

# ST5225: Statistical Analysis of Networks

## Lecture 1: Introduction

WANG Wanjie  
staww@nus.edu.sg

Department of Statistics and Applied Probability  
National University of Singapore (NUS)

Monday 15<sup>th</sup> January, 2018

- Module information
- Introduction of Networks

## Aim and Objective

Network data has become increasingly important in both academia and industry. Many interesting questions can be understood and analysed through networks. *Applications are found in areas such as sociology (Facebook and Twitter networks), computer science (World Wide Web), and biology (gene and protein interaction networks).* With the availability of large network data sets, be it in corporate, governmental or scientific contexts, comes the necessity to work with such data in an appropriate matter. **This course gives a practical introduction to the theory of network analysis.**

## Prerequisite

ST2131/ST5201, or other courses at the same level  
Knowledge about basic concepts of statistics

## Lecture/Tutorial Time

12 meetings on 1pm-4pm, Saturdays (no class at Spring Festival Holiday)  
Tutorials are after lectures

## Office Hour

2pm-4pm, Thursdays

## Module webpage

IVLE (<https://ivle.nus.edu.sg/>) for lecture notes, problem sheets, solutions, **webcasts**

## Programming Language

R (Free download at <https://cran.r-project.org/>)  
Recommend "Rstudio" (<https://www.rstudio.com/>, R installation required).

## Textbook & Supplementary Textbooks

- *Statistical Analysis of Network Data* by Eric D. Kolaczyk
- *Networks, Crowds, and Markets* by David Easley and Jon Kleinberg
- *Statistical Analysis of Network Data with R* by Eric D. Kolaczyk and Gabor Csardi
- *Networks: An Introduction* by M. E. J. Newman

## Coverage

Topics include statistical basic concepts of networks, network sampling, network models, block model and stochastic block model, descriptive and inferential network analysis, network visualisation, and some small networks.

## Assessment

- Assignments: 40%
  - ~ 3 Assignments
  - Assignment submission in class
- Final examination: 60%

## Always keep in mind

- No late Assignments
- No make-up final exam
- For emergencies, please contact me BEFORE deadline for extension

## Advice

- Check IVLE frequently for lecture notes, assignment sheets, solutions, and codes
- Attend most, if not all, lectures & tutorials
- Seriously read through & understand the corresponding materials (especially the examples) in the textbook after lectures
- Attempt homework problem, & other problems from the textbook

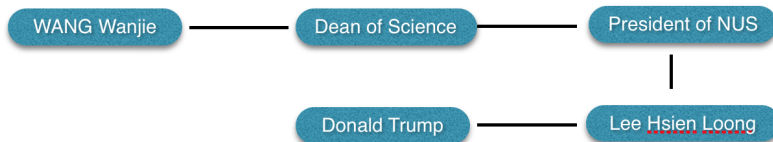
## *Statistical Analysis of Network Data, Chapter 1-2*

- History of Networks
- Examples of Networks
- Basic Concepts of Networks



## Small World Problem (1967)

Any two persons in the world can be connected by at most **6** other people.

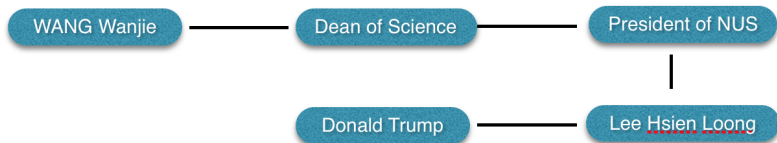


- Well-known as "six degrees of separation"
- Come to my office hour often, and you will be only 5 persons away from Trump!
- Even before the development of internet, such as snapchat, chat-rooms, forums.

*Our world is a "connected" world*

## Small World Problem (1967)

Any two persons in the world can be connected by at most **6** other people.

