Acknowledgements: These lecture notes are partially based on the lecture notes of the Graph Theory course given by Frank de Zeeuw at EPFL in 2016. We are very thankful to Frank for sharing the TeX files with us.

1 Introduction
Lecture 1 – 23.02.2017

1 Definitions

Definition. A graph $G = (V, E)$ consists of a finite set $V$ and a set $E$ of two-element subsets of $V$. The elements of $V$ are called vertices and the elements of $E$ are called edges.

For instance, very formally we can introduce a graph like this:

$$V = \{1, 2, 3, 4\}, \quad E = \{\{1, 2\}, \{3, 4\}, \{2, 3\}, \{2, 4\}\}.$$  

In practice we more often think of a drawing like this:

Technically, this is what is called a labelled graph, but we often omit the labels. When we say something about an unlabelled graph like $\overrightarrow{2}$, we mean that the statement holds for any labelling of the vertices.

Here are two examples of related objects that in this course we do not consider graphs:

The first is a multigraph, which can have multiple edges and loops; the corresponding definition would allow the edge set and the edges to be multisets. The second is a directed graph, in which every edge has a direction; in the corresponding definition the edges would be ordered pairs instead of two-element subsets. In this course we mostly avoid these variants for simplicity, although they are certainly very useful objects. The graphs of these type appear, e.g., when trying to model Internet: represent sites (or web pages) as vertices, and links
by directed edges. Then we obtain a huge directed multigraph, which is essential for many algorithms, like Google search algorithm.

Most facts about graphs in our sense have analogues for multigraphs or directed graphs, although those are often a bit less nice. Another type of graph that we are avoiding is infinite graphs; many facts about finite graphs do not extend to infinite graphs.

Some notation: Given a graph $G$, we write $V(G)$ for the vertex set, and $E(G)$ for the edge set. For an edge $\{x, y\} \in E(G)$, we usually write $xy$, and we consider $yx$ to be the same edge. If $xy \in E(G)$, then we say that $x, y \in V(G)$ are adjacent or connected or that they are neighbors. If $x \in e$, then we say that $x \in V(G)$ and $e \in E(G)$ are incident.

**Definition (Subgraphs).** Two graphs $G, G'$ are isomorphic if there is a bijection $\varphi : V(G) \to V(G')$ such that $xy \in E(G)$ if and only if $\varphi(x)\varphi(y) \in E(G')$. A graph $H$ is a subgraph of a graph $G$, denoted $H \subseteq G$, if there is a graph $H'$ isomorphic to $H$ such that $V(H') \subseteq V(G)$ and $E(H') \subseteq E(G)$.

With this definition we can for instance say that $\square$ is a subgraph of $\exists$. As mentioned above, when we talk about graphs we often omit the labels of the vertices. A more formal way of doing this is to define an unlabelled graph to be an isomorphism class of labelled graphs. We will be somewhat informal about this distinction, since it rarely leads to confusion.

**Definition (Degree).** Fix a graph $G = (V, E)$. For $v \in V$, we write

$$N(v) = \{w \in V : vw \in E\}$$

for the set of neighbors of $v$ (which does not include $v$). Then $d(v) = |N(v)|$ is the degree of $v$. We write $\delta(G)$ for the minimum degree of a vertex in $G$, and $\Delta(G)$ for the maximum degree.

**Definition (Examples).** The following are some of the most common types of graphs.

- **Paths** are the graphs $P_n$ of the form $\overbrace{\cdot \cdot \cdot \cdot \cdot}$ . The graph $P_n$ has $n - 1$ edges and $n$ vertices; we say that $P_n$ has length $n - 1$.

- **Cycles** are the graphs $C_n$ of the form $\overbrace{\cdot \cdot \cdot \cdot \cdot}$. The graph $C_n$ has $n$ edges and $n$ vertices; the length of $C_n$ is defined to be $n$.

- The complete graphs are the graphs $K_n$ on $n$ vertices in which all vertices are adjacent. The graph $K_n$ has $\binom{n}{2}$ edges. For instance, $K_4$ is $\exists$.

- The complete bipartite graphs are the graphs $K_{s,t}$ with a partition $V(K_{s,t}) = X \cup Y$ with $|X| = s, |Y| = t$, such that every vertex of $X$ is adjacent to every vertex of $Y$, and there are no edges inside $X$ or $Y$. Then $K_{s,t}$ has $st$ edges. For example, $K_{2,3}$ is $\exists\exists$.

The following are the most common properties of graphs that we will consider.

**Definition (Bipartite).** A graph $G$ is bipartite if there is a partition $V(G) = X \cup Y$ such that every edge of $G$ has one vertex in $X$ and one in $Y$; we call such a partition a bipartition.

**Definition (Connected).** A graph $G$ is connected if for all $x, y \in V(G)$ there is a path in $G$ from $x$ to $y$ (more formally, there is a path $P_k$ which is a subgraph of $G$ and whose endpoints are $x$ and $y$).

A connected component of $G$ is a maximal connected subgraph of $G$ (i.e., a connected subgraph that is not contained in any larger connected subgraph). The connected components of $G$ form a partition of $V(G)$.
2 Basic Facts

In this section we prove some basic facts about graphs. It is a somewhat arbitrary collection of statements, but we introduce them here to get used to the terminology and to see some typical proof techniques.

Proposition 1.1. In any graph $G$ we have $\sum_{v \in V(G)} d(v) = 2|E(G)|$.

Proof. We double count the number of pairs $(v, e) \in V(G) \times E(G)$ such that $v \in e$. On the one hand, a vertex $v$ is involved in $d(v)$ such pairs, so the total number of such pairs is $\sum_{v \in V(G)} d(v)$. On the other hand, every edge is involved in two such pairs, so the number of pairs must equal $2|E(G)|$. \hfill \Box

This fact is sometimes called the “handshake lemma” because it says that at a party the number of shaken hands is twice the number of handshakes. It has useful corollaries, such as the fact that the number of odd-degree vertices in a graph must be even.

The next lemma gives a condition under which a graph must contain a long path or cycle. Note that “contains a path” means that the graph has a subgraph that is isomorphic to some $P_n$, and similarly for cycles. The proof is an example of an extremal argument, where we take an object that is extremal with respect to some property, and show that this extremality implies some other property of the object.

Proposition 1.2. A graph $G$ with minimum degree $\delta(G) \geq 2$ contains a path of length at least $\delta(G)$.

Proof. Let $v_1 \cdots v_k$ be a maximal path in $G$, i.e., a path that cannot be extended. Then any neighbor of $v_1$ must be on the path, since otherwise we could extend it. Since $v_1$ has at least $\delta(G)$ neighbors, the set \{\(v_2, \ldots, v_k\)\} must contain at least $\delta(G)$ elements. Hence $k \geq \delta(G) + 1$, so the path has length at least $\delta(G)$. \hfill \Box

Note that in general this bound cannot be improved, because $K_{\delta+1}$ has minimum degree $\delta$, but its longest path has length $\delta$. In Problem Set 1 we will prove an analogous statement for cycles.

The following lemma can be helpful when trying to prove certain statements for general graphs that are easier to prove for bipartite graphs. The lemma says that you don’t have to remove more than half the edges of a graph to make it bipartite. The proof is an example of an algorithmic proof, where we prove the existence of an object by giving an algorithm that constructs such an object.

Proposition 1.3. Any graph $G$ contains a bipartite subgraph $H$ with $|E(H)| \geq |E(G)|/2$.

Proof. We prove the stronger claim that $G$ has a bipartite subgraph $H$ with $V(H) = V(G)$ and $d_H(v) \geq d_G(v)/2$ for all $v \in V(G)$. Starting with an arbitrary partition $V(G) = X \cup Y$ (which need not be a bipartition for $G$), we apply the following procedure. We refer to $X$ and $Y$ as “parts”. For any $v \in V(G)$, we see if it has more edges to $X$ or to $Y$; if it has more edges that connect it to the part it is in than it has edges to the other part, then we move it to the other part. We repeat this until there are no more vertices $v$ that should be moved.

There are at most $|V(G)|$ consecutive steps in which no vertex is moved, since if none of the vertices can be moved, then we are done. When we move a vertex from one part to the other, we increase the number of edges between $X$ and $Y$ (note that a vertex may move back and forth between $X$ and $Y$, but still the total number of edges between $X$ and $Y$ increases
in every step). It follows that this procedure terminates, since there are only finitely many edges in the graph. When it has terminated, every vertex in $X$ has at least half its edges going to $Y$, and similarly every vertex in $Y$ has at least half its edges going to $X$. Thus the graph $H$ with $V(H) = V(G)$ and $E(H) = \{xy \in E(G) : x \in X, y \in Y\}$ has the claimed property that $d_H(v) \geq d_G(v)/2$ for all $v \in V(G)$.

The last lemma gives a characterization of bipartite graphs. An “odd cycle” is just a cycle whose length is odd. Again we give an algorithmic proof.

**Proposition 1.4.** A graph is bipartite if and only if it contains no odd cycle.

*Proof. Suppose that $G$ is bipartite with bipartition $V(G) = X \cup Y$, and that $v_1 \cdots v_kv_1$ is a cycle in $G$, with $v_1 \in X$. We must have $v_i \in X$ for all odd $i$ and $v_i \in Y$ for all even $i$. Since $v_k$ is adjacent to $v_1$, it must be in $Y$, so $k$ must be even and the cycle is not odd.

Suppose we have a connected graph $G$ which has no odd cycles. We can obtain a bipartition using the following algorithm. Start with $X,Y$ being empty sets. Pick an arbitrary vertex $x_0$ and put it in $X$. Put all the neighbors of $x_0$ in $Y$. For each $y \in Y$, put into $X$ all neighbors of $y$ that have not yet been assigned. Then for each $x \in X$, put into $Y$ all neighbors that have not yet been assigned. Keep repeating this until all vertices have been assigned. This algorithm is well-defined because no vertex is assigned more than once (thanks to the stipulation that we only consider unassigned vertices). It remains to be shown that the algorithm terminates (i.e., it does not go on endlessly), and that the resulting partition is really a bipartition of $V(G)$.

The algorithm terminates because $G$ is connected (by assumption). Indeed, this means that every $y \in Y$ has a path $yv_1 \cdots v_kx_0$ to $x_0$, and in every step at least one more vertex from this past must get assigned.

Next we show that $V(G) = X \cup Y$ is a bipartition. Suppose that two vertices $x,y$ are adjacent. By construction, there is a path $P$ from $x$ to $x_0$ that uses only edges between $X$ and $Y$, and similarly there is such a path $P'$ from $y$ to $x_0$; note that these paths may intersect, so their union might not be a path. If the edge $xy$ belongs to one of $P,P'$, then $x$ and $y$ are in different parts. Otherwise, let $x_1$ be the vertex where $P$ intersects $P'$, which is the closest to $x,y$ (this could be $x_0$).

We get a cycle $C$ from $x$ to $x_1$ to $y$ to $x$ that has even length. Therefore, the length of the path from $x$ to $x_1$ to $y$ is odd, and it has the property that all its edges are between $X$ and $Y$. Therefore, the parity of the length of the paths from $x$ to $x_1$ and from $x_1$ to $y$ are different, and thus $x,y$ lie in different parts. This shows that all edges of $G$ are between $X$ and $Y$, so $G$ is bipartite.

