Discrete Mathematics 2

Graphs

Simple Graph

Definition 1. **A simple graph *G* = *(V, E)* consists of *V*, a nonempty set of vertices, and *E*, a set of unordered pairs of distinct elements of *V* called edges.**

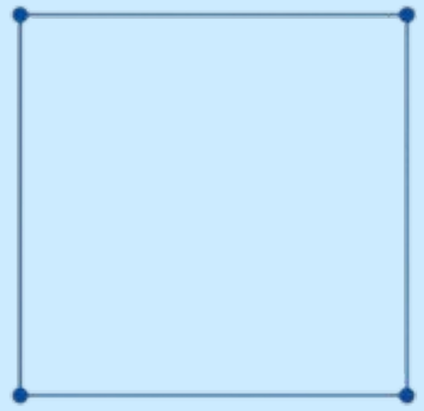
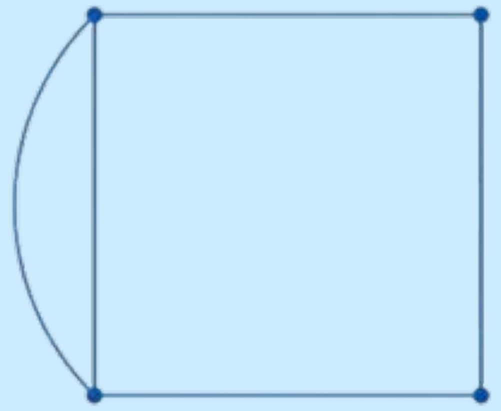
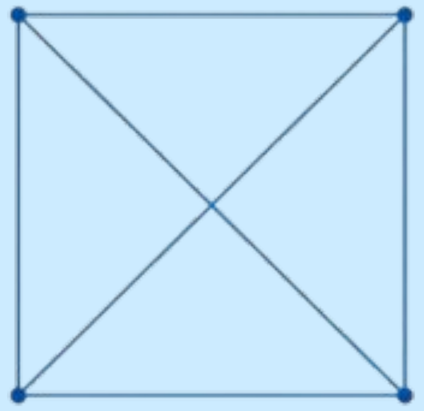
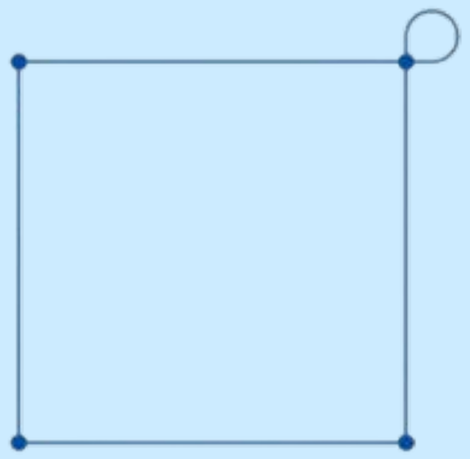
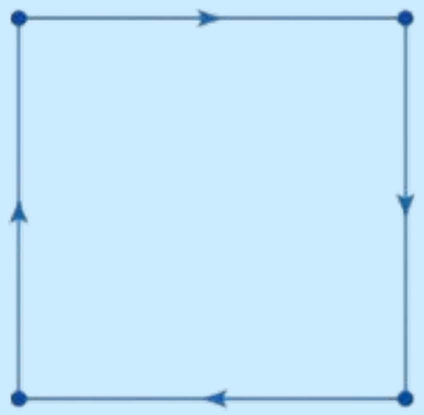
A simple graph is an undirected graph.

Undirected graphs

In undirected graphs, edges have no specific direction. Edges are always "two-way"

Simple graph

Which graph is simple?

A simple graph

What is the maximum number of edges in a simple graph G = (V, E)?

A simple graph

What is the maximum number of edges in a simple graph G = (V, E)?

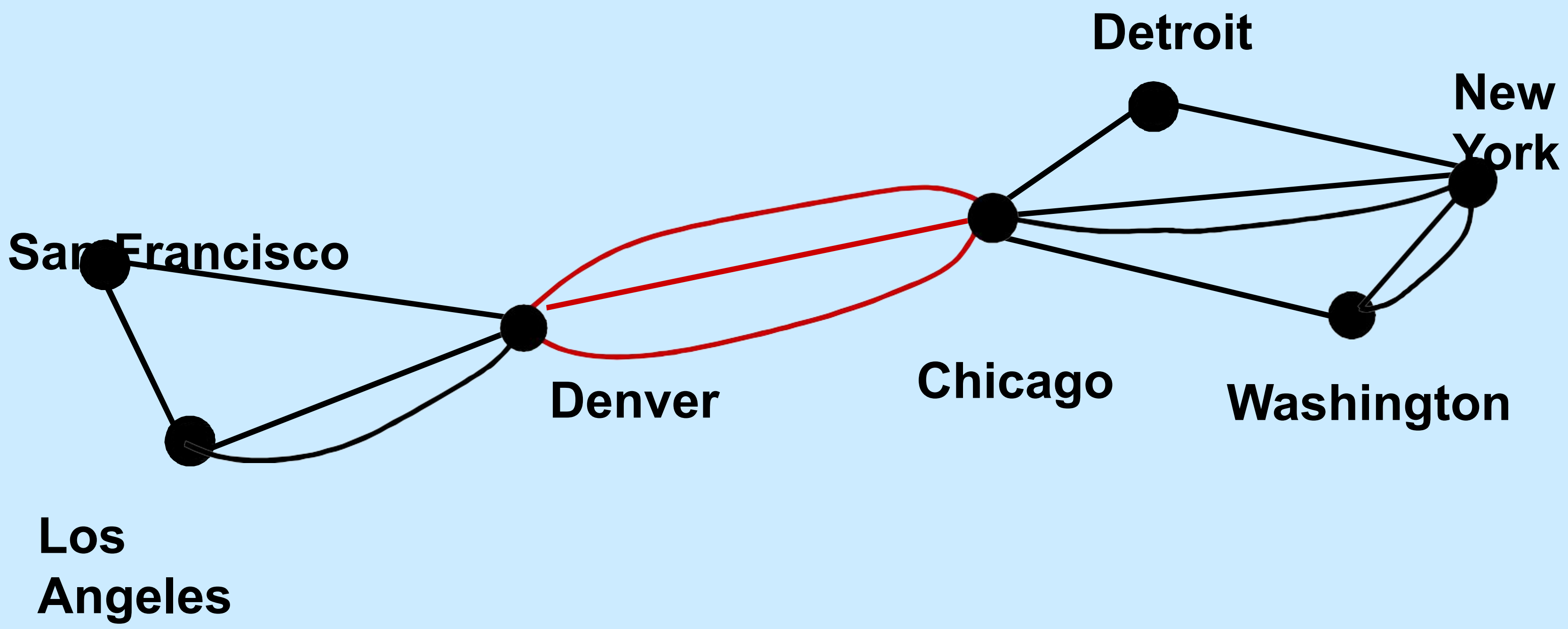
$$\frac{|\boldsymbol{V}| \times (|\boldsymbol{V}| - \mathbf{1})}{\mathbf{2}}$$

A Non-Simple Graph

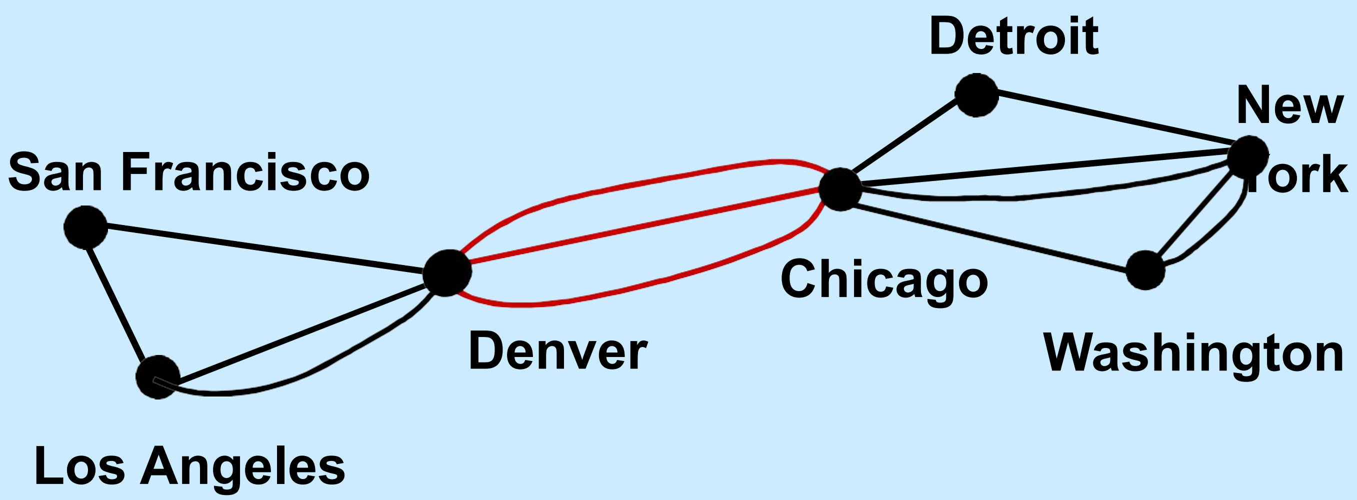
Definition 2. In a **multigraph** $G = (V, E)$ two or more edges may connect the same pair of vertices.

A Multigraph

THERE CAN BE MULTIPLE TELEPHONE LINES BETWEEN TWO COMPUTERS IN THE NETWORK.

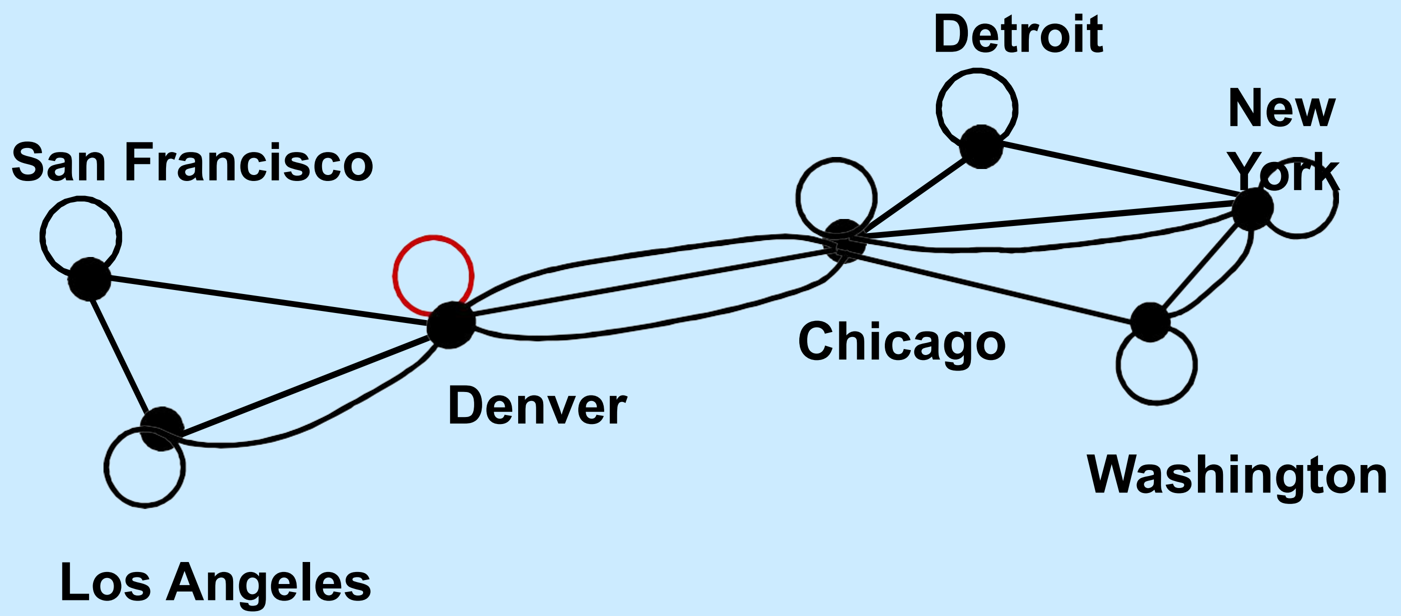


Multiple Edges



Two edges are called *multiple or parallel edges* if they connect the same two distinct vertices.

Loops



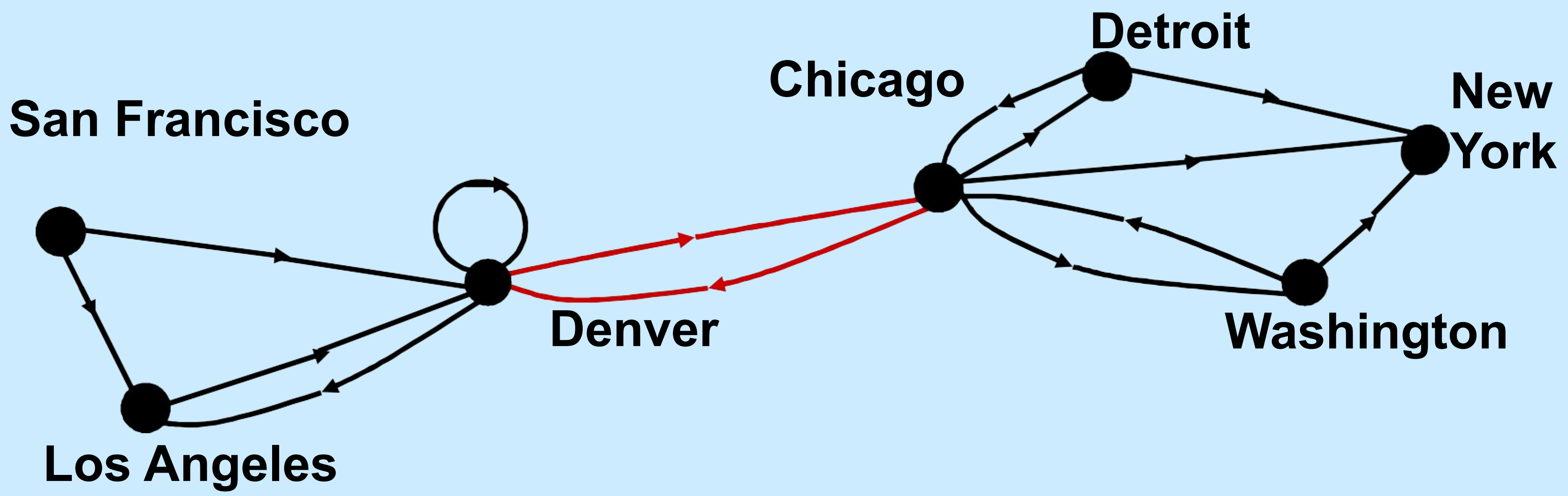
An edge is called a *loop* if it connects a vertex to itself.

A Directed Graph

Definition 4. **In a directed graph $G = (V, E)$ the edges are ordered pairs of (not necessarily distinct) vertices.**

A Directed Graph

SOME TELEPHONE LINES IN THE NETWORK MAY OPERATE IN ONLY ONE DIRECTION . Those that operate in two directions are represented by pairs of edges in opposite directions.

