1.00
Lecture 28 – Random Graphs and Components

John R Williams

Abel Sanchez

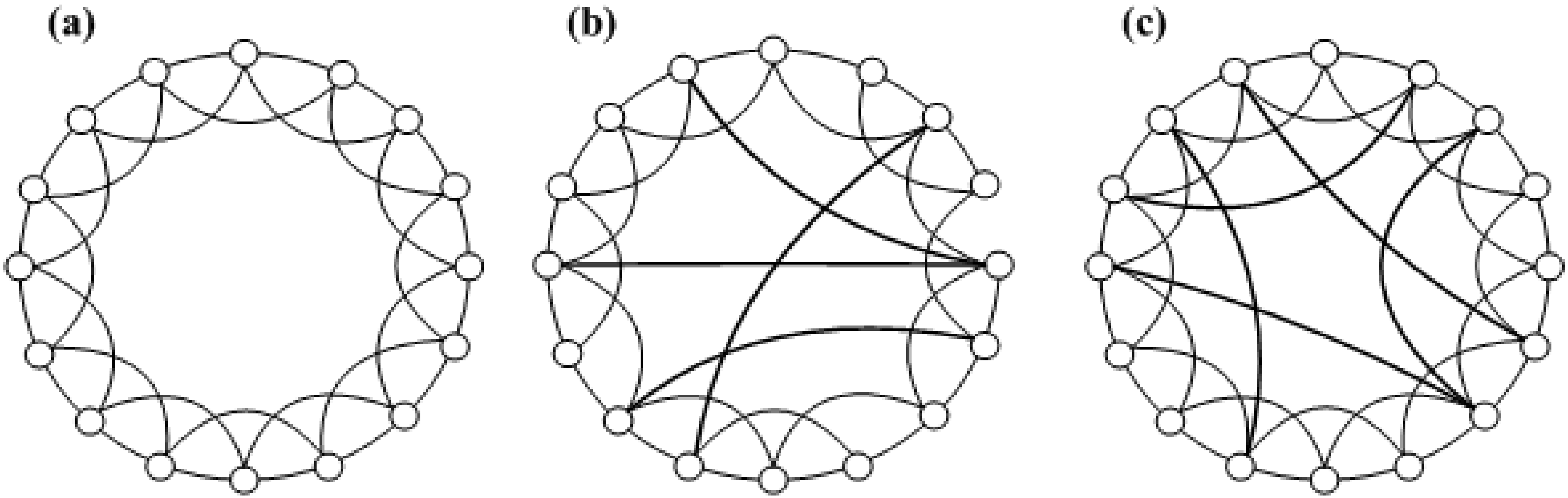
Network Models

- Networks can be classified by a model of their structure
- *Mechanistic*: Formalize a set of mathematical rules that produce a certain type of network
  - Often represent notions of cause and effect
  - e.g. growing a network with preferential attachment
- *Generative*: Generate network structure without the strong causality as in mechanistic models
  - e.g. random graphs: nodal links are created randomly using prescribed degree distribution

Network Models: Small World

- Small-world, or Watts and Strogatz, model
- Captures the effect that shortest path between most pairs of nodes in a network is small
  - Typically just a few steps even in networks of billions
- Concept popularized in the expression “six degrees of separation”
- Evidenced in the Millgram experiment
  - Further reading: Easley & Kleinberg, Section 2.3

Network Models: Small World



Reducing network diameter, (a) a regular graph in which each node is connected to four neighbors, (b) Watts and Strogatz's small-world model and (c) Newman an Watt's improved small-world model

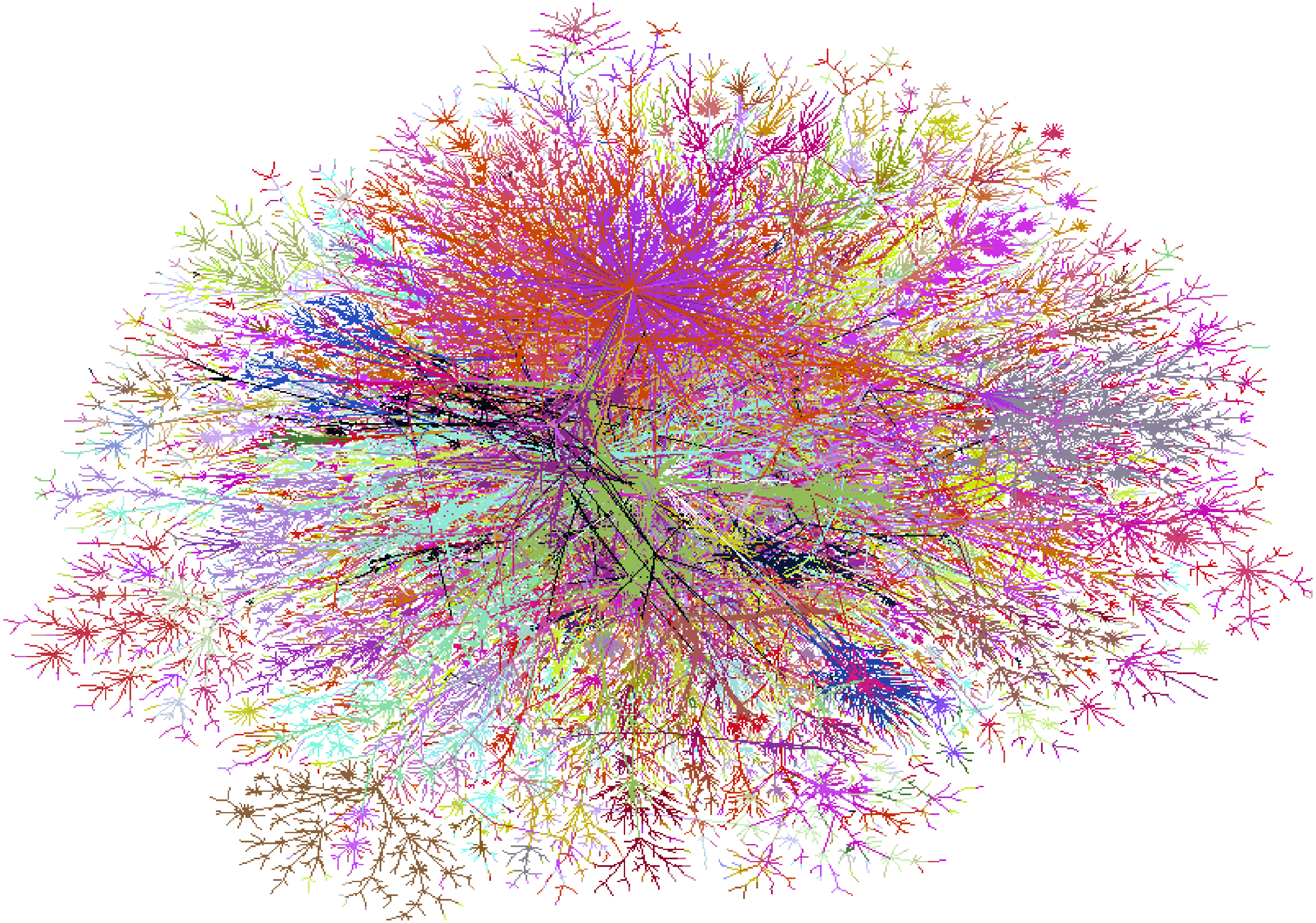
Network Models: Scale-Free

- A scale-free network is network whose degree distribution follows a power law

$$p_k = Ck^{-\alpha}$$

- The internet and the World Wide Web are examples of power laws
  - i.e. only a handful of pages with many links, but millions of pages with only a handful of links
- The structure of these networks are similar at all scales
  - There are parallels with fractals here!