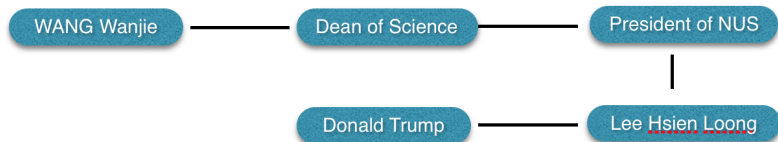


- Well-known as "six degrees of separation"
- Come to my office hour often, and you will be only 5 persons away from Trump!
- Even before the development of internet, such as snapchat, chat-rooms, forums.

*Our world is a "connected" world*

## Small World Problem (1967)

Any two persons in the world can be connected by at most **6** other people.



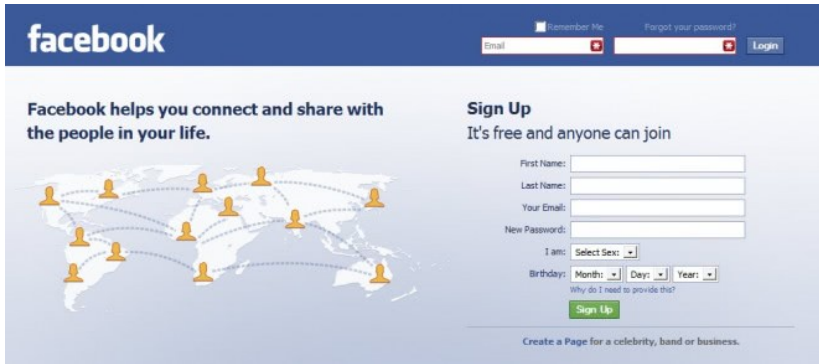
- Well-known as "six degrees of separation"
- Come to my office hour often, and you will be only 5 persons away from Trump!
- Even before the development of internet, such as snapchat, chat-rooms, forums.

*Our world is a "connected" world*

- Small world problem was in 1967
- Studied by relatively small group of researchers from then
- Recently, appeared in many applications due to multiple factors
  - Globalization, the Internet, etc.
  - Social network, biological network, etc.
- Also some new challenges:
  - Flood of high-throughput data;
- epidemic-like spread of interest in recent  $\sim 15$  years

- First introduced in 2016
- This is the second time we offer this module
- Module "Stat. Analysis of Networks" is introduced in more and more universities, mostly for grad. students

Sign up Page of Facebook:



The image shows the Facebook sign-up page. At the top, the Facebook logo is on the left, and login options (Remember Me, Forgot your password?, and a Login button) are on the right. Below the logo, the text "Facebook helps you connect and share with the people in your life." is displayed next to a world map with orange person icons connected by dashed lines. To the right of the map is the "Sign Up" section, which includes the text "It's free and anyone can join" and a series of input fields: First Name, Last Name, Your Email, and New Password. Below these are dropdown menus for "I am:" (with a "Select Sex:" label), "Birthdays:" (with "Month:", "Day:", and "Year:" labels), and a "Why do I need to provide this?" link. A green "Sign Up" button is at the bottom of the form. At the very bottom, there is a link that says "Create a Page for a celebrity, band or business."

facebook

☐ Remember Me [Forgot your password?](#)

Email

Facebook helps you connect and share with the people in your life.

**Sign Up**  
It's free and anyone can join

First Name:

Last Name:

Your Email:

New Password:

I am:

Birthdays:

[Why do I need to provide this?](#)

[Create a Page for a celebrity, band or business.](#)

Connections between People

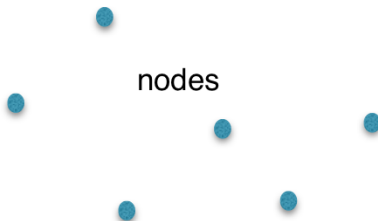
## Singapore Airlines Network:



## Connections between Airports

A network has two basic parts:

