Graphs in the real world

Third in a 3-part series:

- Workshop 1: What is a graph and what can we do with it?
- Workshop 2: Graph algorithms: Traversing the tree and beyond
- **Workshop 3: Graphs in the real world**

Previous workshops are available on the WiDS YouTube channel

Julia Olivieri

jolivier@stanford.edu

Code for today:

https://github.com/juliaolivieri/WiDS_graph_examples

What is a graph?

A graph is a structure to represent relationships between things

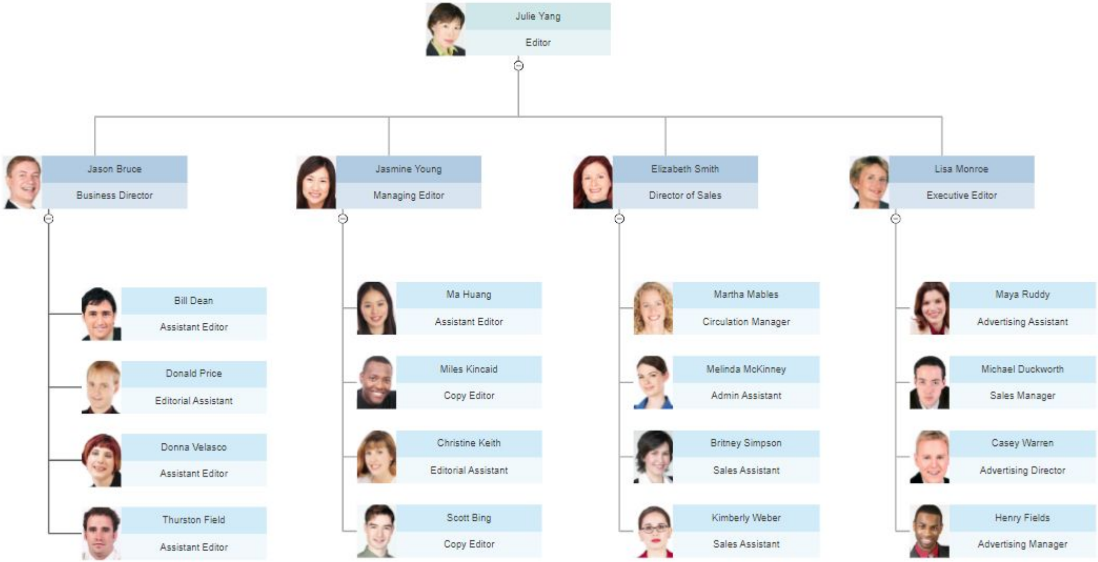


Image from https://www.smartdraw.com/organizational-chart/online-organizational-charts.htm

What is a graph?

A graph is a structure to represent relationships between things

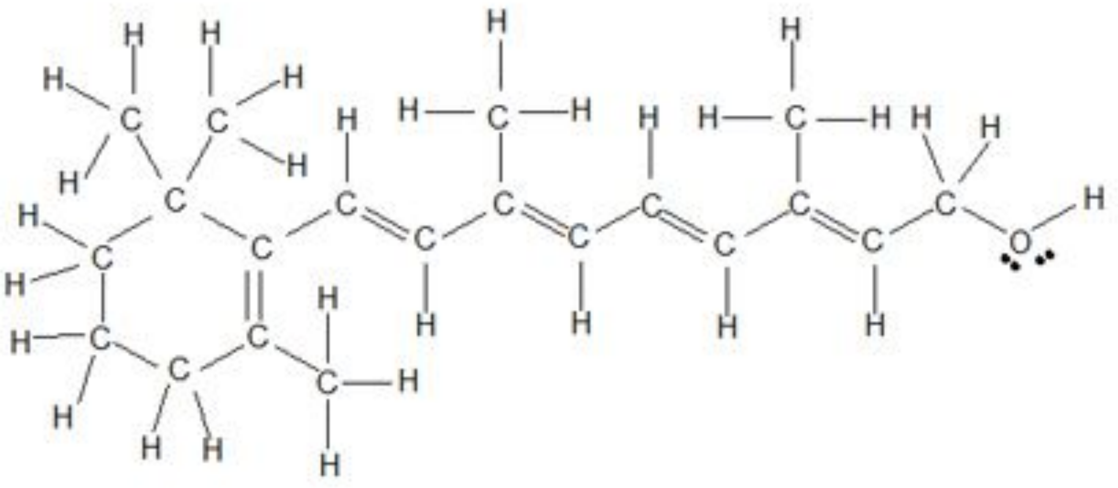


Image from
https://chem.libretexts.org/Bookshelves/Organic_Chemistry/Supplemental_Modules_(Organic_Chemistry)/Fundamentals/Structure_of_Organic_Molecules

What is a graph?

A graph is a structure to represent relationships between things

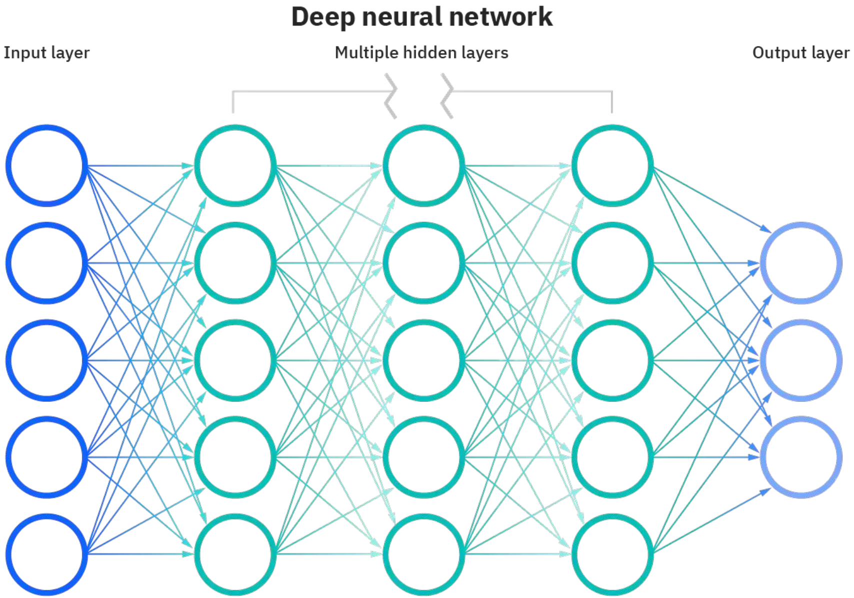


Image from https://www.ibm.com/cloud/learn/neural-networks

What is a graph?

A graph is a structure to represent relationships between things

Image from
https://gatton.uky.edu/about-us/stay-connected/news/2020/links-center-social-network-analysis-workshop-success

What is a graph?

A graph is a structure to represent relationships between things

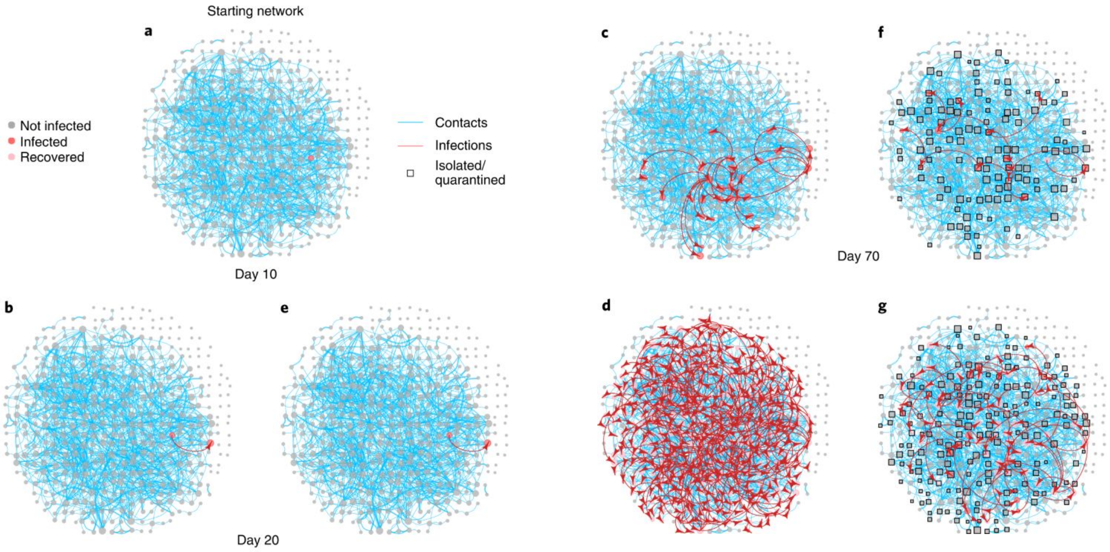


Image from https://www.nature.com/articles/s41591-020-1036-8

What is a graph?

