Graph Theory Definitions

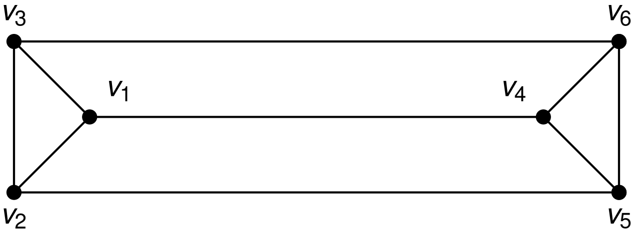
John E. Mitchell

Department of Mathematical Sciences
RPI, Troy, NY 12180 USA

January 22, 2013

Outline

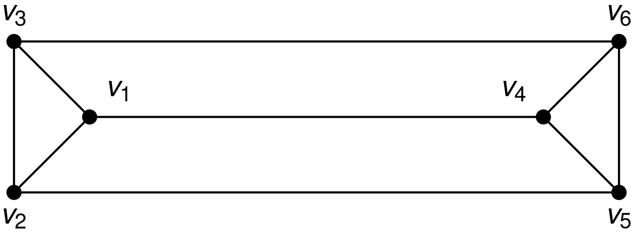
Fundamentals



- A **graph** $G = (V, E)$ is made up of
  1. A finite nonempty set $V = \{v_1, \ldots, v_n\}$ of **nodes** or **vertices**.
  2. A set $E = \{e_1, \ldots, e_m\}$ of **edges**, where each element e_j is a subset of V of size 2.
- **Simple graphs:** A simple graph is one that contains
  1. no **parallel edges**, that is, there is at most one edge between any pair of vertices.
  2. no **loops**, that is, no edges of the form (i, i).

Unless otherwise stated, we will be working with simple graphs.

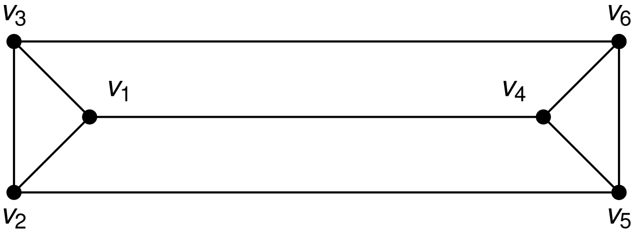
Fundamentals



- A **graph** $G = (V, E)$ is made up of
  1. A finite nonempty set $V = \{v_1, \ldots, v_n\}$ of **nodes** or **vertices**.
  2. A set $E = \{e_1, \ldots, e_m\}$ of **edges**, where each element e_j is a subset of V of size 2.
- **Simple graphs:** A simple graph is one that contains
  1. no **parallel edges**, that is, there is at most one edge between any pair of vertices.
  2. no **loops**, that is, no edges of the form (i, i).

Unless otherwise stated, we will be working with simple graphs.

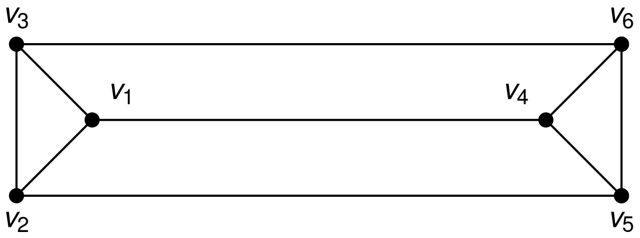
Fundamentals



- A **graph** $G = (V, E)$ is made up of
  1. A finite nonempty set $V = \{v_1, \ldots, v_n\}$ of **nodes** or **vertices**.
  2. A set $E = \{e_1, \ldots, e_m\}$ of **edges**, where each element e_j is a subset of V of size 2.
- **Simple graphs:** A simple graph is one that contains
  1. no **parallel edges**, that is, there is at most one edge between any pair of vertices.
  2. no **loops**, that is, no edges of the form (i, i).

Unless otherwise stated, we will be working with simple graphs.