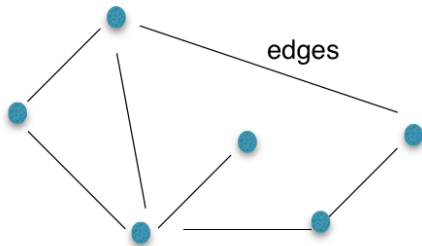
- **nodes:** e.g., people in the facebook network, or airports in the airport network
- **links:** e.g., the friendship connection in the facebook network, or the airlines between airports
- May contain additional info.: e.g., personal info (gender, education, age, etc.) for the facebook network, or the frequency of airlines in the airport network





A network has two basic parts:

- **nodes:** e.g., people in the facebook network, or airports in the airport network
- **links:** e.g., the friendship connection in the facebook network, or the airlines between airports
- May contain additional info. for nodes/edges: e.g., personal info (gender, education, age, etc.) for the facebook network, or the frequency of airlines in the airport network



Mathematical expressions:

- Networks are what we observed in real world. When we care about the mathematical abstractions, we are talking about *graphs*

## Graph

A **graph**  $G = (V, E)$  (as in "graph theory") is a bunch of things (set of **vertices**)  $V$ , plus a set of **edges** among them, i.e., a set  $E \subset V \times V$ , a subset of the ordered pairs of  $V$ .

- **vertices** — nodes
- **edges** — links
- Different terms are preferred in different cases

- Facebook network (social nets)  
 $V = \{\text{the set of all persons}\},$   
 $V \times V = \{\text{all the connections between every pair of people}\},$   
 $E = \{\text{be friended}\}$
- Airport network (technology network)  
 $V = \{\text{the set of all airports}\},$   
 $V \times V = \{\text{all the connections between every pair of airports}\},$   
 $E = \{\text{airlines}\}$
- Bitcoins (business network)  
 $V = \{\text{the set of all users}\},$   
 $V \times V = \{\text{all the connections between every pair of users}\},$   
 $E = \{\text{the transactions}\}$
- Biological networks, informational networks, etc.

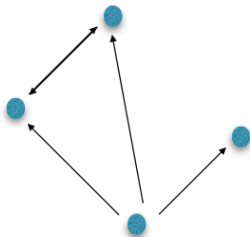
- Facebook network (social nets)  
 $V = \{\text{the set of all persons}\},$   
 $V \times V = \{\text{all the connections between every pair of people}\},$   
 $E = \{\text{be friended}\}$
- Airport network (technology network)  
 $V = \{\text{the set of all airports}\},$   
 $V \times V = \{\text{all the connections between every pair of airports}\},$   
 $E = \{\text{airlines}\}$
- Bitcoins (business network)  
 $V = \{\text{the set of all users}\},$   
 $V \times V = \{\text{all the connections between every pair of users}\},$   
 $E = \{\text{the transactions}\}$
- Biological networks, informational networks, etc.

- Facebook network (social nets)  
 $V = \{\text{the set of all persons}\},$   
 $V \times V = \{\text{all the connections between every pair of people}\},$   
 $E = \{\text{be friended}\}$
- Airport network (technology network)  
 $V = \{\text{the set of all airports}\},$   
 $V \times V = \{\text{all the connections between every pair of airports}\},$   
 $E = \{\text{airlines}\}$
- Bitcoins (business network)  
 $V = \{\text{the set of all users}\},$   
 $V \times V = \{\text{all the connections between every pair of users}\},$   
 $E = \{\text{the transactions}\}$
- Biological networks, informational networks, etc.

- Facebook network (social nets)  
 $V = \{\text{the set of all persons}\},$   
 $V \times V = \{\text{all the connections between every pair of people}\},$   
 $E = \{\text{be friended}\}$
- Airport network (technology network)  
 $V = \{\text{the set of all airports}\},$   
 $V \times V = \{\text{all the connections between every pair of airports}\},$   
 $E = \{\text{airlines}\}$
- Bitcoins (business network)  
 $V = \{\text{the set of all users}\},$   
 $V \times V = \{\text{all the connections between every pair of users}\},$   
 $E = \{\text{the transactions}\}$
- Biological networks, informational networks, etc.

