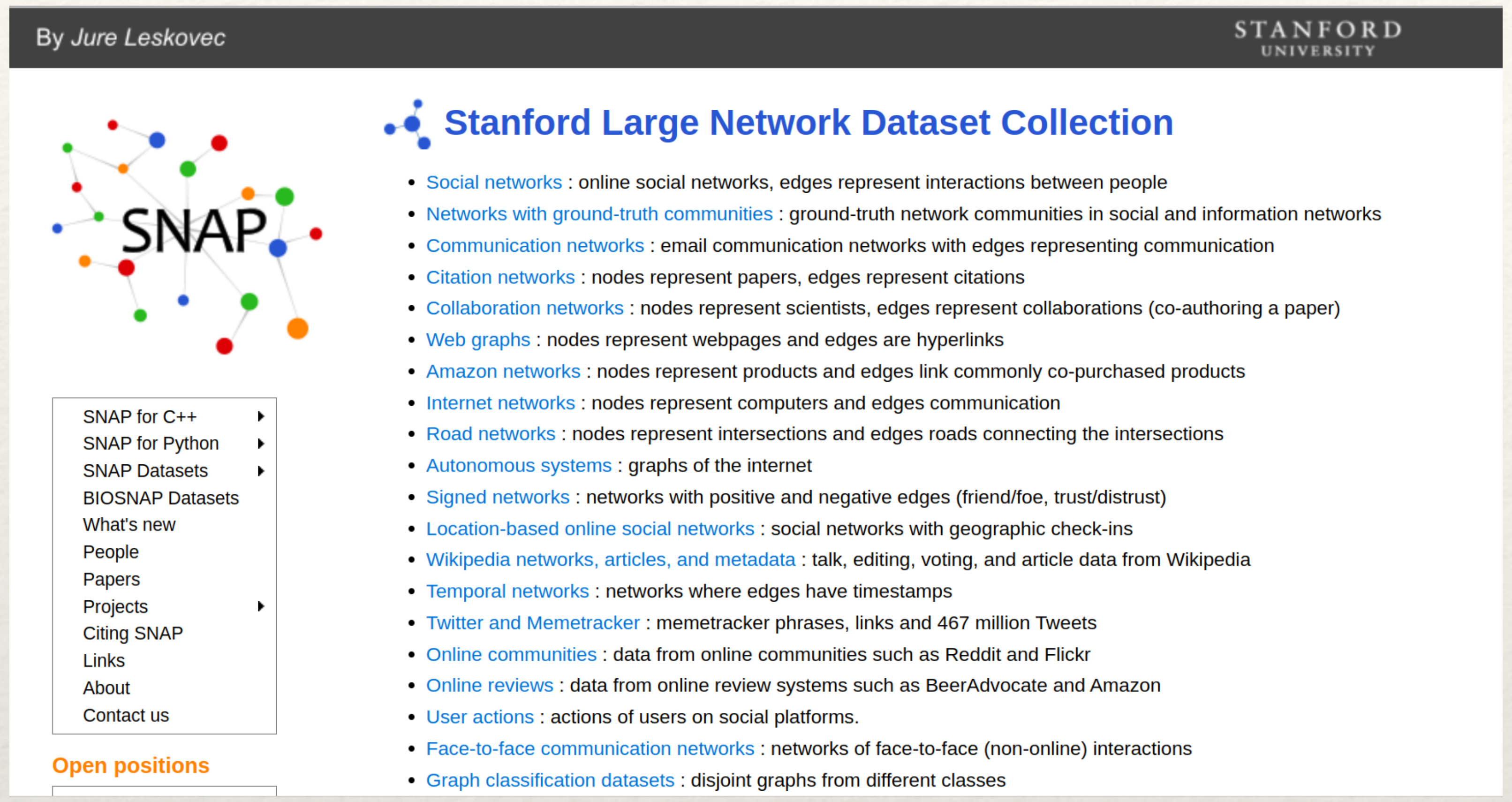
```python
from scipy.sparse import csr_matrix

# using scipy
A = csr_matrix((weights, (idx1, idx2)), shape = (n,n))

# using nx
A = nx.adjacency_matrix(G)

# using gt
A = graph_tool.spectral.adjacency(g)
```

