THE AMERICAN
UNIVERSITY IN CAIRO

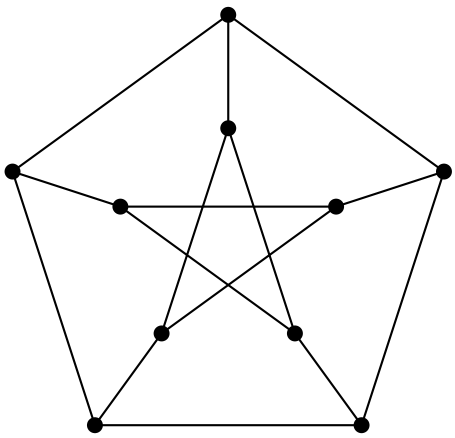
AUC

DEPARTMENT OF MATHEMATICS

SPRING TERM 2026

Graph Theory

Lecture Notes



Lecturer:
Isabel Müller

Term:
Spring 2026

"An die Professorin, der ich meine Wertschätzung nicht zeigen konnte,
und an die Professorin, der ich es niemals vergelten kann."

Contents

Chapter 1

Graphs

1. The Basics

1.1 Recall

1. A **set** is merely an accumulation of objects. These objects are called **elements** of the set. If an object x is an element of S, we write $x \in S$. The set of all elements with a certain property P is denoted via $\{x \mid x \text{ has property } P\}$.
2. An n**-ary relation** R on a set A is a subset of the power set of A^n, i.e., $R \subseteq \mathcal{P}(A^n)$. If $n = 2$, we call the relation **binary**.

   A binary relation R on a set A is called:

   (i) **symmetric** if $R(a, b)$ implies $R(b, a)$ for all $a, b \in A$.

   (ii) **asymmetric** if $R(a, b)$ implies $\neg R(b, a)$ for all $a, b \in A$.

   (iii) **antisymmetric** if $R(a, b) \wedge R(b, a)$ implies $a = b$ for all $a, b \in A$.

   (iv) **reflexive** if $R(a, a)$ for all $a \in A$.

   (v) **irreflexive** if $\neg R(a, a)$ for all $a \in A$.

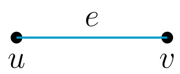
   (vi) **transitive** if $R(a, b) \wedge R(b, c)$ implies $R(a, c)$ for all $a, b, c \in A$.

Definition 1.2

A **graph** $G = (V, E)$ is a pair of sets V and E such that E consists of subsets of V of size two. V is called the set of **vertices** and E the set of **edges**. A graph G is called **finite** if V is a finite set. The **order** $|G|$ of a graph $G = (V, E)$ is the cardinality of its vertex set, so $|G| = |V|$. The **size** $\|G\|$ of G is the cardinality of its edge set, $\|G\| = |E|$.

1.3 Visualisation

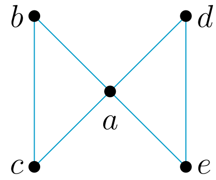
Let $G = (V, E)$ be a graph. We visualise vertices $u, v \cdots \in V$ by dots and edges $e = \{u, v\} \in E$ by the diagram:



Example 1.4. *Bowtie Graph* Let $G = (V, E)$ be the graph with $V = \{a, b, c, d, e\}$ and

$$E = \{\{a, b\}, \{a, c\}, \{a, d\}, \{a, e\}, \{b, c\}, \{d, e\}\}.$$

The graph G has order 5 and size 6. It can be visualized via:



This visualisation motivates its name: **bowtie graph**.

1.5 Notation

1. For a graph $G = (V, E)$ we may denote its vertex set by $V(G)$ or V_G for clarity.
2. Similarly, we often denote E by $E(G)$ or E_G.
3. We denote an edge $\{u, v\}$ simply by uv.
4. Edges are often called $e, e_1, e_2, f \ldots$, while vertices are called $u, v, x, y, \ldots$.

Definition 1.6

Let $G = (V, E)$ be a graph.

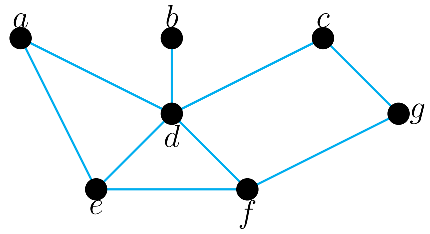
1. If $uv \in E$ is an edge, then we say that u and v are **adjacent** or **neighbours**. If $uv \notin E$, we call u and v **nonadjacent**.
2. If $e = uv \in E$, we say that u and v are the **end vertices** of e or that they are **incident** with e.
3. The **neighborhood** $N(v)$ of a vertex $v \in V$ is the set of all vertices adjacent to v, i.e., $N(v) = \{u \in V \mid uv \in E\}$. The **closed neighborhood** $N[v]$ of v is $N[v] := N(v) \cup \{v\}$.
4. The **neighborhood** $N(S)$ of a set of vertices is defined as $N(S) := \bigcup_{v \in S} N(v)$. Similarly, the **closed neighborhood** $N[S]$ is set to be $N[S] := N(S) \cup S (= \bigcup_{v \in S} N[v])$.
5. The **degree** $\deg(v)$ of $v \in V$ is the number of edges incident with v, i.e., $\deg(v) := |\{e \in E \mid v \in e\}| = |N(v)|$.
6. The **maximum degree** $\Delta(G)$ of G is defined as

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

Similarly, $\delta(G) := \min\{\deg(v) \mid v \in V\}$ is the **minimum degree** of G.

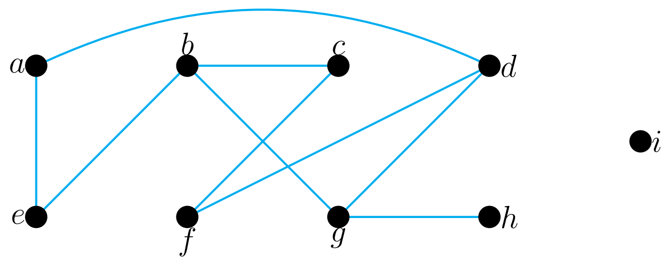
7. The **degree sequence** of a graph G is the sequence containing all degrees of the vertices of G (with repetition) in decreasing order.

Example 1.7. Consider G given by:



Then $\Delta(G) = 5$, $\delta(G) = 1$. $N(e) = \{a, d, f\}$, $N[b] = \{b, d\}$. $N[a, g] = \{a, c, d, e, f, g\}$ Order of G, size of G is 9. Degree sequence $(5, 3, 3, 2, 2, 2, 1)$.

Example 1.8. Consider G given via the diagram:



Then $V(G) = \{a, b, c, d, e, f, g, h, i\}$
$E(G) = \{\{a, d\}, \{a, e\}, \{b, c\}, \{b, e\}, \{b, g\}, \{c, f\}, \{d, f\}, \{d, g\}, \{g, h\}\}$

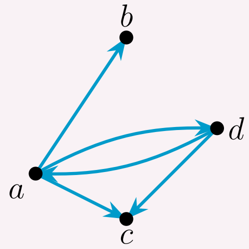
Order $|G| = 9$, size of G is 9, degree sequence $(3, 3, 3, 2, 2, 2, 2, 1, 0)$.
$N(f) = \{c, d\}$, $N[d, e] = \{a, b, c, d, e, f\}$, $\Delta(G) = 3$, $\delta(G) = 0$.

Remark 1.9. A graph can be considered as a set V together with a binary relation E on V which is symmetric and irreflexive.

Definition 1.10 Variants of Graphs

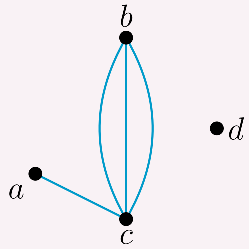
1. If $G = (V, E)$ and we replace E with a set of ordered pairs, then we call G a **directed graph** or **digraph**.

   Ex: $V(G) = \{a, b, c, d\}$, $E(G) = \{(a, b), (a, c), (a, d), (d, a), (d, c)\}$



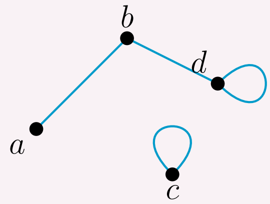
2. If $G = (V, E)$ and we replace E by a multiset (iterations of the same elements are distinguished), then we call G a **multigraph**.

   Ex: $E = [\{a, c\}, \{b, c\}, \{b, c\}, \{b, c\}]$



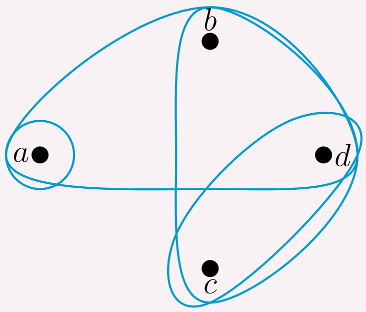
3. If $G = (V, E)$ and we extend E by allowing loops, we call G a **pseudograph**.

