NANYANG
TECHNOLOGICAL
UNIVERSITY
SINGAPORE

Discrete Mathematics

MH1812

Topic 10.1 - Graph Theory I

Dr. Wang Huaxiong

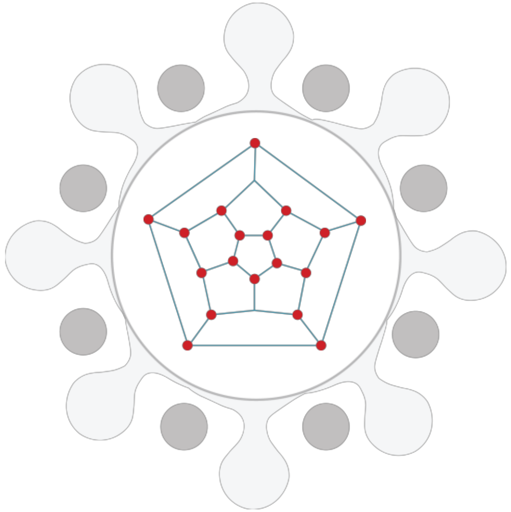
Topic Overview

What’s in store…

Introduction to Graphs

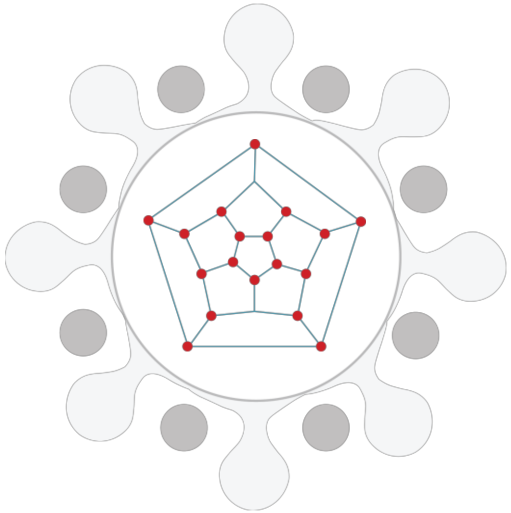
Types of Graphs

Euler Theorem



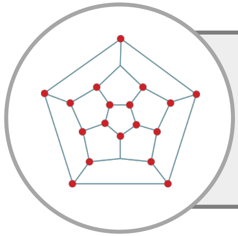
By the end of this lesson, you should be able to…

- Explain what is a graph.
- Explain the difference between the simple graph, multigraph and directed (multi) graph.
- Explain the concepts of the Euler path and circuit.
- Use the Euler theorem in graph theory.



Introduction to Graphs

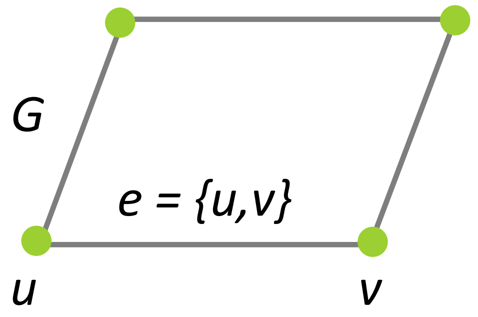
Introduction to Graphs: Definition

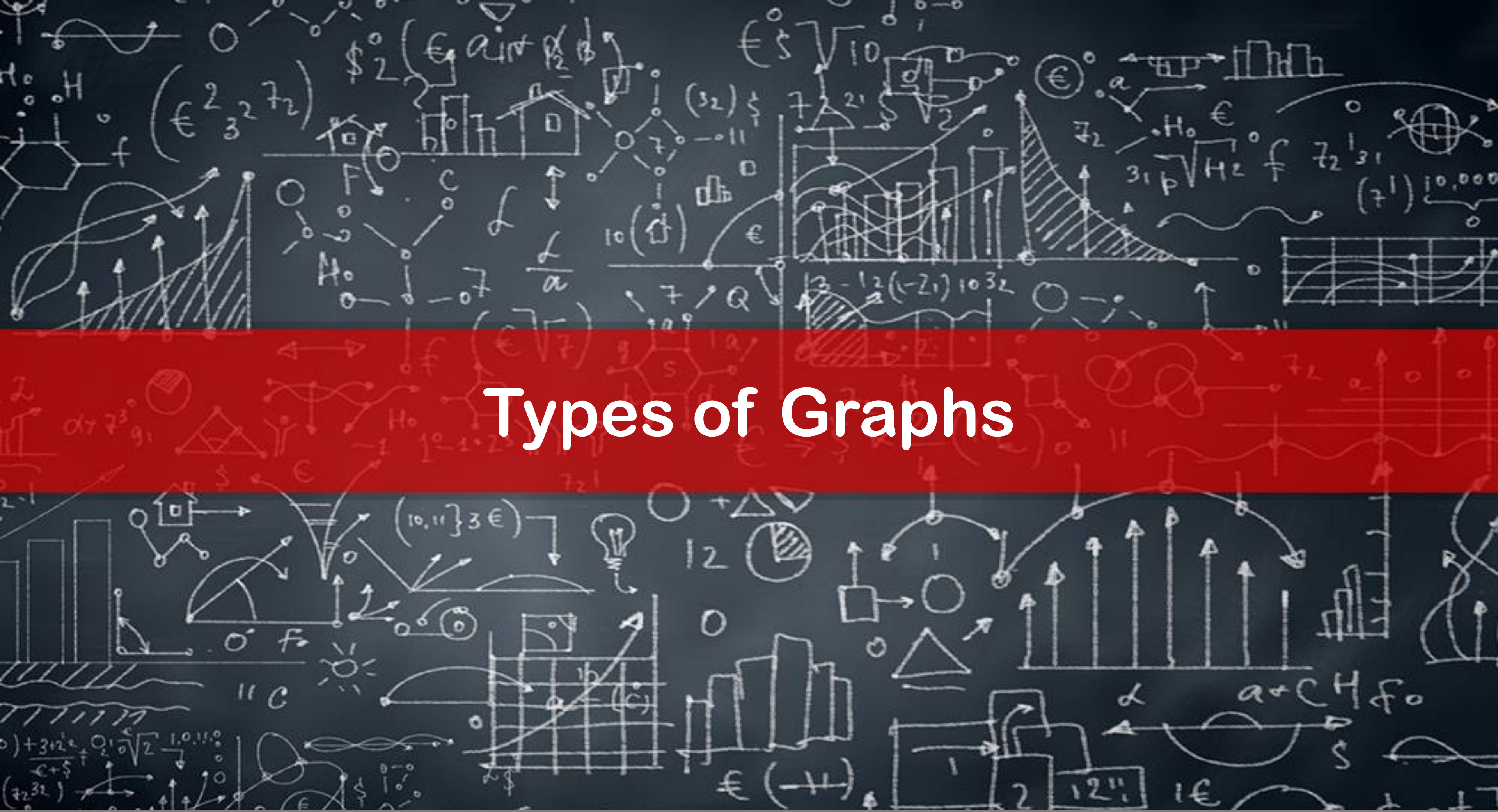


A **graph** $G = (V,E)$ is a structure consisting of a set V of vertices (nodes) and a set E of edges (lines joining vertices).

- Two vertices u and v are **adjacent** in G if $\{u,v\}$ is an edge of G.
- If $e = \{u,v\}$, the edge e is called **incident** with the vertices u and v.

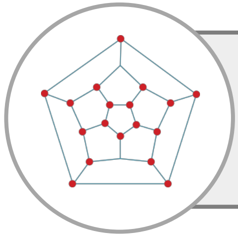
Graphs are useful to represent data.





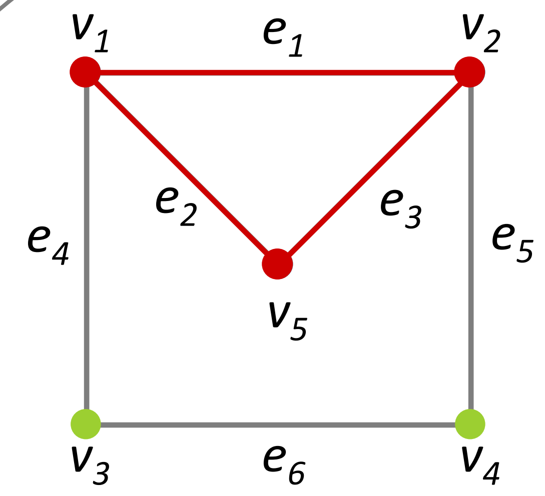
Types of Graphs

Types of Graphs: Subgraphs

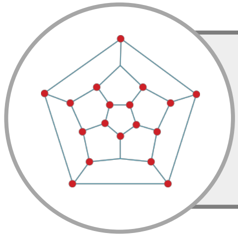


A graph $H = (V_H, E_H)$ is a **subgraph** of $G = (V_G, E_G)$ if V_H is a subset of V_G and E_H is a subset of E_G.

- $V_H = \{v_1, v_2, v_5\}$ is a subset of V_G
- $E_H = \{e_1, e_2, e_3\}$ is a subset of E_G



Types of Graphs: Simple Graphs



A **simple** graph is a graph that has no **loop** (= edge $\{u,v\}$ with $u = v$) and no parallel edges between any pair of vertices.

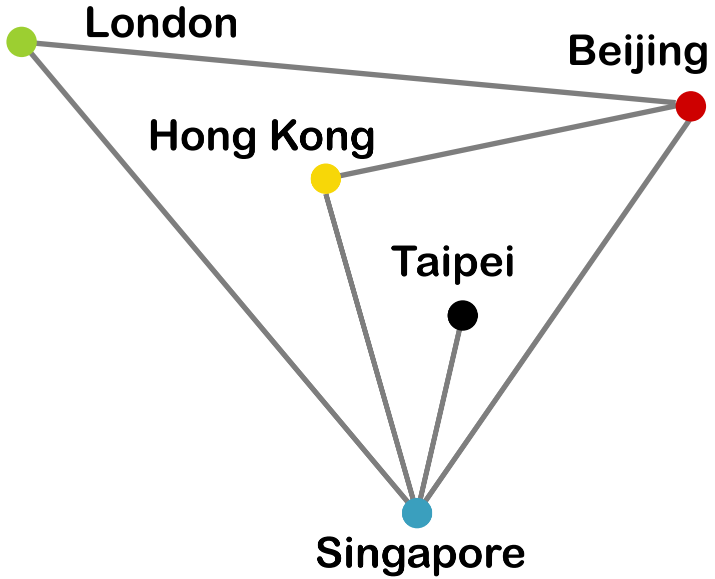
From\To	Hong Kong	Singapore	Beijing	Taipei	London
Hong Kong		4 Flights			
Singapore	2 Flights		3 Flights	1 Flight	1 Flight
Beijing	1 Flight	2 Flights			
Taipei					
London		1 Flight	1 Flight		1 Flight

Draw a graph to see whether there are direct flights between any two cities (in either direction).