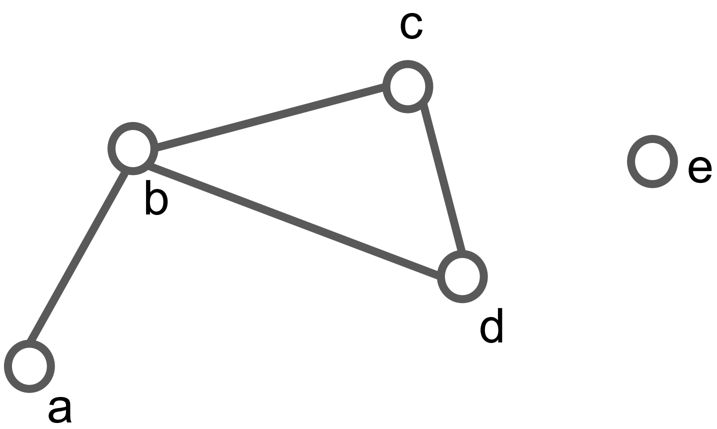
A graph **G** is a pair of sets (**V**, **E**) satisfying the following two conditions:

1. **V** is finite and non-empty
2. Each element of **E** is a 2-element subset of **V**

Example:

V = {1, 2, 3, 4, 5}

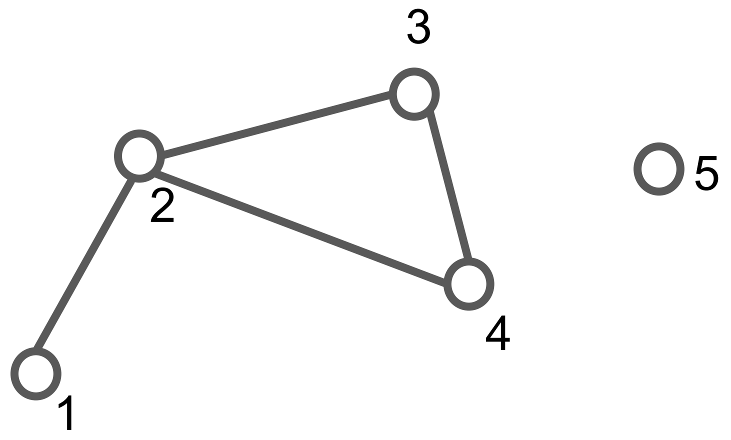
E = {(1, 2), (2, 4), (2, 3), (3, 4)}



Data structures to store graphs: Adjacency list

Adjacency list/adjacency dictionary: For each vertex, we store a list of all the neighbors of that vertex

adj = { 1 : [2],	*adj* = [[2],
2 : [1, 3, 4],	[1, 3, 4],
3 : [2, 4]	[2, 4]
4 : [2, 3]	[2, 3]
5 : [] }	[] }



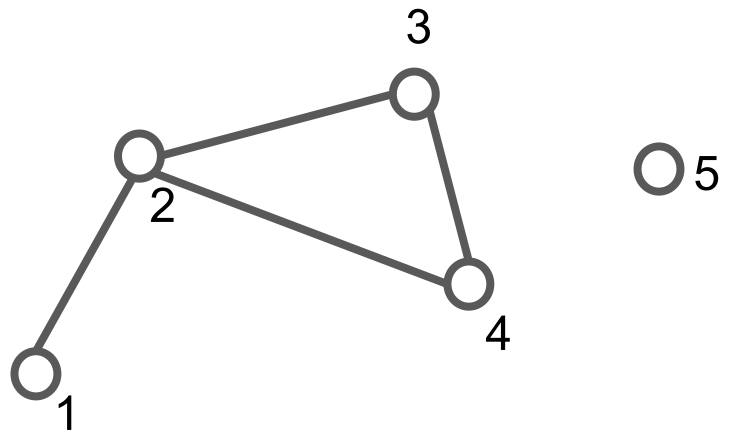
O(|V| + |E|) space complexity, good for traversal algorithms

Data structures to store graphs: Adjacency matrix

Adjacency matrix: Matrix A of dimension |V| x |V| defined by:

$$a_{ij} = \begin{cases} \mathbf{1} \text{ if } (i,j) \in E \\ \mathbf{0} \text{ otherwise} \end{cases}$$

	1	2	3	4	5
1	0	1	0	0	0
2	1	0	1	1	0
3	0	1	0	1	0
4	0	1	1	0	0
5	0	0	0	0	0



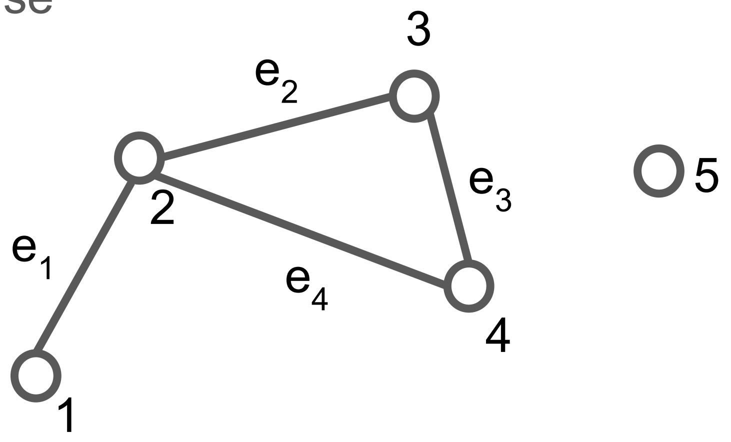
$O(|V|^2)$ space complexity, good for finding if edges exist

Data structures to store graphs: Incidence matrix

Incidence matrix: Matrix B of dimension |V| x |E|, with column for edge E = (i,j), i < j defined by:

$$b_v = \begin{cases} \mathbf{-1} \text{ if } v \in (i,j) \text{ and } i < j \\ \mathbf{1} \text{ if } v \in (i,j) \text{ and } i > j \\ \mathbf{0} \text{ otherwise} \end{cases}$$

	e_1	e_2	e_3	e_4
1	-1	0	0	0
2	1	-1	0	-1
3	0	1	-1	0
4	0	0	1	1
5	0	0	0	0



O(|V||E|) space complexity

Graph data structure comparison

	Adjacency list
Storage	O(|V| + |E|)
Add vertex	O(1)
Add edge	O(1)
Remove vertex	O(|E|)
Remove edge	O(|V|)
Are i and j adjacent?	O(|V|)

The data structure we choose depends on the application

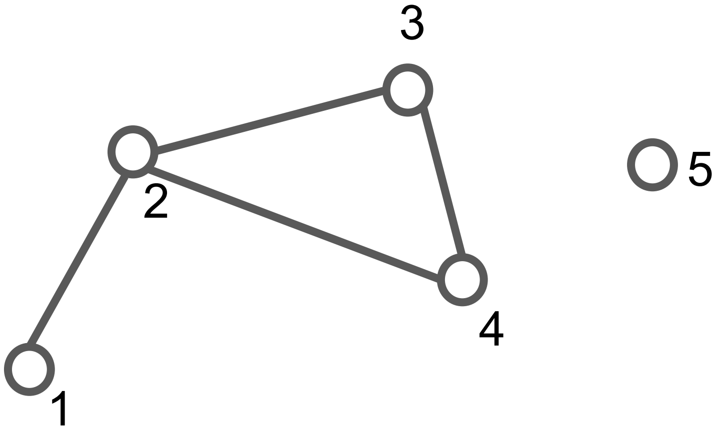
Why would we use an incidence matrix?