Network Models: Scale-Free



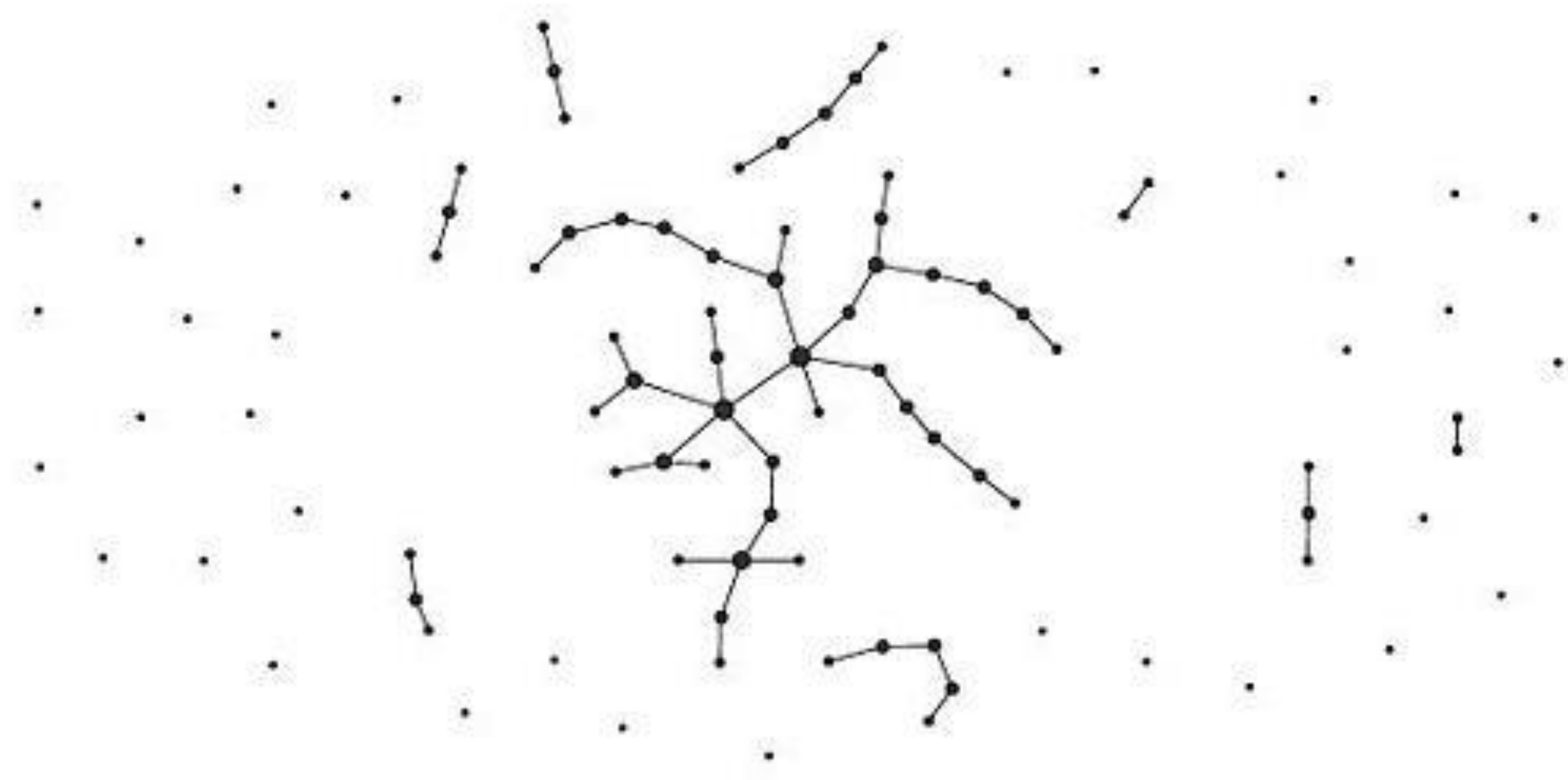
Map of the internet circa 1998, colored by IP address (W.R. Cheswick)

Network Models: Random Graph

- Known as the Erdös-Rényi, Poisson, or Binomial random graph
- Typically denoted $G(n,p)$ for its two parameters
  - n = number of nodes in the network
  - p = probability that a link exists between two nodes
- Mathematically very simple – almost everything about its structure can be calculated analytically
- Not often representative when compared to real-world networks

Network Models: Random Graph

- Realization of a *G*(*100,0.01*) random graph



http://en.wikipedia.org/wiki/File:Erdos_generated_network-p0.01.jpg

Some Concepts: Degree

- The degree, *k*, a node is the number of links terminating or originating at it

$$k_i = \sum_{j=1}^{n} A_{ij}$$

- In weighted networks, this is termed *strength*
- The mean degree, <*k*>, of a node in a network

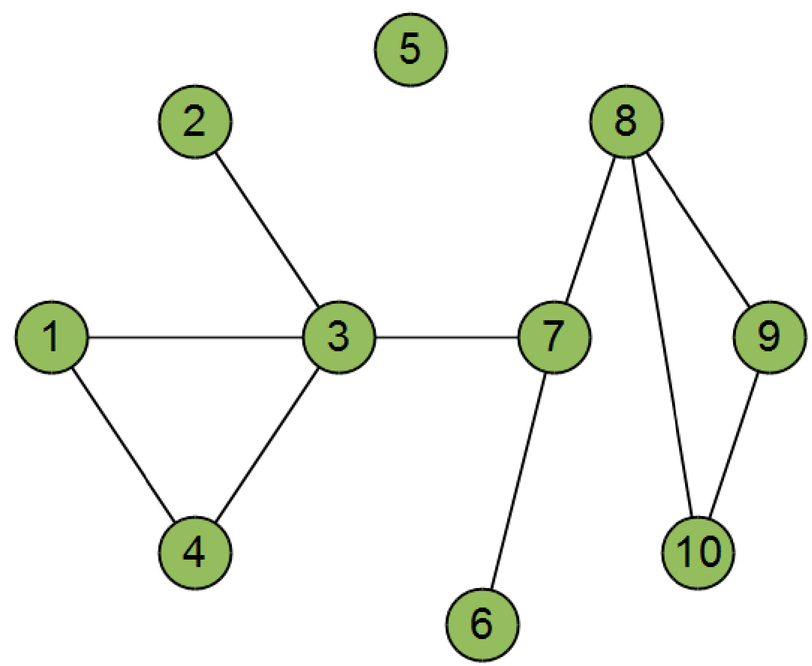
$$\langle k \rangle = \frac{1}{n}\sum_{i=1}^{n} k_i = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}$$

- The sum of all degrees in a network must be equal to twice the number of links, *m*. **Why?**

$$2m = \sum_{i=1}^{n} k_i = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}$$