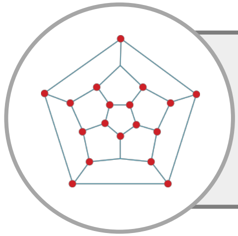
Types of Graphs: Simple Graphs

From\To	Hong Kong	Singapore	Beijing	Taipei	London
Hong Kong		4 Flights			
Singapore	2 Flights		3 Flights	1 Flight	1 Flight
Beijing	1 Flight	2 Flights			
Taipei					
London		1 Flight	1 Flight		1 Flight

Draw a graph to see whether there are direct flights between any two cities (in either direction).



Types of Graphs: Multigraphs



A multigraph is a graph that has no loop and at least *2* parallel edges between some pair of vertices.

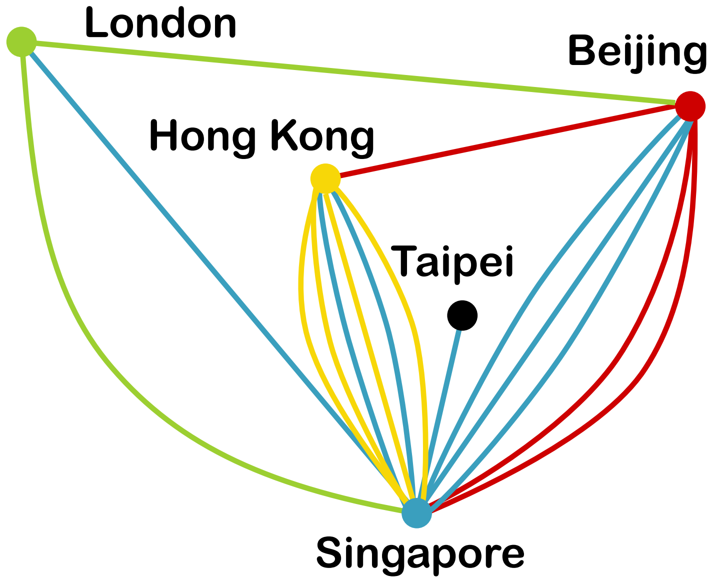
From\To	Hong Kong	Singapore	Beijing	Taipei	London
Hong Kong	■	4 Flights			
Singapore	2 Flights	■	3 Flights	1 Flight	1 Flight
Beijing	1 Flight	2 Flights	■		
Taipei				■	
London		1 Flight	1 Flight		1 Flight

Draw a graph with an edge for each flight that operates between two cities (in either direction).

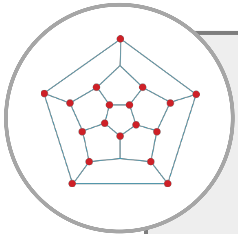
Types of Graphs: Multigraphs

From\To	Hong Kong	Singapore	Beijing	Taipei	London
Hong Kong		4 Flights			
Singapore	2 Flights		3 Flights	1 Flight	1 Flight
Beijing	1 Flight	2 Flights			
Taipei					
London		1 Flight	1 Flight		1 Flight

Draw a graph with an edge for each flight that operates between two cities (in either direction).



Types of Graphs: Directed (Multi) Graphs



A **directed** graph is a graph where edges $\{u,v\}$ are ordered, that is, edges have a direction. Parallel edges are allowed in **directed multigraphs**. Loops are allowed for both.

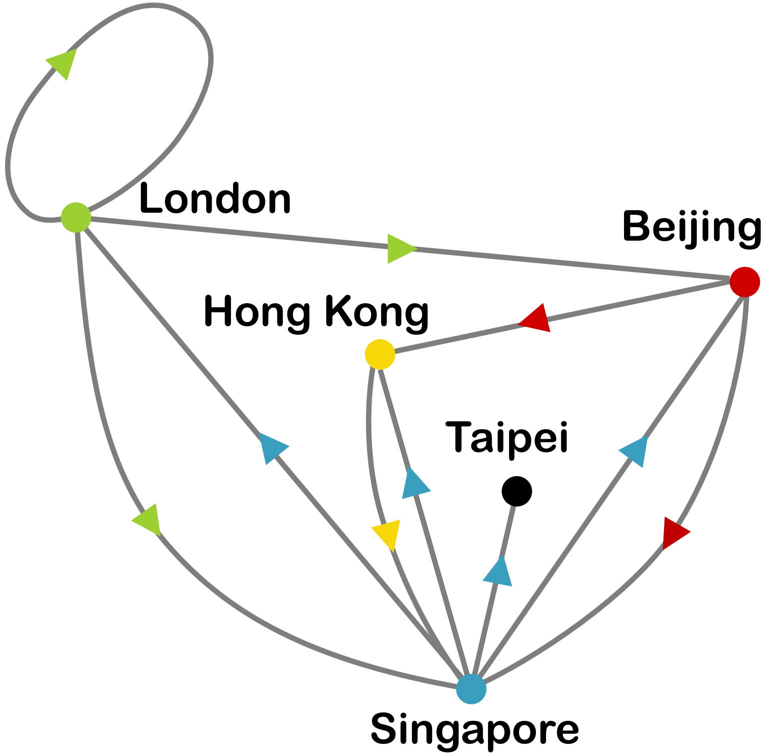
From\To	Hong Kong	Singapore	Beijing	Taipei	London
Hong Kong		4 Flights			
Singapore	2 Flights		3 Flights	1 Flight	1 Flight
Beijing	1 Flight	2 Flights			
Taipei					
London		1 Flight	1 Flight		1 Flight

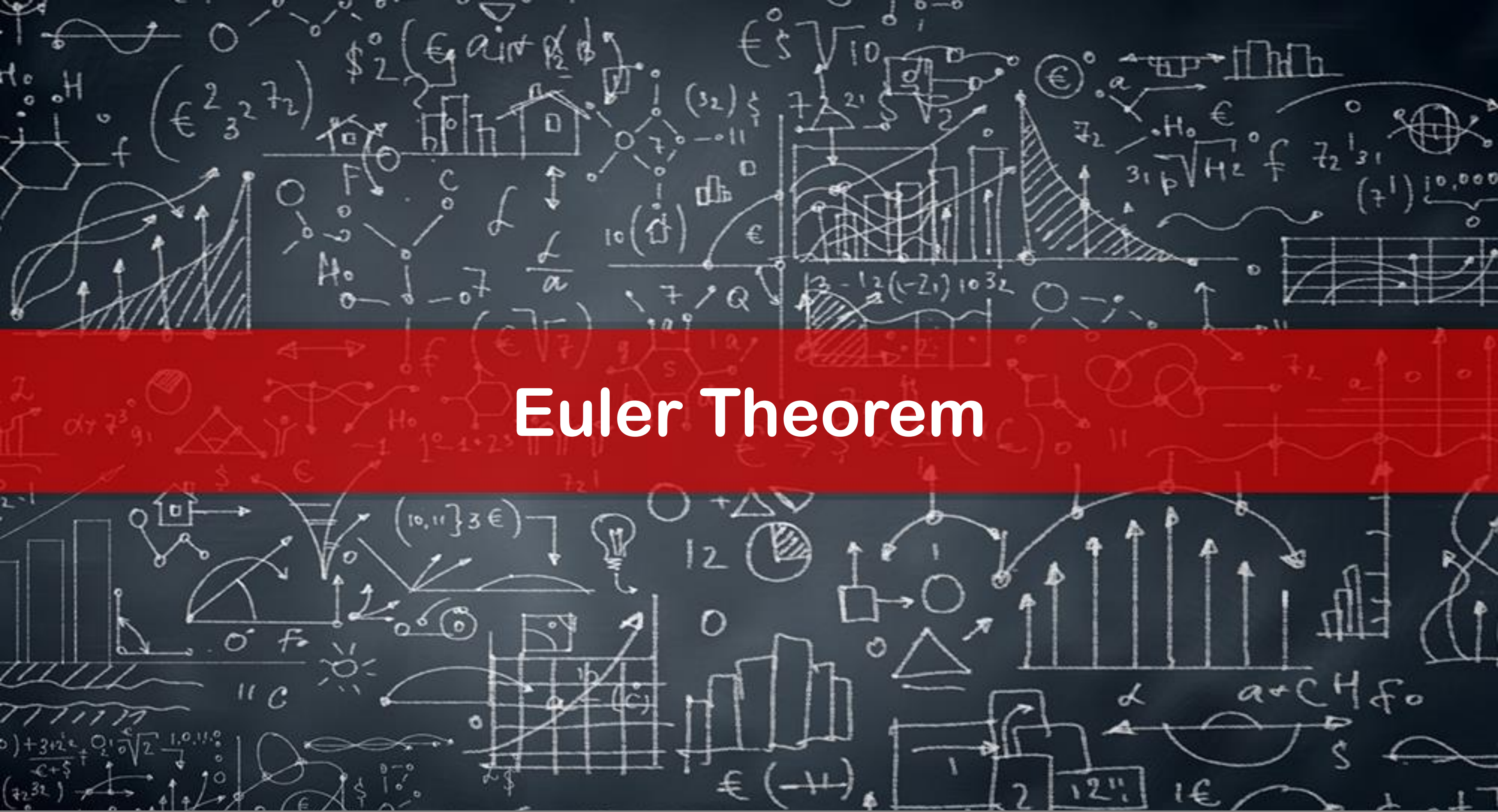
Draw a graph to see whether there are direct flights between any two cities (direction matters).

Types of Graphs: Directed (Multi) Graphs

From\To	Hong Kong	Singapore	Beijing	Taipei	London
Hong Kong		4 Flights			
Singapore	2 Flights		3 Flights	1 Flight	1 Flight
Beijing	1 Flight	2 Flights			
Taipei					
London		1 Flight	1 Flight		1 Flight

Draw a graph to see whether there are direct flights between any two cities (direction matters).



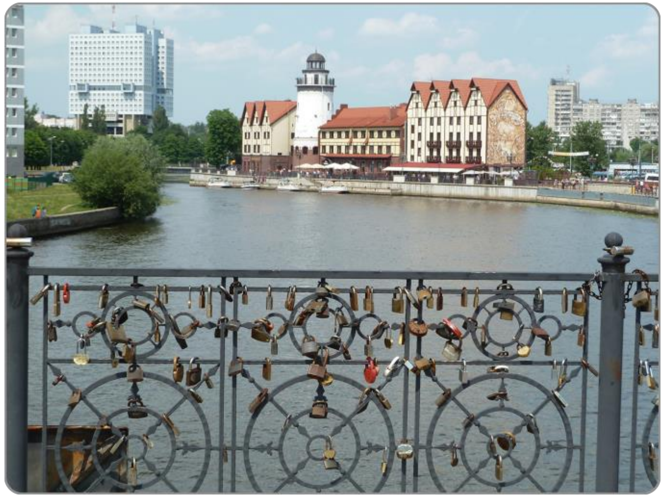


Euler Theorem

Euler Theorem: The Mathematician

Leonhard Euler introduced graphs in 1736 to solve the Königsberg Bridge problem.

