Discrete Mathematical Models

Lecture 23

Kane Townsend

Semester 2, 2024

Special simple graphs

Cycle Graphs

A **cycle graph** on $n \geq 3$ vertices is denoted by C_n.

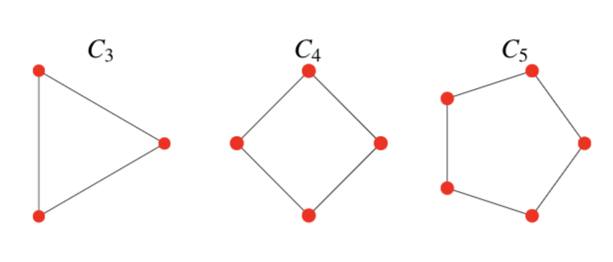


Figure 1: Wolfram Mathworld: Cycle Graph

Cycle Graphs

A **cycle graph** on $n \geq 3$ vertices is denoted by C_n.

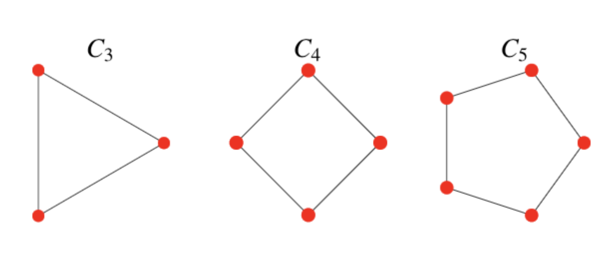


Figure 1: Wolfram Mathworld: Cycle Graph

Note that $|V(C_n)| = n = |E(C_n)|$ and that every vertex is adjacent to two other vertices.

Complete Graphs

A **complete graph** on n vertices is a simple graph in which each pair of distinct vertices are adjacent (*i.e.* are 'joined' by an edge).

Complete Graphs

A **complete graph** on n vertices is a simple graph in which each pair of distinct vertices are adjacent (*i.e.* are 'joined' by an edge).

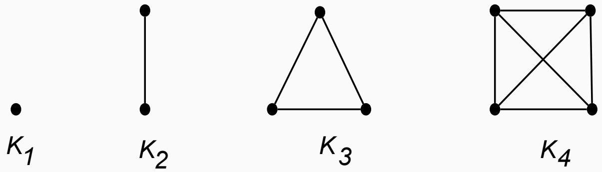
A complete graph on n vertices is denoted by K_n.

Complete Graphs

A **complete graph** on n vertices is a simple graph in which each pair of distinct vertices are adjacent (*i.e.* are 'joined' by an edge).

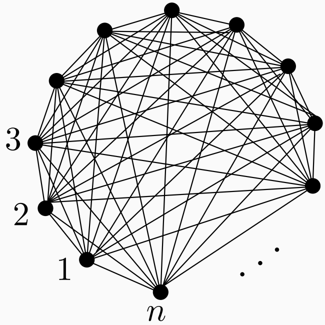
A complete graph on n vertices is denoted by K_n.

Examples:



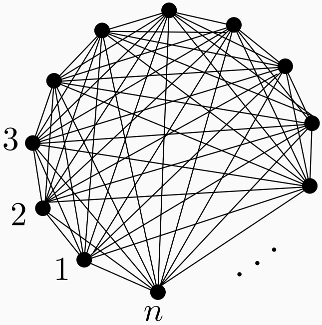
How many edges in K_n?

How many edges are there in K_n, the complete graph of order n?



How many edges in K_n?

How many edges are there in K_n, the complete graph of order n?



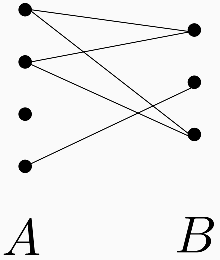
The answer is $\binom{n}{2} = \frac{n(n-1)}{2}$. *Why?*

Bipartite Graphs

A **bipartite** graph is a simple graph whose vertices can be partitioned into two disjoint sets A and B such that **every edge** of the graph connects a vertex in A to a vertex in B.

A **bipartite** graph is a simple graph whose vertices can be partitioned into two disjoint sets A and B such that **every edge** of the graph connects a vertex in A to a vertex in B.

Example:



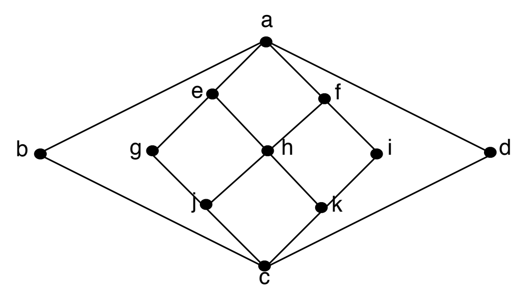
A larger example of a bipartite graph

Sometimes it is not obvious at first glance that a graph is bipartite.

A larger example of a bipartite graph

Sometimes it is not obvious at first glance that a graph is bipartite.

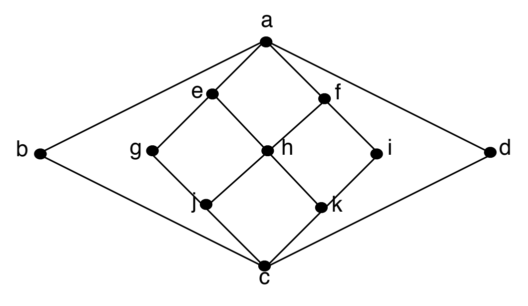
Example: This graph is bipartite:



A larger example of a bipartite graph

Sometimes it is not obvious at first glance that a graph is bipartite.