## ■ Directed Graphs.

Each edge has a direction. An edge  $(i, j) \in E$  means an edge directs from node  $i$  to node  $j$ .

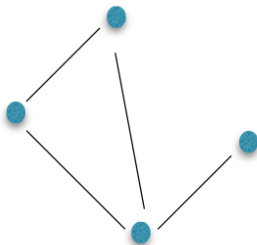


E.g. For the bitcoin network, each transaction has a direction, from the seller to the receiver.

It is possible that an edge has two directions.

## ■ Undirected graphs

If each edge in the graph has two directions ( $(i, j) \in E$  if and only if  $(j, i) \in E$ ), then the relation is symmetric, and we say this graph is being **undirected**.

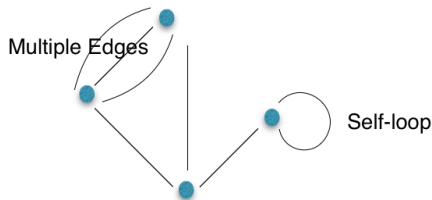


E.g. For the facebook network, the friendship connection is symmetric.

**Note.** Sometimes researchers drop the direction information for easy analysis.



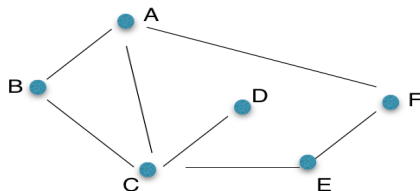
- If a node ties to itself, we call it as *self-loops*.
- If there are multiple edges between the same pair of nodes, we call the graph as a *multi-graph*.



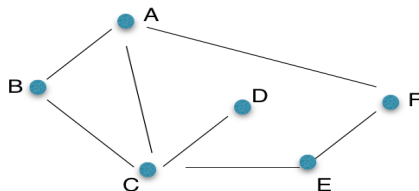
- If a graph has no self-loops or multiple edges, we call it as a **simple graph**.

In this module, we consider *simple graphs* only.

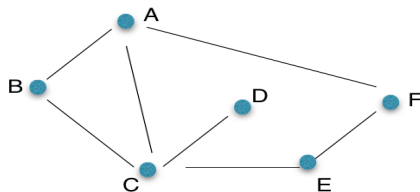
- Network itself is of interest
  - Descriptive statistics
  - network modelling, inferential statistics
- Things happening on a network
  - e.g. Recommendations of a movie/book
  - e.g. Spread of infectious disease



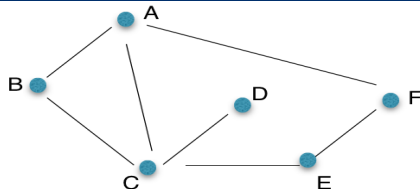
- **Neighbors:** Node  $i$  is called a neighbor of node  $j$ , if  $(i, j) \in E$ 
  - Neighbors of A: Neighbors of E:
- **Degree:** The number of edges incident on a node  $i$  is called the *degree of  $i$* 
  - $d_A = 3, d_E = 2$
  - For directed graphs, we consider the *in-degrees* and *out-degrees* for a node  $i$ , which count the number of edges pointing towards and out from node  $i$ .
  - The *degree* is not a property of a *node*. It is based on the *graph*
- **Degree Sequence:** A vector containing the degrees of each node.



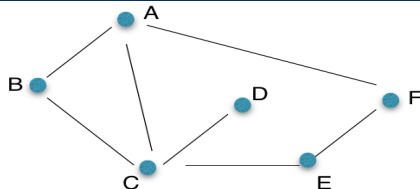
- **Neighbors:** Node  $i$  is called a neighbor of node  $j$ , if  $(i, j) \in E$ 
  - Neighbors of A: Neighbors of E:
- **Degree:** The number of edges incident on a node  $i$  is called the *degree* of  $i$ 
  - $d_A = 3, d_E = 2$
  - For directed graphs, we consider the *in-degrees* and *out-degrees* for a node  $i$ , which count the number of edges pointing towards and out from node  $i$ .
  - The *degree* is not a property of a *node*. It is based on the *graph*
- **Degree Sequence:** A vector containing the degrees of each node.