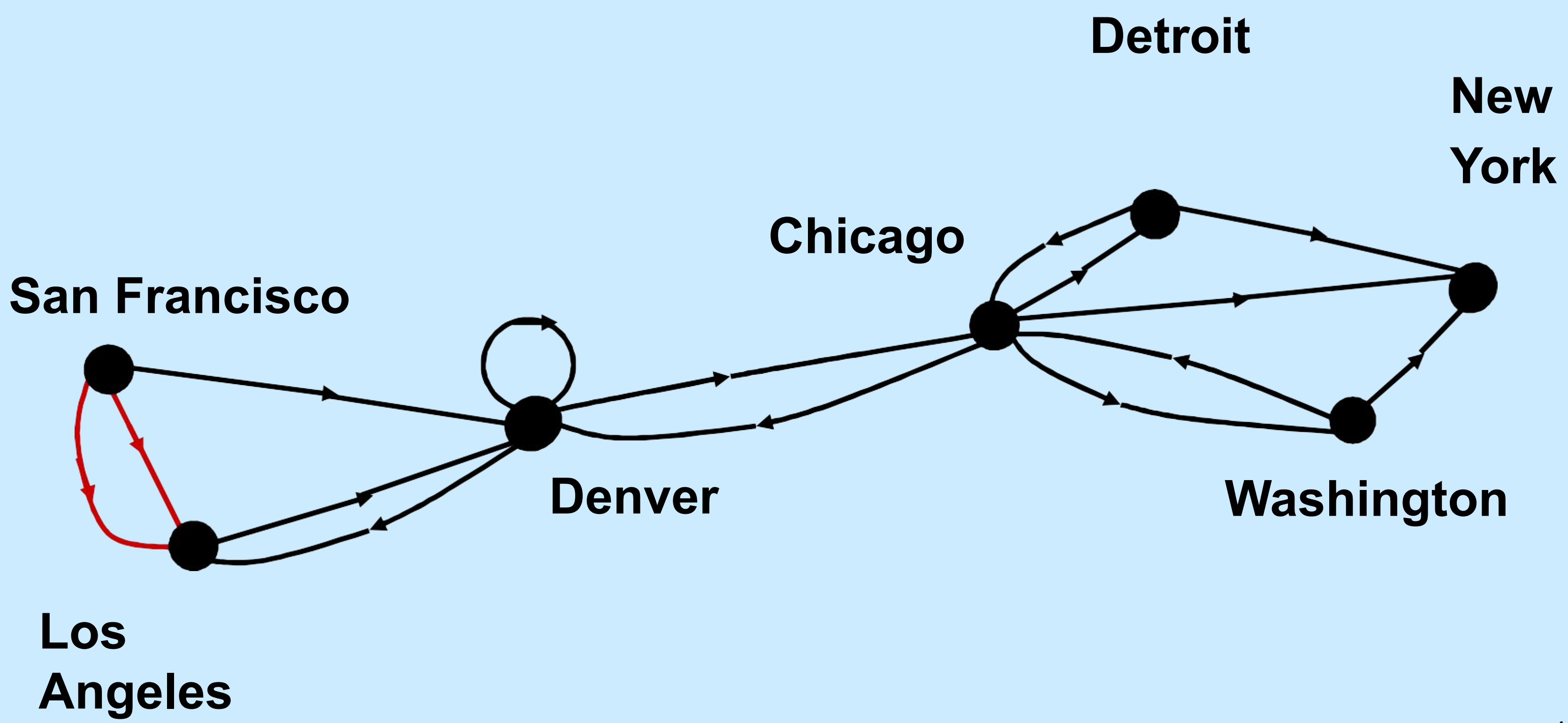


A Directed Multigraph

Definition 5. In a directed multigraph *G = (V, E)* the edges are ordered pairs of (not necessarily distinct) vertices, and in addition there may be multiple edges.

A Directed Multigraph

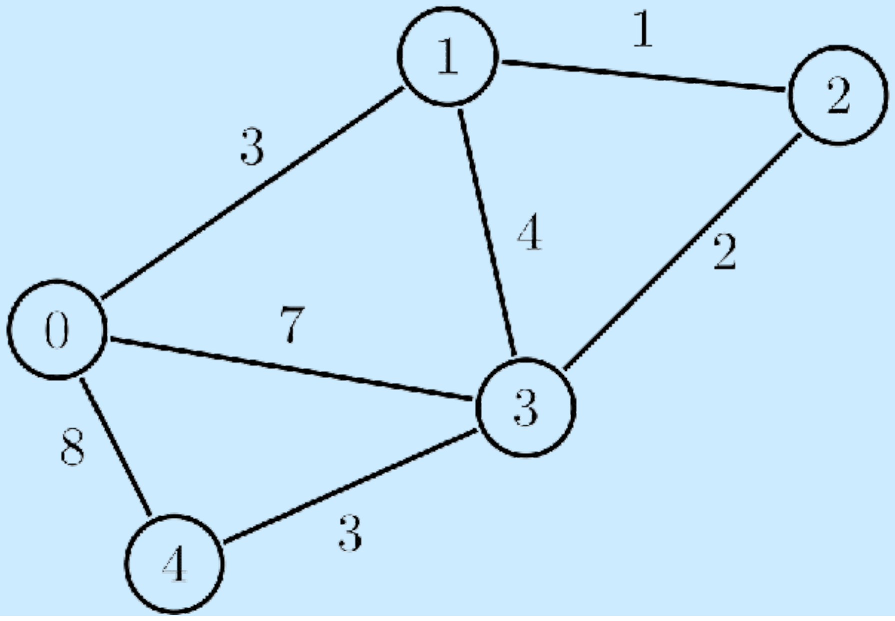
THERE MAY BE SEVERAL ONE-WAY LINES IN THE SAME DIRECTION FROM ONE COMPUTER TO ANOTHER IN THE NETWORK.

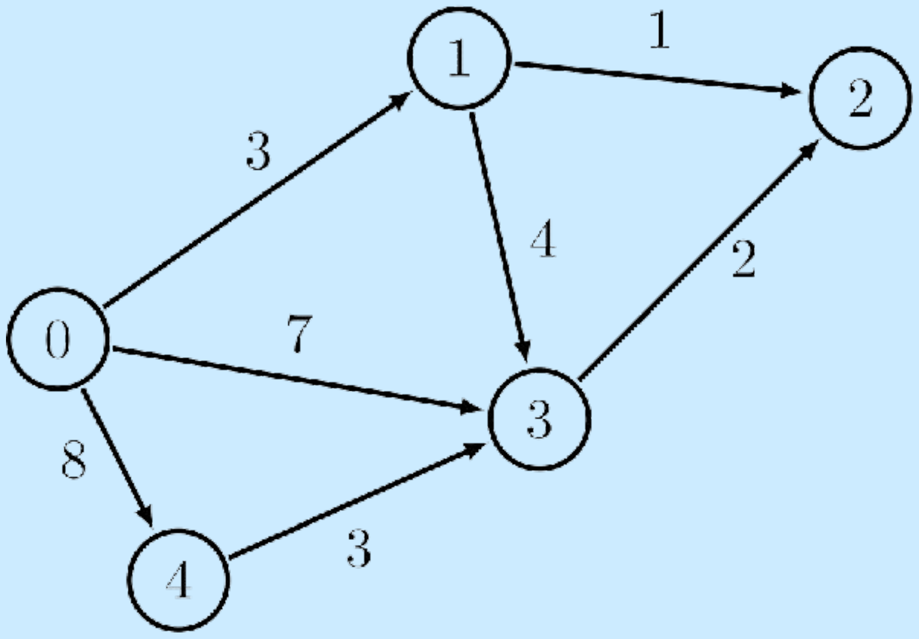


Weighted Graph

In a **weighted graph**, each edge has a **weight** or **cost**

- Usually, the edge weights are nonnegative integers.
- Some graphs allow negative weights





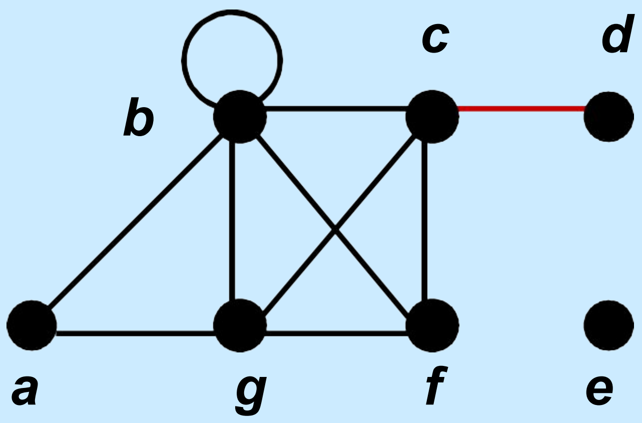
Adjacent Vertices (Neighbors)

Definition 1. Two vertices, *u* and *v* in an undirected graph G are called adjacent (or neighbors) in G, if {*u, v*} is an edge of G.

An edge e connecting *u* and *v* is called incident with vertices *u* and *v*, or is said to connect *u* and *v*. The vertices *u* and *v* are called endpoints of edge {*u, v*}.

Degree of a vertex

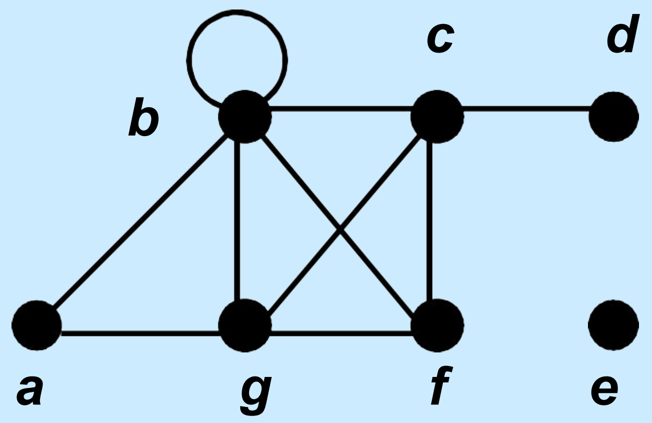
Definition 1. The **degree of a vertex** in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.



deg(d) = 1

Degree of a vertex

Definition 1. The degree of a vertex in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.

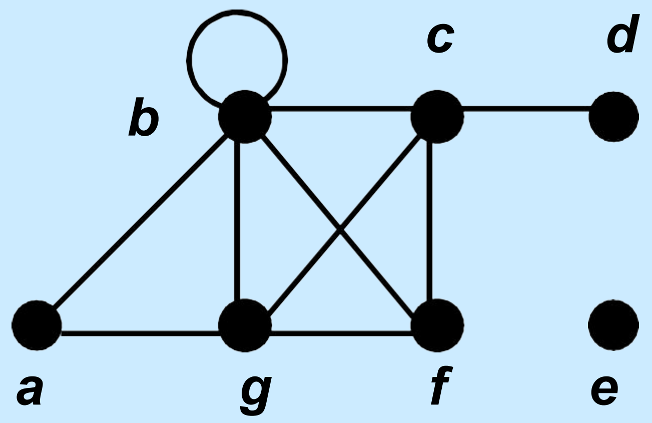


deg(e) = 0

Degree of a vertex

Definition 1. The **degree of a vertex** in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.

deg(b) = 6

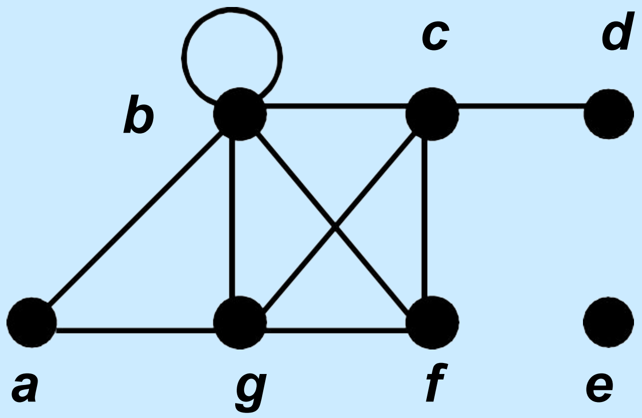


Degree of a vertex

Find the degree of all the other vertices.

deg(*a*)   deg(c)   deg(f)   deg(*g*)

deg(*b*) = 6



deg(*d*) = 1

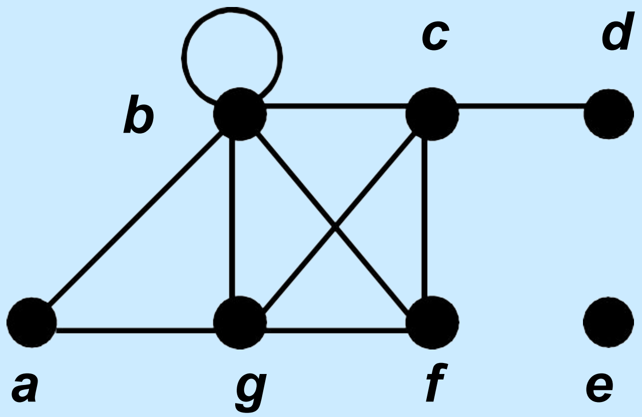
deg(e) = 0

Degree of a vertex

Find the degree of all the other vertices.

deg(*a*) = 2 deg(c) = 4 deg(f) = 3 deg(*g*) = 4

deg(*b*) = 6



deg(*d*) = 1

deg(e) = 0

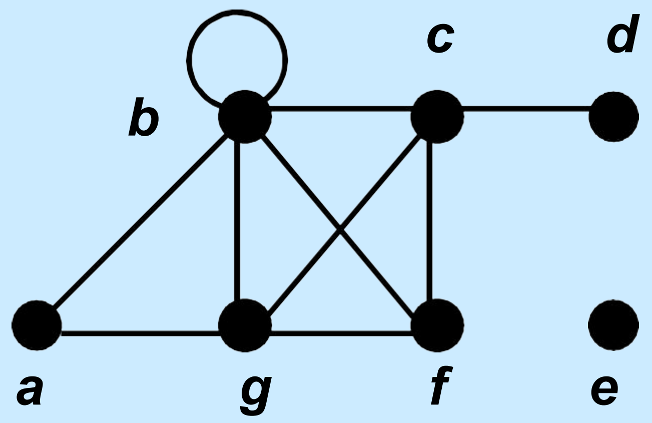
Degree of a vertex

Find the degree of all the other vertices.

deg(*a*) = 2   deg(c) = 4   deg(f) = 3   deg(*g*) = 4

TOTAL of degrees = 2 + 4 + 3 + 4 + 6 + 1 + 0 = 20

deg(*b*) = 6



deg(*d*) = 1

deg(e) = 0

Degree of a vertex

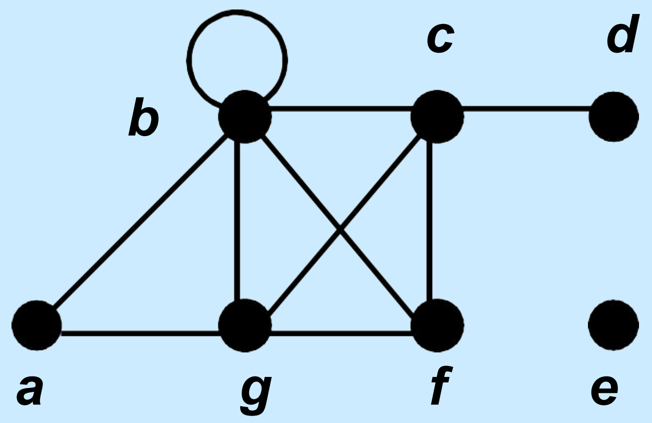
Find the degree of all the other vertices.

deg(*a*) = 2 deg(c) = 4 deg(f) = 3 deg(*g*) = 4

TOTAL of degrees = 2 + 4 + 3 + 4 + 6 + 1 + 0 = 20

TOTAL NUMBER OF EDGES = 10

deg(*b*) = 6

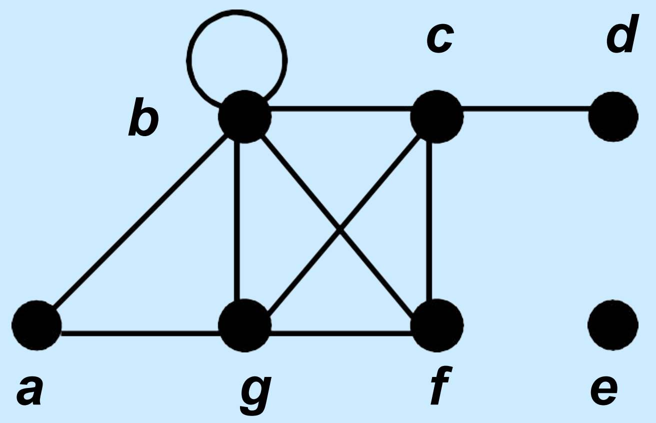


deg(*d*) = 1

deg(e) = 0

Degree of a vertex

An isolated vertex is a vertex of a graph that has no edges, a vertex with degree zero.



Handshaking Theorem

Theorem 1. **Let $G = (V, E)$ be an undirected graph G with e edges. Then**

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

"The sum of the degrees over all the vertices equals twice the number of edges."

NOTE: This applies even if multiple edges and loops are present.

Handshaking Theorem

Let $G = (V, E)$ be an directed graph Then

$$\sum_{v \in V} \deg^{-}(v) = \sum \deg^{+}(v) = |E|$$

"The sum of the in-degrees of all vertices is equal to the sum of the out-degrees of all vertices."