Example: This graph is bipartite:

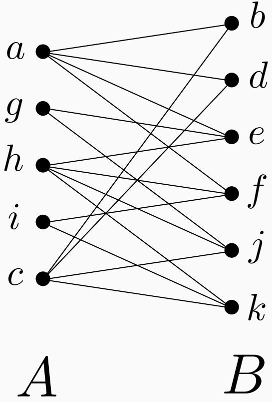


The vertex partitioning is:

$$A = \{a, g, h, i, c\} \quad B = \{b, d, e, f, j, k\}$$

Larger example continued

This is the same graph, redrawn.



Complete Bipartite Graphs

A **complete bipartite** graph is a simple graph whose vertices may be partitioned into two sets A and B such that:

Complete Bipartite Graphs

A **complete bipartite** graph is a simple graph whose vertices may be partitioned into two sets A and B such that:

- $\forall a \in A$ and $\forall b \in B$, the edge $\{a, b\}$ belongs to the graph.

Complete Bipartite Graphs

A **complete bipartite** graph is a simple graph whose vertices may be partitioned into two sets A and B such that:

- $\forall a \in A$ and $\forall b \in B$, the edge $\{a, b\}$ belongs to the graph.
- There are no other edges in the graph.

Complete Bipartite Graphs

A **complete bipartite** graph is a simple graph whose vertices may be partitioned into two sets A and B such that:

- $\forall a \in A$ and $\forall b \in B$, the edge $\{a, b\}$ belongs to the graph.
- There are no other edges in the graph.

If A has m vertices and B has n vertices, the complete bipartite graph on A and B is denoted by $K_{m,n}$.

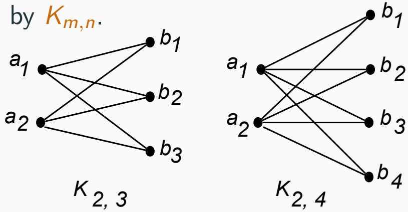
Complete Bipartite Graphs

A **complete bipartite** graph is a simple graph whose vertices may be partitioned into two sets A and B such that:

- $\forall a \in A$ and $\forall b \in B$, the edge $\{a, b\}$ belongs to the graph.
- There are no other edges in the graph.

If A has m vertices and B has n vertices, the complete bipartite graph on A and B is denoted by $K_{m,n}$.

Examples:



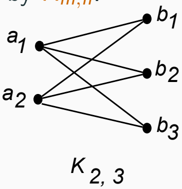
Complete Bipartite Graphs

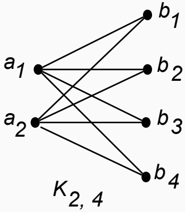
A **complete bipartite** graph is a simple graph whose vertices may be partitioned into two sets A and B such that:

- $\forall a \in A$ and $\forall b \in B$, the edge $\{a, b\}$ belongs to the graph.
- There are no other edges in the graph.

If A has m vertices and B has n vertices, the complete bipartite graph on A and B is denoted by $K_{m,n}$.

Examples:

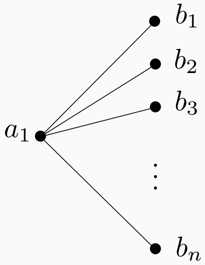




Caution: The name "Complete Bipartite Graph" is misleading. Except for $K_{1,1}$, such graphs are **not complete graphs**.
The adjective 'complete' is with respect to 'bipartite', not 'graph'.

How many edges in $K_{1,n}$?

- *How many edges in the complete bipartite graph $K_{1,n}$?*



How many edges in $K_{m,n}$?

- *How many edges in* $K_{1,n}$*?*
- *How many edges in* $K_{2,n}$*?*

$\vdots$

- *How many edges in* $K_{m,n}$*?*

Subgraphs

Subgraphs

- A **subgraph**, S, of a graph G, is a graph whose vertices are a subset of $V(G)$ and whose edges are a subset of $E(G)$, i.e.

$$V(S) \subseteq V(G)$$
$$E(S) \subseteq E(G)$$

Subgraphs

- A **subgraph**, S, of a graph G, is a graph whose vertices are a subset of $V(G)$ and whose edges are a subset of $E(G)$, i.e.

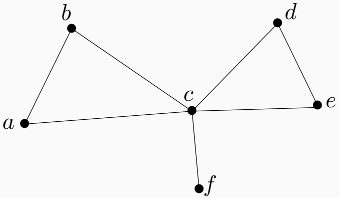
$$V(S) \subseteq V(G)$$
$$E(S) \subseteq E(G)$$

- Since S is a graph, if the edge

$$\{a, b\} \in E(S),$$

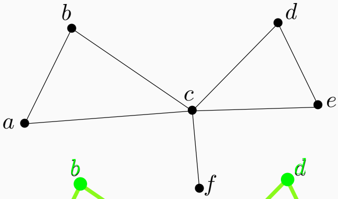
we require its endpoints to be in $V(S)$, i.e. $a, b \in V(S)$.

A subgraph example:

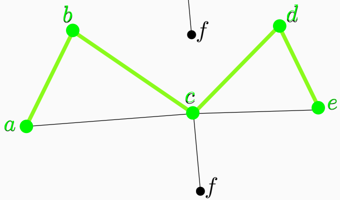
Let G be the graph: 

A subgraph example:

Let G be the graph:

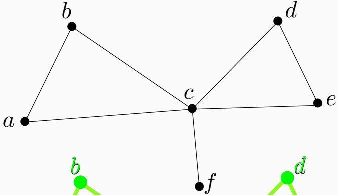


Select some edges and vertices:

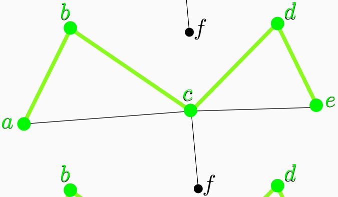


A subgraph example:

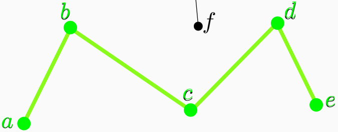
Let G be the graph:



Select some edges and vertices:

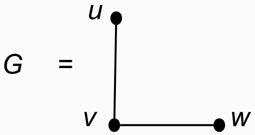


Now S is a subgraph of G:



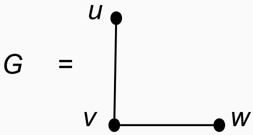
Another subgraphs example:

Let



Another subgraphs example:

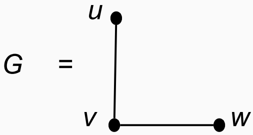
Let



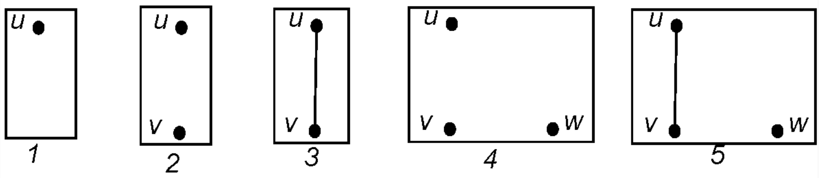
The following five graphs (numbered 1-5) are each subgraphs of G:

Another subgraphs example:

Let



The following five graphs (numbered 1-5) are each subgraphs of G:



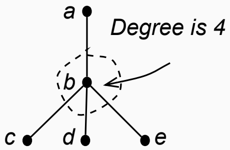
Degree

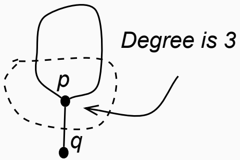
Degree of a vertex

- The **degree** of a vertex is the number of edges incident on it (but with each loop counted twice — once for each ‘end’).

Degree of a vertex

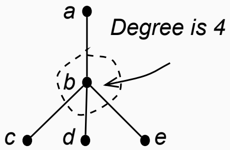
- The **degree** of a vertex is the number of edges incident on it (but with each loop counted twice — once for each ‘end’).

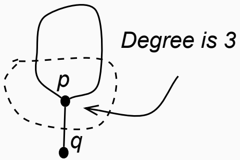




Degree of a vertex

- The **degree** of a vertex is the number of edges incident on it (but with each loop counted twice — once for each 'end').

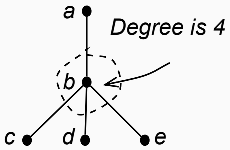


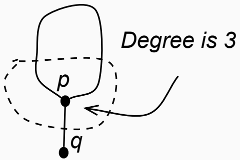


- Let v be a vertex and let k be its degree. We write $\mathbf{deg}(v)=k$.

Degree of a vertex

- The **degree** of a vertex is the number of edges incident on it (but with each loop counted twice — once for each 'end').



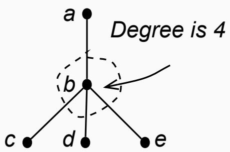


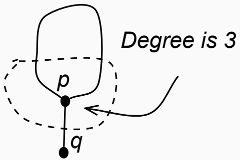
- Let v be a vertex and let k be its degree. We write **deg**$(v)=k$.

- So in the above examples $\deg(b) = 4$ and $\deg(p) = 3$.

Degree of a vertex

- The **degree** of a vertex is the number of edges incident on it (but with each loop counted twice — once for each 'end').



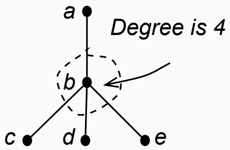


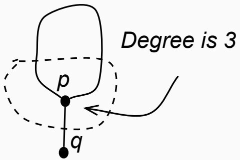
- Let v be a vertex and let k be its degree. We write **deg**$(v){=}k$.
- So in the above examples $\deg(b) = 4$ and $\deg(p) = 3$.

Warning: Some authors count a loop as only adding one to a degree, rather than adding two as we do.

Degree of a vertex

- The **degree** of a vertex is the number of edges incident on it (but with each loop counted twice — once for each 'end').





- Let v be a vertex and let k be its degree. We write $\mathbf{deg}(v)=k$.

- So in the above examples $\deg(b) = 4$ and $\deg(p) = 3$.

Warning: Some authors count a loop as only adding one to a degree, rather than adding two as we do.
We shall always use the 'adding two' version.

Total degree of a graph

The **total degree** of a graph G is the sum of the degrees of all its vertices, $\sum_{v \in V(G)} \deg(v)$.

Total degree of a graph

The **total degree** of a graph G is the sum of the degrees of all its vertices, $\displaystyle\sum_{v \in V(G)} \deg(v)$. *In other words:*

Let $v_1, v_2, \ldots, v_n$ for $n \in \mathbb{N}$, be the vertices of G. Then

$$\text{the total degree of } G = \deg(v_1) + \deg(v_2) + \cdots + \deg(v_n).$$

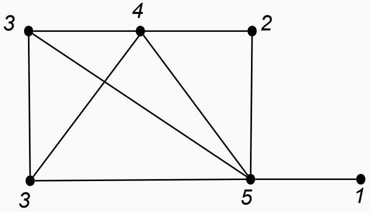
Total degree of a graph

The **total degree** of a graph G is the sum of the degrees of all its vertices, $\sum_{v \in V(G)} \deg(v)$. *In other words:*

Let $v_1, v_2, \ldots, v_n$ for $n \in \mathbb{N}$, be the vertices of G. Then

$$\text{the total degree of } G = \deg(v_1) + \deg(v_2) + \cdots + \deg(v_n).$$

Example: In this graph the degree of each vertex is shown.



