UPPSALA UNIVERSITET

Föreläsningstackneningar

Grafteori

Rami Abou Zahra

Inlämningsdatum
December 5, 2022

Contents

1. TODO

- Internet Spanning Tree protocol (look up)
- Weighed graphs adjacency matrix (what does symmetry mean?)
- Scalarproduct
- See traveling salesman problem (find shortest Hamilton cycle in a weighted complete graph)
- Add example * to list of graphs that you need to know

2. Bridges of Köningsberg

This was the birth of graphtheory. The idea here is that the precise location of where the person is does not matter, only the placement of the bridges and mainland. Therefore, we can encode the position by an abstract point (*vertex*) and connect these to *edges* to represent bridges.

2.1. **Vocabulary.**

We therefore obtain the follwing:

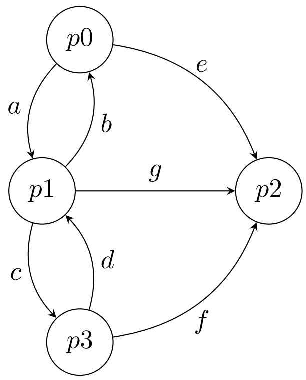

Figure 1.

Definition/Sats 2.1: Multigraph

A *multigraph* G is a tripple $G = (V, E, \iota)$ consisting of:

- A set V of vertices
- A set E of edges
- $\iota : E \to \{A \subseteq V \mid |A| = 1 \text{ or } |A| = 2\}$

Example:
$\iota(c) = \{2, 3\} = \iota(d)$
$\iota(e) = \{1, 4\}$

Anmärkning:
Notice that the graphical view (and the placement of the vertices) is not reflected in the tripple, therefore we can draw the same graph in a completely different manner.

Loops:
This is what happens when $|A| = 1$:

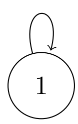

Figure 2.

Parallell edges:
$\iota(e) = \iota(e')$

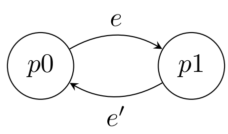

FIGURE 3.

Neighbours/adjacent:

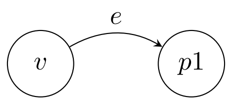

FIGURE 4. v is *incident* to e and a neighbour to w

Anmärkning:
A loop means the vertex belongs to its own Neighbourhood.

Definition/Sats 2.2: Finite graph

We say that a graph G is finite if we have:

$$|V| + |E| < \infty$$

Definition/Sats 2.3: Walk

Let $G = (V, E, \iota)$ be a graph
A *walk* of length k is a sequence $v_0 e_1 v_1 e_2 v_2 \cdots e_k v_k$ where as the notation suggests, $e_1, \cdots, e_k$ are edges and $v_0, \cdots, v_k$ are vertices such that $\iota(e_i) = \{v_{i-1}, v_i\}$ for $i = 1, \cdots, k$

Definition/Sats 2.4: Trail

A *trail* is a walk that uses no edges twice. This is something we want in the Bridges of Köningsberg

Definition/Sats 2.5: Path

A *path* is a walk that uses no vertex twice.

Definition/Sats 2.6: Circuit

A *circuit* is a trail where first and last vertex coincide.
Meaning I start somewhere, dont repeat edges, and return at start place

Definition/Sats 2.7: Cycle

A *cycle* is a circuit where the first and last vertices are the only vertices coinciding

Example:
Using the bridges, an example of a trail and a circuit, but not a cycle because vertex 3 is visited twice $1a3g4f2c3b1$
An example of a cycle would be $1a3b1$

Anmärkning:
Every path is a trail.
Every cycle is a circuit.

Definition/Sats 2.8: Eulerian trails

A trail is called *Eulerian* if it uses every edge in the graph

Definition/Sats 2.9: Eulerian circuits

A circuit using every edge is called an *Eulerian circuit*

Anmärkning:
If a graph admits an Eulerian circuit, then the graph is called *simply Eulerian*

Definition/Sats 2.10: Connected vertex

Let $G = (V, E, \iota)$ be a graph
We say that a vertex $v \in V$ is *connected* to a vertex $w \in V$ if there exists walk (or equivalently a trail/path) starting in v and ending in w

If v is connected to w for all $v, w \in V$, then the graph G is *connected*

What we are saying here is that we call vertices that we can walk to connected.

Anmärkning:
v is conncted to v (every vertex is connected to itself)
Moreover, if v is connected to w, then w is connected to v.
v is connected to w and w connected to z, then v is connected to z.

Anmärkning:
Connection is an equivalence relation.

Definition/Sats 2.11: Connected components

Equivelance classes of the equivalence relation are called *connected components.*

Definition/Sats 2.12: Degree of vertex

Let $G = (V, E, \iota)$ be a graph and $v \in V$. The *degree* of v is $\deg(v)$ and is the number of half-edges incident to v.

The reason we do half-edges is because we want loops to count twice (once for exit, and once on entry)

Definition/Sats 2.13: Euler; 1736

A finite connected graph is Eulerian iff all its vertex degrees are even.

Bevis 2.1

In $\Rightarrow$ direction. Any vertex on the circuit needs to have even degree because you need a half-edge to go into the vertex and another one to go out.
Since it is connected, if I visit every vertex I also visit every edge and these come in pairs

In $\Leftarrow$ direction. Assume $G = (V, E\iota)$ is finite, connected, and only has even degrees.
Assume G as no loops (convenience). We dont know if we can build an Eulerian circuit or even if we have a circuit, but we know that there is a trail (since it is connected)
Therefore, consider a trail $J = v_0e_1v_1 \cdots e_kv_k$.
Since the graph is finite, then there is a maximum trail, suppose J is a maximum trail (implying max length k). Then we cant possibly extend it, so any edge we see at k must already be on the trail.
What we want to show is that $v_0 = v_k$ because of this. Then we actually have a circuit.
Therefore, assume there are $2s \quad (s \in \mathbb{N})$ edges incident to v_k. We know there is an even number of edges (because we excluded loops).
If we look at our trail $v_0e_1v_1 \cdots v_{i-1}e_i \underbrace{v_i}_{v_k} e_{i+1}v_{i+1}$
Then e_i and e_{i+1} are incident to $v_k = v_i$, but so is v_k. But v_k only has one edge, therefore e_1 has to be incident to $v_k = v_0$
We have now shown we have a trail, we show it is Eulerian.
Assume for a contradiction that it is not Eulerian. This means that there are parts not in our trail.
There is $e \in E$ with endpoints $\iota(e) = \{v, w\}$ s.t e is not on J but one of v, w is.
WLOG v is on J. Say $v = v_j$ for some j.
Consider $wev_je_{j+1} \cdots e_k \underbrace{v_k}_{v_0} e_1v_1e_2v_2 \cdots e_jv_j$, we claim that this is a trail. Notice here that we have length $k+1$, which is longer than k. Contradiction. $\square$

Anmärkning:
A useful proof-tool in graphtheory is setting up a situation where we fix a maxlength and argue the contrary.

Anmärkning:
Notice how $\Rightarrow$ was "obvious", we call this *TONCAS* - The Obvious Necessary Conditions Are Sufficient

Anmärkning:
If we have loops, we can simply traverse these loops and add them to our trail. This will not affect the proof.

Corollary:
A finite connected graph admits an Eulerian trail iff either 0 or 2 of its vertex degrees are odd

We can show this by retracing this back to the previous theorem. If we have 0 odd degrees, then the theorem holds.
If we have 2 vertices of odd degree, then we can draw an additional edge between v, w. This means that both of the vertices that had odd degrees have gotten their degrees bumped up by one, so they know have even degree, which implies the theorem (is an Eulerian circuit), so it visits all the edges (and especially the new edge). Then we can remove the new edge from the Eulerian circuit, which gives an Eulerian trail in the original graph.

If we look at the statement of the corollary, it leaves a graph. What happens if it has 1 odd vertex degree? We are gonna show that this is impossible.

Definition/Sats 2.14: Handshake lemma

Let $G = (V, E, \iota)$ be a finite graph.
Then

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

In particular, G has even number of vertices of odd degree. (odd+odd = even, even + even = even)

Bevis 2.2: Handshake lemma

We use a trick from combinatorics (double counting). We identify a quantity and count it in 2 different ways.

We double count half-edges. Every edge gives 2 half-edges, so we $2|E|$ half-edges. On the other hand, every vertex gives $\deg(v)$ half-edges $\Rightarrow \sum_{v \in V} \deg(V)$ half-edges.
It does not matter how I count them, therefore these quantities have to be the same. $\square$

Anmärkning:
We can also use induction to show the Handshake lemma.
Start with 0 edges on V, which implies all the degrees are 0. Then add edges 1 by 1. And whenever you add an edge, the RHS increases by 2.

What happens if we have 4 vertices of odd degree?
We can partition $E = E_1 \cup E_2$ such that E_1 is a edge set of a trail and so is E_2 (but $E_1 \cap E_2 = \emptyset$)

3. Simple graphs

The idea of simple graphs is to forbid parallell edges and loops. Here, we dont care how things are connected, but which things that *are*.

For example, we can encode the game *Towers of Hanoi* as a simple graph by letting n disks be stacked on $3pegs$
We can therefore encode a game state by a string of length n

Definition/Sats 3.1: Simple graph

A *simple graph* is a multigraph without parallell edges or loops
An equivalent definition, it is a pair $G = (V, E)$ where $E \subseteq \mathcal{P}_2(V)$

By $\mathcal{P}_2$ we mean the powersets of size 2:

$$\mathcal{P}_2(V) = \{A \subseteq V \mid |A| = 2\}$$

Anmärkning:
Our ι is gone! This is because by not having parallell edges and loops, then $\iota : E \to \mathcal{P}_2(V)$ is injective and we can identify the output of ι with its input, and that is what the definition of a simple graph is

Anmärkning:
Every graph is a multigraph

Lemma 3.1

Any simple graph on n vertices has at most $\binom{n}{2}$ edges

Bevis 3.1

The edge set $E \subseteq \mathcal{P}_2(V)$ and $|\mathcal{P}_2(V)| = \binom{n}{2}$ □

Anmärkning:
This implies that simple graphs are finite. In multigraphs, we could put arbitrary edges between vertices, but here it is not accepted.

Our vertex set is arbitrary, it doesnt matter if $V = \{1, 2, 3, 4\}$ or $V = \{a, b, c, d\}$, we need to set up a notion of "sameness" in graphs taking into account that the vertex set is arbitrary.

Definition/Sats 3.2: Labelled graph

A *labelled graph* is a simple graph with a fixed vertex set, commonly $V = \{1, 2, \cdots, n\}$ if V is finite.

Lemma 3.2

There are $2^{\binom{n}{2}}$ labelled graphs on n vertices.

Bevis 3.2

Since $V = \{1, 2, \cdots, n\}$ is fixed, two graphs (V, E) and (V, E') coincide iff $E = E'$

Conversely, any subset of $\mathcal{P}_2(V)$ defines an edge set. We are essentially looking for $\mathcal{P}(\mathcal{P}_2(V))$, and the cardinality of this is $2^{|\mathcal{P}_2(V)|} = 2^{\binom{n}{2}}$ $\square$

Definition/Sats 3.3: Morphism

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

A *morphism*

$$\varphi : G \to G'$$

is a map

$$\varphi : V \to V'$$

such that $\{v, w\} \in E \Rightarrow \{\varphi(v), \varphi(w)\} \in E'$

Example:
See example 15

Anmärkning:
Graph-morphisms do not need to be injective/surjective, nor do they need to exist

Graph-morphisms preserve edges between graphs, thats their whole point

Definition/Sats 3.4: Identity morphism

For every simple graph G, theere is an identity morphism $id_G : G \to G$ where $id_G : V \to V$ is the identity map

For simple graphs G, G', G'' and morphisms

$$\varphi : G \to G'$$
$$\varphi' : G' \to G''$$

There is a morphisms $\varphi' \circ \varphi : G \to G''$, given by the map $\varphi' \circ \varphi : V \to V''$

Definition/Sats 3.5: Isomorphism

Two graphs G, G' are *isomorphic* if there is a bijective morphism $\varphi : V \to V'$ and $\{v, w\} \in E \Leftrightarrow \{\varphi(v), \varphi(w)\} \in E'$

Another way of saying this there is $\varphi : G \to G'$ and a $\psi : G' \to G$ such that $\varphi \circ \psi = id_{G'}$ and $\psi \circ \varphi = id_G$

Anmärkning:
Isomorphic graphs are not necessarily the same if they are labelled. We need to make sure the degree of each vertice coincide, and that we dont lose any edges.

Definition/Sats 3.6

The number g_n of non-isomorphic simple graphs on n vertices satisfies the following:

- $g_n = \dfrac{2^{\binom{n}{2}}}{n!}\left(1 + \dfrac{n^2-n}{2^{n-1}} + \dfrac{8n!}{(n-4)!}\cdot\dfrac{(3n-7)(3n-9)}{2^{2n}} + \mathcal{O}\left(\dfrac{n^5}{2^{5n/2}}\right)\right)$

In particular, g_n behaves asymptotically as $2^{\binom{n}{2}}/n!$ in the same way that the probability distribution Hyp becomes Bin for large populations. When we make lots of graphs, eventually, the number of graphs that are isomorphic are so small they dont matter in the grand scheme.

3.1. **Special graphs.**

Some (simple) graphs are so special (5 of 'em) that they are given special names:

- The complete graphs on n vertices, denoted by K_n. All $\binom{n}{2}$ edges are present (every vertex is a neighbour of everything else)
- The path graph of length l, denoted by P_l is just a regular path as a graph
- The cycle graph on n vertices, denoted by C_n $(n \geq 3)$
- The complete bipartite graphs, denoted $K_{a,b}$. Here, V is partitioned as the disjoint union $V = V_a \cup V_b$. This means $|V| = a + b$. There are no edges between two vertices in the same set, but all possible edges are between the two sets.
  Notice that $K_{a,b} \cong K_{b,a}$
- The complete r-partite graphs $K_{a_1,\cdots,a_r}$ has a vertex set $V = \bigcup_{i=1}^{r} V_{a_i}$ such that $|V_{a_i}| = a_i$
  We say that two vertices are neighbours iff they are in different sets.