We did the above under the assumption that $G$ is connected. If it is not, we can apply the above to each connected component, and arbitrarily combine the bipartitions of the components to get a bipartition of $G$. \qed
1 Trees. Basic facts

Definition. A tree is a connected graph without cycles. A forest is a graph without cycles. In a tree or a forest, a vertex of degree one is called a leaf.

Proposition 2.1. Every tree with at least two vertices has a leaf.

Proof. Take a longest path $x_0x_1\cdots x_k$ in the tree (so $k \geq 1$, since the tree has at least two vertices). A neighbor of $x_0$ cannot be outside the path, since then the path could be extended. But if $x_0$ were adjacent to $x_i$ for some $i > 1$, then $x_0x_1\cdots x_ix_0$ would be a cycle. So the only neighbor of $x_0$ is $x_1$, and $x_0$ is a leaf. (Of course, the same works for $x_k$, so there are at least two leaves.)

Proposition 2.2. Any tree $T$ satisfies $|E(T)| = |V(T)| - 1$.

Proof. We use induction on the number of vertices. If $|V(T)| = 1$, then $|E(T)| = 0$. Otherwise, Proposition 2.1 gives a leaf $x_0$ of $T$. Let $T'$ be the graph obtained by removing $x_0$ and its only edge. Then $T'$ is connected, since for any $x, y \in V(T')$ there is a path from $x$ to $y$ in $T$, and this path cannot pass through $x_0$, so it is also a path in $T'$. Since $T$ has no cycles, neither does $T'$, so $T'$ is a tree. By induction we have $|E(T')| = |V(T')| - 1$, so plugging in $|E(T)| = |E(T)| - 1$ and $|V(T')| = |V(T)| - 1$ give the desired formula.

Proposition 2.3. A graph $G$ is a tree if and only if for all $x, y \in V(G)$ there is a unique path between $x$ and $y$.

Proof. First suppose we have a graph $G$ in which any two vertices are connected by a unique path. Then $G$ is certainly connected. Moreover, if $G$ contained a cycle $x_0x_1\cdots x_kx_0$, then $x_0x_k$ and $x_0x_1\cdots x_k$ would be two distinct paths between $x_0$ and $x_k$. Hence $G$ is a tree.

Suppose $G$ is a tree and $x, y \in V(G)$. Since $G$ is connected, there is at least one path from $x$ to $y$. Suppose there are two paths $P \neq P'$ from $x$ to $y$. If these paths only intersect at $x$ and $y$, we can immediately combine them into a cycle, but in general the paths could intersect in a complicated way, so we have to be careful. The paths $P$ and $P'$ could start out from $x$ being the same; let $u$ be the last vertex at which they are the same. Let $v$ be the first vertex after $u$ on $P$ that is again on $P'$. Then there is a cycle that goes along $P$ from $u$ to $v$, and then back along $P'$ from $v$ to $u$. This is a contradiction, so there is a unique path from $x$ to $y$ in $G$.

2 Distances and breadth-first search

Definition. A spanning tree of a graph $G$ is a subgraph $T$ of $G$, which is a tree with $|V(T)| = |V(G)|$.

We will now see a specific algorithm that gives a spanning tree with special properties, called a breadth-first search tree or BFS tree. This algorithm lets us answer various natural questions, like determining the distance between two vertices.
Definition. Let $G$ be a graph. For two vertices $x, y \in V(G)$, the distance $d(x, y) = d_G(x, y)$ is the length of a shortest path between $x$ and $y$.

The diameter of $G$ is the length of the longest shortest path: $diam(G) = \max_{x,y \in V(G)} d(x,y)$.

Given a graph $G$ and a subgraph $H \subset G$, we define

$$\partial(H) = \{xy \in E(G) : x \in V(H), y \not\in V(H)\}$$

for the set of edges going from vertices of $H$ to vertices not in $H$. Another bit of terminology:

To single out one vertex of a tree, we refer to it as a root.

**BFS Algorithm** (given a graph $G$ and a root $r \in V(G)$)

1. Let $T$ be the graph consisting only of $r$;
2. Iterate:
   a. Set $S = \partial(T)$;
   b. For all $xy \in S$, if $T + xy$ does not contain a cycle, replace $T$ by $T + xy$;
   c. If $\partial(T) = \emptyset$, then go to (3);
3. If $|V(T)| = |V(G)|$, return $T$; else return “disconnected”.

**Proposition 2.4.** Let $G$ be a connected graph. The BFS Algorithm lets us find the shortest path between any two vertices in $G$.

**Proof.** Let $r, s \in V(G)$ be the vertices between which we want to find the shortest path. Run the BFS Algorithm with root $r$. Give a vertex $x$ the label $f(x) = k$ if $x$ was added to $T$ in the $k$th iteration of Step 3 (and label $f(r) = 0$). We prove by induction on $f(x)$ that $f(x) = d_T(r, x) = d_G(r, x)$ for all $x$. It trivially holds for $x = r$.

If $f(x) = k$, then by construction $x$ is adjacent (in $T$ and $G$) to a vertex $y$ with $f(y) = k−1$, and by induction we have $d_T(r, y) = d_G(r, y) = k−1$. So there is a path (in $T$ and $G$) of length $k−1$ from $r$ to $y$, which gives a path (in $T$ and $G$) of length $k$ from $r$ to $x$. Moreover, there cannot be a shorter path in $G$ from $r$ to $x$, because then $d_G(r, x) < k$, so by induction we would have $f(x) = d_G(r, x) < k$, a contradiction.

Knowing that $d_T(r, x) = d_G(r, x)$, we can find the shortest path from $r$ to $x$ in $G$ by finding the shortest path in $T$. To do that, we have to keep track of the “predecessor” of each $x$ that we add in step (3); i.e., the vertex $p$ with $f(p) = f(x) − 1$ that $x$ is connected to. Then, to find the shortest path from $r$ to $s$, we start from $s$ and repeatedly take the predecessor, until we reach $r$. \qed

Aside from letting us find shortest paths, determining if $G$ is connected, and finding connected components of $G$, the BFS algorithm lets us compute distances, and it also gives us algorithms for the following tasks on $G$.

- **Compute $diam(G)$**: For each pair of vertices, we can find a shortest path and thus the distance. Do this for all pairs and take the largest distance.
- **Find a shortest cycle in $G$**: For every edge $xy$, find a shortest path between $x$ and $y$ in $G − xy$ (if it exists); combine this path with $xy$ to get a cycle. Compare all these cycles to find the shortest.
Some of these algorithms are very inefficient, but they are already much better than brute force approaches that go over all possible answers. There are all kinds of algorithms that do these tasks faster, but in this course, we don’t care too much about efficiency, and we focus on the graph-theoretical aspects (in particular, proving that the algorithms work). For more sophisticated algorithms, we suggest taking a course in Combinatorial Optimization.

The problem of finding a shortest path is more interesting if we have a weight function \( w : E(G) \rightarrow \mathbb{R}_+ \), and we want a path \( P \) for which the total weight \( \sum_{e \in P} w(e) \) is minimal. The BFS algorithm above does not do this, because it finds a path with the fewest edges, whereas a minimum-weight path may use many lightweight edges. Nevertheless, the best known algorithm for finding minimum-weight paths, known as Dijkstra’s algorithm, is based on this BFS approach.

### 3 Matchings and augmenting paths

**Definition.** Let \( G \) be a graph. A set of edges \( M \subset E(G) \) is a matching if no two edges in \( M \) share a vertex. A matching \( M \) is perfect if every vertex of \( G \) is incident with an edge in \( M \); it is maximal if there is no matching \( M' \) in \( G \) with \( M \subset M' \); it is maximum if there is no matching \( M' \) in \( G \) with \( |M| < |M'| \).

We will address the following questions: How can we find a maximum matching? How can we tell if a graph has a perfect matching? How can we check if a given matching is maximum?

First note that a maximal matching need not be maximum. Take for instance a path with three edges, and the matching consisting of the middle edge. Similarly, a greedy approach that keeps adding edges without removing any, like we used for spanning trees, would probably not lead to a maximum matching. Instead, an algorithm to find maximum matchings will need some kind of backtracking, where we throw away some edges that we previously selected. The following notion lets us do that in a smart way.

**Definition.** Given a matching \( M \) in a bipartite graph \( G \), a path is alternating if for every two consecutive edges on the path, one of the two is in \( M \). Note that any path of length 0 or 1 is alternating. An alternating path with at least one edge is augmenting if its first and last vertex are not matched by \( M \).

**Lemma 2.5.** A matching \( M \) is maximum if and only if there is no augmenting path for \( M \).

**Proof.** We prove that \( M \) is not maximum if and only if there is an augmenting path for \( M \). First suppose that there is an augmenting path \( P \) for the matching \( M \). Say \( P = v_1v_2\cdots v_k \) and the matching edges on the path are \( v_2v_3,\ldots, v_kv_{k+1},\ldots, v_{k-2}v_{k-1} \) (\( k \) must be even). Then we can augment \( M \), by removing these edges from \( M \) and replacing them by \( v_1v_2,\ldots, v_{k-1}v_{k},v_{k-1}v_k \). More formally, we replace \( M \) by \( M' = M \triangle E(P) \). Then \( M' \) is a matching since \( v_1 \) and \( v_k \) were unmatched by \( M \), and we have \( |M'| > |M| \), so \( M \) is not maximum.

Suppose \( M \) is not maximum, so there is a matching \( M' \) with \( |M'| > |M| \). Let \( D \) be the graph with \( V(D) = V(G) \) and \( E(D) = M \triangle M' \). Every vertex has degree 0, 1, or 2 in \( D \). This implies that the connected components of \( D \) are either cycles, paths, or isolated vertices. A cycle in \( D \) has the same number of edge from \( M \) as from \( M' \), so \( |M'| > |M| \) implies that there is a path \( P \) in \( D \) with more edges from \( M' \) than from \( M \). Then \( P \) must be an augmenting path for \( M \).

---

1We write \( S \triangle T \) for the symmetric difference of two sets, i.e., \( S \triangle T = (S \setminus T) \cup (T \setminus S) \).
Note that the proof of Lemma 2.5 actually works in any graph (if we define matchings in arbitrary graphs in the obvious way).

Lemma 2.5 leads to the following algorithm. It looks for an augmenting path and augments on it, until that is no longer possible. It then follows from Lemma 2.5 that the resulting matching is maximum.

Augmenting Path Algorithm to find a maximum matching in a bipartite graph $G$

with bipartition $V(G) = A \cup B$

1. Set $M = \emptyset$;
2. Iterate:
   a. Set $S = A \setminus (\cup M)$, $T = B \setminus (\cup M)$;
   b. Find any alternating path $P$ from $S$ to $T$; if none exists, go to (3);
   c. Augment along $P$ using $M := M \Delta E(P)$; go back to (2);
3. Return $M$.

It is not specified how to find alternating paths, but for bipartite graphs this is fairly simple, by considering only non-$M$-edges from $A$ to $B$, and $M$-edges from $B$ to $A$. Any alternating path from $S$ to $T$ must be of this form, since it starts from an unmatched vertex in $S$. Note that for non-bipartite graphs, it is not at all clear how to find all alternating paths, and because of this it is much harder to give an algorithm that finds maximum matchings in general graphs (even though Lemma 2.5 holds for general graphs).