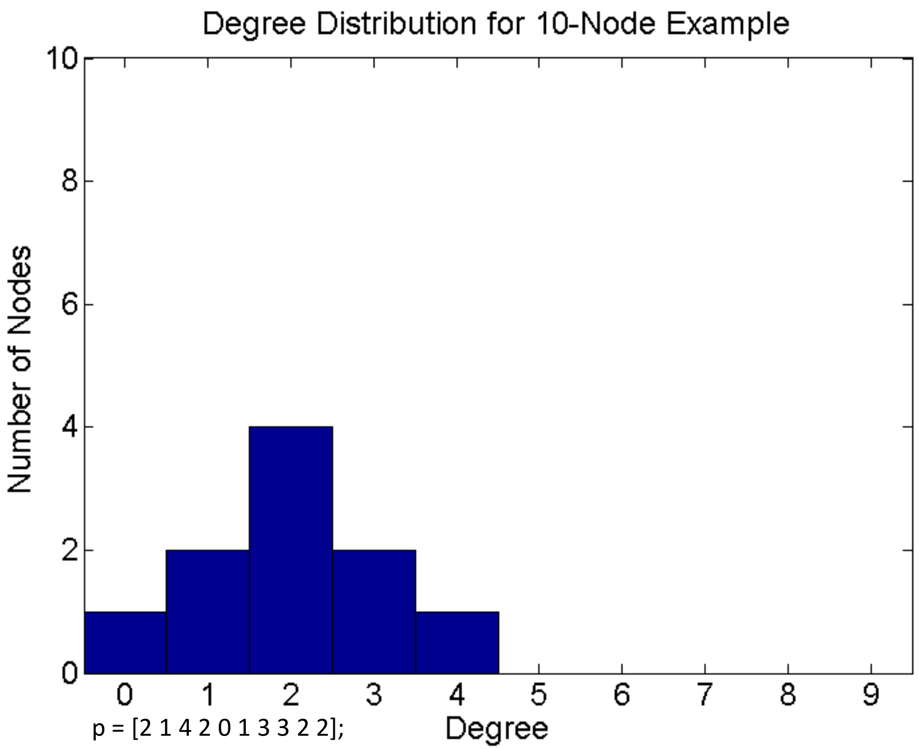
Concepts: Degree Distribution

- A defining characteristic of network structure
- Define p_k to be fraction of nodes in a network that have degree, k
- In this example: $p_0 = 1/10$, $p_1 = 2/10$, $p_2 = 4/10$, $p_3 = 2/10$, $p_4 = 1/10$
- **Can you see why?**
- Can also be thought of as the probability that a chosen node has degree, k

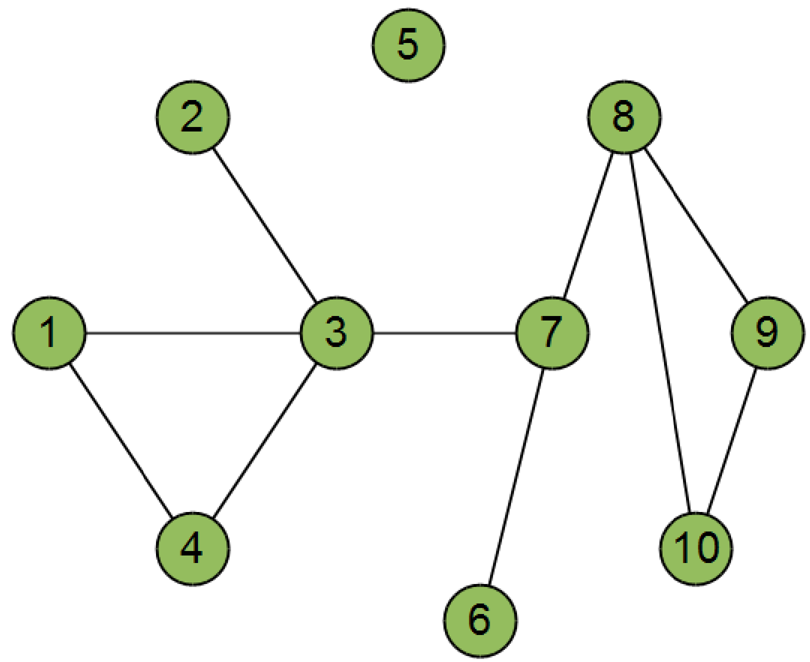


Concepts: Degree Distribution

- Extending the probability concept of degree distribution, it can be plotted as a histogram

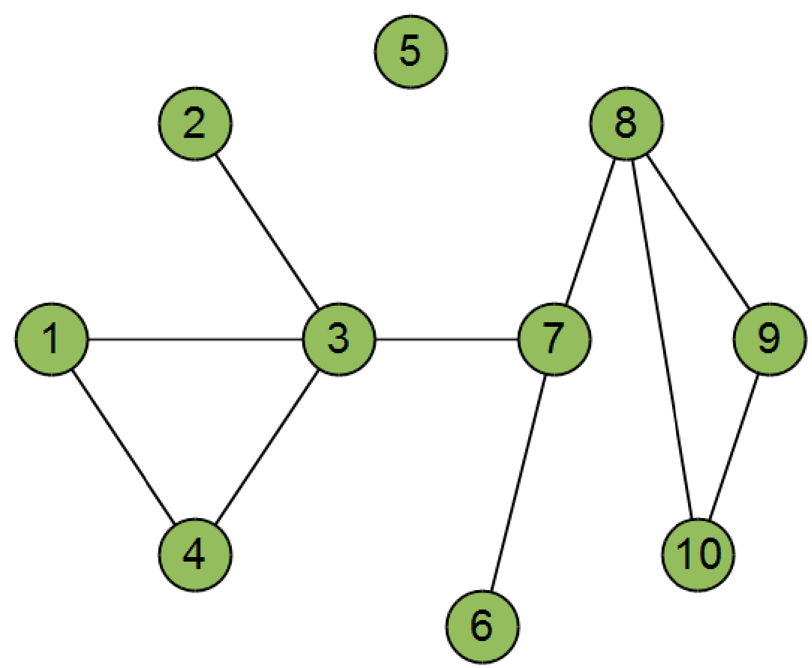



```
p = [2 1 4 2 0 1 3 3 2 2];
links = 0:9;
hist(p,links)
set(gca, 'FontSize', 18, 'XLim',[-0.5 9.5], 'YLim', [0 10]);
xlabel('Degree', 'FontSize', 18)
ylabel('Number of Nodes', 'FontSize', 18)
title('Degree Distribution for 10-Node Example', 'FontSize', 18)
```



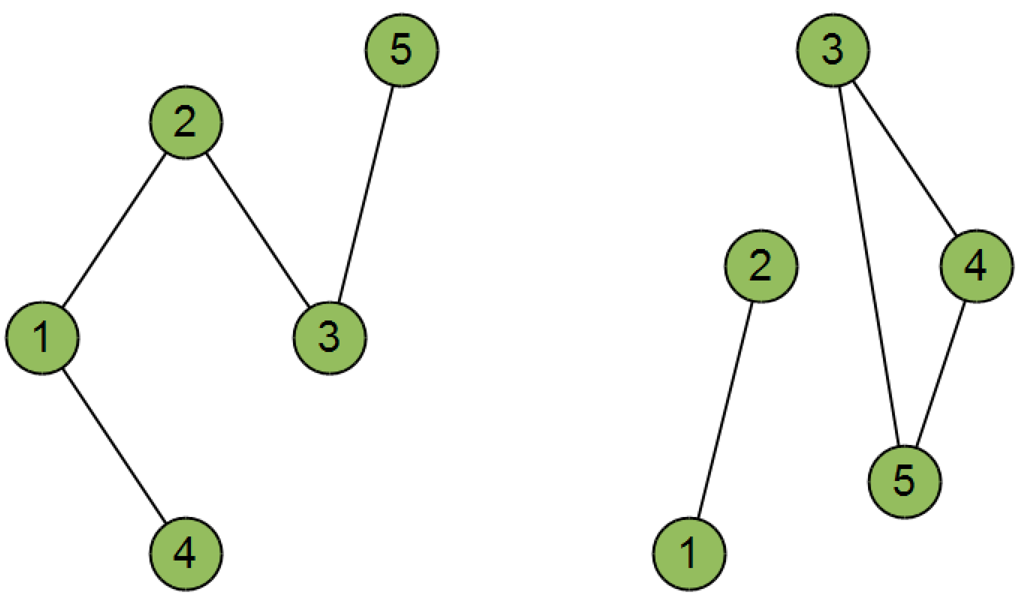
Concepts: Degree Sequence

- Similar information to degree distribution
- The set of degrees, $\{k_1, k_1, \ldots, k_n\}$, for all nodes
- In this example: {2, 1, 4, 2, 0, 1, 3, 3, 2, 2}
- **Can you see why?**