Lemma 3.3

The complete r-partite graph $K_{a_1,\cdots,a_r}$ on n vertices (sum of all $a_i = n$) has

$$|E| = \frac{1}{2}(n^2 - a_1^2 - \cdots - a_r^2)$$

Bevis 3.3

A vertex in set V_{a_i} has $n - a_i$ neighbours
By the Handshake lemma, $2|E| = \sum_{v\in V}\deg(v) = \sum_{i=1}^{r} a_i(n-a_i) = n\sum$

$$2|E| = \sum_{v\in V}\deg(v) = \sum_{i=1}^{r} a_i(n-a_i) = n\underbrace{\sum_{i=1}^{r} a_i}_{=n} - \sum_{i=1}^{r} a_i^2$$

$$n^2 - a_1^2 - \cdots - a_r^2$$

□

Anmärkning:
$K_{a,b}$ has $\frac{1}{2}(n^2 - a^2 - b^2) = ab$ edges and $K_n = K_{1,\cdots,1}$ has $\frac{1}{2}(n^2-n) = \binom{n}{2}$ edges

Definition/Sats 3.7: Subgraph

Let $G = (V, E)$ be a simple graph.
A simple graph $H = (V', E')$ is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$

Definition/Sats 3.8: Induced subgraph

An *induced* subgraph, is a subgraph $H = (V', E')$ of G, such that $E' = \{\{x, y\} \in E \mid x, y, \in V'\}$

Denoted by $H = G[V']$

Definition/Sats 3.9: Edge-induced subgraph

An *edge-induced* subgraph is a subgraph $H = (V', E')$ such that $V' = \{v \in V \mid v \text{ is incident to some } e \in E'\}$

Denoted by $H = G < E' >$

Definition/Sats 3.10: Spanning subgraph

A subgraph $H = (V', E')$ of G is a *spanning* subgraph if $V' = V$

Anmärkning:
There is a way to extend this into multigraphs, but you need to find a way to take care of ι

Definition/Sats 4.1: Tree

A *tree* is a graph that is both connected and contains no cycles

It follows automatically that a tree is a simple graph. No cyclic subgraphs.

One of the key motivations behind studying trees is sorting algorithms. The follow a binary tree, which has a root node and a left-right structure.
To us, this is not that important.

Lemma 4.1: Leaves

Every finite tree on at least 2 vertices contains at least 2 vertices of degree 1.

Such vertices are called *leaves*

Intuition:
If we disregard the definition of the tree for a second, and look at how the trees *actually* look like, we see that we can always find a "top" node and a "bottom" node. What makes these nodes top resp. bottom nodes? Well, they have degree one!

Notice how we called the top node the "top" node, now we can start baking in the definition of the trees to attempt to construct a proof.
If we can show that we will always have a top and bottom node, we are done (because a tree is a connected graph, therefore there always exists a walk from the top node to the bottom, and therefore the degree of the node has to be greater than 1 and since there are no cycles, neither the top nor the bottom node can have degree 2).

The path of max length should start at our top node, and end at our "bottomest" node. We can now construct the following proof:

Bevis 4.1

Let T be a finite tree on vertices ≥ 2.
Consider a path $P = xe_1x_1 \cdots e_ky$ of maximum length

If such path exists, we will show that x, y must be our desired leaves.

Assume WLOG y has $\deg(y) \geq 2$. Then y has a neighbour $z \neq x_{k-1}$
If z is not on P , then $xe_1 \cdots e_kyy, zz$ is a longer path, which is a contradiction.

Otherwise z is on P:

$$x - x_1 - x_2 \cdots - z - \cdots - x_{k-1} - y$$

This gives a cycle (there is a path from y to z), which is a contradiction. Hence, $\deg(y) = 1$ and $\deg(x) = 1$ $\square$

The trick here is that we took some suitable substructure of maximum lenght, and from thiswe arrived at this property. One could say that we initially wanted to show that this follows from the root down to the last node.

Lemma 4.2

Any tree on n vertices has $n - 1$ edges

Bevis 4.2

We will use induction over n:

- $n = 1$: has 0 edges
- Assume the claim is true for some $n \geq 1$, and let T be a tree on $n+1$ vertices.
  By Lemma 4.1, T contains a leaf (at least 2) v. Obtain T' from T by deleting v and its incident edge.
  Now T' has n vertices, and therefore $n-1$ edges.
  This means T has $n = (n+1) - 1$ edges

□

It is fairly easy to count number of labelled trees given n vertices. Counting isomorphic trees only yields an asymptotic relationship (as previous)

Definition/Sats 4.2: Cayley

There are n^{n-2} labelled trees on n vertices

Bevis 4.3

The proof is a little difficult. The main idea is to find a bijection.

On the one hand we have labelled trees, and on the other hand we have sequences (Prufer sequences) of length $n-2$ with n trees from $1, \cdots, n$

We will find 2 algorithms that transform one hand to the other and then show that they are the same if inverted.

- **Algorithm 1**:
  Let T be a tree n vertices $\{1, \cdots, n\}$
  While T has ≥ 3 vertices, remove the leaf with the smallest label, and write down its neighbours label as next entry in the sequences
  Stop when there are 2 vertices left
  Remember here that a leaf has a unique neighbour.

- **Algorithm 2**:
  We now want to go from a Prufer sequence to trees.
  Let $A = (a_1, \cdots, a_{n-2})$ be a Prufer sequence.
  To each $i = 1, \cdots, n$, count how often i appears in the Prufer sequence, +1. Denoted by d_i
  For $s = 1, \cdots, n-2$, find the smallest $j \in \{1, \cdots, n\}$ such that $d_j = 1$ (smallest value that doesnt occur in the Prufer sequence, since if $d_j = 1$ and we are adding one)
  Draw an edge between j and a_s and reduce d_{a_s} and d_j by 1 each.
  In the end, two vertices $u, v \in \{1, \cdots, n\}$ will remain with $d_u = d_v = 1$ (this is a claim, **CHECK**)
  Connect u, v by an edge. At this point you will have a tree.

Claim: Algorithm 1 & 2 are mutually inverse to each other, thus establishing the bijection, and the proof follows.
(In reality we need to actually check that the algorithms work) □

The way we defined trees as being connected and cycle free is not the only definition, in fact, we have the following theorem:

Definition/Sats 4.3

The following are equivalent:

- T is a tree
- For any two vertices $x, y \in T$, there exists a unique path from x to y (key-point: unique, from connectedness we already know there exists a path)
- T is edge-minimal among connected graphs, i.e removing an edge from T will disconnect T
- T is edge-maximal among cycle-free graphs, i.e adding an edge to T must create a cycle.

Bevis 4.4

Let $T = (V, E)$ be a simple graph. We will show all the points above using implications.

- **First point implies the second**
  T is a tree, i.e connected and cycle-free.
  Take arbitrary vertices $x, y \in V$. Since T is connected, there is a path from x to y
  Assume there is a second path. This contradicts that it is cycle-free, therefore the path is unique.

- **Second point implies the third**
  Consider $\{x, y\} \in E$. By the second point, this edge forms the unique path between x, y.
  If we remove the unique path, there will not be a path between x, y and thus disconnects x from y, and we now have a disconnected graph

- **Second point implies the fourth**
  Consider 2 non-adjacent vertices $\{x, y\}$. By the second point, there is a unique path P from x to y.
  Introducing the new edge $\{x, y\}$ creates a cycle.

- **Third point implies the first**
  If T is edge-minimal among connected graphs, then in particular it is connected, we must now show that it is cycle-free.
  Assume T contains a cycle. Deleting any edge of the cycle would *not* disconnect T, therefore T cannot be edge-minimal among connected, which is a contradiction, thus it cannot contain a cycle and therefore it is a tree
- **Fourth point implies the first**
  T is edge-minimal among cycle-free graphs, in particular T is cycle-free. If T is not connected, then we can introduce an edge between two different connected components, and this new edge will *not* introduce a cycle, which contradicts it being edge-minimal.

$\square$

Definition/Sats 4.4: Spanning tree

Let G be a graph. A *spanning tree* of G is a spanning subgraph that is a tree

Anmärkning:
If G is disconnected, then a spanning subgraph of G is disconnected

Definition/Sats 4.5

If G is a connected graph, then G contains a spanning tree.

If G is a finite graph, then we can use the last theorem to show this. The problem arises when G is infinite. In order to show this, we need a more powerful tool.

Definition/Sats 4.6: Zorns lemma

Let A be a non-empty set equipped with a partial order "$\geq$".

A susbet $C \subseteq A$ is a *chain*, if $\forall c, c' \in C$ we have $c \leq c'$ or $c' \leq c$

Assume that for any chain $C \in A$, there is an upper bound $b \in A$ (i.e $c \leq b \, \forall c \in C$).
Then, there exists an element $m \in A$ that is maximal, which means that $m \leq a \Rightarrow m = a$

Bevis 4.5: Theorem 4.5

Let A be the set of all cycle-free spanning subgraphs of G. G here can be a multigraph.

We need to define a partial order. For $H, H' \in A$, define $H \leq H'$ is H is a subgraph of H'
Now you might wonder how subgraphs work wiht multigraphs, since they are cycle free it will work the same.

Then $(A, \leq)$ is a partially ordered set (**CHECK**). Furthermore, $A \neq \emptyset$ because the graph $(V, \emptyset) \in A$

Let C be a chain in A, consisting of elements (V, E_i) for $i \in I$ (I is some index set).
What we want to show is that such a chain has an upper-bound.

Define $H_b = \left(V, \bigcup_{i\in I} E_i\right)$. We want to show that b is an upper-bound for C.

By construction, H_b is a spanning subgraph of G. Why? It contains all of the vertices, and the individual sets are subsets of G.

Moreover, assume it contains a cycle with edges $e_1, \cdots, e_r$. Then, for every $l = 1, \cdots, r$, there exists $i(l) \in I$ such that $e_l \in E_{i(l)}$ (at least one set must be in the union)

Since C is a chain, one of the $H_{i(1)}, \cdots, H_{i(r)}$ contains all other subgraphs simply because they are all comparable. (Say $H_{i(j)}$)

Thus, $e_1, \cdots, e_r$ is contained in the edge set $E_{i(j)}$, hence $H_{i(j)}$ contains a cycle.

This is a contradiction, because H_b is cycle-free. This means $H_b \in A$

Is H_b an upper-bound for the chain C? Yes, H_b bound C because

$$E_s \in \bigcup_{i\in I} E_i \qquad \forall s \in I$$

Our chain was arbitrary, this means that we can do this for any chain.

By Zorns lemma, there is $H \in A$ that is maximal with respect to the partial order we defined. This means, H is edge-maximal among cycle-free spanning subgraphs of G.

This means, H is a spanning tree. (edge-maximal cycle free means it is a tree) $\square$

5. Kirchoffs Matrix-Tree theorem

Definition/Sats 5.1: Complexity

LLet G be a labelled graph. The *complexity* of G, denoted by $t(G)$ is the number of spanning trees of G

Example:
The complexity of a complete graph K_n is n^{n-2}

From now on (this lecture), we assume that $G = (V, E)$ is a labelled ($V = \{1, 2, \cdots, n\}$) finite and simple graph with $E = \{e_1, \cdots, e_n\}$

Definition/Sats 5.2: Ajacency matrix

The *adjacency matrix* A of G is the $n \times n$ matrix having entries:

$$A_{ij} = \begin{cases} 1 & \text{if } i, j \text{ are ajacent} \\ 0 & \text{otherwise} \end{cases}$$

Example:
has adjacency matrix:

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

Anmärkning:

- A is symmetric (since adjacency is a symmetric relationship) $\Rightarrow$ real eigenvalues
- $A_{i,i} = 0$ for all $i = 1, \cdots, n$
- $tr(A) = \sum_{i=1}^{n} A_{i,i} = 0 = \sum_{i=1}^{n} \lambda_i$
- If the only zeroes are in the diagonal, then the graph is complete

We declare one part of the graph to be negative and the other to be positive. This is just a choice, and does not have an effect on the construction of the graph

Definition/Sats 5.3: Incidence matrix

The *incidence matrix* D of G with respect to a fixed orientation is the $n \times n$ matrix having entries:

$$D_{i,j} = \begin{cases} 1 & \text{if } i \text{ is the positive endpoint of } e_j \\ -1 & \text{if } i \text{ is the negative endpoint of } e_j \\ 0 & \text{otherwise} \end{cases}$$

Anmärkning:
$e_i \in E$

Anmärkning:
We may choose a vertex to be negative for one edge, but it does not have to be fixed, for another edge that same vertex might be positive. This does not affect the graph.

How do we relate these matrices to trees? Well, this is the idea of the next few lemmas:

Lemma 5.1

Let D be an incidence matrix to a graph G. Then the following statements are true:

- The sum of any column of D is 0, thus the rank(D)$\leq n-1$ (since the values in a column are either 1 or -1 (once) and 0)
- If G is connected, then the rank$(D) = n-1$
- If G has c components, then rank$(D) = n-c$

This is good, because by looking at D we can say something about if the graph is connected or not.

Bevis 5.1

Denote by r_i the i-th row vector of D:

- The sum of any column is 0 by the definition of D (each row has 2 non-zero entries that are 1 or -1).

