

# CM0884 Graph Theory

Andrés Sicard-Ramírez

Universidad EAFIT

Semester 2018-1

# Administrative Information

---

Course web page

<https://asr.github.io/courses/cm0884-graph-theory/>

Exams, bibliography, etc.

See course web page.

Textbook

Diestel, Reinhard (2017). Graph Theory. 5th ed. Springer.

# The Basics

# Preliminaries

---

## Convention

The set of natural numbers, denoted  $\mathbb{N}$ , includes the zero.

# Preliminaries

---

## Convention

The set of natural numbers, denoted  $\mathbb{N}$ , includes the zero.

## Notation

Let  $A$  be a set. We denote the set of all  $k$ -subsets of  $A$  by  $[A]^k$ .

# Preliminaries

---

## Convention

The set of natural numbers, denoted  $\mathbb{N}$ , includes the zero.

## Notation

Let  $A$  be a set. We denote the set of all  $k$ -subsets of  $A$  by  $[A]^k$ .

## Definition

'A set  $\mathcal{A} = \{A_1, \dots, A_k\}$  of disjoint subsets of a set  $A$  is a **partition** of  $A$  if the union  $\bigcup A$  of all the sets  $A_i \in \mathcal{A}$  is  $A$  and  $A_i \neq \emptyset$  for every  $i$ .' (Diestel 2017, p. 1)

# Graphs

---

## Definition

A **graph** (*grafo*) is an **order** pair  $G = (V, E)$  of disjoint **sets** such that  $E \subseteq [V]^2$ .

# Graphs

---

## Definition

A **graph** (*grafo*) is an **order** pair  $G = (V, E)$  of disjoint **sets** such that  $E \subseteq [V]^2$ .

## Definition

The **vertices** and the **edges** (*aristas*) of a graph  $G = (V, E)$  are the elements of  $V$  and  $E$ , respectively.

## Notation

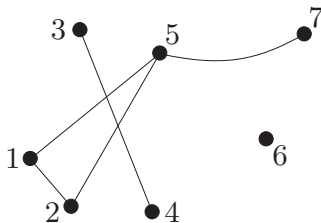
The vertex set of a graph  $G$  is denoted  $V(G)$  and its edge set is denoted  $E(G)$ .



# Graphs

## Example

First example of textbook.\*



$$V = \{1, \dots, 7\},$$

$$E = \{\{1, 2\}, \{1, 5\}, \{2, 5\}, \{3, 4\}, \{5, 7\}\}.$$

---

\*Figure source: Diestel (2017, Fig. 1.1.1).

# Graphs

---

## Example

- $G = (V, E)$  where  $V = \{v_1, v_2, v_3\}$  and  $E = [V]^2$ .
- $G = (V, E)$  where  $V = \{v_1, v_2, v_3\}$  and  $E = \emptyset$ .

# Graphs

---

## Definition

The **empty graph** is the graph  $(\emptyset, \emptyset)$  and it is denoted  $\emptyset$ .

# Graphs

---

## Definition

The **empty graph** is the graph  $(\emptyset, \emptyset)$  and it is denoted  $\emptyset$ .

## Discussion

Should or should not the empty-graph be a graph?

# Graphs

---

## Definition

The **empty graph** is the graph  $(\emptyset, \emptyset)$  and it is denoted  $\emptyset$ .

## Discussion

Should or should not the empty-graph be a graph?

In the abstract of an article about this question, Harary and Read (1974) wrote:

*'The graph with no points and no lines is discussed critically. Arguments for and against its official admittance as a graph are presented. This is accompanied by an extensive survey of the literature. Paradoxical properties of the null-graph are noted. No conclusion is reached.'*

# Graphs

---

## Some remarks on our definition of graph

- The edges in our graphs are **undirected** because  $\{v, w\} = \{w, v\}$ .

# Graphs

---

## Some remarks on our definition of graph

- The edges in our graphs are **undirected** because  $\{v, w\} = \{w, v\}$ .
- Since  $\{v, v\} \notin [V]^2$  because  $|\{v, v\}| = |\{v\}| = 1$ , our graphs have no **loops**.

# Graphs

---

## Some remarks on our definition of graph

- The edges in our graphs are **undirected** because  $\{v, w\} = \{w, v\}$ .
- Since  $\{v, v\} \notin [V]^2$  because  $|\{v, v\}| = |\{v\}| = 1$ , our graphs have no **loops**.
- Since the multiplicity of an element in a set is one, our graphs have no **parallel** edges.



# Graphs

---

## Some remarks on our definition of graph

- The edges in our graphs are **undirected** because  $\{v, w\} = \{w, v\}$ .
- Since  $\{v, v\} \notin [V]^2$  because  $|\{v, v\}| = |\{v\}| = 1$ , our graphs have no **loops**.
- Since the multiplicity of an element in a set is one, our graphs have no **parallel** edges.
- A graph with **undirected** edges, **without loops** and **without parallel** edges is also called a **simple** graph in the literature. E.g. (Bondy and Murty 2008).

# Graphs

---

## Some remarks on our definition of graph

- The edges in our graphs are **undirected** because  $\{v, w\} = \{w, v\}$ .
- Since  $\{v, v\} \notin [V]^2$  because  $|\{v, v\}| = |\{v\}| = 1$ , our graphs have no **loops**.
- Since the multiplicity of an element in a set is one, our graphs have no **parallel** edges.
- A graph with **undirected** edges, **without loops** and **without parallel** edges is also called a **simple** graph in the literature. E.g. (Bondy and Murty 2008).
- The sets  $V$  and  $E$  must be disjoint for ruling out 'graphs' like  $V = \{a, b, \{a, b\}\}$  and  $E = \{\{a, b\}\}$ .

# Graphs

---

## Definition

The **order** of graph  $G$ , denoted  $|G|$ , is its number of vertices.

# Graphs

---

## Definition

The **order** of graph  $G$ , denoted  $|G|$ , is its number of vertices.

## Definition

A graph  $G$  is **finite**, **infinite**, **countable** and so on according to  $|G|$ .

# Graphs

---

## Definition

A graph is **trivial** iff  $|G| = 0$  or  $|G| = 1$ .

## Remark

Diestel (2017, p. 16):

*'Sometimes, e.g. to start an induction, trivial graphs can be useful; at other times they form silly counterexamples and become a nuisance. To avoid cluttering the text with non-triviality conditions, we shall mostly treat the trivial graphs, and particularly the empty graph  $\emptyset$ , with generous disregard.'*

# Graphs

---

## Definition

A vertex  $v$  is **incident** with an edge  $e$  if  $v \in e$ , then  $e$  is an edge **at**  $v$ .

# Graphs

---

## Definition

A vertex  $v$  is **incident** with an edge  $e$  if  $v \in e$ , then  $e$  is an edge **at**  $v$ .

## Definition

The two vertices incident with an edge are its **ends** (*puntos finales*).

# Graphs

---

## Notation

An edge  $\{x, y\}$  also will be written as  $xy$  (or  $yx$ ).



# Graphs

---

## Notation

An edge  $\{x, y\}$  also will be written as  $xy$  (or  $yx$ ).

## Definition

If  $x \in X$  and  $y \in Y$ , then  $xy$  is an  **$X$ - $Y$  edge**.

# Graphs

---

## Notation

An edge  $\{x, y\}$  also will be written as  $xy$  (or  $yx$ ).

## Definition

If  $x \in X$  and  $y \in Y$ , then  $xy$  is an  **$X$ - $Y$  edge**.

## Notation

The set of all  $X$ - $Y$  edges in a set  $E$  is denoted by  $E(X, Y)$ . In addition.

$$E(x, Y) := E(\{x\}, Y),$$

$$E(X, y) := E(X, \{y\}).$$

# Graphs

---

## Notation

An edge  $\{x, y\}$  also will be written as  $xy$  (or  $yx$ ).

## Definition

If  $x \in X$  and  $y \in Y$ , then  $xy$  is an  **$X$ - $Y$  edge**.

## Notation

The set of all  $X$ - $Y$  edges in a set  $E$  is denoted by  $E(X, Y)$ . In addition.

$$E(x, Y) := E(\{x\}, Y),$$

$$E(X, y) := E(X, \{y\}).$$

## Notation

The set of all the edges in a set  $E$  at a vertex  $v$  is denoted  $E(v)$ .

# Graphs

---

## Definition

Two vertices  $x, y$  of a graph  $G$  are **adjacent** or **neighbours**, iff  $\{x, y\}$  is an edge of  $G$ .

# Graphs

---

## Definition

Two vertices  $x, y$  of a graph  $G$  are **adjacent** or **neighbours**, iff  $\{x, y\}$  is an edge of  $G$ .

## Definition

Two edges  $e \neq f$  are **adjacent** iff they have an end in common.

# Complete Graphs

---

## Definition

A graph is **complete** iff all its vertices are pairwise adjacent.

# Complete Graphs

---

## Definition

A graph is **complete** iff all its vertices are pairwise adjacent.

## Notation

A complete graph on  $n$  vertices is denoted  $K^n$ .

# Complete Graphs

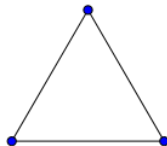
Example (some  $K^n$  graphs\*)



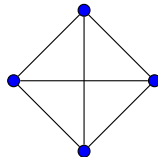
$K^1$



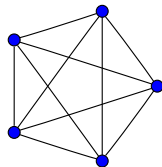
$K^2$



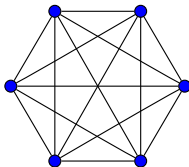
$K^3$



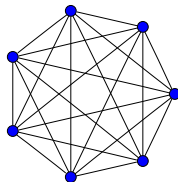
$K^4$



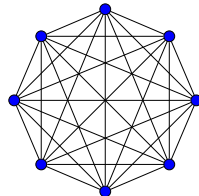
$K^5$



$K^6$



$K^7$



$K^8$



# Isomorphisms

---

## Definition

Whiteboard.

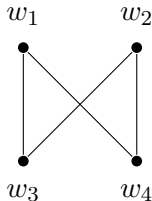
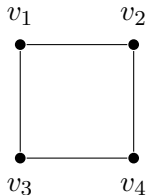
## Notation

If  $G$  and  $G'$  are isomorphic is denoted by  $G \simeq G'$ .

# Isomorphisms

## Example

The following graphs are isomorphic.



The functions  $\psi$  and its inverse preserve adjacency.

$$\psi : \{v_1, v_2, v_3, v_4\} \rightarrow \{w_1, w_2, w_3, w_4\}$$

$$\psi(v_1) = w_1, \psi(v_2) = w_4, \psi(v_3) = w_3 \text{ and } \psi(v_4) = w_2.$$

# Isomorphisms

---

## Remark

Since graphs are not algebraic structures but relational ones, a bijective homomorphism between graphs need not be an isomorphism (see, e.g. Cohn (1981, p. 190)).

# Isomorphisms

---

## Remark

Since graphs are not algebraic structures but relational ones, a bijective homomorphism between graphs need not be an isomorphism (see, e.g. Cohn (1981, p. 190)).

## Definition

A homomorphism  $\psi$  between two relational structures is (see, e.g. Cohn (1981)):

- a **monomorphism** iff  $\psi$  is an injection,
- an **epimorphism** iff  $\psi$  is a surjection,
- an **endomorphism** iff  $\psi$  is from a relational structure to itself,
- an **isomorphism** iff  $\psi$  has an inverse which is also a homomorphism,
- an **automorphism** iff  $\psi$  is an isomorphism and an endomorphism.

# Isomorphisms

---

## Definition

A graph **property** is a **class** of graphs that is closed under isomorphism.

# Isomorphisms

---

## Definition

A graph **property** is a **class** of graphs that is closed under isomorphism.

## Example

- To have a number even of vertices.
- To contain three pairwise adjacent vertices.

# Isomorphisms

---

## Definition

A graph **invariant** (or **parameter**) is a map taking graphs as arguments and assigning equal values to isomorphic graphs.

# Isomorphisms

---

## Definition

A graph **invariant** (or **parameter**) is a map taking graphs as arguments and assigning equal values to isomorphic graphs.

## Example

- The number of vertices (or edges).
- The greatest number of pairwise adjacent vertices.



# Isomorphisms

---

## Definition

A graph **invariant** (or **parameter**) is a map taking graphs as arguments and assigning equal values to isomorphic graphs.

## Example

- The number of vertices (or edges).
- The greatest number of pairwise adjacent vertices.

## Notation

The expression  $x := y$  means that  $x$  is being defined as  $y$ .