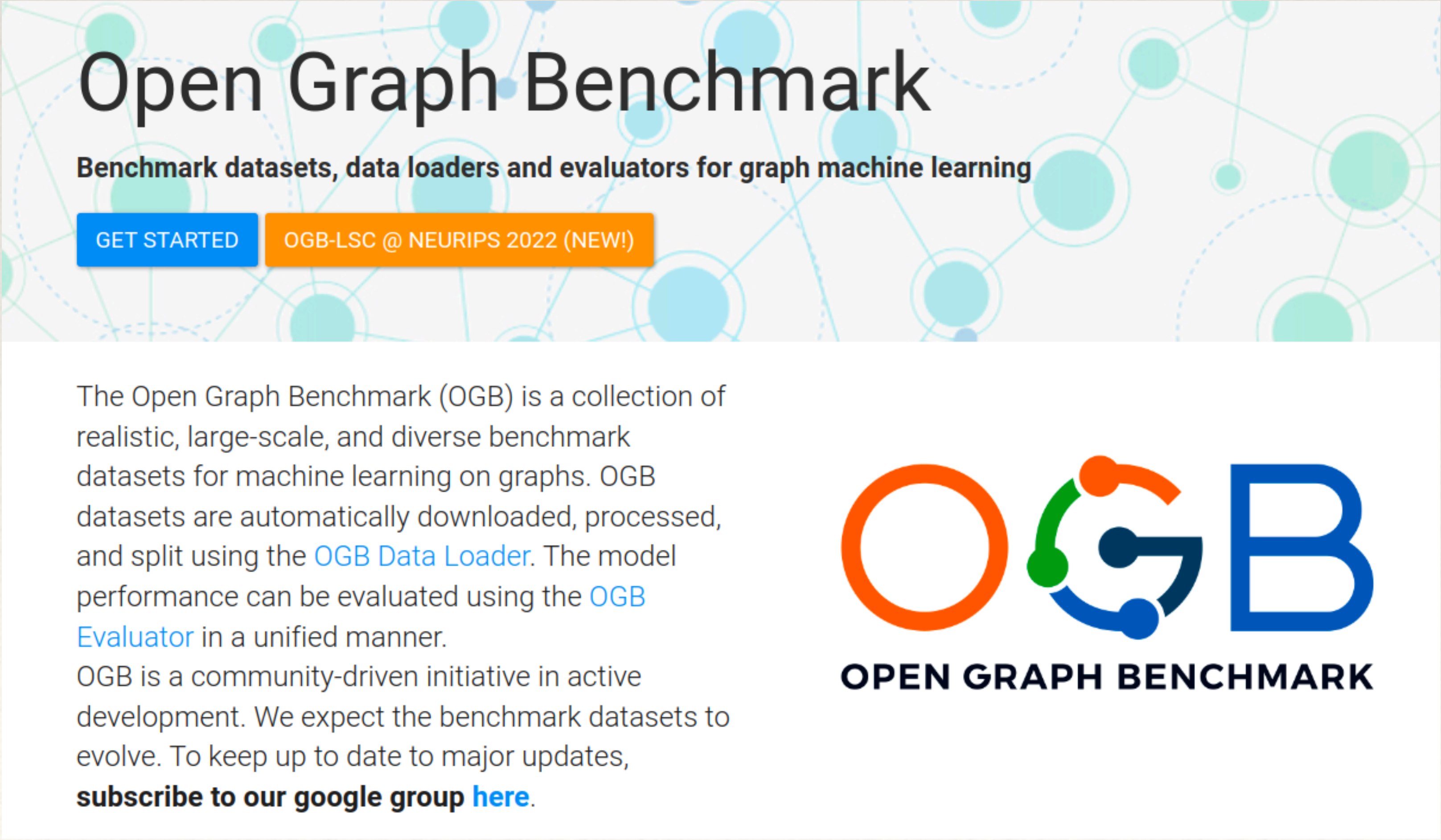
Some useful data sources



http://snap.stanford.edu/data/index.html

Netzschleuder

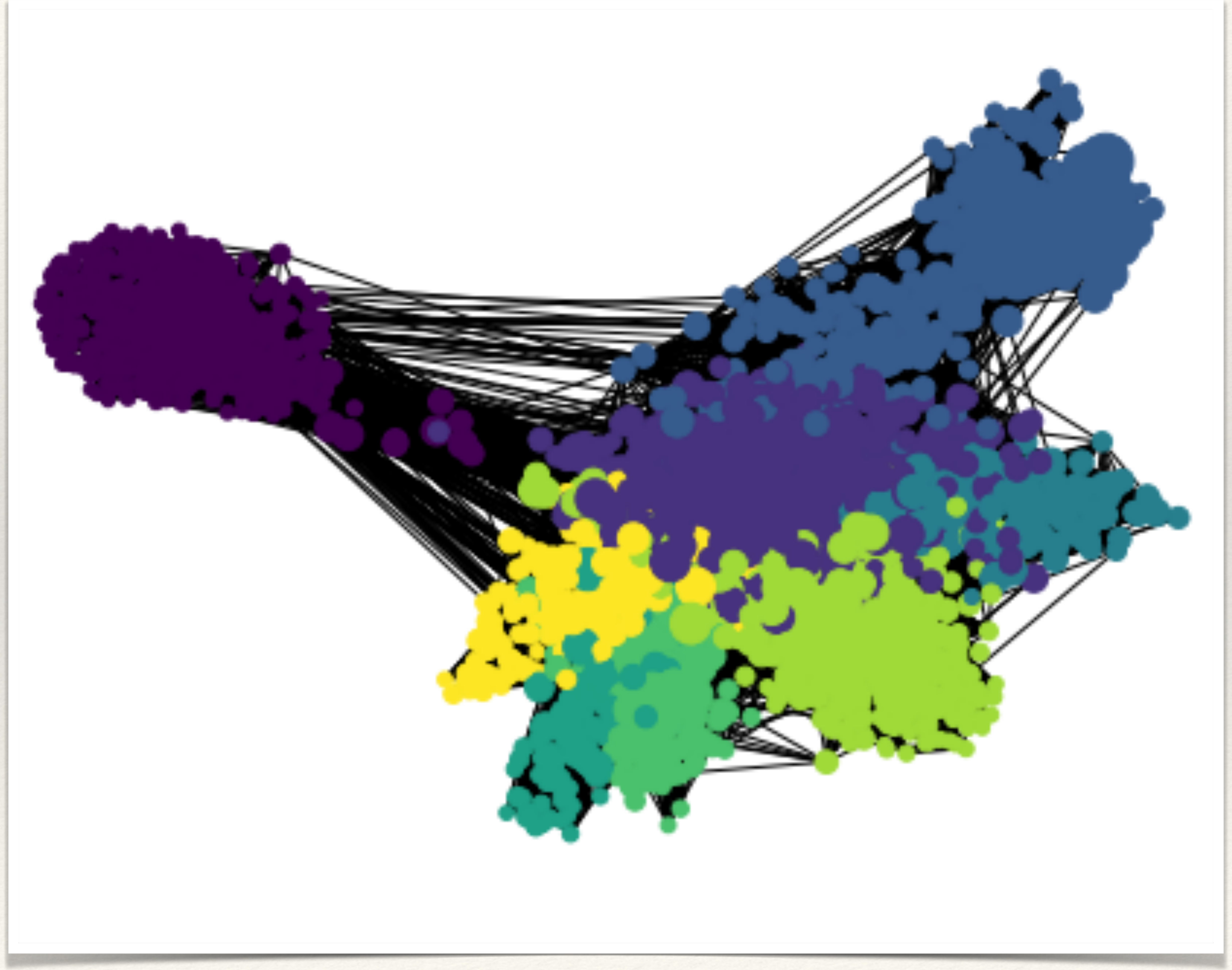
https://networks.skewed.de/



https://ogb.stanford.edu/

Drawing networks

- A **network layout algorithm** places nodes on a plane to visualize the structure of the network
- There are many layout algorithms; the most commonly used are **force-directed layout** (a.k.a. **spring layout**) algorithms:
  - Connected nodes are placed near each other
  - Links have similar length
  - Link crossings are minimized
- This is done by simulating a physical systems where adjacent nodes are connected by springs and otherwise repel each other
- The community structure of the network can be revealed this way if the network is not too dense or too large



Paths and connectivity

Paths and cycles

- N: nodes - or vertices, actors, sites, …
- L: links - or edges, arcs, social ties, …