3 Matchings. Stable matchings

Lecture 3 – 09.03.2017

1 Hall’s Theorem

The following theorem gives a necessary and sufficient criterion for a bipartite graph to have a perfect matching. We say that a matching matches $A \subset V(G)$ if every vertex in $A$ is contained in an edge of the matching. Given $S \subset V(G)$, we define the neighborhood of a set $S$ as $N(S) = \{v \in V(G) \setminus S : \exists u \in S$ such that $uv \in E(G)\}$.

Theorem 3.1 (Hall). Let $G$ be a bipartite graph with bipartition $V(G) = A \cup B$. Then $G$ has a matching that matches $A$ if and only if for all $S \subset A$ we have $|N(S)| \geq |S|$.

Proof. If $G$ has a matching $M$ that matches $A$, then the vertices of any $S \subset A$ are matched by $M$ to $|S|$ distinct neighbors in $B$, which implies that $|N(S)| \geq |S|$.

Suppose that $G$ has no matching that matches $A$. Let $M$ be a maximum matching and $a_0 \in A$ a vertex not matched by $M$. Let $R$ be the set of all vertices in $V(G)$ that can be reached by an alternating path that starts at $a_0$. Note that such a path must start with a
non-$M$-edge from $a_0$ to $B$, followed by an $M$-edge from $B$ to $A$, then a non-$M$-edge from $A$ to $B$, etc. We claim that $|N(R \cap A)| < |R \cap A|$.

First we show that $N(R \cap A) \subseteq R \cap B$. Suppose that $b \in B$ is connected to $a \in R \cap A$. If $ba \in M$, then the alternating path from $a_0$ to $a$ must go through $ba$, which implies that $b \in R$. If $ba \not\in M$, then the alternating path from $a_0$ to $a$ can be extended to an alternating path from $a_0$ to $b$ (unless $b$ is already on that path, in which case also $b \in R$), so again $b \in R$.

Next observe that if $b \in R \cap B$ were unmatched by $M$, then the alternating path from $a_0$ to $b$ would be an augmenting path, which by Lemma 2.5 would contradict $M$ being maximum. Thus every $b \in R \cap B$ is matched by $M$, so we have $|R \cap B| < |R \cap A|$. Combining the last two paragraphs, we have $|N(R \cap A)| \leq |R \cap B| < |R \cap A|$, contradicting the assumption that $|N(S)| \geq |S|$ for all $S \subset A$.

Note that Hall’s condition is not a convenient way of testing if a given graph has a perfect matching, because it requires checking something for all subsets of the vertex set. In practice, it is usually better to run the augmenting path algorithm. As we can see from the proof above, it will either find a maximum matching, or show that there is no perfect matching by providing a witness $(R \cap A)$ that fails Hall’s condition. However, in many theoretical applications, the condition is convenient to check; we will see several such examples in the problem sets.

2 Königs’s Theorem

We have already seen an algorithm for finding a maximum matching, but we might want a more direct method of verifying that a given matching is maximum. The following theorem gives a way to do that: If there is a vertex cover of the same size as the matching, then the matching is maximum.

**Definition.** Given a graph $G$, a vertex cover for $G$ is a set $C \subset V(G)$ such that every edge of $G$ is incident with a vertex in $C$.

Note that the “size” of a matching is the number of edges in it, while the “size” of a vertex cover is the number of vertices in it.

**Theorem 3.2** (König). Let $G$ be a bipartite graph. The maximum size of a matching equals the minimum size of a vertex cover.

**Proof.** The size of any vertex cover is at least the size of any matching, since the cover has to cover every edge of the matching with one of the endpoints of the edge. Thus the minimum size of a vertex cover is at least the maximum size of a matching.

Let $M$ be a maximum matching of $G$. We will show that there is a vertex cover $C$ with $|C| = |M|$, which will complete the proof.

Let $V(G) = A \cup B$ be a bipartition. Let $S \subset A$ be the set of vertices in $A$ that are unmatched by $M$, and let $R$ be the set of $v \in V(G)$ that can be reached by an alternating path from a vertex in $S$. Note that an alternating path from $S$ must use non-$M$-edges from $A$ to $B$, and $M$-edges from $B$ to $A$. 

\[
\begin{array}{c}
A \cap R \\
B \cap R
\end{array}
\quad
\begin{array}{c}
A \setminus R \\
B \setminus R
\end{array}
\]
Define $C = (A \setminus R) \cup (B \cap R)$. We claim that $C$ is a vertex cover for $G$. Suppose there is an edge in $E(G)$ not incident to $C$, which must be an edge $ab$ between $A \cap R$ and $B \setminus R$. If $ab$ were in $M$, then an alternating path from $S$ to $a$ would have to go through $ab$, contradicting the fact that $b \notin R$. If $ab$ were not in $M$, then extending the path from $S$ to $a$ would provide an alternating path from $S$ to $b$, contradicting $b \in B \setminus R$.

Now we prove that $|C| \leq |M|$ (which of course implies $|C| = |M|$). To prove this, we show that every vertex of $C$ is matched by a different edge of $M$. If $a \in C \cap A = A \setminus R$, then $a \notin R$, and $R$ contains the set $S$ of unmatched vertices of $A$, so $a$ must be matched. If $b \in C \cap B$, then there is an alternating path from $S$ to $b$, so $b$ must be matched, because otherwise we get an augmenting path, contradicting the assumption that $M$ is maximum. Thus $C$ is matched by $M$. Moreover, no two vertices of $C$ can be matched by the same edge of $M$, since this would have to be an edge from $B \cap R$ to $A \setminus R$, which would lead to an alternating path from $S$ to $A \setminus R$, a contradiction.

\[ \square \]

Theorem 3.2 is a special case of the duality theorem of linear programming.

3 Stable matchings

Suppose that you have $n$ students that want to do their internships in $n$ companies. Each have sent their applications to all the companies. Each student and each company has their list of preferences, and we want to pair up the students with the companies so that this assignment is stable in the following sense: there is no student and company that would both prefer to work with each other to their assigned pairs.

More formally, consider a bipartite graph $G$ with parts $A, B$, where $|A| = |B| = n$, and in which each vertex has a (strict) order of preferences for the vertices of the other part. We say that a perfect matching is stable, if there is no pair $a \in A, b \in B$, such that both of them would prefer the other to the vertex they are currently matched to.

Below we present an algorithm of Gale and Shapley, which allows to construct such a stable matching.

| The Gale-Shapley Algorithm | to find a stable matching $M$ in a complete bipartite graph $G$ with bipartition $V(G) = A \cup B, |A| = |B|$ |
|---------------------------|--------------------------------------------------|
| (1) Set $M = \emptyset$; |
| (2) Iterate: |
| (a) Take an unmatched vertex $a \in A$ and let $b \in B$ be the vertex that $a$ prefers among the ones $a$ has not tried yet. |
| (b) $a$ “proposes” to $b$: If $b$ is unmatched or $b$ is matched to $a'$, but prefers $a$ over $a'$, then “accept” $a$ and “reject” $a'$: put $M := M - a'b + ab$. Otherwise, “reject”: leave $M$ unchanged; |
| (c) If there is no more unmatched vertices in $A$ that have someone left on the list, then go to (3); |
| (3) Return $M$. |

The complexity of this algorithm is $O(n^2)$.
Proposition 3.3. The matching $M$ that the algorithm outputs is stable.

Proof. First we show that $M$ is perfect. Indeed, if there is a pair of vertices $a \in A, b \in B$, such that both are not in the matching, then $a$ must have proposed to $b$ at some point. However, if a vertex $b \in B$ is in $M$ at some step of the algorithm, then it stays in $M$.

Next, we show that the matching is stable. Assume that $ab \notin M$. Upon completion of the algorithm, it is not possible for both $a$ and $b$ to prefer each other over their current match. If $a$ prefers $b$ to its match, then $a$ must have proposed to $b$ before its current match. If $b$ accepted its proposal, but is matched to another vertex at the end, then $b$ prefers the current match of $b$ over $a$. If $b$ rejected the proposal of $a$, then $b$ was already matched to a vertex that is better for $b$.

Below we show that this algorithm always yield the same matching. We show it by proving that each $a \in A$ ends up with the best possible match in a concrete sense. We will say that $b \in B$ is a valid match of $a$, if there is a stable matching that contains the edge $ab$. We will say that $b$ is the best/worst valid match of $a$ if $b$ is a valid match, and no $b' \in B$ that ranks higher/lower than $b$ on the list of $a$ is a valid match for $a$. We use the same notation for $b \in B$.

Now, let $S^*$ denote the set of edges $\{\{a, \text{best}(a)\} : a \in A\}$. We will prove the following fact.

Proposition 3.4. The Gale-Shapley algorithm always outputs $S^*$.

Proof. We assume, by contradiction, that some execution of the G-S algorithm results in a matching $S$ in which some $a \in A$ is paired with $b \in B$ that is not its best valid match. Since $a$'s propose in decreasing order of preference, this means that some $a$ is rejected by a valid match during the execution of the algorithm. So consider the first moment during the execution in which a vertex of $A$, say $a$, is rejected by a valid match $b$. Since this is the first reject of a valid match, we must have $b = \text{best}(a)$.

Irrespectively of how the rejection happened, after this step $b$ is paired with a vertex $a'$, which is preferable for $b$. Since $b$ is a valid partner of $a$, there exists a stable matching $S'$, containing the edge $ab$. In $S'$ the vertex $a'$ is matched to $b'$. Since the rejection of $b$ by $a$ was the first rejection and $a'$ proposes in decreasing order of preference, we see that $a'$ prefers $b$ over $b'$. Therefore, in $S'$ both $a'$ and $b$ would prefer to change their match, and so $S'$ is not stable. A contradiction.

So for $a \in A$, the Gale-Shapley algorithm is ideal. Unfortunately, the same cannot be said for $b \in B$. On the exercises you are going to prove that in the stable matching $S^*$, each $b \in B$ is paired with her worst valid match.

4 Summary of matchings in arbitrary graphs

Previously we treated matchings in bipartite graphs. In this course we will not cover matchings in arbitrary graphs, mostly because it would take too much time. Nevertheless, we now give a quick overview of the basic facts about matchings in arbitrary graphs.

In an arbitrary graph, it is still true that a matching is maximum if and only if there is no augmenting path for it (the proof that we gave did not use the bipartiteness). However, the augmenting path algorithm that we described does not work, mainly because it is not clear how to find an augmenting path or determine that none exists.

Both Hall’s Theorem and König’s Theorem fail for arbitrary graphs. Take for instance the triangle $K_3$. It is 2-regular, but does not have a perfect matching. The maximum size of a matching is 1, but the minimum size of a vertex cover is 2.
There is an analogue of Hall’s Theorem for arbitrary graphs, known as Tutte’s Theorem, but it is more complicated. There is also a good algorithm due to Edmonds for finding a maximum matching in an arbitrary graph, known as the “blossom algorithm”. It does use augmenting paths, but its method for finding them is considerably more complicated.