Table from https://en.wikipedia.org/wiki/Graph_(abstract_data_type)

Using the adjacency matrix to find number of walks between two vertices

For an adjacency matrix A, the matrix A^k has the following property: The element (i,j) of A^k gives the number of unique walks of length k between i and j

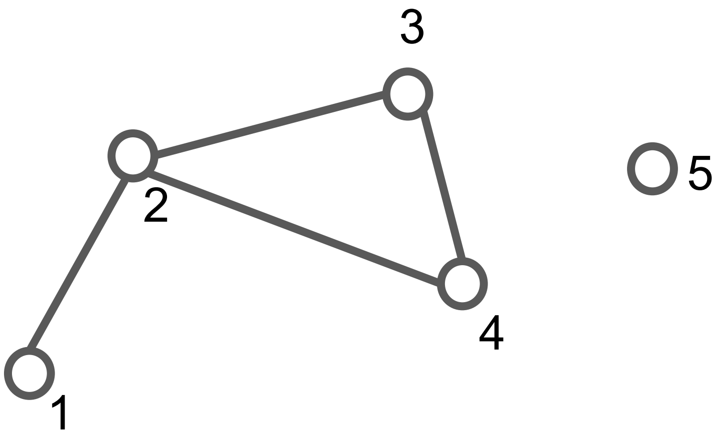
	1	2	3	4	5
1	0	1	0	0	0
2	1	0	1	1	0
3	0	1	0	1	0
4	0	1	1	0	0
5	0	0	0	0	0



Using the adjacency matrix to find number of walks between two vertices

For an adjacency matrix A, the matrix A^k has the following property: The element (i,j) of A^k gives the number of unique walks of length k between i and j

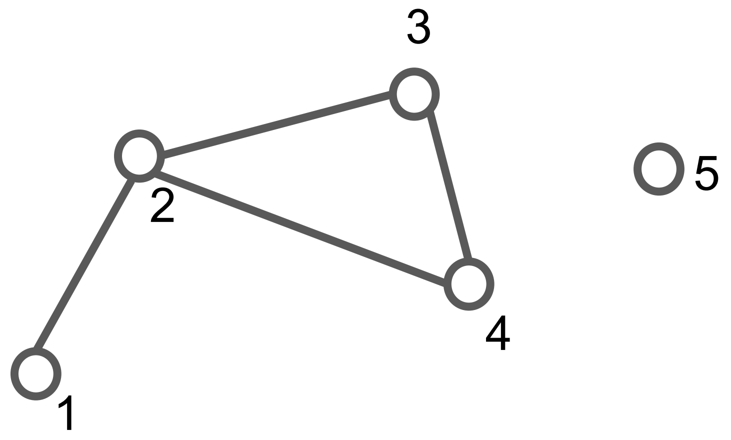
	1	2	3	4	5
1	1	0	1	1	0
2	0	3	1	1	0
3	1	1	2	1	0
4	1	1	1	2	0
5	0	0	0	0	0



Using the adjacency matrix to find number of walks between two vertices

For an adjacency matrix A, the matrix A^k has the following property: The element (i,j) of A^k gives the number of unique walks of length k between i and j

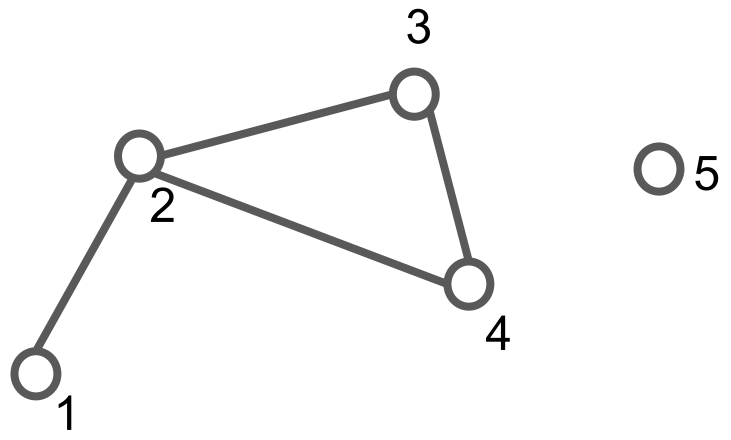
	1	2	3	4	5
1	0	3	1	1	3
2	3	2	4	4	0
3	1	4	2	3	0
4	1	4	3	2	0
5	0	0	0	0	0



Using the adjacency matrix to find number of walks between two vertices

For an adjacency matrix A, the matrix A^k has the following property: The element (i,j) of A^k gives the number of unique walks of length k between i and j

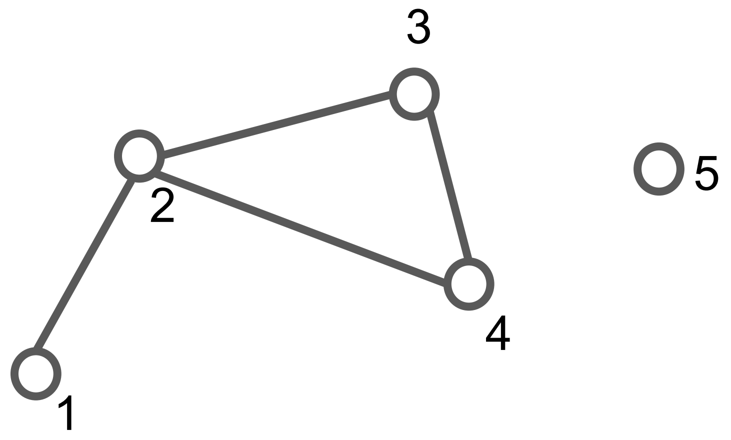
	1	2	3	4	5
1	3	2	4	4	0
2	2	11	6	6	0
3	4	6	7	6	0
4	4	6	6	7	0
5	0	0	0	0	0



Using the adjacency matrix to find number of walks between two vertices

For an adjacency matrix A, the matrix A^k has the following property: The element (i,j) of A^k gives the number of unique walks of length k between i and j

	1	2	3	4	5
1	2	11	6	6	0
2	11	14	17	17	0
3	6	17	12	13	0
4	6	17	13	12	0
5	0	0	0	0	0



The graph Laplacian

The graph Laplacian can be used to easily solve many graph theory problems

- Number of spanning trees
- Number of components
- Construction of low-dimensional embeddings

$$L = D - A \qquad\qquad L = B^T B$$

Where *D* is a diagonal matrix with the degree of each vertex

The graph Laplacian

The graph Laplacian can be used to easily solve many graph theory problems

- Number of spanning trees
- Number of components
- Construction of low-dimensional embeddings