What is the "Königsberg Bridge problem"?



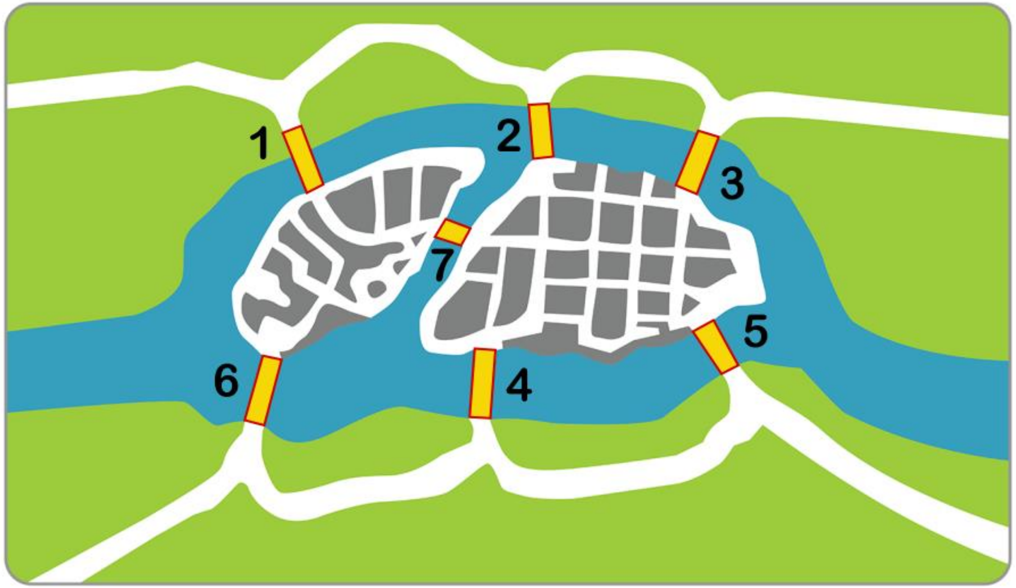
Kaliningrad (Königsberg) in Russia

**Leonhard Euler
1707 - 1783**

Portrait of Leonhard Euler by Jakob Emanuel Handmann under WikiCommons (PD-US)

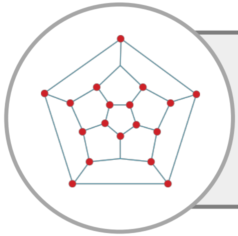
Euler Theorem: Origin (Bridges of Königsberg)

- Königsberg (now known as Kaliningrad in Russia) has *7* bridges.
- People tried (without success) to find a way to walk all *7* bridges without crossing a bridge twice.
- Leonhard Euler proved that it was impossible to walk all seven bridges without crossing a bridge twice.



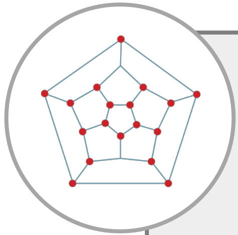
Seven Bridges of Königsberg

Euler Circuit: Definitions



A **Euler path** (Eulerian trail) is a walk on the edges of a graph which uses each edge in the original graph exactly once.

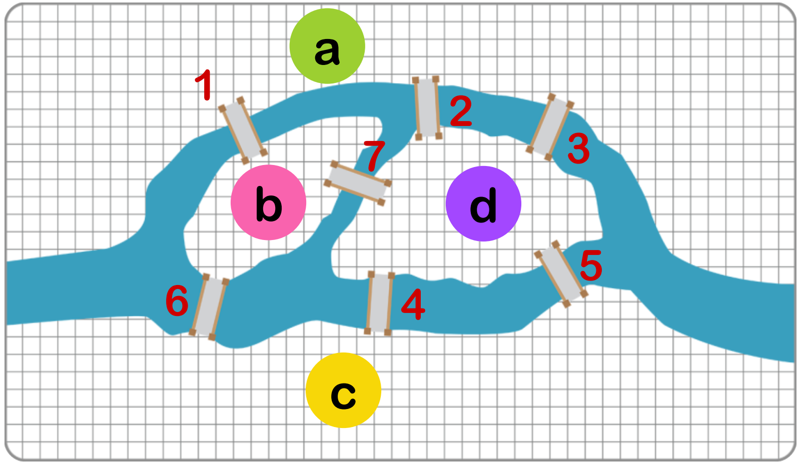
The beginning and end of the walk may or may not be the same vertex.

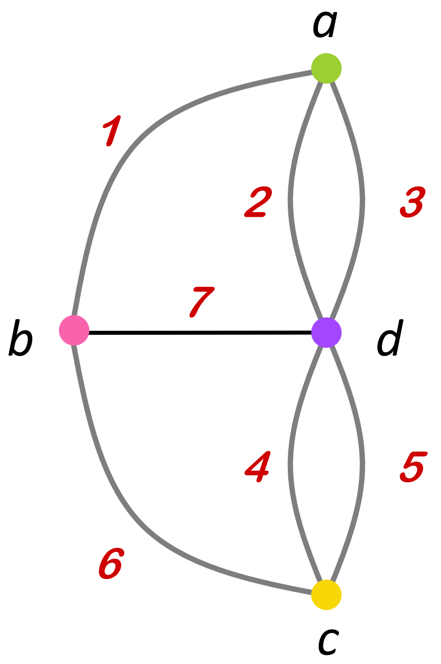


A **Euler circuit** (Eulerian cycle) is a walk on the edges of a graph which starts and ends at the same vertex, and uses each edge in the original graph exactly once.

Euler Circuit

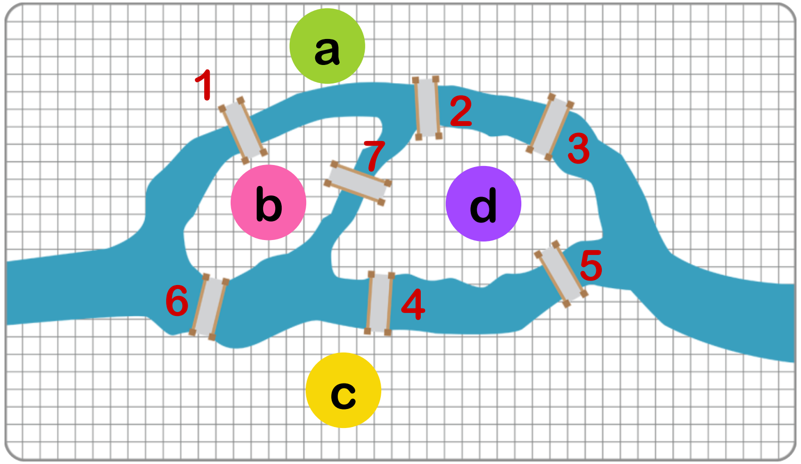
- Suppose the beginning and end are the same node u.
- The graph must be connected.
- At every vertex $v \neq u$, we reach v along one edge and go out along another, thus the number of edges incident at v (called the degree of v) is even.

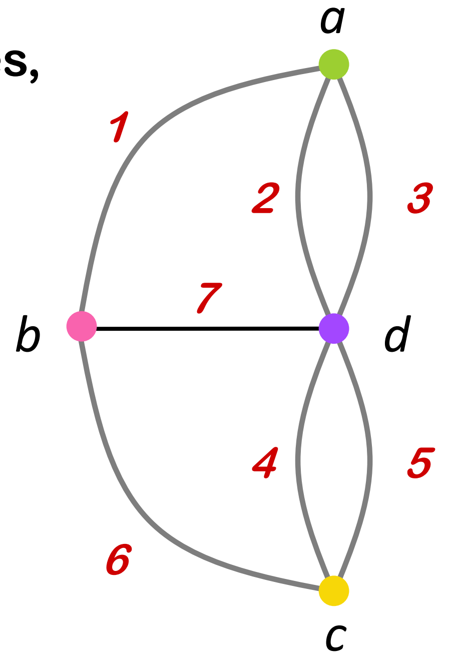




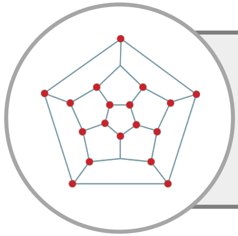
Euler Circuit

- The node *u* is visited once the first time we leave, and once the last time we arrive, and possibly in between (back and forth), thus the degree of *u* is even.
- Since the Königsberg Bridges graph has odd degrees, it has no solution!

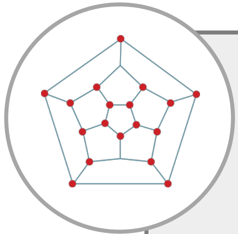




Euler Theorem



The **degree** of a vertex is the number of edges incident with it.



Theorem: consider a connected graph G.

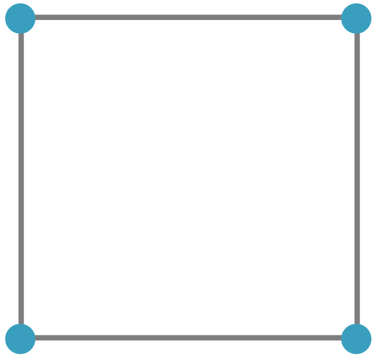
1. If G contains an Euler path that starts and ends at the same node, then all nodes of G have an even degree.
2. If G contains an Euler path, then exactly two nodes of G have an odd degree.

Euler Theorem

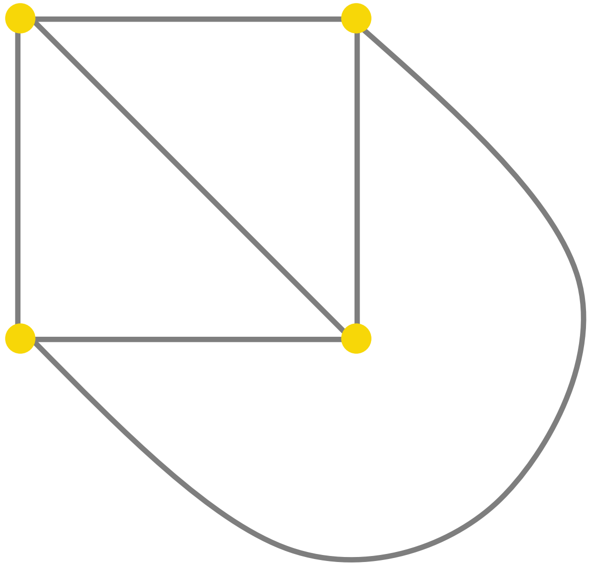
- Suppose G has an Euler path, which starts at v and finishes at w.
- Add the edge $\{v,w\}$.
- Then by the first part of the theorem, all nodes have even degrees, except for v and w which have odd degrees.