path: $\{n_1, n_2, \ldots, n_l\}$ s.t. $\forall i : (n_i, n_{i+1}) \in L$

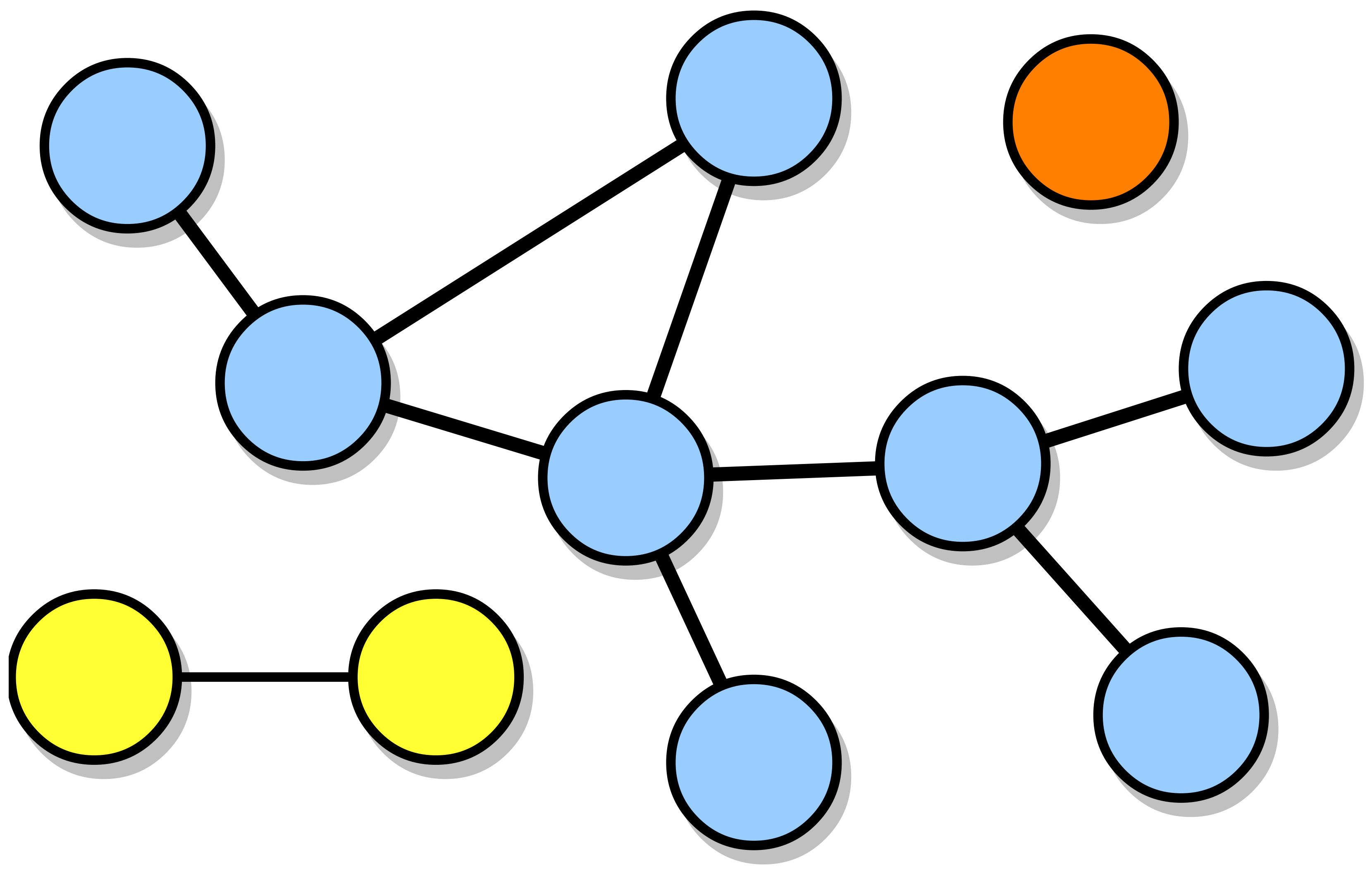
length: l

if we have repeating nodes $\Rightarrow$ **cycles**

no repeating nodes $\Rightarrow$ **simple path**

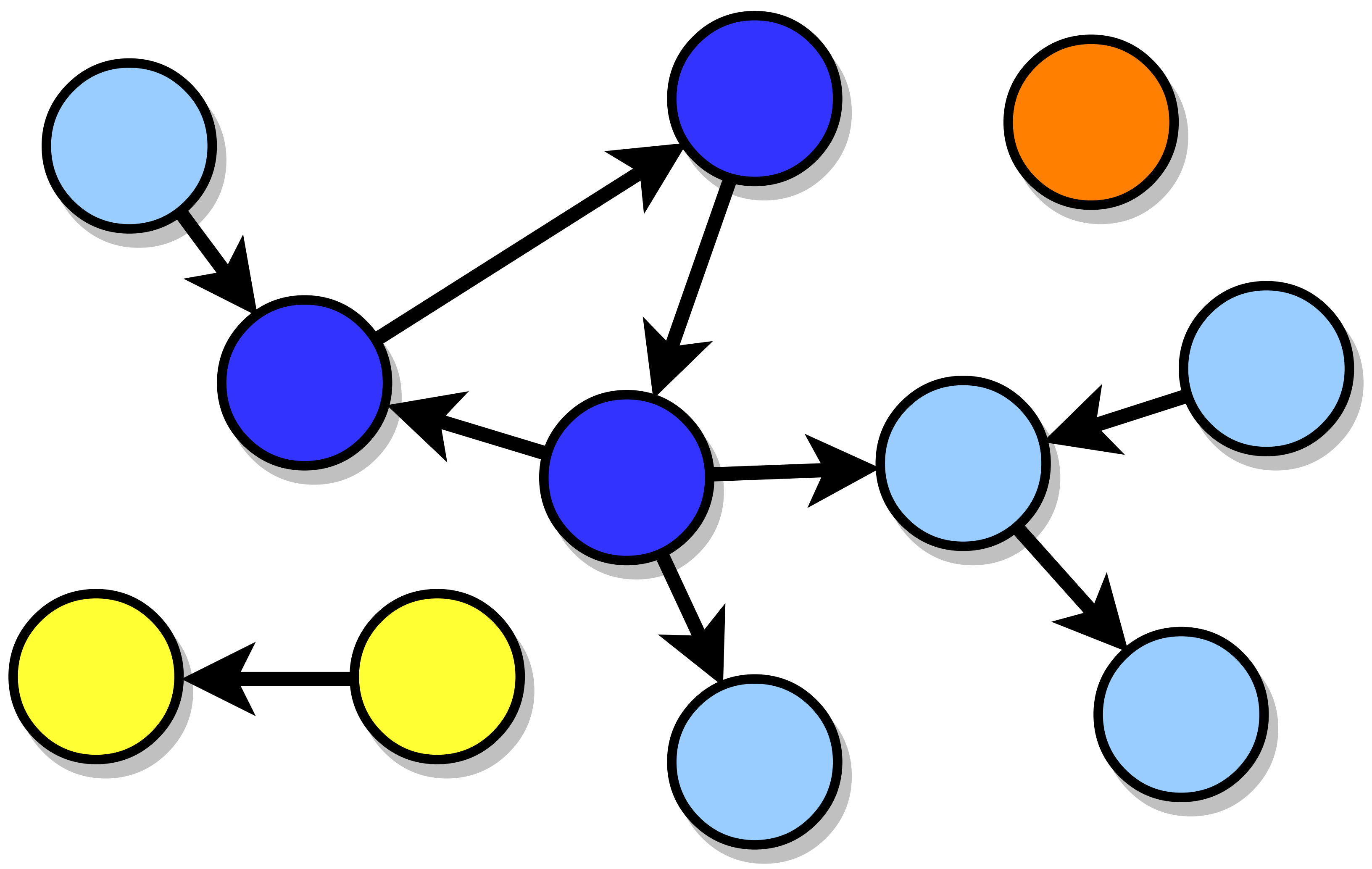
Connectedness and components

- A network is **connected** if there is a path between any two nodes
- If a network is not connected, it is **disconnected** and has multiple connected components
- A **connected component** is a connected subnetwork
  - The largest one is called **giant component**; it often includes a substantial portion of the network
  - A **singleton** is the smallest-possible connected component

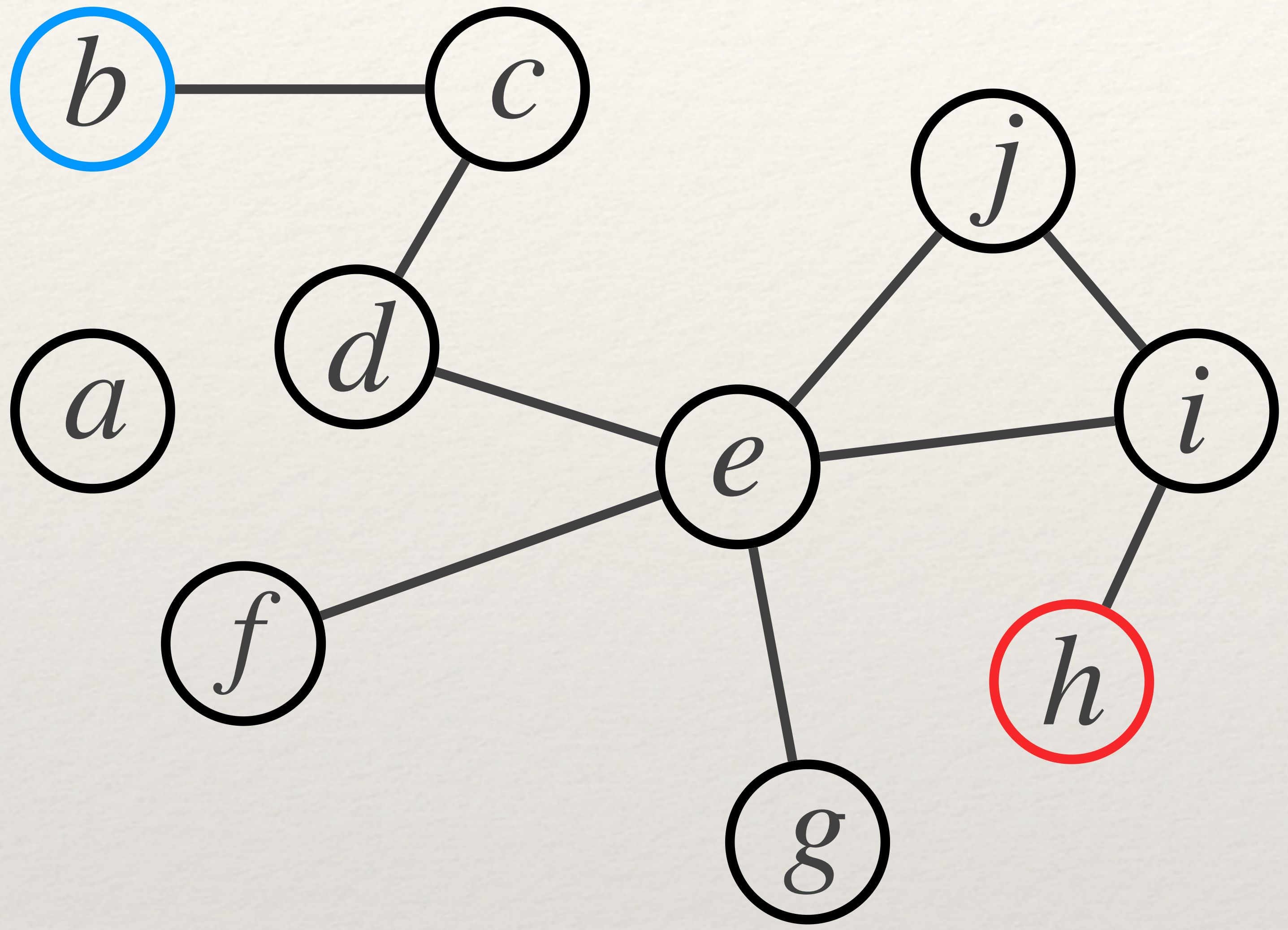


Connectedness and components

- A directed network can be **strongly connected** or **weakly connected** if there is a path between any two nodes, respecting or disregarding the link directions, respectively
- Similarly for **strongly connected** or **weakly connected** components
- The **in-component** of a strongly connected component S is the set of nodes from which one can reach S, but that cannot be reached from S
- The **out-component** of a strongly connected component S is the set of nodes that can be reached from S, but from which one cannot reach S



