Graph Theory and Its Applications

Dr. G.H.J. Lanel

Lecture 1

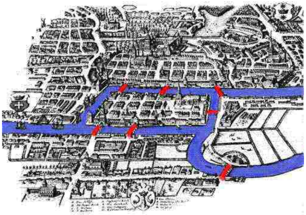
Outline

Outline

1. What is Graph Theory?
   - History
   - At present
2. Introduction to Graph Theory
   - Graphs
   - Propoties of graphs
   - Special graphs
   - Bipartite graphs
   - Cycles
   - Directed graphs

Seven Bridges of Konigsberg

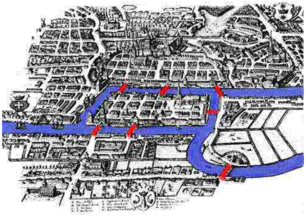
Graph theory started with Euler who was asked to find a nice path across the seven Koningsberg bridges.



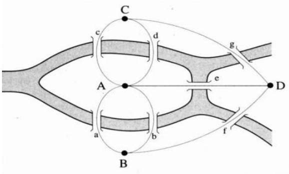
The Eulerian path should cross over each of the seven bridges exactly once.

Seven Bridges of Konigsberg

Graph theory started with Euler who was asked to find a nice path across the seven Koningsberg bridges.

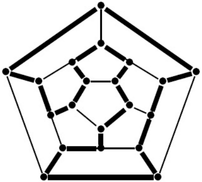


The Eulerian path should cross over each of the seven bridges exactly once.



- In 1859 Sir William Rowan Hamilton developed a game that he sold to a Dublin toy manufacturer.
- The game consisted of a wooden regular dodecahedron with the 20 corner points labeled with the names of prominent cities.
- The objective of the game was to find a cycle along the edges so that each city was on the cycle exactly once.

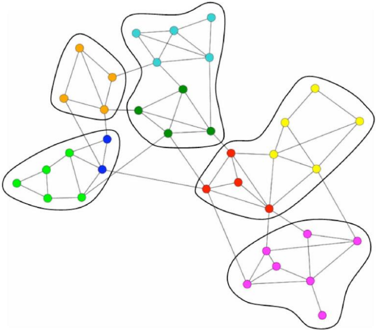
- In 1859 Sir William Rowan Hamilton developed a game that he sold to a Dublin toy manufacturer.
- The game consisted of a wooden regular dodecahedron with the 20 corner points labeled with the names of prominent cities.
- The objective of the game was to find a cycle along the edges so that each city was on the cycle exactly once.



But now graph theory is used for finding communities in networks

where we want to detect hierarchies of substructures.

But now graph theory is used for finding communities in networks



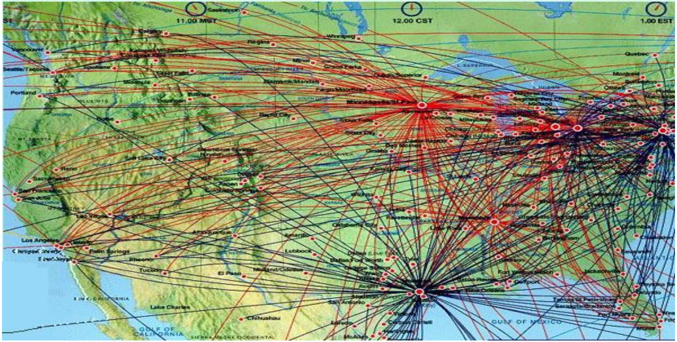
where we want to detect hierarchies of substructures.

A transportation network

and their sizes can become quite big ...

A transportation network

and their sizes can become quite big ...

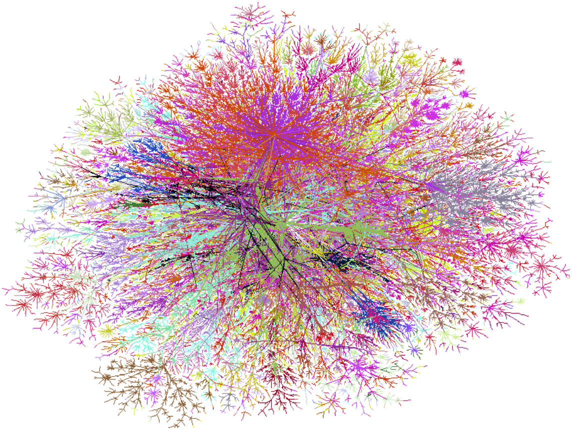


Internet

and their sizes can become quite complicate ...