Euler Theorem: Examples

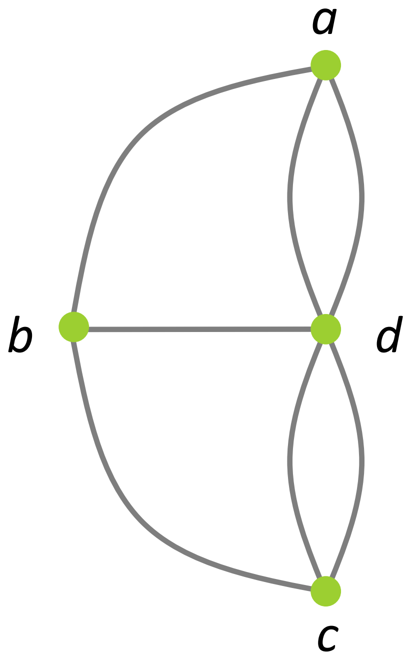
Note: Euler Theorem actually states an "if and only if" statement.



Euler Circuit



No Euler Path



No Euler Path

Topic Summary

Let’s recap…

- Graph theory has numerous applications (e.g., networks, distributed systems, coding theory)

- Parts of a graph:
  - Vertex
  - Edge
  - Adjacent
  - Incident

Let’s recap…

- Types of graphs:
  - Simple graph
  - Multigraph
  - Directed (multi) graph
- Euler path, Euler circuit and Euler theorem

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Discrete Mathematics

MH1812

Topic 10.2 - Graph Theory II

Dr. Wang Huaxiong

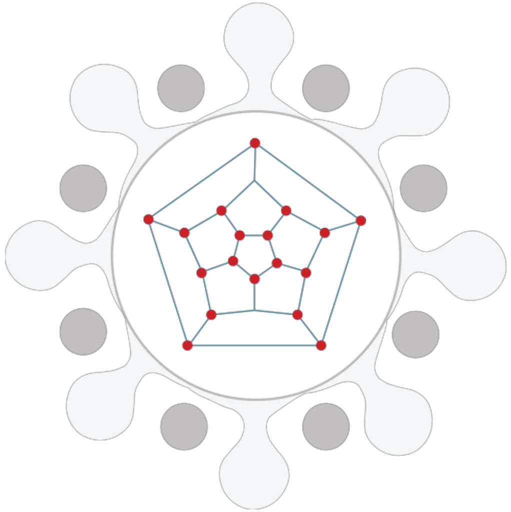
Topic Overview

What’s in store…

Graphs

More on Node Degree

Adjacency Matrix

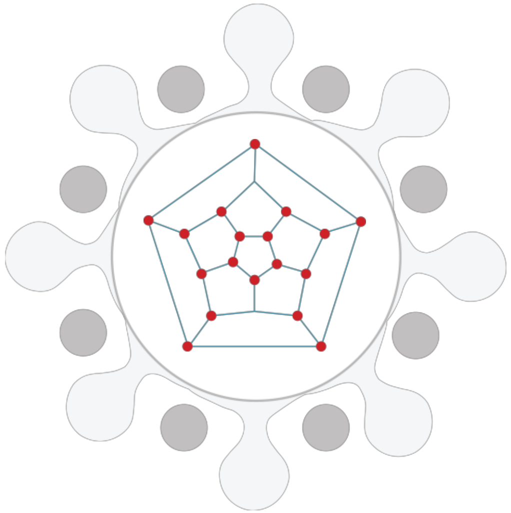


By the end of this lesson, you should be able to...

- Explain the difference between complete graph and bipartite graph.

- Explain how to find the total degree of an undirected graph.

- Explain how a graph can be represented by a matrix.



Graphs

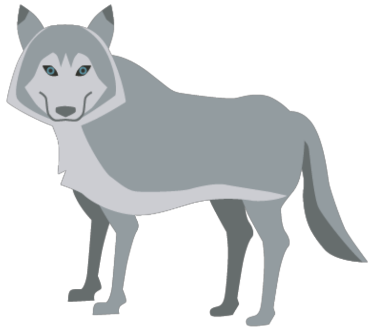
Graphs: Wolf, Goat and Cabbage

A classical puzzle that involves graphs.

From the left bank of the river, the ferryman has to transport the wolf, the goat and the cabbage to the right bank.

The boat is only big enough to transport one object/animal at a time, other than himself.

The wolf cannot be left alone with the goat, and the goat cannot be left alone with the cabbage.

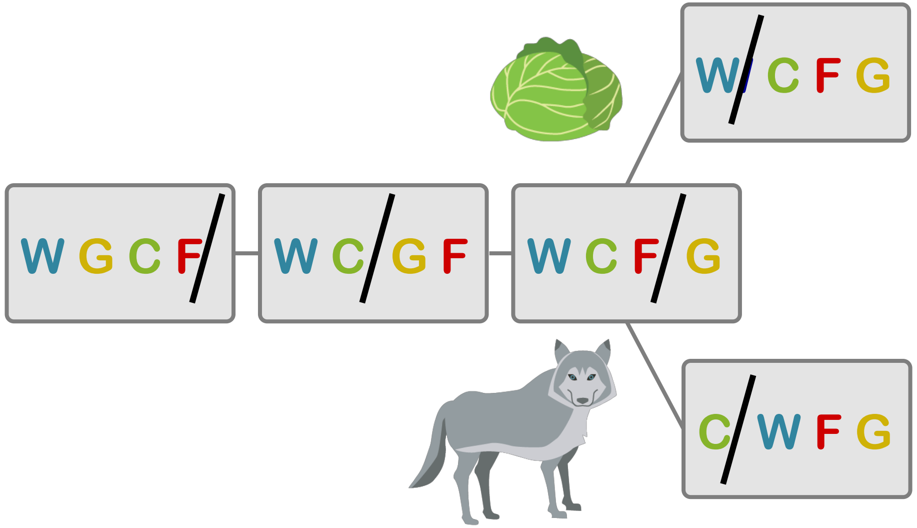


How should the ferryman proceed?

Graphs: Wolf, Goat and Cabbage

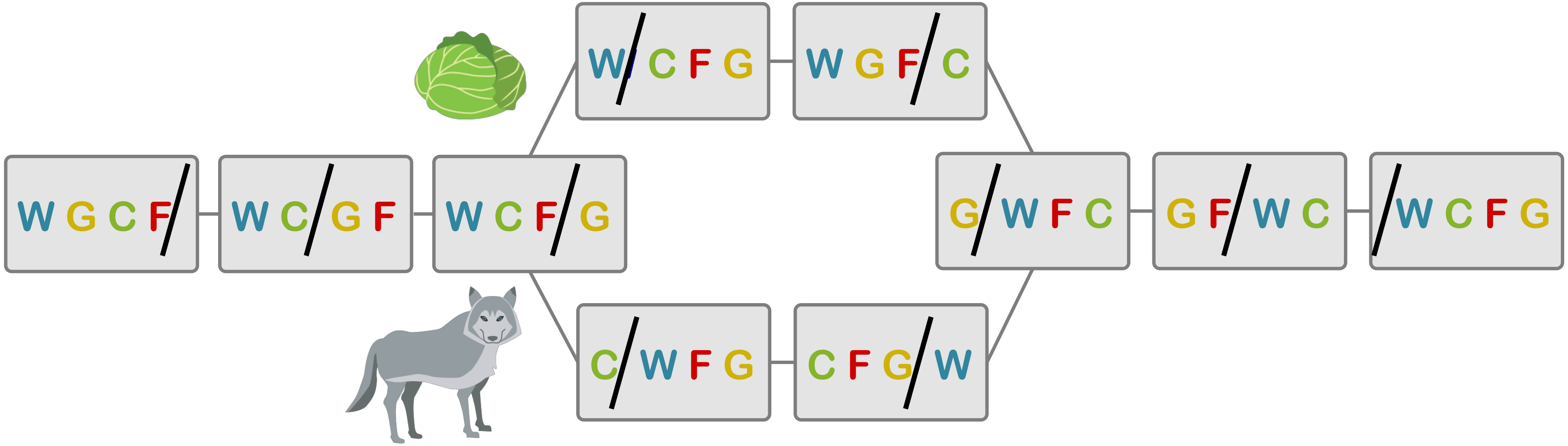
1. The ferryman takes the goat (no other choice)
2. The ferryman returns
3. Either he takes the cabbage or the wolf

F = ferryman
G = goat
W = wolf
C = cabbage

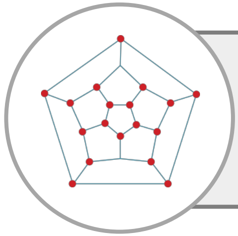


Graphs: Wolf, Goat and Cabbage

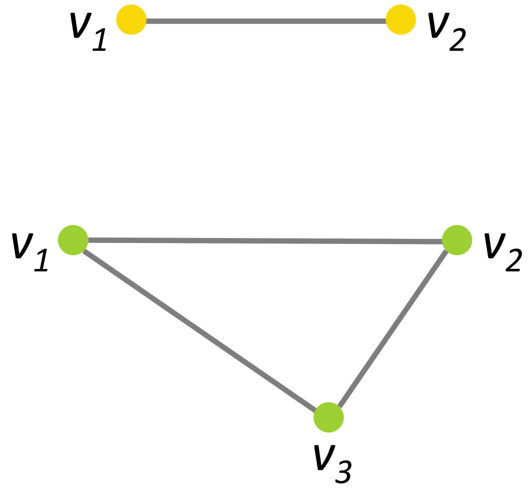
4. Either he takes:
   a. The cabbage, brings back the goat, leaves the goat and takes the wolf across, returns, and takes the goat across.
   b. The wolf, brings back the goat, leaves the goat and takes the cabbage across, returns, and takes the goat across.