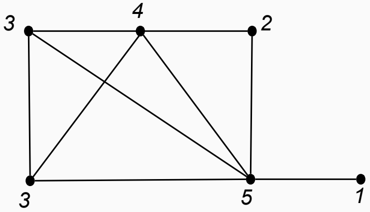
Total degree of a graph

The **total degree** of a graph G is the sum of the degrees of all its vertices, $\sum_{v \in V(G)} \deg(v)$. *In other words:*

Let $v_1, v_2, \ldots, v_n$ for $n \in \mathbb{N}$, be the vertices of G. Then

$$\text{the total degree of } G = \deg(v_1) + \deg(v_2) + \cdots + \deg(v_n).$$

Example: In this graph the degree of each vertex is shown.



The total degree of the graph is $3 + 4 + 2 + 3 + 5 + 1 = 18$.

The Handshake Theorem

The Handshake Theorem: If G is any graph, then the total degree of G equals twice the number of edges of G.

The Handshake Theorem

The Handshake Theorem: If G is any graph, then the total degree of G equals twice the number of edges of G.

i.e. $$\boxed{\sum_{v \in V(G)} \deg(v) = 2|E(G)|}$$

The Handshake Theorem

The Handshake Theorem: If G is any graph, then the total degree of G equals twice the number of edges of G.

i.e. $$\boxed{\sum_{v \in V(G)} \deg(v) = 2|E(G)|}$$

Why?

The Handshake Theorem

The Handshake Theorem: If G is any graph, then the total degree of G equals twice the number of edges of G.

i.e. $$\boxed{\sum_{v \in V(G)} \deg(v) = 2|E(G)|}$$

Why?

Because when we count degrees we are counting edges, but we count both ends of each edge, hence we count all the edges twice.

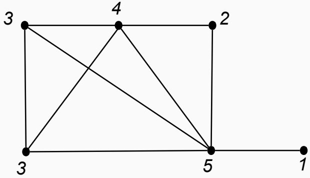
The Handshake Theorem: If G is any graph, then the total degree of G equals twice the number of edges of G.

i.e. $$\boxed{\sum_{v \in V(G)} \deg(v) = 2|E(G)|}$$

Why?

Because when we count degrees we are counting edges, but we count both ends of each edge, hence we count all the edges twice.

Example:



Total degree = 18
Number of edges = 9.

A Corollary

In any graph there is an even number of vertices of odd degree.

A Corollary

$\boxed{\text{In any graph there is an even number of vertices of odd degree.}}$

How does this corollary follow from the handshake theorem?

A Corollary

In any graph there is an even number of vertices of odd degree.

How does this corollary follow from the handshake theorem?

Example:

- The set $\{1, 1, 2, 3\}$ cannot possibly be the set of degrees of the vertices of some graph.

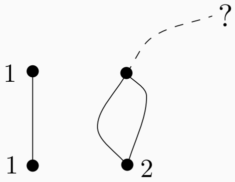
A Corollary

In any graph there is an even number of vertices of odd degree.

How does this corollary follow from the handshake theorem?

Example:

- The set $\{1, 1, 2, 3\}$ cannot possibly be the set of degrees of the vertices of some graph.
- However we try, we always end up with an edge that doesn't have a vertex to connect to:



A useful abbreviation

We often abbreviate

- the graph edge $\{a, b\}$ as ab, and also
- the digraph arc (a, b) as ab.

A useful abbreviation

We often abbreviate

- the graph edge $\{a, b\}$ as ab, and also
- the digraph arc (a, b) as ab.

Note that

- for graphs, ab means *the same* as ba, since $\{a, b\} = \{b, a\}$; but
- for digraphs, ab is *different* from ba since $(a, b) \neq (b, a)$.

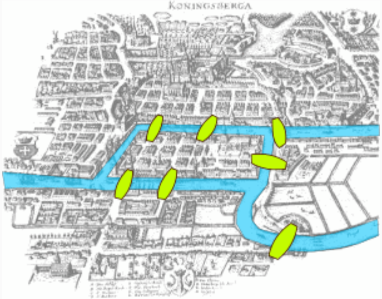
Walks, Paths and Circuits

The bridges of Königsberg

- The city of Königsberg in Prussia was set on both sides of the Pregel River, and included two large islands which were connected to each other and the mainland by seven bridges.

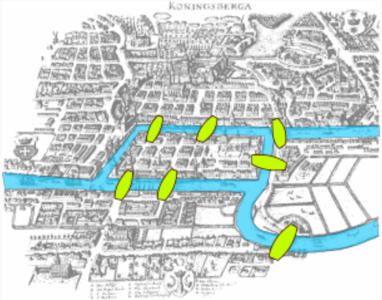
The bridges of Königsberg

- The city of Königsberg in Prussia was set on both sides of the Pregel River, and included two large islands which were connected to each other and the mainland by seven bridges.
- The problem was to find a walk through the city that would cross each bridge once and only once.



The bridges of Königsberg

- The city of Königsberg in Prussia was set on both sides of the Pregel River, and included two large islands which were connected to each other and the mainland by seven bridges.
- The problem was to find a walk through the city that would cross each bridge once and only once.

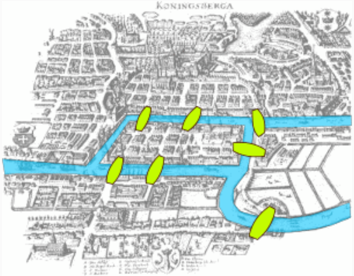


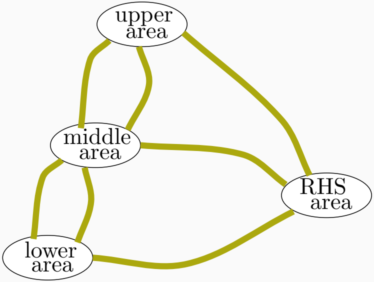
Adapted from:

http://en.wikipedia.org/wiki/Bridges_of_Konigsberg

The bridges of Königsberg

Leonard Euler realized that the task can be modeled as a problem in graph theory, which he invented for the purpose.