Internet

and their sizes can become quite complicate ...



Ordering

It is also used for ranking (ordering) hyperlinks,

Ordering

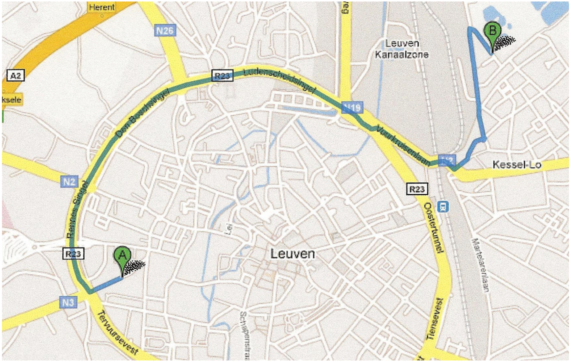
It is also used for ranking (ordering) hyperlinks,

Ordering

It is also used to find the shortest path by using GPS,

Ordering

It is also used to find the shortest path by using GPS,



Outline

Definition of a graph

- Perhaps the most useful object in discrete mathematics (especially for computer science and other applications) is a structure called a graph.
- A graph $G = (V, E)$ is a pair of vertices (or nodes) V and edges E.

Definition of a graph

- Perhaps the most useful object in discrete mathematics (especially for computer science and other applications) is a structure called a graph.
- A graph $G = (V, E)$ is a pair of vertices (or nodes) V and edges E.

For example,

Here $V = \{v_1, v_2, \cdots, v_5\}$ and $E = \{e_1, e_2, \cdots, e_6\}$. An edge $e_k = (v_i, v_j)$ is incident with the vertices v_i and v_j.

Definition of a graph

- Perhaps the most useful object in discrete mathematics (especially for computer science and other applications) is a structure called a graph.
- A graph $G = (V, E)$ is a pair of vertices (or nodes) V and edges E.

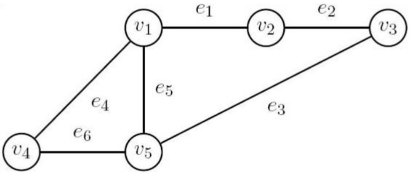
For example,

Here $V = \{v_1, v_2, \cdots, v_5\}$ and $E = \{e_1, e_2, \cdots, e_6\}$. An edge $e_k = (v_i, v_j)$ is incident with the vertices v_i and v_j.

Definition of a graph

- Perhaps the most useful object in discrete mathematics (especially for computer science and other applications) is a structure called a graph.
- A graph $G = (V, E)$ is a pair of vertices (or nodes) V and edges E.

  For example,



  Here $V = \{v_1, v_2, \cdots, v_5\}$ and $E = \{e_1, e_2, \cdots, e_6\}$. An edge $e_k = (v_i, v_j)$ is incident with the vertices v_i and v_j.

Simple graphs

- A self-loop is an edge that joins to an identical vertex.
- A multi-edge is a collection of two or more edges having distinct end vertices.

  For example:

- A simple graph has no self-loops or multiple edges.

Simple graphs

- A self-loop is an edge that joins to an identical vertex.
- A multi-edge is a collection of two or more edges having distinct end vertices.

  For example,

- A simple graph has no self-loops or multiple edges.

Simple graphs

- A self-loop is an edge that joins to an identical vertex.
- A multi-edge is a collection of two or more edges having distinct end vertices.

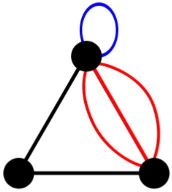
For example,

- A simple graph has no self-loops or multiple edges.

Simple graphs

- A self-loop is an edge that joins to an identical vertex.
- A multi-edge is a collection of two or more edges having distinct end vertices.

  For example,



- A simple graph has no self-loops or multiple edges.