4 Flows. Ford-Fulkerson algorithm and consequences

Lecture 4 – 16.03.2017

1 Flows

In this lecture we will think of graphs as networks along which you are able to transport a certain kind of flow. Let us be a bit more precise. Consider a graph $G = (V, E)$, where each edge has capacity: the function $c : E \to \mathbb{N}$, which assigns each edge an integer. For convenience we assign capacity 0 to all non-edges. Assume also that $G$ has two special vertices: a source $s$ and a sink $t$. The graph $G$ together with the capacity function and source and sink nodes we will call a network. We will be interested in flows in $G$. A flow $f$ in $G$ is a function from pairs of different vertices in $V$ to $\mathbb{R}$, which satisfies the following:
1. $f(v, w) = -f(w, v)$ for any $v, w \in V$ (skew symmetry);
2. $\sum_{w \in V} f(v, w) = 0$ for any $v \in V - \{s, t\}$ (flow conservation);
3. $|f(v, w)| \leq c(vw)$ for any $v, w \in V$ (capacity constraints).

Take $S \subset V$, such that $s \in S$ and $t \in \overline{S}$, where $\overline{S} := V \setminus S$. We call any such pair $(S, \overline{S})$ a cut. For any subsets $U, W$ we define
$$f(U, W) := \sum_{u \in U, w \in W} f(u, w), \quad c(S, \overline{S}) := \sum_{v \in S, w \in \overline{S}} c(vw).$$
We call $c(S, \overline{S})$ the capacity of a cut. The following proposition is intuitively clear, but we will still give a formal proof of it.

**Proposition 4.1.** For every cut $(S, \overline{S})$ we have $f(S, \overline{S}) = f(s, V - s)$.

*Proof.*
$$f(S, \overline{S}) = f(S, V) - f(S, S) = f(s, V - s) + \sum_{v \in s, V - s} f(v, V) = f(s, V - s).$$

The common value of $f(S, \overline{S})$ we call the total value of $f$ and denote by $|f|$. In this notation, for any cut $(S, \overline{S})$ we have $|f| = f(S, \overline{S}) \leq c(S, \overline{S})$. Our next goal is to show that there exists a cut and a flow on which the equality is attained.

2 Ford-Fulkerson algorithm

**Theorem 4.2** (Ford, Fulkerson). In every network the maximum total value of a flow equals the minimum capacity of a cut.
The proof of the theorem is based on the following algorithm:

**Ford-Fulkerson Algorithm** to find a flow \( f \) and a cut \((S, \bar{S})\) such that \( |f| = c(S, \bar{S}) \).

1. Set \( f = 0 \) for all edges;
2. While there is a path \( s := v_0, v_1, \ldots, v_l =: t \) from \( s \) to \( t \) such that \( f(v_{i-1}, v_i) < c(v_{i-1}, v_i) \) for all \( i = 1, \ldots, l \):
   
   (a) Find \( \epsilon := \min_{i=1, \ldots, l} \{ c(v_{i-1}v_i) - f(v_{i-1}, v_i) \} \).
   
   (b) For \( i = 1, \ldots, l \) put \( f(v_{i-1}, v_i) := f(v_{i-1}, v_i) + \epsilon \) and \( f(v_i, v_{i-1}) := f(v_i, v_{i-1}) - \epsilon \);
3. Define \( S \) as the set of all vertices that are reachable by paths from \( s \) along which the capacity of an edge is not fully used.
4. Return \( f \) and \((S, \bar{S})\).

Let us verify that the algorithm works correctly. First, note that \( f \) satisfies the three conditions in the definition of a flow at each step. Second, since in the network all capacities are integral, we have \( \epsilon \) is an integer at each step, and so \( \epsilon \geq 1 \). Therefore, since the flow must be finite, the algorithm terminates in a finite number of steps. Third, \((S, \bar{S})\) is a cut \((s \in S, t \in \bar{S})\). Finally, by the definition of the set \( S \), we have \( f(v, w) = c(vw) \) for each \( v \in S, w \in \bar{S} \), and, therefore, \( f(S, \bar{S}) = c(S, \bar{S}) \).

We note that the same algorithm and proof works for oriented graphs with different capacities for edges with the same endpoints, but going into different directions, with an obvious change in the definition of the capacity of a cut. Also, it is important that for a network with integral capacities the flow found by the algorithm is also integral.

## 2 Non-terminating example

Although the algorithm terminates in a finite number of steps for integer capacities, this may take a time proportional to the capacities even for very simple networks in case we make unlucky choices of the “augmenting paths”. We will see this on the exercise session.

In the case when the capacities are non-integral, the “naive” Ford-Fulkerson algorithm may even get stuck in an infinite loop, making smaller and smaller increments, that do not give the maximum flow in the network even in the limit. Below we we give such an example (note that this example was presented on the lecture in a slightly different way).

Choose \( \phi = \frac{\sqrt{5} - 1}{2} \), so that it satisfies \( \phi^2 = 1 - \phi \). To prove that the algorithm gets stuck in an infinite loop, we watch the residual capacities of the three horizontal edges as the algorithm progresses. The residual capacity of an edge \( xy \) in the direction from \( x \) to \( y \) is the value \( c(xy) - f(x, y) \).
Suppose the Ford-Fulkerson algorithm starts by choosing the central augmenting path, shown in the large figure above. The three horizontal edges, in order from left to right, now have residual capacities 1, 0, and $\phi$. Suppose inductively that the horizontal residual capacities are $\phi^{k-1}$, 0, $\phi^k$ for some positive integer $k$.

1. Augment along $B$, adding $\phi^k$ to the flow; the residual capacities are now $\phi^{k+1}$, $\phi^k$, 0.
2. Augment along $C$, adding $\phi^k$ to the flow; the residual capacities are now $\phi^{k+1}$, 0, $\phi^k$.
3. Augment along $B$, adding $\phi^{k+1}$ to the flow; the residual capacities are now 0, $\phi^{k+1}$, $\phi^{k+2}$.
4. Augment along $A$, adding $\phi^{k+1}$ to the flow; the residual capacities are now $\phi^{k+1}$, 0, $\phi^{k+2}$.

It follows by induction that after $4n+1$ augmentation steps, the horizontal edges have residual capacities $\phi^{2n+2}$, 0, $\phi^{2n+1}$. As the number of augmentations grows to infinity, the value of the flow converges to

$$1 + 2 \sum_{i=1}^{\infty} \phi^i = 1 + \frac{2}{1 - \phi} = 4 + \sqrt{5},$$

While the maximum total flow is $2X + 1$, provided that $X > 3$.

5 Corollaries of Ford-Fulkerson theorem.

Connectivity

Lecture 5 – 23.03.2017

1 Corollaries of Ford-Fulkerson

We start with deducing several important corollaries of the max-flow-min-cut theorem. We would use the following facts:
Fact 5.1. Ford-Fulkerson theorem works for directed graphs, with an obvious change: when counting the capacity of a cut \((S, \bar{S})\), we sum up the capacities of the edges from \(S\) to \(\bar{S}\) only.

Fact 5.2. When all capacities are integer, then the flow produced by Ford-Fulkerson algorithm is integral on every edge.

Let us first deduce Hall’s theorem.

Proof of Hall via Ford-Fulkerson. Consider the bipartite graph \(G = (A \cup B, E)\) that satisfies Hall’s conditions. Let us make a network out of \(G\). Add a source \(s\), connect it to all vertices of \(A\) by oriented edges of capacity 1. Analogously, add a sink \(t\) and connect all vertices of \(B\) to it by edges of capacity 1. Let the edges of \(G\) be oriented from \(A\) to \(B\) and their capacities to be infinite. We stress that all the edges are now oriented in the direction from \(s\) to \(t\) (see the figure below).

If there is an integer flow of value \(|A|\) in \(G\), then the edges \((x, y)\) satisfying \(x \in A, y \in B\), and \(f(x, y) = 1\) constitute a matching of \(A\) in \(G\), and we are done. Otherwise, there is a cut \((X, Y)\) of capacity \(k < |A|\). We know that \(|A \cap Y| + |B \cap X| = k < |A| = |A \cap X| + |A \cap Y|\), from which we conclude that \(|B \cap X| < |A \cap X|\). Let \(W = A \cap X\). The set \(N(W)\) is contained in \(B \cap X\), as otherwise there would be an infinite-capacity edge crossing from \(X\) to \(Y\). Thus, \(|N(W)| = |B \cap X| < |W|\), and we verified that when a perfect matching does not exist, there is a set \(W\) violating Hall’s criterion.

Next we will look at the graphs that are “more connected” than other connected graphs. We use the following notation: if \(G = (V, E)\) is a graph and \(W \subset V, U \subset E\), then \(G - W\) is the graph with vertex set \(V \setminus W\) and edge set \(E \setminus \{e \in E : e \cap W \neq \emptyset\}\), and the graph \(G - U\) is the graph with vertex set \(V\) and edge set \(E \setminus U\).

Definition. Given \(s, t \in V\), we say that two paths, both connecting \(s\) and \(t\) (\(st\)-paths) in \(G\), are internally vertex-disjoint, if they don’t share vertices, apart from \(s\) and \(t\). We say that they are edge-disjoint, if they don’t share edges.

Definition. Given a graph \(G = (V, E)\), we say that a subset \(W \subset V\) is a vertex separator, if \(G - W\) is disconnected. For \(s, t \in V\), we call \(W\) an \(s - t\) vertex separator, if \(s\) and \(t\) belong to different connected components of \(G - W\). We say that a subset \(U \subset E\) is an edge separator, if \(G - U\) is disconnected. We call a separator an \(s - t\) separator, if \(s\) and \(t\) belong to the different components of \(G - U\).

An edge cut is the set of all edges \(E[S, \bar{S}]\) between two non-empty complementary subsets \(S, \bar{S}\) of vertices. Note that any edge cut is an edge separator, but not necessarily the way around. However, any minimal edge separator is an edge cut. In what follows, we work with cuts.

The following theorem is one of the cornerstones of graph theory.
Theorem 5.3 (Menger). Let \( G = (V, E) \) be a graph and \( s, t \in V \).

1. The maximum number of edge-disjoint \( st \)-paths equals the minimum size of an \( s-t \) edge cut.

2. If \( st \notin E \), then the maximum number of vertex-disjoint \( st \)-paths equals the minimum size of an \( s-t \) vertex separator.

Proof. The edge-disjoint version follows almost immediately from Ford-Fulkerson theorem. Indeed, consider a network on \( G \) with a source \( s \) and a sink \( t \), and all edges of \( G \) having capacity 1.

We claim that, first, the maximum flow is equal to the maximum number of edge-disjoint paths. Indeed, if the maximum number of edge-disjoint paths is \( k \), then the flow is at least \( k \). In the other direction, if the maximum flow is \( k > 0 \), then there is a path between \( s \) and \( t \), and it must carry the flow of size 1. Removing the edges of this path, we decrease both the size of the flow and the number of edge disjoint paths by 1. The claim then follows by induction.

Second, we claim that the minimum cut corresponds to the minimum size of an \( s-t \) edge cut, which follows by the definition. Applying Ford-Fulkerson theorem, we get the result.