Graph Terminology

- **Example:** How many edges are there in a graph with 10 vertices, each of degree 6?

Graph Terminology

- **Example:** How many edges are there in a graph with 10 vertices, each of degree 6?
- **Solution:** The sum of the degrees of the vertices is $6 \cdot 10 = 60$. According to the Handshaking Theorem, it follows that $2e = 60$, so there are 30 edges.

Graph Terminology

- Prove that an undirected graph has an even number of vertices of odd degree.

- Prove that any simple graph with at least two vertices contains two vertices that have the same degree.

-

Graph Terminology

- An undirected graph has an even number of vertices of odd degree.
- Let V1 and V2 be the set of vertices of even and odd degrees, respectively (Thus $V1 \cap V2 = \emptyset$, and $V1 \cup V2 = V$).
- Then by Handshaking theorem
  - $2|E| = \sum_{v \in V} deg(v) = \sum_{v \in V1} deg(v) + \sum_{v \in V2} deg(v)$
- Since both $2|E|$ and $\sum_{v \in V1} deg(v)$ are even,
- $\sum_{v \in V2} deg(v)$ must be even.
- Since deg(v) if odd for all $v \in V2$, $|V2|$ must be even. QED
-

Graph Representations

Which data structure is "best" can depend on:

- properties of the graph (e.g., dense versus sparse)
- the common queries about the graph ("is (u ,v) an edge?" vs "what are the neighbors of node u?")

Two representations: Adjacency Matrix and Adjacency List

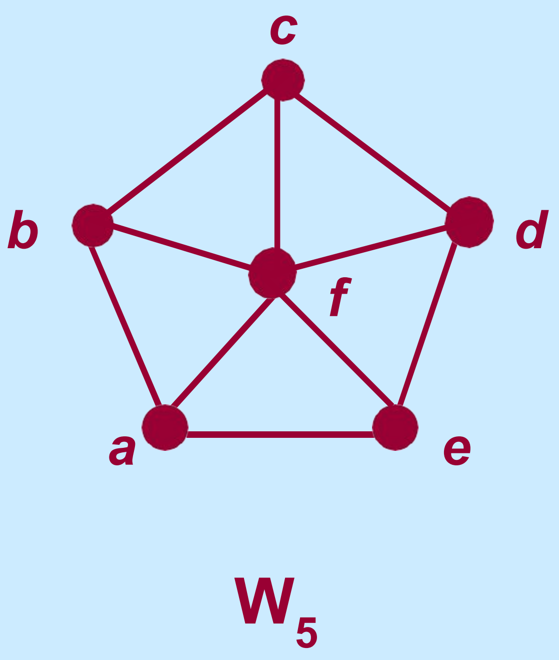
- Different trade-offs, particularly time versus space

Adjacency Matrix

A simple graph $G = (V, E)$ with n vertices can be represented by its adjacency matrix, A, where entry a_{ij} in row i and column j is

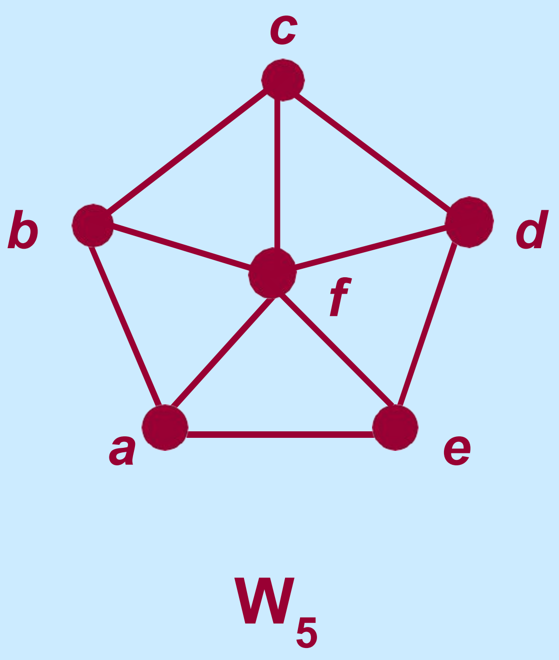
$$a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge in G,} \\ 0 & \text{otherwise.} \end{cases}$$

Finding the adjacency matrix



This graph has 6 vertices a, b, c, d, e, f. We can arrange them in that order.

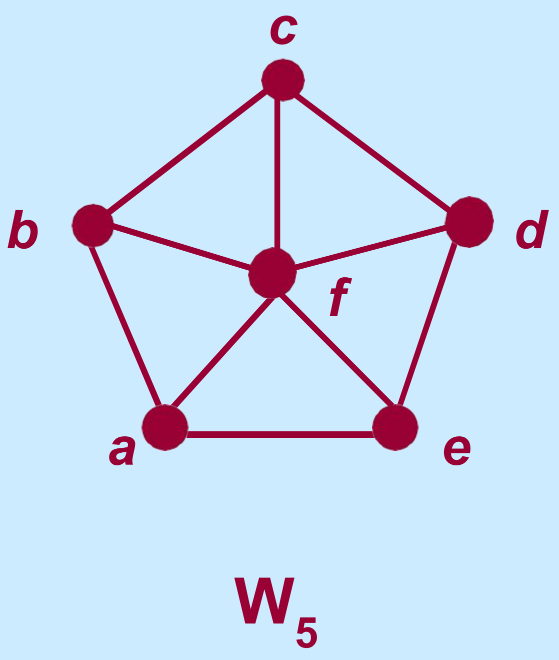
Finding the adjacency matrix



$$
\begin{array}{c|cccccc}
 & \multicolumn{6}{c}{\text{TO}} \\
\text{FROM} & a & b & c & d & e & f \\
\hline
a & 0 & 1 & 0 & 0 & 1 & 1 \\
b & & & & & & \\
c & & & & & & \\
d & & & & & & \\
e & & & & & & \\
f & & & & & &
\end{array}
$$

There are edges from a to b, from a to e, and from a to f

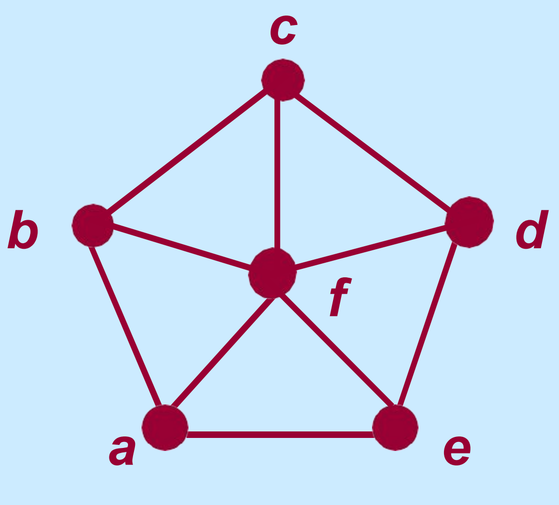
Finding the adjacency matrix



$$\begin{array}{c|cccccc} & \multicolumn{6}{c}{\text{TO}} \\ \text{FROM} & a & b & c & d & e & f \\ \hline a & 0 & 1 & 0 & 0 & 1 & 1 \\ b & 1 & 0 & 1 & 0 & 0 & 1 \\ c & & & & & & \\ d & & & & & & \\ e & & & & & & \\ f & & & & & & \end{array}$$

There are edges from b to a, from b to c, and from b to f

Finding the adjacency matrix



W$_5$

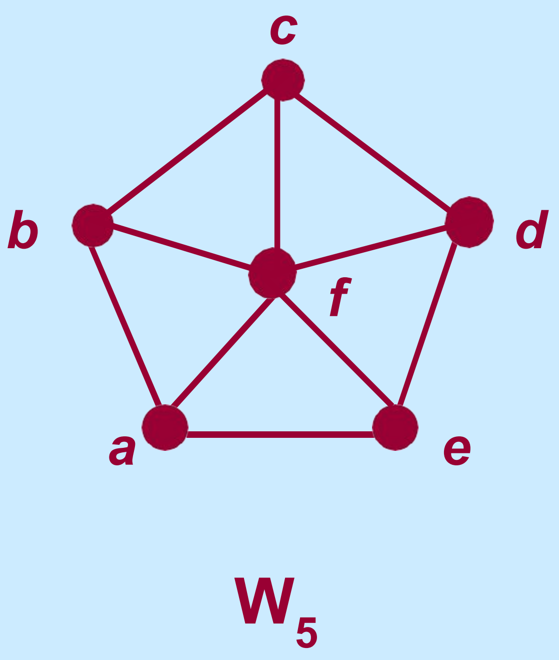
TO

FROM

$$\begin{array}{c|cccccc} & a & b & c & d & e & f \\ \hline a & 0 & 1 & 0 & 0 & 1 & 1 \\ b & 1 & 0 & 1 & 0 & 0 & 1 \\ c & 0 & 1 & 0 & 1 & 0 & 1 \\ d & & & & & & \\ e & & & & & & \\ f & & & & & & \end{array}$$

There are edges from c to b, from c to d, and from c to f

Finding the adjacency matrix



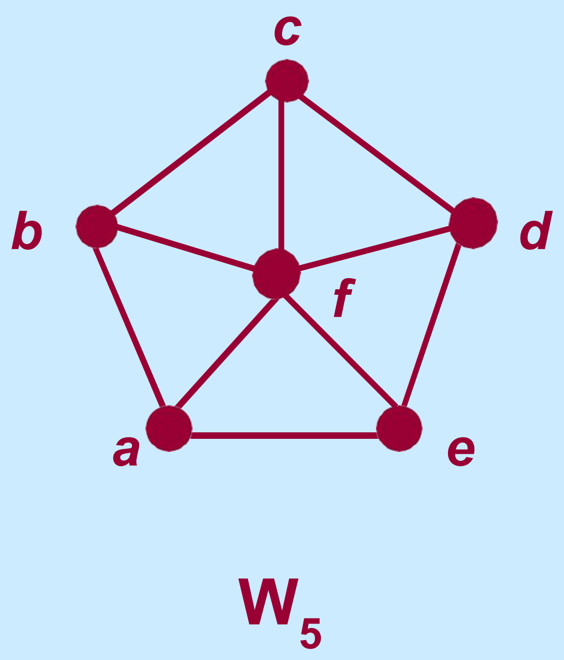
TO

FROM

$$\begin{array}{c|cccccc} & a & b & c & d & e & f \\ \hline a & 0 & 1 & 0 & 0 & 1 & 1 \\ b & 1 & 0 & 1 & 0 & 0 & 1 \\ c & 0 & 1 & 0 & 1 & 0 & 1 \\ d & & & & & & \\ e & & & & & & \\ f & & & & & & \end{array}$$

COMPLETE THE ADJACENCY MATRIX . . .

Finding the adjacency matrix



TO

FROM

$$\begin{array}{c|cccccc|} & a & b & c & d & e & f \\ \hline a & 0 & 1 & 0 & 0 & 1 & 1 \\ b & 1 & 0 & 1 & 0 & 0 & 1 \\ c & 0 & 1 & 0 & 1 & 0 & 1 \\ d & 0 & 0 & 1 & 0 & 1 & 1 \\ e & 1 & 0 & 0 & 1 & 0 & 1 \\ f & \underline{1} & 1 & 1 & 1 & 1 & 0 \end{array}$$

Notice that this matrix is symmetric. That is $a_{ij} = a_{ji}$ Why?

Adjacency Matrix

A directed graph *G* = (*V*, *E*) with n vertices can be represented by its adjacency matrix, A, where entry a_{ij} in row i and column j is

$$a_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \text{ is an edge in } G, \\ 0 & \text{otherwise.} \end{cases}$$

Adjacency Matrix Properties

Running time to:

- Get a vertex's out-edges:
- Get a vertex's in-edges:
- Decide if some edge exists:
- Insert an edge:
- Delete an edge:

$$
\begin{array}{c|cccccc|}
 & a & b & c & d & e & f \\
a & 0 & 1 & 0 & 0 & 1 & 1 \\
b & 1 & 0 & 1 & 0 & 0 & 1 \\
c & 0 & 1 & 0 & 1 & 0 & 1 \\
d & 0 & 0 & 1 & 0 & 1 & 1 \\
e & 1 & 0 & 0 & 1 & 0 & 1 \\
f & \underline{1} & 1 & 1 & 1 & 1 & 0
\end{array}
$$

Space requirements:

Adjacency Matrix Properties

Running time to:

- Get a vertex's out-edges: O(|V|)
- Get a vertex's in-edges: O(|V|)
- Decide if some edge exists: O(1)
- **Insert an edge:** O(1)
- **Delete an edge:** O(1)

Space requirements: O($|V|^2$)

Best for sparse or dense graphs? ***dense***

Adjacency Matrix Properties

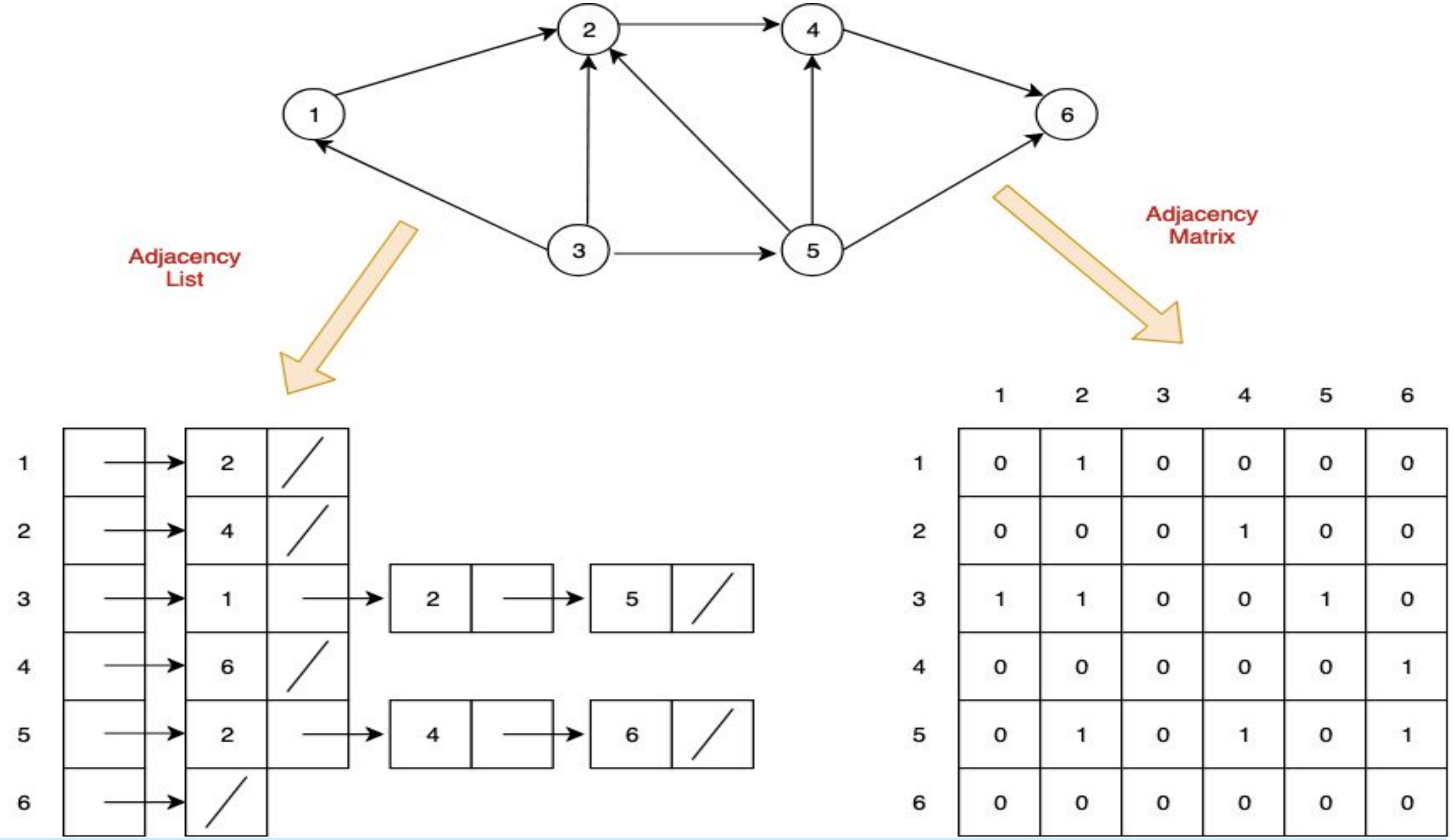
How can we adapt the representation for weighted graphs?