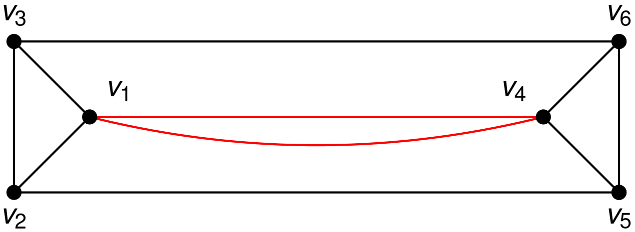
Fundamentals



- A **graph** $G = (V, E)$ is made up of
  1. A finite nonempty set $V = \{v_1, \ldots, v_n\}$ of **nodes** or **vertices**.
  2. A set $E = \{e_1, \ldots, e_m\}$ of **edges**, where each element e_j is a subset of V of size 2.
- **Simple graphs:** A simple graph is one that contains
  1. no **parallel edges**, that is, there is at most one edge between any pair of vertices.
  2. no **loops**, that is, no edges of the form (i, i).

Unless otherwise stated, we will be working with simple graphs.

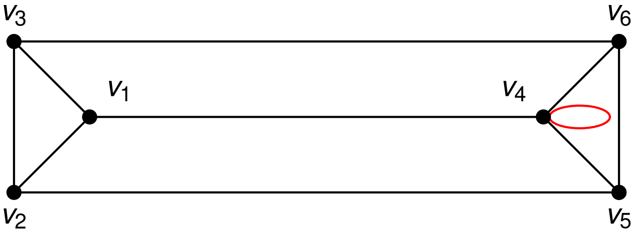
Fundamentals



- A **graph** $G = (V, E)$ is made up of
  1. A finite nonempty set $V = \{v_1, \ldots, v_n\}$ of **nodes** or **vertices**.
  2. A set $E = \{e_1, \ldots, e_m\}$ of **edges**, where each element e_j is a subset of V of size 2.
- **Simple graphs:** A simple graph is one that contains
  1. no **parallel edges**, that is, there is at most one edge between any pair of vertices.
  2. no **loops**, that is, no edges of the form (i, i).

Unless otherwise stated, we will be working with simple graphs.

Fundamentals



- A **graph** $G = (V, E)$ is made up of
  1. A finite nonempty set $V = \{v_1, \ldots, v_n\}$ of **nodes** or **vertices**.
  2. A set $E = \{e_1, \ldots, e_m\}$ of **edges**, where each element e_j is a subset of V of size 2.
- **Simple graphs:** A simple graph is one that contains
  1. no **parallel edges**, that is, there is at most one edge between any pair of vertices.
  2. no **loops**, that is, no edges of the form (i, i).

Unless otherwise stated, we will be working with simple graphs.

Bipartite graphs

A graph $G = (V, E)$ is **bipartite** if there is a partition of the vertices V into two disjoint sets V_1 and V_2 such that each edge joins a node in V_1 to a node in V_2.

The **assignment problem**: Given two sets S and T of equal size, pair off each element of S with an element of T at minimum total cost, where there is a cost for each possible pairing, and the total cost is the sum of the costs of the pairings that are used.

Bipartite graphs

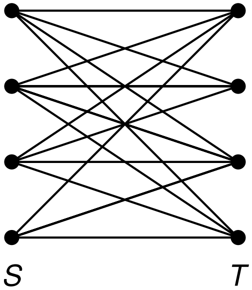
A graph $G = (V, E)$ is **bipartite** if there is a partition of the vertices V into two disjoint sets V_1 and V_2 such that each edge joins a node in V_1 to a node in V_2.

The **assignment problem**: Given two sets S and T of equal size, pair off each element of S with an element of T at minimum total cost, where there is a cost for each possible pairing, and the total cost is the sum of the costs of the pairings that are used.

Bipartite graphs

A graph $G = (V, E)$ is **bipartite** if there is a partition of the vertices V into two disjoint sets V_1 and V_2 such that each edge joins a node in V_1 to a node in V_2.

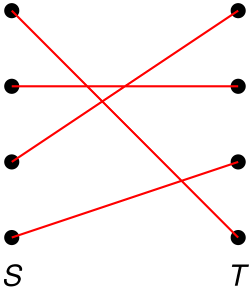
The **assignment problem**: Given two sets S and T of equal size, pair off each element of S with an element of T at minimum total cost, where there is a cost for each possible pairing, and the total cost is the sum of the costs of the pairings that are used.