The proof of the vertex version uses an oriented version of Ford-Fulkerson theorem and a small trick concerning the construction of the network. In the network every vertex \( v \) in \( G \) is transformed into a pair of vertices \( v_{in}, v_{out} \), with \( c(v_{in}v_{out}) = 1 \) and \( c(v_{out}v_{in}) = 0 \). Every edge \( u, v \) of \( G \) is transformed into two edges with infinite capacity: \( u_{out}v_{in} \) and \( v_{out}u_{in} \). Note that the obtained graph is bipartite, with the out-vertices being on the right, in-vertices being on the left, and the only edges going from left to right are the edges of the form \( v_{in}v_{out} \) (see the figure below). Now we solve the max-flow with source \( s_{out} \) and sink \( t_{in} \). First, the flow of size \( k \) corresponds to \( k \) edge-disjoint oriented paths from the source to the sink. Second, in any finite cut we only have the edges of the form \( v_{in}v_{out} \) between the parts. Thus, it is easy to see that a cut of size \( k \) in this network corresponds to an \( s-t \) vertex separator size \( k \), and vice versa.

2  Connectivity

In this section we develop a context for Menger’s theorem, which was proven in the previous section.

Definition. A graph \( G \) is \( k \)-connected if \( |V(G)| > k \) and for every \( x_1, \ldots, x_{k-1} \in V(G) \) the graph \( G - \{x_1, \ldots, x_{k-1}\} \) is connected. The greatest integer \( k \) such that \( G \) is \( k \)-connected is the connectivity \( \kappa(G) \) of \( G \). A graph \( G \) is \( k \)-edge-connected if for every \( e_1, \ldots, e_{k-1} \in E(G) \) the graph \( G - \{e_1, \ldots, e_{k-1}\} \) is connected. The greatest integer \( k \) such that \( G \) is \( k \)-edge-connected is the edge-connectivity \( \kappa'(G) \) of \( G \).
Remark. Note that 1-connected and 1-edge-connected both mean connected.

**Theorem 5.4.** \( \kappa(G) \leq \kappa'(G) \leq \delta(G) \).

**Proof.** The edges incident to a vertex \( v \) of minimum degree form a cut; hence \( \kappa'(G) \leq \delta(G) \). It remains to show \( \kappa(G) \leq \kappa'(G) \). Suppose \( |V(G)| > 1 \) and \((S, \bar{S})\) is a minimum cut, with \( E[S, \bar{S}] \) having size \( \kappa'(G) \). If every vertex of \( S \) is adjacent to every vertex of \( \bar{S} \), then \( \kappa'(G) = |S||\bar{S}| = |S|(n-|S|) \geq n-1 \). At the same time, \( \kappa(G) \leq |V(G)|-1 \), so the inequality holds.

Hence we may assume there exists \( x \in S, y \in \bar{S} \) with \( xy \notin E(G) \). Let \( T \) be the vertex set consisting of all neighbours of \( x \) in \( S \) and all vertices of \( S - \{x\} \) that have neighbours in \( \bar{S} \) (illustrated below). Deleting \( T \) destroys all the edges in the cut \((S, \bar{S})\) (but does not delete \( x \) or \( y \)), so \( T \) is a separating set. Now, by the definition of \( T \) we can injectively associate at least one edge of the cut to each vertex in \( T \), so \( \kappa(G) \leq |T| \leq |E[S, \bar{S}]| = \kappa'(G) \).

\[ \square \]

---

### 6 Connectivity

Lecture 6 – 30.03.2017

----------

#### 1 Global version of Menger’s theorem

**Theorem 6.1** (Global Version of Menger’s Theorem). 1. A graph is \( k \)-connected if and only if it contains \( k \) internally vertex-disjoint paths between any two vertices.

2. A graph is \( k \)-edge-connected if and only if it contains \( k \) edge-disjoint paths between any two vertices.

**Proof.** Part 2 follows from Part 2 of Theorem 5.3 right away. Let us prove Part 1. If a graph \( G \) contains \( k \) internally disjoint paths between any two vertices, then \( |V(G)| > k \) and \( G \) cannot be separated by fewer than \( k \) vertices; thus, \( G \) is \( k \)-connected. Conversely, suppose that \( G \) is \( k \)-connected (and, in particular, has more than \( k \) vertices), but contains vertices \( s,t \) not linked by \( k \) internally disjoint paths. By Theorem 5.3 \( s \) and \( t \) are adjacent; put \( G_0 = G - st \). Then \( G_0 \) contains at most \( k-2 \) internally disjoint \( st \)-paths. Theorem 5.3 we can separate \( s \) and \( t \) in \( G_0 \) by a set \( X \) of at most \( k-2 \) vertices. As \( |V(G)| > k \), there is at least one further vertex \( w \notin X \cup \{s,t\} \) in \( G \). Now \( X \) separates \( w \) in \( G_0 \) from either \( s \) or \( t \) (say, from \( t \)). But then \( X \cup \{s\} \) is a set of at most \( k-1 \) vertices separating \( w \) from \( t \) in \( G \), contradicting the \( k \)-connectedness of \( G \).

\[ \square \]
One of the corollaries of Menger’s theorem is that in any 2-connected graph any two vertices lie on a cycle. In what follows, we prove the generalization of this fact. We require the following lemma.

**Lemma 6.2.** Let $G$ be a $k$-connected graph. For every $x \in V(G)$ and $U \subset V(G)$ with $|U| \geq k$, there are $k$ paths from $x$ to $U$ that are disjoint aside from $x$, with each path having exactly one vertex from $U$.

*Proof.* Add a vertex $u$ that is adjacent to all the vertices in $U$; the resulting graph is still $k$-connected (this requires that $|U| \geq k$). By Theorem 6.1, there are $k$ internally disjoint paths from $x$ to $u$. Removing $u$ from these paths gives paths from $x$ to $U$ that are disjoint aside from $x$. If such a path contains more than one vertex of $U$, then it contains a subpath with exactly one vertex from $U$, which we can take instead. □

**Theorem 6.3** (Dirac). If a graph $G$ is $k$-connected for $k \geq 2$, then for every set $K$ of $k$ vertices in $G$, there exists a cycle in $G$ containing $K$.

*Proof.* We use induction on $k$. The case $k = 2$ follows directly from the case $k = 2$ of Theorem 6.1. Assume $k > 2$ and pick any $x \in K$. By induction, $G$ has a cycle $C$ containing $K \setminus \{x\}$. If $x \in V(C)$, we are done, so we can assume that $x \notin V(C)$.

Suppose that $|V(C)| = k - 1$. By Lemma 6.2 and the fact that $G$ is $(k - 1)$-connected, there are $k - 1$ paths from $x$ to $C$ that are disjoint aside from $x$, each containing exactly one vertex of $C$. We can use any two of the paths from $x$ that end at adjacent vertices $y, z \in V(C)$ to obtain a cycle containing $x$ as well as $K \setminus \{x\}$: Remove the edge $yz$ from $C$, and replace it by the path that goes from $y$ to $x$ and then from $x$ to $z$. Since these paths were disjoint aside from $x$, and also contain no other vertices from $C$, this indeed gives a cycle.

Suppose that $|V(C)| \geq k$. By Lemma 6.2 and the fact that $G$ is $k$-connected, there are $k$ paths from $x$ to $C$ that are disjoint aside from $x$, each containing exactly one vertex of $C$. The $k - 1$ vertices of $K \setminus \{x\}$ partition $V(C)$ into $k - 1$ consecutive segments (for each vertex of $K \setminus \{x\}$, take that vertex and all the following ones, up to the one before the next vertex of $K \setminus \{x\}$). By the pigeonhole principle, there must be a segment that is reached by two of the paths from $x$. Then as in the previous case these two paths can be used to obtain a cycle containing $x$ as well as $K \setminus \{x\}$. □

### 2. Mader’s Theorem

If a graph has high minimum degree, it does not imply that the graph will be highly connected. However, the following statement is true. Let $d(G) = \frac{1}{|V(G)|} \sum_{v \in V(G)} d(v)$ be the average degree of $G$. Then:

**Theorem 6.4** (Mader, 1972). Every graph of average degree at least $4k$ has a $k$-connected subgraph.

*Proof.* For $k \in \{0, 1\}$ the assertion is trivial; we consider $k \geq 2$ and a graph $G = (V, E)$ with $|V| = n$ and $|E| = m$. For inductive reasons it will be easier to prove the stronger assertion that $G$ has a $k$-connected subgraph whenever

1. $n \geq 2k - 1$ and
2. $m \geq (2k - 3)(n - k + 1) + 1$. 

18
(This assertion is indeed stronger, i.e. (i) and (ii) follow from our assumption of $\bar{d}(G) \geq 4k$: (i) holds since $n > \Delta(G) \geq 4k$, while (ii) follows from $m = \frac{1}{2}\bar{d}(G)n \geq 2kn$.)

We apply induction on $n$. If $n = 2k - 1$, then $k = \frac{1}{2}(n + 1)$, and hence

$$m \geq (n - 2) \frac{n + 1}{2} + 1 = \frac{1}{2}n(n - 1)$$

by (ii). Thus $G = K_n \supseteq K_{k+1}$, proving our claim. We therefore assume that $n \geq 2k$. If $v$ is a vertex with $d(v) \leq 2k - 3$, we can apply the induction hypothesis to $G\setminus v$ and are done. So we assume that $\delta(G) \geq 2k - 2$. If $G$ is itself not $k$-connected, then there is a separating set $X \subseteq V$ with less than $k$ vertices, such that $G\setminus X$ has two components on the vertex sets $V_1, V_2$. Let $G_i = G [V_i \cup X]$, so that $G = G_1 \cup G_2$, and every edge of $G$ is either in $G_1$ or $G_2$ (or both). Each vertex in each $V_i$ has at least $\delta(G) \geq 2k - 2$ neighbours in $G$ and thus also in $G_i$, so $|G_1|, |G_2| \geq 2k - 1$. Note that each $|G_i| < n$, so by the induction hypothesis, if no $G_i$ has a $k$-connected subgraph then each

$$e(G_i) \leq (2k - 3)(|G_i| - k + 1).$$

Hence,

$$m \leq e(G_1) + e(G_2) \leq (2k - 3)(|G_1| + |G_2| - 2k + 2) \leq (2k - 3)(n - k + 1) \quad \text{(since } |G_1 \cap G_2| \leq k - 1),$$

contradicting (ii).

\[ \square \]

7 Hamiltonian cycles

Lecture 7 – 06.04.2017

1 Girth and circumference

Definition. The girth $\text{gir}(G)$ of a graph $G$ is the length of the shortest cycle contained in $G$. The circumference $\text{circ}(G)$ is the length of the longest cycle contained in $G$.

The complete graph $K_n$ for $n \geq 3$ has $\text{gir}(K_n) = 3$ and $\text{circ}(K_n) = n$. For a less obvious example, consider the Petersen graph. It clearly has a 5-cycle, and it is pretty easy to check that it has no 3-cycle or 4-cycle, so it has girth 5. Determining the circumference is more challenging. With a little effort one can find a 9-cycle, like the one depicted below.

![Diagram of the Petersen graph](image-url)
The proof that the Petersen graph does not contain a 10-cycle you may see on the slides.

Recall that in the lecture on trees we saw an algorithm for finding the shortest cycle, so we have an algorithm to determine the girth of a graph. The algorithm works by finding, for each edge $xy \in E(G)$, the distance from $x$ to $y$ in $G - xy$, using a breadth-first search tree. The shortest such distance, plus one, is the girth. This works because, given a shortest cycle, if we remove an edge $xy$ from the cycle, then the remaining path must be the shortest path between $x$ and $y$.