- ❖ Store the weight in each cell
- ❖ Need some value to represent ‘not an edge’
  - ➢ 0, -1, or some other value based on how you are using the graph
  - ➢ Might need to be a separate field if no restrictions on weights

Adjacency List

An array of length |V| in which each entry stores a list of all adjacent vertices (e.g., linked list)

Adjacency List



	1	2	3	4	5	6
1	0	1	0	0	0	0
2	0	0	0	1	0	0
3	1	1	0	0	1	0
4	0	0	0	0	0	1
5	0	1	0	1	0	1
6	0	0	0	0	0	0

Adjacency List Properties

Running time to:

Get a vertex’s out-edges:

Get a vertex’s in-edges:

Decide if some edge exists:

Insert an edge:

Delete an edge:

Space requirements:

Best for sparse or dense graphs?

Adjacency List Properties

Running time to:

- **Get a vertex's out-edges:**
  - **O(d) where d is out-degree of vertex**
- Get a vertex's in-edges:
  - O(|E|) (could keep a second adjacency list for this!)
- **Decide if some edge exists:**
  - **O(d) where d is out-degree of source**
- **Insert an edge:**
  - **O(1) (unless you need to check if it's already there)**
- **Delete an edge:**
  - **O(d) where d is out-degree of source**

Space requirements: O(|V|+|E|)

Best for sparse or dense graphs? ***sparse***

Undirected Graphs

Adjacency lists also work well for undirected graphs with one caveat

Put each edge in two lists to support efficient "get all neighbors"

Which is better?

- Graphs are often sparse
  - Streets form grids
  - Airlines rarely fly to all cities
- Adjacency lists should generally be your default choice
  - Slower performance compensated by greater space savings

Graph Representations

Is there any other graph representation?

- **Incidence Matrix**
- **Edge list**

Incidence Matrix

Let G=(V,E) be an undirected graph. Suppose that v_1, v_2,…, v_n are the vertices and e_1,e_2,…,e_m are the edges of G. The incidence matrix with respect to this ordering of V and E is the n×m matrix M = [m_{ij}], where

$$m_{ij} = \begin{cases} 1 & \text{if edge } e_j \text{ is incident with vi} \\ 0 & \text{otherwise.} \end{cases}$$

Path of Length n

Definition 1. A **path of length n** from *u* to *v* in an undirected graph is a sequence of edges $e_1, e_2, \ldots, e_n$ of the graph such that edge e_1 has endpoints x_o and x_1, edge e_2 has endpoints x_1 and x_2,

. . .

and edge e_n has endpoints x_{n-1} and x_n, where $x_0 = u$ and $x_n = v$.

Path of Length n

A path of length n from *u* to *v* in an directed graph is a sequence of edges

$e_1, e_2, \ldots, e_n$ of the graph such that edge e_1

is associated with (x_0, x_1)

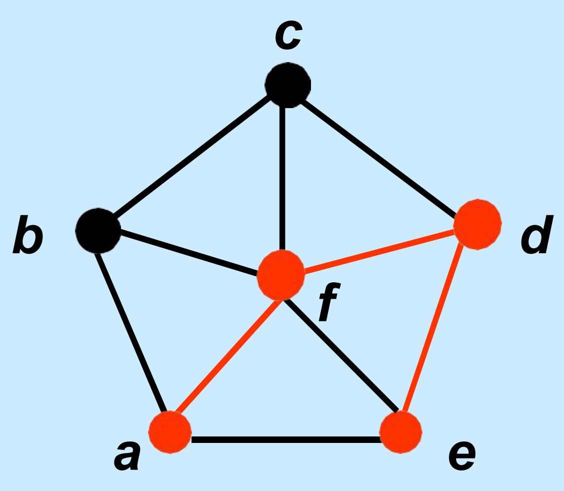
edge e_2 is associated with (x_1, x_2)

. . .

and edge e_n is associated with (x_{n-1}, x_n)

where $x_0 = u$ and $x_n = v$.

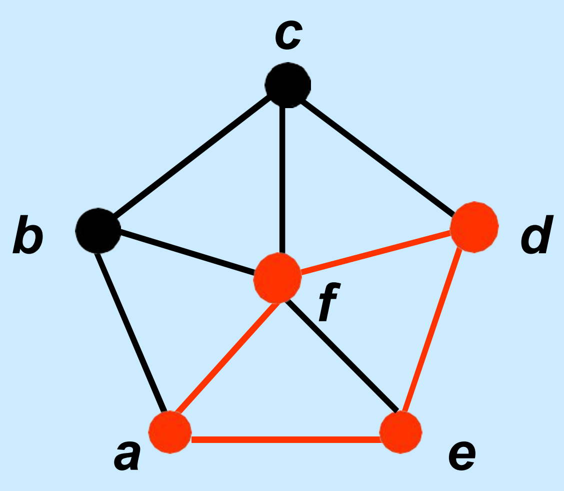
One path from *a* to e



W$_5$

This path passes through vertices f and d in that order.

One path from *a* to *a*



W_5

This path passes through vertices f, d, e, in that order.

It has **length** 4.

It is a **circuit** because it begins and ends at the same vertex.

It is called **simple** because it does not contain the same edge more than once.

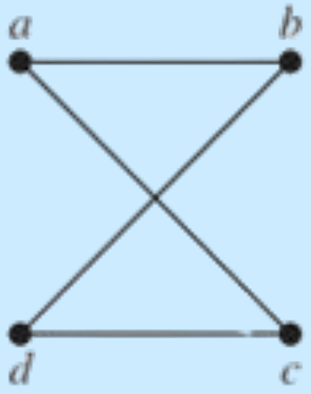
Path of Length n

How to count the number of path of length n?

Path of Length n

How to count the number of path of length n?

Solution: Given the adjacency matrix A of the graph, the number of path of length n from vertex i to vertex j is $A^n[i][j]$

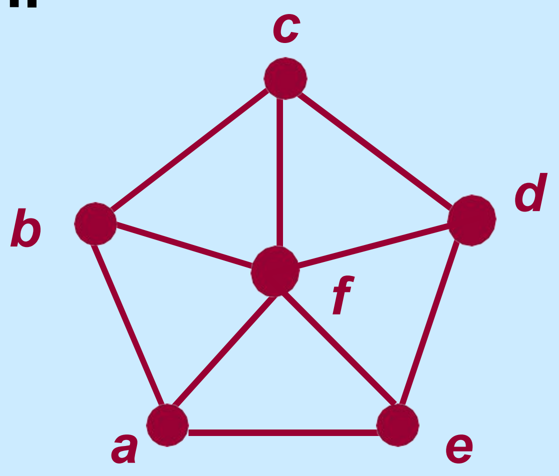


$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}. \quad \mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix},$$

Connected graph

Definition 3. An undirected graph is called **connected** if there is a path between every pair of distinct vertices of the graph.

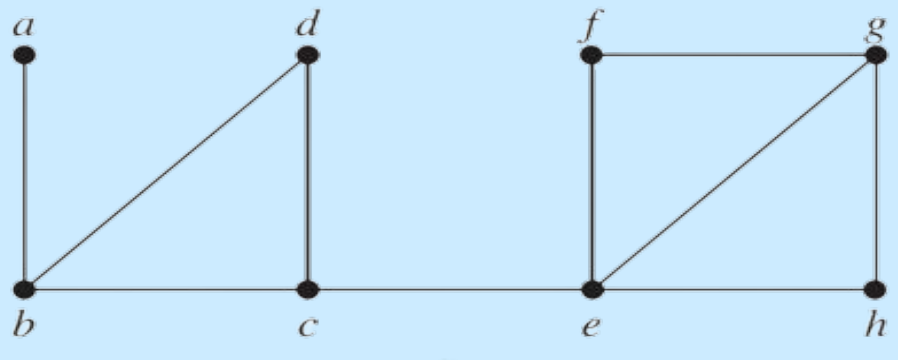
IS THIS GRAPH CONNECTED?



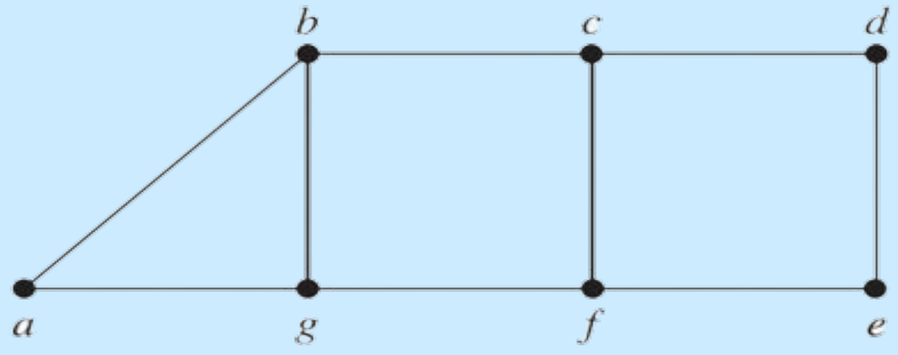
W_5

Connected Graph

There is a simple path between every pair of distinct vertices of a connected undirected graph.



G_1



G_3

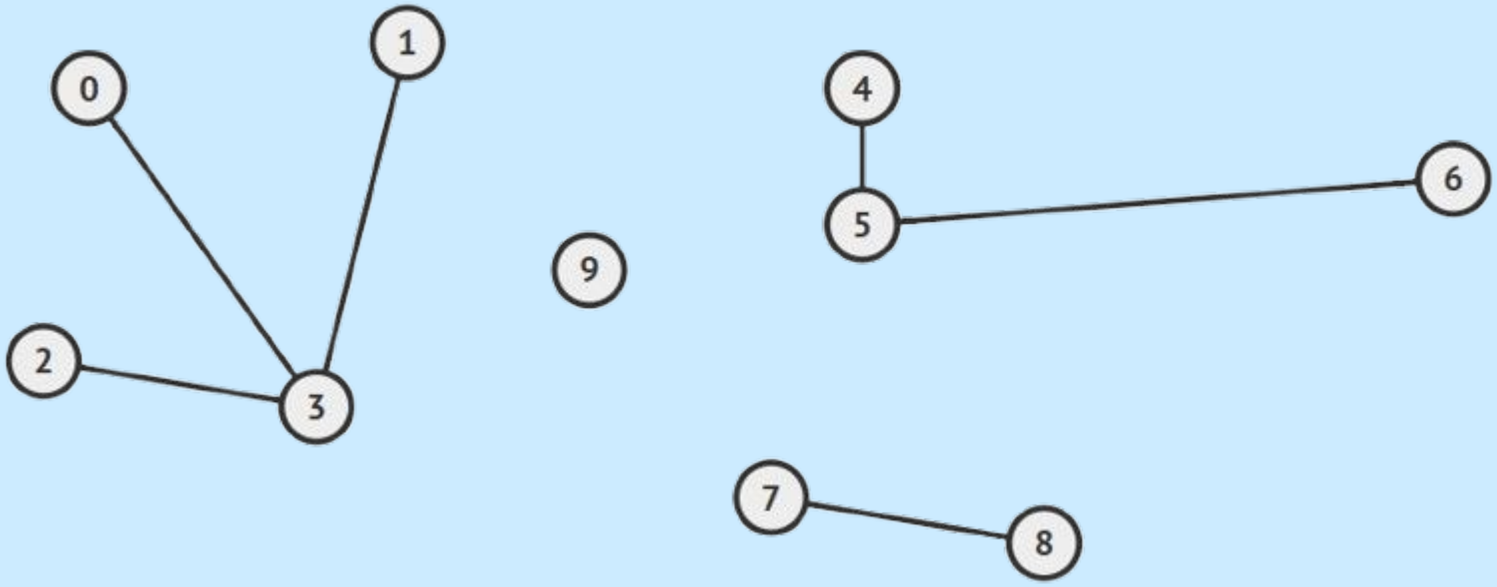
Connected Components

A connected component of a graph G is a connected subgraph of G that is not a proper subgraph of another connected subgraph of G. That is, a connected component of a graph G is a maximal connected subgraph of G.

A graph G that is not connected has two or more connected components that are disjoint and have G as their union

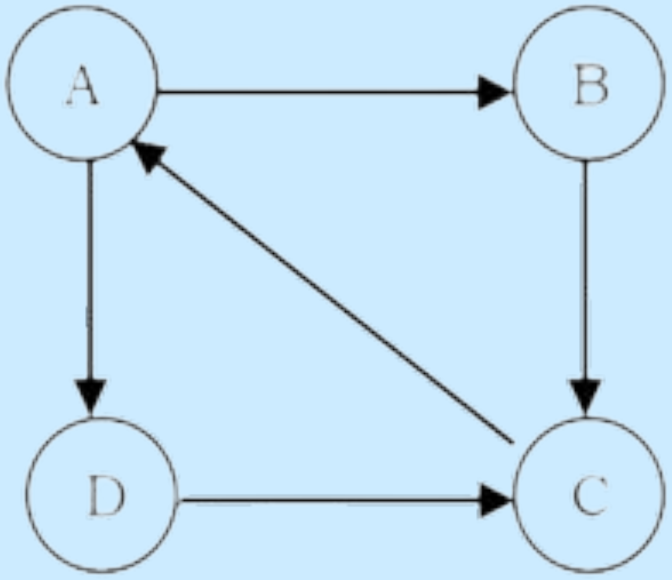
Connected Components

How many connected components are there in the following graph?



Connectedness in Directed Graphs

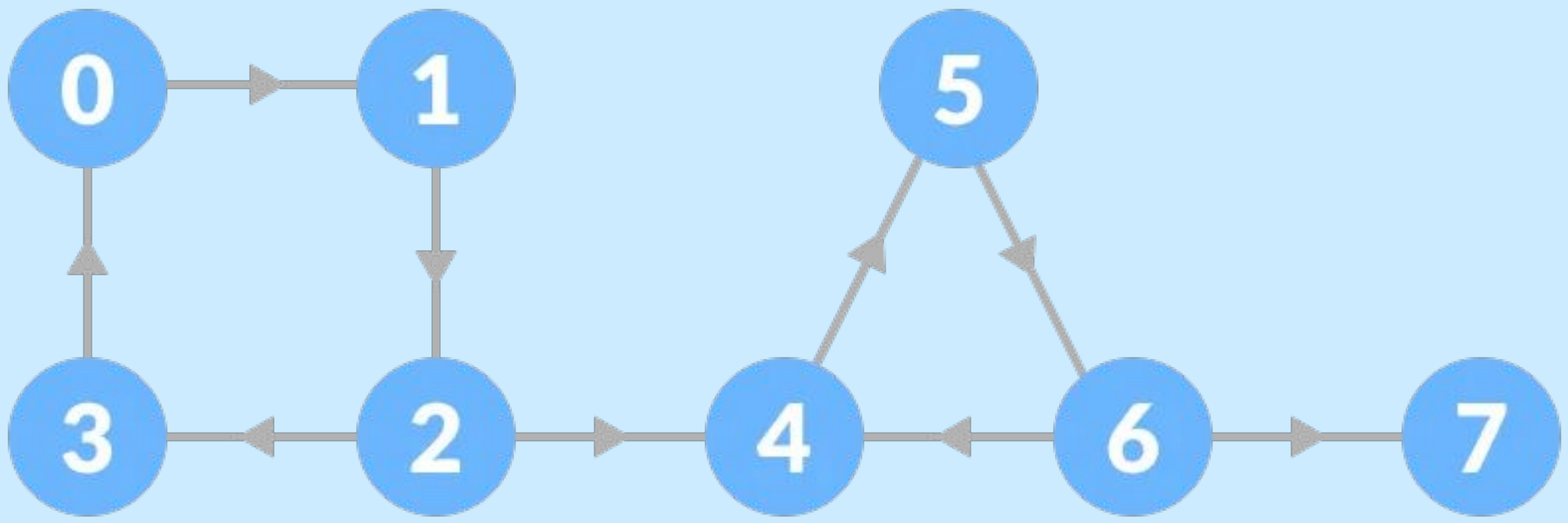
A directed graph is strongly connected if there is a path from a to b and from b to a whenevera and b are vertices in the graph



Strongly connected

Connectedness in Directed Graphs

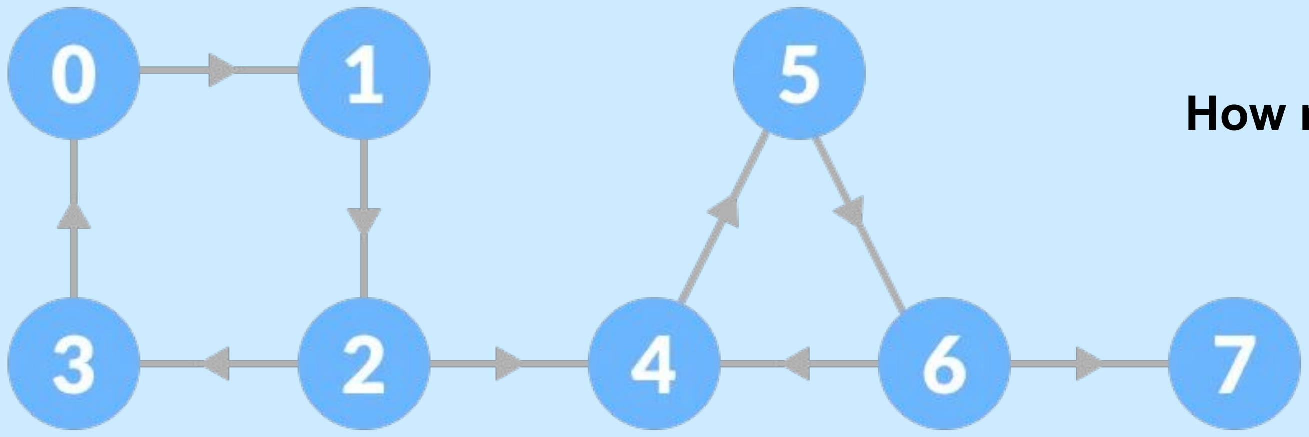
A directed graph is weakly connected if there is a path between every two vertices in the underlying undirected graph.