	1	2	3	4	5
1	1	-1	0	0	0
2	-1	3	-1	-1	0
3	0	-1	2	-1	0
4	0	-1	-1	2	0
5	0	0	0	0	0

Many graph datasets are available online

Many are available to download as simple text files with one line per edge

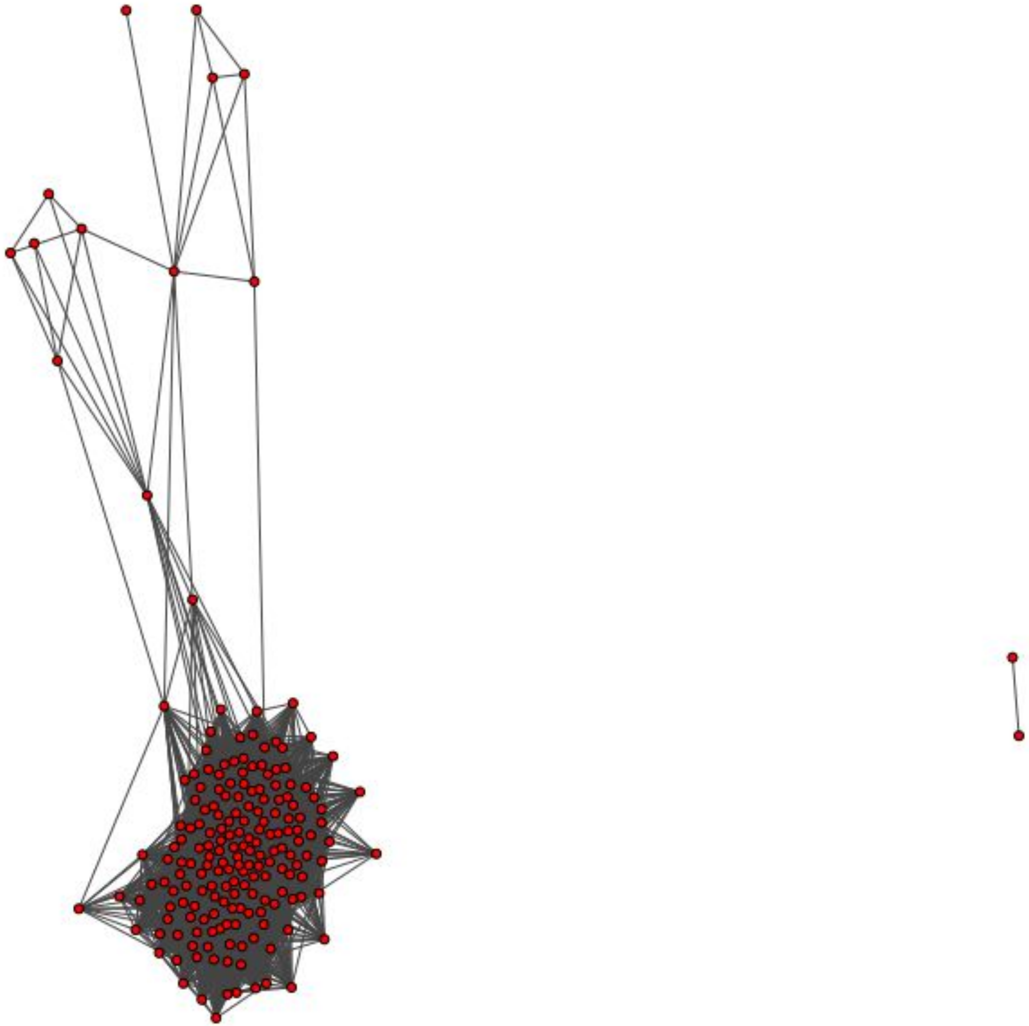
All analysis performed with the python igraph package

https://github.com/juliaolivieri/WiDS_graph_examples

```
# Berkely-Stanford web graph from 2002
# Nodes: 685230 Edges: 7600595
# FromNodeId    ToNodeId
1       2
1       5
1       7
1       8
1       9
1       11
1       17
1       254913
1       438238
254913  255378
254913  255379
254913  255383
254913  255384
254913  255392
254913  255393
254913  255394
254913  255396
254913  255399
254913  255401
254913  255402
254913  255561
254913  255562
254913  255637
254913  255638
254913  255662
```

Twitter

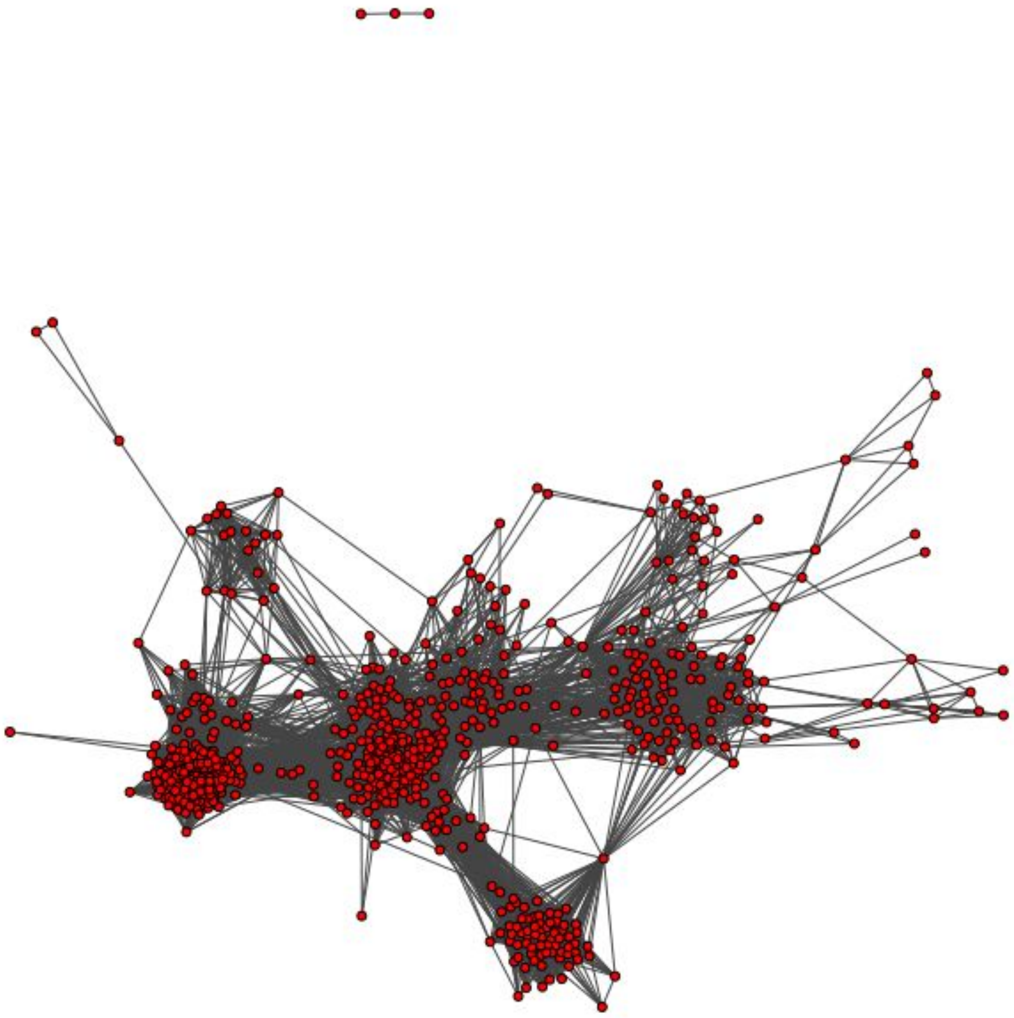
Includes the relationships between all friends of a single user (not plotted)



Downloaded from
https://snap.stanford.edu/data/ego-Twitter.html

Facebook

Includes the relationships between all friends of a single user (not plotted)