Bipartite graphs

A graph $G = (V, E)$ is **bipartite** if there is a partition of the vertices V into two disjoint sets V_1 and V_2 such that each edge joins a node in V_1 to a node in V_2.

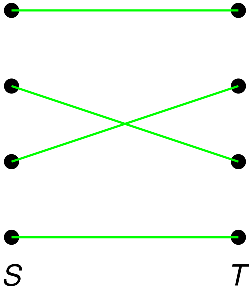
The **assignment problem**: Given two sets S and T of equal size, pair off each element of S with an element of T at minimum total cost, where there is a cost for each possible pairing, and the total cost is the sum of the costs of the pairings that are used.



Bipartite graphs

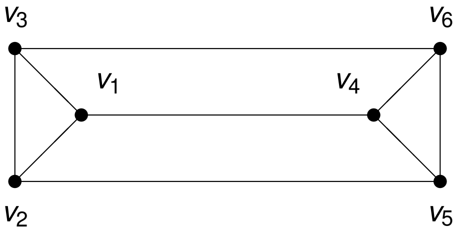
A graph $G = (V, E)$ is **bipartite** if there is a partition of the vertices V into two disjoint sets V_1 and V_2 such that each edge joins a node in V_1 to a node in V_2.

The **assignment problem**: Given two sets S and T of equal size, pair off each element of S with an element of T at minimum total cost, where there is a cost for each possible pairing, and the total cost is the sum of the costs of the pairings that are used.



Outline

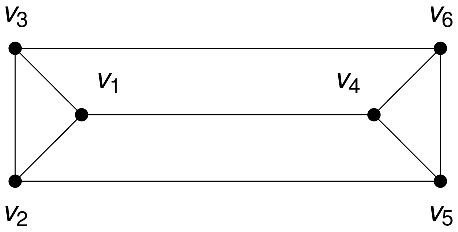
Adjacency



- An edge $e_{ij} = (i, j)$ **meets** or is **incident to** the vertices i and j in V. If such an edge exists, the two vertices are **adjacent**. For example, vertices v_1 and v_4 are adjacent, but vertices v_1 and v_6 are not adjacent.
- A graph can be represented by a **vertex-edge incidence matrix** A with entries given by

$$a_{ij} = \begin{cases} 1 & \text{if edge } e_j \text{ is adjacent to vertex i} \\ 0 & \text{otherwise} \end{cases}$$

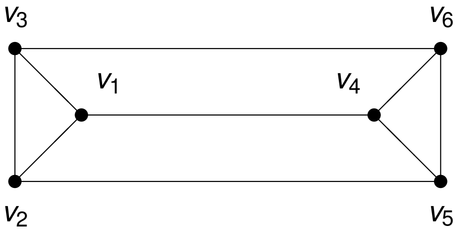
Adjacency



- An edge $e_{ij} = (i, j)$ **meets** or is **incident to** the vertices i and j in V. If such an edge exists, the two vertices are **adjacent**. For example, vertices v_1 and v_4 are adjacent, but vertices v_1 and v_6 are not adjacent.
- A graph can be represented by a **vertex-edge incidence matrix** A with entries given by

$$a_{ij} = \begin{cases} 1 & \text{if edge } e_j \text{ is adjacent to vertex i} \\ 0 & \text{otherwise} \end{cases}$$

Incidence matrix example



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

Edges are listed in the order $(1,2)$, $(1,3)$, $(1,4)$, $(2,3)$, $(2,5)$, $(3,6)$, $(4,5)$, $(4,6)$, and $(5,6)$. Notice that every column of the incidence matrix contains exactly two "ones".

Vertex degree

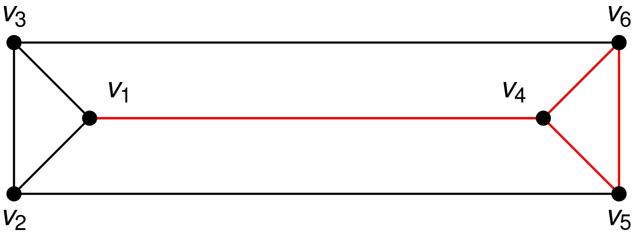
- The number of edges incident to a node is called the **degree** of the node. This is equal to the number of “ones” in the corresponding row of the incidence matrix. Every node in the example graph has degree 3.
- A graph with m vertices is called **complete** if it contains all possible edges, so the degree of every vertex is then $m - 1$. This graph is written K_m.