# Isomorphisms

---

## Definition

If  $A$  is a proposition then we define

$$\mathbf{1}(A) := \begin{cases} 1, & \text{if } A \text{ is true;} \\ 0, & \text{otherwise.} \end{cases}$$

# Isomorphisms

---

## Definition

If  $A$  is a proposition then we define

$$\mathbf{1}(A) := \begin{cases} 1, & \text{if } A \text{ is true;} \\ 0, & \text{otherwise.} \end{cases}$$

## Definition

The **indicator** function of a set  $S$ , denoted  $\mathbf{1}_S$ , is defined by (Lovász 2012):

$$\begin{aligned} \mathbf{1}_S &: S \rightarrow \{0, 1\} \\ \mathbf{1}_S(x) &:= \mathbf{1}(x \in S). \end{aligned}$$

# Isomorphisms

---

## Definition

If  $A$  is a proposition then we define

$$\mathbf{1}(A) := \begin{cases} 1, & \text{if } A \text{ is true;} \\ 0, & \text{otherwise.} \end{cases}$$

## Definition

The **indicator** function of a set  $S$ , denoted  $\mathbf{1}_S$ , is defined by (Lovász 2012):

$$\begin{aligned} \mathbf{1}_S &: S \rightarrow \{0, 1\} \\ \mathbf{1}_S(x) &:= \mathbf{1}(x \in S). \end{aligned}$$

## Relation between properties and invariants

If we identify a graph property  $P$  with its indicator function  $\mathbf{1}_P$ , graph properties are just 0-1 valued graph invariants (Lovász 2012).

# Subgraphs

---

## Definition

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs.

If  $V' \subseteq V$  and  $E' \subseteq E$  then  $G'$  is a **subgraph** of  $G$ , denoted by  $G' \subseteq G$ , and  $G$  is a **supergraph** of  $G'$ .

If  $G' \subseteq G$  and  $G' \neq G$ , then  $G'$  is a **proper** subgraph of  $G$ .

# Subgraphs

---

## Definition

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs.

If  $V' \subseteq V$  and  $E' \subseteq E$  then  $G'$  is a **subgraph** of  $G$ , denoted by  $G' \subseteq G$ , and  $G$  is a **supergraph** of  $G'$ .

If  $G' \subseteq G$  and  $G' \neq G$ , then  $G'$  is a **proper** subgraph of  $G$ .

## Examples

Whiteboard.

# Subgraphs

---

## Definition

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs.

If  $V' \subseteq V$  and  $E' \subseteq E$  then  $G'$  is a **subgraph** of  $G$ , denoted by  $G' \subseteq G$ , and  $G$  is a **supergraph** of  $G'$ .

If  $G' \subseteq G$  and  $G' \neq G$ , then  $G'$  is a **proper** subgraph of  $G$ .

## Examples

Whiteboard.

## Theorem

The subgraph relation forms a **partial order** on all graphs.

## Proof

Whiteboard.

# Subgraphs

---

## Definition

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs. If  $G' \subseteq G$  and  $G'$  contains **all** the edges whose ends are both in  $V'$ , then  $G'$  is an **induced subgraph** of  $G$  and  $V'$  **induces** or **spans**  $G'$  in  $G$ .

## Remark

Note that an induced subgraph is a subgraph obtained **only** by deleting **vertices**.



# Subgraphs

---

## Definition

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs. If  $G' \subseteq G$  and  $G'$  contains **all** the edges whose ends are both in  $V'$ , then  $G'$  is an **induced subgraph** of  $G$  and  $V'$  **induces** or **spans**  $G'$  in  $G$ .

## Remark

Note that an induced subgraph is a subgraph obtained **only** by deleting **vertices**.

## Examples

Whiteboard.

# Subgraphs

---

## Definition

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs. If  $G' \subseteq G$  and  $G'$  contains **all** the edges whose ends are both in  $V'$ , then  $G'$  is an **induced subgraph** of  $G$  and  $V'$  **induces** or **spans**  $G'$  in  $G$ .

## Remark

Note that an induced subgraph is a subgraph obtained **only** by deleting **vertices**.

## Examples

Whiteboard.

## Notation

Let  $G = (V, E)$  be a graph and  $U \subseteq V$  a set of vertices. We write  $G[U]$  for the graph on  $U$  whose edges are the edges of  $G$  with both ends in  $U$ .

# Subgraphs

---

## Definition

If  $G' \subseteq G$  and  $G'$  contains all the vertices of  $G$ , then  $G'$  is a **spanning subgraph** (*subgrafo de expansión*) of  $G$ .

## Remark

Note that a spanning subgraph is a subgraph obtained **only** by deleting **edges**.

## Remark

Note that **every** graph is a spanning subgraph of a **complete** graph.

# New Graphs Using Operations between Sets

---

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs.

## Definition

We define the following graphs:

$$G \cup G' := (V \cup V', E \cup E'),$$

$$G \cap G' := (V \cap V', E \cap E'),$$

$$\overline{G} := (V, [V]^2 \setminus E).$$

# New Graphs Using Operations between Sets

---

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs.

## Definition

We define the following graphs:

$$G \cup G' := (V \cup V', E \cup E'),$$

$$G \cap G' := (V \cap V', E \cap E'),$$

$$\overline{G} := (V, [V]^2 \setminus E).$$

## Definition

If  $G \cap G' = \emptyset$ , then  $G$  and  $G'$  are **disjoint** graphs.

# New Graphs Using Operations between Sets

---

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs,  $U \subseteq V$  a set of vertices and  $F \subseteq [V]^2$  a set of edges.

## Definition

We define the following graphs:

$$G - U := G[V \setminus U],$$

$$G - G' := G - V(G'),$$

$$G - v := G - \{v\},$$

$$G - F := (V, E \setminus F),$$

$$G + F := (V, E \cup F),$$

$$G - e := G - \{e\},$$

$$G + e := G + \{e\}.$$

# New Graphs Using Operations between Sets

---

Let  $G = (V, E)$  and  $G' = (V', E')$  be two graphs,  $U \subseteq V$  a set of vertices and  $F \subseteq [V]^2$  a set of edges.

## Definition

We define the following graphs:

$$G - U := G[V \setminus U],$$

$$G - G' := G - V(G'),$$

$$G - v := G - \{v\},$$

$$G - F := (V, E \setminus F),$$

$$G + F := (V, E \cup F),$$

$$G - e := G - \{e\},$$

$$G + e := G + \{e\}.$$

## Examples

Whiteboard.

# New Graphs Using Other Operations

---

## Definition

Let  $G$  and  $G'$  be two **disjoint** graphs. By  $G * G'$  we denoted the graph obtained **from**  $G \cup G'$  by joining **all** the vertices of  $G$  to **all** the vertices of  $G'$ .



# New Graphs Using Other Operations

---

## Definition

Let  $G$  and  $G'$  be two **disjoint** graphs. By  $G * G'$  we denoted the graph obtained **from**  $G \cup G'$  by joining **all** the vertices of  $G$  to **all** the vertices of  $G'$ .

## Example

$$K^2 * K^3 = K^5.$$

# The Degree of a Vertex

---

## Definition

The **degree** (or **valence**) of a vertex  $v$ , denoted  $d(v)$ , is the number of edges at  $v$ .

# The Degree of a Vertex

---

## Definition

We define the following invariants:

$$\delta(G) := \min \{d(v) \mid v \in V\}$$

**(minimum degree of  $G$ )**

$$\Delta(G) := \max \{d(v) \mid v \in V\}$$

**(maximum degree of  $G$ )**

$$d(G) := \frac{1}{|V|} \sum_{v \in V} d(v)$$

**(average degree of  $G$ )**

$$\epsilon(G) := \frac{|E|}{|V|}$$

**(number of edges by vertex in  $G$ )**

# The Degree of a Vertex

---

## Proposition (1.2.2)

Every graph  $G$  with at least one edge has a subgraph  $H$  with

$$\delta(H) > \epsilon(H) \geq \epsilon(G).$$

## Proof

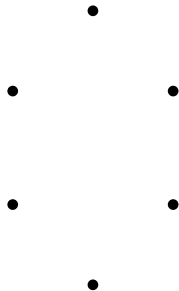
Whiteboard.

# Regular Graphs

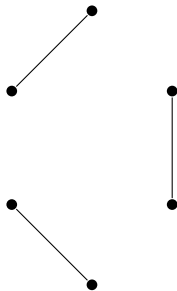
## Definition

For  $k \in \mathbb{N}$ , a graph is  **$k$ -regular** iff all its vertices have **degree**  $k$ .

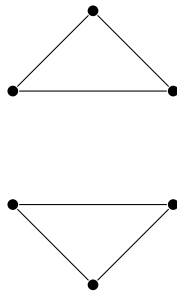
Example (some regular graphs\*)



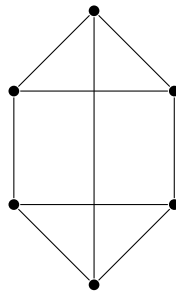
0-regular



1-regular



2-regular



3-regular

\*Example from [https://en.wikipedia.org/wiki/Regular\\_graph](https://en.wikipedia.org/wiki/Regular_graph).

# Regular Graphs

---

## Convention

A 3-regular graph is called a **cubic** graph.

## Example

The **Petersen graph** and the **Heawood graph** are cubic graphs.

# Paths and Cycles

---

## Definition

A **path** is a non-empty graph  $P = (V, E)$  where  $V$  and  $E$  are of the form

$$V = \{x_0, x_1, \dots, x_k\},$$

$$E = \{x_0x_1, x_1x_2, \dots, x_{k-1}x_k\},$$

and the  $x_i$  are **all distinct**.

# Paths and Cycles

---

## Definition

A **path** is a non-empty graph  $P = (V, E)$  where  $V$  and  $E$  are of the form

$$V = \{x_0, x_1, \dots, x_k\},$$

$$E = \{x_0x_1, x_1x_2, \dots, x_{k-1}x_k\},$$

and the  $x_i$  are **all distinct**. The vertices  $x_0$  and  $x_k$  are the **ends** of  $P$ .



# Paths and Cycles

---

## Definition

The **length** of a path is its number of edges.

# Paths and Cycles

---

## Definition

The **length** of a path is its number of edges.

## Notation

For  $k \in \mathbb{N}$ , a path of length  $k$  is denoted by  $P^k$ .

# Paths and Cycles

---

## Definition

The **length** of a path is its number of edges.

## Notation

For  $k \in \mathbb{N}$ , a path of length  $k$  is denoted by  $P^k$ .

## Example

$$P^0 = K^1.$$

## Examples

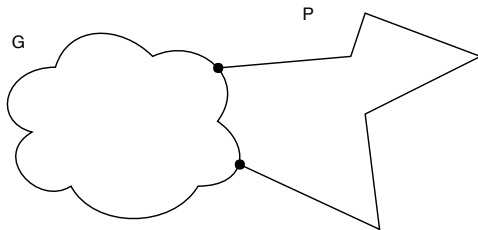
Whiteboard.

# Paths and Cycles

---

## Definition

Let  $G$  be a graph. A path  $P$  is a  **$G$ -path** iff  $P$  is non-trivial and meets  $G$  exactly in its ends.



# Paths and Cycles

---

## Notation

Sometimes we denoted a path by the sequence of its vertices, i.e.,

$$P = x_0x_1 \dots x_k.$$

# Paths and Cycles

---

## Notation

Sometimes we denoted a path by the sequence of its vertices, i.e.,

$$P = x_0x_1 \dots x_k.$$

## Notation

Let  $P = x_0x_1 \dots x_k$  be a path. For  $0 \leq i \leq j \leq k$  we write

$$x_iPx_j := x_i \dots x_j.$$

# Paths and Cycles

---

## Definition

Let  $A$  and  $B$  two sets of vertices. A path  $P = x_0 \dots x_k$  is an  **$A$ - $B$  path** if

$$V(P) \cap A = \{x_0\} \quad \text{and} \quad V(P) \cap B = \{x_k\}.$$

# Paths and Cycles

---

## Definition

Let  $A$  and  $B$  two sets of vertices. A path  $P = x_0 \dots x_k$  is an  **$A$ - $B$  path** if

$$V(P) \cap A = \{x_0\} \quad \text{and} \quad V(P) \cap B = \{x_k\}.$$

## Notation

$a$ - $B$  path  $:= \{a\}$ - $B$  path,

$A$ - $b$  path  $:= A$ - $\{b\}$  path,

$a$ - $b$  path  $:= \{a\}$ - $\{b\}$  path.



# Paths and Cycles

---

## Definition

If  $P = x_0 \dots x_{k-1}$  is a path and  $k \geq 3$ , then the following **graph** is a **cycle**:

$$C := P + x_{k-1}x_0.$$

# Paths and Cycles

---

## Definition

If  $P = x_0 \dots x_{k-1}$  is a path and  $k \geq 3$ , then the following **graph** is a **cycle**:

$$C := P + x_{k-1}x_0.$$

## Definition

The **length** of a cycle is its number of edges (or vertices).

# Paths and Cycles

---

## Definition

If  $P = x_0 \dots x_{k-1}$  is a path and  $k \geq 3$ , then the following **graph** is a **cycle**:

$$C := P + x_{k-1}x_0.$$

## Definition

The **length** of a cycle is its number of edges (or vertices). Let  $k \geq 3$  be an integer. A cycle of length  $k$  is a  **$k$ -cycle** and it is denoted by  $C^k$ .