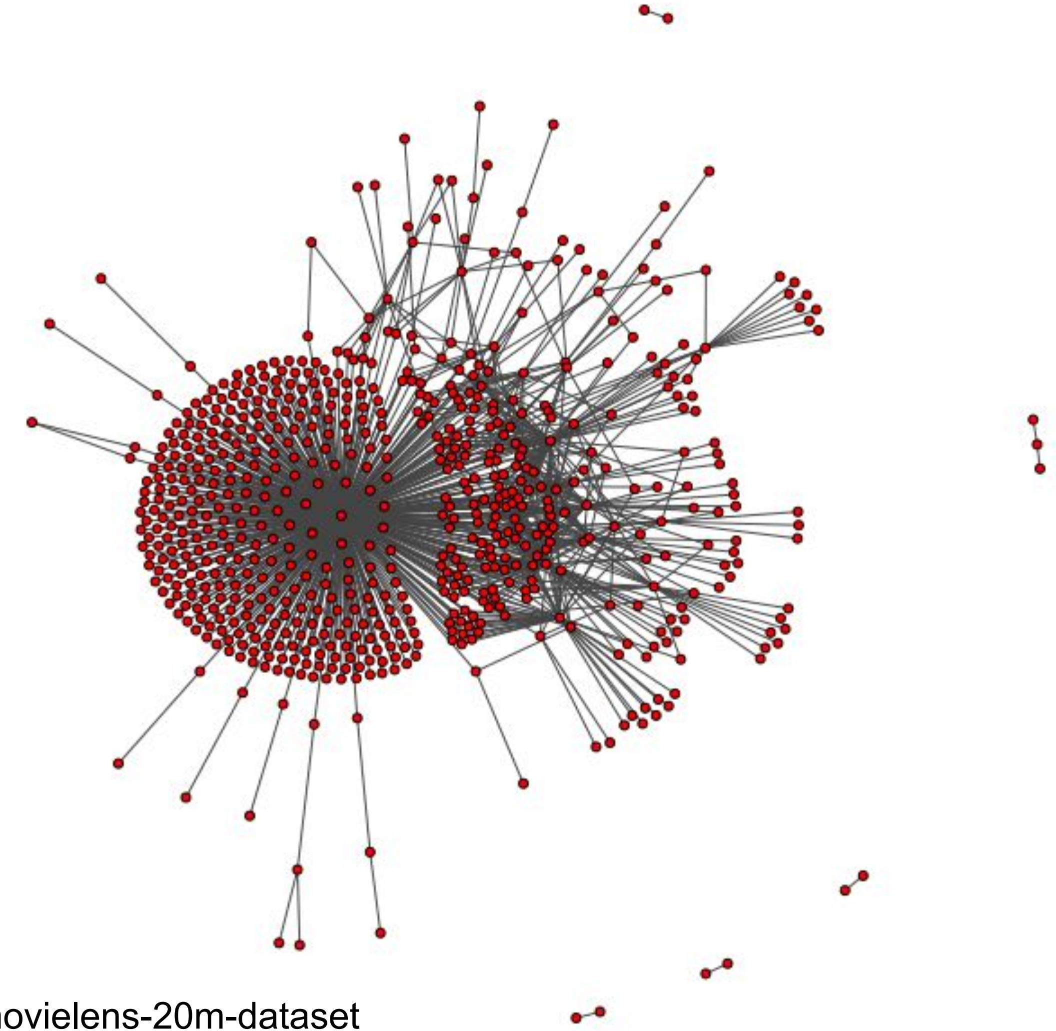


Downloaded from https://snap.stanford.edu/data/ego-Facebook.

Movielens

Vertices are either movies or users, and an edge means that the user has watched that movie.

Only 2000 vertices are plotted for clarity

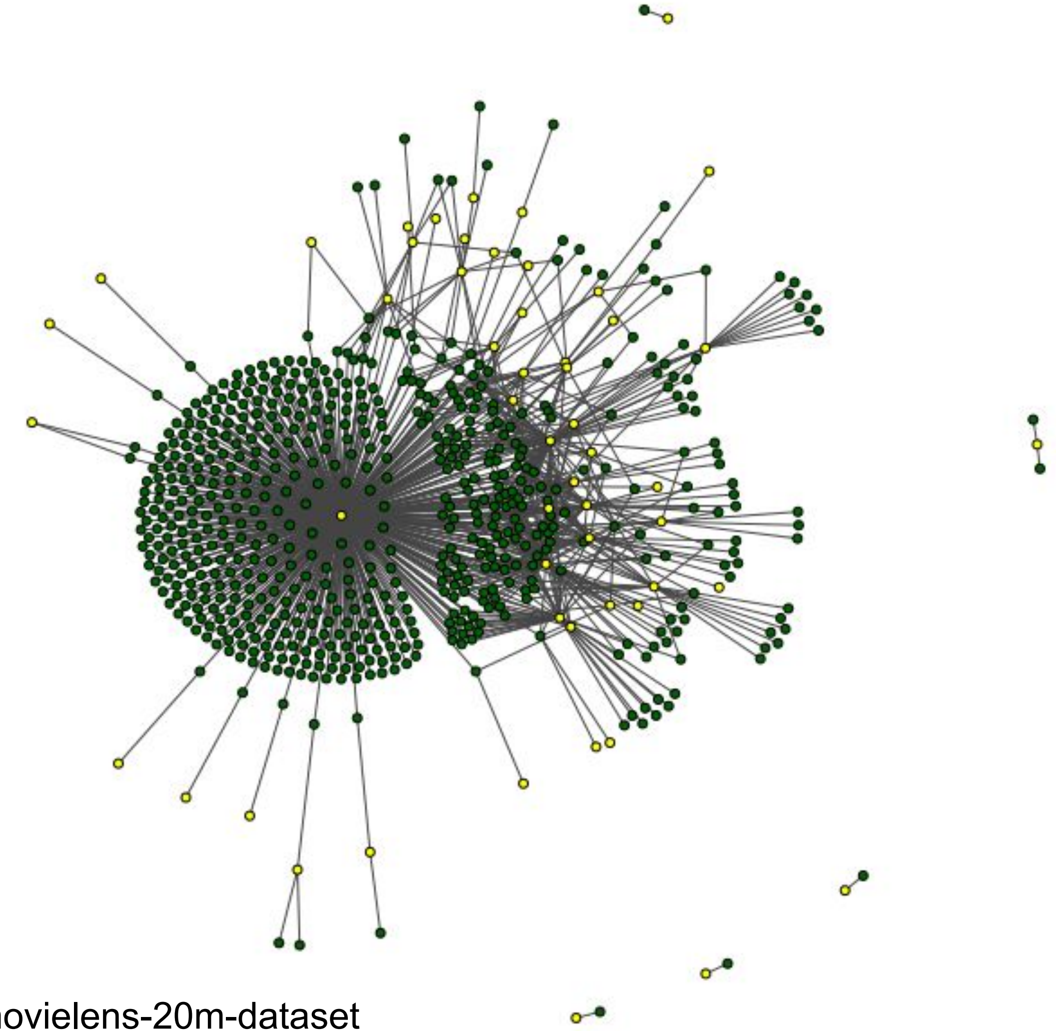


Downloaded from https://www.kaggle.com/grouplens/movielens-20m-dataset

Movielens

Vertices are either movies or users, and an edge means that the user has watched that movie.