Vertex degree

- The number of edges incident to a node is called the **degree** of the node. This is equal to the number of “ones” in the corresponding row of the incidence matrix. Every node in the example graph has degree 3.
- A graph with m vertices is called **complete** if it contains all possible edges, so the degree of every vertex is then $m - 1$. This graph is written K_m.

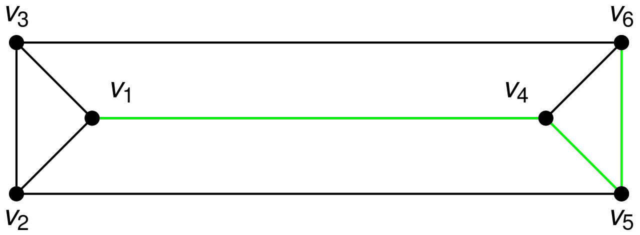
Outline

Paths



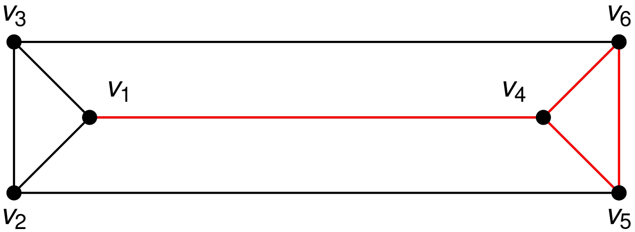
- A node sequence $v_0, v_1, \ldots, v_k$ with $k \geq 1$ is a $v_0 - v_k$ **walk** if $(v_{i-1}, v_i) \in E$ for $i = 1, \ldots, k$. Node v_0 is the origin of the walk and node v_k is the destination. Nodes $\{v_1, \ldots, v_{k-1}\}$ are intermediate nodes. The walk has **length** k. The walk can also be represented by its edges: $e_1, \ldots, e_k$, where $e_i = (v_{i-1}, v_i)$. Eg, we have the walk v_1, v_4, v_5, v_6, v_4.
- A walk is called a **path** if there are no node repetitions. Eg, we have the path v_1, v_4, v_5, v_6.

Paths



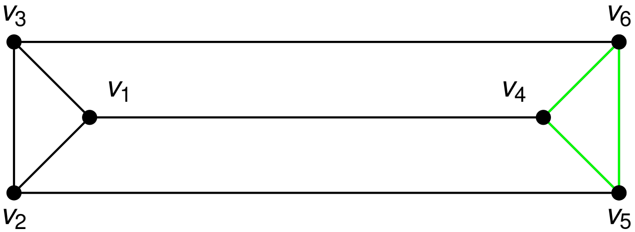
- A node sequence $v_0, v_1, \ldots, v_k$ with $k \geq 1$ is a $v_0 - v_k$ **walk** if $(v_{i-1}, v_i) \in E$ for $i = 1, \ldots, k$. Node v_0 is the origin of the walk and node v_k is the destination. Nodes $\{v_1, \ldots, v_{k-1}\}$ are intermediate nodes. The walk has **length** k. The walk can also be represented by its edges: $e_1, \ldots, e_k$, where $e_i = (v_{i-1}, v_i)$. Eg, we have the walk v_1, v_4, v_5, v_6, v_4.
- A walk is called a **path** if there are no node repetitions. Eg, we have the path v_1, v_4, v_5, v_6.

Cycles



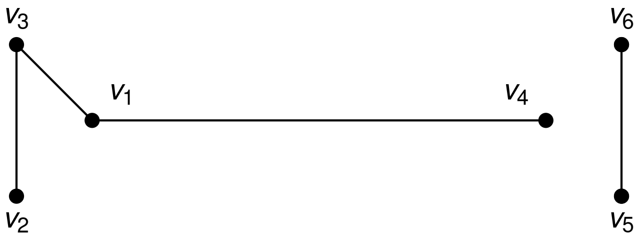
- A $v_0 - v_k$ walk is **closed** if $v_k = v_0$. Eg, we have the closed walk $v_1, v_4, v_5, v_6, v_4, v_1$.
- A closed walk is a **cycle** or **circuit** if $k \geq 3$ and $v_0, v_1, \ldots, v_{k-1}$ is a path. Eg, we have the cycle v_4, v_5, v_6, v_4.
- A graph is **acyclic** if it contains no cycles.
- The **length** of a cycle is the number of edges in the cycle.
  Exercise: Show that a graph is bipartite if and only if it contains no cycles of odd length.