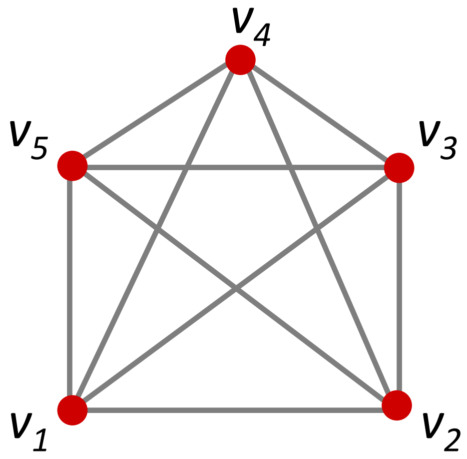


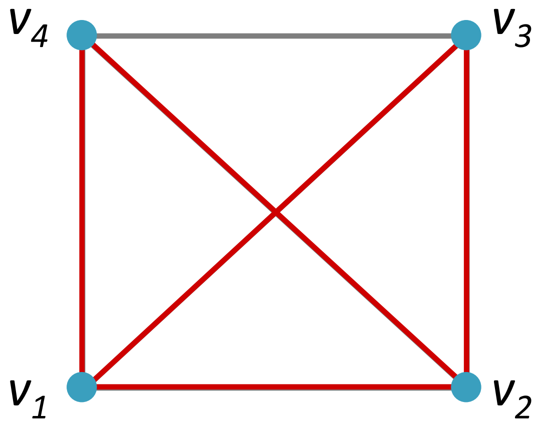
Graphs: Complete Graphs



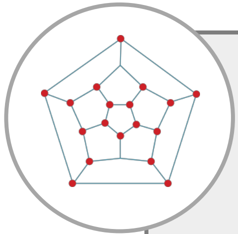
A **complete graph with *n* vertices** is a simple graph that has every vertex adjacent to every other distinct vertex.



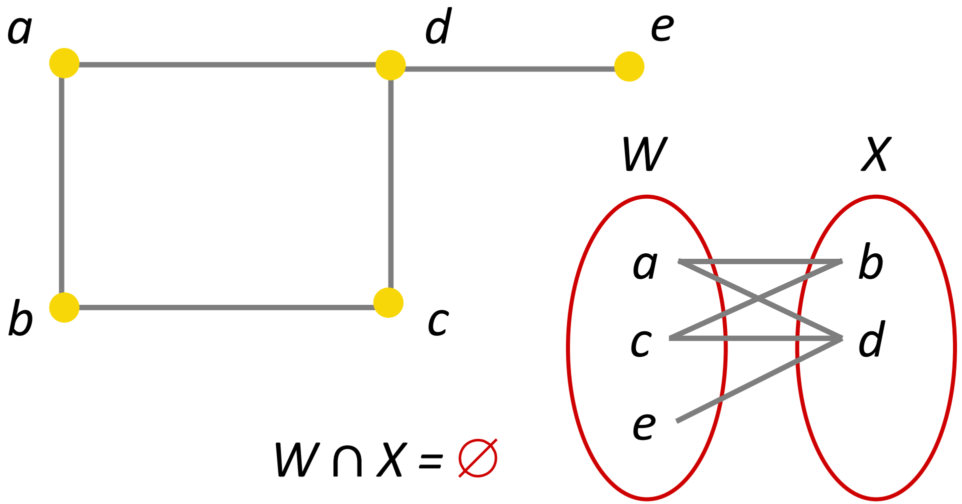




Graphs: Bipartite Graphs

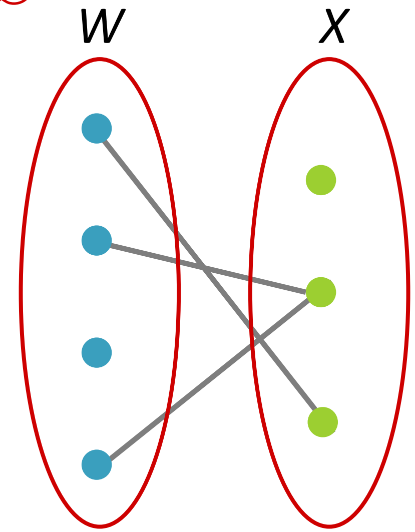


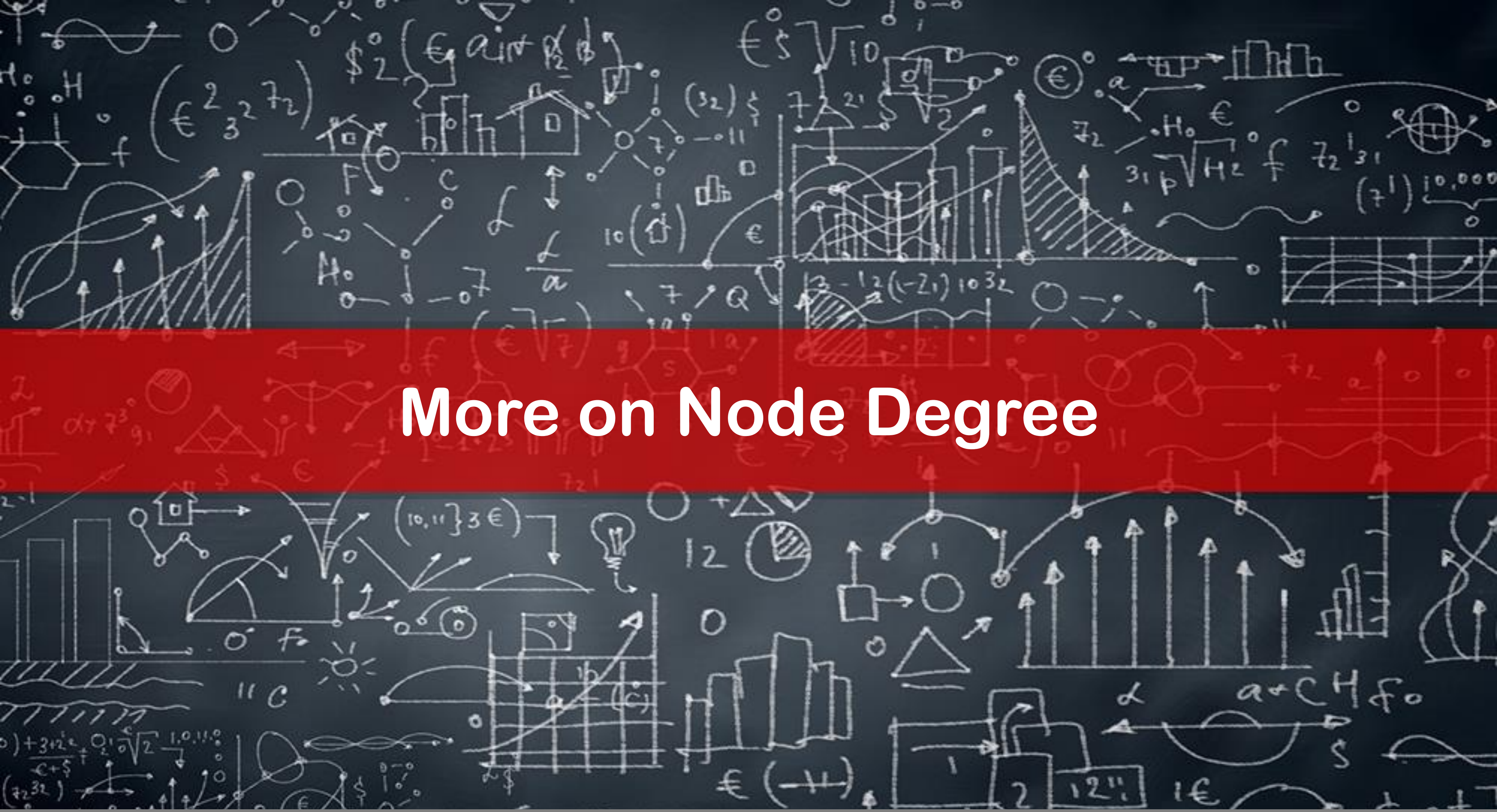
A **bipartite graph** is a graph whose vertices can be partitioned into *2* (disjoint) subsets W and X such that each edge connects a $w \in W$ and a $x \in X$.



$W \cap X = \varnothing$

$W \cap X = \varnothing$

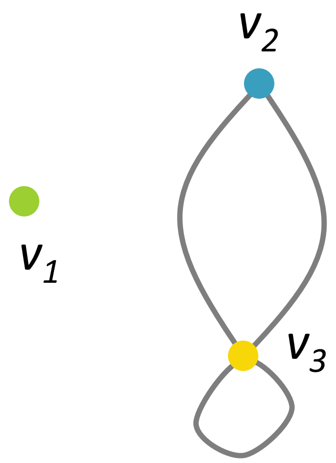




More on Node Degree

Mode on Node Degree: Definitions

The **degree deg(v)** of a vertex v in an undirected graph is the number of edges incident with it (a loop at a vertex contributes twice). In-degree and out-degree are distinguished for directed graphs.



Total degree = $\deg(v_1) + \deg(v_2) + \deg(v_3) = 0 + 2 + 4 = 6$

The **total degree deg(G)** of an undirected graph G is the sum of the degrees of all the vertices of G: $\sum_{v \in V} \deg(v)$

Mode on Node Degree: The Handshaking Theorem

Let $G = (V,E)$ **be an undirected graph with** e **edges.**

Then

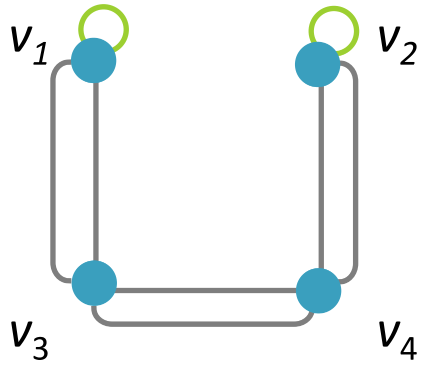
$$\boxed{2e = \sum_{v \in V} \deg(v)}$$

(Note that this even applies if multiple edges and loops are present.)