Shortest paths



{b, c, d, e, i, h}: path from source b to target h

{b, c, d, e, j, i, h}: another path from source b to target h

shortest path: minimal path between two nodes

in **weighted nets**: weights can represent distances from two adjacent nodes

distance between two nodes: length of the shortest path

a (singleton) is considered at distance ∞ from any other nodes

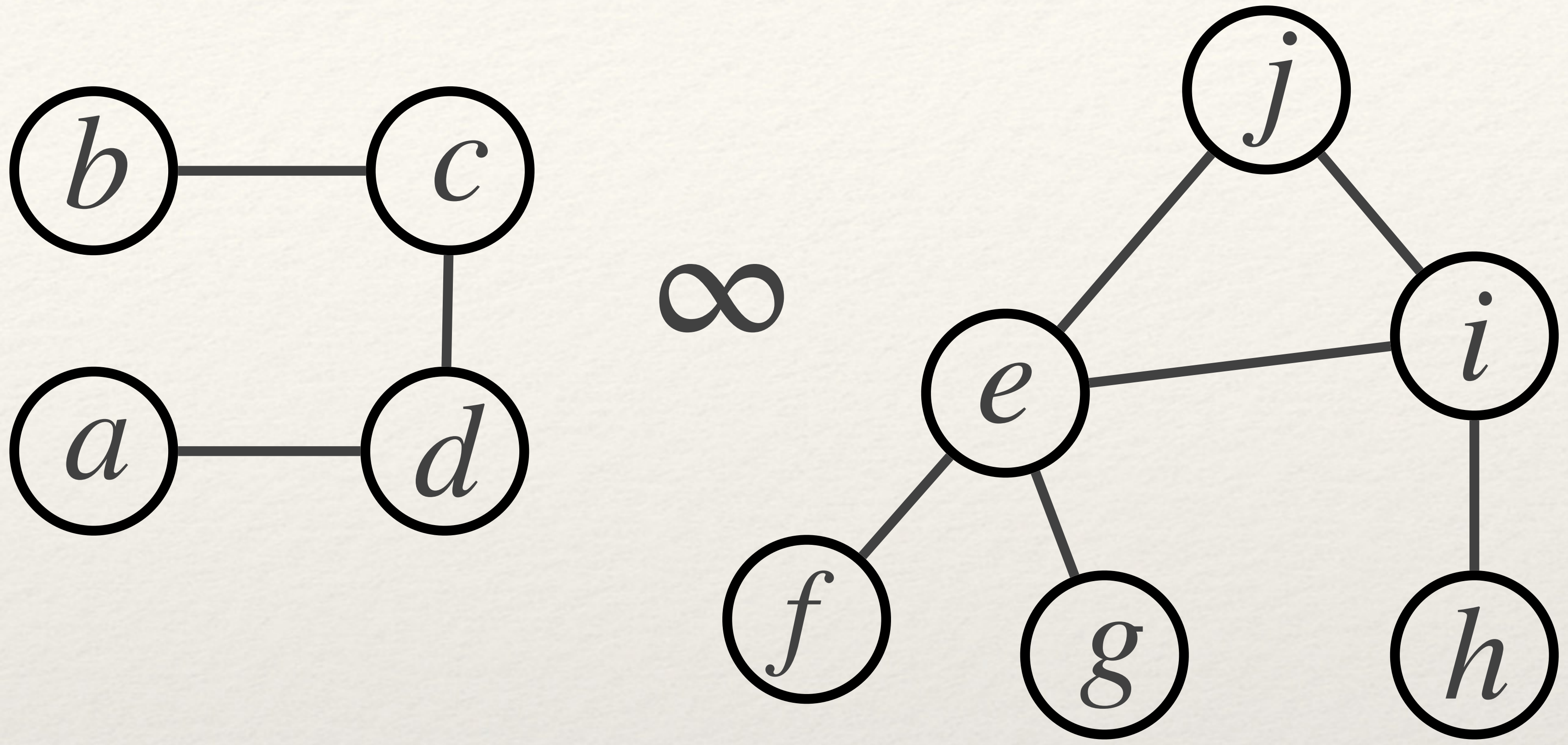
APL and diameter

APL: Average Path Length $$< l > = \frac{\sum_{ij} l_{ij}}{N(N-1)}$$

diameter of the graph: $$l_{max} = \max_{i,j} l_{ij}$$

APL and diameter with disconnected components

APL and diameter are undefined



$$< l > = \frac{\sum_{ij} l_{ij}}{N(N-1)} = \infty$$

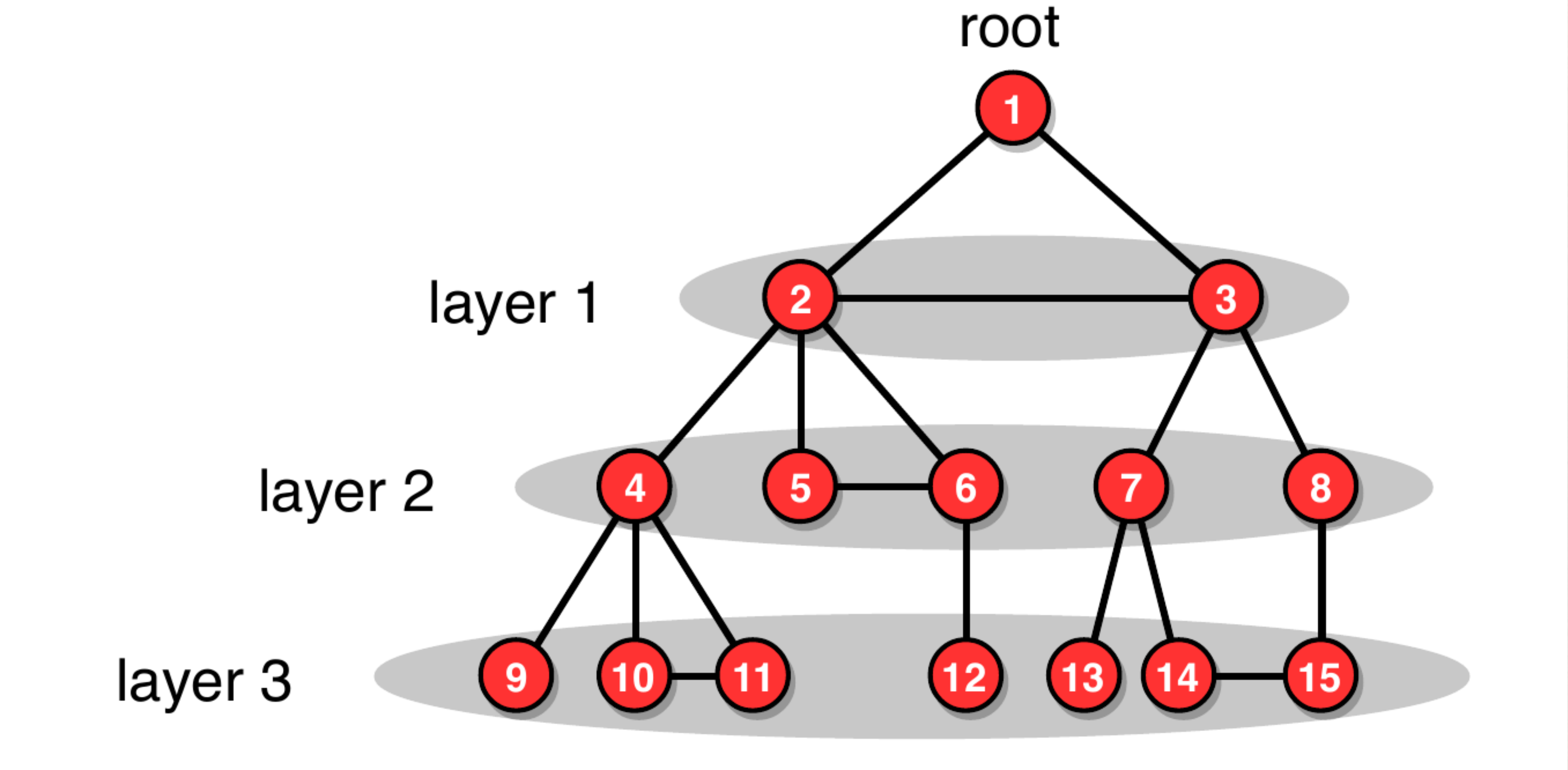
mathematical trick:

$$< l > = \left(\frac{\sum_{ij} \frac{1}{l_{ij}}}{N(N-1)} \right)^{-1}$$

Finding shortest paths

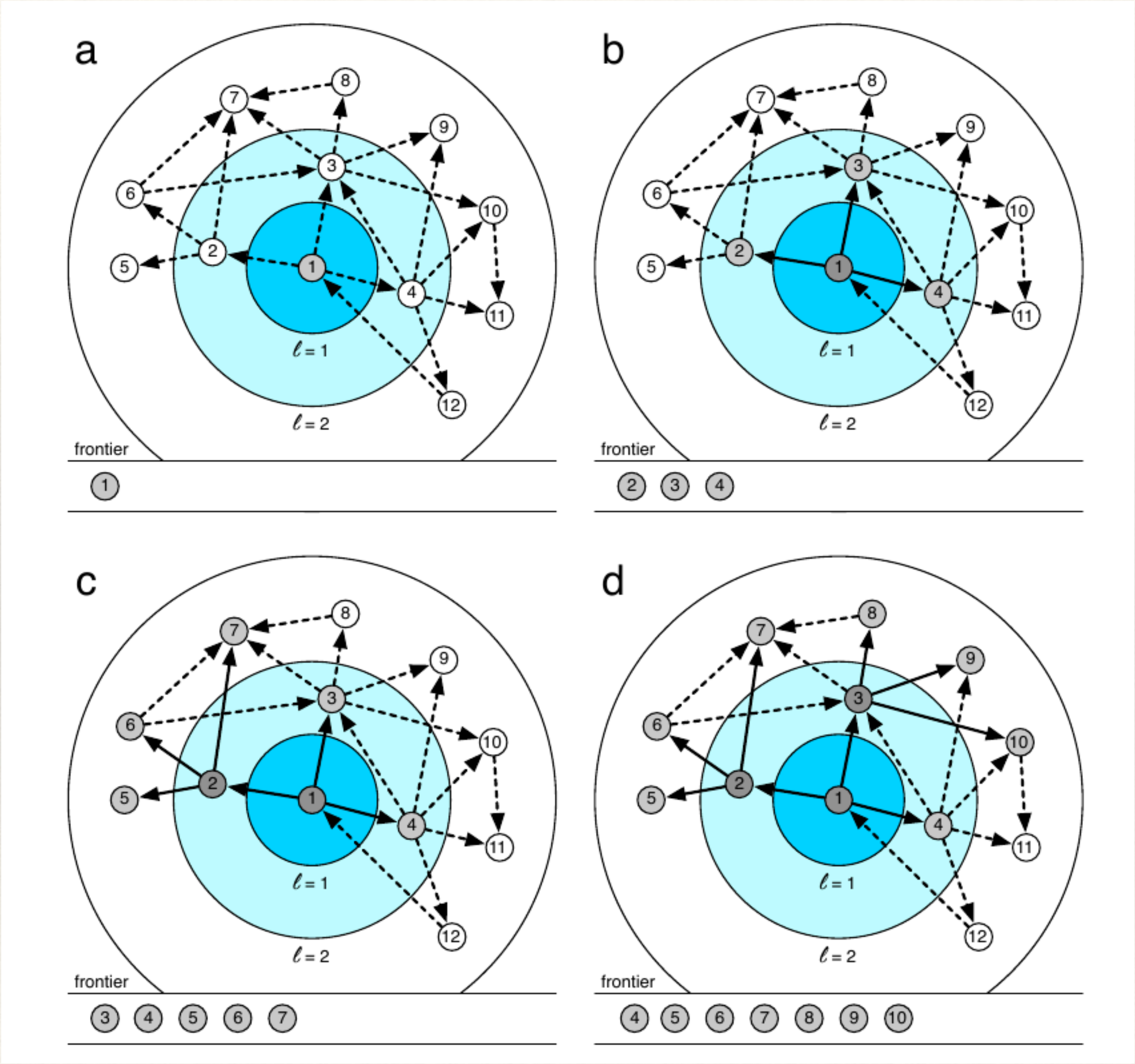
BFS

- Start from a source node (root)
- Visit the entire breadth of the network, within some distance from the source, before we move to a greater depth, farther away from the source
- Start from each node to find all-pairs-shortest-paths (slow: $O(N^2)$)