Cycles



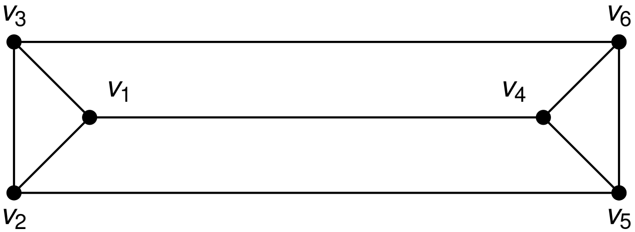
- A $v_0 - v_k$ walk is **closed** if $v_k = v_0$. Eg, we have the closed walk v_1, v_4, v_5, v_6, v_4, v_1.
- A closed walk is a **cycle** or **circuit** if $k \geq 3$ and $v_0, v_1, \ldots, v_{k-1}$ is a path. Eg, we have the cycle v_4, v_5, v_6, v_4.
- A graph is **acyclic** if it contains no cycles.
- The **length** of a cycle is the number of edges in the cycle.
  Exercise: Show that a graph is bipartite if and only if it contains no cycles of odd length.

Cycles



- A $v_0 - v_k$ walk is **closed** if $v_k = v_0$. Eg, we have the closed walk $v_1, v_4, v_5, v_6, v_4, v_1$.
- A closed walk is a **cycle** or **circuit** if $k \geq 3$ and $v_0, v_1, \ldots, v_{k-1}$ is a path. Eg, we have the cycle v_4, v_5, v_6, v_4.
- A graph is **acyclic** if it contains no cycles.
- The **length** of a cycle is the number of edges in the cycle. **Exercise**: Show that a graph is bipartite if and only if it contains no cycles of odd length.

Cycles



- A $v_0 - v_k$ walk is **closed** if $v_k = v_0$. Eg, we have the closed walk $v_1, v_4, v_5, v_6, v_4, v_1$.
- A closed walk is a **cycle** or **circuit** if $k \geq 3$ and $v_0, v_1, \ldots, v_{k-1}$ is a path. Eg, we have the cycle v_4, v_5, v_6, v_4.
- A graph is **acyclic** if it contains no cycles.
- The **length** of a cycle is the number of edges in the cycle.

  Exercise: Show that a graph is bipartite if and only if it contains no cycles of odd length.

Outline

Connected graphs

- Two vertices u and v in V are **connected** in $G = (V, E)$ if there exists a (u, v)-path in G.
- Two vertices are in the same **component** of G if they are connected. Notice that a graph can be partitioned into its components.
- $G = (V, E)$ is **connected** if it has exactly one component. The graph in the example is connected.

Connected graphs

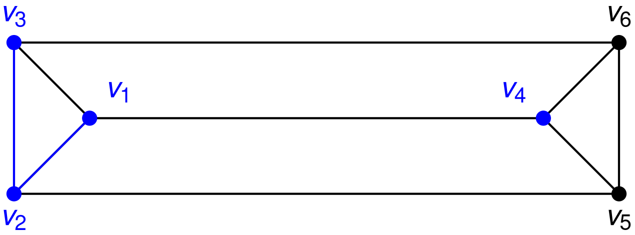
- Two vertices u and v in V are **connected** in $G = (V, E)$ if there exists a (u, v)-path in G.
- Two vertices are in the same **component** of G if they are connected. Notice that a graph can be partitioned into its components.
- $G = (V, E)$ is **connected** if it has exactly one component. The graph in the example is connected.

Connected graphs

- Two vertices u and v in V are **connected** in $G = (V, E)$ if there exists a (u, v)-path in G.
- Two vertices are in the same **component** of G if they are connected. Notice that a graph can be partitioned into its components.
- $G = (V, E)$ is **connected** if it has exactly one component. The graph in the example is connected.