  This means that $\underbrace{\sum_{i=1}^{n} r_i}_{\text{linjärkomb.}} = 0 \Rightarrow \text{rank}(D) \leq n-1$

- Suppose there is a non-trivial linear combination of the form $\sum_{i=1}^{n} \alpha_i \cdot r_i = 0$
  Consider the row k for which $a_k \neq$
  In row k, there is a non-zero entry for every edge incident to the vertex k
  For each of these columns where row k has non-zero entries, there is a unique second entry in the same column (column s), but in a different row (r_j).
  This other entry r_j will have an opposite sign
  We know $\alpha_k \cdot r_{k,s} + \alpha_j r_{j,s} = 0$
  This means that $\alpha_k = \alpha_j$ since they only differ by a sign

  As a consequence of this, if $\alpha_k \neq 0$ then $\alpha_j = \alpha_k$ for all j adjacent to k
  Since G is connected, this argument extends to all of G:s vertices (and thereby all of G), hence the linear combination $\sum_{i=0}^{n} \alpha_i r_i$ is a scalar multiple of $\sum_{i=1}^{n} r_i$

  This gives that the rank$(D) = n-1$

- This follows quite easily from proof from the above point.
  If G has c components, relabel the vertices and edges in such a way that D takes the form of having edges and vertices of component 1 in the diagonal place 1,1.
  Then rank$(D) = n-c$ by applying previous point to each block

□

Lemma 5.2

Any square submatrix of an incidence matrix D has determinant 0 or ± 1

Bevis 5.2

Pure linear algebra. Left as an exercise to the reader □

More intersting things that follows from Lemma 5.2 is if you pick a set of edges $S \subseteq E$. Denote by D_s the submatrix containing exactly the columns that correspond exactly to the edges in S

Then D_s is an incidence matrix to the spanning subgraph (V, S)

In particular, if $|S| = n-1$, then by second point of Lemma 5.1, $\text{rank}(D) = n-1$ iff (V,S) is connecfted iff (V,S) is a spanning tree

Lemma 5.3

Let $S \subseteq E$ with $|S| = n-1$
Let M denote any $(n-1)\times(n-1)$ submatrix of the $n\times(n-1)$ matrix D_s

Then M is regular (invertible) iff (V,S) is a spanning tree of G

Bevis 5.3

We already know that the rank of $D_s = n-1$ iff (V,S) is a spanning tree.
In that case, if we just take D_s and delete any row from it, it will create an invertible matrix, so if I start with a spanning tree, then M is regular

Conversly, assume M is regular. This means that D_s contains at least $n-1$ linear independent rows and the same number of linear independent columns
This means that $\text{rank}(D_s) \leq n-1$, but since this thing only has $n-1$ columns this must be equal to $n-1$
(V,S) is connected on $n-1$ edges, therefore it is a spanning tree.

$\square$

We now have all the criteria to decide if a subgraph is a spanning tree

Lemma 5.4

Let A be the adjacency matrix of G and let D be an incidence matrix of G (fixed labelling and ordering of edges as well as ordering of orientation)

Denote by Δ the $(n\times n)$ diagonal matrix with entries $\Delta_{i,i} = deg(i)$

Then $\Delta - A = DD^T$ and this matrix is the ***Laplacian matrix*** of G

In particular, this product is independent from the orientation of G (because the LHS does not rely on the orientation of the edges)

Bevis 5.4

$(DD^T)_{i,j}$ is the scalar product of the row vectors r_i and $_j$ of D (because of how matrix multiplication works)

If $i = j$, this means that $(DD^T)_{i,i} = \sum_{i=1}^m r_{i,s}^2 = \sum_{s=1}^m D_{i,s}^2 = \sum_{s=1}^m \mathbb{1}\,\{\text{edge } s \text{ is incident to vertex } i\} = deg(i) = (\Delta - A)_{i,i}$

If $i \neq j$, then:

$$(DD^T)_{i,j} = \sum_{s=1}^m D_{i,s}D_{j,s} = \begin{cases} 0 & \text{if } i,j \text{ non-adjacent} \\ -1 & \text{if } i,j \text{ adjacent} \end{cases}$$
$$= (\Delta - A)_{i,j}$$

This thing is non-zero iff $D_{i,s} = \pm 1$ and $D_{j,s} = \mp 1$. This only happens if s is an edge connecting i to j
But between any given vertices, there can be at most one edge $\square$

Lemma 5.5

Let $Q = \Delta - A = DD^T$ the Laplacian of G.
Denote by J (sometimes J_n) the $n \times n$ matrix with all entries $= 1$

Then the adjoint matrix of the Laplacian Q is a scalar multiple of J

Bevis 5.5

Observe:

- $\text{rank}(Q) = \text{rank}(D)$ (since $Q = DD^T$)

If G is disconnected, the $\text{rank}(Q) < n - 1 \Rightarrow$ all cofactors are 0

If G is connected, then $\text{rank}(Q) = n - 1$.

We know from linear algebra that $Q \cdot \text{adj}(Q) = \det(Q) \cdot I_{n \times n} = 0$

This means that every column vector of $\text{adj}(Q)$ is in $\ker(Q)$

But the dimension of the kernel is $n - \text{rank}(Q) = 1$ and $(1, \cdots, 1)^T \in \ker(Q)$ because:

$$(\Delta - A) \cdot \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = deg(i) - \sum_{j \text{ adj. to } i} 1 = deg(i) - deg(i) = 0$$

Therefore, $(1, \cdots, 1)^T$ is a basis of the kernel

This means every column of $\text{adj}(Q)$ is a scalar multiple of $(1, \cdots, 1)^t$

Since $Q^T = (\Delta - A)^T = \Delta^T - A^T = \Delta - A = Q$, also $\text{adj}(Q)^T = \text{adj}(Q)$

But this means that every row $\text{adj}(Q)$ is a scalar multiple of $(1, \cdots, 1)^T$

If every row and every column is a scalar multiple of that vector, then since every row and column intersect, $\text{adj}(Q)$ is a scalar multiple of J $\square$

Definition/Sats 5.4: Kirchoffs Matrix-Tree theorem

Let G be a connected finite simple graph with Laplacian matrix Q.

Then, $\text{adj}(Q) = t(G) \cdot J$ (equal to any cofactor of its Laplacian matrix)

Equivelantly, if $\lambda_1, \cdots, \lambda_{n-1}$ are the non-zero eigenvalues of Q, then $t(G) = \frac{1}{n}\lambda_1 \cdot \cdots \cdot \lambda_{n-1}$ (if not connected, then there are more than one non-zero eigenvalues)

Definition/Sats 5.5: Cauchy-Binet

Assume you have two $n \times n$ matrices A, B where $n \leq m$

Then, $\det(AB^T) = \sum_{s \subseteq \{1, \cdots, m\}} \det(A_s)\det(B_s^T)$ and $|s| = n$

Bevis 5.6: Matrix-Tree theorem

We already know that all the cofactors are the same, so it is enough to evaluate one cofactor of Q

Let $\underset{\sim}{D}$ be the incident matrix with the last row deleted.
This means that $\det(\underset{\sim}{D}\underset{\sim}{D}^T)$ is a cofactor of $Q = DD^T$

By Cauchy-Binet we have $\det(\underset{\sim}{D}\underset{\sim}{D}^T) = \sum_{S \subseteq E} (\det \underset{\sim}{D}_s)^2$ where $|S| = n - 1$

This means that it is a square $(n-1) \times (n-1)$, which by previous lemma has determinant 0 or 1. We have also seen that the determinant is 1 iff (V, S) is a spanning tree. Therefore, the sum above is just the number of spanning trees $(t(G))$ $\square$

Example:
Assume it is the exam and you need to prove Cayleys formula but you have forgotten it and we want to find $t(K_n)$

We have Q with $n-1$ on its diagonal and all the other edges are -1.

How do we find the eigenvalues? Its not recommended to find the characteristic polynomial, but we may be able to guess the eigenvalues (and verify using trace).

$Q \cdot (1, \cdots, 1)^T = 0$ (sum of each row is 0) so $(1, \cdots, 1)^t$ is eigenvector to eigenvalue 0
We try $Q \cdot \underbrace{(0, \cdots, 1, -1, \cdots, 0)^T}_{\text{eigenvector to eigenvalue } n} = (0, \cdots, n, -n, \cdots, 0)^T$. These form $n-1$ linearly independent eigenvectors

Now we can use Kirchoff. By the Matrix-Tree theorem, $t(K_n) = \dfrac{1}{n} n^{n-1} = n^{n-2}$

6. Weights and Distances

Definition/Sats 6.1: Weighted graph

A *weighted graph* is a finite simple graph $G = (V, E)$ together with a weight function $w : E \to (0, \infty)$

If $H = (V', E')$ is a subgraph of G, then its weight is defined to be $w(H) = \sum_{e \in E'} w(e)$

Definition/Sats 6.2: Minimum spanning tree

A *minimum spanning tree* (MST) is a spanning tree T of G such that $w(T)$ is minimal among all spanning trees in G

It is not clear that an MST exists. We started with a simple graph which has finite edges and therefore we have finite subgraphs. We are therefore looking for the minimum of a set, which certainly exists, but it may not be unique.

Lemma 6.1

Let G be a connected weighted graph.

If $w : E \to (0, \infty)$ is injective, then the MST is unique.

Bevis 6.1

Assume w is injective and suppose there are MST:s $T_1 = (V, E_1)$ and $T_2 = (V, E_2)$ with $E_1 \neq E_2$

Then a set D (not incidence matrix) $D = \{e \in E \mid e \in E_1 \vee e \in E_2 \text{ but not both}\}$

Pick $e \in D$ such that $w(e)$ is minimal (we can do this because all of our set is finite, so there must be a minimal weighted edge).
By definition, this edge lives in one of our 2 trees.

WLOG, $e \in E_1$ and therefore $e \notin E_2$

Add e to T_2 creates a cycle (a tree is edge-maximal among cycle-free graphs), an don this cycle there is an edge $e' \notin E_1 \Rightarrow e' \in D$

We now have:

- $w(e) < w(e')$ (by injectivity, they cannot have the same weight)
- $w(e') \leq w(e)$ (because otherwis T_2 would not be a MST)

Contradiction, and end of proof $\square$

Anmärkning:
Lemma 6.1 is not an equivelance. If there are edges with equal weight, it does not mean our MST is unique. The best example is if we start with a tree.

6.1. **Prims Algorithm.**

Of course, let $G = (V, E)$ be a connected, simple, weighted graph.

Set $T = (\{v\}, \emptyset)$ for any $v \in V$ (contains one arbitrary vertex from the graph and no edges)

While T is not spanning, find an edge $e \in E$ between $V(T)$ (all of the vertices that are already in the tree) and $V \backslash V(T)$ (all of the vertices are not already in the tree) such that $w(e)$ is minimal.

Add this edge e to T together with its endpoint from $V \backslash V(T)$

The moment T is spanning, we stop the algorithm.

Example:
Example 53 in lecture notes

When faced with 2 choices, pick as you want

Anmärkning:
You can get different spanning trees depending on which starting vertex you choose. Therefore, how can we know it is the MST? Theorem time!

Definition/Sats 6.3: Prims algorithm produces an MST

Prims algorithm produces an MST

Bevis 6.2

Let PA = Prims algorithm.
PA produces a spanning subgraph, and in fact it produces a connected spanning subgraph with n vertices and $n-1$ edges (by Lemma 4.2 we have a spanning tree tree T)

Let T' be a MST. We show that the weight $w(T)$ is at most ($\leq$) $w(T')$

Suppose $T \neq T'$. Consider the earliest edge e that was included in T but is not in T'

Partition V in 2 disjoint sets V_1 and V_2 such that V_1 contains all vertices that PA added to T before including e.

Before including e, all of the edges in T were in T', so if we look at $T[V_1]$ is a subgraph of T'

T' is a tree, so there must be one edge $f \neq e$ (edge e does not occur in T') in T' that connectes V_1 to V_2 *in* T'

Transform T' by adding e (creates cycle) and removing f (on this cycle is f, remove f destroys cycle)

PA chose e over f, this means that $w(e) \leq w(f)$

What happens now when $f \sim T''$ (when we rmeove f)? Well, the total weight should go down, so $w(T'') \leq w(T')$.

Since T' is an MST, this implies $w(T'') = w(T')$

We can repeat this until $T'' = T$. Since we have moved the latest point in which they differ (by repeating this, we essentially move this point as far back as possible)

Then it follows that $w(T) \leq w(T')$ so T is MST $\square$

Anmärkning:
Prims algorithm is an example of a *greedy algorithm*. This is because we do a locally optimal choice.

Anmärkning:
Regarding runtime, depends on the implimantation and how we choose the minimal edge. A good implimantation has a runtime of $\mathcal{O}(|E| + |V| \log(|V|))$

6.2. **Kruskals Algorithm.**

We start yet again with a connected weighted graph $G = (V, E)$.

Let S be a list of the edges in E, sorted by incfreasing order of weight (lightest edge is first)

Set $T = (V, \emptyset)$ and do the following:

While T is disconnected, delete the first entry in S and att it to T unless it creates a cycle (then just delete it)

Example:
Example 54 in lecture notes

Definition/Sats 6.4: Kruskals algorithm produces a MST

Kruskals algorithm produces a MST

Bevis 6.3

Denote Kruskals algorithm with KA.

Notice that KA by design produces a spanning cycle free connected graph (spanning tree). Denote this by T

If T is *not* MST, denote by T' MST that has the maximum number of edges in common T (among all MST)

Let e be the earliest edge in T that is not in T' (**is e the first one to be deleted but not added?**) (no, first one to be included in T but not in T')

Adding e to T' creates a cycle, and this cycle contains an edge $f \neq e$ that is not in T

Modify T' by adding e and removing f, this creates a new tree $f \sim T''$

Since T' is an MST, we have that $w(T') \leq w(T'')$, but at the same time, KA chose e over f so by removing f and adding e, therefore $w(T'') \leq w(T') \Rightarrow w(T') = w(T'')$ and T'' is MST

However, T'' has one more edge in common with T than T' had. This is a contradiction since T' had the most edges in common. $\square$

Anmärkning:
A good implimantation has a runtime of $\mathcal{O}(|E| \log(|V|))$

You can interpret edge weights, as distances.

Definition/Sats 6.5: Graph distance

Let $G = (V, E)$ be a simple weighted graph with weight function $w : E \to (0, \infty)$

For vertices $v, v' \in V$, we define the *graph distance* between v and v' by

$$d_G(v, v') = \min \left\{ \sum_{e \in E(P)} w(e) \mid \ P \text{ is a path from } v \text{ to } v' \right\}$$

We set $d_G(v, v') = \infty$ if no path from v to v' exists.