However, determining the circumference is NP-hard, so we do not have a fast algorithm for it. The example of the Petersen graph already illustrates this. In general, a BFS tree gives an easy way to find a shortest path, because it has the special property that a shortest path between two vertices in the BFS tree is also the shortest path in the whole graph. But we do not have a tree or other structure with a similar property for longest paths.

2 Hamiltonian cycles

Definition. A Hamilton cycle in a graph $G$ is a cycle that contains all vertices of $G$. A Hamilton path in a graph $G$ is a path that contains all vertices of $G$.

A graph has a Hamilton cycle if and only if $\text{circ}(G) = |V(G)|$. Just like determining the circumference, it is NP-hard to find a Hamilton cycle in a graph (or determine that there is none). A related problem is to find a shortest Hamilton cycle in a graph with weighted edges; this is called the travelling salesman problem and is one of the most famous NP-hard problems.

Although we have no general algorithm for finding Hamilton cycles, we can still prove some theorems that are useful in certain situations.

2.1 Necessary condition

Given a graph $G$ and a set $S \subseteq V(G)$ of vertices, we write $G - S$ for the graph obtained by removing the vertices of $S$ from $G$, along with all the edges that are incident to vertices in $S$.

Lemma 7.1. If $G$ has a Hamilton cycle, then for all $S \subseteq V(G)$, $G - S$ has at most $|S|$ connected components.

Proof. The Hamilton cycle must visit all the components of $G - S$ (viewed as subgraphs of $G$), and to get from one component to another the cycle must pass through a vertex of $S$. Thus every component is connected to $S$ by two edges of the cycle (and possibly by other edges not in the cycle). Since every vertex is incident to two edges of the cycle, we have that twice the number of components is at most twice the number of vertices of $S$.

This lemma can be useful to show that a graph does not have a Hamilton cycle. For example, if in the left-hand graph $G$ below we let $S$ consist of the middle two vertices, then $G - S$ has three connected components, so by Lemma 7.1 the graph has no Hamilton cycle.
On the other hand, one can check that the right-hand graph $H$ satisfies the condition that for all $S \subset V(H)$, $H - S$ has at most $|S|$ components. Nevertheless, the graph has no Hamilton cycle. To see this, observe that for the vertices of degree 2, both incident edges would have to be in the cycle; but then the middle vertex would be incident to three edges of the cycle, which is impossible.

2.2 Sufficient conditions

Next we prove two sufficient conditions for a graph to have a Hamilton cycle. First we show that a graph with many edges must have a Hamilton cycle. However, note that this bound is surprisingly weak, because a graph with this many edges is almost complete. In the proof the following definition will be convenient.

Definition. The complement of a graph $G$ is the graph $\overline{G}$ with vertex set $V(\overline{G}) = V(G)$ and edge set $E(\overline{G}) = \{xy : x, y \in V(G), xy \notin E(G)\}$.

Theorem 7.2. If $G$ is a graph with $|E(G)| > \left(\frac{|V(G)| - 1}{2}\right) + 1$, then $G$ has a Hamilton cycle.

Proof. Set $n = |V(G)|$. The statement is clearly true for $n = 1, 2, 3$, so we assume $n > 3$. Note that $\binom{n-1}{2} + 1 = \binom{n}{2} - (n - 2)$. Thus the condition of the theorem means that $|E(\overline{G})| < n - 2$. Thus $\sum d(\overline{v}) = 2(n - 2) < 2n$, which implies that there must be a vertex $v$ such that $d(\overline{v}) \leq 1$. Then we have $d_G(v) \geq n - 2$. We remove the vertex $v$ from $G$, and we will apply induction to $G - v$. We distinguish the two cases $d(v) = n - 2$ and $d(v) = n - 1$.

Suppose $d(v) = n - 2$. Then

$$|E(G - v)| = |E(G)| - (n-2) > \left(\binom{n-1}{2}\right) + 1 - (n-2) = \left(\binom{n}{2}\right) + 1 = \left(\frac{|V(G - v)| - 1}{2}\right) + 1.$$

Hence, by induction, the graph $G - v$ has a Hamilton cycle $C$. Since $d(v) = n - 2$ and $n > 3$, $v$ must have two neighbors $u, w$ that are adjacent on $C$. Then we can remove $uw$ from $C$ and replace it by $uv$ and $vw$, which results in a Hamilton cycle for $G$.

Suppose $d(v) = n - 1$. In this case we only have $|E(G - v)| > \left(\frac{|V(G - v)| - 1}{2}\right)$, so we cannot apply induction right away. If $G - v$ is complete, then $G - v$ has a Hamilton cycle, and we can add $v$ as in the previous case. Otherwise, we can add an arbitrary edge $e$ to $G - v$, and apply induction to find a Hamilton cycle $C$ in $G - v + e$. If $C$ does not contain $e$, then we can again add $v$ as in the previous case. If $C$ does contain $e$, then removing $e$ from $C$ gives a “Hamilton path” $P$ in $G - v$. Since $d(v) = n - 1$, $v$ is connected to all vertices of $G - v$, and in particular to the endpoints $u, w$ of $P$. Then adding $uv$ and $vw$ to $P$ gives a Hamilton cycle of $G$.

The statement in Theorem 7.2 cannot be improved, in the sense that a weaker bound on $|E(G)|$ does not imply a Hamilton cycle. Take for instance the graph $G$ consisting of $K_{n-1}$
and a single vertex connected to a single vertex of $K_{n-1}$. This graph has $|E(G)| = \binom{n-1}{2} + 1$, but it has no Hamilton cycle, since it has a vertex of degree 1.

As said, the condition in Theorem 7.2 is somewhat weak in the sense that many graphs that have a Hamilton cycle do not satisfy the condition. The following sufficient condition does better by looking at the minimum degree instead of the total number of edges. Its proof is a typical extremal argument that we have seen before, for instance in the proof of a lemma in Lecture 1, which said that a graph must have a cycle of length $\delta(G) + 1$. But note that that lemma by itself is not strong enough to imply a Hamilton cycle.

Theorem 7.3 (Dirac). Let $G$ be a graph with $|V(G)| \geq 3$. If $\delta(G) \geq \frac{1}{2}|V(G)|$, then $G$ has a Hamilton cycle.

Proof. First observe that $G$ must be connected, since otherwise each connected component would contain at least $\delta(G) + 1 > \frac{1}{2}|V(G)|$ vertices, which is impossible.

Take a longest path $P = x_1x_2\cdots x_k$ in $G$. By maximality, all neighbors of $x_1$ and $x_k$ are on the path. Thus $\delta(G) \geq \frac{1}{2}|V(G)|$ gives the following two inequalities:

$$|\{x_i : 1 \leq i \leq k-1, x_ix_k \in E(G)\}| \geq \frac{1}{2}|V(G)|,$$
$$|\{x_i : 1 \leq i \leq k-1, x_{i+1}x_1 \in E(G)\}| \geq \frac{1}{2}|V(G)|.$$

In other words, we have two subsets of size at least $\frac{1}{2}|V(G)|$ that are contained in the set $\{x_1, \ldots, x_{k-1}\}$, which has $k-1 < |V(G)|$ elements. It follows that the two subsets share an element $x_i$, which means that we have $x_ix_k \in E(G)$ and $x_{i+1}x_1 \in E(G)$. Then $C = x_i \cdots x_1x_{i+1}\cdots x_kx_i$ is a cycle.

In fact, $C$ is a Hamilton cycle. Indeed, suppose there is a vertex $u$ not in $C$. Since $G$ is connected, there is a path from $u$ to (say) $x_1$. There is a vertex $v$ on this path that is not on $C$ but that is adjacent to some $x_j$. Then there is a path that goes from $v$ to $x_j$, then all around the cycle $C$ to a neighbor of $x_j$. This path contains $k + 1$ vertices, contradicting the maximality of $P$. \qed

This theorem is again best possible, in the sense that a weaker bound on the minimum degree would not imply a Hamilton cycle. Take for instance the graph $G$ consisting of two copies of $K_k$ sharing a single vertex. This graph has $n = 2k-1$ vertices and minimum degree $\delta(G) = k-1 = \frac{1}{2}|V(G)| - \frac{1}{2}$, but no Hamilton cycle.

---

**8 More on Hamiltonian cycles. Euler tours**

Lecture 8 – 13.04.2017

---

1 ANOTHER SUFFICIENT CONDITION

Recall the following definition.

**Definition.** Let $G$ be a graph. A vertex set $I \subseteq V(G)$ is called independent in $G$ if no two vertices of $I$ are connected by an edge of $G$. The independence number $\alpha(G)$ is the size of the largest independent set in $G$. 

---

22
Theorem 8.1 (Chvátal, Erdős). Let \( G \) be a graph with \( |V(G)| \geq 3 \). If \( \kappa(G) \geq \alpha(G) \), then \( G \) has a Hamilton cycle.

Proof. Let \( k = \kappa(G) \), and let us take a longest cycle \( C \) in \( G \). We know that \( \delta(G) \geq \kappa(G) = k \) and that any graph \( G \) contains a cycle of length at least \( \delta(G) \), so \( C \) must have length at least \( k \). We will prove that \( C \) is Hamiltonian.

Suppose not. Then there is a vertex \( v \in G \) not contained in \( C \). Let us take a maximal collection of \( v \)-\( C \) paths that are disjoint apart from their starting vertex \( v \) (and each path contains exactly one vertex of \( C \)). By Lemma 6.2 we know that it contains at least \( k \) paths. Let \( v_1, \ldots, v_\ell \in C \) be the other endvertices of the paths (so here \( \ell \geq k \)). For \( i = 1, \ldots, \ell \), let \( u_i \) be the vertex immediately following \( v_i \) on \( C \) in the clockwise direction. Note that the \( u_i \) are all different from the \( v_i \). Indeed, if \( u_i = v_j \) then \( v_i v_j \) is an edge of \( C \). But then we can replace the edge \( v_i v_j \) in \( C \) by the paths \( v_i v \) and \( v v_j \) from the collection and get a longer cycle, contradicting our assumption (see figure on the right).

Now by the maximality of our path collection, we know that \( v_1, \ldots, v_\ell \) include all the neighbors of \( v \) in \( C \). In particular, \( v \) is not adjacent to any of the \( u_i \). The set \( \{v, u_1, \ldots, u_n\} \) has size \( \ell + 1 > k \), so by assumption it cannot be independent. \( G \) therefore contains an edge \( u_i u_j \). But then we can remove the edges \( v_i u_i \) and \( v_j u_j \) from \( C \) and replace them by the edge \( u_i u_j \) and the paths \( v_i v \) and \( v v_j \) from our collection (see figure on the left). Again, we get a longer cycle, and this contradiction shows that \( C \) must contain all vertices of \( G \). \qed

2 Tournaments

Definition. A tournament is a directed graph that has exactly one (oriented) edge between any two vertices.

For example, if we take a complete graph and give each edge an orientation then we get a tournament. What is the longest directed path in such a graph? The following statement was a homework exercise:

Theorem 8.2. Every tournament contains a directed Hamilton path.

Of course, not every tournament contains a directed Hamilton cycle. In fact, there is a tournament that does not contain any cycle, whatsoever: the so-called transitive tournament on vertex set \( \{1, \ldots, n\} \) where each edge is oriented from the smaller endpoint to the larger one. But can we find conditions for the existence of a Hamilton cycle?