Outline

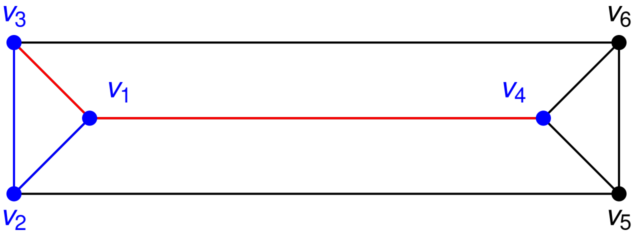
Subgraphs



For $U \subseteq V$, let $E(U) := \{(i,j) \in E : i \in U \text{ and } j \in U\}$, so $E(U)$ is the set of edges with both endpoints in U.

- The graph $G' = (V', E')$ is a **subgraph** of G if $V' \subseteq V$ and $E' \subseteq E$.
- G' is the subgraph **induced** by V' if $E' = E(V')$.
- G' is a **spanning** subgraph if $V' = V$.

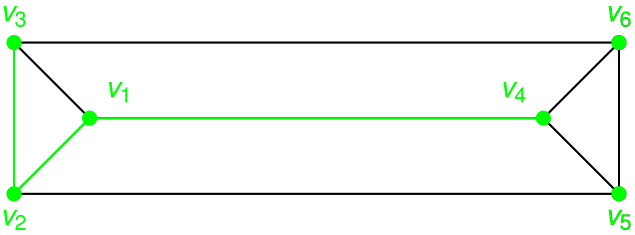
Subgraphs



For $U \subseteq V$, let $E(U) := \{(i,j) \in E : i \in U \text{ and } j \in U\}$, so $E(U)$ is the set of edges with both endpoints in U.

- The graph $G' = (V', E')$ is a **subgraph** of G if $V' \subseteq V$ and $E' \subseteq E$.
- G' is the subgraph **induced** by V' if $E' = E(V')$.
- G' is a **spanning** subgraph if $V' = V$.

Subgraphs

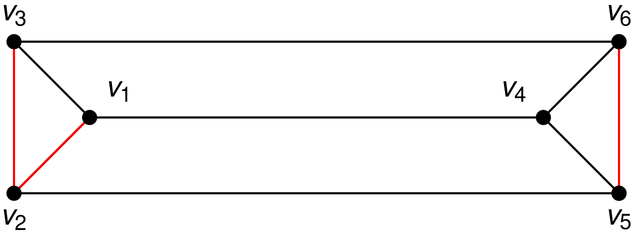


For $U \subseteq V$, let $E(U) := \{(i,j) \in E : i \in U \text{ and } j \in U\}$, so $E(U)$ is the set of edges with both endpoints in U.

- The graph $G' = (V', E')$ is a **subgraph** of G if $V' \subseteq V$ and $E' \subseteq E$.
- G' is the subgraph **induced** by V' if $E' = E(V')$.
- G' is a **spanning** subgraph if $V' = V$.

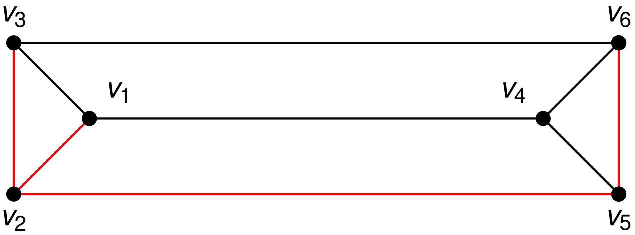
Outline

Trees



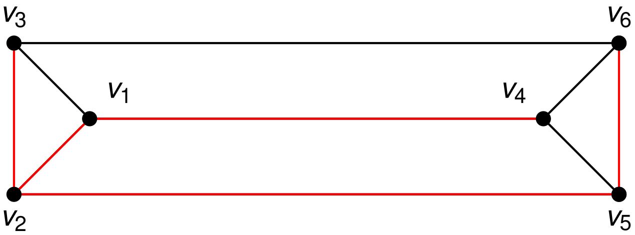
- An acyclic graph is called a **forest**.
- A connected forest is a **tree**.
- A **spanning tree** of $G = (V, E)$ is a spanning subgraph that is a tree.