Adapted from:

http://en.wikipedia.org/wiki/Bridges_of_Konigsberg

Walks

- A **walk** in a graph is a sequence of vertices alternating with edges:

$$v_0, e_1, v_1, e_2, v_2, e_3, \ldots, e_n, v_n$$

  in which each edge e_k has endpoints v_{k-1} and v_k.

Walks

- A **walk** in a graph is a sequence of vertices alternating with edges:

$$v_0, e_1, v_1, e_2, v_2, e_3, \ldots, e_n, v_n$$

  in which each edge e_k has endpoints v_{k-1} and v_k.

- A walk starting at v_0 and ending at v_n is said to **connect** v_0 to v_n.

Walks

- A **walk** in a graph is a sequence of vertices alternating with edges:

$$v_0, e_1, v_1, e_2, v_2, e_3, \ldots, e_n, v_n$$

  in which each edge e_k has endpoints v_{k-1} and v_k.

- A walk starting at v_0 and ending at v_n is said to **connect** v_0 to v_n.
- The **length** of the walk is the number of edges listed;
  length n for the walk above.

Walks

- A **walk** in a graph is a sequence of vertices alternating with edges:

$$v_0, e_1, v_1, e_2, v_2, e_3, \ldots, e_n, v_n$$

  in which each edge e_k has endpoints v_{k-1} and v_k.

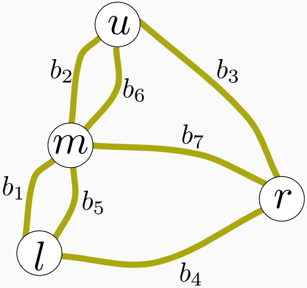
- A walk starting at v_0 and ending at v_n is said to **connect** v_0 to v_n.

- The **length** of the walk is the number of edges listed;
  length n for the walk above.

- A **trivial walk**, say v_0, contains no edges; hence has length 0.

Walks on the Königsberg Graph

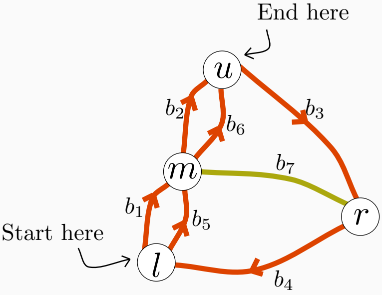
The question becomes, 'Is there a walk on the Königsberg Graph which traverses each edge exactly once?'



Walks on the Königsberg Graph

Try starting in the lower part of town, going via bridge b_1 to the middle island, then via bridge b_2 to the upper part, then via bridge b_3 to the right-most part; continuing as in the listed walk:

$$l b_1 m b_2 u b_3 r b_4 l b_5 m b_6 u$$

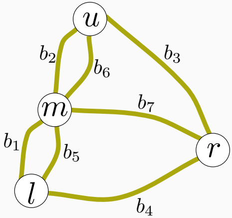


Walks on the Königsberg Graph

- That attempt didn't work, because there is no unused edge to exit u from on the second visit.

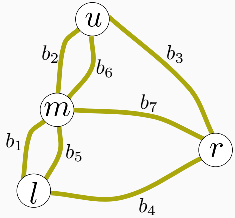
Walks on the Königsberg Graph

- That attempt didn't work, because there is no unused edge to exit u from on the second visit.
- For brevity we'll leave out vertices and just list edges. Try
  - $b_1 b_2 b_3 b_4 b_5 b_6$ ... stuck at u



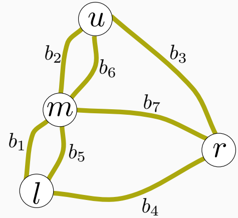
Walks on the Königsberg Graph

- That attempt didn't work, because there is no unused edge to exit u from on the second visit.
- For brevity we'll leave out vertices and just list edges. Try
  - $b_1 b_2 b_3 b_4 b_5 b_6$ ... stuck at u
  - $b_1 b_2 b_3 b_4 b_5 b_7$ ... stuck at r



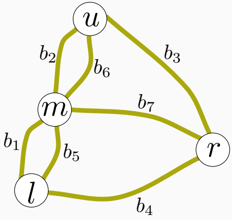
Walks on the Königsberg Graph

- That attempt didn't work, because there is no unused edge to exit u from on the second visit.
- For brevity we'll leave out vertices and just list edges. Try
  - $b_1 b_2 b_3 b_4 b_5 b_6$ ... stuck at u
  - $b_1 b_2 b_3 b_4 b_5 b_7$ ... stuck at r
  - $b_2 b_6 b_1 b_5 b_7 b_4$ ... stuck at l



Walks on the Königsberg Graph

- That attempt didn't work, because there is no unused edge to exit u from on the second visit.
- For brevity we'll leave out vertices and just list edges. Try
  - $b_1 b_2 b_3 b_4 b_5 b_6$ ... stuck at u
  - $b_1 b_2 b_3 b_4 b_5 b_7$ ... stuck at r
  - $b_2 b_6 b_1 b_5 b_7 b_4$ ... stuck at l
  - $b_1 b_2 b_6 b_5 b_4 b_7$ ... stuck at m



Impossibility of Königsberg Bridge Walk

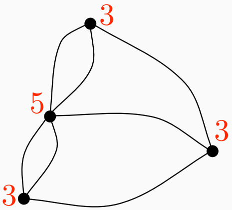
- The kind of walk needed to meet the criteria for the Königsberg Bridge problem - a walk which traverses all edges and repeats none - came to be called an **Euler Path**.