Degree Distribution and Sequence

- Neither gives a complete network description
- Many realizations for a distribution or sequence
- An *ensemble* represents all possible realizations
- In this example: $p_1 = 2/5$, $p_2 = 3/5$



Two realizations of the ensemble for the degree distribution [p1 = 2/5, p3 = 3/5]

Back to Random Graphs

- In a simple, three-node network, suppose that link forms between two nodes with probability, p_l
- Probability of three links

$$P(3) = p_l^3$$

- Probability of two links

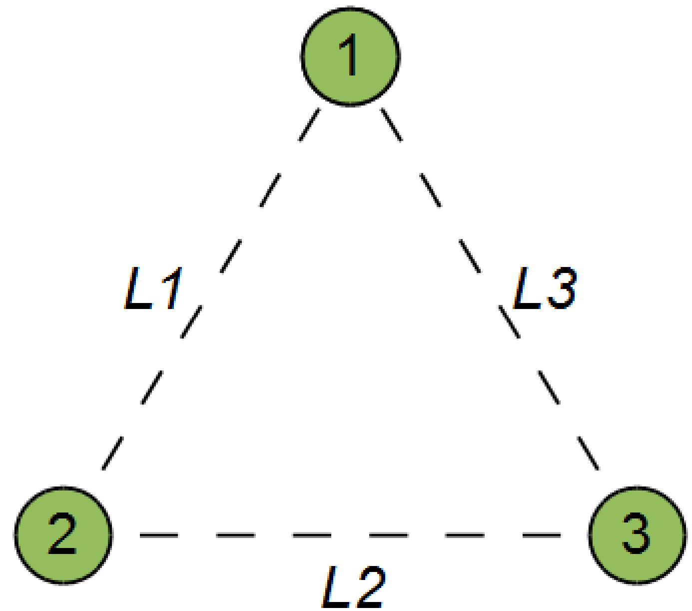
$$P(2) = 3p_l^2(1 - p_l)$$

- Probability of one link

$$P(1) = 3p_l(1 - p_l)^2$$

- Probability of zeros links

$$P(0) = (1 - p_l)^3$$



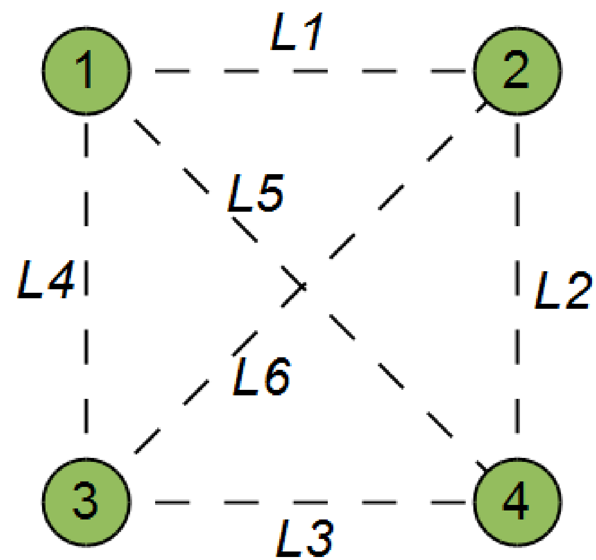
- What is the probability of *m* links on *n* nodes? What is the probability of a node with degree, *k*?

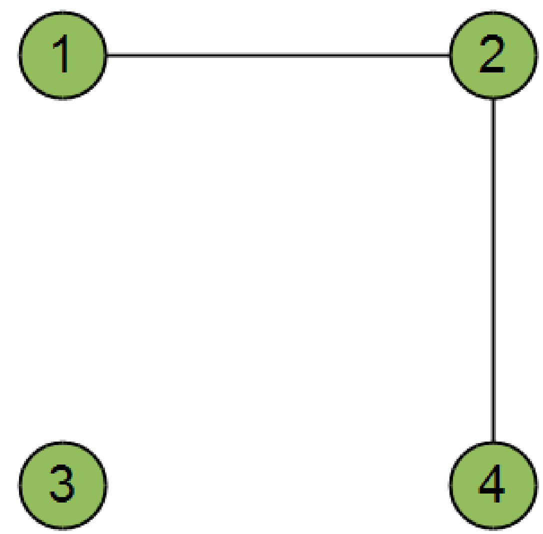
Four-Node Random Graph

- There are six possible links in an undirected four-node network
- What is the probability that two links are present, but four are not?
  – e.g. Link 1 and 2 are present

$$P(L1, L2) = p_l^2 (1 - p_l)^4$$

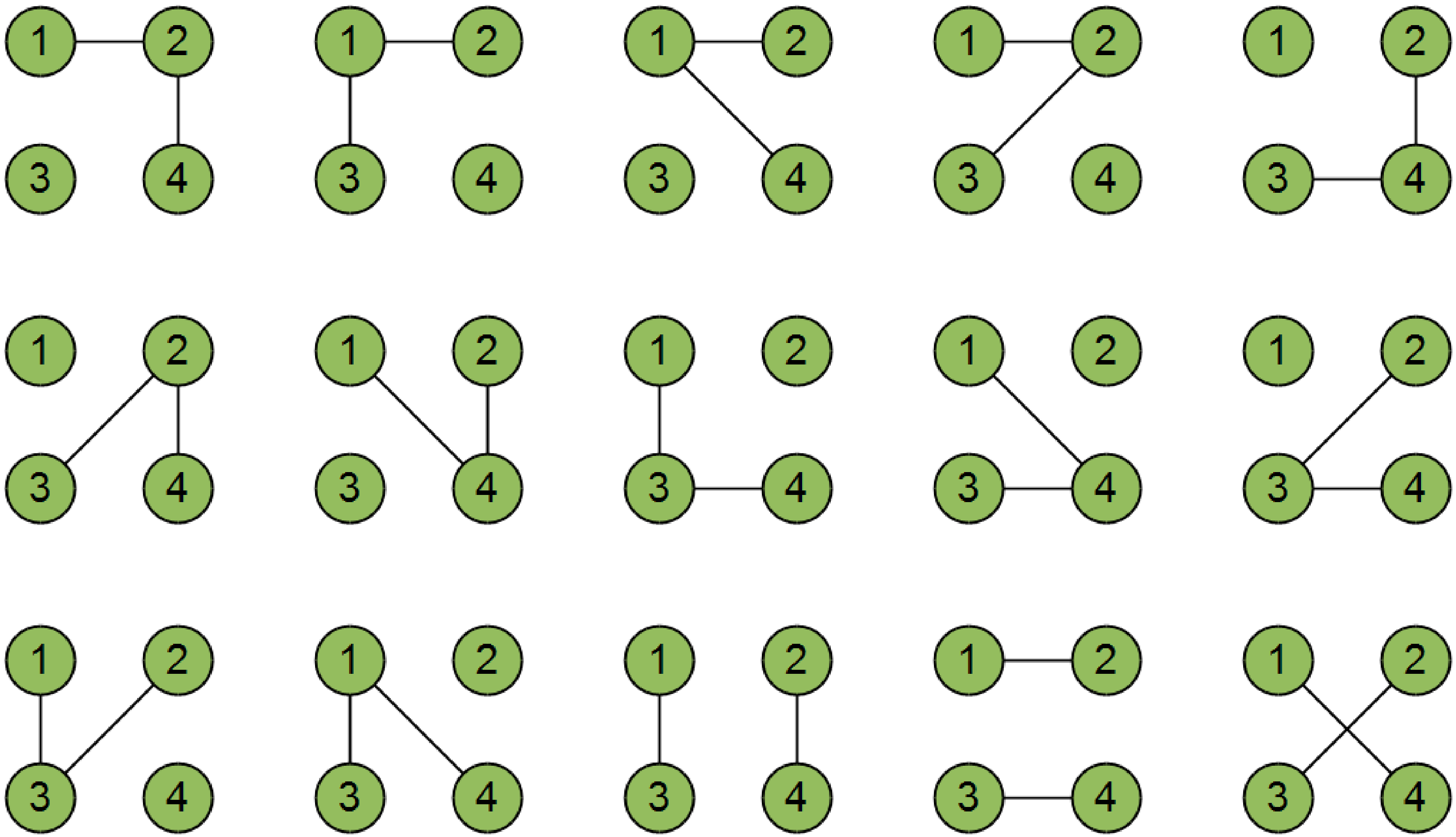
- **What is the probability of ANY two links existing in this four-node network?**