## Examples

Whiteboard.

# Paths and Cycles

---

## Definition

An **acyclic** graph is a graph that contains **no** cycles.

# Paths and Cycles

---

## Definition

An **acyclic** graph is a graph that contains **no** cycles.

## Definition

An **even/odd cycle** is a cycle of even/odd length.

# Paths and Cycles

---

## Definition

The **girth** (*cintura*) of a graph  $G$ , denoted  $g(G)$ , is the length of **a** shortest cycle contained in  $G$ . If  $G$  does not contain a cycle, then  $g(G) := \infty$ .

## Remark

Note that we wrote ‘a shortest cycle’ instead of ‘**the** shortest cycle’.

# Paths and Cycles

---

## Definition

The **girth** (*cintura*) of a graph  $G$ , denoted  $g(G)$ , is the length of **a** shortest cycle contained in  $G$ . If  $G$  does not contain a cycle, then  $g(G) := \infty$ .

## Remark

Note that we wrote ‘a shortest cycle’ instead of ‘**the** shortest cycle’.

## Example

The girths of the **Petersen graph** and the **Heawood graph** are 5 and 6, respectively.

# Paths and Cycles

---

## Definition

Let  $C$  be a cycle. A **chord** of  $C$  is an edge which connects two vertices of  $C$  but it is not part of  $C$ .



# Paths and Cycles

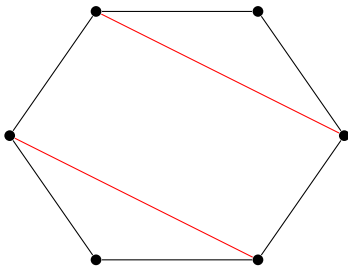
---

## Definition

Let  $C$  be a cycle. A **chord** of  $C$  is an edge which connects two vertices of  $C$  but it is not part of  $C$ .

## Example

A cycle (black) with two chords (red edges).



# Paths and Cycles

---

## Definition

Let  $G$  be a graph. An **induced cycle** in  $G$  is a cycle in  $G$  that has no chords.

## Remark

Note that an induced cycle in  $G$  is an **induced subgraph** in  $G$ .

## Example

See Diestel (2017, Fig. 1.3.3).

# Paths and Cycles

---

## Proposition (1.3.1)

Let  $G$  be a graph. If  $\delta(G) \geq 2$  then  $G$  contains a path of length  $\delta(G)$  and a cycle of length at least  $\delta(G) + 1$ .

# Paths and Cycles

---

## Definition

The **distance** between two vertices  $x$ ,  $y$ , denoted  $d(x, y)$ , is the length of **a** shortest  $x$ - $y$  path in a graph. If there is no such path, then  $d(x, y) := \infty$ .

## Remark

Note that we wrote '**a** shortest  $x$ - $y$  path' instead of '**the** shortest  $x$ - $y$  path'.

# Paths and Cycles

---

## Definition

Let  $G$  be a graph. The **diameter** of  $G$ , denoted  $\text{diam}(G)$ , is the greatest distance between any two vertices in  $G$ , that is,

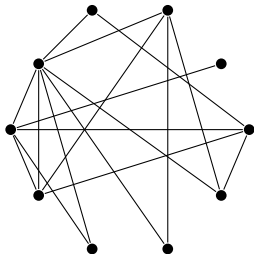
$$\text{diam}(G) := \max_{x,y \in V(G)} d(x,y).$$

# Paths and Cycles

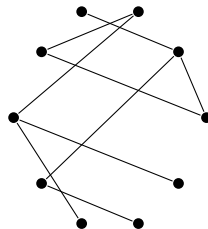
---

## Example

Graphs on 10 vertices with diameters 3 and 7.\*



Diameter 3



Diameter 7

---

\*Example from <http://mathworld.wolfram.com/GraphDiameter.html> .

# Paths and Cycles

---

## Proposition (1.3.2)

Every graph  $G$  containing a cycle satisfies

$$g(G) \leq 2 \operatorname{diam}(G) + 1.$$

Proof by contradiction

Whiteboard.

# Paths and Cycles

---

## Definition

Let  $G$  be a graph. The **eccentricity** of a vertex  $x$ , denoted  $\text{ecc}(x)$ , is the greatest distance between  $x$  and any other vertex of  $G$  (see, e.g. Harary (1969)), that is,

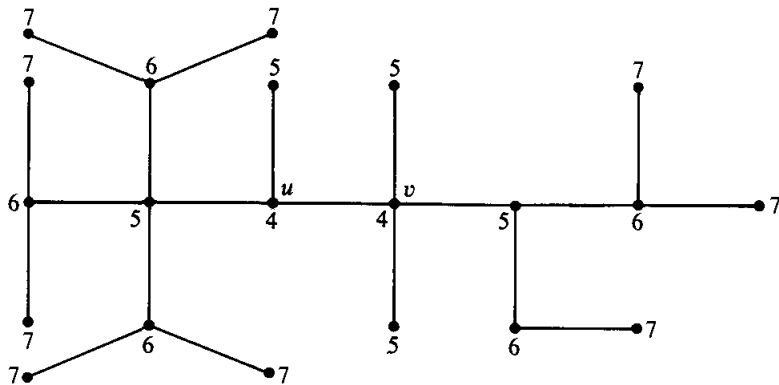
$$\text{ecc}(x) := \max \{d(x, y) \mid y \in V(G)\}.$$



# Paths and Cycles

## Example

A graph where the eccentricity of each vertex is shown.\*



\*Figure source: Harary (1969, Fig. 4.2).

# Paths and Cycles

---

## Definition

Let  $G$  be a graph. The **radius** of  $G$ , denoted  $\text{rad}(G)$ , is the minimum eccentricity of its vertices, that is,

$$\begin{aligned}\text{rad}(G) &:= \min_{x \in V(G)} \max_{y \in V(G)} d(x, y) \\ &= \min \{ \text{ecc}(x) \mid x \in V(G) \}.\end{aligned}$$

# Paths and Cycles

---

## Definition

Let  $G$  be a graph. The **radius** of  $G$ , denoted  $\text{rad}(G)$ , is the minimum eccentricity of its vertices, that is,

$$\begin{aligned}\text{rad}(G) &:= \min_{x \in V(G)} \max_{y \in V(G)} d(x, y) \\ &= \min \{ \text{ecc}(x) \mid x \in V(G) \}.\end{aligned}$$

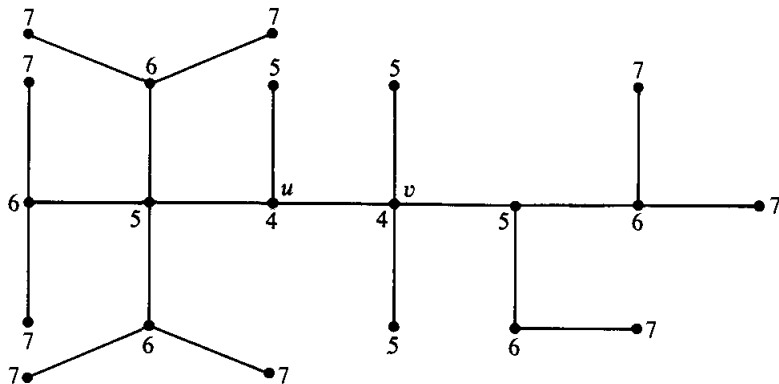
## Definition

Let  $G$  be a graph. A vertex  $v$  is **central** in  $G$  iff  $\text{ecc}(v) = \text{rad}(G)$ .

# Paths and Cycles

## Example

A graph where the eccentricity of each vertex is shown, with  $\text{rad}(G) = 4$  and central vertices  $u$  and  $v$ .\*



\*Figure source: Harary (1969, Fig. 4.2).

# Paths and Cycles

---

## Definition

Let  $G$  be a graph. A **walk** of length  $k$  in  $G$  is a non-empty alternating sequence  $v_0 e_0 v_1 e_1 \dots e_{k-1} v_k$  of vertices and edges in  $G$  such that  $e_i = \{v_i, v_{i+1}\}$ , for all  $i < k$ .

# Connected Graphs

---

## Definition

A non-empty graph  $G$  is **connected** (*conexo*) iff **any** two of its vertices are linked by a **path** in  $G$ . If a graph is not connected it is a **disconnected** graph.

# Connected Graphs

---

## Definition

A non-empty graph  $G$  is **connected** (*conexo*) iff **any** two of its vertices are linked by a **path** in  $G$ . If a graph is not connected it is a **disconnected** graph.

## Example

**Complete** graphs are connected graphs. The **trivial non-empty** graph is a connected graph.

# Connected Graphs

---

## Definition

A non-empty graph  $G$  is **connected** (*conexo*) iff **any** two of its vertices are linked by a **path** in  $G$ . If a graph is not connected it is a **disconnected** graph.

## Example

**Complete** graphs are connected graphs. The **trivial non-empty** graph is a connected graph.

## Example

The **non-trivial 0-regulars** graphs are disconnected graphs.



# Connected Graphs

---

## Definition

A non-empty graph  $G$  is **connected** (*conexo*) iff **any** two of its vertices are linked by a **path** in  $G$ . If a graph is not connected it is a **disconnected** graph.

## Example

**Complete** graphs are connected graphs. The **trivial non-empty** graph is a connected graph.

## Example

The **non-trivial 0-regulars** graphs are disconnected graphs.

## Example

Let  $n$  be a positive integer. An  **$n$ -regular** graph can be a connected or a disconnected graph.

# Connected Graphs

---

## Example

The **Petersen graph** and the **Heawood graph** are connected graphs.

# Connected Graphs

---

## Proposition (1.4.1)

The vertices of a connected graph  $G$  can always be enumerated, say as  $v_1, \dots, v_n$ , so that  $G_i := G[v_1, \dots, v_i]$  is connected for every  $i$ .

# Connected Graphs

---

## Proposition (1.4.1)

The vertices of a connected graph  $G$  can always be enumerated, say as  $v_1, \dots, v_n$ , so that  $G_i := G[v_1, \dots, v_i]$  is connected for every  $i$ .

Proof by induction on  $|G|$ .

1. Choose any vertex as  $v_1$ , so  $G_1 := G[v_1]$  is connected.
2. Inductive hypothesis: Assume that  $v_1, \dots, v_i$  have been chosen for some  $i < |G|$ , and  $G_i := G[v_1, \dots, v_i]$  are connected.
3. Now, choose a vertex  $v \in G - G_i$ .
4. Since  $G$  is connected it contains a  $v$ - $v_1$  path  $P$ .
5. Choose as  $v_{i+1}$  the last vertex of  $P$  in  $G - G_i$ .
6. Then  $v_{i+1}$  has a neighbour in  $G_i$ .
7. The connectedness of  $G_{i+1}$  follows by the inductive hypothesis and the previous steps. ■

# Components

---

## Remark

Recall that the subgraph relation forms a partial order on all graphs (see [this](#) theorem).

# Components

---

## Remark

Recall that the subgraph relation forms a partial order on all graphs (see [this](#) theorem).

## Definition

Let  $G$  be a graph. A **maximal** connected subgraph of  $G$  is a **component** of  $G$ .

# Components

---

## Remark

Recall that the subgraph relation forms a partial order on all graphs (see [this](#) theorem).

## Definition

Let  $G$  be a graph. A [maximal](#) connected subgraph of  $G$  is a **component** of  $G$ .

## Example

See Diestel ([2017](#), Fig. 1.4.1).

# Components

---

## Remark

Recall that the subgraph relation forms a partial order on all graphs (see [this](#) theorem).

## Definition

Let  $G$  be a graph. A [maximal](#) connected subgraph of  $G$  is a **component** of  $G$ .

## Example

See Diestel ([2017](#), Fig. 1.4.1).

## Remark

The components of a graph are [induced subgraphs](#).



# Components

---

## Remark

Let  $G = (V, E)$  be a graph. The vertex sets of the components of  $G$  partition the set  $V$ .

# Components

---

## Remark

Let  $G = (V, E)$  be a graph. The vertex sets of the components of  $G$  **partition** the set  $V$ .

## What about the empty-graph?

If a graph is connected then it is a not-empty graph, so the empty-graph has no components.

# $k$ -Connected Graphs

---

## Definition

For  $k \in \mathbb{N}$ , a graph  $G = (V, E)$  is  **$k$ -connected** (or  **$k$ -vertex-connected**) iff

- i)  $|G| > k$  and
- ii) for every  $X \subseteq V(G)$ , if  $|X| < k$  then  $G - X$  is a connected graph.

# $k$ -Connected Graphs

---

## Definition

For  $k \in \mathbb{N}$ , a graph  $G = (V, E)$  is  **$k$ -connected** (or  **$k$ -vertex-connected**) iff

- i)  $|G| > k$  and
- ii) for every  $X \subseteq V(G)$ , if  $|X| < k$  then  $G - X$  is a connected graph.