Only 2000 vertices are plotted for clarity

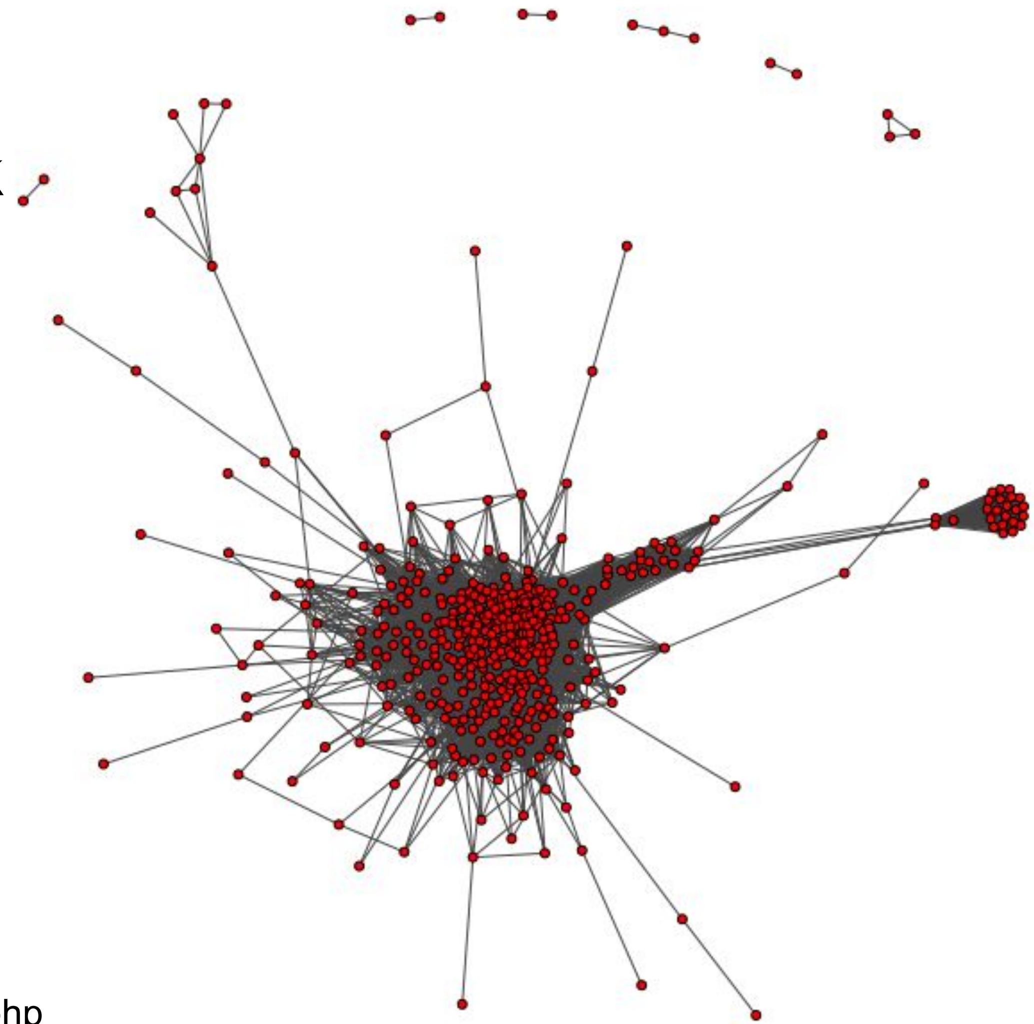


Downloaded from
https://www.kaggle.com/grouplens/movielens-20m-dataset

Mouse gene regulatory network

Vertices are genes and edges between them represent regulatory relationships.

Only 500 vertices are plotted for clarity

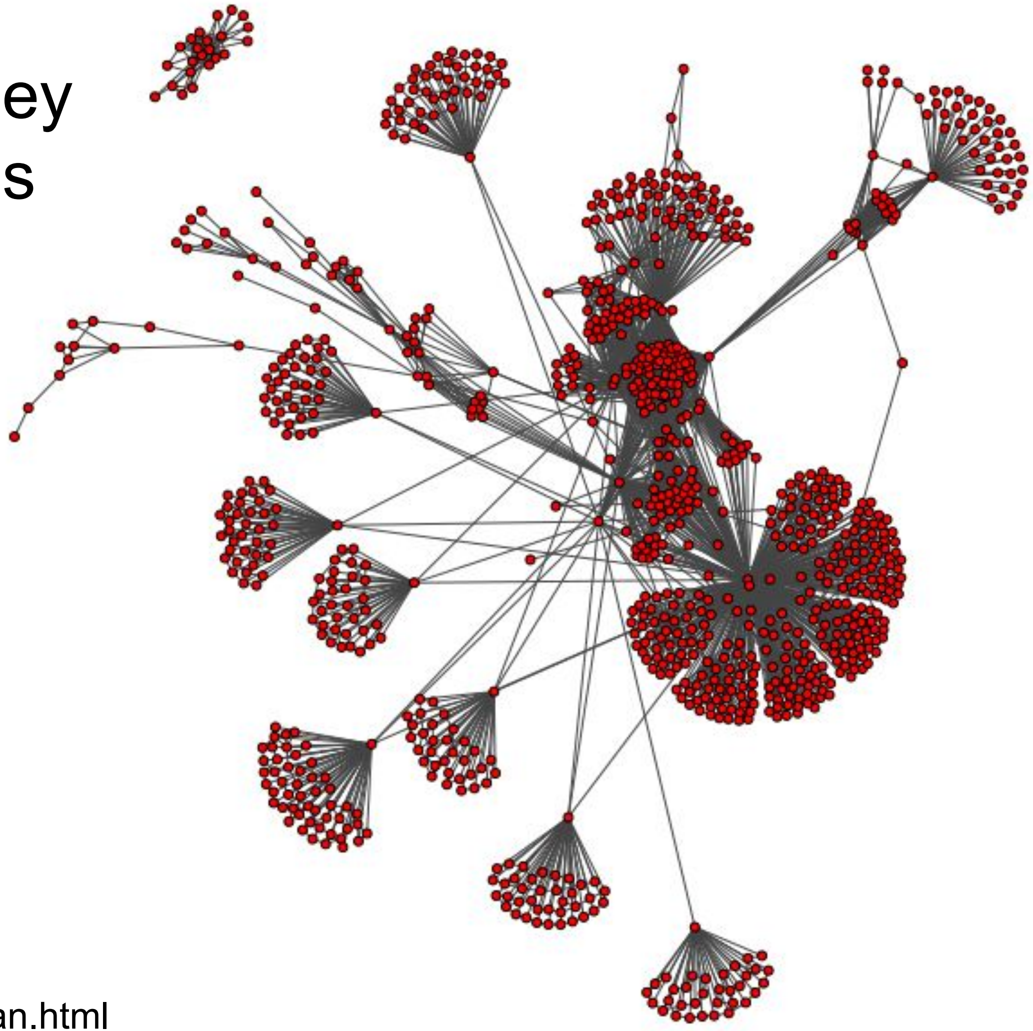


Downloaded from https://networkrepository.com/bio-mouse-gene.php

Stanford-Berkeley web connections

Vertices are Berkeley or Stanford webpages and edges are links between them

Only 1000 vertices are plotted for clarity



Downloaded from https://snap.stanford.edu/data/web-BerkStan.html

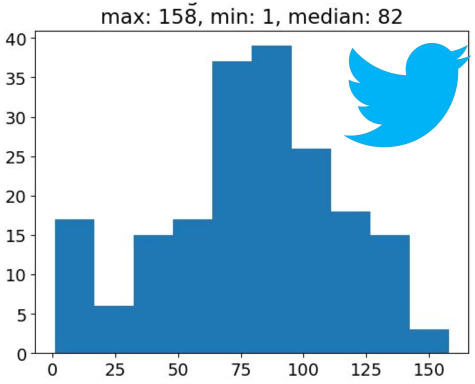
<table>
<tr><th></th><th></th><th></th><th></th><th></th><th></th></tr>
<tr><td>|V|</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