Four-Node Random Graph

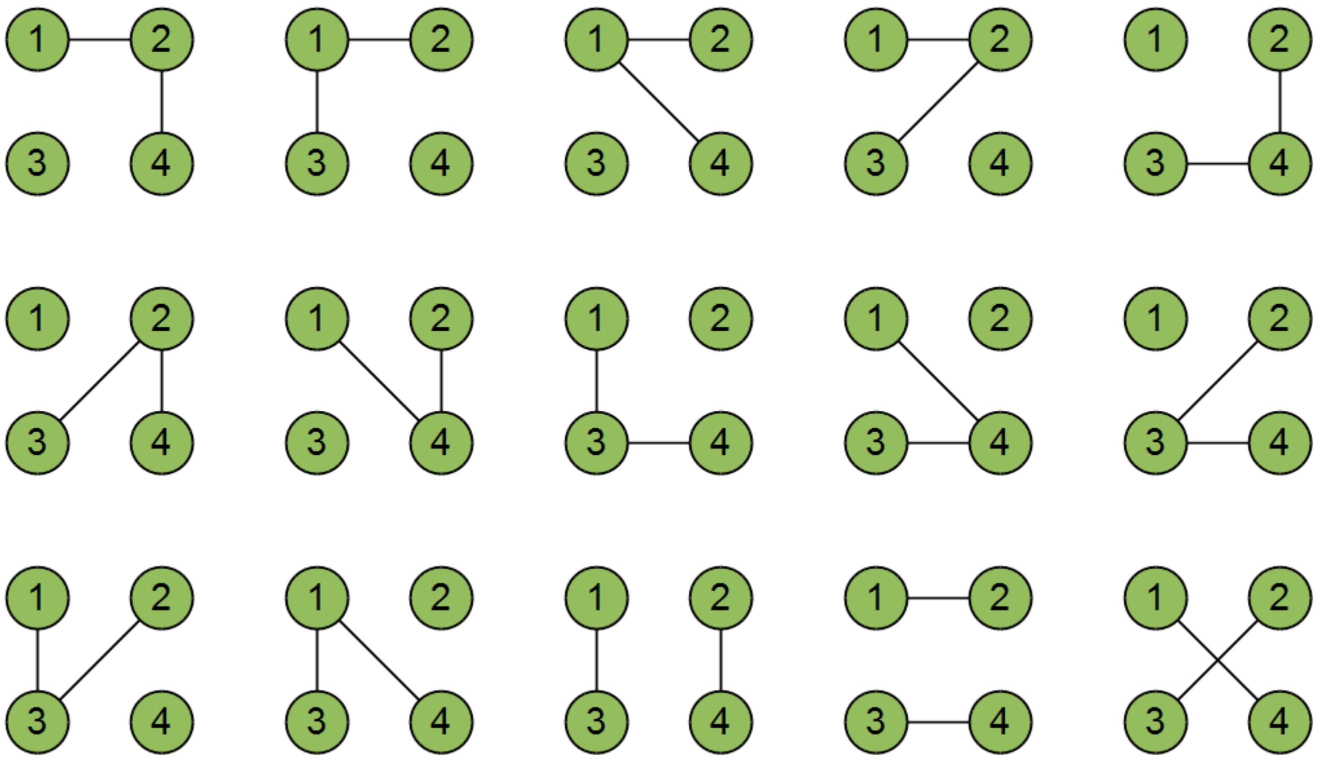
- Any two links can be formed in 15 ways, below



$$P(2) = 15\, p_l^2 \left(1 - p_l\right)^4$$

Four-Node Random Graph

- The first link can be chosen in ${}_4C_2$ ways
- The second link can be chosen in $({}_4C_2 - 1)$ ways
- Divide by two to remove duplicates: 6 x 5 / 2 = 15



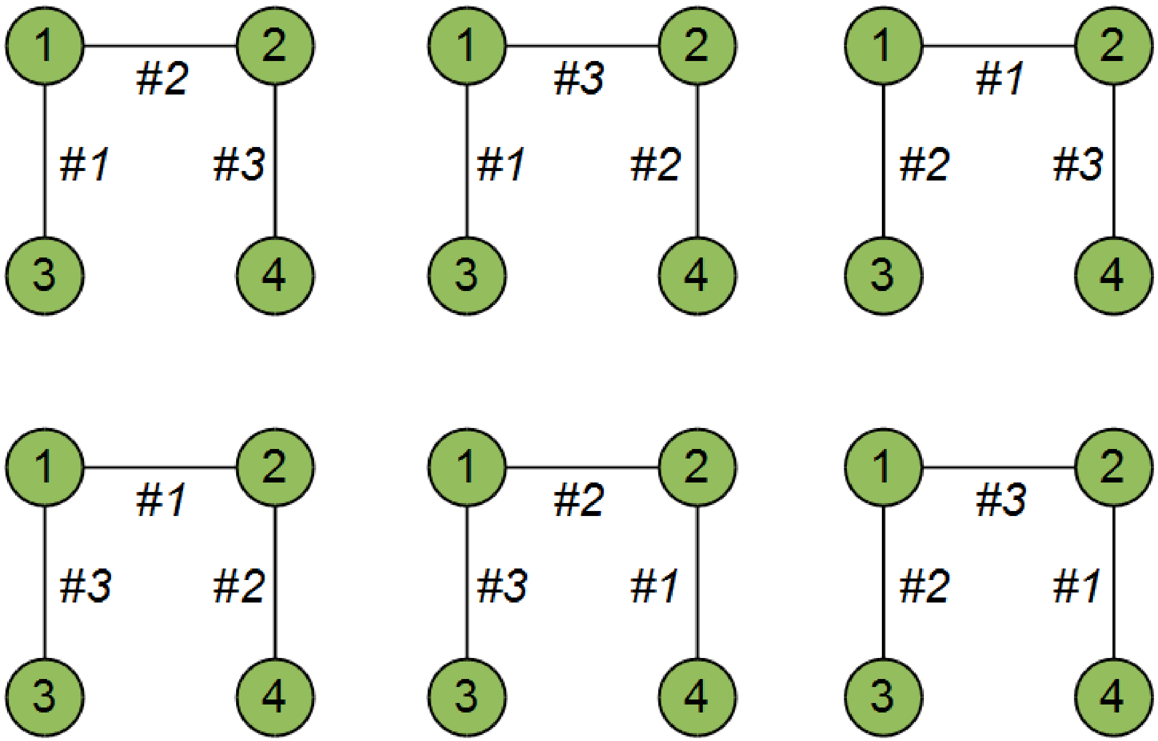
Generalize to N-Node Graphs

- The number of possible links in an n-node graph is $t = {}_{n}C_{2}$
- Suppose we want m links in our n-node random network
  - The first link can be chosen in t ways
  - The second link can be chosen in $(t - 1)$ ways
  - The m^{th} link can be chosen in $(t - m + 1)$ ways
  - Divide by m! to remove duplicate choices

$$\frac{(t)(t-1)\ldots(t-m+1)}{m!} = {}_{t}C_{m} = \binom{t}{m}$$

What Are Duplicate Choices?