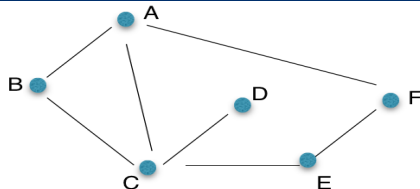
- **Neighbors:** Node  $i$  is called a neighbor of node  $j$ , if  $(i, j) \in E$ 
  - Neighbors of A: Neighbors of E:
- **Degree:** The number of edges incident on a node  $i$  is called the *degree* of  $i$ 
  - $d_A = 3, d_E = 2$
  - For directed graphs, we consider the *in-degrees* and *out-degrees* for a node  $i$ , which count the number of edges pointing towards and out from node  $i$ .
  - The *degree* is not a property of a *node*. It is based on the *graph*
- **Degree Sequence:** A vector containing the degrees of each node.



- **Paths.** A way to get from one node to another along edges
  - Maybe several paths between these two nodes, e.g.,  $A - C$ ,  $A - B - C$ ,  $A - F - E - C$
  - If there is no edge traversed more than once, then the path is called a *simple path*
- **Distance.** The length of the shortest path between two nodes defines the *distance* between them, written as  $d(i, j)$ .
  - $d(A, A) = 0$ ,  $d(A, B) = 1$ ,  $d(A, D) = 2$ ,  $d(A, E) = ?$
- **Sub-graph.** One graph  $H = (U, F)$  is a *subgraph* of  $G = (V, E)$ , if  $U \subset V$  and  $F \subset E$ .
  - Usually, we form subgraphs by picking a subset of the nodes of  $G$  and then including all the edges between them.

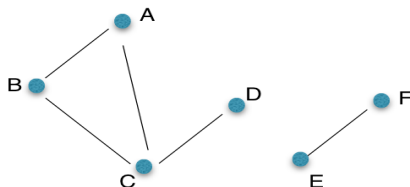


- **Paths.** A way to get from one node to another along edges
  - Maybe several paths between these two nodes, e.g.,  $A - C$ ,  $A - B - C$ ,  $A - F - E - C$
  - If there is no edge traversed more than once, then the path is called a *simple path*
- **Distance.** The length of the shortest path between two nodes defines the *distance* between them, written as  $d(i, j)$ .
  - $d(A, A) = 0$ ,  $d(A, B) = 1$ ,  $d(A, D) = 2$ ,  $d(A, E) = ?$
- **Sub-graph.** One graph  $H = (U, F)$  is a *subgraph* of  $G = (V, E)$ , if  $U \subset V$  and  $F \subset E$ .
  - Usually, we form subgraphs by picking a subset of the nodes of  $G$  and then including all the edges between them.

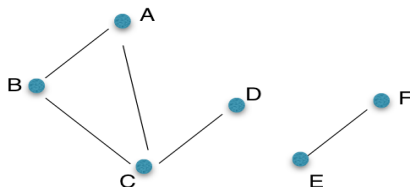


- **Paths.** A way to get from one node to another along edges
  - Maybe several paths between these two nodes, e.g.,  $A - C$ ,  $A - B - C$ ,  $A - F - E - C$
  - If there is no edge traversed more than once, then the path is called a *simple path*
- **Distance.** The length of the shortest path between two nodes defines the *distance* between them, written as  $d(i, j)$ .
  - $d(A, A) = 0$ ,  $d(A, B) = 1$ ,  $d(A, D) = 2$ ,  $d(A, E) = ?$
- **Sub-graph.** One graph  $H = (U, F)$  is a *subgraph* of  $G = (V, E)$ , if  $U \subset V$  and  $F \subset E$ .
  - Usually, we form subgraphs by picking a subset of the nodes of  $G$  and then including all the edges between them.