Breadth-first search (BFS)

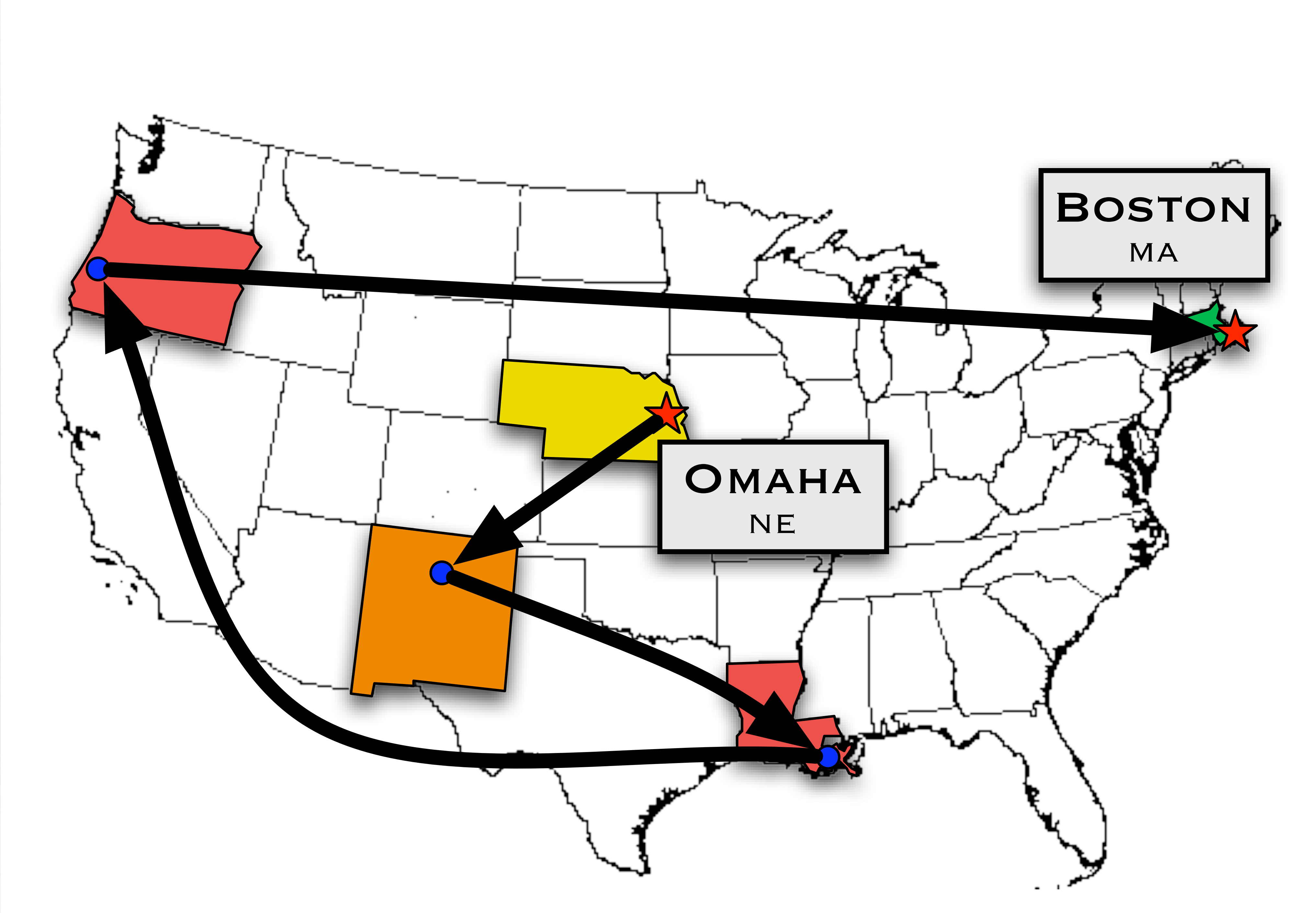
- Each node has an attribute storing its **distance** ℓ from the source, initially $\ell = \inf$ except $\ell(\text{source}) = 0$
- A queue (FIFO) holds the **frontier**, initially contains the source
- A directed **shortest path tree**, initially all the nodes and no links
- Iterate until the frontier is empty:
  1. Remove next node i in frontier
  2. For each neighbor/successor j of i with $\ell(j) = \inf$:
     1. Queue j into frontier
     2. $\ell(j) = \ell(i) + 1$
     3. Add link $(i \to j)$ to shortest-path tree



Small world phenomenon

Milgram's experiment

- Instructions: send to personal acquaintance who is more likely to know target
- 160 letters to people in Omaha, NE and Wichita, KS
- 2 targets in Mass: the wife of a student in Sharon and a stockbroker in Boston
- 42 letters made it back (only 26%)
- Average: 6.5 steps (range: 3-12 steps)
- Much lower than most people expected!
- "Small world" effect is still surprising