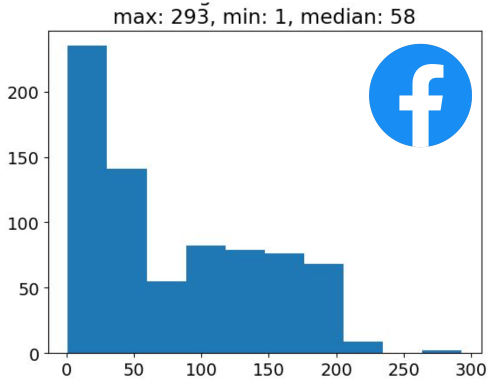
					
|V|	193	747	26,232	43,126	685,230

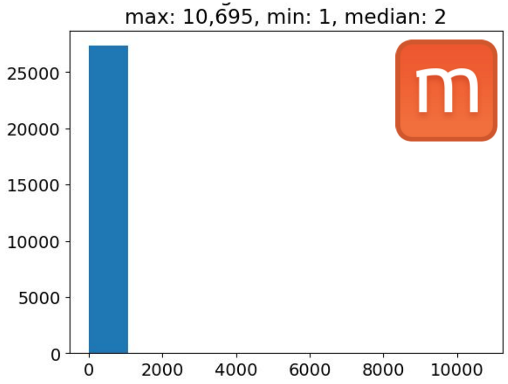
					
|V|	193	747	26,232	43,126	685,230
|E|					

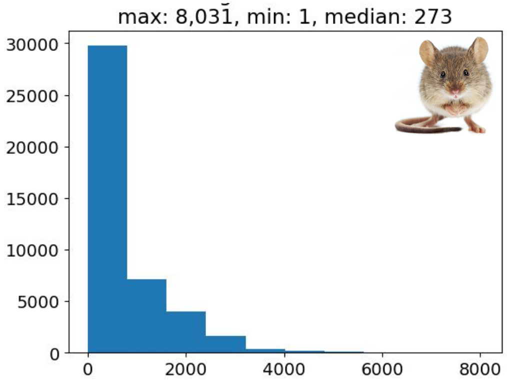
					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470

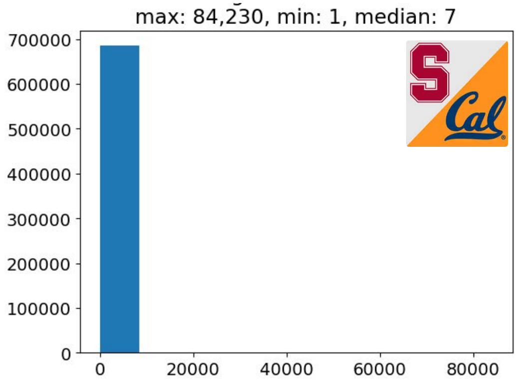
How does the distribution of degrees differ?





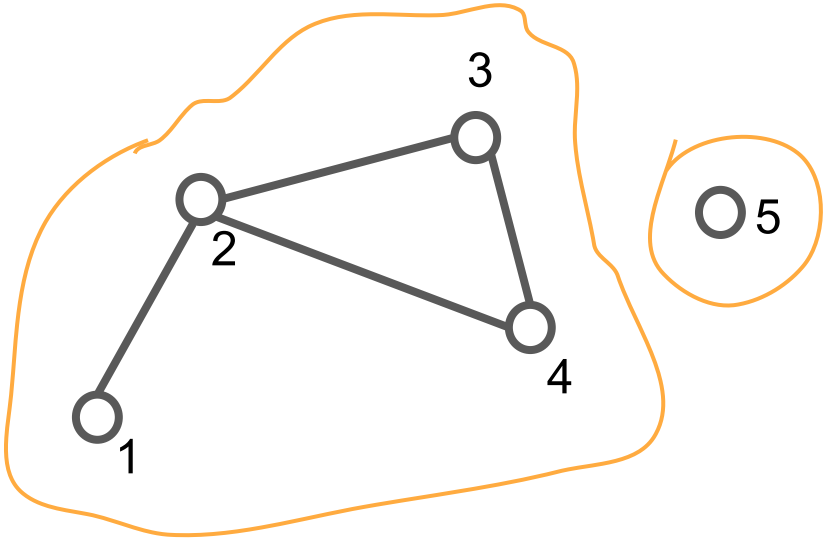






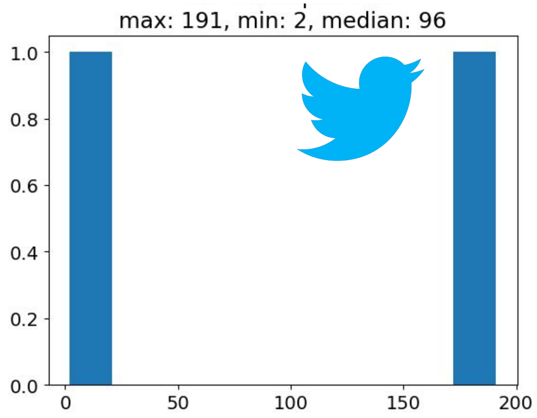
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components					

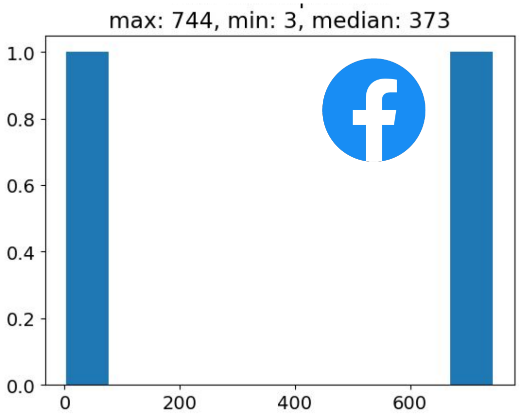
Component of a graph: subgraph within which every pair of vertices is connected by a path

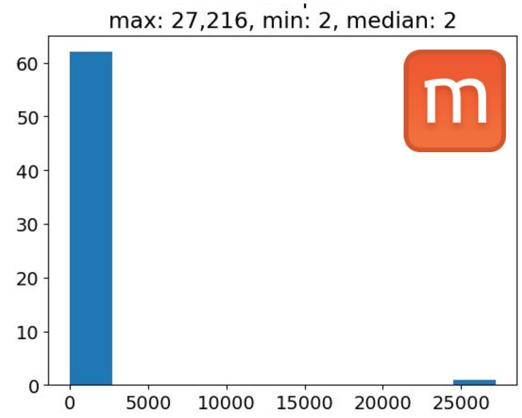


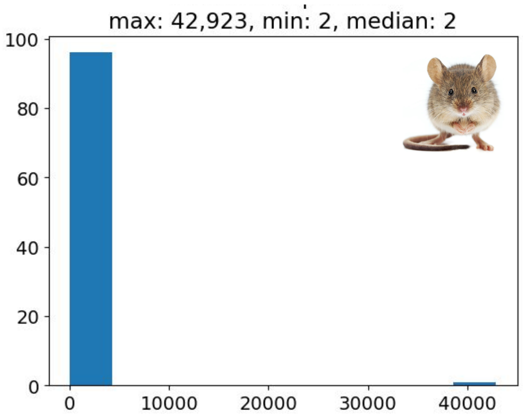
					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components	2	2	63	97	676

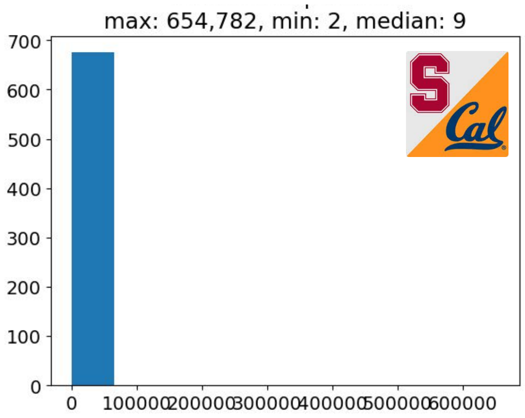
How many vertices are in each component?