- For example, in our four-node network, the SAME three links could be chosen in six ways
- The number of duplicates for *m* links is *m*!



Generalize to N-Node Graphs

- Putting this together, the total probability of drawing a graph with *m* links from $G(n,p_l)$, in which all p_l are independent and equal is:

$$\Pr(m) = \binom{t}{m} p_l^m \left(1 - p_l\right)^{t-m}$$

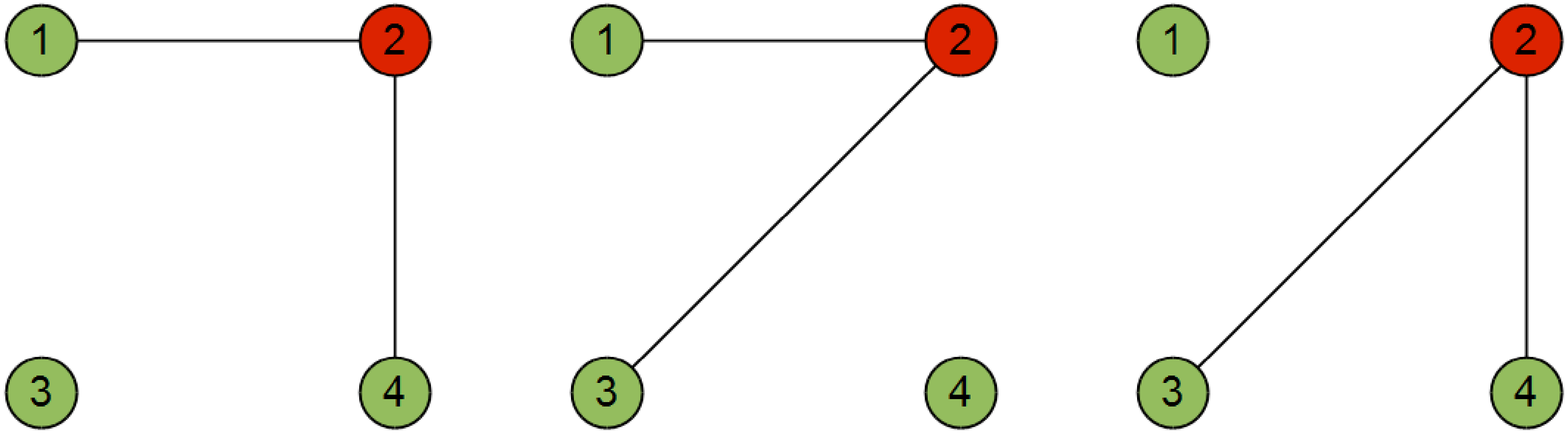
- NOTE: The number of possible links in an n-node graph is $t = {}_nC_2$

Four-Node Random Graph

- It is also possible to infer the probability that a node has a certain degree, *k*
- For a node in this network, there are three ways it can have a degree of two

$$P_2(2) = 3p_l^2(1 - p_l)$$

- Choice of nodes to link with is limited to $n - 1$



Generalize to N-Node Graphs

- What is the probability that a node, *i*, in an n-node random network has *k* links?
- The number of possible links to *i* is n – 1
- For any set of *k* links to *i*
  - The first link can be chosen in (n – 1) ways
  - The second link can be chosen in (n – 2) ways
  - The k^{th} link can be chosen in (n – k) ways
  - Divide by k! to remove duplicate choices

$$\frac{(n-1)(n-2)\ldots(n-k)}{k!} = {}_{n-1}C_k = \binom{n-1}{k}$$

Generalize to N-Node Graphs

- Putting this together, the probability of a node having degree *k* in a random graph, $G(n,p_l)$, in which all p_l are independent and equal is:

$$\Pr(k) = \binom{n-1}{k} p_l^k \left(1 - p_l\right)^{n-1-k}$$

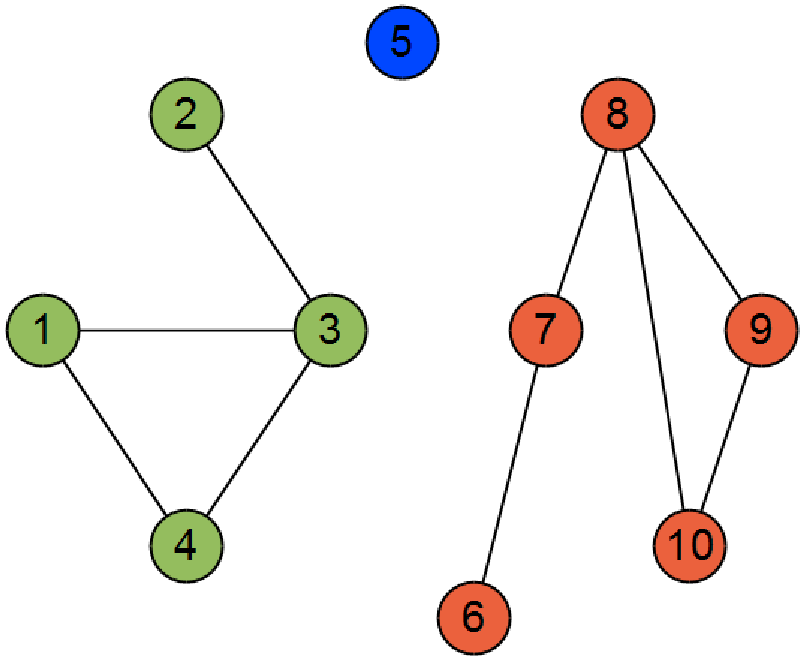
- As n becomes very large, and p is small, this is approximately equal to a Poisson distribution

$$\Pr(k) = e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!} = e^{-p(n-1)} \frac{\left(p(n-1)\right)^k}{k!}$$

- NOTE: In a random network, $\langle k \rangle = p(n-1)$

Components

- If path between all pairs of nodes does not exist, a network is said to be *disconnected*
- *Components* are sub-groups of connected nodes
- If the entire network is connected, there is only one component



Three components in a disconnected network, colored by their component number

Finding Components

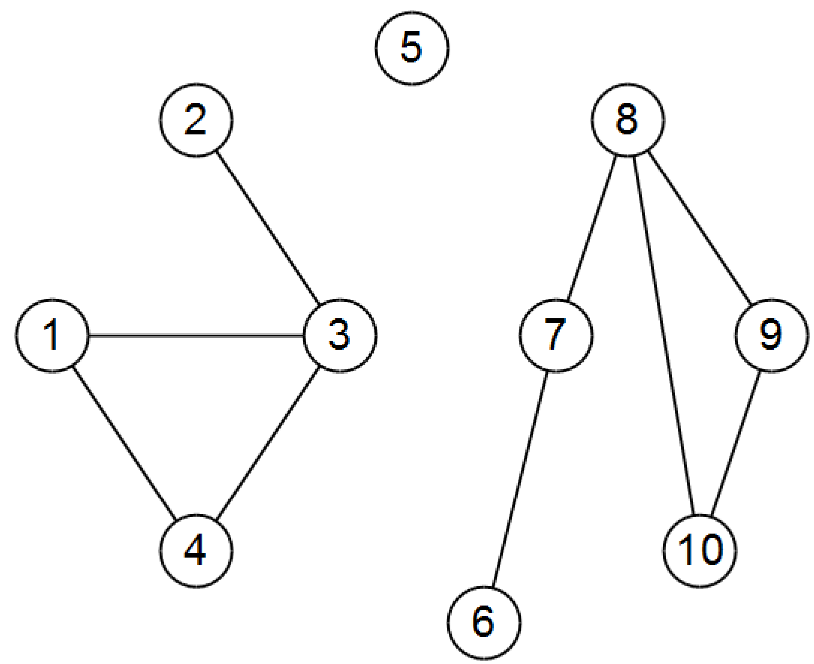
- The Breadth-First Search algorithm can be modified to classify nodes into components
- Run BFS from *every* node in the network
- BFS will reach *all* nodes that start node is connected to
- Label each node in a single BFS run as being in the same component
- Once defined we can run statistics on components e.g. number, size, size distribution