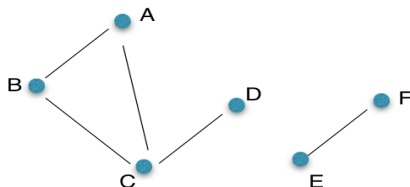




- **Connected**. A graph is called *connected*, if there is a path between any pair of nodes.
  - The graph in previous slide is connected, but in this slide not.
- **Components**. Each network can be decomposed into connected parts, called "*connected components*"
  - 2 components in this graph.
- **Giant Component**. The component that contains most nodes (largest component).
  - Usually, there is only one giant component in one graph.

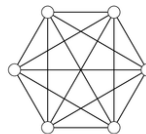
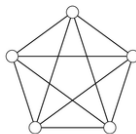
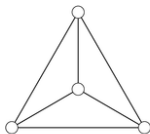


- **Connected**. A graph is called *connected*, if there is a path between any pair of nodes.
  - The graph in previous slide is connected, but in this slide not.
- **Components**. Each network can be decomposed into connected parts, called "*connected components*"
  - 2 components in this graph.
- **Giant Component**. The component that contains most nodes (largest component).
  - Usually, there is only one giant component in one graph.

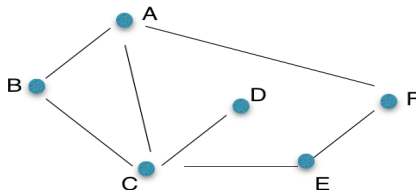


- **Connected**. A graph is called *connected*, if there is a path between any pair of nodes.
  - The graph in previous slide is connected, but in this slide not.
- **Components**. Each network can be decomposed into connected parts, called "*connected components*"
  - 2 components in this graph.
- **Giant Component**. The component that contains most nodes (largest component).
  - Usually, there is only one giant component in one graph.

- **Complete**. A graph where each node is joined to every other node by an edge is called *complete*.

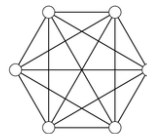
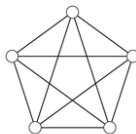
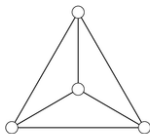


- **Cliques**. A complete subgraph is called a *clique*.

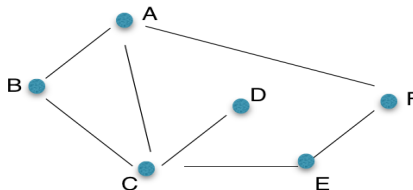


- **Maximal Clique**. A clique with no other cliques containing it.

- **Complete**. A graph where each node is joined to every other node by an edge is called *complete*.



- **Cliques**. A complete subgraph is called a *clique*.



- **Maximal Clique**. A clique with no other cliques containing it.

## Additional Information:

- Additional info. on nodes are usually called the *attributes* of a node
- Additional info. on edges is sometimes record as *edge weights*
  - The length of a path is defined as the sum of the weights along the edges traversed in the path
  - The distance is defined as the length of the shortest path
  - Note that the distance can be a decimal
  - Usually used in a *similarity* graph

- These are descriptive statistics, which help us to catch the properties of a graph
- For mathematical analysis and software, we need mathematical expressions of the whole graph

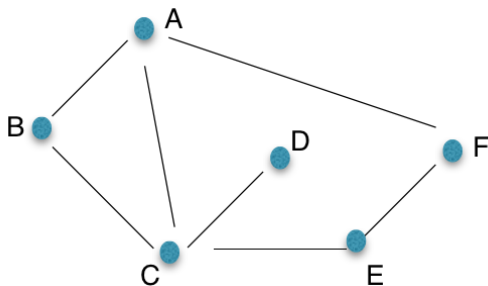
## Adjacency matrix

For a graph  $G = (V, E)$ , with  $|V| = n$ , the **adjacency matrix** associated with  $G$  is defined as an  $n \times n$  matrix  $A$ , where