Anmärkning:
For non-weighted graphs, we choose $w(e) = 1$ for all $e \in E$

Lemma 6.2

Let $G = (V, E)$ be a connected weighted graph. Then the following statements are true:

- $\forall v, v' \in V$, we have $d_G(v, v') \geq 0$ and $d_G(v, v') = 0 \Leftrightarrow v = v'$
- $d_G(v, v') = d_G(v', v)$
- Triangular inequality holds: $d_G(v, v') + d_G(v', v'') \geq d_G(v, v'')$

Anmärkning:
This is a set with a metric.

Bevis 6.4

irst one is obvious, and so it the second

For the Triangular inequality, let P be a path of $d_G(v, v')$ from v to v' and Q be a path of length $d_G(V', v'')$ from v' to v'' (we know this path exists because there must be some path that realises these distances)

We obtain a walk from v to v'' by concatenating Q after P, problem is this is not a path, but erasing cycles from this walk creates a path. This only decreases the length. We found a path from v to v'' which is shorter than the sum of the distances

Hence, $d_G(v, v'') \leq d_G(v, v') + d_G(v', v'')$ $\square$

Definition/Sats 6.6: Diameter

The *diameter* of a weighted graph $G = (V, E)$ is the maximum distance between any 2 vertices:

$$\text{diam}(G) = \max \{d_G(v, v') \mid v, v' \in V\}$$

6.3. **Dijkstras Algorithm.**

Given a weighted graph $G = (V, E)$, select an inital vertex v_0 and initialize a distance function $d(v_0, *)$ (where $*$ is the argument of our distance function and is some other vertex) by:

$$d(v_0, v) = \begin{cases} 0 & v = v_0 \\ \infty & \text{otherwise} \end{cases}$$

Define a set Q of unvisited vertices. Initially, $Q = V$

Let v_0 be the currently visited vertex, and proceed as follows:

First, remove the current vertex v from Q

Second, for all neighbours v' to v in Q, check if $d(v_0, v) + w(\{v, v'\}) < d(v_0, v')$

If yes, then going through v is shorter than whatever path you have seen, so update $d(v_0, v')$ to $d(v_0, v) + w(\{v, v'\})$, otherwise keep $d(v_0, v')$

Third, New current vertex is $v \in Q$ with the smallest value $d(v_0, v)$

Repeat first to third step until Q is empty, and return $d(v_0, *)$

Anmärkning:
Interested in a concrete distance between vertices we can stop as soon as we hit it in step 3

Anmärkning:
Finding the path that realises the minimum distance, we can remember the previous vertex that we visited, and therefore we can trace back the steps and get a path

Anmärkning:
Runtime of $\mathcal{O}(|E| + |V| \log(|V|))$

Example:
Example 60 in lecture notes

7. Hamilton Cycles

We have previously discussed *Eulerian Circuits*, which was a circuit that used every edge in the graph

Definition/Sats 7.1: Closed Walk

Another way of defining an Eulerian circuit is by saying *a closed walk* using every *edge* exactly once

Definition/Sats 7.2: Hamilton Cycle

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

A *Hamilton Cycle* in G is a cyle containing ever vertex of G

If G admits a Hamilton cycle, then we say that G is Hamiltonian

You may think this looks similar to Eulerian circuits (and by definition they are similar), however with Eulerian circuits we had Eulers theorem, but there is no such thing for Hamiltonian graphs (deciding wether a graph is Hamiltonian is an NP-hard problem)

Definition/Sats 7.3: Dirac

Let $G = (V, E)$ be a graph on at least 3 vertices $n \geq 3$ such that every vertex has degree at least $\lceil \frac{n}{2} \rceil$

Then, G is Hamiltonian

Anmärkning:
If you have a Hamiltonian graph, introducing edges will not make it non-Hamiltonian

Bevis 7.1

Assume we have our graåh $G = (V, E)$ with $n = |V| =\geq 3$ and $\min_{v \in V} \deg(v) \geq \frac{n}{2}$

- G is connected (if it is not, we have no chance of seeing a Hamiltonian cycle):
  If G was not connected, then the vertices in the smallest connected component would violate the degree condition since if we have several connected components, we have at least 2 which means that the smallest component has at least $\frac{n}{2}$ vertices and therefore has at least $\frac{n}{2} - 1$ neighbours, which violates the condition
- G contains a cycle:
  Let P be a path in G of maximum length, say $P = v_0 e_1 v_1 \cdots e_k v_k$
  Since this path is maximum, all of the neighbours of v_k must already be on the path.
  By the degree condition we have at least $\frac{n}{2}$ (by the degree condition), the same is true for neighbours of v_0
  There is an edge $e_i = \{v_{i-1}, v_i\}$ such that v_i is adjacent to v_0 and v_{i-1} is adjacent to v_k
  This means we have a cycle, we can call this cycle C
- C is a Hamiltonian cycle:
  If C is not Hamiltonian, there exists a vertex that is not on the cycle (ie, a vertex v not on C but is adjacent to some v_j)
  Consider the path beginning in v and going to v_j and traversen along C through all vertices on C
  This path is longer than P, which contradicts P being a path of maximum length

Therefore, the path we found is a Hamiltonian cycle.

□

Anmärkning:
The bound $\dfrac{n}{2}$ for $\min_{v \in V} \deg(v)$ is optimal

Definition/Sats 7.4: Ore

Let G be a finite simple graph on $n \geq 3$ vertices such that for any pair of non-adjacent vertices $v, w \in V$ we have $\deg(v) + \deg(W) \geq n$

Then G is Hamiltonian

Anmärkning:
The proof for this one is the same as the proof for Diracs theorem

Definition/Sats 7.5: Closure

The *closure* of a finite simple graph $G = (V, E)$ is the result of the following procedure:

- For every pair of non-adjacent vertices $v, w \in V$, draw the edge $\{v, w\}$ if $\deg(v) + \deg(w) \geq n$
- Stop once no new edges can be introduced

Example:
If G satisfies the condition in Ores theorem, then $\text{clos}(G) = K_n$ is a complete graph and therefore Hamiltonian

Lemma 7.1: clos(G) is well defined

$\text{clos}(G)$ does not depend on the order in which edges are inserted.

Bevis 7.2

Consider $G = (V, E)$
Assume we construct $\text{clos}(G)$ in two different ways

- First one by adding edges $e_1, e_2, \cdots, e_k$
- Second one by adding edges $f_1, f_2, \cdots, f_{k'}$

Observe that we do not know if $k = k'$, however, if we construct this in 2 different ways, this means that there is a smallest j such that $e_j \neq f_j$

Let G_{j-1} be the graph constructed up to this point.

Let $e_j = \{v, w\} \Rightarrow \deg_{G_{j-1}}(v) + deg_{G_{j-1}}(w) \geq n$

These degrees do not decrease by adding $f_j, f_{j+1}, \cdots$. This means that eventually, the same edges will need to be introduced by one of the f:s

Hence, there is an $l > j$ with $f_l = \{v, w\}$

Modify $f_1, \cdots, f_{k'}$ in the following way:
$f_1, \cdots, f_{j-1}, f_l, f_j, f_{j+1}, f_{l-1}, f_{l+1}, \cdots, f_{k'}$

These sequences now coincide later than j, so we can repeat this argument until the two sequences coincide, and then the proof is finished. □

Definition/Sats 7.6: Bondy-Chvatal

A graph $G = (V, E)$ is Hamiltonian iff its closure is Hamiltonian

Bevis 7.3

$\Rightarrow$ is easy, because we just add edges (adding edges does not destory Hamiltonicity)

$\Leftarrow$ is a little trickier. Assume that $\text{clos}(G)$ is constructed using edges $e_1, \cdots, e_k$ yielding graphs $G_1, \cdots, G_k = \text{clos}(G)$

If G_j is Hamiltonian, then $G_{j+1}, \cdots, G_k$ (adding edges argument)

Assume that G_{j+1} is Hamiltonian but G_j is not (therefore Hamiltonicity occurs at G_{j+1})
Notice that those two graphs G_{j+1} and G_j differ by one edge (namely $e_{j+1} = \{v, w\}$)

This means that the Hamilton cycle in G_{j+1} contains this edge e_{j+1} and therefore $\{v, w\}$

G_j contains a path P from v to w traversing all vertices (take the Hamilton cycle and delete one edge)

Since $\{v, w\}$ is added to G_j, we have from the definition of closure that $\mathrm{G}_j(v) + \mathrm{G}_j(w) \geq n$ and all neighbours to v, w lie on P (not surprising, every vertex lies on P)

Repeat the construction from the proof of Diracs theorem, this gives us a cycle which contains vertices on the path, but this path contains all vertices, and therefore we obtain a Hamilton cycle in G_j $\square$

Anmärkning:
If we end with something Hamiltonian, we must have started with something Hamiltonian

Definition/Sats 7.7: Degree Sequence

Let G be a simple graph on n vertices. The *degree sequence* of G is a finite sequence $(d_1, \cdots, d_n)$ containing the vertex of G in non-descending order $(d_1 \leq d_2 \leq \cdots \leq d_n)$

An arbitrary sequence $(a_1, \cdots, a_n)$ is called *Hamiltonian* if all graphs with a degree sequence $d_1, \cdots, d_n$ such that $d_i \geq a_i$ for all $i = 1, \cdots, n$ are Hamiltonian

Example:
By Diracs theorem, the sequence $\left(\frac{n}{2}, \cdots, \frac{n}{2}\right)$ is Hamiltonian

Definition/Sats 7.8: Chvatal

Let $n \geq 3$
An integer sequence $a_1, \cdots, a_n$ with $0 < a_1 \leq \cdots \leq a_n \leq n$ is Hamiltonian iff for every $i < \frac{n}{2}$, we have $a_i < i \Rightarrow a_{n-i} > n - i$

Example:
The d-dimensional cube.
Fix $d \geq 2$. Let the set of vertices be strings over a binary alphabet $\{0, 1\}$ of length d and edges: two strings are adjacent if they differ in exactly position

For $d = 2$ we have: 10, 11, 00, 01
For $d = 3$ we have: 000, 001, 010, 011, 100, 101, 110, 111

Observe that all vertices in the d dimensional cube have degree d. This follows from the definition, since we have $\binom{d}{1}$ ways we can change and still be adjacent

Proposition:
The d-dimensional cube is Hamiltonian for all $d \geq 2$

Notice that we have 2^d vertices

Bevis 7.4

Let G_d be the d dimensional cube

Induction over d:

- $d = 2$ Obvious
- Assume that G_d has a Hamilton cycle using the edge from $\{0 \cdots 0, 10 \cdots 0\}$
- Consider G_{d+1}. This contains 2 copies of G_d (depending on the value of the first entry of the strings)
- Build a Hamilton cycle as follows:
  - Traverse the Hamilton cycle in the first copy from $00 \cdots 0$ to $010 \cdots 0$
  - Then go to $110 \cdots 0$
  - Traverse the Hamilton cycle in the second copy back from $110 \cdots 0$ to $10 \cdots 0$
  - Close back to $0 \cdots 0$
  - This is a Hamilton cycle in G_{d+1} containing the edge $\{0 \cdots 0, 10 \cdots 0\}$

$\square$

Anmärkning:
This can be used in computer science and goes by the term *Grey Code*

Example (*):
Petersen graph
Vertex set: Set of all 2-element subsets of $\{1, 2, 3, 4, 5\}$
Edges: Two subsets are adjacent iff they are disjoint

There are 10 vertices, and 15 edges

Proposition:
The Petersen graph is *not* Hamiltonian, however, upon removing any one vertex, the remaining graph is Hamiltonian and the Petersen graph admits a Hamilton path (a path that contains all vertices)

Bevis 7.5

We only have time to proof the first part (the other part is in the lecture notes)

We have 2 cycles in the graph that both are C_5 and 5 edges that connect the cycles together

Any Hamiltonian cycle needs to traverse each edge twice. $\square$

8. Max-flow-min-cut theorem

Definition/Sats 8.1: Directed graph

A *directed simple graph* is a pair $G = (V, E)$ where $E = \subseteq V \times V \setminus \{v, v \mid v \in V\}$

We interpret an edge v, w as an arrow pointing of v to w

Anmärkning:
In a directed graph $(v, w) \neq (w, v)$

Definition/Sats 8.2: Flow network

A *flow network* consists of a directed simple graph $G = (V, E)$ together with a weight function $w : E \to (0, \infty)$ (to every edge we assign a weight) and two distringuished vertices, a *source* s and a *sink* t

Definition/Sats 8.3: Capacity function

The function $c : V \times V \to [0, \infty)$ defined by:

$$c(v, v') = \begin{cases} w(v, v') & \text{if } (v, v') \in E \\ 0 & \text{otherwise} \end{cases}$$

is called the *capacity function* of the flow network

Anmärkning:
We do not want an edge from v to w and an edge from w to v, we can do a simplification:

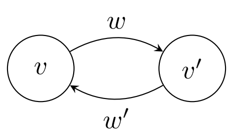

Figure 5.

Is replaced by:

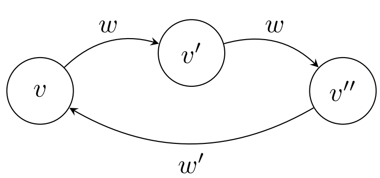

Figure 6.

Definition/Sats 8.4: Flow function

A flow f on the flow network $G = (V, E)$ with capacity function c is a function $f : V \times V \to [0, \infty)$ such that:

- **Capacity constraint** I cannot pump more water through the pipe than I have capacity
$$f(v, v') \leq c(v, v') \quad \forall v, v' \in V$$
- **Conservation constraint** However much water I pump in, I should get out (no loss and no creation out of nothing).