   Ex: $E = \{\{a, b\}, \{b, d\}, \{c, c\}, \{d, d\}\}$



4. If we allow edges to be arbitrary sets of vertices instead of 2-elementary ones, we call G a **hypergraph**.

   Ex: $E = \{\{a\}, \{a, b, d\}, \{b, c, d\}, \{c, d\}\}$



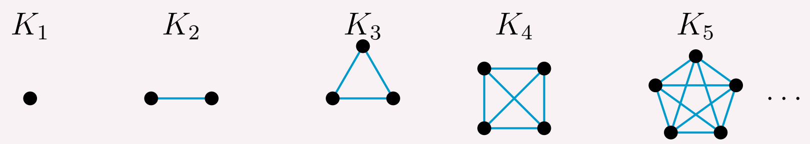
1.11 Setting

In this lecture, unless otherwise stated, by a graph we mean a finite, simple graph with $|V| \geq 1$.

Definition 1.12

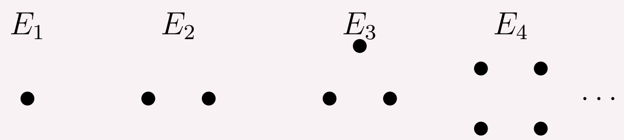
- The **complete graph** K_n for $n \geq 1$ is the graph consisting of n vertices such that any two vertices are adjacent.

  e.g.



- The **empty graph** E_n is the graph consisting of n vertices and no edges.

  e.g.



Theorem 1.13 The Handshaking Lemma

If $G = (V, E)$ is a graph, then

$$\sum_{v \in V} \deg(v) = 2|E|. \quad (*)$$

Proof. We proceed by induction on $n := |E|$.

n=0: If $|E| = 0$, then $\deg(v) = 0$ for any $v \in V$, whence clearly

$$0 = \sum_{v \in V} \deg(v) = 2|E| = 0.$$

n → n+1: Assume $(*)$ holds for any $G' = (V', E')$ with $|E'| = n$ (I.H.) and consider $G = (V, E)$ with $|E| = n + 1 (\geq 1)$ arbitrary. Let $e \in E$ arbitrary and consider $G' = (V, E \setminus \{e\})$. Then, if $e = uv$, we get $|E(G)| = |E(G')| + 1$ and

$$\deg_G(u) = \deg_{G'}(u) + 1 \quad \text{and} \quad \deg_G(v) = \deg_{G'}(v) + 1, \text{ whence}$$

$$\begin{aligned} 2|E(G)| &= 2|E(G')| + 2 \\ &\overset{\text{I.H.}}{=} \sum_{w \in V} \deg_{G'}(w) + 2 \\ &= \sum_{w \in V \setminus \{u,v\}} \deg_{G'}(w) + \deg_{G'}(u) + 1 + \deg_{G'}(v) + 1 \\ &= \sum_{w \in V} \deg_G(w), \quad \text{as desired.} \end{aligned}$$ □

Corollary 1.14

Any graph G has an even number of vertices of odd degree.

Proof. Exercise. $\square$

Corollary 1.15

For any graph $G = (V, E)$ we have

$$\delta(G) \leq 2\frac{|E|}{|V|} \leq \Delta(G).$$

Proof.

$$|V| \cdot \delta(G) = \sum_{v \in V} \delta(G) \leq \sum_{v \in V} \deg(v) \leq \sum_{v \in V} \Delta(G) = |V|\Delta(G)$$

Using Theorem 1.13, $\sum \deg(v) = 2|E|$. Dividing by $|V|$ yields the result. $\square$

Lemma 1.16

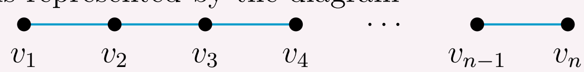
If $|G| \geq 2$, then G contains at least two vertices of the same degree.

Proof. If G has two vertices of degree 0, then we are done. Otherwise, we may assume that G has none. If $|G| = n$, and $v \in V$, then $1 \leq \deg(v) \leq n - 1$. Note that this leaves us with $n - 1$ choices of degrees for n many different vertices. Hence, at least two vertices must have the same degree. $\square$

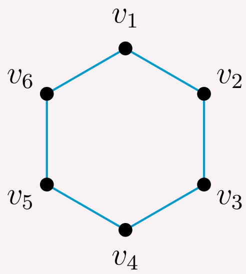
Remark 1.17. The above line of thought is called the **pigeon hole principle**. If there are n many pigeons wanting to fit into $n - 1$ many holes, then at least two of them have to cuddle up in the same hole.

Definition 1.18

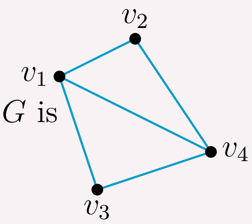
1) The **path** P_n is the graph on n vertices $v_1, \ldots, v_n$ with the edge set $E(P_n) = \{v_i v_{i+1} \mid 1 \leq i < n\}$, i.e. P_n is represented by the diagram

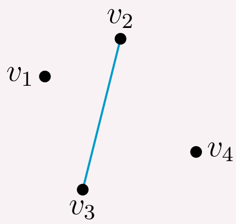


2) The **cycle** C_n is the graph on n vertices with edge set $E(C_n) = \{v_i v_{i+1} \mid 1 \leq i < n\} \cup \{v_n v_1\}$. E.g.

C_6 is 

3) Let $G = (V, E)$ be an arbitrary graph. The **complement** $\overline{G}$ of G is the graph $\overline{G} = (V, \overline{E})$, where $\overline{E} = \{uv \mid u, v \in V, uv \notin E\}$.

e.g. If G is 

, then $\overline{G}$ is 

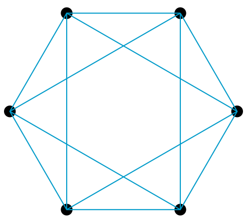
Definition 1.19

We call a graph G **regular** if any of its vertices has the same degree. If this degree is r, we say that G is r**-regular**.

Remark 1.20.

1. A graph G is regular iff $\delta(G) = \Delta(G)$.
2. K_n is $(n-1)$-regular and E_n is 0-regular.
3. An r-regular graph of order n has $\frac{1}{2}nr$ many edges.

Example 1.21. The graph below is 4-regular of order 6.



2. Subgraphs

There are two ways in which one graph can be part of another graph.

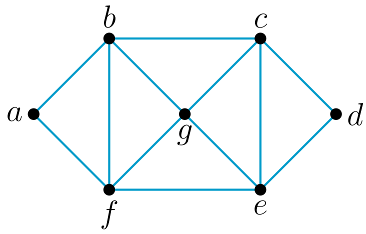
Definition 1.22

1. A graph H is called a **subgraph** of some graph G, written $H \subseteq G$, if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. We also say G **contains** H.
2. If $H \subseteq G$, we say that H is an **induced subgraph** of G, written $H \sqsubseteq G$, if $E(H) = \{uv \in E(G) \mid u, v \in V(H)\}$.

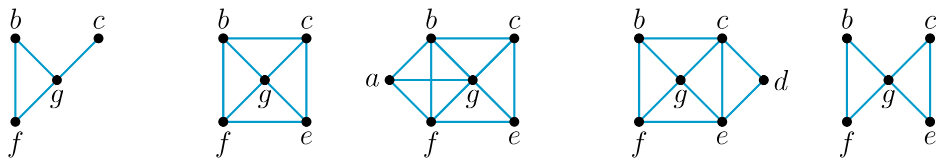
Remark 1.23.

1. $H \subseteq G$ is induced if for any two vertices in H we have: If they are adjacent in G, then they are adjacent in H.
2. Every induced subgraph is a subgraph but not vice versa.
3. If G is a graph and $S \subseteq V(G)$, then there is only one induced subgraph $H \sqsubseteq G$ with vertex set S, i.e. $V(H) = S$. We denote this graph by $\langle S \rangle$ and call it the subgraph of G induced by S.

Example 1.24. Consider G given as



Then



subgraph	✓	✓	✗	✓	✓
induced	✗	✓	✗	✓	✗

3. Walks in Graphs

Definition 1.25

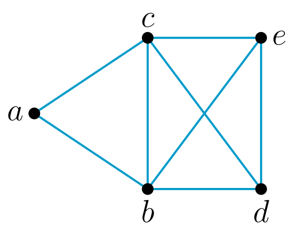
A (v_0, v_k)-**walk** in a graph is a sequence of vertices $(v_0, v_1, \dots, v_k)$ s.t. any two consecutive vertices v_i and v_{i+1} are adjacent. We call the edges $\{v_0v_1, v_1v_2, \dots, v_{k-1}v_k\}$ the **edges of the walk**. We say that the walk is **closed** if $v_0 = v_k$. The **length** of a walk is the number of edges in it (counting repetition).

Definition 1.26

We distinguish the following types of walks:

- A **trail** is a walk whose edges are pairwise distinct.
- A **circuit** is a closed walk whose edges are pairwise distinct.
- A **path** is a walk whose vertices are distinct.
- A **cycle** is a closed walk $(v_0, \dots, v_k = v_0)$ with $k \geq 3$ and whose vertices $v_0, \dots, v_{k-1}$ are pairwise distinct.