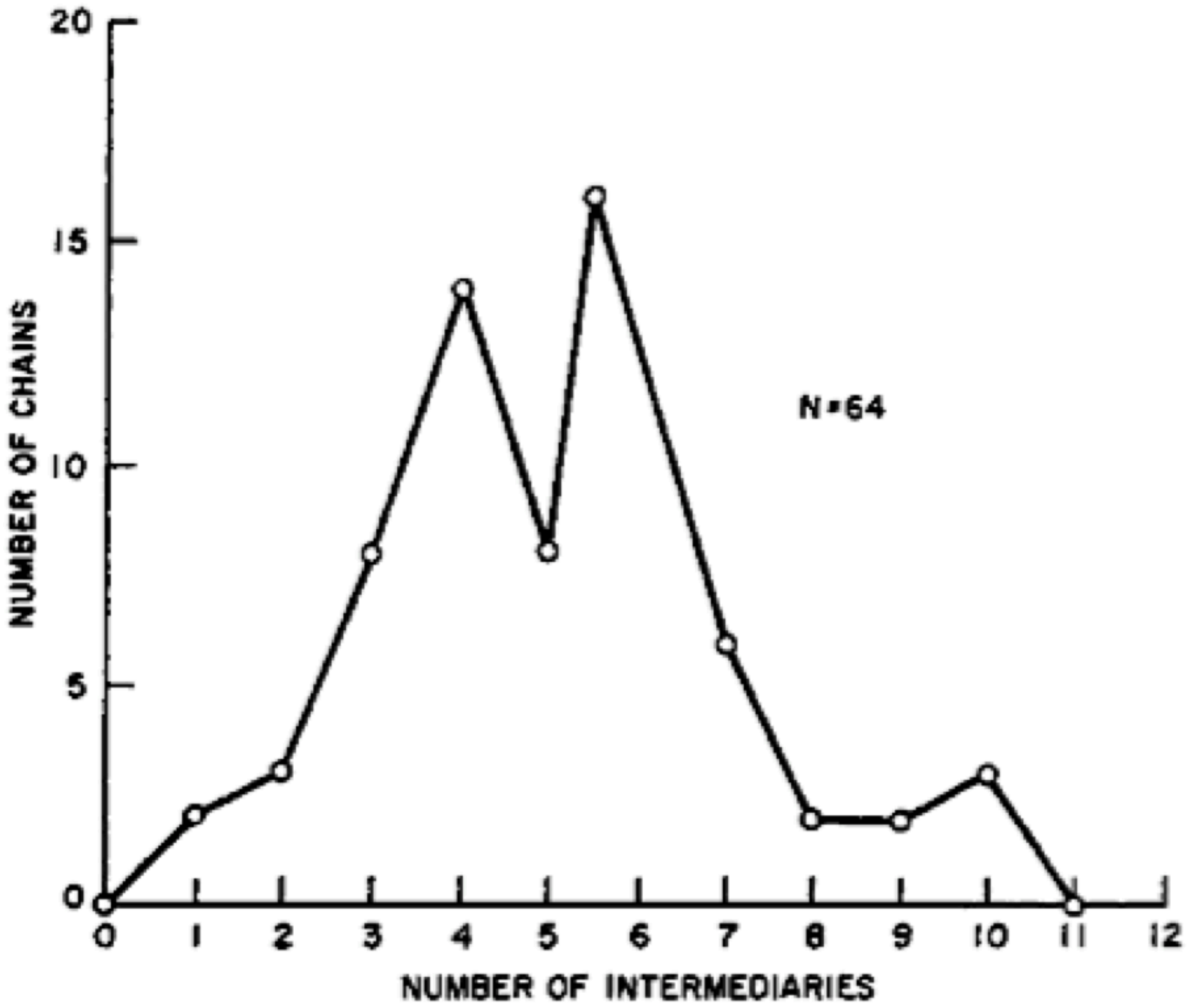


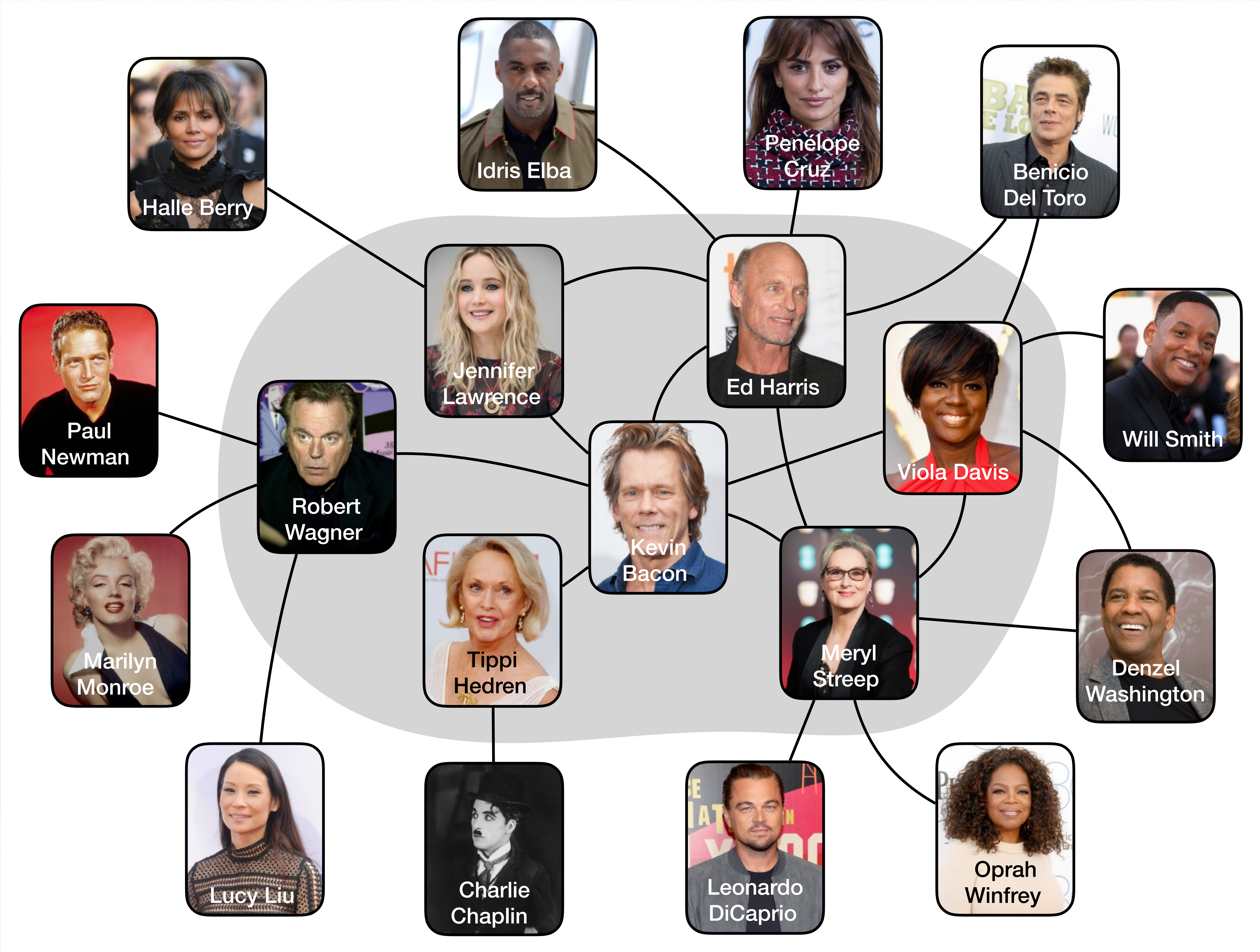
Figure 2.10: A histogram from Travers and Milgram's paper on their small-world experiment [391]. For each possible length (labeled "number of intermediaries" on the x-axis), the plot shows the number of successfully completed chains of that length. In total, 64 chains reached the target person, with a median length of six.

Other small world experiments

- Erdös number
- Oracle of Bacon
- Yahoo! email (Kleinberg, 2003)
- Instant Messaging (Leskovec and Horvitz, 2007)
- Facebook (Boldi, Vigna and others, 2011)

The oracle of Bacon

- Movie co-star network
- oracleofbacon.com



- Not only Kevin Bacon (not the center of the Universe…

- Can you find two stars separated by more that four links? Play the game and try!

Instant messaging

- 240 millions of nodes
- median: 7
- average: 6.6

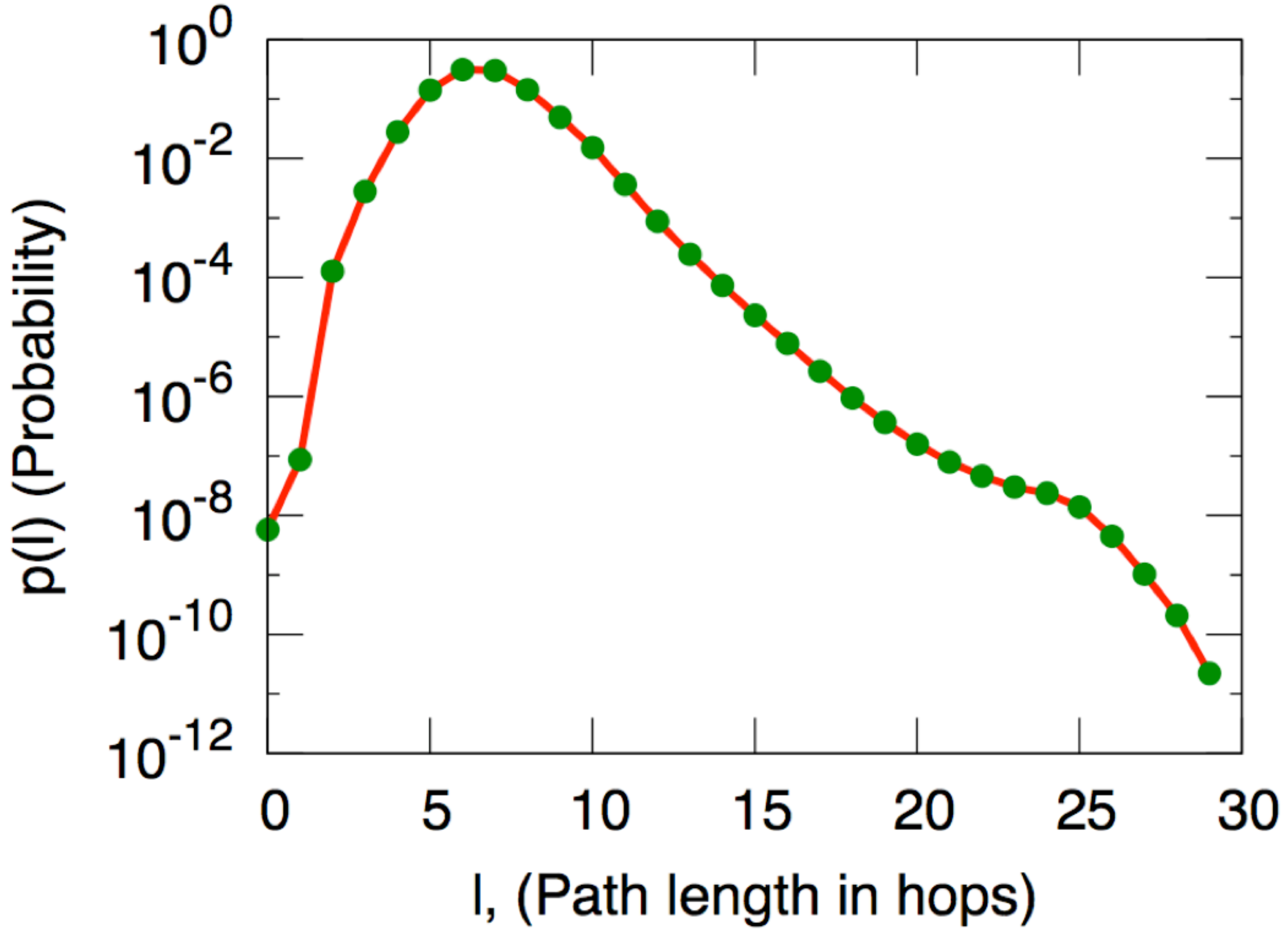


Figure 2.11: The distribution of distances in the graph of all active Microsoft Instant Messenger user accounts, with an edge joining two users if they communicated at least once during a month-long observation period [273].

The New York Times

Separating You and Me? 4.74 Degrees

By John Markoff and Somini Sengupta

Nov. 21, 2011

The world is even smaller than you thought.

Adding a new chapter to the research that cemented the phrase "six degrees of separation" into the language, scientists at Facebook and the University of Milan reported on Monday that the average number of acquaintances separating any two people in the world was not six but 4.74.

The original "six degrees" finding, published in 1967 by the psychologist Stanley Milgram, was drawn from 296 volunteers who were asked to send a message by postcard, through friends and then friends of friends, to a specific person in a Boston suburb.