## Example

Every **non-empty** disconnected graph is 0-connected (by false antecedent).x

## Example

Every **non-empty** graph is 0-connected (by false antecedent).

# $k$ -Connected Graphs

---

## Definition

For  $k \in \mathbb{N}$ , a graph  $G = (V, E)$  is  **$k$ -connected** (or  **$k$ -vertex-connected**) iff

- i)  $|G| > k$  and
- ii) for every  $X \subseteq V(G)$ , if  $|X| < k$  then  $G - X$  is a connected graph.

## Example

Every **non-empty** disconnected graph is 0-connected (by false antecedent).x

## Example

Every **non-empty** graph is 0-connected (by false antecedent).

## Example

The 1-connected graphs are the **non-trivial** connected graphs.

# $k$ -Connected Graphs

---

## Example

A  $K^n$  graph, with  $n \geq 1$ , is an  $(n - 1)$ -connected graph.

# $k$ -Connected Graphs

---

## Example

A  $K^n$  graph, with  $n \geq 1$ , is an  $(n - 1)$ -connected graph.

## Exercise

To build a 1-connected but not 2-connected graph.

# $k$ -Connected Graphs

---

## Example

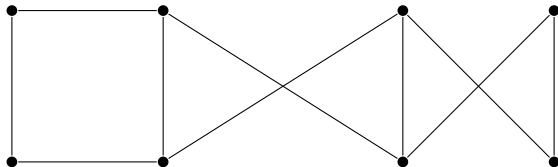
A  $K^n$  graph, with  $n \geq 1$ , is an  $(n - 1)$ -connected graph.

## Exercise

To build a 1-connected but not 2-connected graph.

## Example

A 2-connected but not 3-connected graph.





# Connectivity

---

## Remark

Note that if a graph is  $k$ -connected, with  $k \geq 1$ , then it also is  $(k - 1)$ -connected.

# Connectivity

---

## Remark

Note that if a graph is  $k$ -connected, with  $k \geq 1$ , then it also is  $(k - 1)$ -connected.

## Definition

The greatest integer  $k$  such that a graph  $G$  is  $k$ -connected is the **connectivity** of  $G$ , denoted  $\kappa(G)$ .

# Connectivity

---

## Remark

Note that if a graph is  $k$ -connected, with  $k \geq 1$ , then it also is  $(k - 1)$ -connected.

## Definition

The greatest integer  $k$  such that a graph  $G$  is  $k$ -connected is the **connectivity** of  $G$ , denoted  $\kappa(G)$ .

## Example

$\kappa(G) = 0$  if  $G$  is disconnected.

# Connectivity

---

## Remark

Note that if a graph is  $k$ -connected, with  $k \geq 1$ , then it also is  $(k - 1)$ -connected.

## Definition

The greatest integer  $k$  such that a graph  $G$  is  $k$ -connected is the **connectivity** of  $G$ , denoted  $\kappa(G)$ .

## Example

$\kappa(G) = 0$  if  $G$  is disconnected.

## Question

Why  $\kappa(K^1)$  is not 1, but 0?

# Connectivity

---

## Remark

Note that if a graph is  $k$ -connected, with  $k \geq 1$ , then it also is  $(k - 1)$ -connected.

## Definition

The greatest integer  $k$  such that a graph  $G$  is  $k$ -connected is the **connectivity** of  $G$ , denoted  $\kappa(G)$ .

## Example

$\kappa(G) = 0$  if  $G$  is disconnected.

## Question

Why  $\kappa(K^1)$  is not 1, but 0? Because  $|K^1| \not\geq 1$ .

# Connectivity

---

## Remark

Note that if a graph is  $k$ -connected, with  $k \geq 1$ , then it also is  $(k - 1)$ -connected.

## Definition

The greatest integer  $k$  such that a graph  $G$  is  $k$ -connected is the **connectivity** of  $G$ , denoted  $\kappa(G)$ .

## Example

$\kappa(G) = 0$  if  $G$  is disconnected.

## Question

Why  $\kappa(K^1)$  is not 1, but 0? Because  $|K^1| \not\geq 1$ .

## Example

$\kappa(K^n) = n - 1$ , for all  $n \geq 1$ .

# $l$ -Edge-Connected Graphs

---

## Definition

For  $l \in \mathbb{N}$ , a graph  $G = (V, E)$  is  **$l$ -edge-connected** iff

- i)  $|G| > 1$ , i.e.  $G$  is **non-trivial**, and
- ii) for every set  $F \subseteq E(G)$ , if  $|F| < l$  then  $G - F$  is a connected graph.

# $l$ -Edge-Connected Graphs

---

## Definition

For  $l \in \mathbb{N}$ , a graph  $G = (V, E)$  is  **$l$ -edge-connected** iff

- i)  $|G| > 1$ , i.e.  $G$  is **non-trivial**, and
- ii) for every set  $F \subseteq E(G)$ , if  $|F| < l$  then  $G - F$  is a connected graph.

## Example

Every **non-trivial** graph is 0-edge-connected (by false antecedent).



# $l$ -Edge-Connected Graphs

---

## Definition

For  $l \in \mathbb{N}$ , a graph  $G = (V, E)$  is  **$l$ -edge-connected** iff

- i)  $|G| > 1$ , i.e.  $G$  is **non-trivial**, and
- ii) for every set  $F \subseteq E(G)$ , if  $|F| < l$  then  $G - F$  is a connected graph.

## Example

Every **non-trivial** graph is 0-edge-connected (by false antecedent).

## Example

The 1-edge-connected graphs are the **non-trivial** connected graphs.

# $l$ -Edge-Connected Graphs

---

## Definition

For  $l \in \mathbb{N}$ , a graph  $G = (V, E)$  is  **$l$ -edge-connected** iff

- i)  $|G| > 1$ , i.e.  $G$  is **non-trivial**, and
- ii) for every set  $F \subseteq E(G)$ , if  $|F| < l$  then  $G - F$  is a connected graph.

## Example

Every **non-trivial** graph is 0-edge-connected (by false antecedent).

## Example

The 1-edge-connected graphs are the **non-trivial** connected graphs.

## Example

A  $K^n$  graph, with  $n \geq 2$ , is an  $(n - 1)$ -edge-connected graph.

# Edge-Connectivity

---

## Remark

Note that if a graph is  $l$ -edge-connected, with  $l \geq 1$ , then it also is  $(l - 1)$ -edge-connected.

# Edge-Connectivity

---

## Remark

Note that if a graph is  $l$ -edge-connected, with  $l \geq 1$ , then it also is  $(l - 1)$ -edge-connected.

## Definition

The greatest integer  $l$  such that a graph  $G$  is  $l$ -edge-connected is the **edge-connectivity** of  $G$ , denoted  $\lambda(G)$ .

# Edge-Connectivity

---

## Remark

Note that if a graph is  $l$ -edge-connected, with  $l \geq 1$ , then it also is  $(l - 1)$ -edge-connected.

## Definition

The greatest integer  $l$  such that a graph  $G$  is  $l$ -edge-connected is the **edge-connectivity** of  $G$ , denoted  $\lambda(G)$ .

## Example

$\lambda(G) = 0$  if  $G$  is disconnected.

# Edge-Connectivity

---

## Remark

Note that if a graph is  $l$ -edge-connected, with  $l \geq 1$ , then it also is  $(l - 1)$ -edge-connected.

## Definition

The greatest integer  $l$  such that a graph  $G$  is  $l$ -edge-connected is the **edge-connectivity** of  $G$ , denoted  $\lambda(G)$ .

## Example

$\lambda(G) = 0$  if  $G$  is disconnected.

## Example

$\lambda(K^n) = n - 1$ , for all  $n \geq 2$ .

# Connectivity and Edge-Connectivity

---

## Question

Can the connectivity and the edge-connectivity be equals? Yes!

---

\*Figure source: Diestel (2017, Fig. 1.4.3).

# Connectivity and Edge-Connectivity

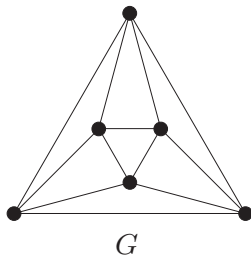
---

## Question

Can the connectivity and the edge-connectivity be equals? Yes!

## Example

A graph  $G$  with  $\kappa(G) = \lambda(G) = 4$ .\*



---

\*Figure source: Diestel (2017, Fig. 1.4.3).



# Connectivity and Edge-Connectivity

---

## Question

Can the connectivity be smaller than the edge-connectivity? Yes!

---

\*Figure source: Diestel (2017, Fig. 1.4.3).

# Connectivity and Edge-Connectivity

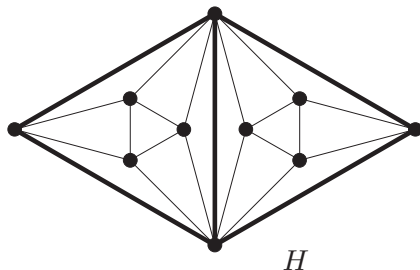
---

## Question

Can the connectivity be smaller than the edge-connectivity? Yes!

## Example

A graph  $H$  with  $\kappa(H) = 2$  and  $\lambda(H) = 4$ .\*



---

\*Figure source: Diestel (2017, Fig. 1.4.3).

# Edge-Connectivity and Minimum Degree

---

## Question

Can the edge-connectivity and the minimum degree be equals? Yes!

# Edge-Connectivity and Minimum Degree

---

## Question

Can the edge-connectivity and the minimum degree be equals? Yes!

## Remark

Recall that  $\delta(G)$  denotes the minimum degree of a graph  $G$ .

## Example

Let  $G$  be a  $K^n$  graph, with  $n \geq 2$ , then  $\lambda(G) = \delta(G) = n - 1$ .

# Edge-Connectivity and Minimum Degree

---

## Question

Can the edge-connectivity be smaller than the minimum degree? Yes!

# Edge-Connectivity and Minimum Degree

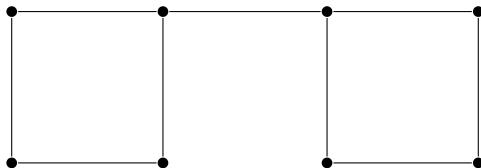
---

## Question

Can the edge-connectivity be smaller than the minimum degree? Yes!

## Example

A graph  $G$  with  $\lambda(G) = 1$  and  $\delta(G) = 2$ .



# Connectivity, Edge-Connectivity and Minimum Degree

---

## Proposition (1.4.2)

If  $G$  is **non-trivial**, then  $\kappa(G) \leq \lambda(G) \leq \delta(G)$ .

# Forests and Trees

---

## Definition

A **forest** (*bosque*) is an **acyclic graph**. A **tree** is a **connected** acyclic graph.



# Forests and Trees

---

## Definition

A **forest** (*bosque*) is an **acyclic graph**. A **tree** is a **connected** acyclic graph.

## Remark

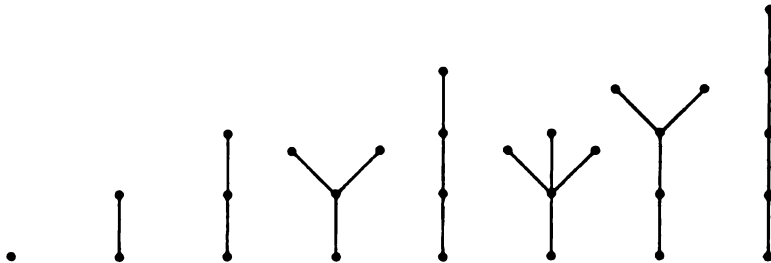
A forest is a graph whose **components** are trees.

# Forests and Trees

---

## Example

A forest formed with all the trees with at most five vertices.\*



---

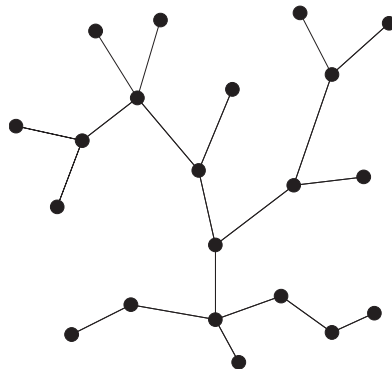
\*Figure source: Biggs, Lloyd and Wilson (1998, Fig. 3.1).

# Forests and Trees

---

## Example

A tree.\*



---

\*Figure source: Diestel (2017, Fig. 1.5.1).

# Forests and Trees

---

## Definition

In a tree, the vertices of degree 1 are its **leaves** (except when the root of tree exists and it has degree 1), the other vertices are its **inner vertices**.

# Forests and Trees

---

## Definition

In a tree, the vertices of degree 1 are its **leaves** (except when the root of tree exists and it has degree 1), the other vertices are its **inner vertices**.

## Remark

Every **non-trivial** tree has a leaf, so if we remove a leaf from a tree, we still have a tree.