Example 1.27. Consider G via



Give examples for a:

- **walk** (d, b, c, d, b, a)
  da-walk, length 5
- **trail** (d, c, a, b, c, e)
  de-trail, length 5
- **path** (d, c, a, b, e)
  de-path, length 4
- **closed walk** (e, b, c, a, b, d, e)
  e-closed walk, length 6
- **circuit** (d, c, a, b, c, e, d)
  d-circuit, length 6
- **cycle** (d, c, a, b, e, d)
  d-cycle, length 5

Lemma 1.28

If $\delta(G) \geq 2$, then G contains a cycle as a subgraph.

Proof. Let $P = (v_0, \dots, v_k)$ be a path in G of maximal length. This exists, as G is finite. Further, as $\delta(G) \geq 2$, we get $k \geq 2$. As $\deg(v_0) \geq \delta(G) \geq 2$, v_0 has at least two neighbors. One of them is v_1. Let us denote the other one by u. If $u \neq v_i$ for all $1 \leq i \leq k$, then $\tilde{P} = (u, v_0, v_1, \dots, v_k)$ is still a path and of greater length than P, contradicting our assumptions. Hence, $u = v_i$ for some $1 \leq i \leq k$. But then the sequence $(v_0, v_1, \dots, v_i = u, v_0)$ is the desired cycle subgraph of G. $\square$

Corollary 1.29 Contrapositive

If G does not contain any cycles, then $\delta(G) \leq 1$.

Theorem 1.30

Every uv-walk in a graph contains a uv-path.

Proof. We proceed by strong induction on the length $n \geq 1$ of the walk. **I.B. n=1.** If the uv-walk is of length one, then it is exactly (u, v), which is also a path. **I.S.** Assume every uv-walk of length at most $n \geq 1$ contains a uv-path (I.H.). Assume there is a uv-walk $W = (u = w_0, w_1, \dots, w_n, w_{n+1} = v)$ of length $n + 1$. If W is already a path, we are done. Otherwise there are i, j s.t. $0 \leq i < j \leq n + 1$ and $w_i = w_j$. But then the walk $\tilde{W}$ which arises from W by deleting the vertices $w_{i+1}, \dots, w_{j-1}, w_j$, i.e. $\tilde{W} = (u = w_0, \dots, w_i, w_{j+1}, \dots, w_{n+1} = v)$ is still a uv-walk, but of length at most n. Using I.H., we know that $\tilde{W}$ contains a uv-path, whence also W contains (the same) uv-path. □

4. Connectivity

Definition 1.31

A graph is **connected** if there exists an uv-path in G for any vertices $u, v \in V(G)$. Otherwise, it is called **disconnected**.

Intuition

A graph is connected if you could pick it up entirely by just lifting one vertex. If it is not connected, then the subgraph you lift that way is called a connected component.

Definition 1.32

A **connected component** of G is a maximal connected induced subgraph of G. i.e. $C \sqsubseteq G$ is a connected component iff (i) C is connected and (ii) for any $v \in V(G) \setminus V(C)$ the induced subgraph on $V(C) \cup \{v\}$ is **not** connected.

Remark 1.33. G is connected iff it has exactly one connected component.

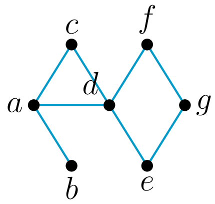
Even among connected graphs, there are different levels of being connected. E.g. the graph K_5  "feels" more connected than the graph . In order to properly describe this intuition, we need more notations.

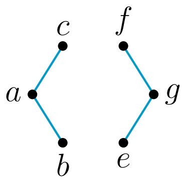
Definition 1.34 Vertex and Edge Deletion

Let G be a graph, $S \subseteq V_G$ and $T \subseteq E_G$.

1. By $G - S$ we denote the graph arising from G by removing from V_G all vertices in S and their incident edges.
2. If $S = \{v\}$, we write $G - v$.
3. By $G - T$ we denote the graph arising from G by removing only the edges in T, but no vertices.
4. If $T = \{e\}$, we write $G - e$.

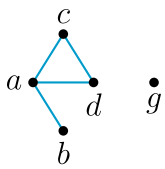
Example 1.35. Consider G as given below. Note that G only has one connected component.

$G =$ 

Then $G - d$ is 

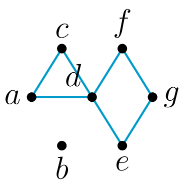
and has 2 connected components.

The vertex d is called a **cut vertex**.

Further, $G - \{e, f\}$ is 

. It also has 2 connected components.

The set $\{e, f\}$ is called a **cut set**.

Further, $G - ab$ is 

Again, it has 2 connected components.

We call the edge ab a **bridge**.

Definition 1.36

Let G be a graph.

1. We call $v \in V_G$ a **cut vertex** if $G - v$ has more connected components than G itself.
2. We call $e \in E_G$ a **bridge** if $G - e$ has more connected components than G itself.
3. We call $S \subseteq V_G$ a **cut set** if $G - S$ is disconnected.
4. A connected graph which does not contain any cut vertices is called **non-separable**.

1.37 Observation

1. If G is connected then v is a cut vertex of G iff $\{v\}$ is a cut set.
2. The vertex v is a cut vertex iff there are vertices u and w, different from v s.t. every uw-path uses v.
3. A graph has no cut sets iff it is a complete graph.

Definition 1.38

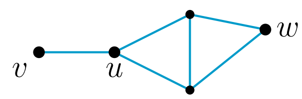
For a non-complete graph G, we define its **connectivity** $\kappa(G)$ as the minimal size of a cut set. For K_n, we set $\kappa(K_n) = n - 1$.

Lemma 1.39

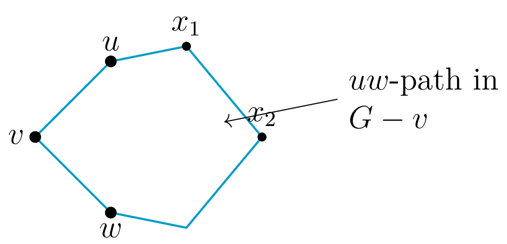
If G is a nonseparable graph of order at least 3, then $\delta(G) \geq 2$ and every vertex of G is contained in a cycle.

Proof. Consider G nonseparable with $|G| \geq 3$. By definition, G is connected, i.e. $\delta(G) \geq 1$.

First we show that $\delta(G) \geq 2$. Otherwise, we have $\delta(G) = 1$, i.e. there is some vertex v s.t. $\deg(v) = 1$. Let u be the unique neighbor of v and w any other vertex of G (which exists as $|G| \geq 3$). Then clearly any vw-path must use the unique neighbor u of v, whence u is a cut vertex. This contradicts the fact that G is inseparable. Hence, $\delta(G) \geq 2$, as desired.

Picture: 

Now, consider $v \in V_G$ arbitrary. We want to show that v is contained in a cycle in G. As $\delta(G) \geq 2$, v has at least 2 neighbors, say u and w. As G is nonseparable, $G - v$ is still connected. In particular, there is a uw-path $(u = x_0, x_1, x_2, \dots, x_k = w)$ in $G - v$. But then the walk $(x_0 = u, x_1, \dots, x_k = w, v, x_0 = u)$ is the desired cycle containing v.

Picture: 

$\square$

Definition 1.40

We say that G is **k-connected** if $\kappa(G) \geq k$, i.e. if G is connected and $G - S$ is still connected for any $S \subseteq V_G$ with $|S| < k$.

Lemma 1.41

The following hold:
1) G is connected iff $\kappa(G) \geq 1$.
2) G is 1-connected iff G is connected.
3) G is 2-connected iff G is connected and has no cut vertices.
4) G is 2-connected iff G is non-separable.
5) If G is 2-connected, then it contains at least one cycle (for $|G| \geq 3$).
6) If G is k-connected, then G is j-connected for all $j \leq k$.
7) $|G| > \kappa(G)$.
8) $\kappa(G) \leq \delta(G)$.

Proof. 1)–6) are easy observations – verify them by yourselves.

7) If $G = K_n$, then $|G| = n > n - 1 = \kappa(G)$. Otherwise, assume $\kappa(G) = k$, i.e. ex. $S \subseteq V_G$ s.t. $|S| = k$ and $G - S$ is disconnected. For $G - S$ to be disconnected, it must contain at least 2 vertices, whence

$$|G| \geq |S| + 2 = \kappa(G) + 2 > \kappa(G).$$

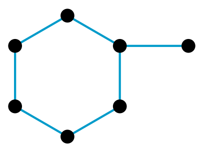
8) Assume $\kappa(G) > \delta(G)$ and let $v \in V_G$ s.t. $\deg(v) = \delta(G)$. Note that $|G| > \kappa(G) > \delta(G) = |N(v)|$, whence $G - N(v)$ contains at least one vertex besides v. But clearly, $G - N(v)$ is disconnected (as $\deg^{G-N(v)}(v) = 0$). Hence, $N(v)$ is a cut set and $\kappa(G) \leq |N(v)| = \delta(G)$, contradicting the assumptions. □

5. Bipartite Graphs

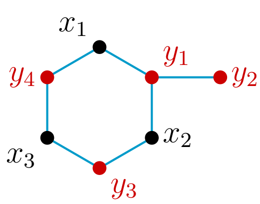
Definition 1.42

A graph G is called **bipartite** if we can partition the vertex set V_G into two disjoint sets $V_G = X \cup Y$ s.t. every edge of G has one end vertex in X and the other in Y.