A first guess might be that it is enough if every vertex has both an incoming and an outgoing edge (this is clearly required). However, this is not sufficient, as shown by the following graph.
There is, however, a fairly simple necessary and sufficient condition.

**Definition.** A directed graph is **strongly connected** if for any two vertices \( u \) and \( v \), there is a directed path from \( u \) to \( v \) (and vice versa).

We use \( u \rightarrow v \) to denote that there is an edge \( uv \) directed from \( u \) to \( v \).

**Theorem 8.3.** A tournament \( T \) has a Hamilton cycle if and only if it is strongly connected.

**Proof.** If \( T \) is Hamiltonian, then it is clearly strongly connected: for any \( u \) and \( v \) the Hamilton cycle contains a path from \( u \) to \( v \).

Now, suppose \( T \) is not Hamiltonian. Let \( C \) be a longest directed cycle in \( T \) and take a \( v \notin C \). If \( C \) has 2 consecutive vertices \( u, u' \) such that \( u \rightarrow v \) and \( v \rightarrow u' \), then there is a longer cycle on the vertex set \( C \cup \{v\} \):

![Diagram showing a cycle with vertex set \( C \cup \{v\} \).]

Otherwise all edges between \( v \) and \( C \) go in only one direction. By our the assumption on \( C \), this must hold for all \( v \notin C \). Let \( A \) be the set of \( v \notin C \) such that edges go from \( v \) to \( C \), and let \( B \) be the set of \( v \notin C \) such that edges go from \( C \) to \( v \). If one of \( A \) or \( B \) is empty, then \( T \) cannot be strongly connected. For example, if \( B \) is empty then there is no path from any vertex of \( C \) to any vertex of \( A \).

So, suppose both of \( A \) and \( B \) are nonempty. If there was an edge oriented from some vertex \( b \in B \) to a vertex \( a \in A \) then we could extend \( C \), contradicting maximality (we could replace any edge \( x \rightarrow y \) in \( C \) with the path \( x \rightarrow b \rightarrow a \rightarrow y \)). So all edges between \( A \) and \( B \) are directed from \( A \) towards \( B \), but then there is no path from any vertex in \( B \) to any vertex in \( A \), so again \( T \) is not strongly connected.

![Diagram showing directed edges from \( A \) to \( B \) and \( B \) to \( A \) to illustrate directed graph with no strong connection.]

Note that no such description for Hamiltonicity is known for graphs. As mentioned earlier, it is algorithmically hard to decide whether or not a given graph contains a Hamilton cycle. For tournaments this is not the case. For example, one can turn the above proof into a relatively fast algorithm. In some sense, this shows that tournaments are simpler structures than graphs.

### 3 Euler tours

**Definition.** A trail is a walk with no repeated edges.
**Definition.** An Eulerian trail in a (multi)graph \( G = (V, E) \) is a walk in \( G \) passing through every edge exactly once. If this walk is closed (starts and ends at the same vertex) it is called an Eulerian tour.

This notion originates from the “seven bridges of Königsberg” problem – originally solved by Euler – that asked if it was possible to walk through all the seven bridges of Königsberg in one go without crossing any of them twice.

This question can be turned into a graph problem asking for an Euler trail. Euler solved the problem by noticing that the existence of Euler trails is closely related to the degree parities.

**Theorem 8.4.** A connected (multi)graph has an Eulerian tour if and only if each vertex has even degree.

The proof of this theorem is based on the following simple lemma.

**Lemma 8.5.** In a graph where all vertices have even degree, every maximal trail is a closed trail.

**Proof.** Let \( T \) be a maximal trail. If \( T \) is not closed, then \( T \) has an odd number of edges incident to the final vertex \( v \). However, as \( v \) has even degree, there is an edge touching \( v \) that is not contained in \( T \). This edge can be used to extend \( T \) to a longer trail, contradicting the maximality of \( T \).

**Proof of 8.4.** To see that the condition is necessary, suppose \( G \) has an Eulerian tour \( C \). If a vertex \( v \) was visited \( k \) times in the tour \( C \), then each visit used 2 edges incident to \( v \) (one incoming edge and one outgoing edge). Thus, \( d(v) = 2k \), which is even.

To see that the condition is sufficient, let \( G \) be a connected graph with even degrees. Let \( T = e_1e_2\ldots e_\ell \) (where \( e_i = (v_{i-1}, v_i) \)) be a longest trail in \( G \). Then it is maximal, of course. According to the Lemma, \( T \) is closed, i.e., \( v_0 = v_\ell \). \( G \) is connected, so if \( T \) does not include all the edges of \( G \) then there is an edge \( e \) outside of \( T \) that touches it, i.e., \( e = uv_i \) for some vertex \( v_i \) in \( T \). Since \( T \) is closed, we can start walking through it at any vertex. But if we start at \( v_i \) then we can append the edge \( e \) at the end: \( T' = e_{i+1}\ldots e_\ell e_1e_2\ldots e_i e \) is a trail in \( G \) which is longer than \( T \), contradicting the fact that \( T \) is a longest trail in \( G \). Thus, \( T \) must include all the edges of \( G \) and so it is an Eulerian tour.

**Corollary 8.6.** A connected multigraph \( G \) has an Eulerian trail if and only if it has either 0 or 2 vertices of odd degree.
Proof. Suppose $T$ is an Eulerian trail from vertex $u$ to vertex $v$. If $u = v$ then $T$ is an Eulerian tour and so by [8.4] it follows that all the vertices in $G$ have even degree. If $u \neq v$ then let us add a new edge $e = uv$ to $G$. In this new multigraph $G \cup \{e\}$, $T \cup \{e\}$ is an Euler tour. By [8.4] we see that all the degrees in $G \cup \{e\}$ are even. This means that in the original multigraph $G$, the vertices $u, v$ are the only ones that have odd degree.

Now we prove the other direction of the corollary. If $G$ has no vertices of odd degree then by [8.4] it contains an Eulerian tour which is also an Eulerian trail. Suppose now that $G$ has 2 vertices $u, v$ of odd degree. Then add a new edge $e$ to $G$. Now all vertices of the resulting multigraph $G \cup \{e\}$ have even degree, so, by [8.4] it has an Eulerian tour $C$. Removing the edge $e$ from $C$ gives an Eulerian trail of $G$ from $u$ to $v$. 

9 Planar graphs
Lecture 9 – 27.04.2017

1 Drawings of graphs

We have so far only considered abstract graphs, although we often used pictures to illustrate the graph. In this lecture, we prove some facts about pictures of graphs and their properties. To set this on a firm footing, we give a formal definition of what we mean by “picture”, although in most of the lecture we will be less formal.

Definition. A drawing of a graph $G$ consists of an injective map $f : V(G) \to \mathbb{R}^2$, and a curve $C_{xy}$ (the image of an injective continuous map $[0,1] \to \mathbb{R}^2$) from $x$ to $y$ for every $xy \in E(G)$, such that if for $x, y, z \in V(G)$ we have $z \neq x, y$ then $f(z) \notin C_{xy}$. A drawing is planar if the curves $C_{xy}$ do not intersect each other except possibly at endpoints.

Definition. A graph is planar if it has a planar drawing.

For example, the graphs $K_4$ and $K_{2,3}$ are planar graphs.

Warning: Some proofs in this lecture will be not quite rigorous. This is mainly because arbitrary continuous curves can behave in unintuitive ways, which makes it hard to make intuitive arguments fully rigorous. Whenever an argument is not quite rigorous, we will point out why. In most cases, the arguments can be made rigorous using the Jordan Curve Theorem, which states that a non-self-intersecting closed curve in $\mathbb{R}^2$ divides $\mathbb{R}^2$ into an inside and an outside, and any path between a point inside and a point outside must pass through the closed curve. Although this statement is very intuitive, proving it turns out to be quite hard. We will not try to prove it in this course, since it would take us too far into topology and away from graph theory.

Definition. A face of a drawing $D$ of a graph $G$ is a maximal connected set in $\mathbb{R}^2$ after the vertices and edges of $D$ are removed. We write $F_D(G)$ for the set of faces of $D$. 

26
Every drawing has an outer face, which is the unique unbounded face in the complement of the drawing. Note that although typically an edge lies in the boundary of two faces, an edge can be in the boundary of a single face, which happens if and only if the edge is a cut-edge; this fact will create some complications later on.

2 Non-planar graphs

To show that a graph is planar, we only have to supply a planar drawing. It is often a little harder to show that a graph is not planar.

Proposition 9.1. The graph $K_5$ is not planar.

Proof. The graph contains a $K_3$, which can basically be drawn in only one way. If in a drawing the fourth vertex is inside this $K_3$ and the fifth is outside, then the edge between them must cross the $K_3$, which means the drawing is not planar. If both vertices are inside the $K_3$, then the three edges of one vertex divide the inside face into three faces. The other vertex must then be in one of these three faces, and one of its edges must cross the boundary of this face. A similar argument applies when both vertices are outside the $K_3$.

Note that to make this proof rigorous, we would need the Jordan Curve Theorem to show that a drawing of a $K_3$ has an inside and an outside, and that to connect a vertex inside to a vertex outside we would have to cross the $K_3$. \qed

Proposition 9.2. The graph $K_{3,3}$ is not planar.

Proof. The graph contains a $C_6$, and the remaining three edges connect opposite corners of the $C_6$ in a symmetric way. Two of these three edges would have to be both inside or outside the $C_6$, and there one of them would create a closed curve that the other one would have to cross. Again, this proof relies on the Jordan Curve Theorem. \qed

If a graph $H$ is a subgraph of a graph $G$ and $H$ is not planar, then $G$ is also not planar, since a planar drawing of $G$ would give a planar drawing of $H$. A stronger version of this is the following. Call a graph $H'$ a subdivision of $H$ if it can be obtained from $H$ by repeatedly replacing an edge $xy$ by a path $xyz$ for some new vertex $z$ (informally, we simply place the new vertex $z$ somewhere on top of the edge $xy$, subdividing it into two edges $xz$ and $zy$). It is easy to see that a subdivision of $H$ is planar if and only if $H$ is planar; the new vertices do not have any effect on whether or not we can draw the graph without crossing edges.

Proposition 9.3. The Petersen graph is not planar.

Proof. It contains a subdivision of $K_{3,3}$. The vertices of two parts are marked by red and blue, while the edges of the Petersen graph used in constructing a subdivision are made thick (and paths consisting of edges of one color, as well as single thick edges, correspond to one edge in $K_{3,3}$). \qed
that it does not contain such a subdivision. This is the content of the following theorem. Note that it states an equivalence between a topological statement (the graph having a planar drawing) and a combinatorial statement (the graph not containing a certain subdivision). The proof of this theorem is fairly hard and we will not include it in this course.