Finding Components: Pseudocode

```
for i = 1:n
    component(i) = -1; % flag each node as unassigned
end

comp = 0; % initialize the component number counter

for i = 1:n
    if component(i) < 0 % check that this node not already
                        % assigned to a component
        comp = comp + 1; % increment the component counter

        % Run BFS from node i and mark every encountered
        % node as a member of the component, comp

    end
end
```

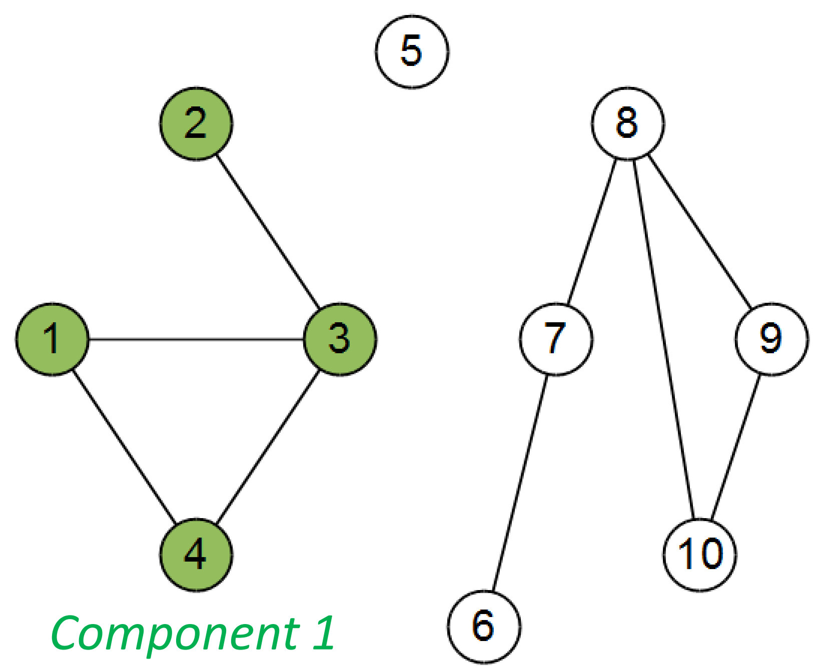
Finding Components: An Example

- BFS from node 1



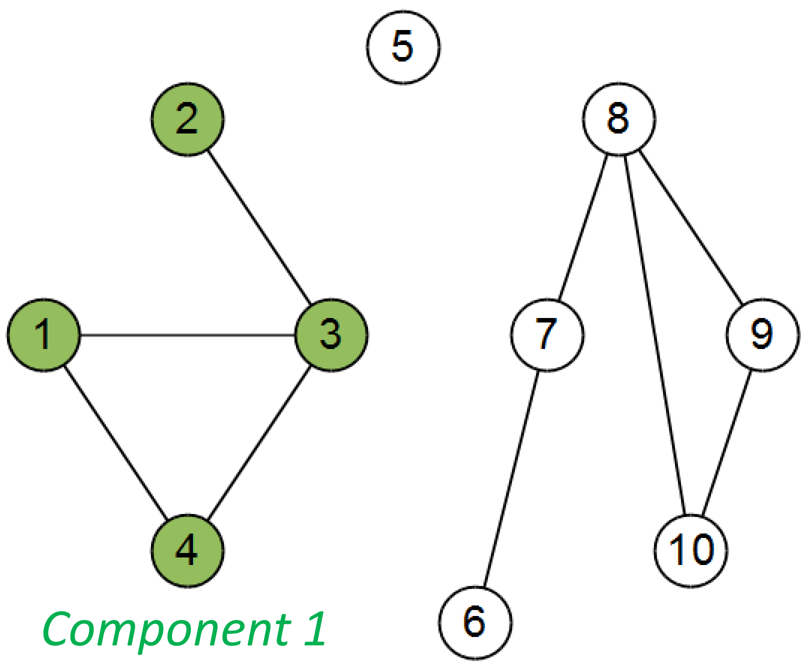
Finding Components: An Example

- BFS from node 1
  - Define component 1



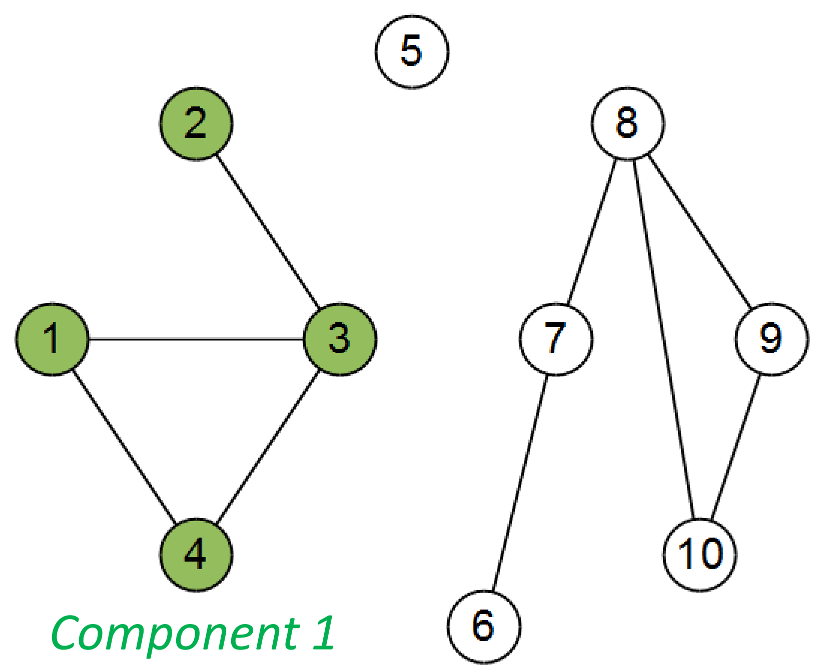
Finding Components: An Example

- BFS from node 1
  - Define component 1
- BFS from node 2, 3, 4
  - Already defined



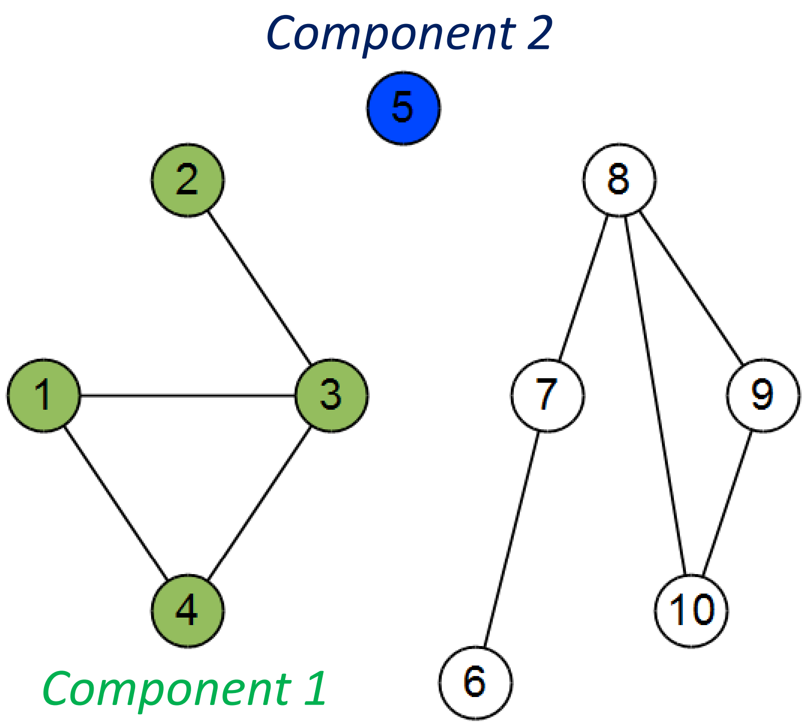
Finding Components: An Example

- BFS from node 1
  - Define component 1
- BFS from node 2, 3, 4
  - Already defined
- BFS from node 5