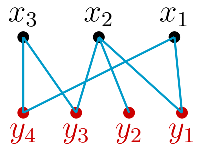
Example 1.43. Consider $G :=$



We can partition the vertices of G into two sets via $X = \{x_1, x_2, x_3\}$ and $Y = \{y_1, y_2, y_3, y_4\}$.



Rearranging the position of the vertices makes it clear that G is bipartite:



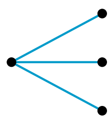
We see that there are no edges between any two vertices in X or in Y.

Remark 1.44. A graph G is bipartite if and only if we can color the vertices of G with two colors s.t. the end vertices of each edge have different colors.

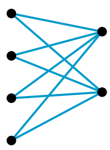
Definition 1.45

Let $m, n \in \mathbb{Z}_+$. The **complete bipartite graph** $K_{m,n}$ is the bipartite graph with $X = \{x_1, \dots, x_m\}$, $Y = \{y_1, \dots, y_n\}$, $V_G = X \cup Y$ and $E_G = \{xy \mid x \in X, y \in Y\}$.

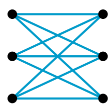
Example 1.46. Below are some examples of complete bipartite graphs.



$K_{1,3}$



$K_{4,2}$



$K_{3,3}$

The following theorem helps us decide whether or not a given graph is bipartite.

Theorem 1.47

A graph is bipartite iff it does not contain odd cycles.

Proof. "$\Rightarrow$": Assume G is bipartite and nevertheless there is a cycle of odd length, say $(x_0, x_1, \dots, x_{2k}, x_{2k+1} = x_0)$. By Remark 1.44, we can color V_G in two colors, $C1$ and $C2$, s.t. adjacent vertices have different colors. Then, if x_0 has color $C1$, x_1 has color $C2$ whence x_2 has color $C1$. That way we see that the color of x_i is

$$\begin{cases} C1 \text{ if } i \text{ is even} \\ C2 \text{ if } i \text{ is odd} \end{cases}.$$

Following that logic, the vertex $x_0 = x_{2k+1}$ should have color $C1$ and color $C2$ at the same time, which is a contradiction.

"$\Leftarrow$": Now consider that G does not contain odd cycles. We will show that G is bipartite by providing a partition. We may assume that G is connected as otherwise we work component per component. Pick $v \in V_G$ arbitrary and define

$$X = \{w \in V_G \mid \text{the shortest } vw \text{ path has even length}\} \text{ and}$$

$$Y = \{w \in V_G \mid \text{the shortest } vw \text{ path has odd length}\}.$$

Clearly, X and Y are disjoint. We will show that there are no adjacent vertices in X or Y respectively. Note that $v \in X$.

Aiming for a contradiction, assume that there are vertices $w_1, w_2 \in X$ which are adjacent. Clearly, $w_1 \neq v$, as otherwise the shortest vw_2-path was exactly vw_2 of length 1. Similarly, $w_2 \neq v$. Let $P_1 = (v = x_0, x_1, \ldots, x_{2k} = w_1)$ and $P_2 = (v = y_0, y_1, \ldots, y_{2\ell} = w_2)$ be the shortest vw_1- and vw_2-paths. Suppose that $x_i = y_j$ for some $0 < i \leq 2k$ and $0 < j \leq 2\ell$. If $i < j$, then $(v = x_0, x_1, \ldots, x_i, y_{j+1}, \ldots, y_{2\ell} = w_2)$ is a vw_2 path shorter than P_2, a contradiction. Similarly, $j < i$ is impossible, whence $i = j$, whenever $x_i = y_j$.

Now, pick the largest i s.t. $x_i = y_i$. As $x_0 = v = y_0$, such an i always exists. Then we obtain the following cycle

$$C = (\underbrace{x_i, x_{i+1}, \ldots, x_{2k}}_{2k-i} = w_1, \underbrace{w_2}_{1} = y_{2\ell}, \underbrace{y_{2\ell-1}, \ldots, y_{i+1}}_{2\ell-i}, y_i = x_i).$$

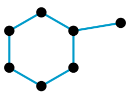
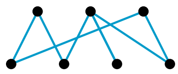
This is a cycle, as P_1 and P_2 were paths and i was maximal s.t. $x_i = y_i$. Further, the length of C is odd, as it equals

$$(2k - i) + 1 + (2\ell - i) = 2(k + \ell - i) + 1.$$

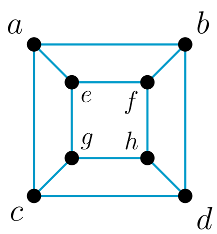
This contradicts our assumption that G does not contain odd cycles. We hence proved that no two vertices w_1 and w_2 from X can be adjacent. The arguments for $v_1, v_2 \in Y$ is analogous (try to write it down). This concludes the proof. $\square$

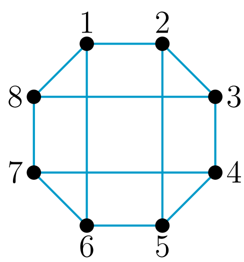
6. Graph Isomorphisms

In Example 1.43, we rearranged the given graph G_1 as G_2.

$G_1 =$  as $G_2 =$ 

We understand G_1 and G_2 as the same, even though on the first glance, they look very similar. Another example is given by

$G =$ 

$H =$ 

We can relabel the vertices of G via $a \mapsto 1, b \mapsto 2, c \mapsto 8, d \mapsto 3, e \mapsto 7, f \mapsto 4, g \mapsto 6$ and $h \mapsto 5$ and obtain H. The aim of this section is to formalise this concept.

Definition 1.48

We say that a graph G is **isomorphic** to a graph H if there exists a bijection $\varphi : V_G \to V_H$ s.t. for any $u, v \in V_G$ we have that $\{u, v\} \in E_G$ if and only if $\{\varphi(u), \varphi(v)\} \in E_H$. Then, the map φ is called an **isomorphism** and we write $G \cong H$.

Remark 1.49. Let $G \cong H$ via $\varphi : V_G \to V_H$. Then:

1) $|V_G| = |V_H|$ and $|E_G| = |E_H|$ and $\overline{G} \cong \overline{H}$.
2) The degree sequence of G equals the degree sequence of H.
3) G is connected iff H is connected.
4) $\deg_G(v) = \deg_H(\varphi(v))$ for all $v \in V_G$.

Chapter 2

Distance in Graphs

1. introduction

We have a natural understanding of the "distance" between two objects in our physical space. But there are many other ways of defining distances. E.g., the distance between people could be the positive difference of their birth years or the number of acquaintances you need to connect one to the other.

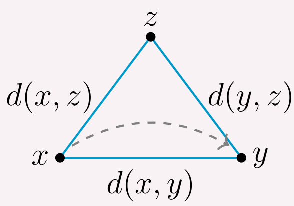
In this chapter we will introduce a notion of distance of vertices in a graph. But first let us note what are the characterising properties that make us call all these concepts "distances".

Definition 2.1

Let X be any set. We call a function $d : X \times X \to \mathbb{R}^{\geq 0} \cup \{\infty\}$ a **metric** if it satisfies for all $x, y, z \in X$:

1) $d(x, y) \geq 0$
2) $d(x, y) = 0$ iff $x = y$
3) $d(x, y) = d(y, x)$
4) $d(x, z) \leq d(x, y) + d(y, z)$ (**Triangle Inequality**)

We then call the pair (X, d) a **metric space**.



Example 2.2. Consider $X = \mathbb{R}$ and $d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^{\geq 0}$ via $d(x, y) := |x - y|$. Then $(\mathbb{R}, d)$ is a metric space.

Now we are ready to define a metric on an arbitrary graph.

Definition 2.3

Let G be any graph and $u, v \in V_G$. We define the **distance** $d(u, v)$ between u and v as the length of the shortest uv-path in G, i.e.

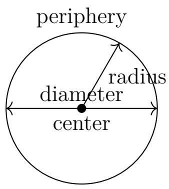
$$d(u, v) := \min\{\text{length}(P) \mid P \text{ is a } uv\text{-path}\}.$$

If there is no such path, we set $d(u, v) := \infty$.
2) If $d(u, v) = k$, then any uv-path of length k is called a **geodesic**.

Remark 2.4.

1) We may write $d_G(u, v)$ to emphasize that we consider the distance in G.
2) While in $(\mathbb{R}, d)$ geodesics are unique, in general this is not the case. Consider for example two opposite poles on a sphere.
3) $d(x, y) = \infty$ iff x and y are in different connected components.
4) (V_G, d) is a metric space for any connected graph G.

We call something eccentric if it is away from the usual. Similarly, in graphs we measure by eccentricity how far a vertex is from the center. Consider the following notions on a cycle:



Definition 2.5

1) The **eccentricity** $ecc(v)$ of a vertex v is its greatest distance to any other vertex, i.e. $ecc(v) = \max\{d(u, v) \mid u \in V_G\}$.
2) The **radius** $rad(G)$ is the smallest possible eccentricity and the **diameter** $diam(G)$ is the largest possible eccentricity.
3) The **center** $C(G)$ is the set $\{v \in V_G \mid ecc(v) = rad(G)\}$ and the **periphery** $P(G)$ is the set $\{v \in V_G \mid ecc(v) = diam(G)\}$.