# Forests and Trees

---

## Theorem (1.5.1)

The following four assertions are equivalent for a graph  $T$ :

- i)  $T$  is a tree;

# Forests and Trees

---

## Theorem (1.5.1)

The following four assertions are equivalent for a graph  $T$ :

- i)  $T$  is a tree;
- ii) any two vertices of  $T$  are linked by a unique **path** in  $T$ ;

# Forests and Trees

---

## Theorem (1.5.1)

The following four assertions are equivalent for a graph  $T$ :

- i)  $T$  is a tree;
- ii) any two vertices of  $T$  are linked by a unique **path** in  $T$ ;
- iii)  $T$  is **minimally** (respect to the subgraph relation) **connected**, i.e.  $T$  is connected but  $T - e$  is disconnected for every edge  $e \in T$ ;



# Forests and Trees

---

## Theorem (1.5.1)

The following four assertions are equivalent for a graph  $T$ :

- i)  $T$  is a tree;
- ii) any two vertices of  $T$  are linked by a unique **path** in  $T$ ;
- iii)  $T$  is **minimally** (respect to the subgraph relation) **connected**, i.e.  $T$  is connected but  $T - e$  is disconnected for every edge  $e \in T$ ;
- iv)  $T$  is **maximally** (respect to the subgraph relation) **acyclic**, i.e.  $T$  contains no cycle but  $T + xy$  does, for any two non-adjacent vertices  $x, y \in T$ .

# Forests and Trees

---

## Corollary (1.5.2)

The vertices of a tree can always be enumerated, say as  $v_1, \dots, v_n$ , so that every  $v_i$  with  $i \geq 2$  has a unique neighbour in  $\{v_1, \dots, v_{i-1}\}$ .

## Proof

Use Proposition 1.4.1.

# Spanning Trees

---

## Definition

A **spanning tree** (*árbol de expansión, árbol generador o árbol recubridor*) of a graph  $G$  is a **spanning subgraph** of  $G$  which is a tree.

# Spanning Trees

---

## Definition

A **spanning tree** (*árbol de expansión, árbol generador o árbol recubridor*) of a graph  $G$  is a **spanning subgraph** of  $G$  which is a tree.

## Example

See <https://www.cs.usfca.edu/~galles/visualization/DFS.html> .

# Spanning Trees

## Example

Spanning trees for the graphs with red vertices.\*



\*Figure from <http://mathworld.wolfram.com/SpanningTree.html>.

# Spanning Trees

---

## Corollary

Every **connected** graph contains a **spanning** tree.

# Spanning Trees

---

## Corollary

Every **connected** graph contains a **spanning** tree.

## Proof

- Build a minimal connected subgraph and apply **Theorem 1.5.1.iii** or
- Build a maximal acyclic subgraph and apply **Theorem 1.5.1.iv**.

# Rooted Trees

---

## Definition

The **root** of a tree is a vertex considered as special.

## Definition

A **rooted tree** is a tree with a fixed root  $r$ .

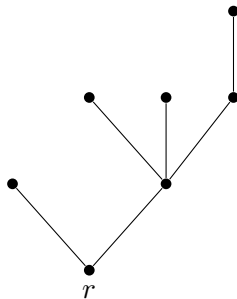


# Rooted Trees

---

## Example

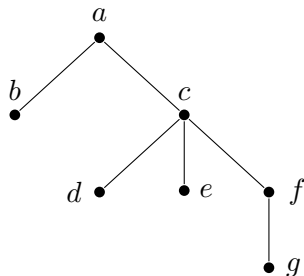
A tree with root  $r$ .



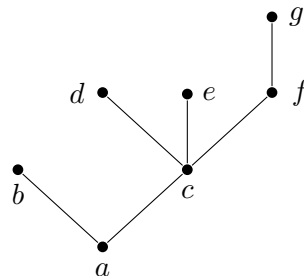
# Rooted Trees

## Example

Four representations of a tree with root  $a$ .\*



Downward



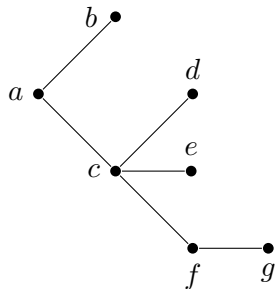
Upward

Continued on next slide

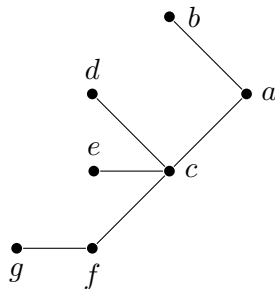
\*Based on the discussion about how to draw trees in (Knuth 1997, § 2.3).

# Rooted Trees

## Example (continuation)



Left to right



Right to left

# Rooted Trees

---

## Definition

Let  $T$  be a tree with root  $r$ . We define the **tree-order** on  $V(T)$  associated with  $T$  and  $r$  by:

$$x \leq y := x \in rTy.$$

# Rooted Trees

---

## Definition

Let  $T$  be a tree with root  $r$ . We define the **tree-order** on  $V(T)$  associated with  $T$  and  $r$  by:

$$x \leq y := x \in rTy.$$

## Theorem

The tree-order associated with a rooted tree  $T$  is a partial order on  $V(T)$ .

## Proof

Whiteboard.

# Rooted Trees

---

## Remarks

In the tree-order of a rooted tree:

- the root is the least element,
- the leaves are its maximal elements and
- the ends of any edge are comparable

# Rooted Trees

---

## Remarks

In the tree-order of a rooted tree:

- the root is the least element,
- the leaves are its maximal elements and
- the ends of any edge are comparable

## Definition

The **down-closure** of a vertex  $v$  is defined by

$$\lceil v \rceil := \{x \mid x \leq v\}.$$

# Rooted Trees

---

## Remarks

In the tree-order of a rooted tree:

- the root is the least element,
- the leaves are its maximal elements and
- the ends of any edge are comparable

## Definition

The **down-closure** of a vertex  $v$  is defined by

$$\lceil v \rceil := \{x \mid x \leq v\}.$$

## Remark

Let  $v$  be a vertex. The down-closure of  $v$  is a **chain**.



# Normal Spanning Trees

---

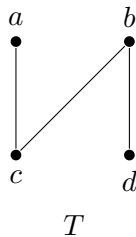
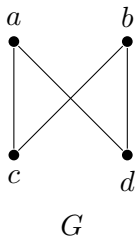
## Definition

A rooted spanning tree  $T$  of a graph  $G$  is a **normal spanning tree** (also called **Trémaux tree**) iff the ends of every edge of  $G$  (i.e. every two adjacent vertices) are comparable in the tree-order on  $V(G)$  induced by  $T$  (Diestel and Leader 2001).

# Normal Spanning Trees

## Example

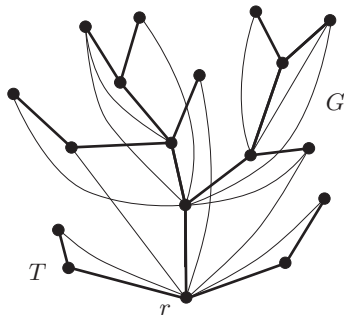
A graph  $G$  and a spanning tree  $T$  of  $G$ . The tree  $T$  is a normal spanning tree only when its root is  $a$  or  $d$ .



# Normal Spanning Trees

## Example

A normal spanning tree  $T$  with root  $r$  of the graph  $G$ .\*



\*Figure source: Diestel (2017, Fig. 1.5.2).

# Normal Spanning Trees

---

## Proposition (1.5.6)

Every connected graph contains a normal spanning tree, with any specified vertex as its root.

# Normal Spanning Trees

---

## Exercise (1.26<sup>+</sup>)

**Depth-first search algorithm:** Let  $G$  be a connected graph, and let  $r \in G$  be a vertex. Starting from  $r$ , move along the edges of  $G$ , going whenever possible to a vertex not visited so far. If there is no such vertex, go back along the edge by which the current vertex was first reached (unless the current vertex is  $r$ ; then stop).

Show that the edges traversed form a normal spanning tree in  $G$  with root  $r$ .

# Normal Spanning Trees

---

## Exercise (1.26<sup>+</sup>)

**Depth-first search algorithm:** Let  $G$  be a connected graph, and let  $r \in G$  be a vertex. Starting from  $r$ , move along the edges of  $G$ , going whenever possible to a vertex not visited so far. If there is no such vertex, go back along the edge by which the current vertex was first reached (unless the current vertex is  $r$ ; then stop).

Show that the edges traversed form a normal spanning tree in  $G$  with root  $r$ .

## Remark

See an animation of the depth-first search algorithm in <https://www.cs.usfca.edu/~galles/visualization/DFS.html>.

# Normal Spanning Trees

---

## Remark

The property of being a normal spanning tree can be expressed in monadic second-order logic (Courcelle and Engelfriet 2012).

# Normal Spanning Trees

---

## Remark

The property of being a normal spanning tree can be expressed in monadic second-order logic (Courcelle and Engelfriet 2012).

## Remark

A formal verification of deep-first search algorithms was done by Lammich and Neumann (2015).



# Bipartite Graphs

---

## Definition

Let  $r \geq 2$  be an integer. A graph is  **$r$ -partite** iff its vertex set can be **partitioned** into  $r$  classes such that every edge has its ends in different classes.

## Remark

Note that in an  $r$ -partite graph the vertices in the same **partition** class are not adjacent.

## Convention

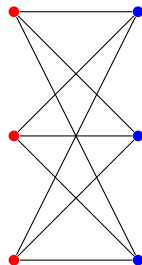
The 2-partite and 3-partite graphs are called **bipartite** and **tripartite**, respectively.

# Bipartite Graphs

---

## Example

A bipartite graph.

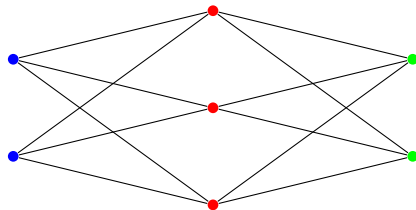


# Bipartite Graphs

---

## Example

A tripartite graph.

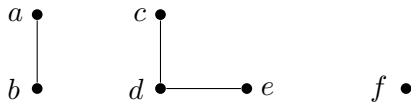


# Bipartite Graphs

---

## Example

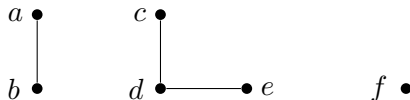
Is the following graph, which has three **components**, a bipartite graph?



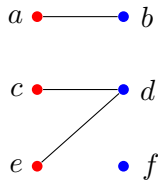
# Bipartite Graphs

## Example

Is the following graph, which has three **components**, a bipartite graph?



Yes!



# Bipartite Graphs

---

## Example

Another bipartite graph.



# Bipartite Graphs

---

## Proposition (1.6.1)

A graph is bipartite iff all its **cycles** are of even length.

# Bipartite Graphs

---

## Proposition (1.6.1)

A graph is bipartite iff all its **cycles** are of even length.

Proof ( $\Rightarrow$ ).

See (Bollobás 2002, Theorem 1.4).

1. Let  $G$  be a bipartite graph with two vertex classes  $V_1$  and  $V_2$  and let  $C := x_1x_2 \dots x_kx_1$  be a cycle in  $G$ .
2. We suppose that  $x_1 \in V_1$  (if not, just rename the vertex classes).
3. Therefore,  $x_2 \in V_2$ ,  $x_3 \in V_1$ , and so on. Hence,  $x_i \in V_1$  iff  $i$  is odd.
4. Since  $x_k \in V_2$ , we can conclude that  $C$  is an even cycle. ■



# Bipartite Graphs

---

Proof ( $\Leftarrow$ ).

1. Let every cycle of  $G$  an even cycle and let suppose  $G$  is connected.
2. Pick a vertex  $x \in V(G)$ .
3. Define the vertex sets

$$V_1 := \{y \in V(G) \mid d(x, y) \text{ is odd}\},$$

$$V_2 := V(G) \setminus V_1.$$

4. Note that  $V_1 \cap V_2 = \emptyset$  and  $V_1 \cup V_2 = V(G)$  because  $G$  is connected. Therefore, the set  $\{V_1, V_2\}$  partitions  $V(G)$ .
5. Note that if  $G$  had any edge between two vertices of the same set  $V$ , the graph  $G$  would have an odd cycle.
6. Hence,  $G$  is bipartite.
7. Now, if  $G$  is disconnected build the partition repeating of previous steps on each component of  $G$ .

# Bipartite Graphs

---

Proposition (1.6.1, previous version)

A graph is bipartite iff all its **cycles** are of even length.

# Bipartite Graphs

---

Proposition (1.6.1, previous version)

A graph is bipartite iff all its **cycles** are of even length.

Proposition (1.6.1, final version)

A graph is bipartite iff it contains no **odd cycle**.

# Bipartite Graphs

---

## Proposition (1.6.1, previous version)

A graph is bipartite iff all its **cycles** are of even length.

## Proposition (1.6.1, final version)

A graph is bipartite iff it contains no **odd cycle**.

## Discussion

Which is the smallest bipartite graph?