Impossibility of Königsberg Bridge Walk

- The kind of walk needed to meet the criteria for the Königsberg Bridge problem - a walk which traverses all edges and repeats none - came to be called an **Euler Path**.
- Euler showed that no such walk exists on the Königsberg Graph.

Impossibility of Königsberg Bridge Walk

- The kind of walk needed to meet the criteria for the Königsberg Bridge problem - a walk which traverses all edges and repeats none - came to be called an **Euler Path**.
- Euler showed that no such walk exists on the Königsberg Graph.
- *How? Think about the degrees of the vertices:*



Impossibility of Königsberg Bridge Walk

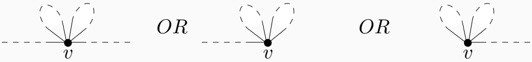
- *There are too many vertices of odd degree in the Königsberg Graph for an Euler path to be possible. Can you prove it?*

Impossibility of Königsberg Bridge Walk

- *There are too many vertices of odd degree in the Königsberg Graph for an Euler path to be possible. Can you prove it?*
- *Hint: can an odd degree vertex occur in the middle of a walk?*

Impossibility of Königsberg Bridge Walk

- *There are too many vertices of odd degree in the Königsberg Graph for an Euler path to be possible.* *Can you prove it?*
- *Hint: can an odd degree vertex occur in the middle of a walk?*



Even degree vertex in a walk

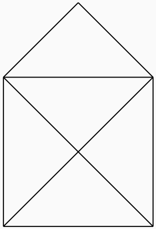
Odd degree vertex in a walk

Odd degree vertex in a walk

A related puzzle

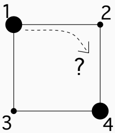
You may have come across the following puzzle:

Can you draw the following 'house' diagram without taking your pen off the paper or overwriting edges?



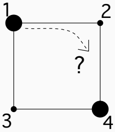
Can you?

Counting Walks with Adjacency Matrices

The graph 

has adjacency matrix $M = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}.$

Counting Walks with Adjacency Matrices

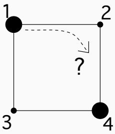
The graph 

has adjacency matrix

$$M = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}.$$

There are 0 ways to walk from vertex 1 to vertex 4 in a single step.

Counting Walks with Adjacency Matrices

The graph  has adjacency matrix

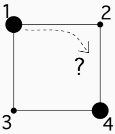
$$M = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}.$$

There are 0 ways to walk from vertex 1 to vertex 4 in a single step.

There are 2 ways to walk from vertex 1 to vertex 4 in two steps:

Counting Walks with Adjacency Matrices

The graph

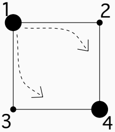


has adjacency matrix

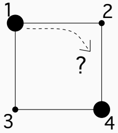
$$M = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}.$$

There are 0 ways to walk from vertex 1 to vertex 4 in a single step.

There are 2 ways to walk from vertex 1 to vertex 4 in two steps:



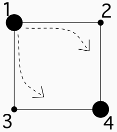
Counting Walks with Adjacency Matrices

The graph 

has adjacency matrix
$$M = \begin{array}{cc} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}.$$

There are 0 ways to walk from vertex 1 to vertex 4 in a single step.

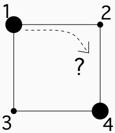
There are 2 ways to walk from vertex 1 to vertex 4 in two steps:



$$M^2 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$= \begin{array}{cc} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 2 & 0 & 0 & 2 \\ 0 & 2 & 2 & 0 \\ 0 & 2 & 2 & 0 \\ 2 & 0 & 0 & 2 \end{pmatrix} \end{array}$$

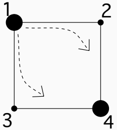
Counting Walks with Adjacency Matrices

The graph 

has adjacency matrix
$$M = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}.$$

There are 0 ways to walk from vertex 1 to vertex 4 in a single step.

There are 2 ways to walk from vertex 1 to vertex 4 in two steps:



$$M^2 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$= \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 2 & 0 & 0 & 2 \\ 0 & 2 & 2 & 0 \\ 0 & 2 & 2 & 0 \\ 2 & 0 & 0 & 2 \end{pmatrix} \end{array}$$

For any graph, the number of ways to walk from vertex i to vertex j in n steps is given in terms of its adjacency matrix M by the $(i, j)^{\text{th}}$ entry of M^n.

Some special kinds of walks

Some properties that a walk on a graph may, or may not, possess are:

Some special kinds of walks

Some properties that a walk on a graph may, or may not, possess are:

- being ‘closed’;

Some special kinds of walks

Some properties that a walk on a graph may, or may not, possess are:

- being 'closed';
- having no repeated edges;

Some special kinds of walks

Some properties that a walk on a graph may, or may not, possess are:

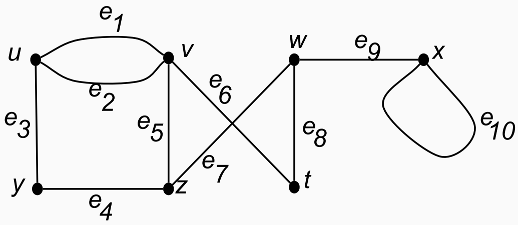
- being ‘closed’;
- having no repeated edges;
- having no repeated vertices.

Some special kinds of walks

Some properties that a walk on a graph may, or may not, possess are:

- being 'closed';
- having no repeated edges;
- having no repeated vertices.

We will now look at each of these potential properties in turn, with examples using the graph G below.