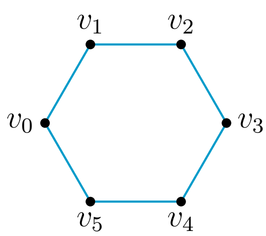
Example 2.6. 1) Consider P_5, the path of length 4, i.e.

Then

$$\begin{aligned}
d(v_1, v_i) &= i - 1, \text{ whence } ecc(v_1) = \max\{0, 1, 2, 3, 4\} = 4.\\
d(v_2, v_i) &= |i - 2|, \text{ whence } ecc(v_2) = \max\{1, 0, 1, 2, 3\} = 3.\\
d(v_3, v_i) &= |i - 3|, \text{ whence } ecc(v_3) = \max\{2, 1, 0, 1, 2\} = 2.\\
d(v_4, v_i) &= |i - 4|, \text{ whence } ecc(v_4) = \max\{3, 2, 1, 0, 1\} = 3.\\
d(v_5, v_i) &= |i - 5|, \text{ whence } ecc(v_5) = \max\{4, 3, 2, 1, 0\} = 4.
\end{aligned}$$

Hence $rad(P_5) = \min\{ecc(v) \mid v \in V\} = \min\{4, 3, 2, 3, 4\} = 2$.
Also $C(P_5) = \{v \in V \mid ecc(v) = rad(P_5)\} = \{v_3\}$.
Further $diam(P_5) = \max\{ecc(v) \mid v \in V\} = \max\{4, 3, 2, 3, 4\} = 4$.
And $P(P_5) = \{v \in V \mid ecc(v) = diam(P_5)\} = \{v_1, v_5\}$.

2) Consider $G := C_6$, the cycle of length 6, i.e. $G =$



Then

$$\begin{aligned}
d(v_0, v_i) &= 3 - |3 - i|, & ecc(v_0) &= \max\{0, 1, 2, 3, 2, 1\} = 3.\\
d(v_1, v_i) &= |3 - |4 - i||, & ecc(v_1) &= \max\{1, 0, 1, 2, 3, 2\} = 3.\\
d(v_2, v_i) &= |3 - |5 - i||, & ecc(v_2) &= \max\{2, 1, 0, 1, 2, 3\} = 3.\\
d(v_3, v_i) &= |3 - i|, & ecc(v_3) &= \max\{3, 2, 1, 0, 1, 2\} = 3.\\
d(v_4, v_i) &= |3 - |1 - i||, & ecc(v_4) &= \max\{2, 3, 2, 1, 0, 1\} = 3.\\
d(v_5, v_i) &= |3 - |2 - i||, & ecc(v_5) &= \max\{1, 2, 3, 2, 1, 0\} = 3.
\end{aligned}$$

Hence, $rad(G) = \min\{ecc(v) \mid v \in V_G\} = \min\{3, 3, 3, 3, 3, 3\} = 3$
whence $C(G) = \{v \in V_G \mid ecc(v) = rad(G)\} = V_G$.
Further, $diam(G) = \max\{ecc(v) \mid v \in V_G\} = \max\{3, 3, 3, 3, 3, 3\} = 3$.
and $P(G) = \{v \in V_G \mid ecc(v) = diam(G)\} = V_G$.

Lemma 2.7

For any graph G we have $rad(G) \le diam(G) \le 2rad(G)$.

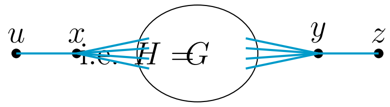
Proof. We have $rad(G) \le diam(G)$ by definition. For the other inequality, pick $v \in C(G)$ arbitrary and consider $u, w \in V_G$ arbitrary s.t. $d(u, w) = diam(G)$. Then

$$d(u, w) \le d(u, v) + d(v, w) \le ecc(v) + ecc(v) = 2rad(G). \qquad \square$$

Theorem 2.8

Every graph G is isomorphic to the graph induced by the center of another graph H, i.e. ex. H s.t. $G \cong \langle C(H) \rangle$.

Proof. Let G be arbitrary. We build a new graph H which contains G as an induced subgraph via: $V_H = V_G \cup \{u, x, y, z\}$, i.e. adding 4 new vertices to G. Further, let $E_H = E_G \cup \{ux, yz\} \cup \{xv, vy \mid v \in V_G\}$.



Now $ecc(v) = 2$ for any $v \in V_G$. Nevertheless, $d(u, z) = 4$ and $d(x, z) = d(y, u) = 3$, whence $ecc(w) > 2$ for all $w \in V_H \setminus V_G$. Thus, $rad(H) = 2$ and $C(H) = V_G$, whence $\langle C(H) \rangle \cong G$. □

Lemma 2.9

A graph G is isomorphic to the graph induced by the periphery of another graph H iff either every vertex has eccentricity 1 or no vertex does.

Proof. "⇒" We use proof by contraposition. Assume ex. $u \in V_G$ s.t. $ecc(u) = 1 < diam(G)$. In particular, $G \neq P(G)$. Now, aiming for a contradiction, assume ex. H s.t. $G \leq H$ and $P(H) = V_G$. As $G \neq P(G)$, we know that $H \neq G$ and $diam(H) \geq 2$. As $u \in V_G = P(H)$, there is some $w \in V_H$ s.t. $d(u, w) = diam(H)$. But then, $w \in P(H) \cong V_G$, and as $ecc(u) = 1$, we also get $d(u, w) = 1 < diam(H)$. Hence, $P(H)$ cannot be V_G.

"⇐" If all vertices in G have eccentricity 1 or 0, then G is complete and $G \cong P(G)$. For the second case, assume $rad(G) > 1$. And consider H s.t. $V_H = V_G \cup \{v\}$ contains one new vertex which is connected to everyone else, i.e. $E_H = E_G \cup \{vx \mid x \in V_G\}$. Then, as $ecc(x) \geq 2$ for all $x \in V_G$,

$$ecc_H(x) = \begin{cases} 2 & \text{if } x \in V_G \\ 1 & \text{if } x = v \end{cases}.$$

Hence, $diam(H) = 2$ and $\langle P(H) \rangle = G$, as desired. □

2. Adjacency Matrices

We saw the visual benefits of studying graphs by their diagram. This is very useful to illustrate ideas and study small graphs. In applications on the other hand, when studying e.g. correlations of weather phenomena or social links, graphs tend to have thousands of vertices. Here, it is no longer practical to use neither the set- nor the diagram representation of graphs. The way computers store and analyze graphs is by using adjacency matrices.

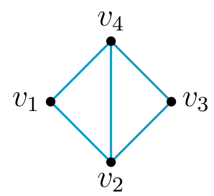
Definition 2.10

Let G be a graph of order n with vertices $V_G = \{v_1, v_2, \dots, v_n\}$. The **adjacency matrix** of G is the matrix $A_G = (a_{ij}) \in M_{n\times n}$ defined via

$$a_{ij} = \begin{cases} 1 & \text{if } v_iv_j \in E \\ 0 & \text{otherwise.} \end{cases}$$

We also write $A(i,j)$ for a_{ij}.

Example 2.11. Consider G given by



Then $A_G \in M_{4\times 4}$

$$A_G = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

is the adjacency matrix of G.

Remark 2.12. If $A_G = (a_{ij})$ is an adjacency matrix of a graph G, then

1) $a_{ii} = 0$ for all $1 \le i \le |G|$

2) A is symmetric.

3) $\sum_{j=1}^{|G|} a_{ij} = \deg(v_i)$ and thus $\sum_{i,j=1}^{|G|} a_{ij} = \sum_{i=1}^{|G|} \deg(v_i) = 2|E|$.

4) A_G is only unique up to reordering the vertices.

Example 2.13. Let revisit the graph G from 2.11. The fact that $A_G(2,3) \neq 0$ means that v_2 and v_3 are adjacent. And $A(1,3) = 0$ says that v_1 and v_3 are not. Now consider

$$A_G^2 = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 2 & 1 \\ 1 & 3 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 3 \end{pmatrix}.$$

Let's interpret the values of A_G^2. Now, $A_G^2(1,3) = 2$. How did we compute it? $A_G^2(1,3) = \sum_{j=1}^{4} a_{1j}a_{j3}$. Now $a_{1j}a_{j3} = 1$ iff v_1v_j and v_jv_3 are edges iff (v_1, v_j, v_3) is a walk of length 2 from v_1 to v_3. Hence, $A_G^2(1,3) = \sum a_{1j}a_{j3}$ is the number of walks from v_1 to v_3 of length 2. This generalises and provides a strong tool to study graphs.

Theorem 2.14

Let G be a graph with $V_G = \{v_1, \dots, v_n\}$ and A_G the corresponding adjacency matrix. Then the entry $A_G^k(i,j)$ is the number of possible walks from v_i to v_j of length k.