Mode on Node Degree: The Handshaking Theorem

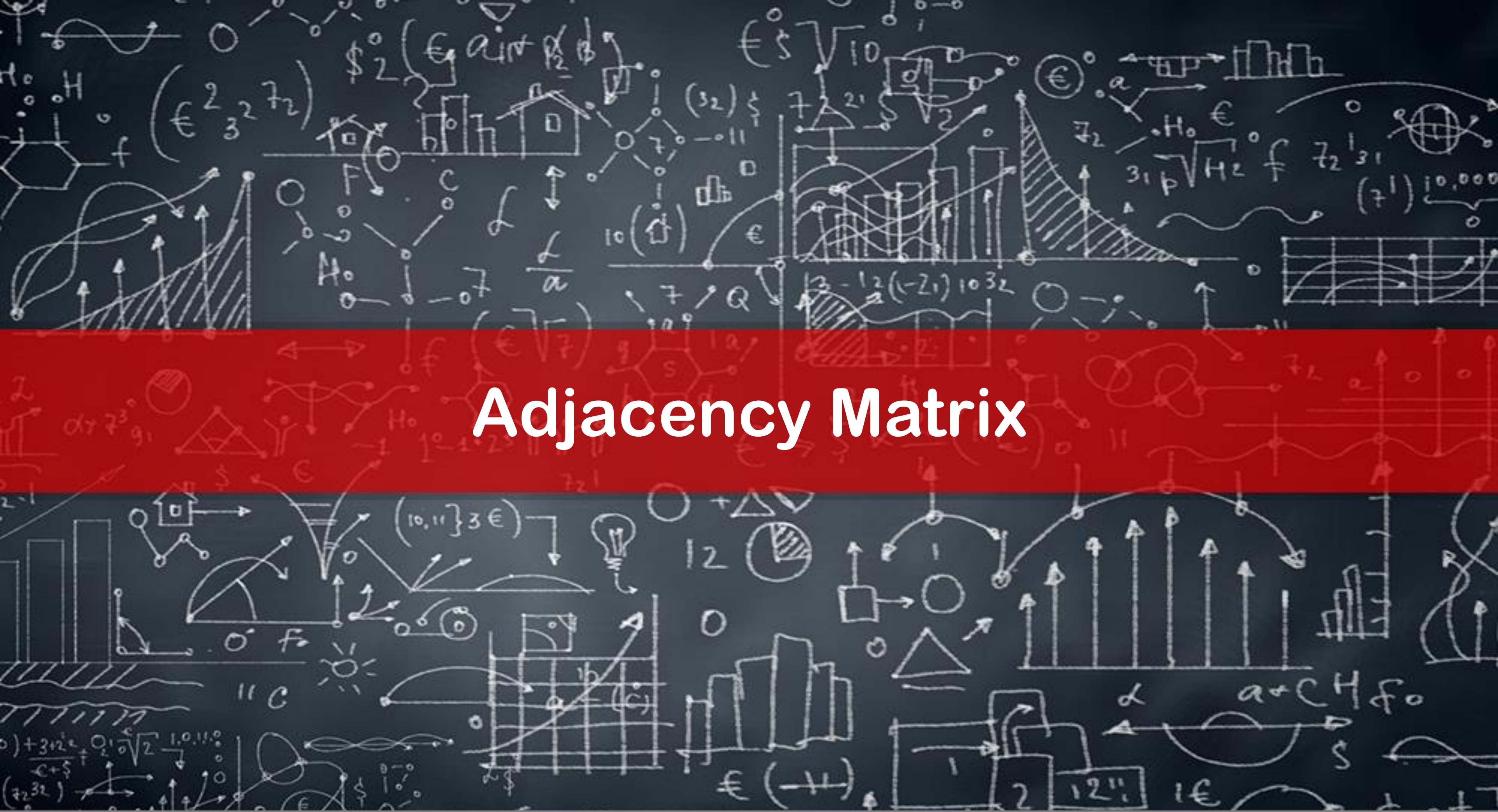
Proof

Choose an $e \in E(G)$ with endpoints $v, w \in V$. e contributes 1 to $\deg(v)$ and 1 to $\deg(w)$. This is true even when $v = w$. Thus, each edge contributes 2 to the total degree.



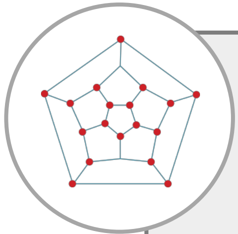
$$\deg(v_1) = \deg(v_2) = \deg(v_3) = \deg(v_4) = 4$$

$$2e = \sum \deg(v) = 4 \times 4 = 16 \text{ and } e = 8$$

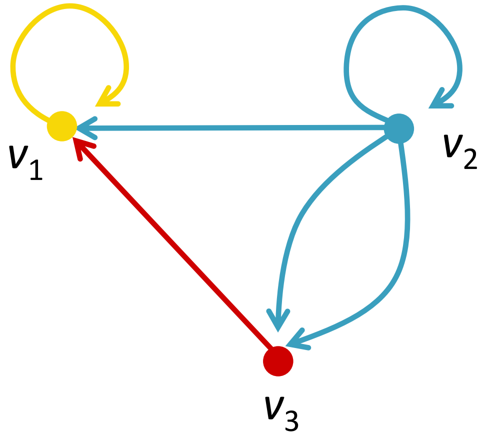


Adjacency Matrix

Adjacency Matrix: Definition



A graph can be represented by a matrix $A = (a_{ij})$ called adjacency matrix, with a_{ij} = the number of arrows from v_i to v_j.



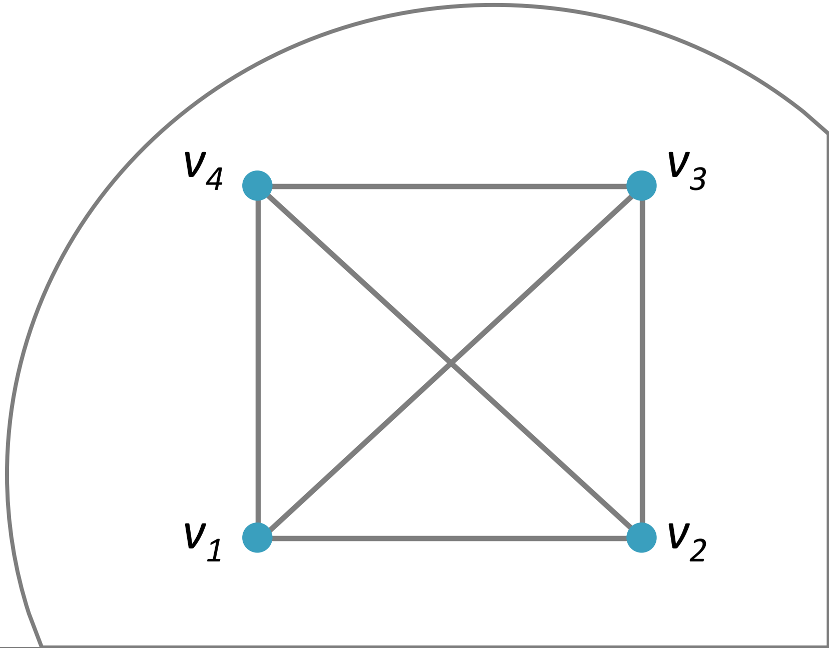
$$A = \begin{array}{c|ccc} & v_1 & v_2 & v_3 \\ \hline v_1 & 1 & 0 & 0 \\ v_2 & 1 & 1 & 2 \\ v_3 & 1 & 0 & 0 \end{array}$$

What is the adjacency matrix of a complete graph?

Adjacency Matrix: Example

What is the adjacency matrix of a complete graph?

$$\begin{array}{c|cccc} & v_1 & v_2 & v_3 & v_4 \\ \hline v_1 & 0 & 1 & 1 & 1 \\ v_2 & 1 & 0 & 1 & 1 \\ v_3 & 1 & 1 & 0 & 1 \\ v_4 & 1 & 1 & 1 & 0 \end{array}$$



Topic Summary

Let’s recap…

- Types of graphs:
  - Complete graph
  - Bipartite graph
- Handshaking theorem
- A graph represented by a matrix