Not strongly connected but weakly connected

STRONGLY CONNECTED COMPONENTS

The subgraphs of a directed graph G that are strongly connected but not contained in larger strongly connected subgraphs, that is, the maximal strongly connected subgraphs, are called the strongly connected components or strong components of G

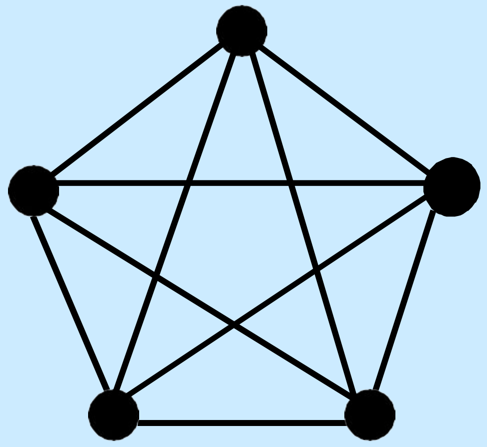


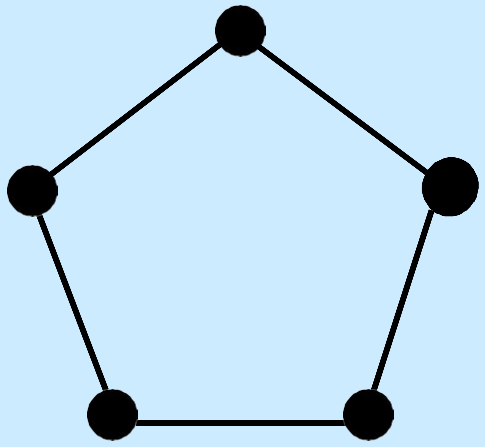
How many?

Subgraph

Definition 6. **A subgraph of a graph $G = (V, E)$ is a graph $H = (W, F)$ where $W \subseteq V$ and $F \subseteq E$.**

C_5 is a subgraph of K_5

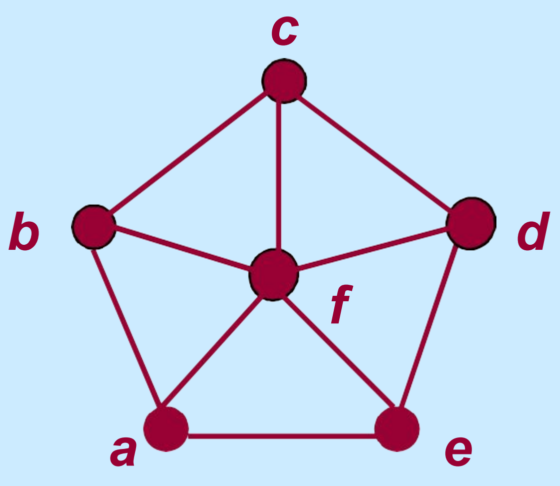

K_5


C_5

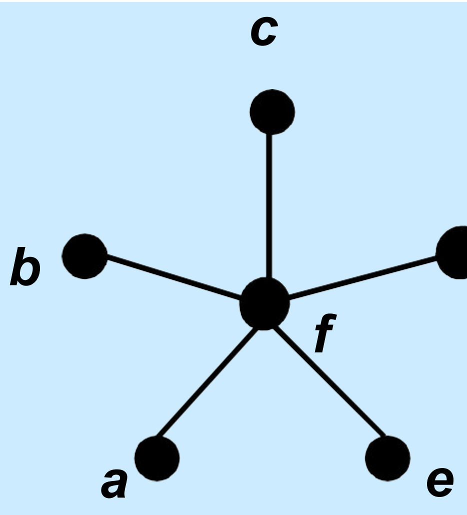
Union

Definition 7. The **union** of 2 simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the simple graph with vertex set $V = V_1 \cup V_2$ and edge set $E = E_1 \cup E_2$. The union is denoted by $G_1 \cup G_2$.

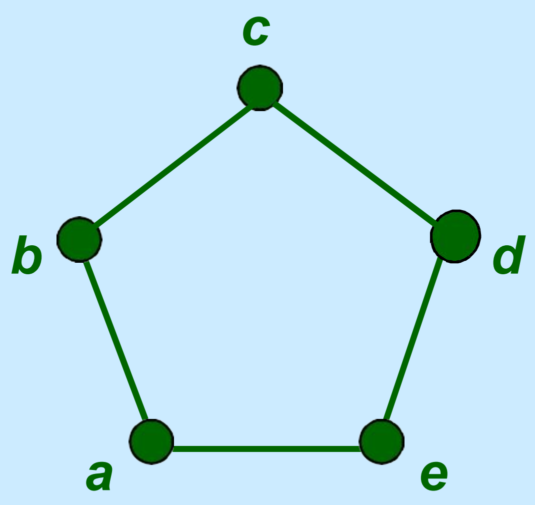
W_5 is the union of S_5 and C_5



W_5



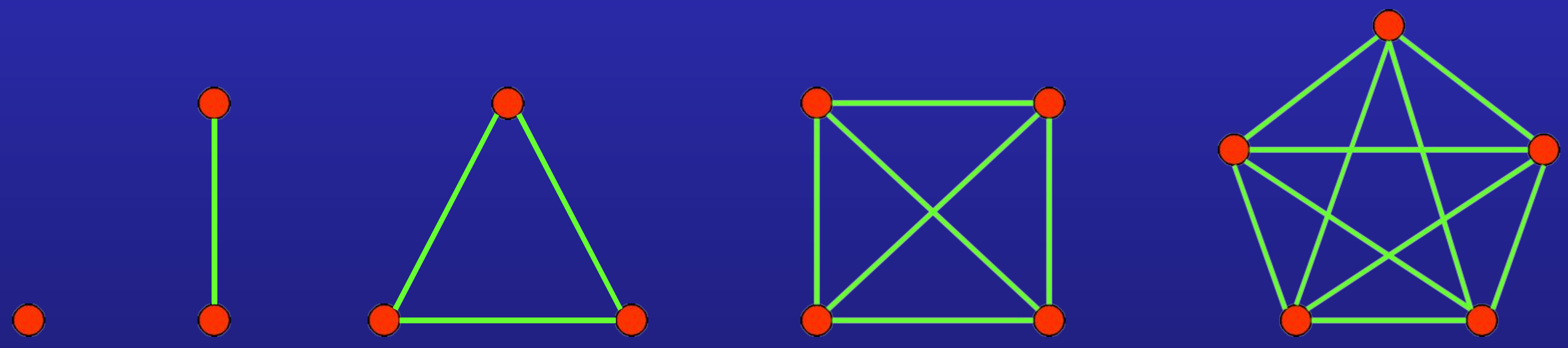
S_5



C_5

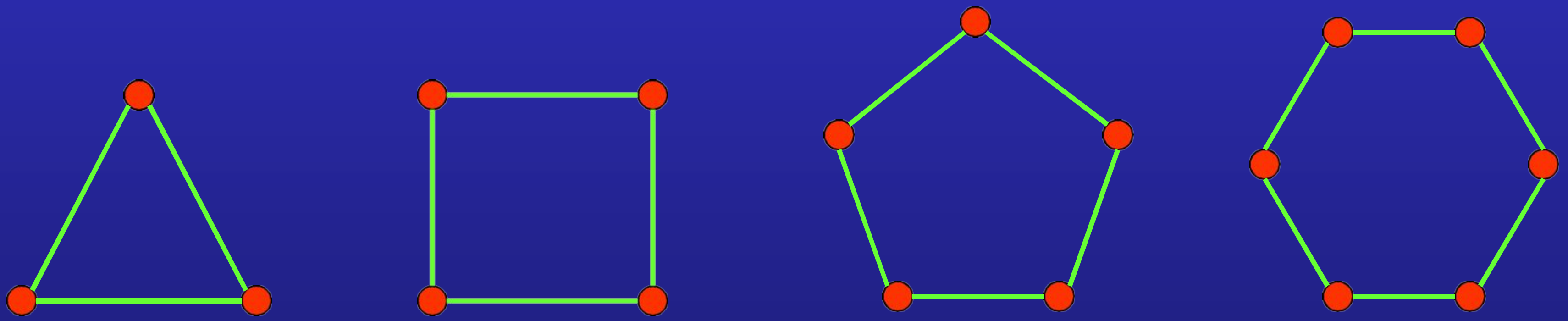
Special Graphs

Definition: The **complete graph** on n vertices, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices.



Special Graphs

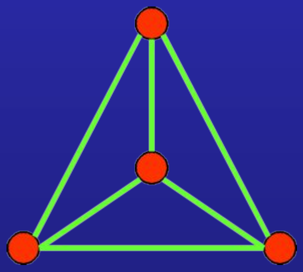
Definition: The **cycle** C_n, $n \geq 3$, consists of n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$.



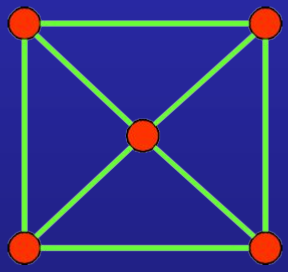
C_3 C_4 C_5 C_6

Special Graphs

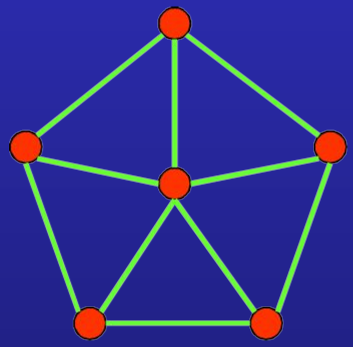
Definition: We obtain the **wheel** W_n when we add an additional vertex to the cycle C_n, for $n \geq 3$, and connect this new vertex to each of the n vertices in C_n by adding new edges.



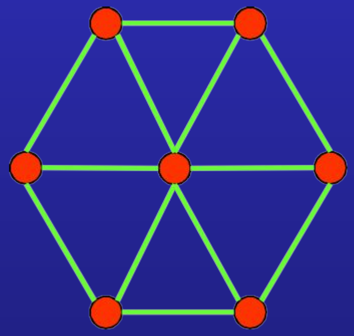
W_3



W_4



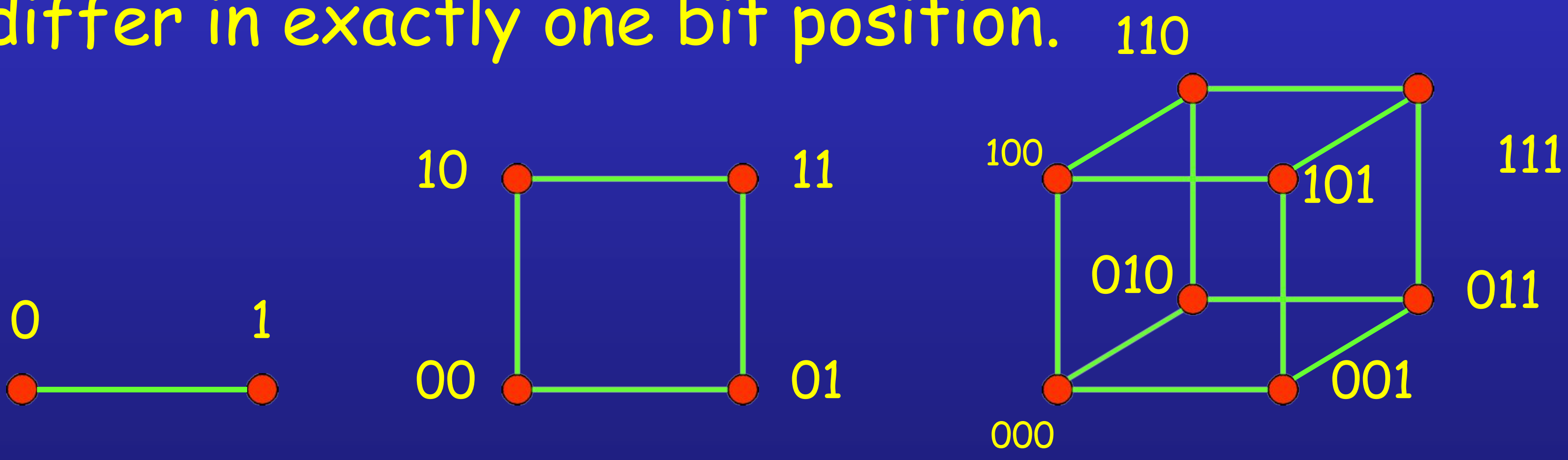
W_5



W_6

Special Graphs

Definition: The **n-cube**, denoted by Q_n, is the graph that has vertices representing the 2^n bit strings of length n. Two vertices are adjacent if and only if the bit strings that they represent differ in exactly one bit position.



Q_1 Q_2 Q_3

Special Graphs

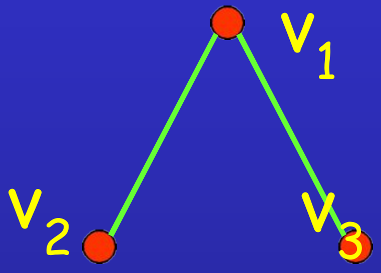
Definition: A simple graph is called **bipartite** if its vertex set V can be partitioned into two disjoint nonempty sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 with a vertex in V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2).

For example, consider a graph that represents each person in a village by a vertex and each marriage by an edge.

This graph is **bipartite**, because each edge connects a vertex in the **subset of males** with a vertex in the **subset of females** (if we think of traditional marriages).

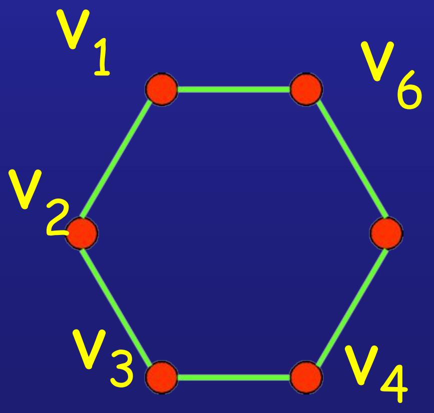
Special Graphs

Example I: Is C_3 bipartite?