Proof. We proceed by induction on the power k. (Note that $k = 0$ works too). $\underline{k=1}$: We get that $A(i,j) = \begin{cases} 0 & \text{iff } v_iv_j \notin E_G \text{ iff there are } 0\ v_iv_j\text{-walks of length } 1 \\ 1 & \text{iff } v_iv_j \in E_G \text{ iff there is } 1\ v_iv_j\text{-walk of length } 1 \end{cases}$.

$\underline{k \to k+1}$: Assume that $A^k(i,j)$ gives exactly the number of v_iv_j-walks of length exactly k. Let's denote $A^k = (b_{ij})$ and $A = (a_{ij})$. Note that there is a v_iv_j-walk of length $k+1$ iff there ex. a vertex v_ℓ s.t. there is a v_iv_ℓ-walk of length k and an $v_\ell v_j$-walk of length one. Hence

$$\begin{aligned} |\{v_iv_j\text{-walk of length } k+1\}| &= \sum_{\ell \mid v_\ell \in N(v_j)} |\{v_iv_\ell\text{-walk of length } k\}| \\ &\overset{\text{I.H.}}{=} \sum_{\ell \mid v_\ell \in N(v_j)} b_{i\ell} = \sum_{\ell=1}^{n} b_{i\ell}a_{\ell j} \\ &= \sum_{\ell=1}^{n} A^k(i,\ell) \cdot A(\ell,j) = A^{k+1}(i,j). \end{aligned}$$

$\square$

Corollary 2.15

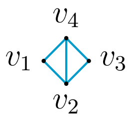
Let G be a graph with $V_G = \{v_1, \dots, v_n\}$ and A_G the adjacency matrix. Then $d(v_i,v_j) = \min\{k \mid A^k(i,j) \neq 0\}$. (Recall that $A_G^0 = I_n$).

Definition 2.16

Let G be a graph with adjacency matrix A. For every $k \in N$ we define the **Stoll matrix** S_k via

$$S_k = \sum_{i=0}^{k} A^k = I_n + A + A^2 + \dots + A^k.$$

Remark 2.17. As $S_k(i,j) = \sum_{i=0}^{k} A^i(i,j)$, we get that $S_k(i,j)$ is the number of v_iv_j-walks of length at most k.

Example 2.18. Recall the graph $G =$  with $A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$.

$A^2 = \begin{pmatrix} 2 & 1 & 2 & 1 \\ 1 & 3 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 3 \end{pmatrix}$ and $A^3 = \begin{pmatrix} 2 & 5 & 2 & 5 \\ 5 & 4 & 5 & 5 \\ 2 & 5 & 2 & 5 \\ 5 & 5 & 5 & 4 \end{pmatrix}$.

Then $S_0 = I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$, $S_1 = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$, $S_2 = \begin{pmatrix} 3 & 2 & 2 & 3 \\ 2 & 4 & 2 & 3 \\ 2 & 2 & 3 & 2 \\ 2 & 3 & 2 & 4 \end{pmatrix}$.

$S_3 = \begin{pmatrix} 5 & 7 & 4 & 8 \\ 7 & 8 & 7 & 8 \\ 4 & 7 & 5 & 7 \\ 7 & 8 & 7 & 8 \end{pmatrix}$. This means there are for example 4 v_1v_3 walks of length at most 3, namely (v_1, v_2, v_3), (v_1, v_4, v_3), (v_1, v_2, v_4, v_3) and (v_1, v_4, v_2, v_3).

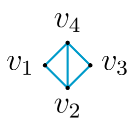
Theorem 2.19

Let G be a graph with $V_G = \{v_1, \ldots, v_n\}$, adjacency matrix A and Stoll matrices S_k. Then the following hold.

1) $d(v_i, v_j)$ is the least k s.t. $S_k(i, j) \neq 0$.
2) $ecc(v_i)$ is the least k s.t. the i-th row of S_k has no zero entries.
3) $rad(G)$ is the least k s.t. S_k contains at least one row without zero entries (or ∞ otherwise).
4) $diam(G)$ is the least k s.t. S_k does not contain any zero entries.
5) G is disconnected iff S_{n-1} contains a zero.

Definition 2.20

Let G be a graph with $V_G = \{v_1, \ldots, v_n\}$. The **distance matrix** of G is the matrix $D \in M_{n \times n}$ s.t. $D(i, j) = d(v_i, v_j)$.

Example 2.21. Back to our example $G =$  . Then the distance matrix D is

$$\begin{pmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}.$$

Example 2.22. **Erdős Number** Paul Erdős - Hungarian Mathematician, published over 1500 papers. Consider G with $V_G =$ all mathematicians, $E_G = \{xy \mid x \text{ and } y \text{ published together}\}$. Then $\deg(\text{Erdős}) > 500$ and the Erdős number of x is $d(\text{Erdős}, x)$.

Chapter 3

Trees

1. Introdction

The intuition for graph theoretic trees comes from actual trees in nature. Here, the stem splits into several branches that afterwards never rejoin.

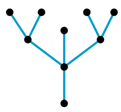
Definition 3.1

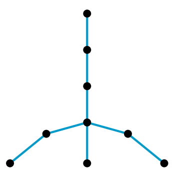
A graph which does not contain cycles is called **acyclic**. We call a graph G a **tree** if it is connected and acyclic. An arbitrary acyclic graph is called a **forest**. In a forest, any vertex of degree 1 is called a **leaf**.

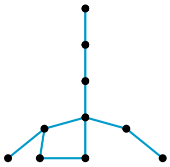
Remark 3.2.

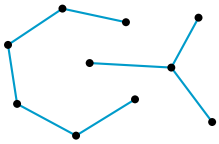
1) The graphs P_n, K_1, K_2 and $K_{1,n}$ are trees for any $n \in \mathbb{N}$.
2) Every tree is a forest.
3) Every connected component in a forest is a tree.
4) Every subgraph of a forest is a forest.

Example 3.3.

1)  **is a tree**.

2)  **is a tree**.

3)  **is not a tree**.

4)  **is not a tree**, but it **is a forest**.

Lemma 3.4

Any tree of order at least 2 has at least two leaves.

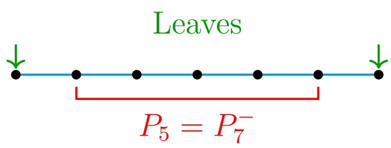
Proof. Let T be a tree with $|T| \geq 2$. In particular, T is connected. Consider a path of maximal length $P = (v_0, v_1, \ldots, v_n)$ in T. As $|T| \geq 2$, we know that $v_0 \neq v_n$. We claim that v_0 and v_n are leaves, i.e. $\deg(v_0) = \deg(v_n) = 1$. We execute the argument for v_0. As usual, we know that $N(v_0) \subseteq \{v_1, v_2, \ldots, v_n\}$. Let $u \in N(v_0)$ arbitrary, i.e. $u = v_i$ for some $i \geq 1$. But then $(v_0, v_1, \ldots, v_i, v_0)$ is a closed walk which is a cycle for all $i \geq 2$. As T does not contain cycles, we conclude that $i = 1$ and v_1 is the only neighbour of v_0. Hence $\deg(v_0) = 1$ and v_0 is a leaf. The argument for v_n is analogous. $\square$

Definition 3.5 Tree Pruning

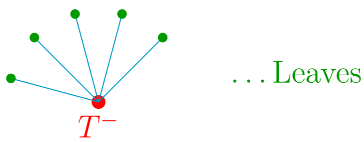
Let T be a tree of order at least 3. We denote by T^- the induced subgraph of T obtained by deleting all leaves of T.

Example 3.6.

1) If $T = P_7$ the path of length 6, then $T^- = P_7^- = P_5$ is the path of length 4:



2) If $T = K_{1,n}$ the complete bipartite graph, then $T^- = K_1 = E_1$ consists of one vertex only:



Lemma 3.7

Any tree of order n has exactly $n - 1$ edges.

Proof. We proceed by induction on $|T|$. If $|T| = 1$, then $T = K_1$, which has zero edges and the claim holds. Now assume we know that any tree of order n has exactly $n-1$ many edges and consider T of order $n+1$ arbitrary. By Lemma 3.4, T has a leaf u. Clearly, $T - u$ is still connected and of order n, whence $T - u$ has exactly $n - 1$ edges. As u was a leaf in T, T has exactly one edge more than $T - u$, whence

$$\|T\| = n = (n+1) - 1, \text{ as desired.}$$

□

Corollary 3.8

A forest of order n, consisting of k-many connected components, has exactly $n - k$ many edges.

We will now see that given a graph G is connected, Lemma 3.7 is not only a necessary, but even a sufficient condition for G to be a tree.

Theorem 3.9

A graph G of order n is a tree iff it is connected and has exactly $n - 1$ many edges.

Proof. "$\Rightarrow$" Clear by definition of a tree and Lemma 3.7.

"$\Leftarrow$" Assume G is connected of order n and contains exactly $n - 1$ many edges. If G contains a cycle, take any edge e_1 within the cycle and consider $G - e_1$. Then $G - e_1$ is still connected and of order n. If $G - e_1$ still contains a cycle, we proceed likewise and after $k \leq n-1$ many steps we obtain a graph $G - \{e_1, e_2, \ldots, e_k\}$ which is of order n, connected and without cycles, whence it is a tree. But $G - \{e_1, \ldots, e_k\}$ has $(n-1) - k < n - 1$ many edges, contradicting Lemma 3.7. □

Theorem 3.10

A graph of order n is a tree iff it is acyclic and has $n - 1$ many edges.