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Discrete Mathematics

MH1812

Topic 10.3 - Graph Theory III

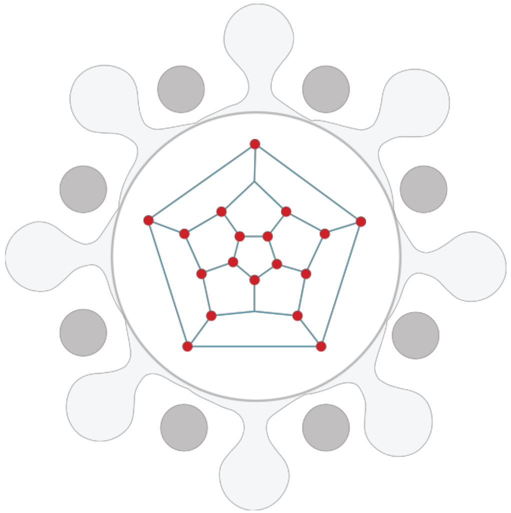
Dr. Wang Huaxiong

Topic Overview

What's in store…

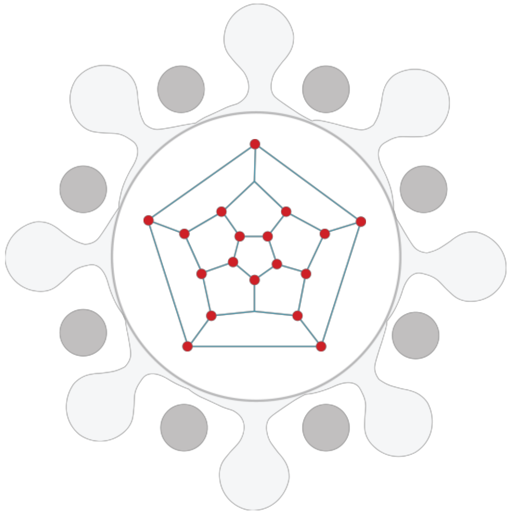
Hamiltonian Circuit

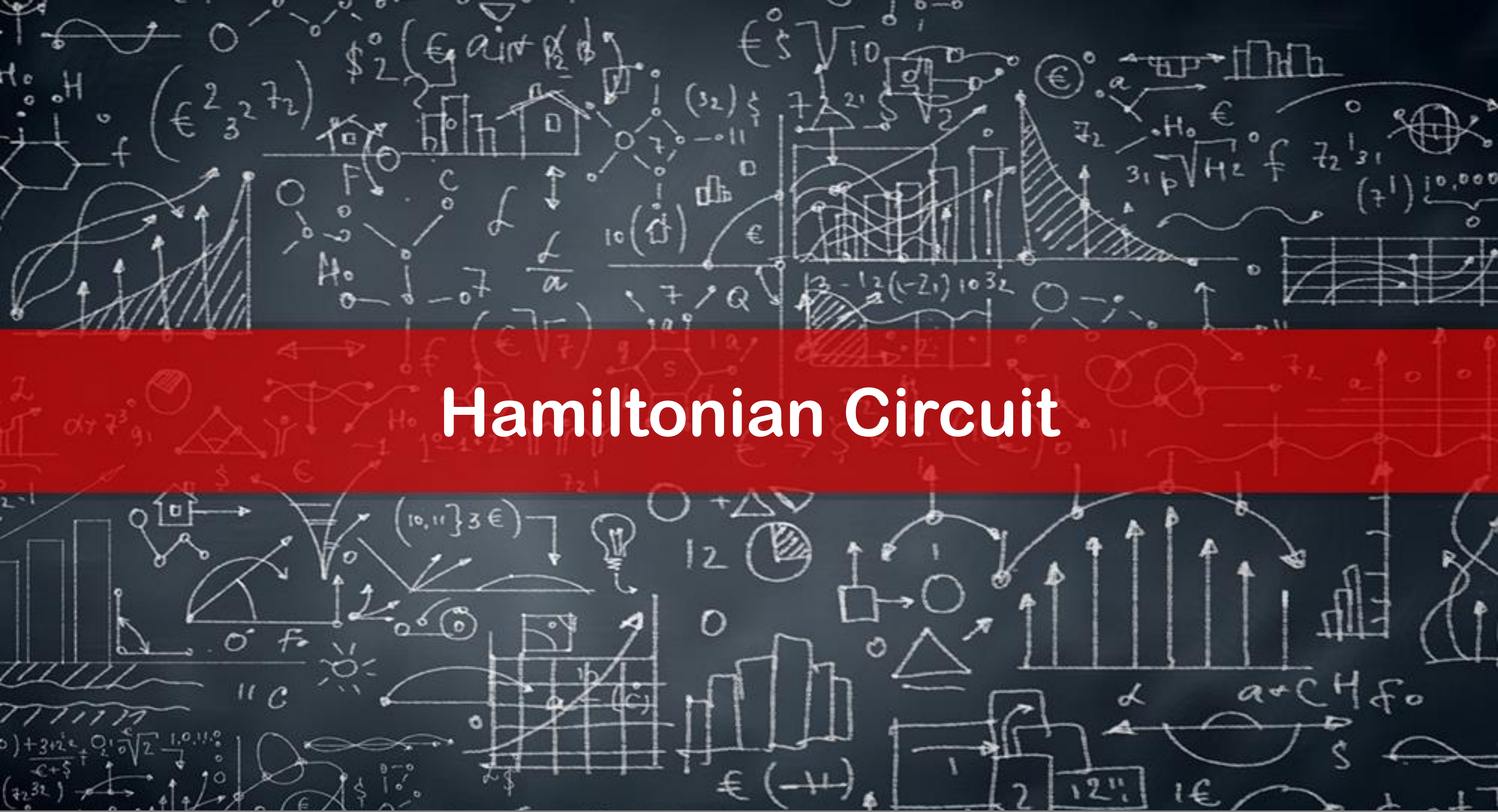
Graph Isomorphism



By the end of this lesson, you should be able to...

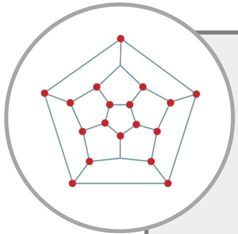
- Explain the concepts of the Hamiltonian circuit.
- Explain what is graph isomorphism.





Hamiltonian Circuit

Hamiltonian Circuit: Definition



A **Hamiltonian path** of a graph G is a walk such that every vertex is visited exactly once.

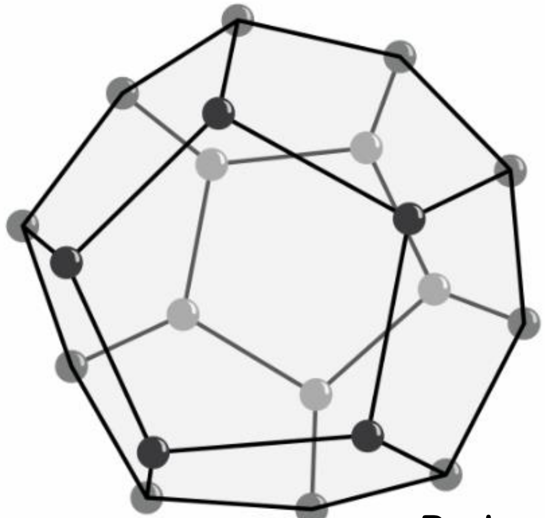
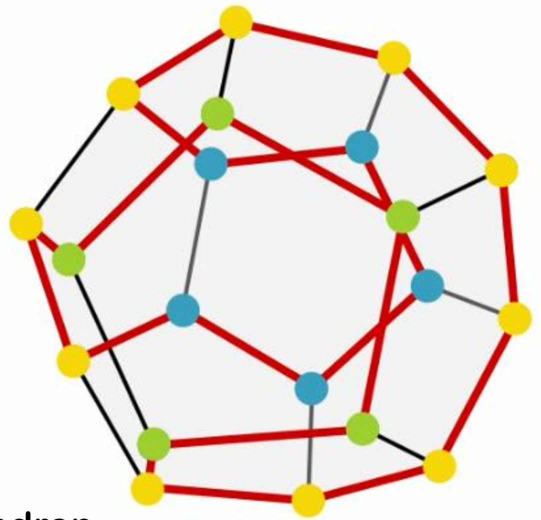
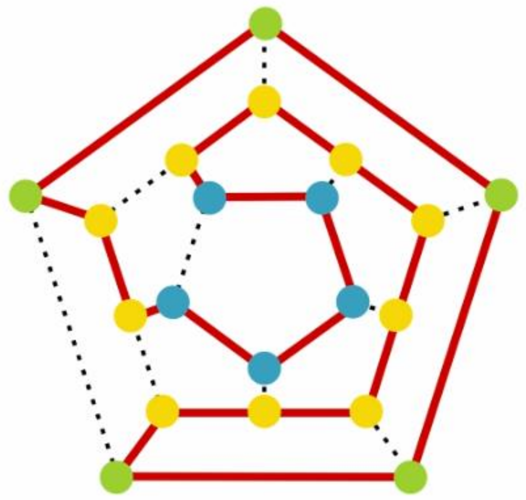
A **Hamiltonian circuit** of a graph G is a closed walk such that every vertex is visited exactly once (except the same start/end vertex).

William Rowan Hamilton
1805 - 1865

William Rowan Hamilton under WikiCommons (PD-US)

Hamiltonian Circuit: The Icosian Game (1857)

- Along the edges of a dodecahedron, find a path such that every vertex is visited a single time, and the ending point is the same as the starting point.
- Hamilton sold it to a London game dealer in 1859 for 25 pounds.

Dodecahedron

Hamiltonian Circuit: Hamiltonian vs. Eulerian

- Path (or trail) vs. circuit (or cycle):
  - For circuits, the walk starts and finishes at the same vertex.
  - But for a path, the starting vertex is different from the ending one.
- Eulerian: walk through every edge exactly once.
- Hamiltonian: walk through every vertex exactly once.

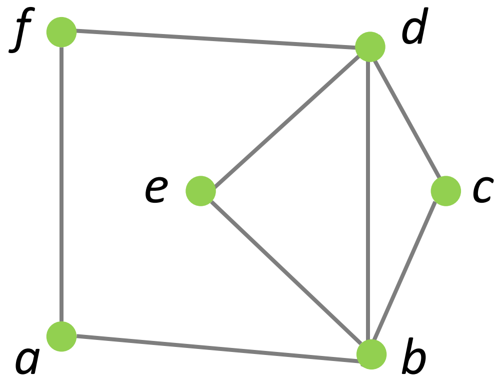
William Rowan Hamilton
1805 - 1865

Leonhard Euler
1707 - 1783

Portrait of Leonhard Euler by Jakob Emanuel Handmann under WikiCommons (PD-US)

William Rowan Hamilton under WikiCommons (PD-US)

Hamiltonian Circuit: Examples

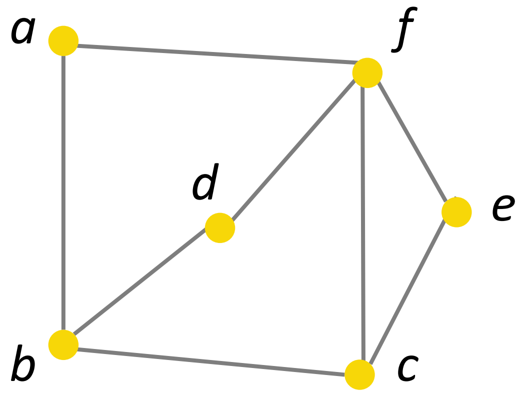


- Euler circuit
- Hamiltonian path
- No Hamiltonian circuit

$a \rightarrow b \rightarrow e \rightarrow d \rightarrow c \rightarrow b \rightarrow d \rightarrow f \rightarrow a$

$f \rightarrow a \rightarrow b \rightarrow e \rightarrow d \rightarrow c$

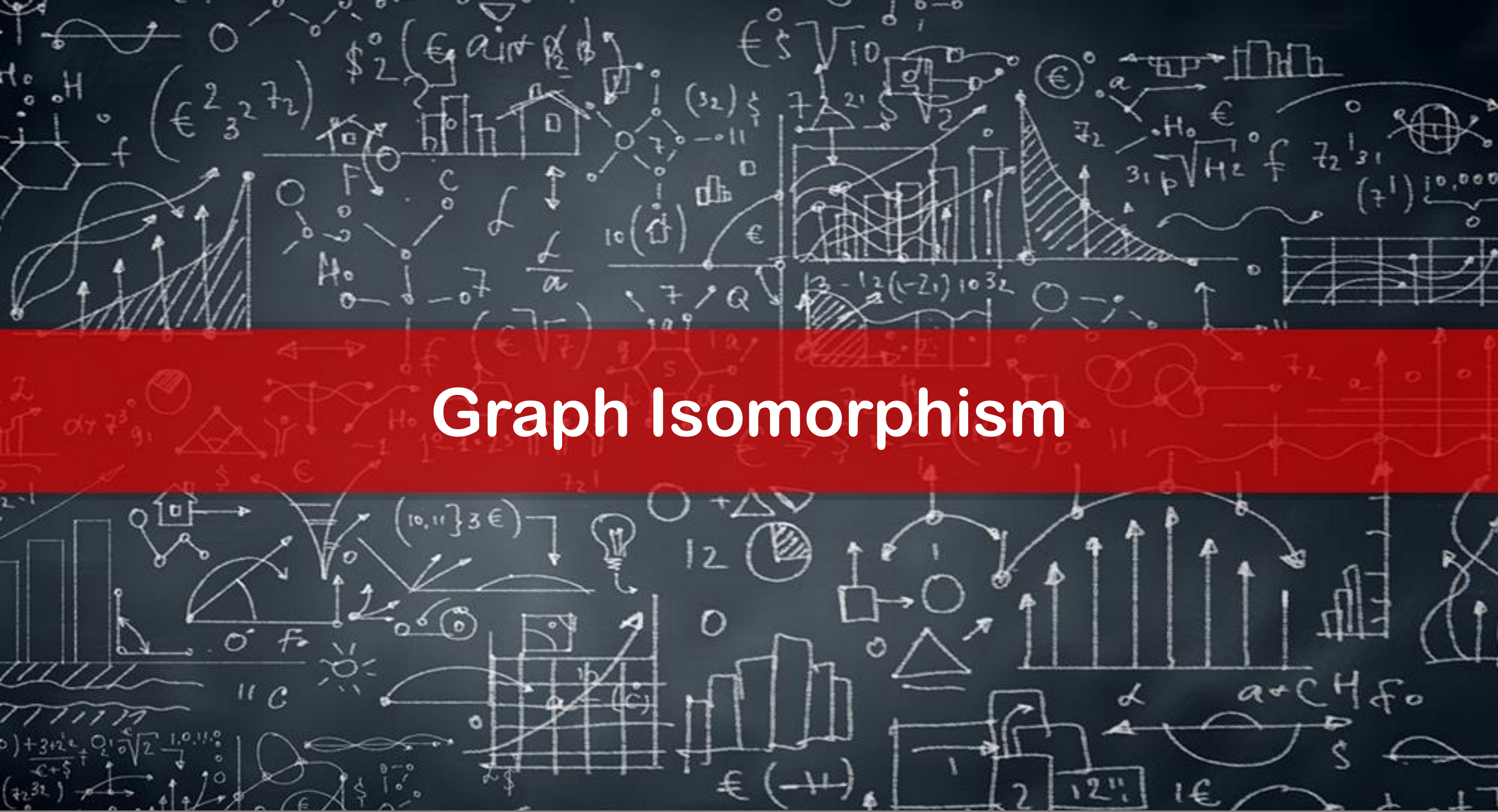
Hamiltonian Circuit: Examples



- **No Euler circuit, but Euler path**
- **Hamiltonian path**
- **No Hamiltonian circuit**

$$c \rightarrow e \rightarrow f \rightarrow c \rightarrow b \rightarrow d \rightarrow f \rightarrow a \rightarrow b$$