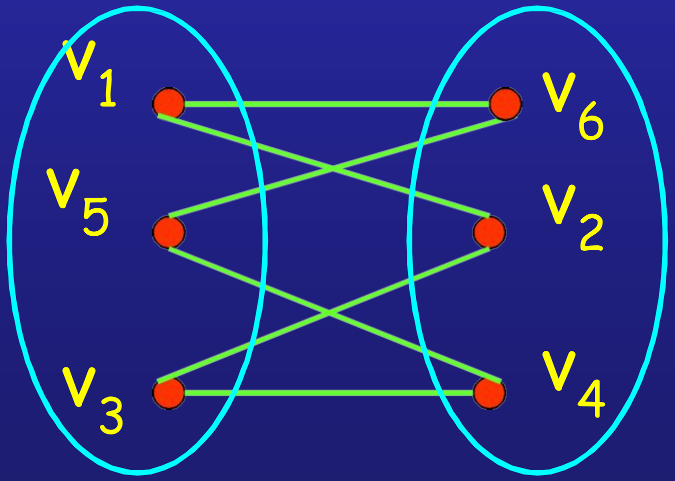


No, because there is no way to partition the vertices into two sets so that there are no edges with both endpoints in the same set.

Example II: Is C_6 bipartite?

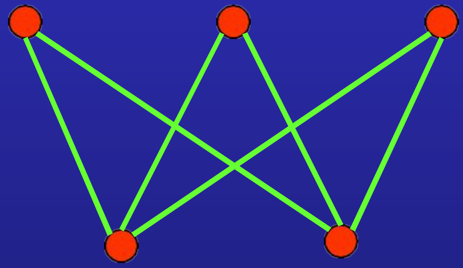


Yes, because we can display C_6 like this:

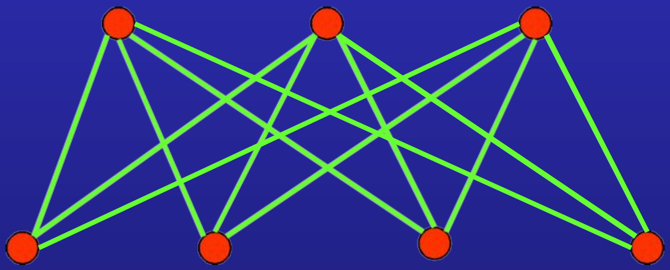


Special Graphs

Definition: The **complete bipartite graph** $K_{m,n}$ is the graph that has its vertex set partitioned into two subsets of m and n vertices, respectively. Two vertices are connected if and only if they are in different subsets.



$K_{3,2}$



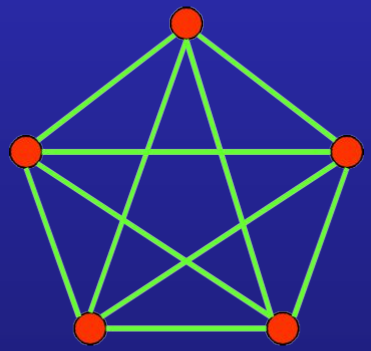
$K_{3,4}$

Operations on Graphs

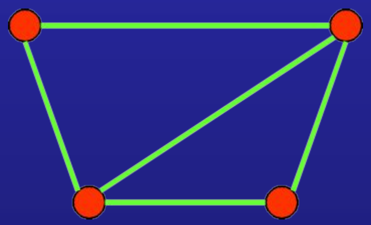
Definition: A **subgraph** of a graph $G = (V, E)$ is a graph $H = (W, F)$ where $W \subseteq V$ and $F \subseteq E$.

Note: Of course, H is a valid graph, so we cannot remove any endpoints of remaining edges when creating H.

Example:



K_5



subgraph of K_5

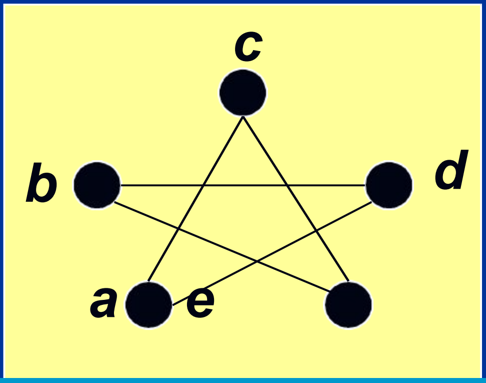
Euler Circuit

- A Euler Circuit in a graph G is a simple circuit that includes all edges and vertices of G.
- A Euler Path in a graph G is a simple path that includes all edges and vertices of G.

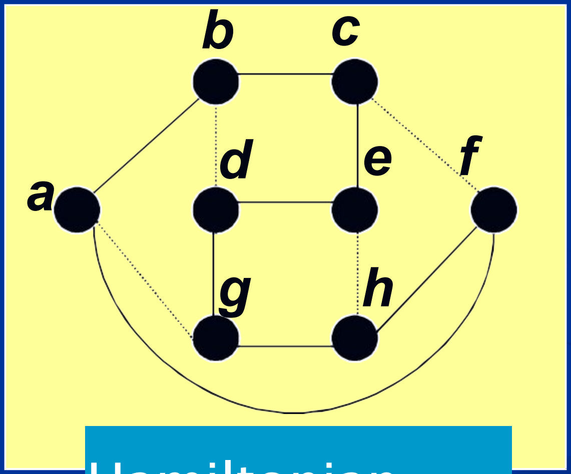
Visit each edge once

Hamiltonian circuit

- … a circuit where each vertex in G is used exactly once



Hamiltonian & Euler circuit



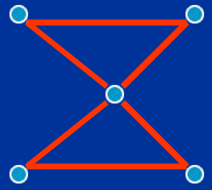
Hamiltonian circuit

Visit each vertex once

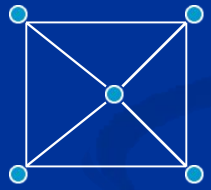
Euler cycle and paths

Euler path: a path traversing all the edges of the graph exactly once

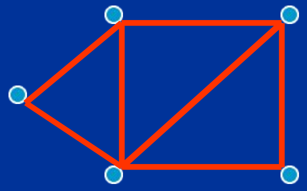
Euler cycle: a cycle traversing all the edges of the graph exactly once



Has Euler cycle



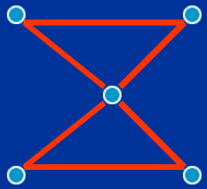
No Euler cycle no Euler path



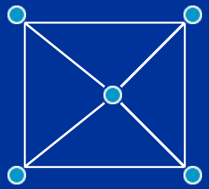
Has Euler path, but no Euler cycle

Necessary and sufficient conditions for Euler cycles

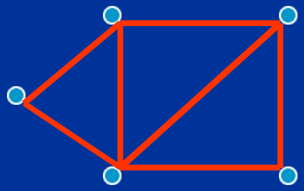
Theorem 1: A connected multigraph has an Euler cycle if and only if each of its vertices has even degree.



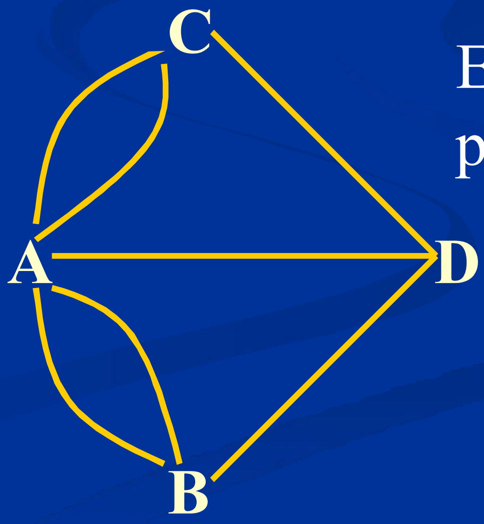
Has Euler cycle



No Euler cycle no Euler path



Has Euler path, but no Euler cycle



Euler cycle??? path???

Necessary condition for Euler cycle

Theorem 1: A connected multigraph has an Euler cycle if and only if each of its vertices has even degree.

Proof Sketch, PART 1:

- Assume the graph has an Euler cycle.
- Observe that every time the cycle passes through a vertex, it contributes 2 to the vertex's degree

(since the cycle enters via an edge incident with this vertex and leaves via another such edge)

Necessary and sufficient conditions for Euler paths

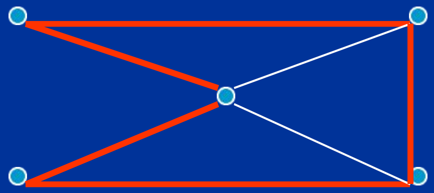
Theorem 2. A connected multigraph has an Euler path but not an Euler cycle if and only if it has exactly two vertices of odd degree.

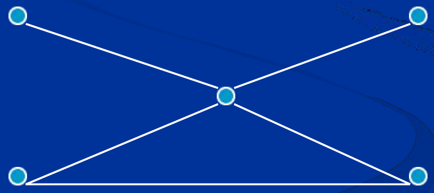
Hamilton paths and cycles

Hamilton cycle: visits every vertex of the graph exactly once before returning, as the last step, to the starting vertex.

Hamilton path: visits every vertex of the graph exactly once.

Examples:





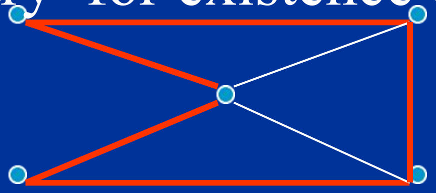
Hamilton paths and cycles (cont.)

No property is known to efficiently verify existence of a Hamilton cycle/path for general graphs. Moreover, the problem is known to be as difficult as the TSP (find the shortest H. cycle through n cities).

Dirac's Theorem: If G is a simple graph with $n \geq 3$ vertices such that the degree of every vertex is at least $n/2$, then G has a Hamilton cycle.

Ore's Theorem: If G is a simple graph with $n \geq 3$ vertices such that the $deg(u) + deg(v) \geq n$ for every pair of nonadjacent vertices u and v, then G has a Hamilton cycle.

Ore's condition is more general than Dirac's but neither is necessary for existence of a Hamilton cycle.

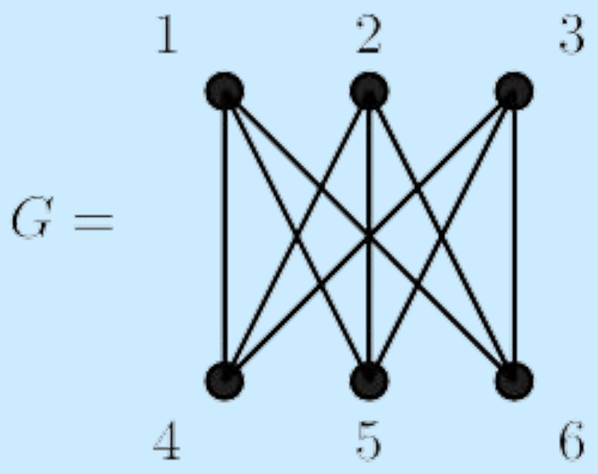


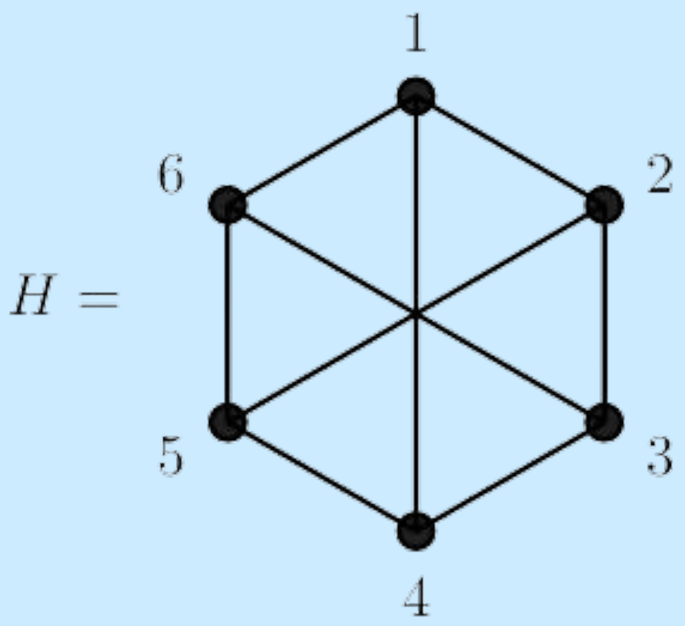
Graph Isomorphism

Graph Isomorphism

Two graphs G=(V,E) and H=(W,F) are isomorphic if there is a bijective function f: V → W such that for all *v*, *w* ∈ V:

$$\{v,w\} \in E \Leftrightarrow \{f(v),f(w)\} \in F$$





Variant for labeled graphs

Let G = (V,E), H=(W,F) be graphs with vertex labelings *l*: V $\rightarrow$ L, *l'*: W $\rightarrow$ L.

G and H are isomorphic labeled graphs, if there is a bijective function f: V $\rightarrow$ W such that

- For all $v, w \in$ V: $\{v,w\} \in$ E $\Leftrightarrow$ $\{f(v),f(w)\} \in$ F
- For all $v \in$ V: $l(v) = l'(f(v))$.

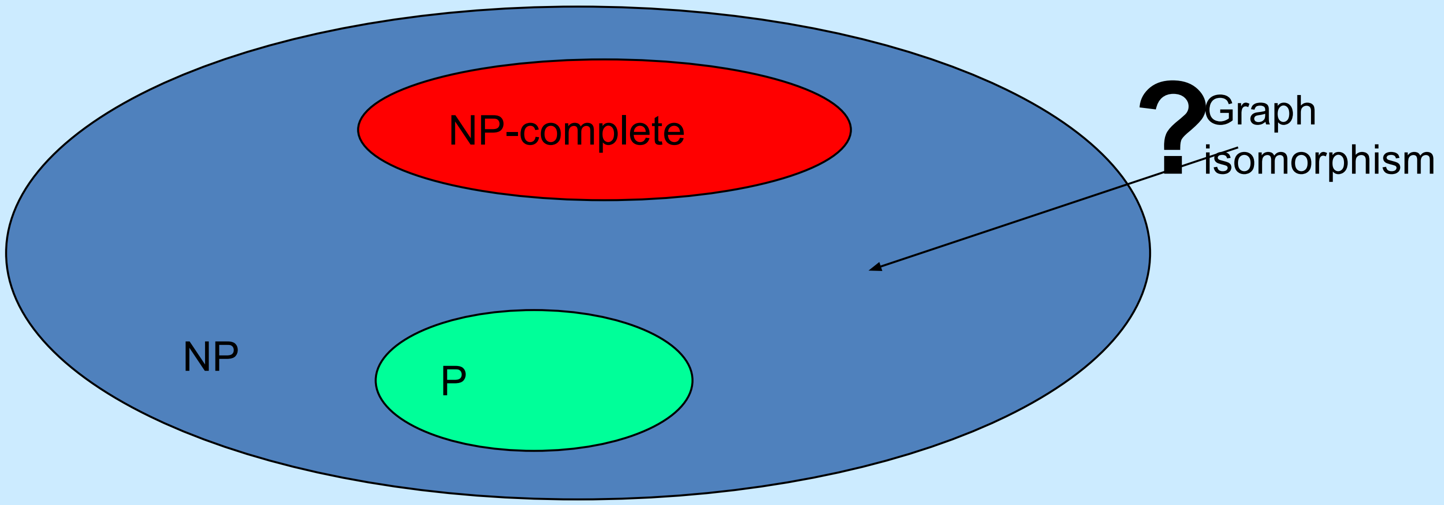
Complexity of graph isomorphism

Problem is in NP, but

No NP-completeness proof is known

No polynomial time algorithm is known

If P ≠ NP



Complexity of graph isomorphism

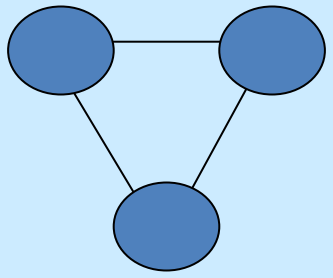
László Babai (2017) proposed an algorithm with complexity of **quasi-polynomial (not polynomial time)**

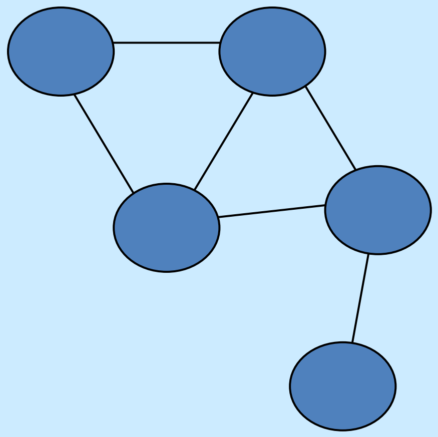
$2^{f(n)}$ *where* $f(n)$ *is* $O((\mathrm{lo}$

Isomorphism-complete

Problems are isomorphism-complete, if they are `equally hard' as graph isomorphism

- Isomorphism of bipartite graphs
- Isomorphism of labeled graphs
- Automorphism of graphs
  - Given: a graph G=(V,E)
  - Question: is there a non-trivial automorphism, i.e., a bijective function *f*: V → V, not the identity, with for all *v,w*∈V:
    - {*v,w*} ∈ E, if and only if {f(*v*),f(*w*)} ∈ E.