Simple graphs

- The degree $d(v)$ of a vertex $v \in V$ is the number of edges incident to v.
- **Proposition**: Let $G = (V, E)$ be a graph. Then
$$\sum_{v \in V} d(v) = 2|E|.$$
- **Corollary**: The number of vertices of odd degree is even in G. For example,

Simple graphs

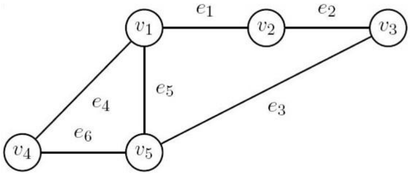
- The degree $d(v)$ of a vertex $v \in V$ is the number of edges incident to v.
- **Proposition**: Let $G = (V, E)$ be a graph. Then
$$\sum_{v \in V} d(v) = 2|E|.$$
- **Corollary**: The number of vertices of odd degree is even in G. For example,

Simple graphs

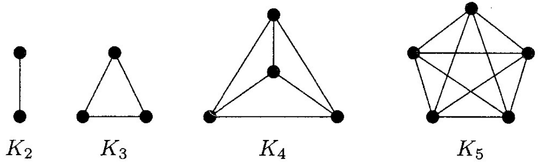
- The degree $d(v)$ of a vertex $v \in V$ is the number of edges incident to v.
- **Proposition**: Let $G = (V, E)$ be a graph. Then
$$\sum_{v \in V} d(v) = 2|E|.$$
- **Corollary**: The number of vertices of odd degree is even in G. For example,

Simple graphs

- The degree $d(v)$ of a vertex $v \in V$ is the number of edges incident to v.
- **Proposition**: Let $G = (V, E)$ be a graph. Then
$$\sum_{v \in V} d(v) = 2|E|.$$
- **Corollary**: The number of vertices of odd degree is even in G. For example,



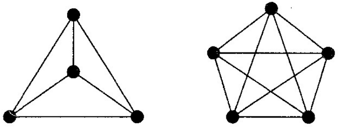
- A complete graph K_n is a simple graph with $n(n-1)/2$ possible edges. When $n = 2, 3, 4, 5$, we have the following graphs.



- A k-regular graph is a simple graph with vertices of equal degree k.

- The complete graph K_n is $(n-1)$-regular

- A complete graph K_n is a simple graph with $n(n-1)/2$ possible edges. When $n = 2, 3, 4, 5$, we have the following graphs.

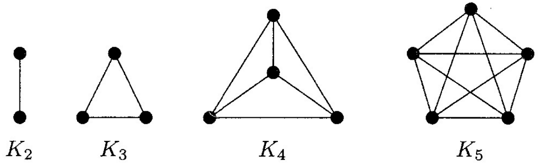


K_2 K_3 K_4 K_5

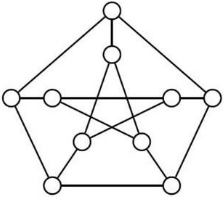
- A k-regular graph is a simple graph with vertices of equal degree k.

- The complete graph K_n is $(n-1)$-regular

- A complete graph K_n is a simple graph with $n(n-1)/2$ possible edges. When $n = 2, 3, 4, 5$, we have the following graphs.

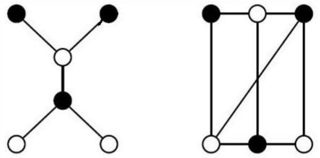


- A *k*-regular graph is a simple graph with vertices of equal degree *k*.



- The complete graph K_n is $(n-1)$-regular

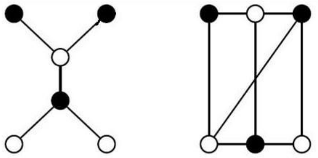
- A bipartite graph is one where $V = V_1 \cup V_2$ such that there are edges only between V_1 and V_2 (the black and white nodes below).



- A complete bipartite graph is one where all edges between V_1 and V_2 are present (i.e. $E = |V_1| \cdot |V_2|$). It is noted as K_{n_1,n_2}, where $n_1 = |V_1|$ and $n_2 = |V_2|$.

- **Question**: When is complete bipartite graph regular?

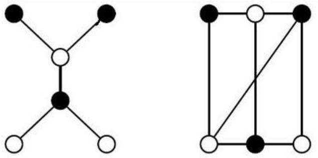
- A bipartite graph is one where $V = V_1 \cup V_2$ such that there are edges only between V_1 and V_2 (the black and white nodes below).