  For $v \in V \setminus \{s, t\}$ we have:
$$\sum_{x \in V} f(x, v) = \sum_{x \in V} f(v, x)$$

Definition/Sats 8.5: Value of flow

The *value* $|f|$ of the flow is the total out flow at the source:
$$|f| = \sum_{x \in V} f(s, x) - \sum_{x \in V} f(x, s)$$

Anmärkning:
From the conservation consraint, this difference (value) must go away somewhere and the only place it can go is in the sink.

This means that the value is also equal to the total inflow at the sink, and therefore equal to:
$$\sum_{x \in V} f(x, t) - \sum_{x \in V} f(t, x)$$

How do we know this value is positive? Well, if it is negative, then by the identity above we can swap s, t to ensure we have a positive value.

Definition/Sats 8.6: $s - t$-cut

Let $G = (V, E)$ be a flow network with source s and sink t and capacity c. An $s - t$-*cut* is a partition of V into sets S, T such that $s \in S$ and $t \in T$

Definition/Sats 8.7: Capacity of cut

The *capacity of the cut*, denoted by $c(S, T)$ and is:
$$c(S, T) = \sum_{(v, v') \in S \times T} c(v, v')$$

Anmärkning:
For any flow f and any $s - t$-cut S, T, we have that the capacity of the cut is an upperbound to the value of the flow:
$$c(S, T) \geq |f|$$

We will show that we only have equality in the most extreme case (cut capacity is as high/low as possible)

Lemma 8.1

Let f be a flow on the flow network $G = (V, E)$ and let S, T be an $s - t$ cut of G

Assume that the value of the flow is equal to the capacity of the cut:

$$|f| = c(S, T)$$

Then $|f|$ is maximal among all flows on G, and the capacity of the cut is minimal among all S, T cuts

Bevis 8.1

We need to show there is no flow with strictly larger value and no cut with strictly smaller capacity

If f' is a flow with $|f| \geq |f|$, then it needs to pass through the S, T cut as well.

Hence, $|f'| \leq c(S, T) = |f| \Rightarrow |f'| = |f|$ and $|f|$ is maximal

Similarly, any $s - t$-cut S', T' needs to satisfy $c(S', T') \geq |f| = c(S, T)$, hence if $c(S', T') \leq c(S, T)$ then $c(S, T) = c(S', T')$ and $c(S, T)$ is minimal $\square$

Definition/Sats 8.8: Residual network

Let $G = (V, E)$ be a flow network with source s and sink t

Let f be a flow on G.

The *residual network* G_f is the flow network with residual capacity c_f set as follows:

- For $u, v \in V$, set:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(v, u) & \text{if } (v, u) \in E \\ 0 & \text{otherwise} \end{cases}$$

This defines a capacity function. Let E_f be the set of pairs (u, v) for which we have some positive capacity $(c_f(u, v) > 0)$

Example:
Lets look at a flow network $G = (V, E)$ which looks as follows:

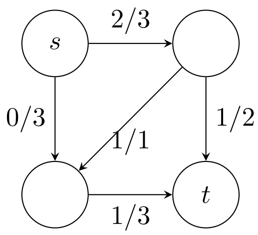

FIGURE 7.

Definition/Sats 8.9: Path

Let G_f be the residual network of a flow network G with respect to a flow f

A $v_0 - v_k$ is a sequence $v_0 e_1 \cdots e_k v_k$ where $v_0, v_1, \cdots, v_k$ are distinct vertices and all edges point in the same direction, namely $e_i = (v_{i-1}, v_i)$ for $i = 1, \cdots, k$

Definition/Sats 8.10: Augmenting path

An $s - t$-path is called an *augmenting path*

Definition/Sats 8.11: Capacity of path

We can define the capacity of a path P by $c(P) = \min_{e \in E(P)} c(e)$

Anmärkning:
We are working in the residual network, which means by definition that all our capacities are positive, and therefore $c(P) > 0$ for any augmenting path P

Lemma 8.2

Let f be a flow in G sich that G_f admits an augmenting path P

Then f', defined by:

$$f'(u,v) = \begin{cases} f(u,v) + c_f(P) & \text{if } (u,v) \text{ is an edge in } P \\ f(u,v) - c_f(P) & \text{if } (v,u) \text{ is an edge in } P \\ f(u,v) & \text{otherwise} \end{cases}$$

for all edges $(u,v) \in E$ (in the original network) is a flow on G with $|f'| = |f| + c_f(P) > |f|$

Definition/Sats 8.12: Ford-Fulkerson

Let f be a flow on the network G.
The following are equivalent:

- f is a maximum flow (the value of the flow is maximum)
- The residual network G_f contains no augmenting path
- There is an $s-t$-cut S, T with capacity $c(S,T) = |f|$
  In particular, $\max_{f \text{ flow}} |f| = \min_{S,T\ s-t-cut} c(S,T)$

Bevis 8.2

- **Third implies first**
  Already shown above
- **First implies second**
  Shown in lemma
- **Second implies third**
  Assume G_f does *not* contain an augmenting path P. This means I cant walk from s to t in the residual network while only going along the direction of the edges I have.

  Define $S = \{v \in V \mid v \text{ can be reached by a } s-v-\text{ path in } G_f\}$
  Define $T = V \backslash S$

  Then $s \in S$ and $t \in T$ (otherwise there is an augmenting path in G_f)
  In particular, these sets define an $s-t$-cut

  By construction, every arrow (u,v) in G where $u \in S$ and $v \in T$, must have its full capacity used by f.
  Otherwise, in the construction of the residual network, there would still be an arrow $\cdots$ $(u,v) \in G_f$ and $v \in S$

  Moreover, every arrow in the direction (v,u) must have flow 0 (reason is the same)

  Hence, we have that $|f| \geq c(S,T)$, but we have seen earlier that $c(S,T) \leq |f|$, which means we have equality

$\square$

Anmärkning:
This theorem makes no existence of a maximal flow, it only talks about what happens when we *do* have a maximal flow which is a bit unsatisfying.

A maxflow does however exist! Regard a flow f simply as a vector in $\mathbb{R}^{|E|}$
Then the e-th coordinate of the vector is from $[0, c(e)]$
This means that the set of all flows is closed and bounded (therefore compact in $\mathbb{R}^{|E|}$) (and therefore has a supremum)
The map $f \mapsto |f|$ is continous.

We now have a continous map on a compact set. We now know from calcul*us*, there is a f with maximum value $|f|$

What happens if all my capacities are integers?

Definition/Sats 8.13: Integer flow theorem

If G is a network with an integer-valued capacity function $c : E \to \mathbb{Z}_{>0}$, then there is an integer valued maximum flow. (It only assigns integers flow along every edge)

Algorithm [Ford-Fulkeson]
Given a flow network G, start with the flow that is 0 everywhere $f \equiv 0$

Repeat the following steps:

- Construct G_f
- Find na augmenting path P in G_f
  Use it to construct a new flow with larger value
  If no augmenting path exists, stop.

This is not really an algorithm, since it is not guaranteed to stop.
It will however terminate if all of our capacities are rational numbers.

Definition/Sats 9.1

Let $G = (V, E)$ be a finite simple graph (undirected)

A *matching* on G is a set $M \subseteq E$ such that no two edges in M share a common endpoint

We shall only consider matchings in bipartite graphs (not complete bipartite, so we do not need to have all possible edges between the two sets, but all edges are within)

Notation:
If we have some set of vertices, we denote by $N(S) = \{v \in V \mid v \text{ is adj. to some } s \in S\}$, the neighbourhood of S

Necessary condition: For A to be matched in B
$|N(Q)| \geq |Q|$ for all $Q \subseteq A$

Definition/Sats 9.2: Hall's marriage theorem

Let $G = (V, E)$ be a finite simple undirected bipartite graph with $V = A \cup B$

Then G has a matching of A into B iff $|N(Q)| \geq |Q|$ for all $Q \subseteq A$

Bevis 9.1

We will use flows. We have already discussed $\Rightarrow$ direction.

$\Leftarrow$, the task is to construct a matching if the condition is satisfied

Assume $|N(Q)| \geq |Q|$. Create a flow network as follows:

- Add two extra vertices on each side (source and sink). Connect the source to all vertices in A and the sink to all in B. All the edges flow from the source to sink
- Give the added arrows capacity of 1, and the rest capacity of something large (as close to infinity as possible)
- Specify an $s-t$-cut $S' = \{s\}, T' = \{t\} \cup A \cup B$, we have $c(S', T') = |A| < \infty$
- By the max-flow-min-cut theorem, the capacity will also be finite $c(S, T) < \infty$ for any minimum cut.
  This means that no edge (u, v) where $u \in A \cap S$ and $v \in B \cap T$ exists since if it did it would contribute to the cut.
- Then $N(S \cap A) \subseteq S \cap B$. With this, we can compute $c(S, T)$:
  $$c(S,T) = \sum_{(u,t) \in S \times T} c(u,v) = \sum_{v \in T \cap A} c(s,v) + \sum_{u \in B \cap S} c(u,t)$$
  $$= |T \cap A| + |B \cap S| \geq (|A| - |S \cap A|) + |N(S \cap A)| \geq |A| - |S \cap A| + |S \cap A| = |A|$$
- This means S', T' is a minimum cut and has capacity $|A|$
- This means there exists an integer flow of $|f| = |A|$ (max flow min cut), this flow powers one unit through each vertex of A. Following the flow gives a matchin of A into B

$\square$

Anmärkning:
The construction in the proof gives a bijection between $\{\text{matchings in } G\} \leftrightarrow \{\text{integer flows in the associated network}\}$ such that $|M| = |f|$

Corollary:
Let $G = (A \cup B, E)$ be a finite simple bipartite graph. If $|N(Q)| \geq |Q| - d$ for all $Q \subseteq A$ and some fixed $d \in \mathbb{N}$, then G contains a matching with $|A| - d$ edges

Definition/Sats 9.3: Vertex cover

Let $G = (V, E)$ be a finite simple graph. A *vertex cover* is a set $S \subseteq V$ such thtat every edge has an endpoint in S

Definition/Sats 9.4: Covering number

The covering number $\beta(G)$ is the minimum cardinality of any vertex cover of G

Anmärkning:
If we have integer flow, we have cuts. Do we have correspondance for the cuts as well? Yes we do

There is a bijection (G bipartite) between {vertex covers in G} $\leftrightarrow$ {$s-t$-cuts with finite capacity in the associated networ
such that $|Q| = c(S, T)$

Definition/Sats 9.5: König

Let G be a finite simple bipartite graph

Then, the maximum cardinality of a matching in G equals the minimum cardinality of a vertex cover $\beta(G)$

Anmärkning:
Looks a lot like max flow min cut

Bevis 9.2

Suppose we have a maximum matching $|M|$, then it corresponds through bijection to a maximum flow $|f| = |M|$. By the max flow min cut theorem, $|f| = c(S, T)$. But by another bijection, this corresponds to another minimum vertex cover $= \beta(G)$ $\square$

Anmärkning:
Set M is a set of edges

Anmärkning:
In a general graph that is not bipartite we do not have this equality
Any edge in a matching requires a separate vertex from a vertex cover. This means that the maximum cardinality of the matching $|M| \leq \beta(G)$ even in an arbitrary graph. If not bipartite, then the inequality may be strict

Definition/Sats 9.6: Perfect matching

A *perfect matching* or *1-factor* M on a finite simple graph $G = (V, E)$ is a matching on G such that every vertex $v \in V$ is an endpoint of some edge $e \in M$

Definition/Sats 9.7

Let $k \in \mathbb{N}$. A k-regular graph is a graph where every vertex has degree k

Definition/Sats 9.8: k-factor

A *k-factor* is a k-regular spanning subgraph

Anmärkning:
A Hamilton cycle in G is a 2-factor of G

Anmärkning:
In order to have perfect matching, the number of vertices must be even.

Notation:
For a set of vertices $S \subseteq V$, write $G - S = G[V \backslash S]$
If G contains a perfect matching and we obtain in $G - S$ a number of connected components on an odd number of vertices (*odd components*), what must have happened? These connected components must have at least one vertex that was matched and deleted.
Then the number $\mathrm{o}(G - S)$ of such odd components $\leq |S|$

Definition/Sats 9.9: Tuttes 1-factor theorem

A finite simple graph $G = (V, E)$ admits a perfect matching iff $\mathrm{o}(G - S) \leq |S|$ for all $S \subseteq V$

Bevis 9.3

$\Rightarrow$ is in the "Notation" section

Let $G = (V, E)$ be a graph satisfying this condition but wihtout a perfect matching. We want to show that such a graph cannot exist

Adding an edge does not affect the condition. Why? If we add this edge in such a way, it will be completely inside S or completely in one of the components and does not change the fact. What can happen is that it connectes components, odd components + odd components are even, even + odd are okay, everything is okay

We can therefore add how ever many edges. Assume WLOG that G is edge-maximal satisfying the condition and does not contain a perfect matching.

Let $n = |V|$. For $S = \emptyset$, we get $o(G - S) = 0$, so n is even

Choose $S = \{v \in V \mid deg(v) = n - 1\}$
Consider $G - S$. We get 2 cases:

- All components of $G - S$ are complete
  If this happens, find a matching on each component only leaving out one vertex each. This is possible because in a complete graph I can match whatever I want. In an even componenet I dont have ot leave out anything, it is only the odd ones I need to care about
  The left over vertices from odd components can be matched with vertices from S.
  Afterwards, an even number of vertices from S will still be unmatched. Match them up with each other.

- Let k be a component of $G - S$ that is *not* complete.