Trees



- An acyclic graph is called a **forest**.
- A connected forest is a **tree**.
- A **spanning tree** of $G = (V, E)$ is a spanning subgraph that is a tree.

Trees



- An acyclic graph is called a **forest**.
- A connected forest is a **tree**.
- A **spanning tree** of $G = (V, E)$ is a spanning subgraph that is a tree.

Tree exercises

- **Exercise**: Let $G = (V, E)$ be a graph on m nodes. Show that the following statements are equivalent:
  1. G is a spanning tree.
  2. There is a unique path between each pair of nodes.
  3. G contains $m - 1$ edges and is connected.
  4. G contains $m - 1$ edges and is acyclic.
  5. G is acyclic and connected.
- **Exercise**: Show that if $G = (V, E)$ is a spanning tree and $e' \notin E$ then $G' := (V, E \cup e')$ contains exactly one cycle. (This cycle is the **fundamental cycle** for this spanning tree and edge.) **Exercise**: Show that if C is the edge set of the fundamental cycle and if the edge $e^* \in C$ then the graph $G^* := (V, E \cup e' \setminus e^*)$ is also a spanning tree.

Tree exercises

- **Exercise**: Let $G = (V, E)$ be a graph on m nodes. Show that the following statements are equivalent:
  1. G is a spanning tree.
  2. There is a unique path between each pair of nodes.
  3. G contains $m - 1$ edges and is connected.
  4. G contains $m - 1$ edges and is acyclic.
  5. G is acyclic and connected.
- **Exercise**: Show that if $G = (V, E)$ is a spanning tree and $e' \notin E$ then $G' := (V, E \cup e')$ contains exactly one cycle. (This cycle is the **fundamental cycle** for this spanning tree and edge.)
  Exercise: Show that if C is the edge set of the fundamental cycle and if the edge $e^* \in C$ then the graph $G^* := (V, E \cup e' \setminus e^*)$ is also a spanning tree.

Tree exercises

- **Exercise**: Let $G = (V, E)$ be a graph on m nodes. Show that the following statements are equivalent:
  1. G is a spanning tree.
  2. There is a unique path between each pair of nodes.
  3. G contains $m - 1$ edges and is connected.
  4. G contains $m - 1$ edges and is acyclic.
  5. G is acyclic and connected.
- **Exercise**: Show that if $G = (V, E)$ is a spanning tree and $e' \notin E$ then $G' := (V, E \cup e')$ contains exactly one cycle. (This cycle is the **fundamental cycle** for this spanning tree and edge.)
  Exercise: Show that if C is the edge set of the fundamental cycle and if the edge $e^* \in C$ then the graph $G^* := (V, E \cup e' \setminus e^*)$ is also a spanning tree.

Tree exercises

- **Exercise**: Let $G = (V, E)$ be a graph on m nodes. Show that the following statements are equivalent:
  1. G is a spanning tree.
  2. There is a unique path between each pair of nodes.
  3. G contains $m - 1$ edges and is connected.
  4. G contains $m - 1$ edges and is acyclic.
  5. G is acyclic and connected.
- **Exercise**: Show that if $G = (V, E)$ is a spanning tree and $e' \notin E$ then $G' := (V, E \cup e')$ contains exactly one cycle. (This cycle is the **fundamental cycle** for this spanning tree and edge.)
  Exercise: Show that if C is the edge set of the fundamental cycle and if the edge $e^* \in C$ then the graph $G^* := (V, E \cup e' \setminus e^*)$ is also a spanning tree.

Tree exercises

- **Exercise**: Let $G = (V, E)$ be a graph on m nodes. Show that the following statements are equivalent:
  1. G is a spanning tree.
  2. There is a unique path between each pair of nodes.
  3. G contains $m - 1$ edges and is connected.
  4. G contains $m - 1$ edges and is acyclic.
  5. G is acyclic and connected.
- **Exercise**: Show that if $G = (V, E)$ is a spanning tree and $e' \notin E$ then $G' := (V, E \cup e')$ contains exactly one cycle. (This cycle is the **fundamental cycle** for this spanning tree and edge.) **Exercise**: Show that if C is the edge set of the fundamental cycle and if the edge $e^* \in C$ then the graph $G^* := (V, E \cup e' \setminus e^*)$ is also a spanning tree.