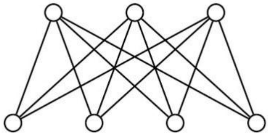
- A complete bipartite graph is one where all edges between V_1 and V_2 are present (i.e. $E = |V_1| \cdot |V_2|$). It is noted as K_{n_1,n_2}, where $n_1 = |V_1|$ and $n_2 = |V_2|$.

- **Question**: When is complete bipartite graph regular?

- A bipartite graph is one where $V = V_1 \cup V_2$ such that there are edges only between V_1 and V_2 (the black and white nodes below).

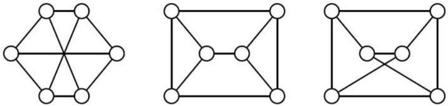


- A complete bipartite graph is one where all edges between V_1 and V_2 are present (i.e. $E = |V_1| \cdot |V_2|$). It is noted as K_{n_1,n_2}, where $n_1 = |V_1|$ and $n_2 = |V_2|$.



- **Question**: When is complete bipartite graph regular?

- **Question**: Which graph is bipartite?.

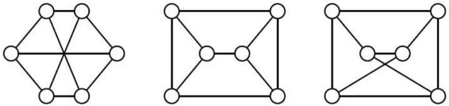


- It suffices to find 2 colors that separate the edges as below,

- The second example is not bipartite because it has a triangle.

- Proposition: A graph is bipartite if and only if it has no cycles of odd length.

- **Question**: Which graph is bipartite?.

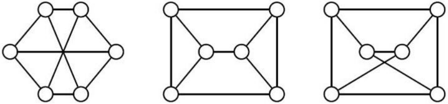


- It suffices to find 2 colors that separate the edges as below,

- The second example is not bipartite because it has a triangle.

- Proposition: A graph is bipartite if and only if it has no cycles of odd length

- **Question**: Which graph is bipartite?.

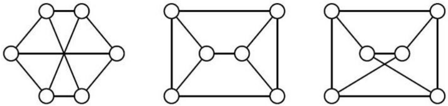


- It suffices to find 2 colors that separate the edges as below,

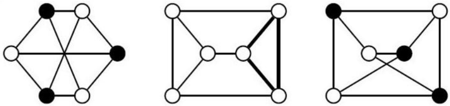
- The second example is not bipartite because it has a triangle.

- **Proposition**: A graph is bipartite if and only if it has no cycles of odd length.

- **Question**: Which graph is bipartite?.



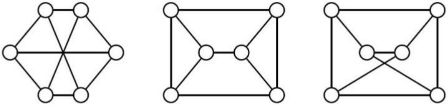
- It suffices to find 2 colors that separate the edges as below,



- The second example is not bipartite because it has a triangle.

- **Proposition**: A graph is bipartite if and only if it has no cycles of odd length.

- **Question**: Which graph is bipartite?.



- It suffices to find 2 colors that separate the edges as below,

- The second example is not bipartite because it has a triangle.

- **Proposition**: A graph is bipartite if and only if it has no cycles of odd length.

- A walk of length k from vertex v_0 to vertex v_k is a non-empty graph $W = (V_1, E_1)$ of the form

$$V_1 = \{v_0, v_1 \cdots , v_k\}, E_1 = \{(v_0, v_1), \cdots , (v_{k-1}, v_k)\},$$

where edge j connects vertices $j - 1$ and j (i.e. $|V_1| = |E_1| + 1$).

- A trail is a walk with all different edges.
- A walk or trail is closed when $v_0 = v_k$.
- A circuit is a closed trail.
- A path is a trail with all different vertices.
- A cycle is a closed path.

- A walk of length k from vertex v_0 to vertex v_k is a non-empty graph $W = (V_1, E_1)$ of the form

$$V_1 = \{v_0, v_1 \cdots, v_k\}, E_1 = \{(v_0, v_1), \cdots, (v_{k-1}, v_k)\},$$

where edge j connects vertices $j - 1$ and j (i.e. $|V_1| = |E_1| + 1$).