Special cases are easier

Polynomial time algorithms for

- ❖ Graphs of bounded degree (Luks' Algorithm (1982))
- ❖ Planar graphs (Hopcroft & Wong (1974))
- ❖ Trees (tree coding, centered tree matching)

Expected polynomial time for random graphs

An equivalence relation on vertices

Say $v \sim w$, if and only if there is an automorphism mapping v to w.

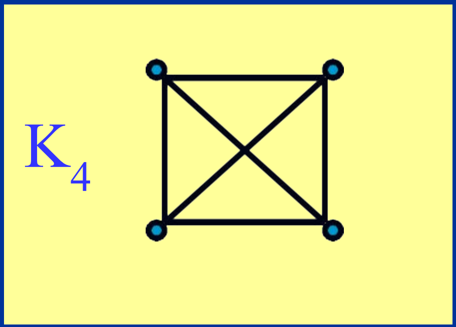
$\sim$ is an equivalence relation

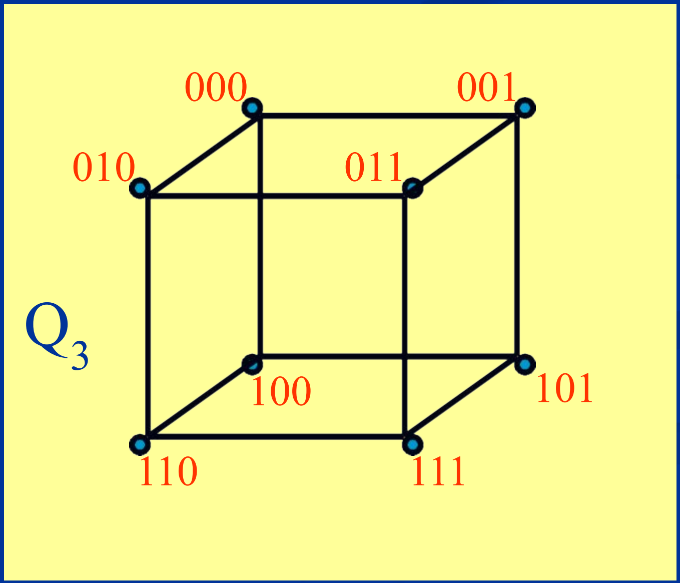
Partitions the vertices in *automorphism classes*

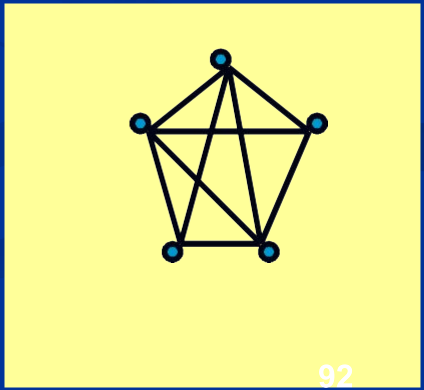
Tells on structure of graph

Planar graphs

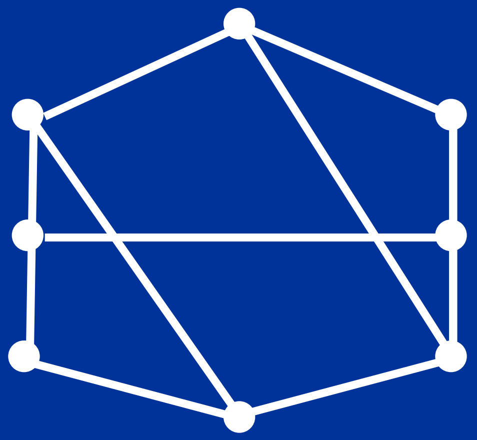
- A graph is *planar* if it can be drawn in the plane without any edges crossing
- A graph that is so drawn in the plane is a plane graph (planar representation).



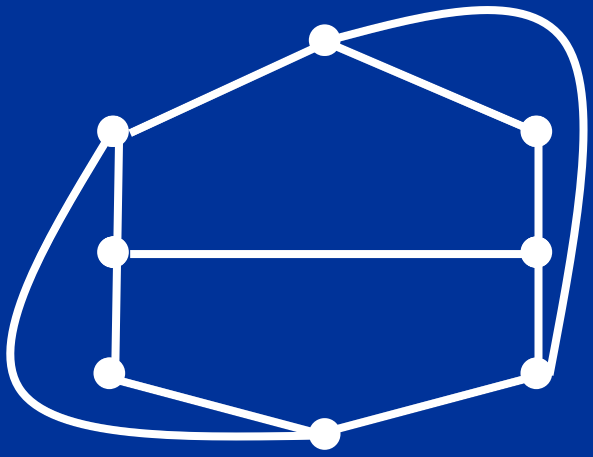




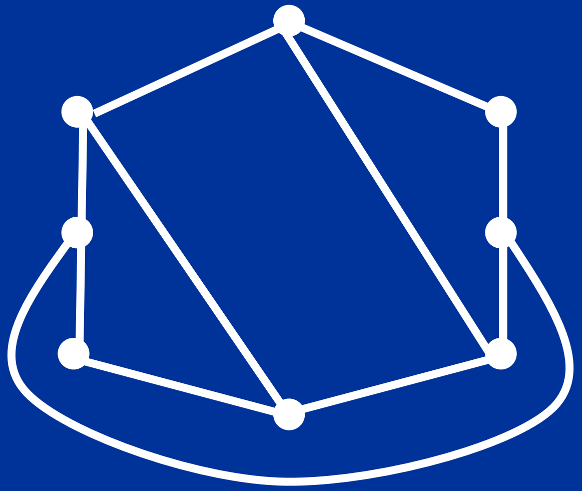
Plane graphs



(a) planar,
not a plane graph



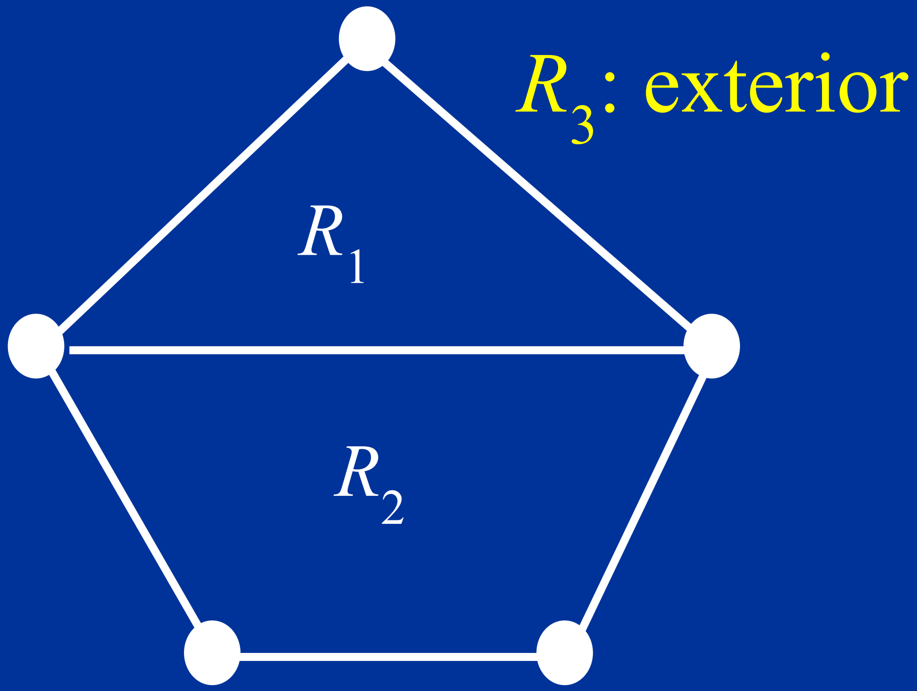
(b) a plane graph



(c) another
plane graph

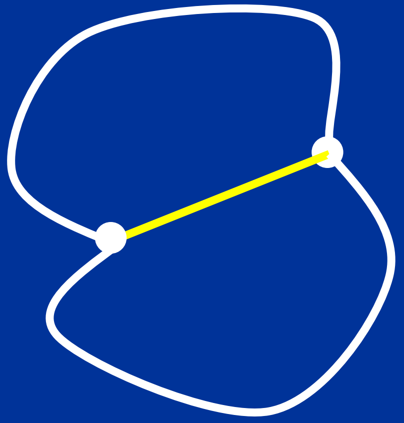
Planar graphs

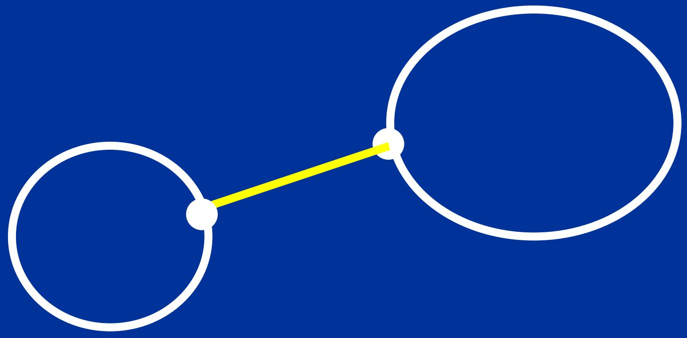
- Let G be a plane graph. The connected pieces of the plane that remain when the vertices and edges of G are removed are called the regions (faces) of G.



Planar graphs

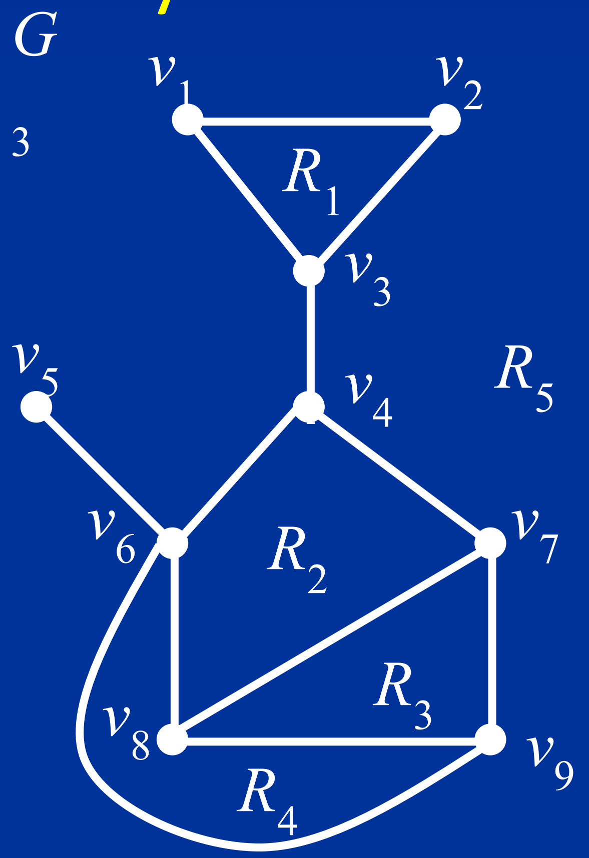
Every plane graph has exactly one unbounded region, called the exterior region (outer face). The vertices and edges of G that are incident with a region R form a subgraph of G called the boundary of R.





Planar graphs

Every plane graph has exactly one unbounded region, called the exterior region. The vertices and edges of G that are incident with a region R form a subgraph of G called the boundary of R.

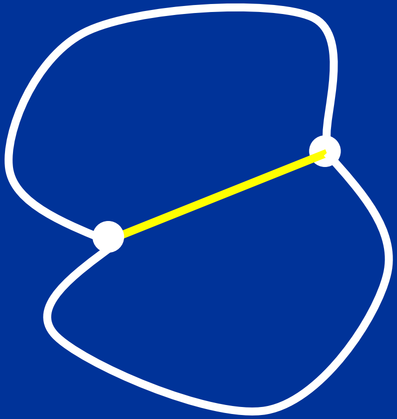


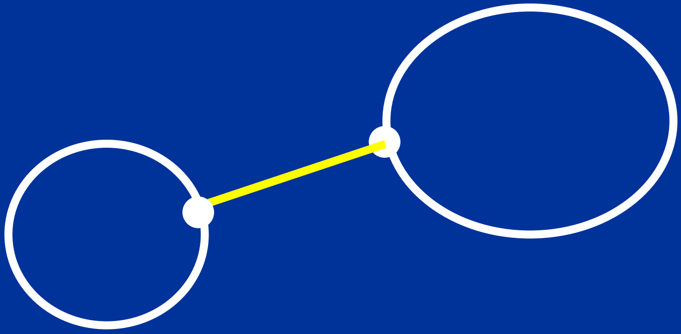
G_3 has 5 regions.

Planar graphs

Degree of a region, which is defined to be the number of edges on the boundary of this region

$$\sum_{r \in G} \deg(r) = 2e$$

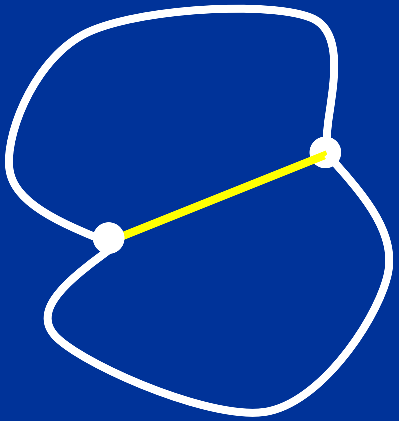


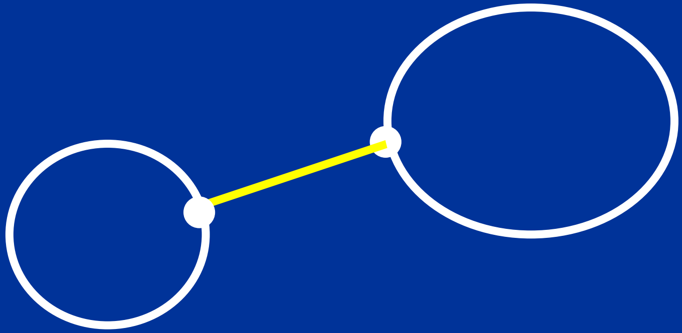


Euler’s Formula

Euler’s Formula: If G is a connected plane graph with n vertices, m edges, and r regions, then

$$n - m + r = 2.$$





Euler's Formula

pf: (by induction)

(basis) If $m = 0$, so n = 1, r =1,

and $n - m + r = 2$.

Constructing a sequence of subgraphs $G_1, G_2, \ldots, G_e = G$, successively adding an edge at each stage. Because G is connected , G_n can be obtained from G_{n-1} by arbitrarily adding an edge that is incident with a vertex already in G_{n-1}.

Assume that the formula is true for $G_1, G_2, \ldots, G_k$, that $r_k = m_k - n_k + 2$.