# Bipartite Graphs

---

## Definition

An  $r$ -partite graph is **complete** iff every two vertices from different **partition** classes are adjacent.

## Definition

A **complete multipartite graph** is a graph that is complete  $k$ -partite for some  $k$ .

# Bipartite Graphs

---

## Definition

An  $r$ -partite graph is **complete** iff every two vertices from different **partition** classes are adjacent.

## Definition

A **complete multipartite graph** is a graph that is complete  $k$ -partite for some  $k$ .

## Convention

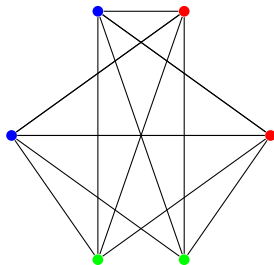
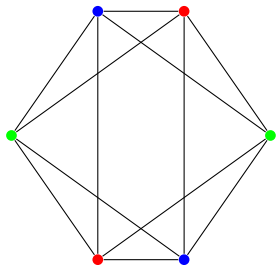
We denote by  $K_{n_1, \dots, n_r}$  the complete  $r$ -partite graph where  $n_1, \dots, n_r$  are the sizes of each vertex set in the **partition**.

# Bipartite Graphs

---

## Example

Two drawings of the tripartite graph  $K_{2,2,2}$ .

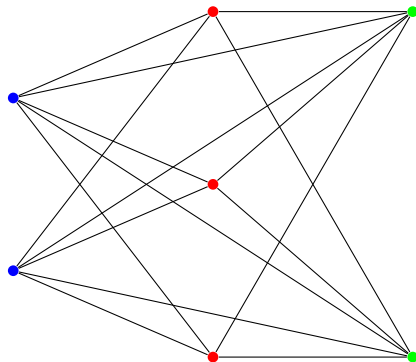


# Bipartite Graphs

---

## Example

The tripartite graph  $K_{2,3,2}$ .





# Bipartite Graphs

---

## Notation

The complete  $r$ -partite graph  $\overline{K^{n_1}} * \dots * \overline{K^{n_r}}$  is denoted by  $K_{n_1, \dots, n_r}$ .

# Representating Graphs

---

Let  $G$  be a graph with  $V = \{v_1, \dots, v_n\}$  and  $E = \{e_1, \dots, e_m\}$ .

# Representating Graphs

---

Let  $G$  be a graph with  $V = \{v_1, \dots, v_n\}$  and  $E = \{e_1, \dots, e_m\}$ .

## Definition

The **incidence matrix**  $B = (b_{ij})_{n \times m}$  of  $G$  is defined by

$$b_{ij} := \begin{cases} 1, & \text{if } v_i \in e_j; \\ 0, & \text{otherwise.} \end{cases}$$

# Representating Graphs

---

Let  $G$  be a graph with  $V = \{v_1, \dots, v_n\}$  and  $E = \{e_1, \dots, e_m\}$ .

## Definition

The **incidence matrix**  $B = (b_{ij})_{n \times m}$  of  $G$  is defined by

$$b_{ij} := \begin{cases} 1, & \text{if } v_i \in e_j; \\ 0, & \text{otherwise.} \end{cases}$$

## Definition

The **adjacency matrix**  $A = (a_{ij})_{n \times n}$  of  $G$  is defined by

$$a_{ij} := \begin{cases} 1, & \text{if } v_i v_j \in E; \\ 0, & \text{otherwise.} \end{cases}$$

# Other Notions of Graphs

---

## Notation

Let  $A$  be a set. The power set of  $A$  is denoted by  $\mathcal{P}(A)$ .

# Other Notions of Graphs

---

## Notation

Let  $A$  be a set. The power set of  $A$  is denoted by  $\mathcal{P}(A)$ .

## Definition

A **hypergraph** is an order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** such that the elements of  $E$  are non-empty subsets of  $V$ , i.e.  $E \subseteq \mathcal{P}(V) \setminus \{\emptyset\}$ .

# Other Notions of Graphs

---

## Notation

Let  $A$  be a set. The power set of  $A$  is denoted by  $\mathcal{P}(A)$ .

## Definition

A **hypergraph** is an order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** such that the elements of  $E$  are non-empty subsets of  $V$ , i.e.  $E \subseteq \mathcal{P}(V) \setminus \{\emptyset\}$ .

## Remark

Note that graphs are hypergraphs where the elements of their edge sets have cardinality two.

# Other Notions of Graphs

## Example

Hypergraph example.\*

$V = \{1, \dots, 7\}$ ,  $E = \{e_1, \dots, e_6\}$ , and

$$e_1 = \{4, 5, 6\},$$

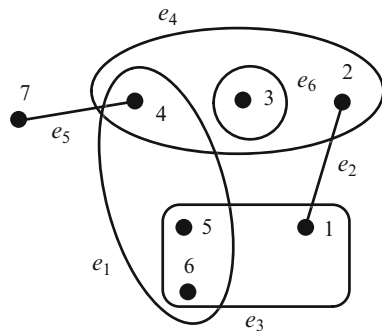
$$e_2 = \{1, 2\},$$

$$e_3 = \{1, 5, 6\},$$

$$e_4 = \{2, 3, 4\},$$

$$e_5 = \{4, 7\},$$

$$e_6 = \{3\}.$$



\*Figure source: H. Zhang et al. (2018, Fig. 1.1).



# Other Notions of Graphs

---

## Definition

A **directed graph** (or **digraph**) is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and two functions  $\text{init} : E \rightarrow V$  and  $\text{ter} : E \rightarrow V$ .

# Other Notions of Graphs

---

## Definition

A **directed graph** (or **digraph**) is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and two functions  $\text{init} : E \rightarrow V$  and  $\text{ter} : E \rightarrow V$ .

The functions  $\text{init}$  and  $\text{ter}$  assign to every edge  $e$  an **initial vertex**  $\text{init}(e)$  and a **terminal vertex**  $\text{ter}(e)$ .

# Other Notions of Graphs

---

## Definition

A **directed graph** (or **digraph**) is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and two functions  $\text{init} : E \rightarrow V$  and  $\text{ter} : E \rightarrow V$ .

The functions  $\text{init}$  and  $\text{ter}$  assign to every edge  $e$  an **initial vertex**  $\text{init}(e)$  and a **terminal vertex**  $\text{ter}(e)$ .

An edge  $e$  is **directed** from  $\text{init}(e)$  to  $\text{ter}(e)$ .

# Other Notions of Graphs

---

## Definition

A **directed graph** (or **digraph**) is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and two functions  $\text{init} : E \rightarrow V$  and  $\text{ter} : E \rightarrow V$ .

The functions  $\text{init}$  and  $\text{ter}$  assign to every edge  $e$  an **initial vertex**  $\text{init}(e)$  and a **terminal vertex**  $\text{ter}(e)$ .

An edge  $e$  is **directed** from  $\text{init}(e)$  to  $\text{ter}(e)$ .

Two or more edges are **multiple edges** iff they are edges between the same pair of vertices. If also they have the same direction they are **parallel edges**.

# Other Notions of Graphs

---

## Definition

A **directed graph** (or **digraph**) is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and two functions  $\text{init} : E \rightarrow V$  and  $\text{ter} : E \rightarrow V$ .

The functions  $\text{init}$  and  $\text{ter}$  assign to every edge  $e$  an **initial vertex**  $\text{init}(e)$  and a **terminal vertex**  $\text{ter}(e)$ .

An edge  $e$  is **directed** from  $\text{init}(e)$  to  $\text{ter}(e)$ .

Two or more edges are **multiple edges** iff they are edges between the same pair of vertices. If also they have the same direction they are **parallel edges**.

An edge  $e$  is a **loop** iff  $\text{init}(e) = \text{ter}(e)$ .

# Other Notions of Graphs

---

## Definition

A **multigraph** is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and one function  $E \rightarrow V \cup [V]^2$ .

# Other Notions of Graphs

---

## Definition

A **multigraph** is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and one function  $E \rightarrow V \cup [V]^2$ .

## Remark

Note that multigraphs can have loops and multiple edges.

# Other Notions of Graphs

---

## Definition

A **multigraph** is a order pair  $(V, E)$  of disjoint sets of **vertices** and **edges** and one function  $E \rightarrow V \cup [V]^2$ .

## Remark

Note that multigraphs can have loops and multiple edges.

## Remark

Note that graphs are multigraphs without loops or multiple edges.



# Colouring

# Colouring Graphs

---

## Definition

Let  $G = (V, E)$  be a graph and  $S$  be a set whose elements are the available **colours**. A **vertex colouring** of  $G$  is a function

$$c : V \rightarrow S$$

such that  $c(v) \neq c(w)$  whenever  $v$  and  $w$  are adjacent.

# Colouring Graphs

---

## Definition

A  **$k$ -colouring** of a graph  $G = (V, E)$  is a vertex colouring

$$c : V \rightarrow \{1, \dots, k\}$$

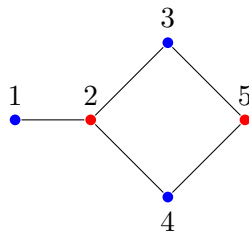
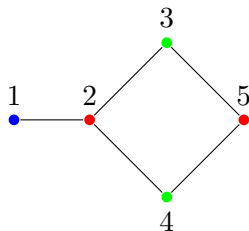
of  $G$ .

# Colouring Graphs

---

## Example

A 3-colouring and a 2-colouring for the same graph.



# Colouring Graphs

---

## Definition

Let  $G$  be a graph and  $k$  be the smallest integer such that  $G$  has a  $k$ -colouring. The number  $k$  is the **(vertex-)chromatic number** of  $G$ ; denoted  $\chi(G)$ .

# Colouring Graphs

---

## Definition

Let  $G$  be a graph and  $k$  be the smallest integer such that  $G$  has a  $k$ -colouring. The number  $k$  is the **(vertex-)chromatic number** of  $G$ ; denoted  $\chi(G)$ .

## Definition

A graph  $G$  is  **$k$ -colourable** iff  $\chi(G) \leq k$ .

# Colouring Planar Graphs

## Example

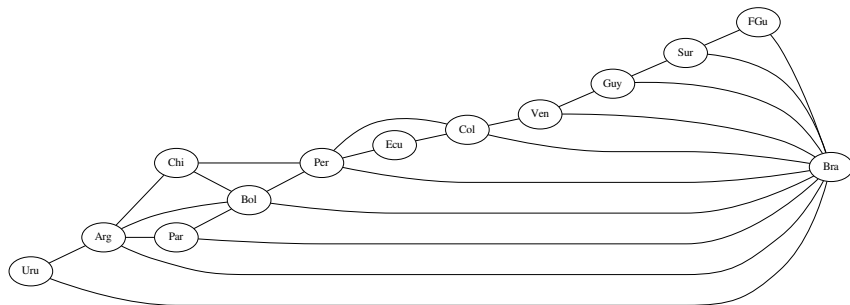
How many colours do we need for colouring South America's map?



# Colouring Planar Graphs

## Example (continuation)

Graph associated with South America's map where the vertices are the countries and one edge between two vertices means that the countries share a border.



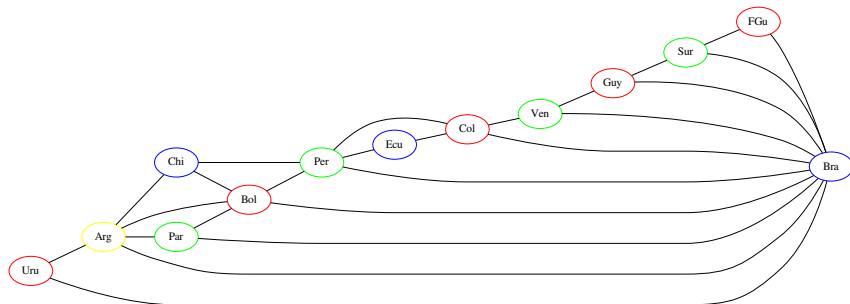
Continued on next slide



# Colouring Planar Graphs

## Example (continuation)

Let  $G$  be the associated graph with South America's map. Then  $\chi(G) = 4$ .



# Colouring Planar Graphs

---

Theorem (Four colour theorem, 5.1.1)

Every planar graph is 4-colourable.

Flows

# Flows in Networks

---

## Theorem (Max-flow min-cut theorem, 6.2.2)

In a network, the maximum total value of a flow equals the minimum capacity of a cut (Ford and Fulkerson 1956).

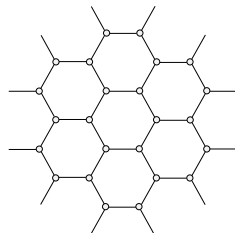
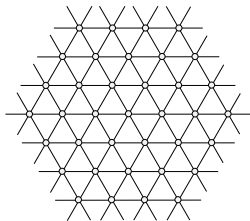
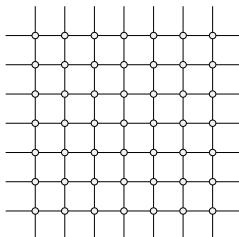
# Infinite Graphs

# Infinite Graphs

---

## Example

The square, triangular and hexagonal lattices.\*



---

\*Figure source: Bondy and Murty (2008, Fig. 1.27).