- A trail is a walk with all different edges.
- A walk or trail is closed when $v_0 = v_k$.
- A circuit is a closed trail.
- A path is a trail with all different vertices.
- A cycle is a closed path.

- A walk of length k from vertex v_0 to vertex v_k is a non-empty graph $W = (V_1, E_1)$ of the form

$$V_1 = \{v_0, v_1 \cdots, v_k\}, E_1 = \{(v_0, v_1), \cdots, (v_{k-1}, v_k)\},$$

where edge j connects vertices $j-1$ and j (i.e. $|V_1| = |E_1| + 1$).

- A trail is a walk with all different edges.
- A walk or trail is closed when $v_0 = v_k$.
- A circuit is a closed trail.
- A path is a trail with all different vertices.
- A cycle is a closed path.

- A walk of length k from vertex v_0 to vertex v_k is a non-empty graph $W = (V_1, E_1)$ of the form

$$V_1 = \{v_0, v_1 \cdots, v_k\}, E_1 = \{(v_0, v_1), \cdots, (v_{k-1}, v_k)\},$$

where edge j connects vertices $j - 1$ and j (i.e. $|V_1| = |E_1| + 1$).

- A trail is a walk with all different edges.
- A walk or trail is closed when $v_0 = v_k$.
- A circuit is a closed trail.
- A path is a trail with all different vertices.
- A cycle is a closed path.

- A walk of length k from vertex v_0 to vertex v_k is a non-empty graph $W = (V_1, E_1)$ of the form

$$V_1 = \{v_0, v_1 \cdots, v_k\}, E_1 = \{(v_0, v_1), \cdots, (v_{k-1}, v_k)\},$$

  where edge j connects vertices $j - 1$ and j (i.e. $|V_1| = |E_1| + 1$).

- A trail is a walk with all different edges.

- A walk or trail is closed when $v_0 = v_k$.

- A circuit is a closed trail.

- A path is a trail with all different vertices.

- A cycle is a closed path.

- A walk of length k from vertex v_0 to vertex v_k is a non-empty graph $W = (V_1, E_1)$ of the form

$$V_1 = \{v_0, v_1 \cdots, v_k\}, E_1 = \{(v_0, v_1), \cdots, (v_{k-1}, v_k)\},$$

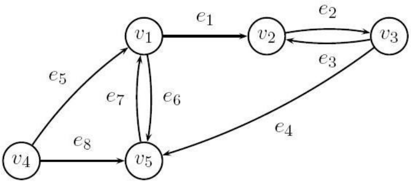
where edge j connects vertices $j - 1$ and j (i.e. $|V_1| = |E_1| + 1$).

- A trail is a walk with all different edges.
- A walk or trail is closed when $v_0 = v_k$.
- A circuit is a closed trail.
- A path is a trail with all different vertices.
- A cycle is a closed path.

In a directed graph or digraph, each edge has a direction.

Each vertex v has an in-degree $d_{in}(v)$ and an out-degree $d_{out}(v)$.

In a directed graph or digraph, each edge has a direction.



Each vertex v has an in-degree $d_{in}(v)$ and an out-degree $d_{out}(v)$.

Acyclic graphs

A directed acyclic graph is a graph without cycles.

Proposition: Every directed acyclic graph contains at least one vertex with zero in-degree.

Proof: By contradiction, assume $d_{in}(v) > 0$ for all vertices, then each vertex i has a predecessor $p(i)$ such that $(v_{p(i)}, v_i) \in E$.

Start from an arbitrary v_0 to form a list of predecessors as below,

Since $|V|$ is bounded, one must eventually return to a vertex that was already visited; hence there is a cycle.

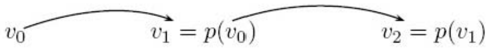
Acyclic graphs

A directed acyclic graph is a graph without cycles.

Proposition: Every directed acyclic graph contains at least one vertex with zero in-degree.

Proof: By contradiction, assume $d_{in}(v) > 0$ for all vertices, then each vertex i has a predecessor $p(i)$ such that $(v_{p(i)}, v_i) \in E$.

Start from an arbitrary v_0 to form a list of predecessors as below,



Since $|V|$ is bounded, one must eventually return to a vertex that was already visited; hence there is a cycle.

End!