  There is a missing edge, but it is connected. In particular, this means we can have the following situation:
  There are vertices $x, a, b \in V(k)$ such that $\{x, a\}, \{a, b\}$ are edges, but $\{x, b\}$ is not

  Moreoever, $a \notin S \Rightarrow \exists$ a vertex $c \in V(G)$ not adjacent to a
  Since G was edge-maximal without a perfect matching, there exists perfect matchings as follows:
  - **M1** for $\{V, E \cup \{\{x, b\}\}\}$
  - **M2** for $(V, E \cup \{\{a, c\}\})$

In G, consider a maximum path P starting at C with an edge from M_1 and then alternating between edges from M_2 and M_1 $\square$

Definition/Sats 10.1: Higher Connectivity

Let $k \in \mathbb{Z}_{\geq 0}$. A finite simple graph $G = (V, E)$ is *k-conncted* if:
- $|V| > k$
- $G - X$ is a connected graph for all sets $X \subseteq V$ with $|X| < k$ (stays connected if less than k vertices removed)

The largest k for which G is k-connected, is called the *connectivity* of G, $\kappa(G)$

Example:
Let K_n be a complete graph on n vertices. No matter how many vertices I remove, what remains will always be connected. K_n is certainly $n-1$-connected, but not n connected because we don't have strictly $> n$ vertices. Therefore $\kappa(K_n) = n - 1$

If $\kappa(G) = 0$, then this means either that $|V| = 1$ (which is K_1), or G is disconnected since the second point in Theorem 10.1 breaks.

Anmärkning:
Every connected graph with at least 2 vertices is 1-connected.

Definition/Sats 10.2: Separation

Let $G = (V, E)$ be a finite simple graph, let $v, w \in V$ and $A, B \subseteq V$
- A set $X \subseteq V$ separates v from w if $v, w \notin X$ and every path from v to w contains a vertex in X (*see: särskiljande in automata*)
- A set X *separates* A from B if every path from A to B contains a vertex in X

Anmärkning:
$A \cap B \subseteq X$

Definition/Sats 10.3

The minimum size of a set separating v from w is denoted by $\kappa(v, w)$, suggesting this has something to do with connectivity

Definition/Sats 10.4: Vertex independent

Two path from v to w are called vertex-independent if they *do not* share a common vertex besides the end-points

Anmärkning:
When we say that 2 paths are disjoint, then we mean that the vertex set it disjoint

Anmärkning:
X separating v from w is a stronger statement than X separating $\{v\}$ from $\{w\}$

Definition/Sats 10.5

For a finite simple graph $G = (V, E)$ that is *not* complete, we have

$$\kappa(G) = \min_{v,w \in V} \kappa(v, w) \quad v, w \text{ are non-adj.}$$

Bevis 10.1

Since $G \neq K_n$, the connectivity equals the cardinality of the smallest set X whose removal disconnects G. This means, chose v_0, w_0 from different components of $G - X$. Because they get separated by X, this means $\min_{v,w \in V} \kappa(v, w) \leq \kappa(v_0, w_0) \leq |X| = \kappa(G)$

Choose v_0, w_0 such that $\kappa(v_0, w_0)$ attains the minimum value (which happens since G is finite). Then there exists $X \subseteq V$ with $|X| = \kappa(v_0, w_0)$ that separates v_0 from w_0. Hence $G - X$ is disconnected, which means that teh connectivity of G $\kappa(G) \leq |X| = \min_{v,w \in V} \kappa(v, w)$ $(v, w$ non-adjacent) by construction of X □

Definition/Sats 10.6: Menger

Let $G = (V, E)$ be a finite simple graph and let $v, w \in V$ be non-adjacent. Then: $\kappa(v, w)$ equals the maximum number of independent path from v to w.

Anmärkning:
We have a minimum of something, and claiming it is equal to the maximum of something else. We of course use the min-flow max cut theorem

Bevis 10.2

Replace every edge $\{x, y\} \in E$ by 2 directed paths with capacity ∞.
Split every vertex besides $x \in V \setminus \{v, w\}$ into vertices x_0, x_1 with arrow from $x_0 \to x_1$ with capacity 1 such that every incoming arrow to x points to x_0, and every outgoing arrow starts at x_1

v is the source, w is the sink.

Integer flows $\leftrightarrow$ vertex-independent paths frmo v to w

We also have $v - w$ cuts of finite capacity, which are just partitions (S, T) of the vertex set of the flow network

$c(S, T) = |X|$

Now use $\max |f| = \min c(S, T)$ □

Corollary: (Global version of Menger)
$G = (V, E)$ is k-conncted iff it contains k independent paths between any two vertices.

Bevis 10.3

$\Rightarrow$:

G is k-conncted means $\kappa(G) \geq k \Rightarrow \kappa(v, w) \geq k$ for all $v, w \in V$ that are non-adjacent $\overset{Mang.}{\Rightarrow}$ there exists k independent paths from v to w

For v, w adjacent, assume there are at most $k-1$ independent paths. After removing $\{v, w\} \in E$ will kill one of those paths and we get G'. There are now at most $k-2$ independent paths and v, w are non-adjacent. Now we can use Menger theorem again v, w are separated by some X with $|X| \leq k-2$. Observe that G has strictly more than k vertices(otherwise it cannot have the connectivity). There is a vertex in $y \in V \backslash X$ and $y \neq v, y \neq w$
v WLOG is separated from y by $X \Rightarrow X \cup \{w\}$. Removing $X \cup \{w\}$ separates v from y in G and removes the troublesome edge
But $|X \cup \{w\}| \leq k-1$, so $\kappa(G) \leq k-1$, which is a contradiction

$\Leftarrow$:

If there are non-adjacent vertices then by assumption and Menger, we get:

$$\kappa(G) = \min_{v,w} \kappa(v, w) \geq k$$

If we do not have any non-adjacent vertices, then $G = K_n$ with at least $n \geq k+1 \Rightarrow \kappa(G) \geq k$ $\square$

Definition/Sats 10.7: Menger V2

Let $G = (V, E)$ be a finite simple graph, $A, B \subseteq V$. Then the minimum size of a set X that separates A from B equals the maximum number of disjoint paths with one endpoint in A and the other in B

Definition/Sats 10.8

A finite simple graph is 2-connected iff it can be constructed from a cycle graph by succesively adding paths to it with both endpoints in the previously constructed graph.

Definition/Sats 10.9: Cut vertex

Let $G = (V, E)$ be a finite simple graph
A *cut vertex* whose removal increases the number of components of G

Definition/Sats 10.10: Bridge

A *bridge* is an edge whose removal increases the number of components of G

Definition/Sats 10.11: Block

A *block* is a maximal (with respect to inclusion) connected subgraph H wihtout a cutvertex in H
Blocks are either:

- Isolated vertices
- Bridges + endpoints
- Max 2-connected subgraphs

2 blocks intersect in ≤ 1 vertex (which is a cut vertex)$\Rightarrow$ any edge belongs to exactly one block

Let A be the set of cutvertices, and $\mathcal{E}$ be the set of blocks

Definition/Sats 10.12: Block-graph

The *block-graph* of G is the bipartite graph on the vertex set $A \cup B$ (A, B disjoint) where an edge is draw between $a \in A$, $\varepsilon \in \mathcal{E}$

Lemma 10.1

The block graph of a connected finite simple graph is a tree

Bevis 10.4

f the graph is connected, then the block graph is connected. The only thing we need to check is if it contains a cycle.
Since the graph is bipartite, it must be a cycle of length 4, and in particular on this cycle there must be at least 2 blocks.
This cycle can be represented by a cycle in the original graph that goes through at least 2 blocks. The problem is that this is impossible because 2 blocks overlap in one vertex (cycles are 2-connected and therefore contained in a single block) $\square$

Definition/Sats 10.13: Contraction

Let $G = (V, E)$. Look at an edge $e \in E$. The *contraction* G/e is a grpha constructed as follows:

- Replace $e = \{x, y\}$ and x, y by a vertex $v_{x,y}$ that is adjacent to all neighbours of x or y

Lemma 10.2

If $G = (V, E)$ is 3-connected and $|V| > 4$, then there exists some edge $e \in E$ such that G/e is again 3-connected

Anmärkning:
If G 3-connected $\Leftrightarrow$ can obtain K_4 from G by a sequence of edge-contractions

Definition/Sats 10.14: Tutte

A finite simple graph G is 3-connected iff there is a sequence of finite simple graphs $G_0, \cdots, G_n$ where $G_0 = K_4$ and end with $G_n = G$ and for every $i = 0, \cdots, n-1$ the graph G_{i+1} has an edge $\{x, y\}$ with $\deg_{G_{i+1}}(x)\deg_{G_{i+1}}(y) \geq 3$ and $G_i = G_{i+1}/\{x, y\}$

11. Planar Graphs

When can we draw a finite simple graph without intersecting edges?

Plane-ar graphs, suggesting vertices are points in $\mathbb{R}^2$ and edges are continous/piecewise smooth/linear curves connecting vertices.

Definition/Sats 11.1: Planar Graph

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

We say that G is *planar* if it *can* be drawn (embedded) in $\mathbb{R}^2$ such that vertices are placed at different points and no edges intersect.

Anmärkning:
Just because a graph is not planar does not mean it *cannot* be drawn in $\mathbb{R}^2$. We can rearrange and redraw.

Definition/Sats 11.2: Faces

Let G be a planar graph. Any planar embedding (a way of drawing the graph such that no edges intersect) divides the plane $\mathbb{R}^2$ into connected components called *faces*, all but one of which are bounded.

Definition/Sats 11.3: Jordans curve theorem

Any simple closed continous divides the plane into an outside region (unbounded) and an inside region (bounded)

Example:
If T is a tree, it is planar and any embedding has one face (the unbounded one).

Anmärkning:
In order to have a bounded face, there must be a cycle in the graph.

Anmärkning:
If T is a tree $\Leftrightarrow$ T connected, planar, and any embedding has 1 face.

Anmärkning:
Faces depend on *how* we embedd our graph, so why does the number even matter since it is not defined?

Definition/Sats 11.4: Eulers formula

Let $G = (V, E)$ be a connected planar graph.
Denote by f the number of faces of *some* planar embedding of G

Then, $|V| - |E| + f = 2 \Leftrightarrow 2 + |E| - |V| = f$

In particular, any two planar embedding of G has the same number of faces.

Definition/Sats 11.5: Planar dual

Let $G = (V, E)$ be a planar graph. Fix a planar embedding of G. The *planar dual* G^* is the multigraph whose vertices are the faces of the embedding and whose edge set is E.

An edge $e \in E$ connects vertices f_1, f_2 in G^* if e was part of the boundary of the faces f_1, f_2

Anmärkning:
It is possible that $f_1 = f_2$; this gives loops.

Anmärkning:
G^* depends on the embedding of G. Two different embeddings gives non-isomorphic G^*

Bevis 11.1: Eulers formula

Let $G = (V, E)$ be connected and planar. Fix an embedding of G and construct the planar dual G^*

Because G is connected, we have a spanning tree, therefore, fix a spanning tree $T = (V, E_T)$ of G and consider the spanning subgraph $T^* = (V(G^*), E \backslash E_T)$

If T^* was disconnected, then this would mean that there is a vertex in G^*, i.e a face of G completely surrounded by edges in T but then T would contain a cycle, which is not possible since T is a spanning tree.

On the other hand, if T^* contains a cycle, then this cycle needs to contain vertices in the interior, so T has a vertex inside the cycle that is disconnected from the outside, which is not possible since T is a tree (and therefore connected)

T^* is connected, and cycle free, and is spanning. We therefore have a spanning tree of G^*

T has $|V|$, and $|V| - 1$ edges (tree).
T^* has f vertices, so $f - 1$ edges

By construction, every edge in G is either in T or in $T^* \Rightarrow |E| = |E_T| + |E \backslash E_T| = |V| - 1 + f - 1 = |V| + f - 2$

$\square$

Corollary:
If G is planar on at least 3 vertices, then $|E| \leq 3\,|V| - 6$

If G is planar and bipartite on at least 3 vertices, then $|E| \leq 2\,|V| - 4$

Bevis 11.2

Place tokens/coins to each side of an edge in a planar embedding of G

You placed $2\,|E|$ coins, and at the same time at least $3f$ coins (since every cycle requires at least 3 vertices)

Therefore, $3f \leq 2\,|E|$, together with $f = 2 + |E| - |V|$ gives $|E| \leq 3\,|V| - 6$ $\square$

Anmärkning:
We do not need to assume that G is connected.

Anmärkning:
In a bipartite graph, it is essentially the same thing, except instead of $3f$ coins we have $4f$ coins

Corollary:
K_5 and $K_{3,3}$ are therefore non-planar since K_5 has 10 edges and 5 vertices, but $10 \not\leq 3 \cdot 5 - 6 = 9$ and $K_{3,3}$ has 9 edges, 6 vertices, and is bipartite but $9 \not\leq 2 \cdot 6 - 4 = 8$

Definition/Sats 11.6: Topological minor

Let G, H be finite simple graphs

A *subdivision* of H is a graph where edges of H were replaced by vertex independent paths. If G contains a subdivision of H as a subgraph, then H is a *topological minor* of G

Another way of looking at it is that the mapping of the subdivision of H onto G is injective

Definition/Sats 11.7: Minor

Let G, H be finite simple graphs.

The graph H is a *minor* if G has a subgraph G' such that H can be obtained from G' by a sequence of edge-contractions

Anmärkning:
If G contains K_5 or $K_{3,3}$ as topological minors, then G is non-planar.

The point is now, that essentially these are the only two graphs that force to be non-planar, i.e these are the only obstacles that force planarity

Definition/Sats 11.8: Kuratowski

A finite simple graph G is planar iff it does not contain K_5 or $K_{3,3}$ as topological minors.

Definition/Sats 11.9: Wagner

A finite simple graph G is planar iff it does not contain K_5 or $K_{3,3}$ as minors.

Proof outline:
Step 1

Lemma 11.1: "Wagner⇔Kuratowski"

Let G, H be finite simple graphs

- If H is a topological minor of G, then H is a minor of G
- If $K_{3,3}$ is a minor of G, H, then $K_{3,3}$ is a topological minor of G
- If K_5 is a minor of G, H, then K_5 or $K_{3,3}$ is a topological minor of G

Step 2:
"True for 3-connected graphs":

Lemma 11.2

Let $G = (V, E)$ be a finite simple 3-connected graph without K_5 or $K_{3,3}$ as minors. Then, G is planar.

Anmärkning:
This means Wegner and Kuratowski will be true for 3-connected graphs, we need to show for all connected graphs, which will be the final step.