When a short path is *really* short?

- It depends on the size of the network!
- Observe the relationship between APL and network size when considering networks (or subnetworks) of different sizes
- We say that the average path length is **short** when **it grows very slowly** with the size of the network, say, logarithmically:

$$< l > \approx logN$$ "Small World"

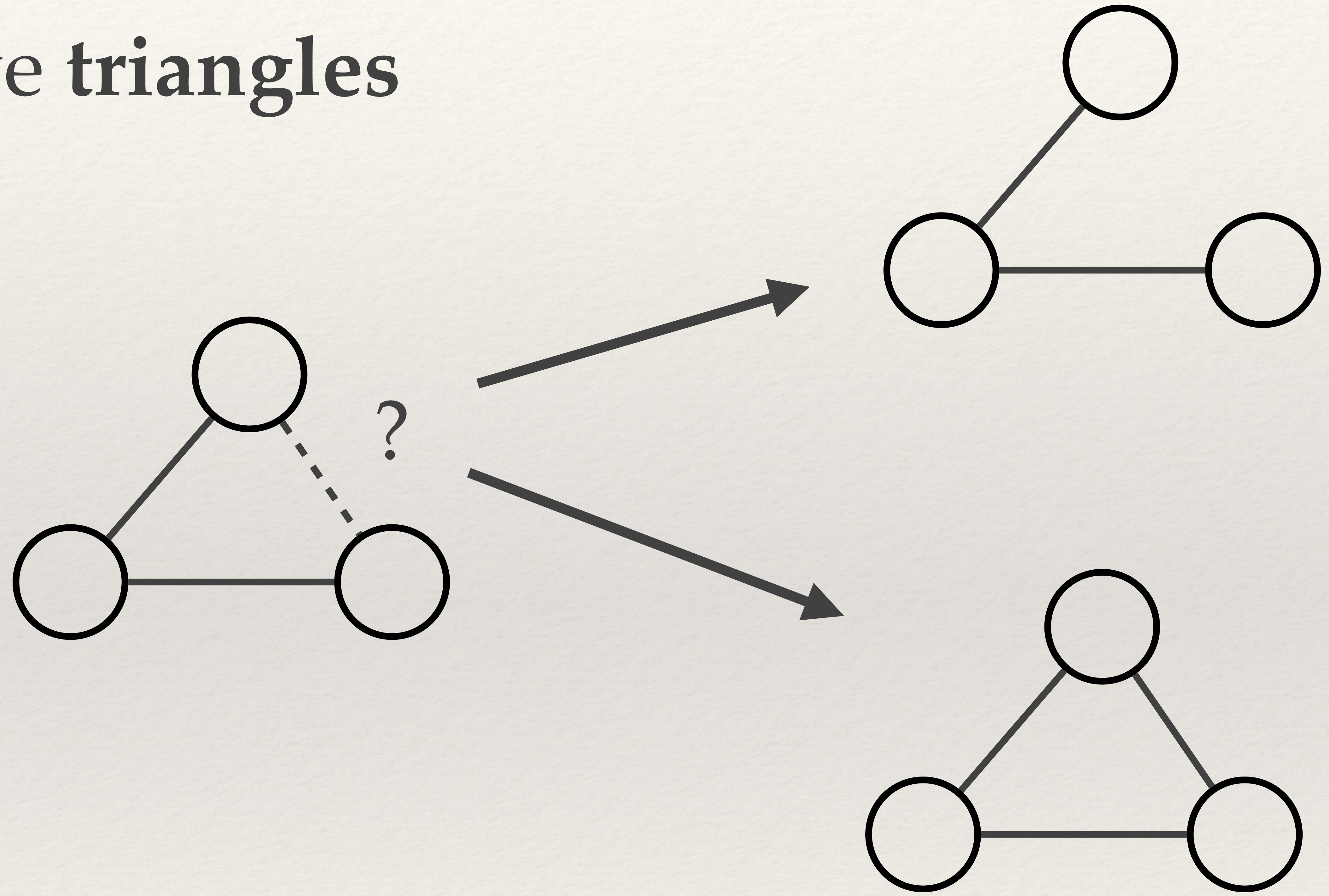
Small worlds

Table 2.1 Average path length and clustering coefficient of various network examples. The networks are the same as in Table 1.1, their numbers of nodes and links are listed as well. Link weights are ignored. The average path length is measured only on the giant component; for directed networks we consider directed paths in the giant strongly connected component. To measure the clustering coefficient in directed networks, we ignore link directions.

Network	Nodes (N)	Links (L)	Average path length ($\langle \ell \rangle$)	Clustering coefficient (C)
Facebook Northwestern Univ.	10,567	488,337	2.7	0.24
IMDB movies and stars	563,443	921,160	12.1	0
IMDB co-stars	252,999	1,015,187	6.8	0.67
Twitter US politics	18,470	48,365	5.6	0.03
Enron Email	87,273	321,918	3.6	0.12
Wikipedia math	15,220	194,103	3.9	0.31
Internet routers	190,914	607,610	7.0	0.16
US air transportation	546	2,781	3.2	0.49
World air transportation	3,179	18,617	4.0	0.49
Yeast protein interactions	1,870	2,277	6.8	0.07
C. elegans brain	297	2,345	4.0	0.29
Everglades ecological food web	69	916	2.2	0.55

A friend of a friend

- in social networks we have **triangles**



common in social networks

Clustering Coefficient

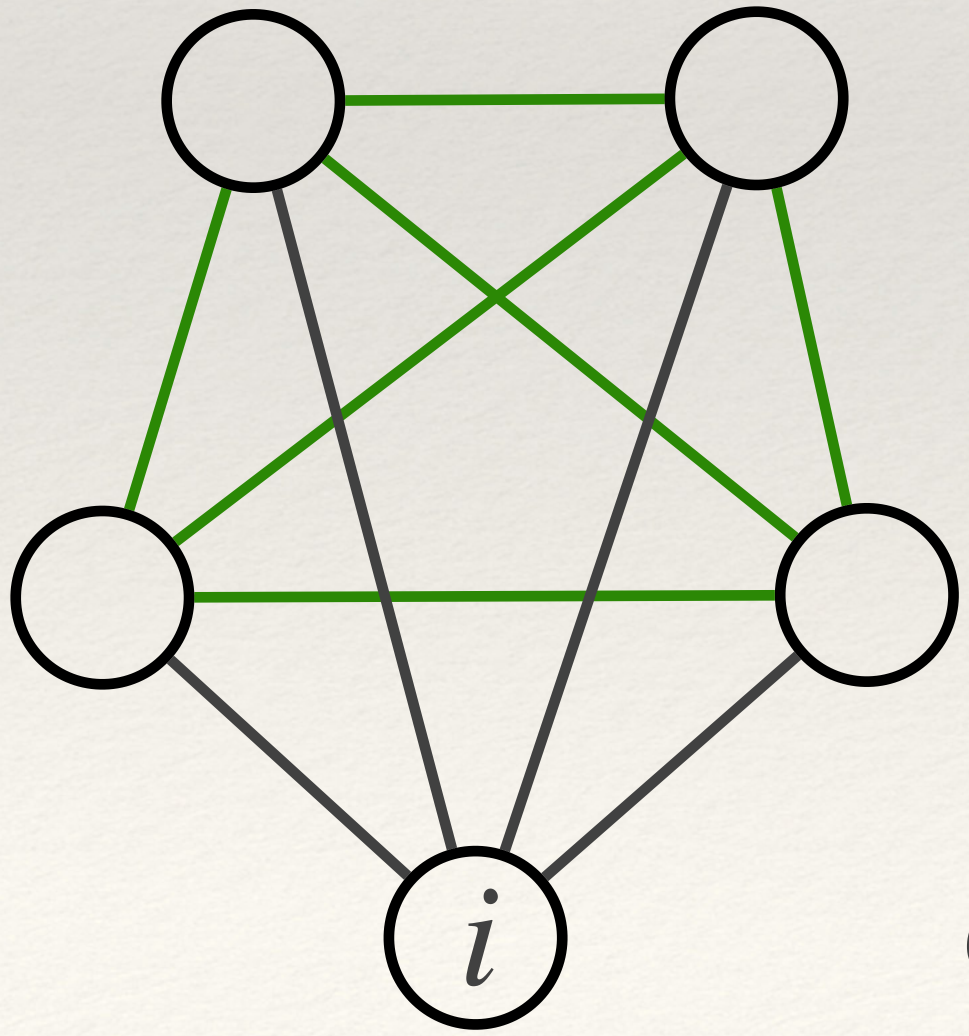
- The **clustering coefficient** of a node is the **fraction of pairs of the node's neighbors that are connected to each other.**
- if N_i is the set of neighbors of i, and $\boldsymbol{\tau}(i)$ is the number of triangles involving i:

$$C(i) = \frac{\tau(i)}{\tau_{max}(i)} = \frac{\tau(i)}{\binom{k_i}{2}} = \frac{2\tau(i)}{k_i(k_i - 1)} \qquad : |N_i| = k_i$$