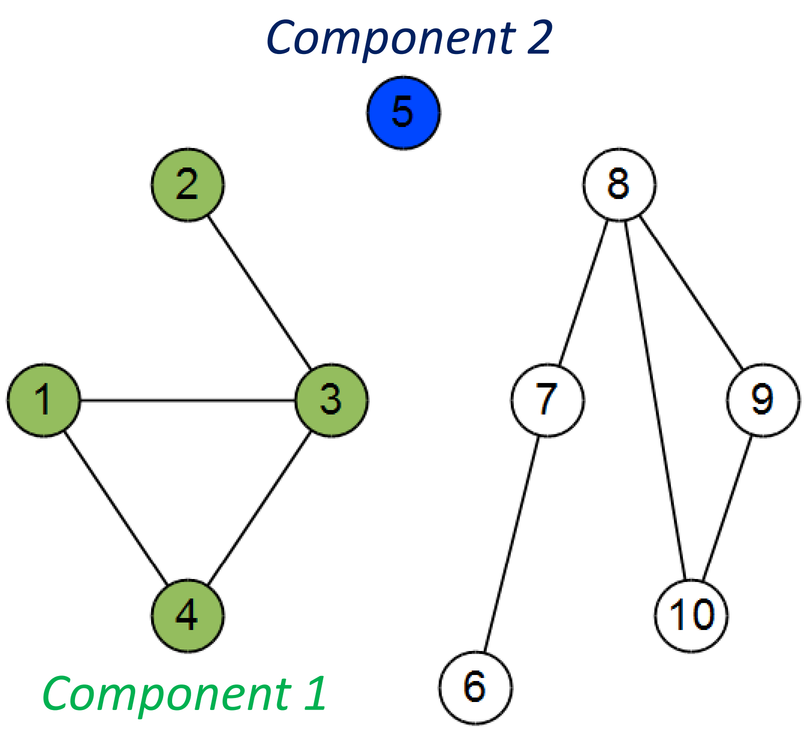
Finding Components: An Example

- BFS from node 1
  - Define component 1
- BFS from node 2, 3, 4
  - Already defined
- BFS from node 5
  - Define component 2



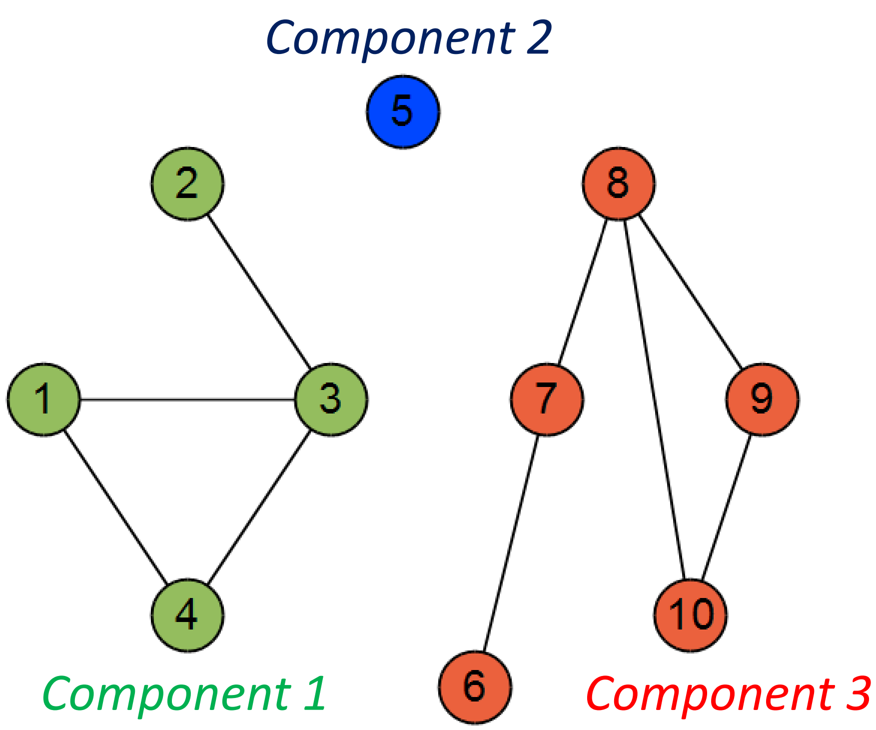
Finding Components: An Example

- BFS from node 1
  - Define component 1
- BFS from node 2, 3, 4
  - Already defined
- BFS from node 5
  - Define component 2
- BFS from node 6



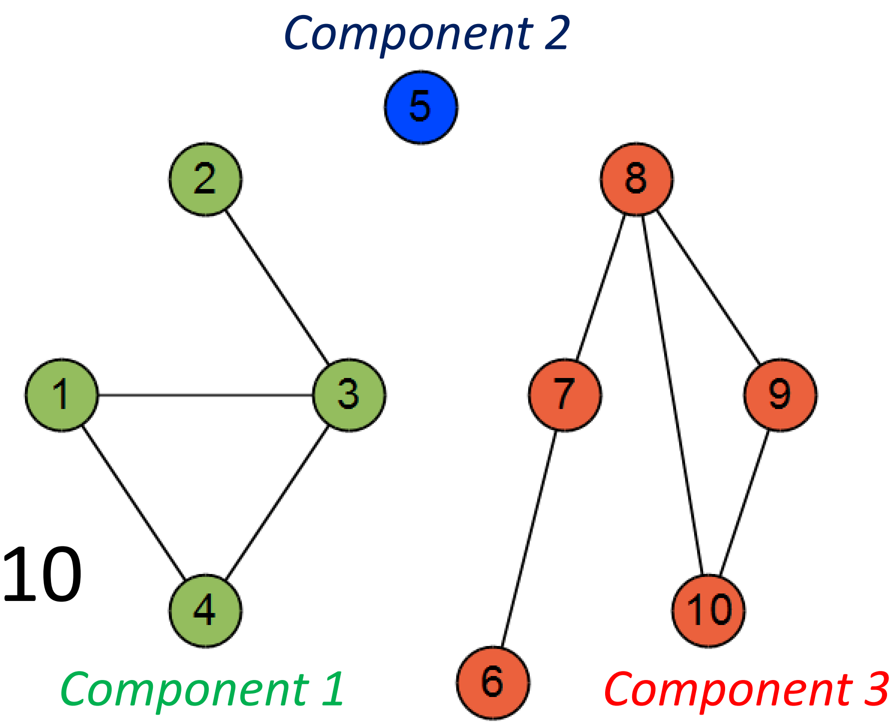
Finding Components: An Example

- BFS from node 1
  - Define component 1
- BFS from node 2, 3, 4
  - Already defined
- BFS from node 5
  - Define component 2
- BFS from node 6
  - Define component 3



Finding Components: An Example

- BFS from node 1
  - Define component 1
- BFS from node 2, 3, 4
  - Already defined
- BFS from node 5
  - Define component 2
- BFS from node 6
  - Define component 3
- BFS from node 7, 8, 9, 10



Finding Components: An Example

- BFS from node 1
  - Define component 1
- BFS from node 2, 3, 4
  - Already defined
- BFS from node 5
  - Define component 2
- BFS from node 6
  - Define component 3
- BFS from node 7, 8, 9, 10
  - Already defined