Step 3
"Edge-maximal graphs without K_5 or $K_{3,3}$ as topological minors are 3-connected"

Lemma 11.3

Let $G = (V, E)$ be a finite simple graph with $\kappa(G) \leq 2$. Let $V_1, V_2 \subseteq V$ such that $V_1 \cap V_2$ is a *separating set* of G with $|V_1 \cap V_2| = \kappa(G)$

Set $G_i = G[V_i]$ for $i = 1, 2, \cdots$.
If G is edge-maximal without K_5 or $K_{3,3}$ as a topological minor, then so are G_1 and G_2

Then $G[V_1 \cap V_2] = K_2$. Using this, one can obtain a contradiction that this entire setup is impossible

Lemma 11.4

Let $G = (V, E)$ be a finite simple graph on at least 4 vertices. If G is edge-maximal without K_5 or $K_{3,3}$ as topological minors, then G is 3-connected

This finished proof outline. If we have a finite simple graph that does not contain K_5 or $K_{3,3}$. By step 3 we can add edges till it is 3-connected wihtout introducing them as topological minor, then step 1 says they are not minor, then step 2 says it is planar.

Bevis 11.3: Step 2

Strategy is to use induction on number of vertices. For our graph to be 3-connected, we need to have at least 4 vertices

- For $|V| = 4$, we have $G = K_4$, which we know is planar
- Assume $n := |V| \geq 5$ and that the lemma holds for all graphs on strictly less than n vertices.

  G by assumption is 3-connected, so there is an edge $e = \{x, y\} \in G$ such that G/e is 3-connected.

  If G did not K_5 or $K_{3,3}$ as a minor, then G after edge-contraction still will not contain K_5 or $K_{3,3}$ as a minor.

  We have a graph with less edges and still 3-connected. By induction hypothesis, this graph will be planar

Fix planar embedding G/e. Remove vertex $v_{x,y}$ from G/e, then it is still planar.
In the resulting embedding of G/e-$\{v_{x,y}\}$ there is a face f that contained $v_{x,y}$

Since G/e is 3-connected, removing a vertex means at most that it is 2-connected (every vertex lies in a cycle, in particular, the boundary of f is a cycle C).

Let X be the set of neighbours of the vertex $x \in G$, except for y. Y is the neighbours to y except for x. Then $X \subseteq V(C)$

Take the embedding of G/e which we know exists, and remove all edges of the form $\{v_{x,y}, w\}$ where $w \in Y \backslash X$

This gives an embedding of $G - \{y\}$.

Note that X partitions C (cycle around the face) with its vertices $x_1, \cdots, x_r$ counterclockwise into paths P_i going from x_i to x_{i+1}

We need to show that all vertices in Y lie on the same P_i:

- $y_1 \in Y \backslash X$ lies on some P_i and there is $y_2 \in Y$ not on P_i, then this is a subdivision of $K_{3,3}$ which is a contradiction since we assumed that G does not have $K_{3,3}$ as topological minors
- y, x have 3 neighbours in common, then this is a subdivision of K_5, contradiction since we assumed G does not have K_5 as topological minors
- $\deg(y) = 3$, 2 neighbours are shared with x (say $x_i \& x_k$). If x_k, x_i do'nt lie on adjoing P_i, then there is a vertex x_l, x_j between them. This forms a subdivision of $K_{3,3}$ whic is impossible.

In all other cases, they lie on the same segment on the cycle. $\square$

12. Vertex coloring

Definition/Sats 12.1: Proper k-coloring

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

A *proper k-coloring of* G is a map that sends $V \to \{1, \cdots, k\}$ such that no two adjacent vertices mapped to the same color

Definition/Sats 12.2: Chromatic number

The *chromatic number* $\chi(G)$ is the smallest integer k such that G has a proper k-coloring

Anmärkning:
We will often drop the "proper" in front of coloring.

Anmärkning:
$\chi(K_n) = n$
$\chi(C_n) = \begin{cases} 2 & n \text{ is even} \\ 3 & n \text{ is odd} \end{cases}$

Definition/Sats 12.3: Clique

The set $C \subseteq V$ is a *clique* in $G = (V, E)$ if the induced subgraph $G[C]$ is complete

Definition/Sats 12.4: Clique number

The *clique number* $\omega(G)$ is the largest k such that G has a clique of size k

Definition/Sats 12.5: Independent

A set $S \subseteq V$ is *independent* if there are no edges between any two vertices of S

Definition/Sats 12.6: Independence number

The *Independence number* $\alpha(G)$ is the maximum size of an independent set in G

Lemma 12.1

For $G = (V, E)$ (finite and simple), then we have $\chi(G) \cdot \alpha(G) \geq |V|$ and $\chi(G) \geq \omega(G)$

Bevis 12.1

It takes n colors to color a clique on n vertices. This problem does not go away if we have more than just a complete graph, so $\chi(G) \geq \omega(G)$

For the second part, any $\chi(G)$-coloring partitions the vertex set V into independent sets $V_1, \cdots, V_{\chi(G)}$, where $V_i = \{v \in V \mid v \text{ has color } i\}$

Then:

$$|V| = \sum_{i=1}^{\chi(G)} |V_i| \geq \sum_{i=1}^{\chi(G)} \alpha(G) = \chi(G)\alpha(G)$$

□

Anmärkning:
Upper bounds comes from constructing coloring.

Sometimes, we don't have our graph explicitly given, so how does one go about construct a coloring?

12.1. **Greedy coloring algorithm.**

Let $G = (V, E)$ be a finite simple graph. Suppose we have an ordering on V:

$$V = \{v_1 \prec v_2 \prec v_3 \cdots \prec v_n\}$$

And suppose we have an ordering of th eset of colors $\{1 < 2 < \cdots\}$

Take the samllest (with respect to $\prec$) vertex v_i that is not yet colored and give it the smallest color that has not been assignet to one of v_i:s already colored neighbours

So v_1 gets color 1, v_2 gets 1 *if it is not adjacent to* v_1, otherwise color 2

The problem with this algorithm is that it is really bad. However, it is also useful because we can use this to get general upper bounds on the chromatic number.

Lemma 12.2

For a finite simple graph $G = (V, E)$ with maximum degree Δ, the chromatic number satisfies

$$\chi(G) \leq \Delta + 1$$

Anmärkning:
In the case of odd cycles or complete graphs we have equality

Bevis 12.2

Take any ordering on V and run the greedy coloring algorithm.
Any vertex has at most Δ neighbours and therfore already Δ colored neighbours, hence the algorithm will need at most $\Delta + 1$ colors. □

Lemma 12.3

There exists an ordering on V such that the greedy coloring algorithm with respect to this ordering requires $\chi(G)$ colors.

Bevis 12.3

Fix a $\chi(G)$ coloring of G and order vertices according to their colors (list all the vertices with color 1, then all of color 2 and so on)

Every coloring class is an independent set. In particular, the greedy coloring algorithm will not require more than $\chi(G)$ colors. $\square$

Definition/Sats 12.7: Brooks

Let $G = (V, E)$ be a finite simple and connected graph with maximum degree Δ. Then the chromatic number $\chi(G) \geq \Delta$ unless G is complete or an odd cycle.

Lemma 12.4

Under the same assumption sas the theorem, if there exists a vertex v with a $\deg(v) < \Delta$, then $\chi(G) \leq \Delta$

Bevis 12.4

In order to prove the lemma, we employ something called *Breadth-frist search* (BFS):

- Start at v
- List the neighbours of v and mark them as discovered
- Go to the first vertex in the list, append all its yet undiscovered neighbours to the list (they are now discovered)
- Delete the first vertex of the list
- Repeat until the list is empty

In a connected graph, once the list is empty then we will have seen every vertex.

The BFS produces an ordering on V which is called the BFS-ordering

To prove the lemma, we take the inverse BFS ordering:
Let $\prec$ be the inverse BFS-ordering, say $v_n \prec \cdots \prec v_2 \prec v_1 = v$ and use the greedy coloring algorithm

For $i \geq 2$ (all vertices but the last one), the vertex v_i sees at most $\Delta - 1$ already colored neighbours, and v_1 has at most $\Delta - 1$ neighbours. Tihs means every vertex has at most $\Delta - 1$ restrictions, and we can therfore do it in $\Delta - 1 + 1 = \Delta$ colors

$\square$

Bevis 12.5: Brooks

We have laready done it for graphs that are
Let $G = (V, E)$ be a finte simple graph that is connected and $k - regular$ and also not complete or an odd cycle

Case $k = 1$
Check which graph is the only possible, realize that it is impossible

Case $k = 2$
G is n even cycle, so $\chi(G) = 2$

Assume $k \geq 3$

- $\kappa(G) = 1$. Set $G_i = \{G[\{v\} \cup V_i]\}$ where $i = 1, 2$

  Note that the graph is k-regular, so therefore $\Delta(G_1) = \Delta(G_2) = k$

  But $\deg_{G_1}(v), \deg_{G_2}(v) < k$. By the previous lemma, there exists k-colorings of G_1, G_2. All that is left to do is to glue these colorings together but we need to make sure that they have the same colors. Permute the colors to ensure that a vertex v gets color say 1 in both G_1, G_2, then glue these colorings together to get a k-coloring of G.

- $\kappa(G) = 2$ and there is a minimal separating set $\{u, v\} \subseteq V$ consisting of two vertices that are non-adjacent. This is a stronger assumption than saying that the connectivity is 2.
  $G_i = G[V_i \cup \{u, v\}]$
  If WLOG v has $\deg_{G_i}(u) \leq \Delta - 2$ for both i, then we can k-color both G_i in such a way that u, v are assigned different colors, but the same colors in G_1, G_2. THen glue the colorings.

  If this condition is not satisfied, then both u, v have deg $= 1$ in one of G_1, G_2

□

Definition/Sats 12.8: Perfect graphs

A finite simple graph $G = (V, E)$ is *perfect* if $\chi(H) = \omega(H)$ for all induced subgraphs H of G

Example:
Any graph containing an induced odd cycle of length ≥ 5 is *not* perfect. Why? Well, because if we have the subgraphs we require 3 colors to color the subgraph, so $\chi(C_5) = 3$ but $\omega(C_5) = 2$

Example:
Let G be a bipartite graph. If H contains an edge, then $\chi(H) = 2$ and $\omega(H) = 2$ since G has not triangles.

Definition/Sats 12.9: Complement

Let $G = (V, E)$ be a finite simple graph. The *complement* $\overline{G}$ of G is the simple graph
$\overline{G} = (V, \mathcal{P}_2(V) \backslash E)$

Two vertices are adjacent in $\overline{G}$ iff they were non-adjacent in G

Definition/Sats 12.10: Weak perfect graph theorem, Lovasz

A finite simple graph is perfect iff its complement is perfect

Definition/Sats 12.11: Strong perfect graph theorem, Chudnovski,Robertson, Seymour, Thomas

A finite simple graph is perfect iff it contains neither an odd cycle of length ≥ 5 nor the complement thereof as an induced subgraph

Anmärkning:
The strong perfect graph theorem $\Rightarrow$ weak perfect graph theorem

Definition/Sats 13.1

For any integer $n \geq 1$, there is a finite simple graph $G_n = (V_n, E_n)$ that does *not* contain a triangle (if it does not contain a triangle, then the largest clique number $\omega(G_n) \leq 2$) such that $\chi(G_n) = n$

Definition/Sats 13.2: Mycelskian

The *Mycelskian* of a finite simple graph $G = (V, E)$, denoted by $M(G)$ is:

- $2m + 1$ vertices $V = \{v_1, \cdots, v_m, w_1, \cdots, w_m, x\}$
- Edges: On the vertices $v_1, \cdots, v_n$, we have a copy of the original graph (keep the same edges).

  For each w_i, draw an edge between w_i and all neighbours (in G) of v_i
  Draw an edge between each w_i and x

Anmärkning:
$M(K_2) = C_5$

Lemma 13.1

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

- If G is triangle-free, then so is $M(G)$
- If $\chi(G) = k$, then $\chi(M(G)) = k + 1$

Bevis 13.1

Since $\{w_1, \cdots, w_m\}$ is an independent set in $M(G)$, any triangle in $M(G)$ can contain at most 1 vertex from that set. In particular, x is never part of a triangle

By assumption, no triangle is spanned in $\{v_1, \cdots, v_m\}$ (our original graph is assumed to be triangle-free)

The only possible way to get a triangle is to have 2 vertices from our original vertex set and one from $\{w_1, \cdots, w_m\}$

But! If we could find such triangle v_i, v_j, v_k, w_k, then w_k must be connected to v_i, v_j:s neighbour which is v_k. But this means we must have had a triangle on v_i, v_j, v_k in G, which is a contradiction. Therefore, $M(G)$ is triangle-free

For the second statement, we have to show that $M(G)$ can be colored in $k+1$ colors, and not in k colors. Therefore, suppose $\chi(M(G)) \leq k+1$. Fix a k-coloring of G. Extend it to $M(G)$ by giving each w_i the same color as v_i, now one vertex remains (x), give it the $(k+1)$-th color. This is a $(k+1)$-coloring of G

Now we show we cannot do it with fewer colors ($\chi(M(G)) \geq k+1$). Therefore, suppose $M(G)$ is k-colorable.
Then, WLOG, x has color k. This means that the vertices $w_1, \cdots, w_m$ are colored with $1, \cdots, k-1$ (since by construction, they are all neighbours to x). Because we start with an arbitrary k coloring, we cannot assume it does *not* occur in the vertices v.
Therefore, let $A \subseteq \{v_1, \cdots, v_m\}$ that are colored k. For each such v_i, change its color to the color of w_i
This is possible since A is an independent set. Moreoever, any neighbour of $v_i \in A$ is also a neighbour to w_i, then I am able to assign the same color to v_i as the one for w_i, so we get a proper k-coloring of $M(G)$ that only uses $k-1$ colors on $\{v_1, \cdots, v_m\}$

This is a problem, since by assumption, $\chi(G)$ is k, so there cannot be a $k-1$ coloring, which is a contradiction. □

Bevis 13.2

For $n = 1$:
We can simply take $G_1 = K_1$

For $n = 2$
We can simply take $G_2 = K_2$

Take any finite simple graph $G = (V, E)$. Say we have $V = \{v_1, \cdots, v_m\}$

Using Lemma 13.1, iterating M proves the theorem □

Anmärkning:
We have shown if we prevent large cliques, then maybe we can prevent a large chromatic number.