# Ramsey Theory for Graphs

# Ramsey's Original Theorems

---

## Theorem (9.1.1)

For every  $r \in \mathbb{N}$  there exists an  $n \in \mathbb{N}$  such that every graph of order at least  $n$  contains either  $K^r$  or  $\overline{K^r}$  as an induced subgraph. By Ramsey (1930).



# Historical Remarks

# Historical Remarks

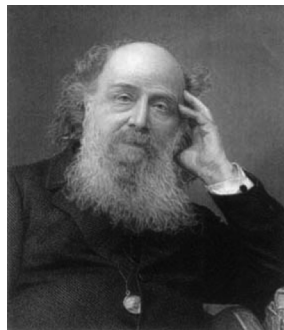
---

## On the term 'graph'

Sylvester (1878) introduced the term 'graph' on a note in *Nature* in the context of mathematics and chemistry (Biggs, Lloyd and Wilson 1998).

In relation to this term, the above authors wrote (p. 65):

'So the credit (or blame) for the use of this term must be ascribed to Sylvester.'



James Joseph Sylvester (1814–1897)\*

---

\*Image from the [MacTutor History of Mathematics Archive](#).

# Historical Remarks

---

## Previous definitions of graphs

König (1916, p. 453):

*‘Es sei eine endliche Anzahl von Punkten gegeben; gewisse Paare, die man aus diesen Punkten auswählen kann, sollen durch eine oder mehrere (endlich viele) Kanten verbunden werden. Eine auf diese Weise entstehende Figur wird im allgemeinen als ein Graph bezeichnet.’*

Translation (Biggs, Lloyd and Wilson 1998, p. 203):

*‘Let a finite number of **points** be given: then one can choose certain pairs of the points so that one or more (but finitely many) **edges** join them. A figure constructed in this way we shall generally call a **graph**.’*

# Historical Remarks

---

## Previous definitions of graphs

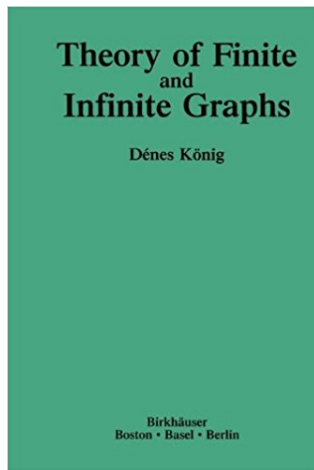
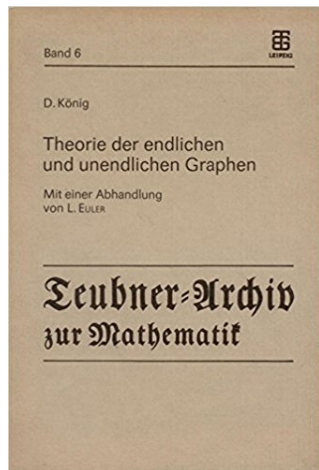
Whitney (1931, p. 378):

*'Let a finite number of curves, or **edges**, whose end-points we call **vertices**, intersect at no other points than these vertices. Let the system be connected, that is, any two vertices are joined by a succession of edges, each two successive edges having a vertex in common. This forms a **graph**.'*

# Historical Remarks

---

The first book on graph theory was written in German by König in 1936 and only translated to English in 1990



# Logic of Graphs

# Logic of Graphs

---

## First-order logics

In the **first-order** theory of graphs, the universe of discourse are the vertices and the language has a proper binary predicate `edg` representing the relation of adjacency between vertices.

## Example

Our definition of graph satisfies the following axioms (Goldberg 1993):

$\forall v(\neg \text{edg}(v, v))$	(no loops)
$\forall v \forall w(\text{edg}(v, w) \Rightarrow \text{edg}(w, v))$	(edges are undirected)

# Logic of Graphs

---

## Monadic second-order logic

In **monadic** second-order logic in addition to the quantification over individual variables, we can also quantifier over **sets of variables**, i.e. we can quantifier over **properties**.

## Notation

We use uppercase variables for denoting sets of vertices, and lowercase variables for denoting individual vertices.



# Logic of Graphs

---

## Monadic second-order logic

In **monadic** second-order logic in addition to the quantification over individual variables, we can also quantifier over **sets of variables**, i.e. we can quantifier over **properties**.

## Notation

We use uppercase variables for denoting sets of vertices, and lowercase variables for denoting individual vertices.

## Example

A graph satisfies the following sentence if only if it is **disconnected** (Courcelle and Engelfriet 2012):

$$\exists X[\exists x.x \in X \wedge \exists y.y \notin X \wedge \forall x \forall y(\text{edg}(x, y) \Rightarrow (x \in X \Leftrightarrow y \in X))]$$

# Undecidable Problems

# The $r$ -Neighbourhood Problem

---

## Definition

A **rooted graph** is a graph with a vertex considered as special (see, e.g. Gross, Yellen and P. Zhang (2013)).

# The $r$ -Neighbourhood Problem

---

## Definition

A **rooted graph** is a graph with a vertex considered as special (see, e.g. Gross, Yellen and P. Zhang (2013)).

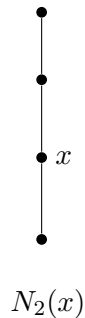
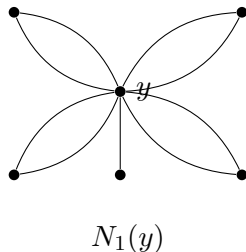
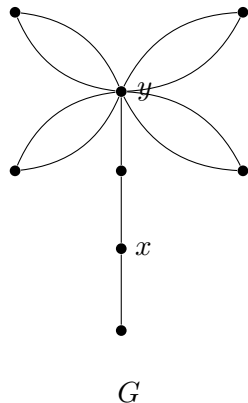
## Definition

Let  $G$  be a graph,  $v$  a vertex and  $k$  a positive integer. The  **$r$ -neighbourhood** of  $v$ , denoted  $N_r(v)$ , is the **subgraph induced** by the set of vertices of **distance** at most  $r$  from  $v$ . The graph  $N_r(v)$  is a rooted graph with root  $v$ .

# The $r$ -Neighbourhood Problem

## Example

A graph  $G$  and two  $r$ -neighbourhoods.



# The $r$ -Neighbourhood Problem

---

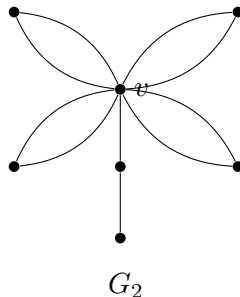
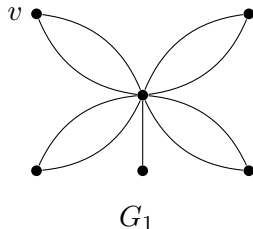
## Definition

Let  $G$  be a graph. The  **$r$ -neighbourhood** set of  $G$ , denoted  $\mathcal{N}_r(G)$ , is the set of **isomorphism** classes of  $r$ -neighbourhoods of vertices in  $G$ .

# The $r$ -Neighbourhood Problem

## Example

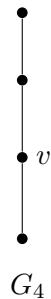
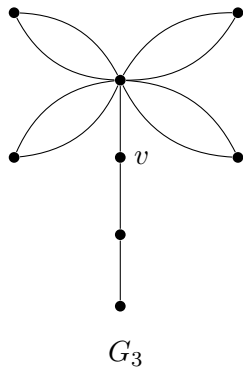
For the graph  $G$  in [this](#) example, the set  $\mathcal{N}_2(G)$  consists of the rooted graphs  $G_1$ ,  $G_2$ ,  $G_3$ ,  $G_4$  and  $G_5$ .



Continued on next slide

# The $r$ -Neighbourhood Problem

## Example (continuation)





# The $r$ -Neighbourhood Problem

---

## The $r$ -neighbourhood problem

Given a positive integer  $k$  and a finite set  $\Phi$  of rooted graphs, is there a (connected) graph  $G$  whose  $r$ -neighbourhood set is  $\Phi$ ?

---

\*Also in a Russian publication translated as V. K. Bulitko (1973), Graphs with Prescribed Environments of the Vertices.

# The $r$ -Neighbourhood Problem

---

## The $r$ -neighbourhood problem

Given a positive integer  $k$  and a finite set  $\Phi$  of rooted graphs, is there a (connected) graph  $G$  whose  $r$ -neighbourhood set is  $\Phi$ ?

## Theorem

The  $r$ -neighbourhood problem is undecidable (Winkler 1983, Theorem 5).\*

---

\*Also in a Russian publication translated as V. K. Bulitko (1973), Graphs with Prescribed Environments of the Vertices.

# Undecidable Problems

---

## Some references to other undecidable problems

- Csóka (2012). 'An Undecidability Result on Limits of Sparse Graphs'.
- Jacobs (1994). 'Undecidability of Winkler's  $r$ -Neighborhood Problem for Covering Digraphs'.
- Burr (1984). 'Some Undecidable Problems Involving the Edge-Coloring and Vertex-Coloring of Graphs'.
- Foldes and Steinberg (1980). 'A Topological Space for which Graph Embeddability is Undecidable'.

# Some Named Graphs

# Some Named Graphs

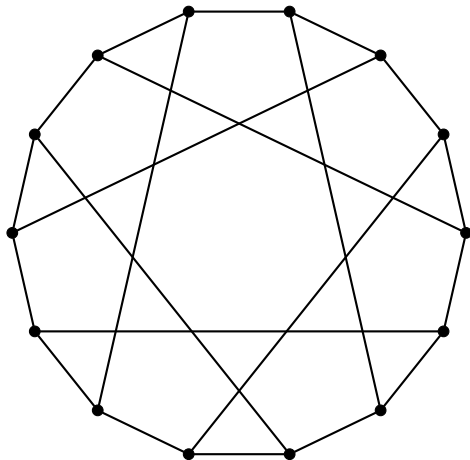
---

## Remark

The graphs on this section were drawn using the `tkz-berge.sty` package by Matthes (2011).

# Heawood Graph

---



Vertices 14

Edges 21

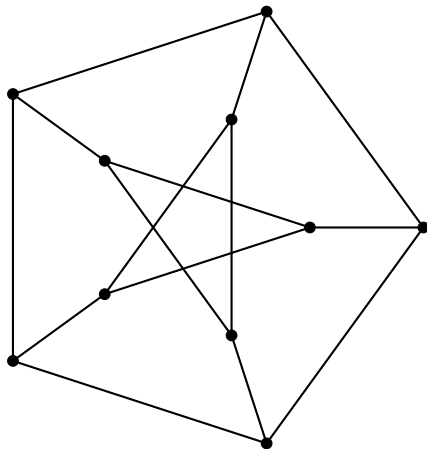
3-regular

Girth 6

Connected

# Petersen Graph

---



Vertices 10

Edges 15

3-regular

Girth 5

Connected

# Appendix: Order Theory



# Partially Ordered Sets

---

## Definition

A binary relation  $\preceq$  on a set  $A$  is a **partial ordering** iff it satisfies the following properties:

$$\forall x(x \preceq x) \quad (\text{reflexivity})$$

$$\forall x \forall y (x \preceq y \wedge y \preceq x \Rightarrow x = y) \quad (\text{anti-symmetry})$$

$$\forall x \forall y \forall z (x \preceq y \wedge y \preceq z \Rightarrow x \preceq z) \quad (\text{transitivity})$$

# Partially Ordered Sets

---

## Definition

A binary relation  $\preceq$  on a set  $A$  is a **partial ordering** iff it satisfies the following properties:

$$\forall x (x \preceq x) \quad \text{(reflexivity)}$$

$$\forall x \forall y (x \preceq y \wedge y \preceq x \Rightarrow x = y) \quad \text{(anti-symmetry)}$$

$$\forall x \forall y \forall z (x \preceq y \wedge y \preceq z \Rightarrow x \preceq z) \quad \text{(transitivity)}$$

## Definition

Let  $\preceq$  be a partial ordering on a set  $A$ . The relational structure  $(A, \preceq)$  is a **partially ordered set** (or **poset**).

# Notable Elements

---

Let  $(A, \preceq)$  be a poset.

## Definition

An element  $a \in A$  is the **greatest element** (*máximo*) of  $(A, \preceq)$  iff  $b \preceq a$  for all  $b \in A$ .

## Notable Elements

---

Let  $(A, \preceq)$  be a poset.

### Definition

An element  $a \in A$  is the **greatest element** (*máximo*) of  $(A, \preceq)$  iff  $b \preceq a$  for all  $b \in A$ .

### Definition

An element  $a \in A$  is the **least element** (*mínimo*) iff  $a \preceq b$  for all  $b \in A$ .

# Notable Elements

---

Let  $(A, \preceq)$  be a poset.