Proof. "$\Rightarrow$" Clear.

"$\Leftarrow$" Assume G is of order n with $n - 1$ many edges and acyclic, i.e. G is a forest. But by Corollary 3.8, if G has k-many connected components then $\|G\| = n - k = n - 1$, whence $k = 1$ and G is connected and hence a tree. □

Corollary 3.11 Summary

Let G be a graph of order n. Then TFAE:

1) G is connected and acyclic (i.e. a tree).
2) G is connected and has $n - 1$ many edges.
3) G is acyclic and has $n - 1$ many edges.

3.12 Homework

Every edge in a tree is a bridge.

Lemma 3.13

For any two vertices $u, v \in V_T$ in a tree T, there is a unique uv-path.

Proof. As T is connected, there clearly is a uv-path for any $u, v \in V_T$. Now assume that $P_1 = (u = x_0, x_1, \ldots, x_k = v)$ and $P_2 = (u = y_0, y_1, \ldots, y_\ell = v)$ are two distinct uv-paths. Then $P_1 \cup P_2$ is again a tree. Let i be minimal s.t. $x_i \neq y_i$. Then $(P_1 \cup P_2) - y_i y_{i-1}$ is still connected, contradicting the fact that every edge in a tree is a bridge. $\square$

Corollary 3.14

Let T be a tree and $v \in V_T$. Then $ecc(v)$ is the length of the longest path starting from v.

Lemma 3.15

Let T be a tree of order at least 2. Consider $u, v \in V_T$ s.t. $ecc(v) = d(u, v)$. Then u is a leaf.

Proof. Let $P = (v = x_0, x_1, \ldots, x_k = u)$ be the unique vu path. If u were not a leaf, then it had at least one neighbour $w \notin P$. But then $(v = x_0, x_1, \ldots, x_k, w)$ would be a path starting in v and longer than P, contradicting Corollary 3.14. $\square$

Lemma 3.16

Let T be a tree of order at least 3. Then $C(T) = C(T^-)$.

Proof.

1) Show that $C(T) \subseteq T^-$, i.e. $C(T)$ contains no leaf. To this end, let u be a leaf and v its unique neighbour. As $|T| \geq 3$, v is not a leaf itself and $d(u, w) = d(v, w) + 1$ for any $w \in V_T \setminus \{u\}$, whence $ecc(u) > ecc(v)$ and hence $u \notin C(T)$.

2) Show that $ecc_{T^-}(v) = ecc_T(v) - 1$ for every non-leaf $v \in V_T$. To that end, consider an arbitrary non-leaf $v \in V_T$ and pick $u \in V_T$ s.t. $d(v, u) = ecc(v)$. By 3.15, u is a leaf. Let P be the unique vu-path in T and note that u is the only leaf on P. Hence only u will be deleted from P in T^-. As this holds for all paths in T starting in v of length $ecc(v)$, we obtain that $ecc_{T^-}(v) = ecc_T(v) - 1$, as desired.

3) We conclude from 1) + 2) that for any vertex $v \in T^-$, $ecc_{T^-}(v) = ecc_T(v) - 1$, whence $v \in C(T)$ iff $v \in C(T^-)$ (and $rad(T^-) = rad(T) - 1$).

$\square$

Lemma 3.17

Let T be a tree. Then $C(T)$ is either K_1 or K_2.

Proof. We do induction on $|T|$. If $|T| = 1$, then $T = K_1$ is its own center and we are done. Similarly for $|T| = 2$, where $T = K_2$. Now assume that the claim holds for all trees of order $n \geq 3$ and consider a tree T with $|T| = n + 1$ arbitrary. By 3.16, we know that $C(T) = C(T^-)$. By 3.4 we know that T contains at least two leaves, whence $|T^-| \leq |T| - 2 < n$. Hence, by I.H., $C(T) = C(T^-)$ is either K_2 or K_1 as desired. $\square$

Lemma 3.18

Let T be a tree of order n and G an arbitrary graph s.t. $\delta(G) \geq n - 1$. Then G contains T as a subgraph.

Proof. We use induction on $|T|$. If $|T| = 1$, then $T = K_1$ is a subgraph of any graph G. Now assume we proved the claim for all trees of order at most n. Consider T with $|T| = n+1$ and G with $\delta(G) \geq n$ arbitrary. Let u be a leaf of T and denote by $T' := T - u$. Then $|T'| = n$, whence T' can be seen as a subgraph of G. Let v be the unique neighbour of u in T. Then $\deg_G(v) \geq \delta(G) \geq n$, but as $|T'| = n$ and v cannot be its own neighbour, there exist some $u' \in G$ adjacent to v and not contained in T'. Hence, the subgraph $(V_{T'} \cup \{u'\}, E_{T'} \cup \{vu'\})$ is the desired subgraph of G isomorphic to T. $\square$

Summary

1) A tree of order n contains exactly $n - 1$ edges.
2) Any tree of order at least two contains at least two leaves.
3) A graph of order n is a tree iff it is connected of size $n - 1$.
4) A graph of order n is a tree iff it is acyclic and of size $n - 1$.
5) A graph is a tree iff for any vertices u, v there is a unique uv-path.
6) The centre of any tree is either K_1 or K_2.
7) Any graph G contains any tree of order at most $\delta(G) + 1$ as a subgraph.

2. Spanning Trees

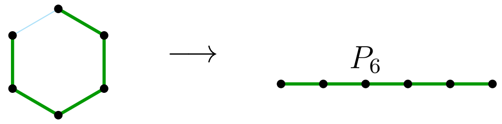
Definition 3.19

Let G be any graph. We call a subgraph $T \subseteq G$ a **spanning tree** for G if it is a tree and contains all vertices of G.

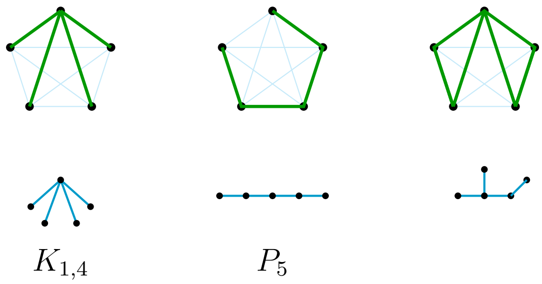
Remark 3.20. From the previous chapter it is clear that a spanning tree of a graph G of order n has n many vertices and $n - 1$ many edges.

Example 3.21. Consider the following graphs and spanning trees.

1) $G = C_6$, a possible spanning tree:



2) $G = K_5$, possible spanning trees:



Lemma 3.22

Every connected graph contains at least one spanning tree.

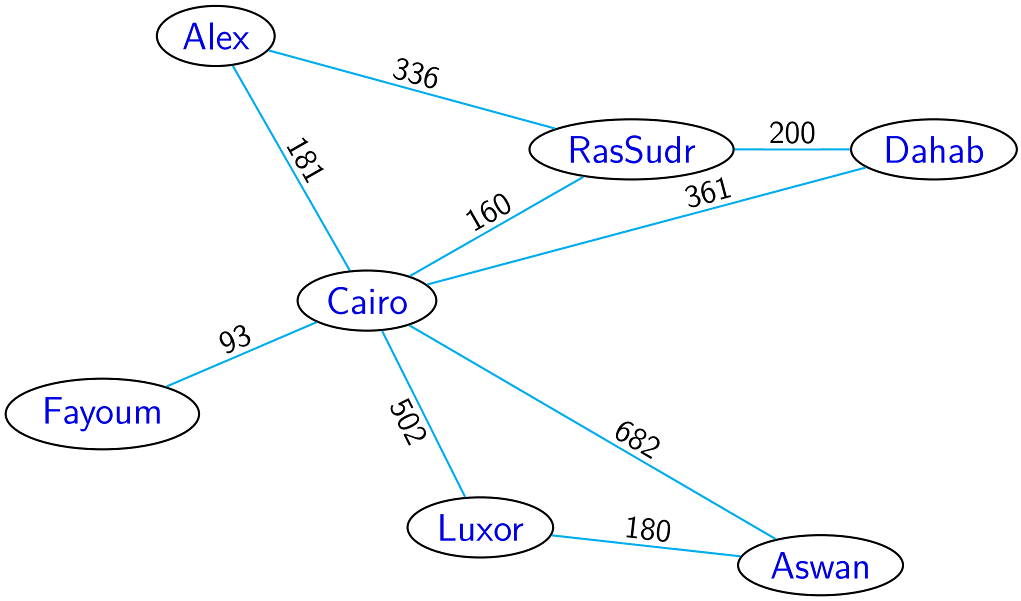
Proof. Assume G is connected and let T be a subgraph of G of maximal order s.t. T is a tree. We need to show that $V_T = V_G$. Otherwise, as G is connected, there is some vertex $u \in V_G \setminus V_T$ which is adjacent to some vertex $v \in V_T$. Now, consider the new subgraph $\hat{T} = (V_T \cup \{u\}, E_T \cup \{uv\})$. As $\deg_{\hat{T}}(u) = 1$, u is not contained in any cycles in $\hat{T}$, whence $\hat{T}$ is still a tree. As this contradicts maximality of $|T|$, we conclude that T must contain all vertices of G, whence it is a spanning tree for G. $\square$

Definition 3.23

A function $w : E_G \to \mathbb{R}$ is called a **weight function** on G. A graph G together with a weight function (i.e. the triple (V_G, E_G, w)) is called a **weighted graph**.