Given $k, l \in \mathbb{N}$, is there a G such that $\chi(G) > k$ and G contains no $C_3, C_4, \cdots, C_l$?

Definition/Sats 13.3: Erdös

For any integer k, there exists a finite simple graph G with $\chi(G) > k$ such that G contains no cycle of length $\leq k$

Definition/Sats 13.4: Heawood

If $G = (V, E)$ is a planar graph, then $\chi(G) \leq 5$

Bevis 13.3

We will use induction on the number of vertices $|V|$.

Case $|V| = 1$, then there is only one graph, this means $G = K_1$, and $\chi(G) = 1$ which is ≤ 5

Assume that for some $n \geq 1$, we have $\chi(G) \leq 5$ for all planar graphs on n vertices
Let G be planar on $n+1$ vertices. We need to show that this G is 5 colorable.
In G, chose a vertex of degree ≤ 5. Let $G' = G - v$. If we remove a vertex from a planar graph it will still be planar and on n vertices, which is by the induction hypothesis $\chi(G') \leq 5$

- **Case 1**: If $\deg(v) \leq 4$, then any 5-coloring of G' can be extended to G since v has degree 4 and can therefore see at most 4 colors but we have 5 at our disposal so we can chose the color we don't see

- **Case 2**: $\deg(v) = 5$. If any two neighbours of v share a color, then we can argue in the same case as case 1. Otherwise, assume the neighbours have different colors.
  Fix a planar embedding of G. Label the neighbours of v with $v_1, \cdots, v_5$ in a counterclockwise fashion (can be counterclockwise, up to the reader). WLOG, because we can always rename our colors, v_i has color i for $i = 1, \cdots, 5$

  We define $V_{1,3} = \{\text{all vertices in } G' \text{ colored 1 or 3}\}$. This vertex set gives rise to an induced subgraph $G'[V_{1,3}]$

  Then $C_{1,3}^{(1)}$ is going to be the connected component of $G'[V_{1,3}]$ that contains v_1. Similarly, $C_{1,3}^{(3)}$ is going to be the connnected component of $G'[V_{1,3}]$ that contains v_3

  If $C_{1,3}^{(1)} \neq C_{1,3}^{(3)}$, then the two components are disjoint, and we can switch colors 1 and 3 on $C_{1,3}^{(3)}$

  Then, v_3 will be colored 1, and v only sees 4 colors. We can then use the argument from case 1.

  If $C_{1,3}^{(1)} = C_{1,3}^{(3)}$, then there is a path from v_1 to v_3 using only colors 1 and 3A. Together with v , this forms a cycle C. Then the complement of this cycle in the embedding $(\mathbb{R}^2 \backslash C)$ has two connected components (we talked about this when we talked about planar graphs) and v_2 and v_4 lie in different components (faces?).

  Analogously (in the same way), construct $C_{2,4}^{(2)}$ and $C_{2,4}^{(4)}$
  Then, $C_{2,4}^{(2)}$ and $C_{2,4}^{(4)}$ are disjoint, since one of the components is inside of the cycle, and one of the components is outside. Then we can use the same trick as earlier (swap colors 2 and 4) in $C_{2,4}^{(2)}$ to get a coloring of G' where v_2 and v_4 share a color, but then we are in a setting where case 1 works.

$\square$

Anmärkning:
Why does this not work with 4 colors? We are not actually using v_5, so why not? We were using the fact that any planar graph has a vertex degree ≤ 5, we need to work around this.

Anmärkning:
Any planar graph has a vertex of degree ≤ 5

Definition/Sats 13.5: 4-color theorem, Appel & Haken

Any planar graph is 4-colorable, aka the 4-color theorem

Any minimal counterexample can be reduced to one of 1834 configurations. And for each of those configurations it was shown using computers that for each of those configurations the theorem holds

Definition/Sats 13.6: Grötzsch

Any planar graph without triangles is 3-colorable

13.1. **Spectral graph theory.**

Lemma 13.2

Let $G = (V, E)$ be a finite simple graph, and let H be an induced subgraph of G.
Denote their adjacency matrices by A_G and A_H respectively.
Then:

- $\lambda_{\min}(A_G) \leq \lambda_{\min}(A_H) \leq \lambda_{\max}(A_H) \leq \lambda_{\max}(A_G)$
- The minimum degree of G (denoted by $\delta(G)$) is $\delta(G) \leq \lambda_{\max(A_G)} \leq \Delta(G)$ where Δ is the maximum degree.

Definition/Sats 13.7: Wilf

For any finite simple graph with adjacency matrix A_G, we have that $\chi(G) \leq 1 + \lambda_{\max}(A_G)$

Anmärkning:
If the graph is regular, then $\lambda_{\max}(A_G) = \Delta(G)$

Bevis 13.4: Wilf

Among all induced subgraphs H of G, there is a minimum H (with respect to inclusion) such that $\chi(H) = \chi(G)$
Let $v \in V(H)$, then $H - v$ admits a $\chi(G) - 1$ coloring. If $\deg(v) < \chi(G) - 1$, one could extrend this coloring to a $(\chi(G) - 1)$ coloring, which contradicts the condition.

Hence, we have that the minimum degre in H is at least $\chi(G) - 1$. H is a graph with its own adjacency matrix, so $\chi(G) \leq 1 + \delta(H) \leq 1 + \lambda_{\max}(A_H) \leq \lambda_{\max}(A_G)$ □

Definition/Sats 13.8: Hoffman

For any finite simple graph G wiht $|E| \geq 1$, we have that $\chi(G) \geq \dfrac{1 + \lambda_{\max}(A_G)}{-\lambda_{\min}(A_G)}$

14.1. Edge-colorings.

Definition/Sats 14.1: Proper k-edge coloring

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

A *proper k-edge coloring* is a map $E \to \{1, \cdots, k\}$ such that no two edges with a common endpoint share the same color

Definition/Sats 14.2: Chromatic index/Edge chromatic number

Denoted by $\chi'(G)$, is the smallest integer k such that G has a proper k-edge-coloring

Anmärkning:
The chromatic index $\chi'(G) \geq \Delta(G)$

Anmärkning:
The set of all edges of the same color are *matchings* since they don't share endpoints, just like for vertices where all vertices with the same color are separated.

Definition/Sats 14.3: Königs line-coloring theorem

For every finite siple bipartite graph $G = (V, E)$ with maximum degree Δ, we have $\chi'(G) = \Delta$

Bevis 14.1

Let $G = (V, E)$ be a finite simple bipartite graph with $V = A \cup B$ where $A \cap B = \emptyset$. We will proceed using induction on the number of edges $m = |E|$:

- $m = 0$, then maximum degree is $\Delta = 0$ and there are no edges to color, so it works
- Assume the theorem holds for some $m \geq 0$ and consider G with $m + 1$ vertices.
  Let Δ be the maximum degree of G, and fix some edge $\{v, w\} \in E$. Denote by G' the graph obtained by deleting this edge $\{v, w\}$ from G

  By induction hypothesis, there is a Δ-edge-coloring of G'. In G', v, w are incident to at most $\Delta - 1$ edges, so they must see at least $\Delta - 1$ different colors, i.e there are colors i, j such that v is not incident to an edge with color i and w is not incident to an edge with color j

  If $i = j$ we can get a Δ-edge-coloring of G by giving $\{v, w\}$ the color i. So assume $i \neq j$. WLOG, $v \in A$ and $w \in B$. Consider a trail in G' starting in v and using edges with colors j and i alternatingly. Moving along the trail alternates between vertices in A and vertices in B. Moving from $A \to B$ we must be going via edges of color j and vice versa.

  Notice that no such trail can contain w since w is not incident to a k-colored edge.

  Let $E_{i,j} = \{e \in E \mid e \text{ has color } i \text{ or } j\}$
  Let $C_{i,j}^{(v)}$ be the connected component of $G' < E_{i,j} >$ that contains v

  What we have shown is that $w \notin C_{i,j}^{(v)}$, this means that in this component, we can change the coloring without affecting w. Afterwards, both v and w will not be incident to an edge of color j.

$\square$

Definition/Sats 14.4: Vizing

Let $G = (V, E)$ be a finite simple graph with max degree Δ. Then either $\chi'(G) = \Delta$ (G is *class 1*) or $\chi(G) = \Delta + 1$ (G is *class 2*)

This is all that can happen.

Anmärkning:
If K_n is a complete graph, then $\chi'(K_n) = \begin{cases} n-1 & n \text{ even} \\ n & n \text{ odd} \end{cases}$

Definition/Sats 14.5

he *line-graph* $L(G)$ of a finite simple grpah $G = (V, E)$ has vertex set E, where two vertices $e_1, e_2 \in E$ are adjacent iff they share an endpoint in G

Anmärkning:
If $H = G < S >$ for $S \subseteq E(G)$, then the line graph $L(H)$ will be $L(G)[S]$
So $L(G < S >) = L(G)[S]$, i.e edge induced subgraph of G correspond exactly to the vertex induced subgraphs of $L(G)$

Anmärkning:
Proper edge-colorings of G are precisely the proper vertex-coloring of $L(G)$. As a consequence, this means that $\chi'(G) = \chi(L(G))$

Anmärkning:
Vertices of degree d in G correspond to cliques of size d in $L(G)$. This means that the maximum degree $\Delta(G) = \omega(L(G))$

Anmärkning:
Matchings of G (edge sets that don't share an endpoint) get translated to independent sets in $L(G)$

Anmärkning:
If G is bipartite, $S \subseteq E \Rightarrow G < S >$ is bipartite
By Königs line-color theorem, this means that the chromatic index $\chi'(G < S >) = \Delta(G < S >)$, but by earlier remarks, $\chi'(G < S >) = \chi(L(G < S >)) = \omega(L(G < S >)) = \Delta(G < S >)$
But $\chi(L(G < S >)) = \chi(L(G)[S])$, and $\omega(L(G < S >)) = \omega(L(G)[S])$

However, S was chosen arbitrary, so we have that the chromatic number is equal to the clique number for all line-graphs of bipartite graphs. A name for this is that they were *perfect*

14.2. **Ramsey Theory.**

The setup is, take a complete graph K_n and color the edges either red or blue. Now, we are no-longer talking about *proper* edge colorings.

Definition/Sats 14.6

et $k \in \mathbb{N}$. The k-th Ramsey number $R(k)$ is the smallest integer n such that any red/blue coloring of K_n contains a *monochromatic* K_k

Alternatively, define $B = \{\text{blue edges}\}$. A red/blue-coloring of $K_n = (V, E)$ is uniquely corresponding to the edge-induced subgraph (V, B)

So $R(K) = n$ means n is the smallest integer such that any graph on n vertices either has a clique of size k that corresponds to the blue monochromaticity, or the independent set of size k.

There is a problem, it is not clear that these numbers n, k do exist. Maybe we can chose some exotic numbers that break this? However:

Definition/Sats 14.7: Ramsey

For any $k \geq 2$ we have $R(k) \leq 2^{2k-3}$

Bevis 14.2

n $K_{2^{2k-3}}$. Construct a vertex set $V_1, \cdots, V_{2k-2}$ together with some distringuished vertices $v_i \in V_1$ in each of these sets such that:

- $|V_i| = 2^{2k-2-i}$ for $i = 1, \cdots, 2k-2$
- $V_i \subseteq V_{i-1} \setminus \{v_{i-1}\}$ for $i = 2, \cdots, 2k-2$
- V_{i-1} has only edges of one color to all vertices in V_i for $i = 2, \cdots, 2k-2$

We have that $V_1 = V(K_{2^{2k-3}})$. Pick v_1 as you like (does not matter which we pick).

Assume we have V_{i-1}, v_{i-1} according to the points above. Then, partition $V_{i-1} \setminus \{v_{i-1}\}$ into two sets:

$$\{v \mid \{v, v_{i-1}\} \text{ is blue}\} \qquad \{v \mid \{v, v_{i-1}\} \text{ is red}\}$$

Since $|V_{i-1} \setminus \{v_{i-1}\}| = 2^{2k-1-i} - 1$, one of the two sets has at least 2^{2k-2-1} vertices. Pick v_i from this set, and $v_i \in V_i$ arbitrarily

This gives vertices $v_1, v_2, \cdots, v_{2k-3}$ and v_{2k-2}.

These vertices v_{i-1} connect to V_i with all red/blue edges. How many of those do we have? Well $2k-3$, so at least $k-1$ of them must connect to blue or red.

Pick those v_i:s together with v_{2k-2} to obtain a monochromatic K_k □

Example:
$R(1) = 1$
$R(2) = 2$
$R(3) = 6$
$R(4) = 18$
$43 \leq R(5) \leq 49$

Proposition: $R(3) = 6$
There is a red/blue-coloring of K_5 without a monochromatic triangle. This is left as an exercise to the reader

Every red/blue coloring of K_6 has a monochromatic triangle:

- K_6 has $\binom{6}{3} = 20$ triangles. Consider triplets (x, y, z) of vertices such that $\{x, y\}$ is red and $\{y, z\}$ is blue.

- **Case 1:**
  - Look at y, which sets in K_6 so it sees 5 neighbours. If all of those neighbours have the same color, how many such triplets do we have that involve y as a middle vertex? Trick question, there cannot be such! So y belongs to 0 such triplets

- **Case 2:**
  - If all neighbours of y have same color except one have same color, then y belongs to 4 triplets

- **Case 3:**
  - Three choices for one color, two choices for the other, so 6 triplets

The worst case is having case 3 at each vertex in K_6. How many triplets are there? Well, $6 \cdot 6 = 36$ triplets.