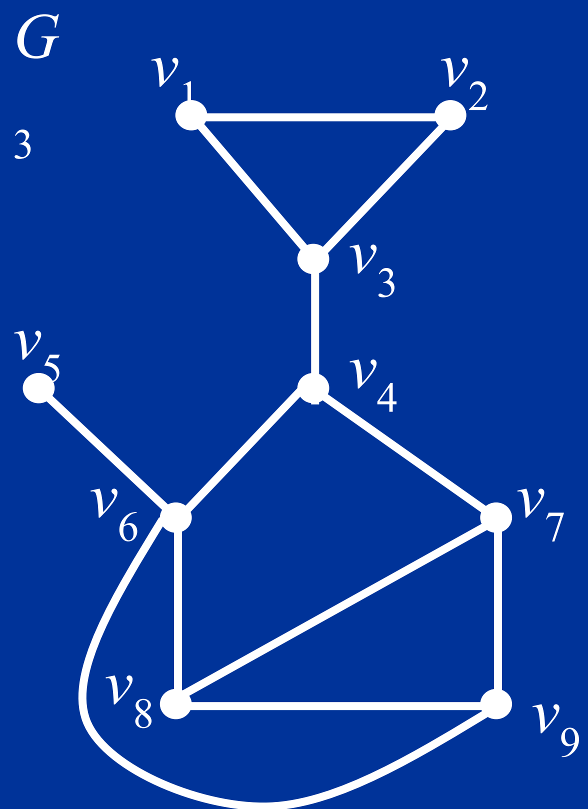
Euler's Formula

Let $\{a_{k+1}, b_{k+1}\}$ be the edge that is added to G_k to obtain G_{k+1}. There are two cases:

Case 1: Both a_{k+1} and b_{k+1} are already in G_k => they must be on the boundary of a common region R (G_{k+1} is planar). (it is impossible to add the edge without two edge crossing if both vertices belongs to different regions). The adding edge split R into two regions:

$r_{k+1} = r_k + 1$, $n_{k+1} = n_k + 1$, $m_{k+1} = m_k$.

Because $r_k = m_k - n_k + 2$ (assumption), $r_{k+1} = m_k - n_{k+1} + 2$.



Euler’s Formula

Let $\{a_{k+1}, b_{k+1}\}$ be the edge that is added to G_k to obtain G_{k+1}. There are two cases:

Case 2: a_{k+1} is in G_k while b_{k+1} is adding => The adding edge does not produce any new region

$r_{k+1} = r_k$, $n_{k+1} = n_k + 1$, $m_{k+1} = m_k$.

Because $r_k = m_k - n_k + 2$ (assumption), $r_{k+1} = m_k - n_{k+1} + 2$.

In both cases, $r_{k+1} = m_k - n_{k+1} + 2$ for G_{k+1}
=> $r = m - n + 2$ for G

Euler's Formula - Corollary1

If G is a connected planar simple graph with n $(n \geq 3)$ vertices, m edges

$$m \leq 3n - 6.$$

Prove: The degree of each region is at least three (no multiple edges, loops)

Sum of the degrees of the regions is 2m

$$2m = \sum_{v \in V} \deg(v) \geq 3r$$

Hence, $r \leq 2/3\ m$

$n - m + r = 2$, => $m \leq 3n - 6$.

Planar graph

Every planar graph contains a vertex of degree 5 or less.

pf: Let G be a planar graph of n vertices and m edges.

If $\deg(v) \geq 6$ for every $v \in V(G)$

$$\Rightarrow \sum_{v \in V} \deg(v) \geq$$

$$\Rightarrow \quad m \geq 3n \qquad \rightarrow\leftarrow$$ COROLLARY1

Planar graph

A plane graph G is called maximal planar if, for every pair u, v of nonadjacent vertices of G, the graph $G+uv$ is nonplanar.

- If G is a maximal simple planar graph with $p \geq 3$ vertices and q edges, then
$$q = 3p - 6.$$

- If G is a maximal simple planar bipartite graph with $p \geq 3$ vertices and q edges, then
$$q = 2p - 4$$

Planar graph

$\boxed{\text{The graphs } K_5 \text{ and } K_{3,3} \text{ are nonplanar.}}$

(1) K_5 has $p = 5$ vertices and $q = 10$ edges.

$$q > 3p - 6 \quad \Rightarrow K_5 \text{ is nonplanar.}$$

(2) Suppose $K_{3,3}$ is planar, and consider any embedding of $K_{3,3}$ in the plane.
Suppose the embedding has r regions.

$$p - q + r = 2 \quad \Rightarrow \quad r = 5$$

$K_{3,3}$ is bipartite
⇒ The boundary of every region has ≥4 edges.

$$\Longrightarrow 4r \leq \sum_{\forall \text{ region } R} |\{\text{the edges of the boundary of } R\}| = 2q = 18$$

$$\longrightarrow\longleftarrow$$

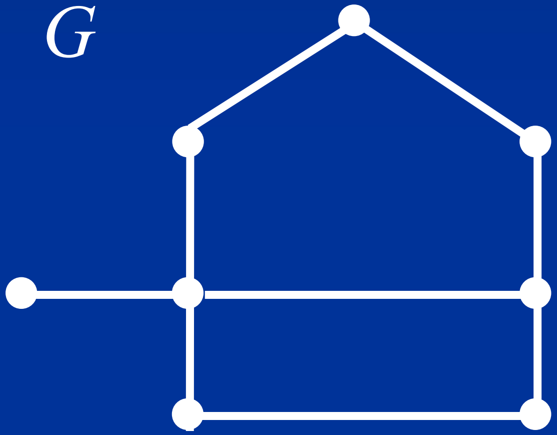
Planar graph

A subdivision of a graph G is a graph obtained by inserting vertices (of degree 2) into the edges of G.

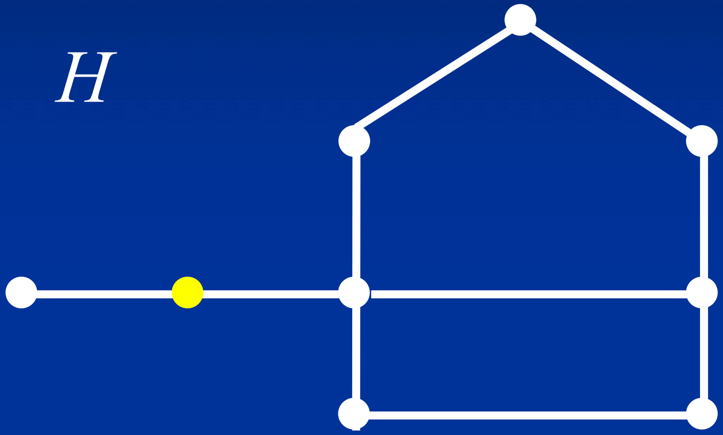
Two graphs G1 and G2 are called homeomorphic if they are constructed from the same graph through a sequence of divisions

Subdivisions of graphs.

G

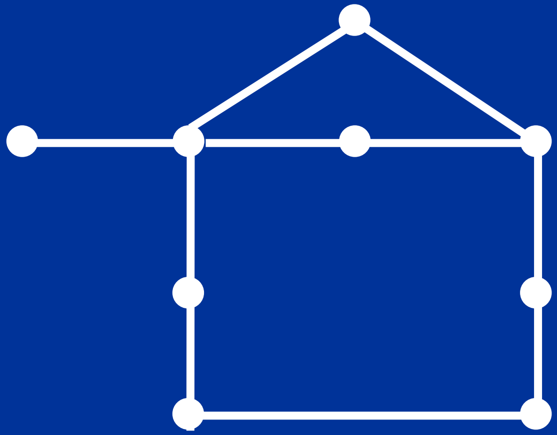


H



H is a subdivision of G.

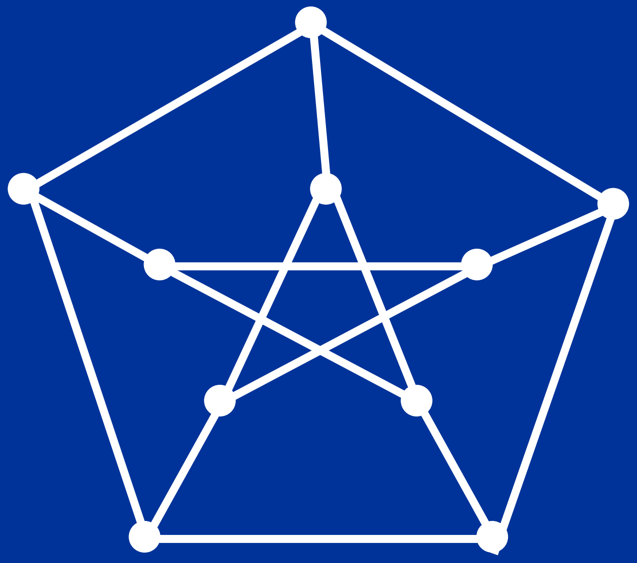
F



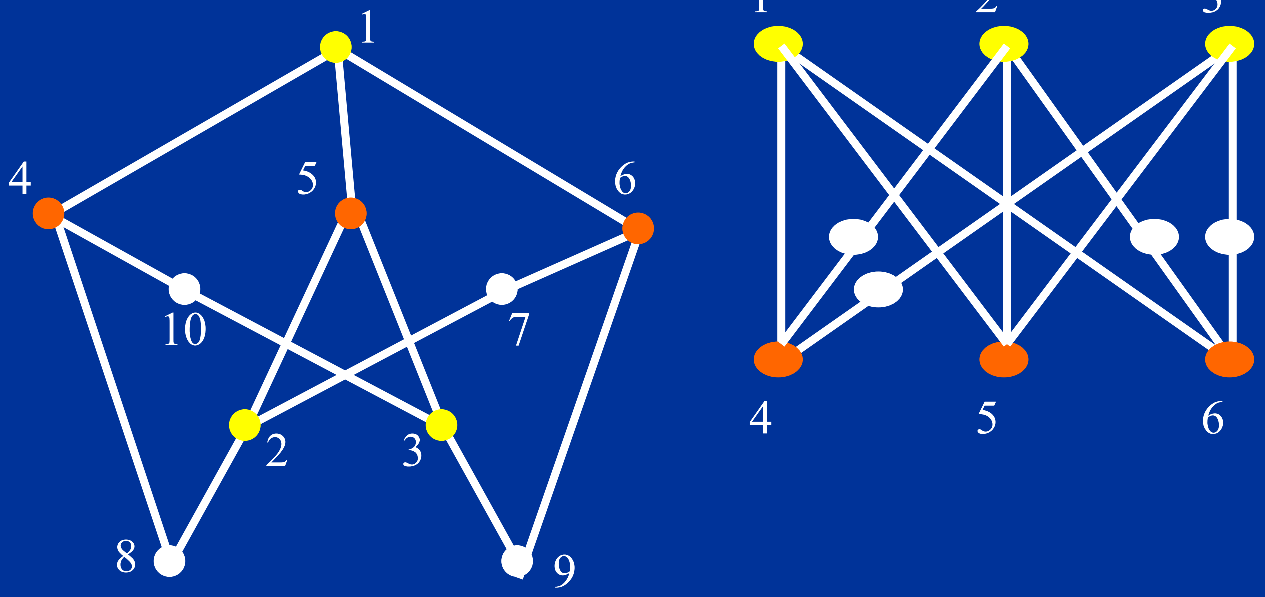
F is not a subdivision of G.

Kuratowski's Theorem

A graph is planar if and only if it contains no subgraph that is isomorphic to or is a subdivision of K_5 or $K_{3,3}$.



(a) Petersen



(b) Subdivision of $K_{3,3}$

Graph Coloring

A coloring of the vertices of a graph G is an assignment of colors to the vertices. A coloring is proper if adjacentvertices always have different colors. This is also called the vertex coloring problem

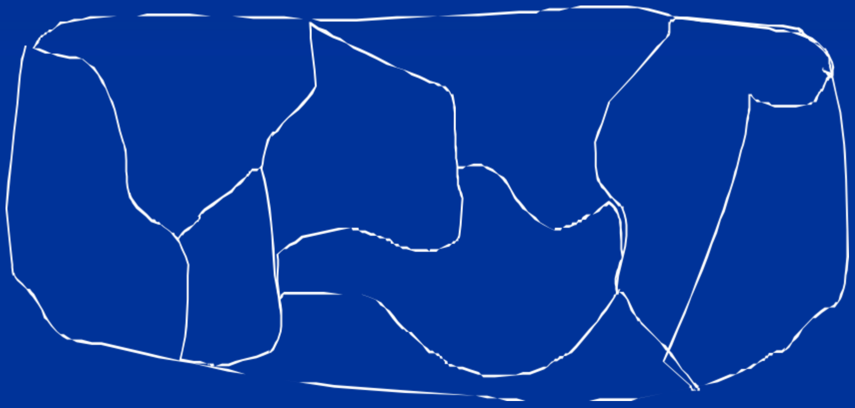
An m-coloring of G is a mapping from V onto the set {1,2,...,m} of m "colors"

Graph coloring includes:

- Vertex coloring
- Edge coloring
- Face coloring

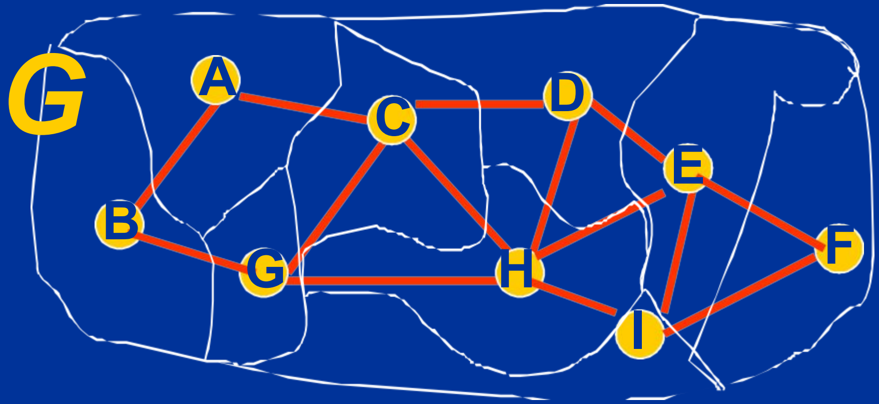
Graph Coloring

- What is the least number of colors needed to color a map?



Dual Map

Region = vertex
Common border = edge

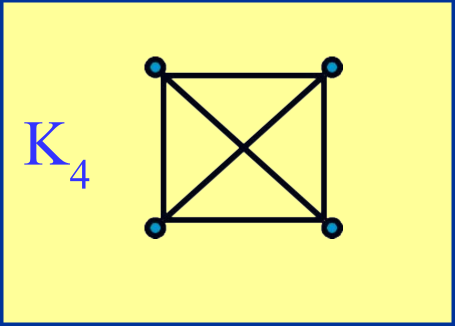


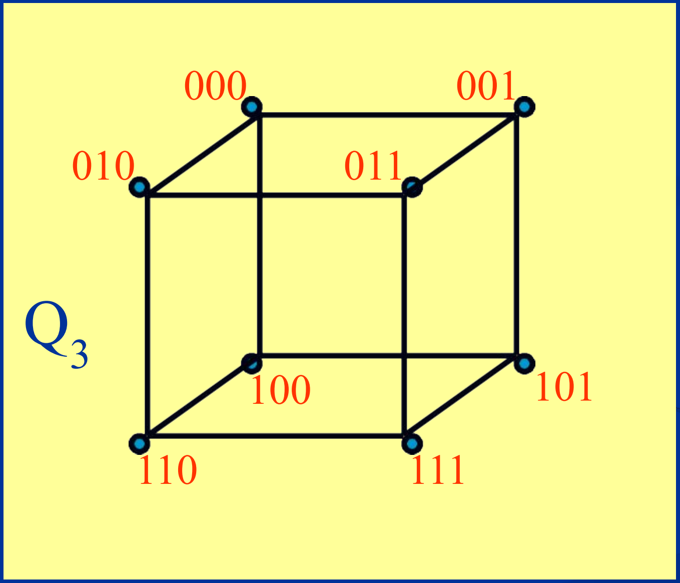
Chromatic number $\chi(G)$

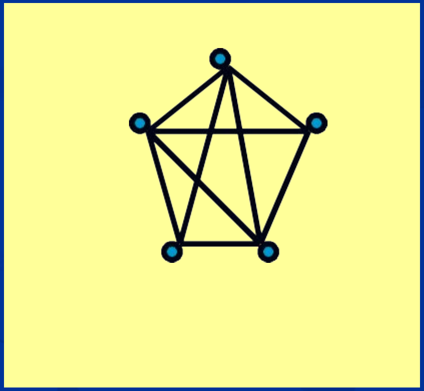
What is the ***least number of colors*** needed to color the vertices of a graph so that no two adjacent vertices are assigned the same color?

Four color theorem

Every planar graph is 4-colorable







Chromatic Numbers (cont.)

- There is no efficient algorithm for finding $\chi(G)$ for an arbitrary graph. Most computer scientists believe that such an algorithm doesn't exist.

- There is an efficient algorithm to determine whether a given graph is bipartite, i.e., 2-colorable.
- Applications:
  - map coloring
  - scheduling