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

- $A_{ij} = 1$  means there is an edge from node  $i$  to node  $j$
- The neighbors of node  $i$  is the set  $\{j; A_{ij} = 1\}$
- The degree of  $i$  is  $d_i = \sum_{j=1}^n A_{ij}$
- out-degree:  $\sum_{j=1}^n A_{ij}$ ; in-degree:  $\sum_{j=1}^n A_{ji}$
- This is one way we input a graph into software



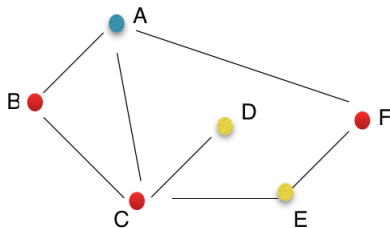


$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

- Aims to 'discover' all the other nodes connected to a given node.
- Two commonly used algorithms, *breadth-first search* and *depth-first search*
- Both with computation complexity  $O(|V| + |E|)$  time
- Useful in different cases.
  - Breadth-first search is useful for shortest path algorithms, such as minimum spanning tree and Dijkstra's algorithm.
  - Depth-first search is usually useful in a larger algorithm, such as algorithms decomposing the graph into connected components.

Remark.  $O(|V| + |E|)$  means there is only a constant coefficient difference between  $|V| + |E|$  and the computation time

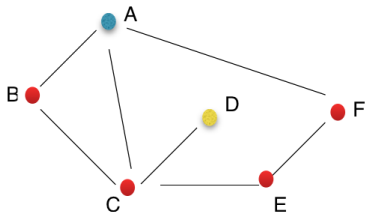
BFS works outward from  $i$ , discovering nodes adjacent to  $i$ , then continuing to nodes adjacent to these nodes, until all reachable nodes are discovered.



Node of interest:  $A$

- First Step:  $A$
- Second Step:  $B, C, F$  (all the nodes connected with  $A$ )
- Third Step:  $D, E$  (nodes connected to  $B, C, D$ )

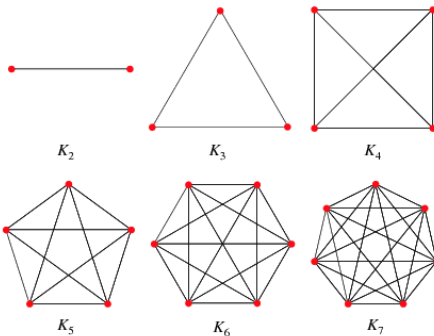
DFS works outward from  $i$  by delving as deeply into  $G$  from the first adjacent node to  $i$ , and then it backtracks to the most recently discovered node  $j$  for the other undiscovered edges.



Node of interest:  $A$

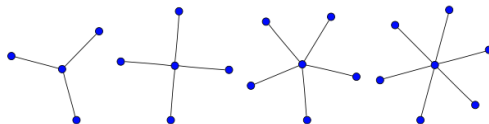
- First Step:  $A$
- Second Step:  $B, C, E, F$  (The left-most path starting from  $A$ )
- Third Step:  $D$  (goes back to  $E$ , and to  $C$ , and then discover  $D$ )

- Some special types of graphs are frequently used as examples, or studied as a characteristic of graphs
- Complete Graphs

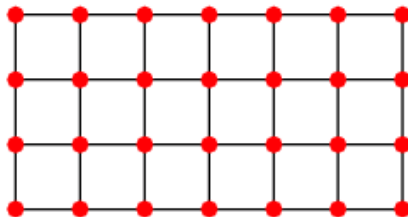


- If a graph contains a larger complete graph, then this graph is more "closely" connected

## ■ Star Graphs



## ■ Lattice Graph



## ■ Trees, wheels, etc....