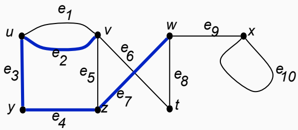
Closed walks

A walk $v_0, e_1, v_1, e_2, \ldots, e_n, v_n$ is called a **closed** when $v_0 = v_n$.

Closed walks

A walk $v_0, e_1, v_1, e_2, \ldots, e_n, v_n$ is called a **closed** when $v_0 = v_n$.

Examples:



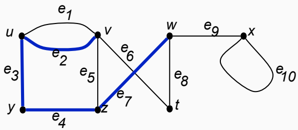
The walk

$$v \; e_2 \; u \; e_3 \; y \; e_4 \; z \; e_7 \; w$$

has length 4 and is **not** closed because $v = v_0 \neq v_n = v_4 = w$.

A walk $v_0, e_1, v_1, e_2, \ldots, e_n, v_n$ is called a **closed** when $v_0 = v_n$.

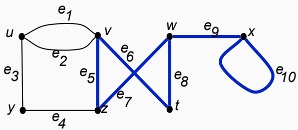
Examples:



The walk

$$v\ e_2\ u\ e_3\ y\ e_4\ z\ e_7\ w$$

has length 4 and is **not** closed because $v = v_0 \neq v_n = v_4 = w$.



The walk

$$v\ e_6\ t\ e_8\ w\ e_9\ x\ e_{10}\ x\ e_9\ w\ e_7\ z\ e_5\ v$$

has length 7 and **is** closed because $v = v_0 = v_n = v_7 = v$.

Paths

A **path** is a walk that does not repeat any edge.

Paths

A **path** is a walk that does not repeat any edge.

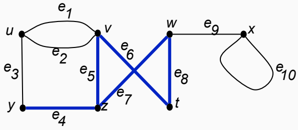
A **simple path** is a path which does not repeat any vertex — except the first and last if the path is closed.

Paths

A **path** is a walk that does not repeat any edge.

A **simple path** is a path which does not repeat any vertex — except the first and last if the path is closed.

Examples:



The walk

$$y\ e_4\ z\ e_7\ w\ e_8\ t\ e_6\ v\ e_5\ z$$

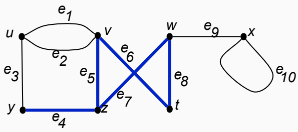
has length 5 and is a **path** because the five edges are all different, but is **not simple** because $z = v_1 = v_5$

Paths

A **path** is a walk that does not repeat any edge.

A **simple path** is a path which does not repeat any vertex — except the first and last if the path is closed.

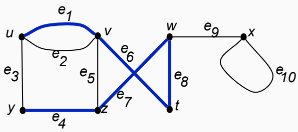
Examples:



The walk

$$y\ e_4\ z\ e_7\ w\ e_8\ t\ e_6\ v\ e_5\ z$$

has length 5 and is a **path** because the five edges are all different, but is **not simple** because $z = v_1 = v_5$



The walk

$$y\ e_4\ z\ e_7\ w\ e_8\ t\ e_6\ v\ e_1\ u$$

has length 5 and is a **simple path** because the six vertices are all different.

Circuits

A **circuit** is a closed path.

Circuits

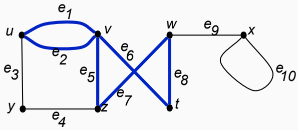
A **circuit** is a closed path.

A **simple circuit** is a simple closed path.

A **circuit** is a closed path.

A **simple circuit** is a simple closed path.

Examples:



The walk

$$z\,e_7\,w\,e_8\,t\,e_6\,v\,e_1\,u\,e_2\,v\,e_5\,z$$

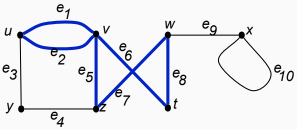
has length 6 and is a **circuit** as it is closed with all different edges, but is **not simple** because $v = v_3 = v_5$.

Circuits

A **circuit** is a closed path.

A **simple circuit** is a simple closed path.

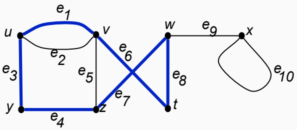
Examples:



The walk

$$z\, e_7\, w\, e_8\, t\, e_6\, v\, e_1\, u\, e_2\, v\, e_5\, z$$

has length 6 and is a **circuit** as it is closed with all different edges, but is **not simple** because $v = v_3 = v_5$.



The walk

$$z\, e_7\, w\, e_8\, t\, e_6\, v\, e_1\, u\, e_3\, y\, e_4\, z$$

has length 6 and is a **simple circuit** as it is closed without repeated vertices except the first and last $z = v_0 = v_6$.

Walks on digraphs

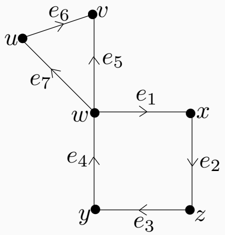
- A walk in a **digraph** is sometimes called a **directed walk**.

Walks on digraphs

- A walk in a **digraph** is sometimes called a **directed walk**.
- In a directed walk, the sequence of edges alone always uniquely determines the path.

- A walk in a **digraph** is sometimes called a **directed walk**.
- In a directed walk, the sequence of edges alone always uniquely determines the path. Examples:

G :

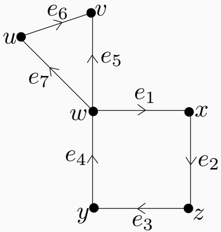


- $e_1e_2e_3e_4e_7e_6$ is a directed walk in G
- $e_1e_2e_3e_4$ is a closed directed walk in G
- $\cancel{e_7e_6e_5}$ is NOT a directed walk in G

Walks on digraphs

- A walk in a **digraph** is sometimes called a **directed walk**.
- In a directed walk, the sequence of edges alone always uniquely determines the path. Examples:

G :



- $e_1e_2e_3e_4e_7e_6$ is a directed walk in G
- $e_1e_2e_3e_4$ is a closed directed walk in G
- $\cancel{e_7e_6e_5}$ is NOT a directed walk in G

- Note that for a simple graph or digraph (with no loops and no parallel edges), **any** walk is uniquely determined by its sequence of vertices.