## Definition

An element  $a \in A$  is the **greatest element** (*máximo*) of  $(A, \preceq)$  iff  $b \preceq a$  for all  $b \in A$ .

## Definition

An element  $a \in A$  is the **least element** (*mínimo*) iff  $a \preceq b$  for all  $b \in A$ .

## Definition

An element  $a \in A$  is a **maximal** of  $(A, \preceq)$  iff there is no  $b \in A$  such that  $a \prec b$ .

## Notable Elements

---

Let  $(A, \preceq)$  be a poset.

### Definition

An element  $a \in A$  is the **greatest element** (*máximo*) of  $(A, \preceq)$  iff  $b \preceq a$  for all  $b \in A$ .

### Definition

An element  $a \in A$  is the **least element** (*mínimo*) iff  $a \preceq b$  for all  $b \in A$ .

### Definition

An element  $a \in A$  is a **maximal** of  $(A, \preceq)$  iff there is no  $b \in A$  such that  $a \prec b$ .

### Definition

An element  $a \in A$  is a **minimal** of  $(A, \preceq)$  iff there is no  $b \in A$  such that  $b \prec a$ .

# Notable Elements

## Example

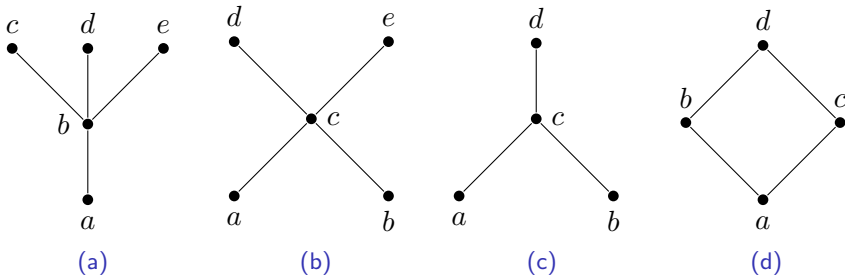


Fig.	Least element	Greatest element	Maximals	Minimals
(a)	$a$		$c, d, e$	$a$
(b)			$d, e$	$a, b$
(c)		$d$	$d$	$a, b$
(d)	$a$	$d$	$d$	$a$

# Totally Ordered Sets

---

## Definition

A binary relation  $\preceq$  on a set  $A$  is a **total ordering** iff it satisfies the following properties:

$$\forall x \forall y (x \preceq y \wedge y \preceq x \Rightarrow x = y) \quad (\text{anti-symmetry})$$

$$\forall x \forall y \forall z (x \preceq y \wedge y \preceq z \Rightarrow x \preceq z) \quad (\text{transitivity})$$

$$\forall x \forall y (x \preceq y \vee y \preceq x) \quad (\text{totality})$$

## Remark

Note that totality implies reflexivity.



# Totally Ordered Sets

---

## Definition

A binary relation  $\preceq$  on a set  $A$  is a **total ordering** iff it satisfies the following properties:

$\forall x \forall y (x \preceq y \wedge y \preceq x \Rightarrow x = y)$	(anti-symmetry)
$\forall x \forall y \forall z (x \preceq y \wedge y \preceq z \Rightarrow x \preceq z)$	(transitivity)
$\forall x \forall y (x \preceq y \vee y \preceq x)$	(totality)

## Remark

Note that totality implies reflexivity.

## Definition

Let  $\preceq$  be a total ordering on a set  $A$ . The relational structure  $(A, \preceq)$  is a **totally ordered set** (also called **linearly ordered set** or **chain**).

# Totally Order Sets

---

## Remark

The term **chain** also can refer to a totally ordered subset of some partially ordered set (Vialar 2016).

# Appendix: Topology

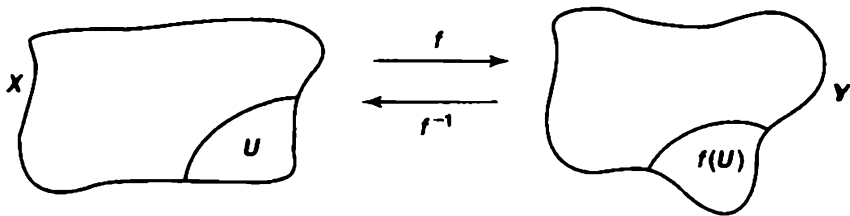
# Topology

## Definition

Let  $(X, \mathcal{T}_X)$  and  $(Y, \mathcal{T}_Y)$  be two topological spaces. A function  $f : X \rightarrow Y$  is a **homeomorphism** iff:

- the function is a bijection and
- both the function and the inverse function are continuous.









That is,  $f(U)$  is open if and only if  $U$  is open.\*










\*Figure source: Munkres (2000, Fig. 18.1).

# References







---

-  Biggs, Norman L., Lloyd, E. Keith and Wilson, Robin J. [1976] (1998). *Graph Theory*. 1736–1936. Reimpression with corrections. Clarendon Press (cit. on pp. 138, 218, 219).
-  Bollobás, Béla [1998] (2002). *Modern Graph Theory*. 3rd printing. Graduate Texts in Mathematics. Springer-Verlag. DOI: 10.1007/978-1-4612-0619-4 (cit. on pp. 175, 176).
-  Bondy, J. A. and Murty, U. S. R. (2008). *Graph Theory*. Springer-Verlag (cit. on pp. 14–18, 214).
-  Burr, Stefan A. (1984). ‘Some Undecidable Problems Involving the Edge-Coloring and Vertex-Coloring of Graphs’. In: *Discrete Mathematics* 50, pp. 171–177. DOI: 10.1016/0012-365X(84)90046-3 (cit. on p. 235).
-  Cohn, P. M. [1965] (1981). *Universal Algebra*. Revised edition. Vol. 6. Mathematics and Its Applications. D. Reidel Publishing Company (cit. on pp. 35, 36).
-  Courcelle, Bruno and Engelfriet, Joost (2012). *Graph Structure and Monadic Second-Order Logic. A Language-Theoretic Approach*. CUP (cit. on pp. 167, 168, 224, 225).
-  Csóka, Endre (2012). ‘An Undecidability Result on Limits of Sparse Graphs’. In: *The Electronic Journal of Combinatorics* 19.2. P21 (cit. on p. 235).
-  Diestel, Reinhard [1997] (2017). *Graph Theory*. 5th ed. Springer. DOI: 10.1007/978-3-662-53622-3 (cit. on pp. 4–6, 9, 21, 82, 101–104, 127–130, 139, 163).









# References

-  Diestel, Reinhard and Leader, Imre (2001). 'Normal Spanning Trees, Aronszajn Trees and Excluded Minors'. In: *Journal of the London Mathematical Society* 63.1, pp. 16–32. DOI: [10.1112/S0024610700001708](https://doi.org/10.1112/S0024610700001708) (cit. on p. 161).
-  Eells, James and Toledo, Domingo, eds. (1992). *Hassler Whitney. Collected Papers*. Vol. 1. Birkhäuser. DOI: [10.1007/978-1-4612-2972-8](https://doi.org/10.1007/978-1-4612-2972-8) (cit. on p. 256).
-  Foldes, Stephane and Steinberg, Richard (1980). 'A Topological Space for which Graph Embeddability is Undecidable'. In: *Journal of Combinatorial Theory, Series B* 29.3, pp. 342–344. DOI: [10.1016/0095-8956\(80\)90092-1](https://doi.org/10.1016/0095-8956(80)90092-1) (cit. on p. 235).
-  Ford, L. R. and Fulkerson, D. R. (1956). 'Maximal Flow through a Network'. In: *Canadian Journal of Mathematics* 8, pp. 399–404. DOI: [10.4153/CJM-1956-045-5](https://doi.org/10.4153/CJM-1956-045-5) (cit. on p. 212).
-  Goldberg, Leslie Ann (1993). 'Polynomial Space Polynomial Delay Algorithms for Listing Families of Graphs'. In: *Proceedings of the Twenty-fifth Annual ACM Symposium on Theory of Computing (STOC '93)*, pp. 218–225. DOI: [10.1145/167088.167160](https://doi.org/10.1145/167088.167160) (cit. on p. 223).
-  Gross, Jonathan L., Yellen, Jay and Zhang, Ping, eds. [2004] (2013). *Handbook of Graph Theory*. 2nd ed. Discrete Mathematics and Its Applications. CRC Press (cit. on pp. 227, 228).
-  Harary, Frank (1969). *Graph Theory*. Addison-Wesley (cit. on pp. 88, 89, 92).

# References

-  Harary, Frank and Read, Ronald C. (1974). 'Is The Null-graph a Pointless Concept?' In: *Graphs and Combinatorics*. Ed. by Bari, Ruth A. and Harary, Frank. Vol. 406. Lecture Notes in Mathematics. Springer, pp. 37–44. DOI: [10.1007/BFb0066433](https://doi.org/10.1007/BFb0066433) (cit. on pp. 11–13).
-  Jacobs, D. P. (1994). 'Undecidability of Winkler's r-Neighborhood Problem for Covering Digraphs'. In: *Journal of Combinatorial Theory, Series B* 60.2, pp. 254–267. DOI: [10.1006/jctb.1994.1017](https://doi.org/10.1006/jctb.1994.1017) (cit. on p. 235).
-  Knuth, Donald E. [1968] (1997). *The Art of Computer Programming*. 3rd ed. Vol. 1. Fundamental Algorithms. Addison-Wesley Professional (cit. on p. 154).
-  König, Dénes (1916). 'Über Graphen und ihre Anwendung auf Determinantentheorie und Mengenlehre (On Graphs and their Applications in Determinant Theory and Set Theory)'. In: *Mathematische Annalen* 77.4, pp. 453–465. DOI: [10.1007/BF01456961](https://doi.org/10.1007/BF01456961) (cit. on p. 219).
-  Lammich, Peter and Neumann, René (2015). 'A Framework for Verifying Depth-First Search Algorithms'. In: *Proceedings of the 2015 Conference on Certified Programs and Proofs (CPP '15)*, pp. 137–146. DOI: [10.1145/2676724.2693165](https://doi.org/10.1145/2676724.2693165) (cit. on pp. 167, 168).
-  Lovász, László (2012). *Large Networks and Graph Limits*. Vol. 60. Colloquium Publications. American Mathematical Society. DOI: [10.1090/coll/060](https://doi.org/10.1090/coll/060) (cit. on pp. 42–44).

## References

-  Matthes, Alain (2011). *NamedGraphs v 1.00c*. Documentation accompanying the `tkz-berge.sty` v 1.00c package (cit. on p. 237).
-  Munkres, James R. [1974] (2000). *Topology*. 2nd ed. Prentice Hall (cit. on p. 252).
-  Ramsey, F. P. (1930). 'On a Problem of Formal Logic'. In: *Proceedings of the London Mathematical Society* s2-30.1, pp. 264–286. DOI: [10.1112/plms/s2-30.1.264](https://doi.org/10.1112/plms/s2-30.1.264) (cit. on p. 216).
-  Sylvester, J. J. (1878). 'Chemistry and Algebra'. In: *Nature* 17.432, p. 284. DOI: [10.1038/017284a0](https://doi.org/10.1038/017284a0) (cit. on p. 218).
-  Vialar, Thierry (2016). *Handbook of Mathematics*. Hdbom (cit. on p. 250).
-  Whitney, Hassler (1931). 'A Theorem on Graphs'. In: *Annals of Mathematics* 32.2. Reprinted in Eells and Toledo (1992), pp. 378–390. DOI: [10.2307/1968197](https://doi.org/10.2307/1968197) (cit. on p. 220).
-  Winkler, Peter M. (1983). 'Existence of Graphs with a Given Set of  $r$ -Neighborhoods'. In: *Journal of Combinatorial Theory, Series B* 34.2, pp. 165–176. DOI: [10.1016/0095-8956\(83\)90016-3](https://doi.org/10.1016/0095-8956(83)90016-3) (cit. on pp. 233, 234).
-  Zhang, Hongliang, Song, Lingyang, Han, Zhu and Zhang, Yingjun (2018). *Hypergraph Theory in Wireless Communication Networks*. Springer. DOI: [10.1007/978-3-319-60469-5](https://doi.org/10.1007/978-3-319-60469-5) (cit. on p. 192).



$d(G)$ , average degree, 59

$\text{diam}(G)$ , diameter, 85

$g(G)$ , girth, 78, 79

$\text{rad}(G)$ , radius, 90, 91

$\chi(G)$ , chromatic number, 205, 206

$\Delta(G)$ , maximum degree, 59

$\delta(G)$ , minimum degree, 59

$\epsilon(G)$ , number of edges by vertex, 59

$\kappa(G)$ , connectivity, 113–118

$\lambda(G)$ , edge-connectivity, 123–126

$|G|$ , order, 19, 20

---

\*TODO: The links to page numbers are not working. Tested with TeX Live 2018, pdfTeX 3.14159265-2.6-1.40.19, beamer.cls v3.50 and makeindex v2.15.