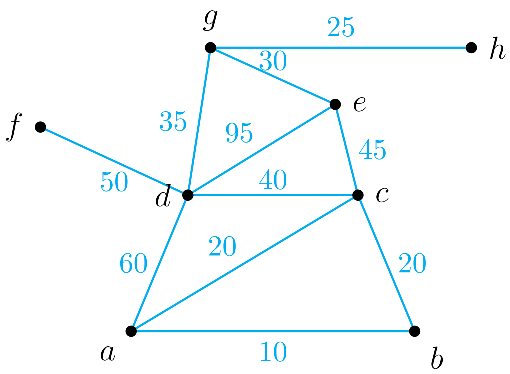
Example 3.24. *Visualisation*

We visualise the weighting of a graph by denoting the weight $w(e)$ on top of the edge e, e.g.

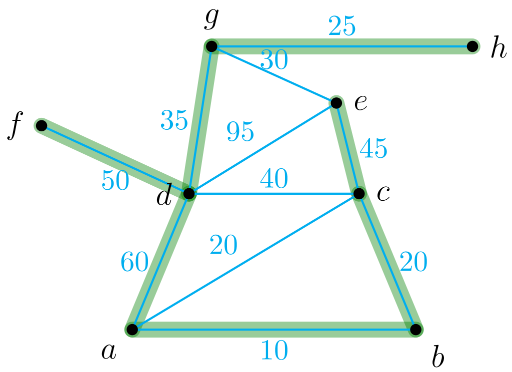


Here the weight function of an edge $e = uv$ is given by the (birds eye) distance between u and v.

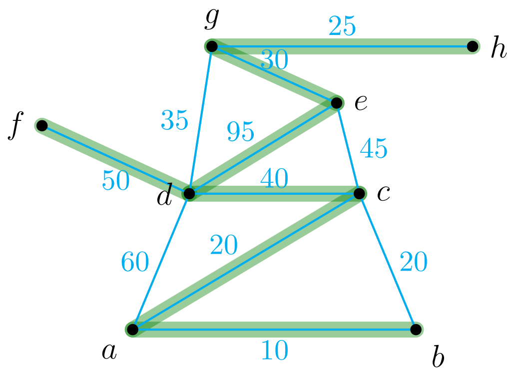
Example 3.25. Consider the following weighted graph.



We can find several spanning trees. Let's name some and compute their weight.



Total weight:
$45 + 20 + 10 + 60 + 50 + 35 + 25$
$= 245$



Total weight:
$10 + 20 + 40 + 50 + 95 + 30 + 25$
$= 270$

Definition 3.26

Let (G, w) be a connected weighted tree. A **minimum-weight spanning tree** T is a spanning tree of G s.t. the sum of the weights of its edges is minimal among all possible spanning trees of G, i.e. if $\tilde{T}$ is another spanning tree, then $\sum_{e \in E_T} w(e) \leq \sum_{e \in E_{\tilde{T}}} w(e)$.

Now how can we find a minimal spanning tree effectively? Consider the following algorithm:

3.27 Kruskal's Algorithm (1956)

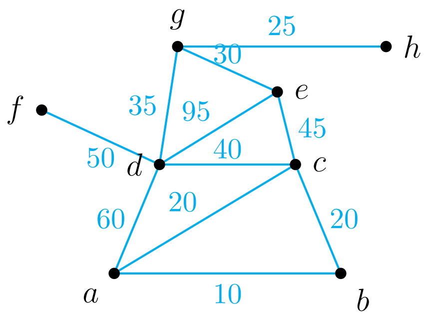
Consider the set of vertices as a forest $F = (V_G, \emptyset)$ where each vertex is a maximal subtree of F. Let $E := E_G$.

While (F is not a tree $\wedge$ $E \neq \emptyset$)

- Pick $e \in E$ of minimal weight. Let $E := E \setminus \{e\}$.
- If e connects two trees in F, let $E_F = E_F \cup \{e\}$.
- (i.e. $F + e$ is still acyclic)

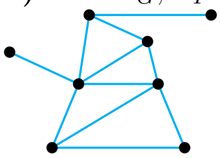
This algorithm stops after at most $|E_G|$ many repetitions.

Example 3.28. We apply the algorithm on the following weighted graph:

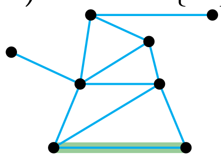


Let us mark edges we add to F green and the ones we disregard, red.

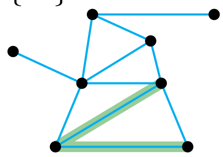
1) $E = E_G, E_F = \emptyset$



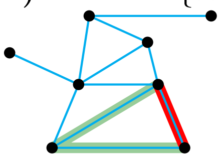
2) $E = E - \{ab\}, E_F = \{ab\}$



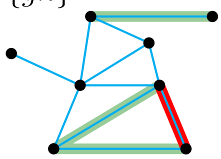
3) $E = E - \{ac\}, E_F = E_F \cup \{ac\}$



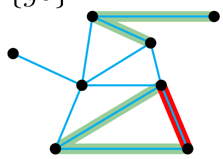
4) $E = E - \{bc\}, E_F = E_F$



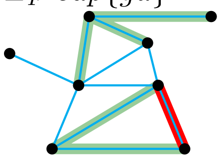
5) $E = E - \{gh\}, E_F = E_F \cup \{gh\}$



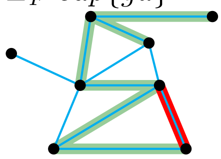
6) $E = E - \{ge\}, E_F = E_F \cup \{ge\}$



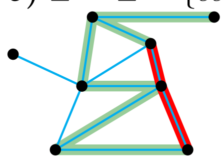
7) $E = E - \{gd\}, E_F = E_F\ cup\{gd\}$



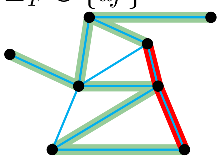
8) $E = E - \{gd\}, E_F = E_F\ cup\{gd\}$



9) $E = E - \{ce\}, E_F = E_F$



10) $E = E - \{df\}, E_F = E_F \cup \{df\}$



Here the algorithm stops, as $F = (V_G, E_F)$ with $E_F = \{ab, ac, gh, eg, dg, cd, df\}$ is a single tree whence the conditions in the while loop are violated.
(The output is the spanning tree F. Note that the second condition in the while loop was still valid, as $E = \{ad, de\} \neq \emptyset$).

Theorem 3.29

Kruskal's algorithm is correct, i.e. it always terminates and its output is a minimum-weight spanning tree.

Proof. 1) Termination: As after $|E_G|$-many steps the condition $E \neq \emptyset$ is violated, the algorithm always terminates.

2) The output F is a spanning tree: As $V_F = V_G$, it clearly contains all vertices of G. Further, in each step the regarded edge e either connects two disconnected trees into one larger tree, or, if it would connect two vertices of the same subtree in F, is disregarded. Hence after each step, F is still a forest, i.e. acyclic. It remains to show that F is connected. If the algorithm stops because F is a tree, then it is clearly connected. If it stops because we went through all the edges, then any edge of G

not contained in F would connect two vertices of the same connected component. Thus F has as many connected components as G, which is one, as G is connected.

3) F is a minimum-weight spanning tree. Aiming for a contradiction, assume this is not the case. Let $\{e_1, \dots, e_{n-1}\}$ be all the edges in F, enumerated in the order they were added to F by the algorithm. Among all possible minimum-weight spanning trees, let T be one that agrees with F on the largest initial segment of $(e_1, \dots, e_{n-1})$, i.e. if k is the smallest index s.t. $e_{k+1} \notin T$, then there is no minimum-weight spanning tree which contains $\{e_1, \dots, e_{k+1}\}$. As by assumption F is not minimum-weight, we have $k < n - 1$. As T is a spanning tree which does not contain e_{k+1}, we know that $T + e_{k+1}$ contains a cycle C. As F did not contain cycles, there is one edge $e \in C \subseteq T$ which is not in F. Now $T + e_{k+1} - e$ is a connected graph of order n and size $n-1$, whence still a spanning tree. It contains the edges $\{e_1, \dots, e_k, e_{k+1}\}$, hence it can no longer be of minimum weight. This means that $w(e_{k+1}) > w(e)$. But as $e \notin F$ and in particular $e \notin \{e_1, \dots, e_k\}$ this means e was available at the step of the algorithm after we added e_k and of less weight than e_{k+1}. This contradicts the assumption that the algorithm chooses the edge of minimal weight which keeps F acyclic. □

Lemma 3.30

If G is a connected weighted graph s.t. distinct edges have distinct weights, then there is a unique minimum-weight spanning tree.

Proof. Homework. □