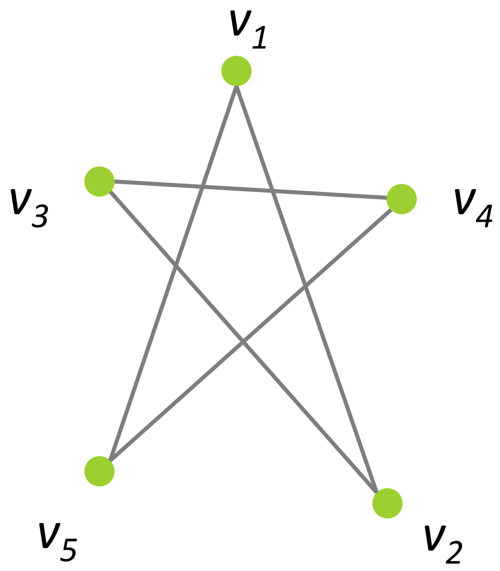
$$e \rightarrow c \rightarrow f \rightarrow d \rightarrow b \rightarrow a$$

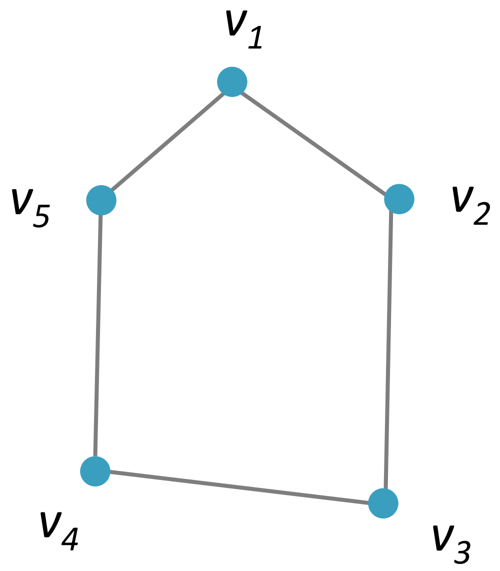


Graph Isomorphism

Graph Isomorphism: Pictorial Representations

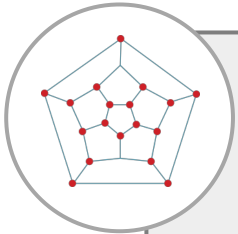
A graph can have many pictorial representations.





$$\begin{array}{c|c} & \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

Graph Isomorphism: Definition

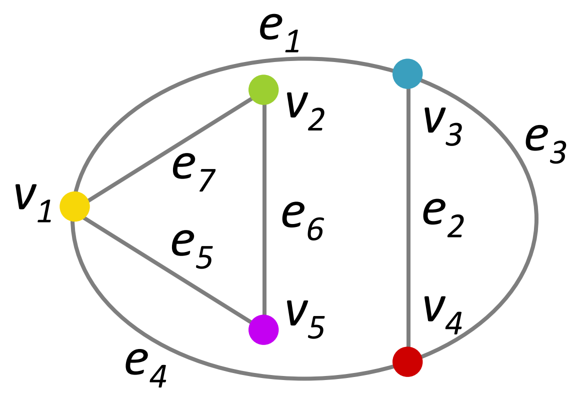


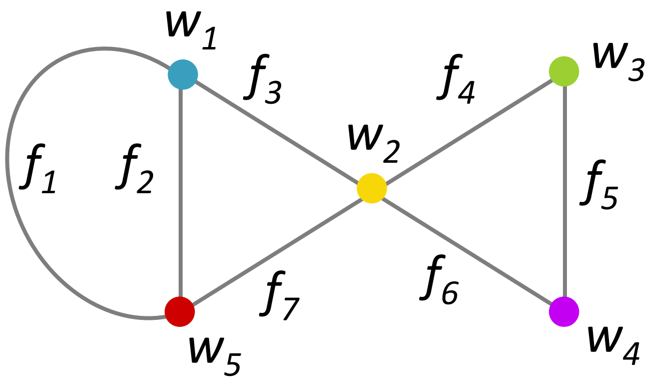
A graph $G = (V_G, E_G)$ is isomorphic to a graph $H = (V_H, E_H)$ if and only if there exists two bijections mapping the vertex sets and edge sets, respectively:

$$g: V_G \to V_H,\ h: E_G \to E_H$$

such that an edge $e \in E_G$ is incident on $v, w \in V_G \Leftrightarrow$ the edge $h(e) \in E_H$ is incident on $g(v), g(w) \in V_H$.

Graph Isomorphism: Example





Vertex and edge bijections:

$g = \{(v_1,w_2), (v_2,w_3), (v_3,w_1), (v_4,w_5), (v_5,w_4)\}$

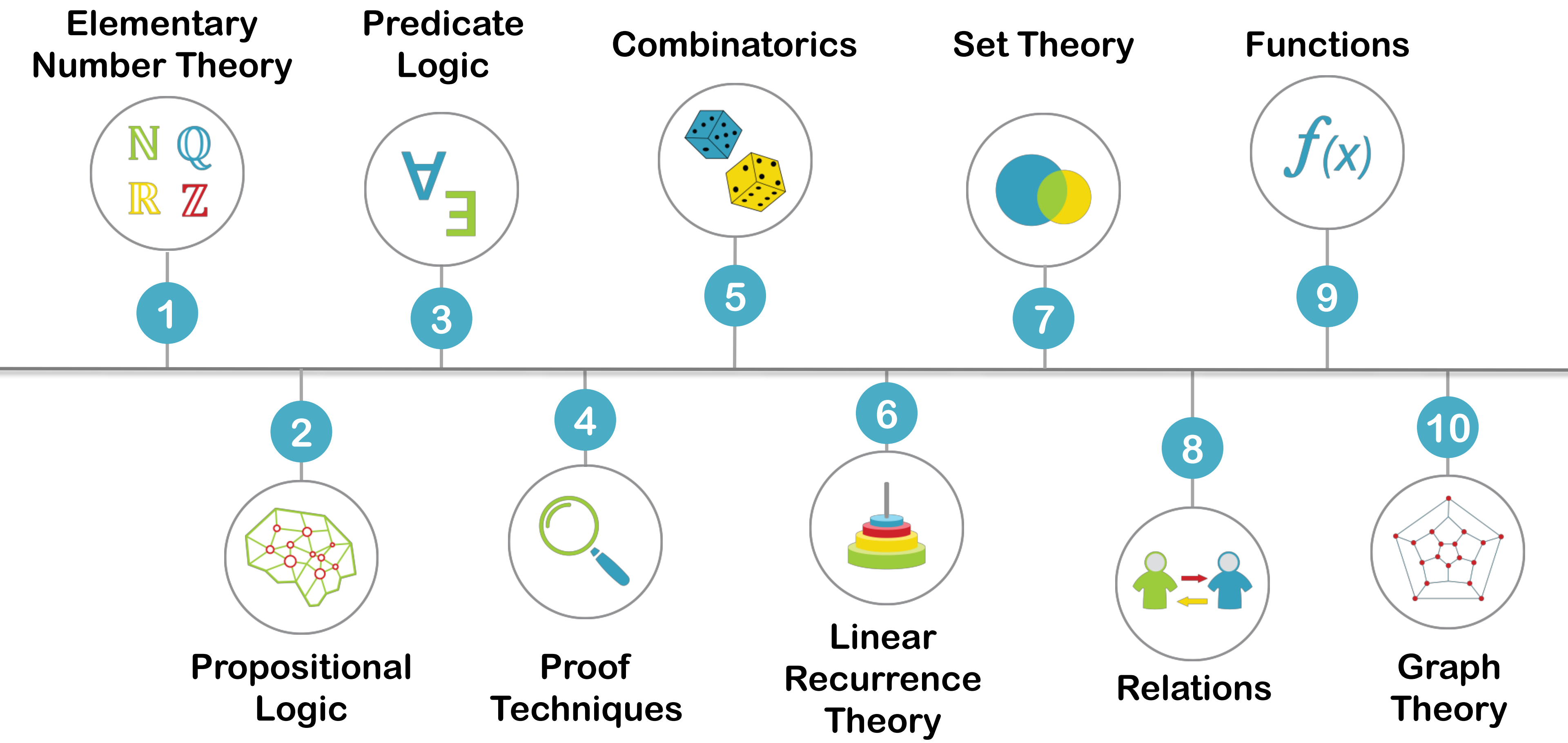
$h = \{(e_1, f_3), (e_2, f_2), (e_3, f_1), (e_4, f_7), (e_5, f_6), (e_6, f_5), (e_7, f_4)\}$

Topic Summary

Let's recap…

- Basic definitions: graph, vertex (node), edge, loop
- Node degree, graph degree, handshaking theorem
- Types of graphs: simple, multigraph, (un)directed, complete, bipartite
- Euler path and circuit
- Hamiltonian path and circuit
- Adjacency matrix and graph isomorphism

Your Learning Roadmap


Elementary Number Theory
1
N Q R Z
Propositional Logic
2
Predicate Logic
3
∀ ∃
Proof Techniques
4
Combinatorics
5
Linear Recurrence Theory
6
Set Theory
7
Relations
8
Functions
9
f(x)
Graph Theory
10