Tree exercises

- **Exercise**: Let $G = (V, E)$ be a graph on m nodes. Show that the following statements are equivalent:
  1. G is a spanning tree.
  2. There is a unique path between each pair of nodes.
  3. G contains $m - 1$ edges and is connected.
  4. G contains $m - 1$ edges and is acyclic.
  5. G is acyclic and connected.
- **Exercise**: Show that if $G = (V, E)$ is a spanning tree and $e' \notin E$ then $G' := (V, E \cup e')$ contains exactly one cycle. (This cycle is the **fundamental cycle** for this spanning tree and edge.) **Exercise**: Show that if C is the edge set of the fundamental cycle and if the edge $e^* \in C$ then the graph $G^* := (V, E \cup e' \setminus e^*)$ is also a spanning tree.

Tree exercises

- **Exercise**: Let $G = (V, E)$ be a graph on m nodes. Show that the following statements are equivalent:
  1. G is a spanning tree.
  2. There is a unique path between each pair of nodes.
  3. G contains $m - 1$ edges and is connected.
  4. G contains $m - 1$ edges and is acyclic.
  5. G is acyclic and connected.
- **Exercise**: Show that if $G = (V, E)$ is a spanning tree and $e' \notin E$ then $G' := (V, E \cup e')$ contains exactly one cycle. (This cycle is the **fundamental cycle** for this spanning tree and edge.)
  Exercise: Show that if C is the edge set of the fundamental cycle and if the edge $e^* \in C$ then the graph $G^* := (V, E \cup e' \setminus e^*)$ is also a spanning tree.

Directed graphs

- A **directed graph** or **digraph** $D = (V, A)$ consists of a finite set V of vertices (or nodes) and a set $A = \{e_1, \ldots, e_m\}$ whose elements are *ordered* subsets of V of size 2 called **arcs.**
- The **node-arc incidence matrix** of a digraph D with m nodes and n arcs is the $m \times n$ matrix A with

$$a_{ij} = \begin{cases} 1 & \text{if } e_j = (k, i) \text{ for some } k \in V \setminus i \\ -1 & \text{if } e_j = (i, k) \text{ for some } k \in V \setminus i \\ 0 & \text{otherwise} \end{cases}$$

  Note that the rows of A sum to zero, so they are linearly dependent.
- A **directed walk**, **path**, or **cycle** is a walk, path, or cycle in the underlying undirected graph where the direction of the arcs agrees with the direction of the walk, path or cycle.

Directed graphs

- A **directed graph** or **digraph** $D = (V, A)$ consists of a finite set V of vertices (or nodes) and a set $A = \{e_1, \ldots, e_m\}$ whose elements are *ordered* subsets of V of size 2 called **arcs.**
- The **node-arc incidence matrix** of a digraph D with m nodes and n arcs is the $m \times n$ matrix A with

$$a_{ij} = \begin{cases} 1 & \text{if } e_j = (k, i) \text{ for some } k \in V \setminus i \\ -1 & \text{if } e_j = (i, k) \text{ for some } k \in V \setminus i \\ 0 & \text{otherwise} \end{cases}$$

  Note that the rows of A sum to zero, so they are linearly dependent.
- A **directed walk**, **path**, or **cycle** is a walk, path, or cycle in the underlying undirected graph where the direction of the arcs agrees with the direction of the walk, path or cycle.

Directed graphs

- A **directed graph** or **digraph** $D = (V, A)$ consists of a finite set V of vertices (or nodes) and a set $A = \{e_1, \ldots, e_m\}$ whose elements are *ordered* subsets of V of size 2 called **arcs**.
- The **node-arc incidence matrix** of a digraph D with m nodes and n arcs is the $m \times n$ matrix A with

$$a_{ij} = \begin{cases} 1 & \text{if } e_j = (k, i) \text{ for some } k \in V \setminus i \\ -1 & \text{if } e_j = (i, k) \text{ for some } k \in V \setminus i \\ 0 & \text{otherwise} \end{cases}$$

  Note that the rows of A sum to zero, so they are linearly dependent.
- A **directed walk**, **path**, or **cycle** is a walk, path, or cycle in the underlying undirected graph where the direction of the arcs agrees with the direction of the walk, path or cycle.