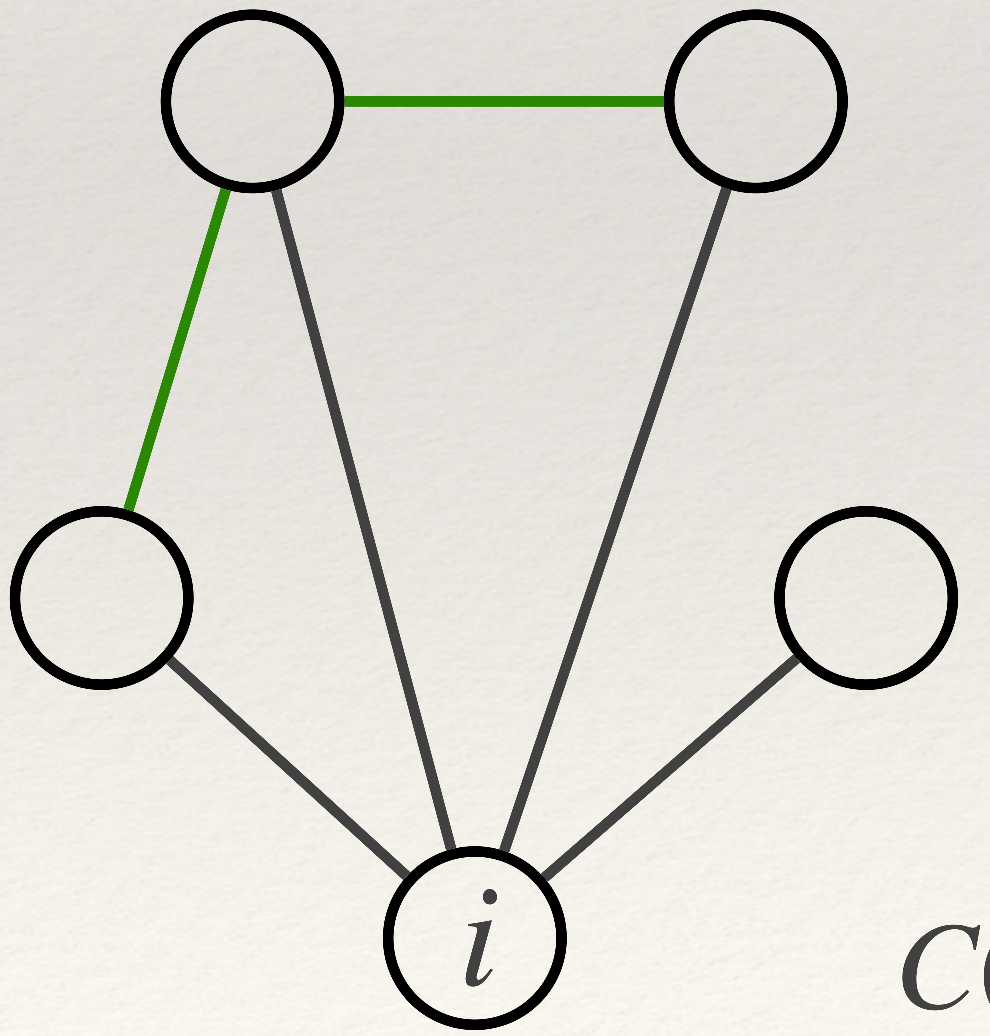
Network clustering coefficient

- The clustering coefficient of the network is the average of the clustering coefficients of the nodes

$$C = \frac{\sum_{i:k_i>1} C(i)}{N_{k>1}}$$



$$C(i) = \frac{2 \cdot 6}{4 \cdot 3} = 1$$



$$C(i) = \frac{2 \cdot 2}{4 \cdot 3} = \frac{1}{3}$$

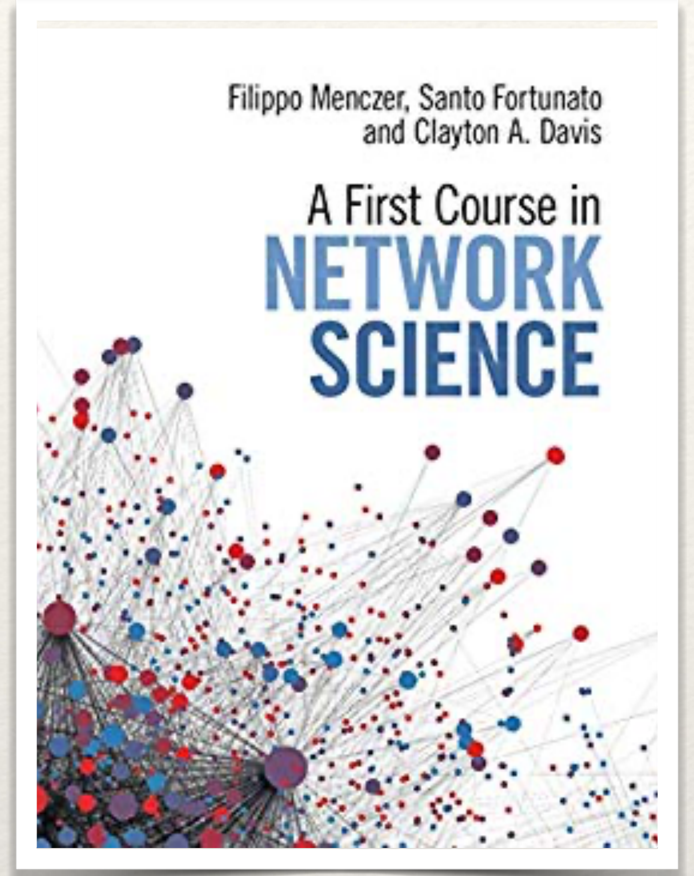
Network clustering coefficient

- Some networks, e.g., social networks, tend to have high clustering coefficients because of **triadic closure**: we meet through common friends
- Other networks, e.g., bipartite and tree-like networks, have low clustering coefficient

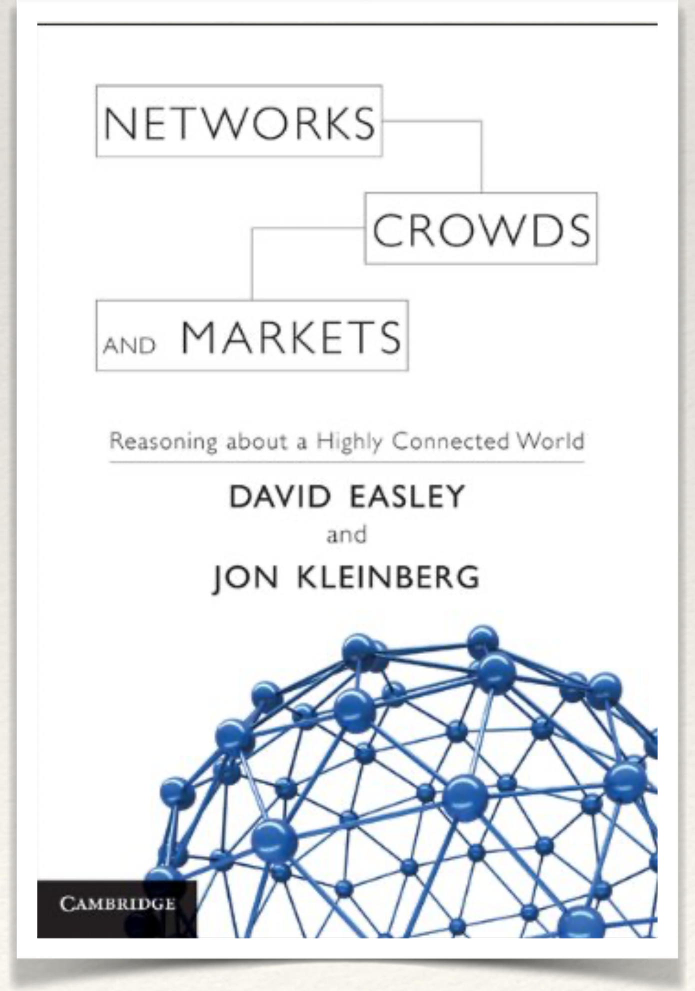
Table 2.1 Average path length and clustering coefficient of various network examples. The networks are the same as in Table 1.1, their numbers of nodes and links are listed as well. Link weights are ignored. The average path length is measured only on the giant component; for directed networks we consider directed paths in the giant strongly connected component. To measure the clustering coefficient in directed networks, we ignore link directions.

Network	Nodes (N)	Links (L)	Average path length ($\langle \ell \rangle$)	Clustering coefficient (C)
Facebook Northwestern Univ.	10,567	488,337	2.7	0.24
IMDB movies and stars	563,443	921,160	12.1	0
IMDB co-stars	252,999	1,015,187	6.8	0.67
Twitter US politics	18,470	48,365	5.6	0.03
Enron Email	87,273	321,918	3.6	0.12
Wikipedia math	15,220	194,103	3.9	0.31
Internet routers	190,914	607,610	7.0	0.16
US air transportation	546	2,781	3.2	0.49
World air transportation	3,179	18,617	4.0	0.49
Yeast protein interactions	1,870	2,277	6.8	0.07
C. elegans brain	297	2,345	4.0	0.29
Everglades ecological food web	69	916	2.2	0.55

References



[ns1] Chapter 1 & 2



[ns2] Chapter 2 —->

https://www.cs.cornell.edu/home/kleinber/networks-book/networks-book-ch02.pdf