Connectivity

Connected Graphs

- A graph is **connected** if every pair of vertices can be connected by a walk (and therefore by a path).

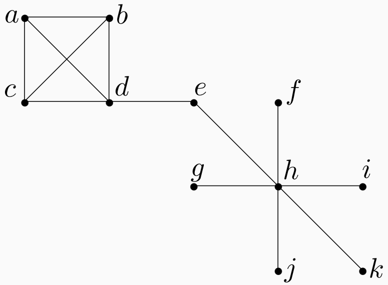
Connected Graphs

- A graph is **connected** if every pair of vertices can be connected by a walk (and therefore by a path).
- A **component** of a graph is a maximal connected subgraph.

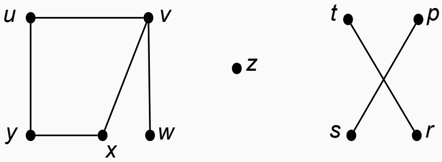
Connected Graphs

- A graph is **connected** if every pair of vertices can be connected by a walk (and therefore by a path).
- A **component** of a graph is a maximal connected subgraph.

Here is an example of a connected graph:

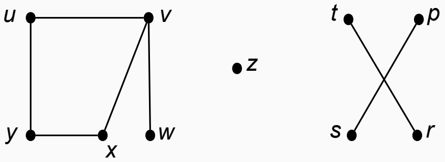


Example



The graph above is **not connected** and has **4 components**:

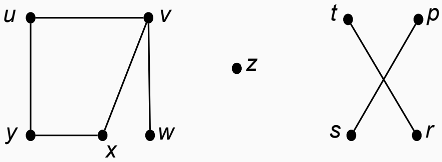
Example



The graph above is **not connected** and has **4 components**:

1. the subgraph with vertex set $\{u, v, w, x, y\}$ and edge set $\{\{u, v\}, \{u, y\}, \{v, w\}, \{v, x\}, \{x, y\}\}$,

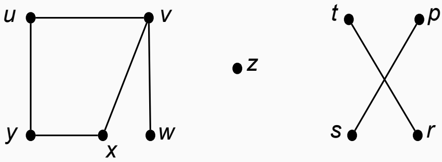
Example



The graph above is **not connected** and has **4 components**:

1. the subgraph with vertex set $\{u, v, w, x, y\}$ and edge set $\{\{u, v\}, \{u, y\}, \{v, w\}, \{v, x\}, \{x, y\}\}$,
2. the subgraph with vertex z and no edges,

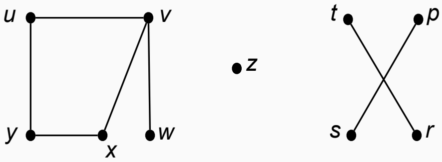
Example



The graph above is **not connected** and has **4 components**:

1. the subgraph with vertex set $\{u, v, w, x, y\}$ and edge set $\{\{u, v\}, \{u, y\}, \{v, w\}, \{v, x\}, \{x, y\}\}$,
2. the subgraph with vertex z and no edges,
3. the subgraph with two vertices t, r and one edge $\{t, r\}$,

Example



The graph above is **not connected** and has **4 components**:

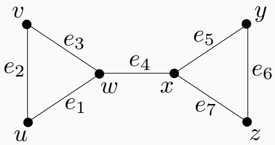
1. the subgraph with vertex set $\{u, v, w, x, y\}$ and edge set $\{\{u, v\}, \{u, y\}, \{v, w\}, \{v, x\}, \{x, y\}\}$,
2. the subgraph with vertex z and no edges,
3. the subgraph with two vertices t, r and one edge $\{t, r\}$,
4. the subgraph with two vertices p, s and one edge $\{p, s\}$.

Bridges and Cut Vertices

- A **bridge** in a connected graph is an edge which on erasure disconnects the graph.

Bridges and Cut Vertices

- A **bridge** in a connected graph is an edge which on erasure disconnects the graph. Examples:

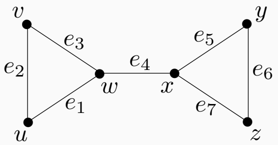
$G:$ 

  - e_4 is a bridge in G
  - e_3 is NOT a bridge in G

Bridges and Cut Vertices

- A **bridge** in a connected graph is an edge which on erasure disconnects the graph. Examples:

G :

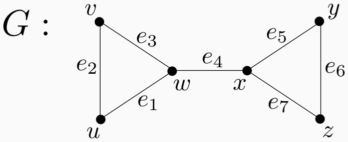


  - e_4 is a bridge in G
  - e_3 is NOT a bridge in G

- A **cut vertex** in a connected graph is a vertex which on erasure disconnects the graph.

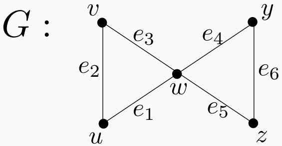
Bridges and Cut Vertices

- A **bridge** in a connected graph is an edge which on erasure disconnects the graph. Examples:



  - e_4 is a bridge in G
  - e_3 is NOT a bridge in G

- A **cut vertex** in a connected graph is a vertex which on erasure disconnects the graph. Examples:



  - w is a cut vertex in G
  - v is NOT a cut vertex in G