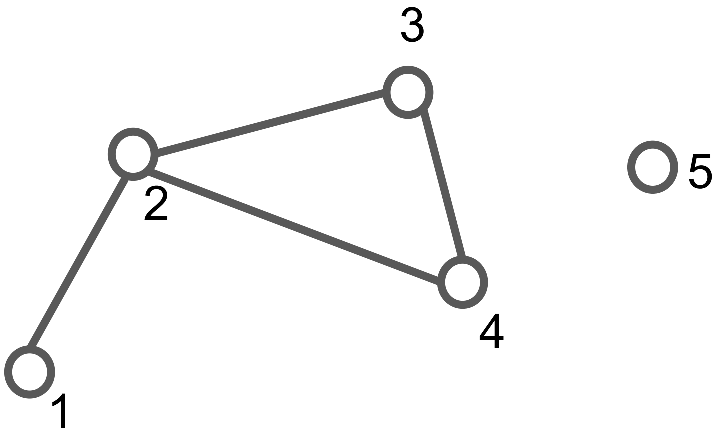


					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components	2	2	63	97	676
Density					

Density of a graph: The fraction of possible edges that are present

$$D = \frac{|E|}{\binom{|V|}{2}} = \frac{2|E|}{|V|(|V|-1)}$$

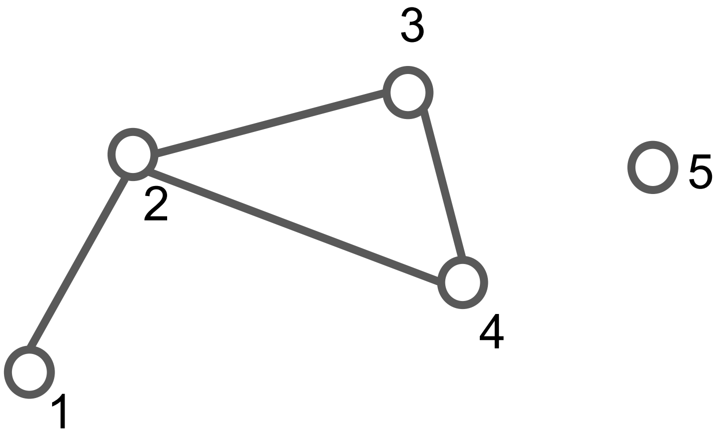
$$D = \frac{2 \times 4}{5 \times 4} = 0.4$$



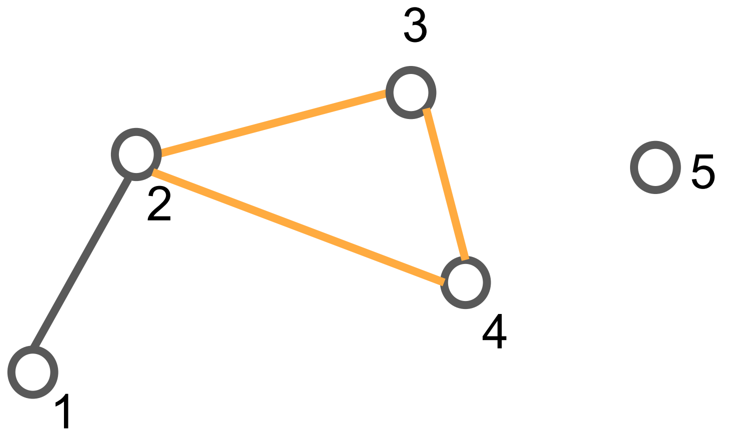
					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components	2	2	63	97	676
Density	0.410	0.108	0.000	0.016	0.000

					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components	2	2	63	97	676
Density	0.410	0.108	0.000	0.016	0.000
Girth					

Girth of a graph: The length of the shortest cycle



Girth of a graph: The length of the shortest cycle



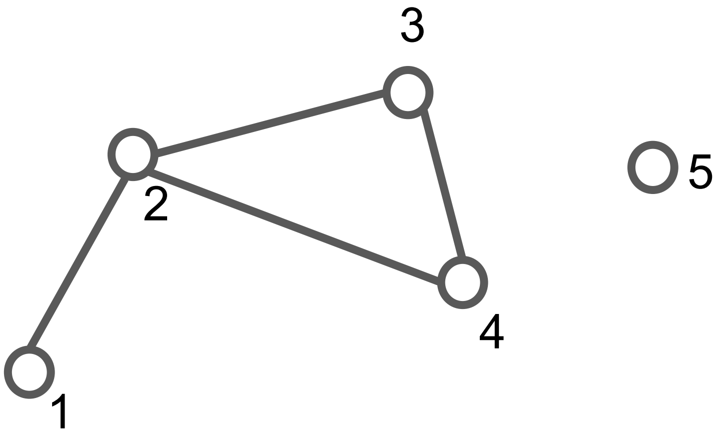
					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components	2	2	63	97	676
Density	0.410	0.108	0.000	0.016	0.000
Girth	3	3	4	3	3

					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components	2	2	63	97	676
Density	0.410	0.108	0.000	0.016	0.000
Girth	3	3	4	3	3
Assortativity					

Assortativity coefficient: The Pearson correlation coefficient of degree between vertices that share an edge

Correlation coefficient = -0.87

i	*j*	*d(i)*	*d(j)*
1	**2**	1	3
2	**3**	3	2
2	**4**	3	2
3	**4**	2	2



Assortativity coefficient: The Pearson correlation coefficient of degree between vertices that share an edge

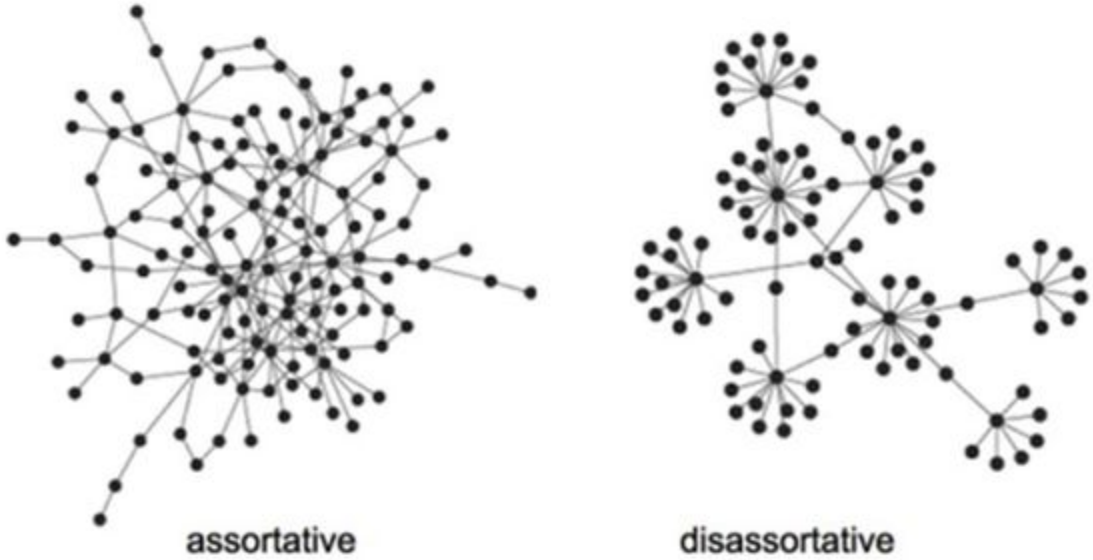


Image from
https://www.quora.com/What-is-the-difference-between-modularity-and-assortativity-in-network-science

					
|V|	193	747	26,232	43,126	685,230
|E|	7,597	30,025	174,844	14,461,095	6,649,470
Components	2	2	63	97	676
Density	0.410	0.108	0.000	0.016	0.000
Girth	3	3	4	3	3
Assortativity	-0.034	0.503	-0.111	0.002	-0.113

Thank you!

Email me with questions: jolivier@stanford.edu