**Theorem 9.4 (Kuratowski).** A graph is planar if and only if it does not contain a subdivision of $K_5$ or $K_{3,3}$.

## 3 Euler’s formula

**Theorem 9.5 (Euler).** Let $G$ be a connected planar graph and $D$ a planar drawing of $G$. Then

$$|V(G)| - |E(G)| + |F_D(G)| = 2.$$  

**Proof.** We use induction on $|E(G)|$. Since $G$ is connected, we have $|E(G)| \geq |V(G)| - 1$, so we can start with the base case $|E(G)| = |V(G)| - 1$. In that case, $G$ is a tree, so any planar drawing has only one face and we have $|F_D(G)| = 1$. Then

$$|V(G)| - |E(G)| + |F_D(G)| = |V(G)| - (|V(G)| - 1) + 1 = 2.$$

Assume that $|E(G)| > |V(G)| - 1$. Then $G$ contains a cycle $C$. Pick an edge $e \in E(C)$. The graph $G - e$ is connected, and the drawing $D$ directly gives a planar drawing $D'$ of $G - e$. Moreover, since $e$ is in a cycle, it is not a cut-edge of $G$, so $e$ is on the boundary of two different faces (one inside $C$, one outside $C$; here we use the Jordan Curve Theorem). Removing $e$ merges these two faces, and does not affect any other faces, so $|F_{D'}(G - e)| = |F_D(G)| - 1$. By induction, we have

$$|V(G - e)| - |E(G - e)| + |F_{D'}(G - e)| = 2.$$

Plugging in $|V(G - e)| = |V(G)|$, $|E(G - e)| = |E(G)| - 1$, and $|F_{D'}(G - e)| = |F_D(G)| - 1$ gives Euler’s formula for $G$. 

Note that Euler’s formula implies that for two planar drawings $D_1, D_2$ of a graph we have $|F_{D_1}(G)| = 2 - |V(G)| + |E(G)| = |F_{D_2}(G)|$, so the number of faces in a planar drawing of a graph is independent of the drawing.

**Proposition 9.6.** Let $G$ be a planar graph with $|V(G)| \geq 3$. Then $|E(G)| \leq 3|V(G)| - 6$.

**Proof.** We can assume that $G$ is maximally planar, i.e., that adding any edge to $G$ would make it non-planar. Indeed, given any planar graph, we can add edges until it is maximally planar; if the bound holds for the maximally planar graph, then it also holds for the original graph. Then $G$ is connected, every edge is in the boundary of two faces, and every face has exactly three edges on its boundary, since if any of these properties did not hold, then we could add an edge.

Double counting the pairs $(e, f)$, where the edge $e$ is on the boundary of the face $f$, gives us

$$2|E(G)| = 3|F_D(G)|.$$

Plugging $|F_D(G)| = \frac{2}{3}|E(G)|$ into Euler’s formula gives

$$2 = |V(G)| - |E(G)| + \frac{2}{3}|E(G)|,$$

which rearranges to $|E(G)| = 3|V(G)| - 6$. Since this equality holds for any maximally planar graph, the inequality $|E(G)| \leq 3|V(G)| - 6$ holds for any planar graph. 

28
It is possible to prove Proposition 9.6 in a more direct way, without assuming the graph to be maximally planar. Basically, every edge bounds at most two faces, and every face is bounded by at least three edges, so we should have $2|E(G)| \geq 3|F_D(G)|$. Plugging that into Euler’s formula directly gives the inequality $|E(G)| \leq 3|V(G)| - 6$ (to apply Euler’s formula, we need the graph to be connected, but we can apply it to each connected component, which leads to the same bound). However, it is not always true that every edge bounds at most two faces and every face is bounded by at least three edges; take for instance the path $P_5$, which has one face bounded by two edges. For $P_2$ we have $2|E(G)| \geq 3|F_D(G)|$ anyway, but it is tricky to prove that this holds for all graphs. The step of making the graph maximally planar is a convenient way around this.

Proposition 9.6 gives us a quick proof that $K_5$ is not planar:

$$|E(K_5)| = \binom{5}{2} = 10 > 9 = 3 \cdot 5 - 6 = 3|V(K_5)| - 6.$$  

This does not quite work for $K_{3,3}$ or the Petersen graph, but their non-planarity can also be proved using Euler’s formula (see Problem Set 9 for $K_{3,3}$).

**Corollary 9.7.** If $G$ is planar, then it has a vertex of degree at most five.

**Proof.** Combining Proposition 9.6 with the degree formula from the first lecture gives

$$\sum_{v \in V(G)} d(v) = 2|E(G)| \leq 6|V(G)| - 12.$$  

This implies that the average degree of a vertex is strictly less than six, so there must be a vertex with degree at most five.

### 4 Coloring planar graphs

**Definition.** A vertex coloring of a graph $G$ is a map $c : V(G) \to \mathbb{N}$ such that $c(x) \neq c(y)$ whenever $xy \in E(G)$. The chromatic number $\chi(G)$ of $G$ is the minimum image size of a vertex coloring of $G$; in other words, it is the minimum number of colors that $V(G)$ can be colored with.

**Theorem 9.8 (Five Color Theorem).** If $G$ is planar then $\chi(G) \leq 5$.

**Proof.** Fix a planar drawing of $G$. We use induction on $|V(G)|$. The statement is obvious for $|V(G)| \leq 5$. By Corollary 9.7, there is a vertex $v \in V(G)$ of degree at most five. By induction, we can color $G - v$ with five colors. If this coloring uses at most four colors on $N(v)$, then we can color $v$ with the fifth color and we are done. Thus we can assume that $v$ has five neighbors $x_1, \ldots, x_5$, and that $x_i$ has color $i$. We can also assume that the edges $vx_1, \ldots, vx_5$ leave $v$ in that order when we go around $v$ in the clockwise direction (say).

We call a path in $G$ an $ij$-path if all its vertices have color $i$ or $j$.

Suppose that there is no 13-path from $x_1$ to $x_3$. Let $R$ be the set of vertices that are reachable from $x_1$ by a 13-path. By assumption, $x_3$ is not in $R$. Then we can swap the colors 1 and 3 for all the vertices in $R$. This gives a valid coloring, and it leaves $x_1$ and $x_3$ both colored with 3. Then we can color $v$ with 1 and we are done.

Now suppose that there is a 13-path from $x_1$ to $x_3$; together with $v$ this path forms a cycle $C$, all of whose vertices are colored with 1 or 3 (or uncolored in the case of $v$). Let $S$ be the set of vertices reachable by 24-paths from $x_2$. Then the cycle $C$ separates $S$ from $x_4$ (here we use the Jordan Curve Theorem), so $x_4$ is not in $S$. Thus we can swap colors 2 and 4 in $S$, and then color $v$ with 2.
This theorem is actually not best possible. The famous Four Color Theorem states that for any planar graph we have $\chi(G) \leq 4$. However, the proof is extremely hard and has only been completed using computers to check many cases.

10 Graph colorings and planar drawings

Lecture 10 – 04.05.2017

1 Vertex colorings

We have seen colorings of planar graphs on the previous lecture. Actually, many real-life problems may be interpreted as graph coloring problems. Here is one example from scheduling:

**Example.** The students at a certain university have annual examinations in all the courses they take. Naturally, examinations in different courses cannot be held concurrently if the courses have students in common. How can all the examinations be organized in as few parallel sessions as possible? To find a schedule, consider the graph $G$ whose vertex set is the set of all courses, two courses being joined by an edge if they give rise to a conflict. Clearly, independent sets of $G$ correspond to conflict-free groups of courses. Thus, the required minimum number of parallel sessions is the chromatic number of $G$.

We will prove a couple of simple bounds on the chromatic number of $G$. Let $G = (V, E)$ be a graph. We say that $G$ is $k$-degenerate if every subgraph of $G$ has a vertex of degree less than or equal to $k$. The following proposition connects the degeneracy of $G$ and the chromatic number of $G$. We de-facto used it last lecture in the proof of Five Color Theorem.

**Lemma 10.1.** If $G$ is $k$-degenerate, then $\chi(G) \leq k + 1$. In particular, $\chi(G) \leq \Delta(G) + 1$.

**Proof.** We prove it by induction on the number of vertices. The statement is clearly true for all $G$ with at most $k + 1$ vertices. Find a vertex of degree $\leq k$ in $G$. The graph $G - v$ is $k$-degenerate, so it can be properly colored into $k + 1$ colors. Then color $v$ into the color that does not appear among the colors of its neighbors. This gives a proper coloring of $G$.

As for the second part, it is obvious that any $G$ is $\Delta(G)$-degenerate. □

Recall that the independence number $\alpha(G)$ is the size of the largest independent subset $S \subset V(G)$, that is, a set of vertices $S$ with no two vertices connected by an edge.

**Lemma 10.2.** For any graph $G$ we have $\chi(G) \geq \frac{|V(G)|}{\alpha(G)}$.

**Proof.** Given a coloring with $\chi(G)$ colors, the color classes (label them $S_1, \ldots, S_{\chi(G)}$) are independent sets, and thus have size at most $\alpha(G)$. Hence we have

$$|V(G)| = \sum_{i=1}^{\chi(G)} |S_i| \leq \sum_{i=1}^{\chi(G)} \alpha(G) = \chi(G)\alpha(G).$$

□
2 Edge colorings

**Definition.** An edge coloring of a graph \( G \) is a map \( c : E(G) \to \mathbb{N} \) such that \( c(e) \neq c(e') \) whenever \( e, e' \) are distinct edges that share a vertex. The edge-chromatic number \( \chi_e(G) \) of \( G \) is the minimum image size of an edge coloring of \( G \); in other words, it is the minimum number of colors that \( E(G) \) can be colored with.

Each color class in an edge coloring is a matching, so an edge coloring is partition of the graph into matchings.

An even cycle has edge-chromatic number 2 and an odd cycle has edge-chromatic number 3. The picture on the right is an edge coloring of the Petersen graph with four colors. It is not difficult to see that its edge chromatic number is actually equal to 4.

**Lemma 10.3.** For any graph \( G \) with at least one edge we have \( \Delta(G) \leq \chi_e(G) \leq 2\Delta(G) - 1 \).

**Proof.** There must a vertex of degree \( \Delta(G) \), and an edge coloring must give a different color to each of the \( \Delta(G) \) edges at that vertex. This implies \( \chi_e(G) \geq \Delta(G) \).

The upper bound follows by a proof similar to the proof of Lemma [10.1]. Take an arbitrary ordering of the edges, and consecutively color each edge with the smallest color that is not yet used on the colored edges that it shares a vertex with. Since an edge shares a vertex with at most \( 2(\Delta(G) - 1) \) edges, it follows that \( 2\Delta(G) - 1 \) colors suffice.

The greedy algorithm for edge coloring can be significantly improved on. This is shown in the proof of the following theorem, which tells us that any graph \( G \) has edge-chromatic number either \( \Delta(G) \) or \( \Delta(G) + 1 \). Both are possible, since even cycles have \( \chi_e(G) = \Delta(G) \) and odd cycles have \( \chi_e(G) = \Delta(G) + 1 \). However, this algorithm still does not always give the exact number, and in fact it is NP-hard to determine which of the two values is the edge-chromatic number of a given graph.

**Theorem 10.4 (Vizing).** For any graph \( G \) we have \( \chi_e(G) \leq \Delta(G) + 1 \).

**Proof.** We use induction on the number of edges; the statement clearly holds for a graph without edges. Given an edge \( xy \in E(G) \), we describe an algorithm that, given an edge coloring of \( G - xy \) with at most \( \Delta(G) + 1 \) colors, produces an edge coloring of \( G \) with the same number of colors. To find a color for \( xy \), the algorithm may have to change the colors of other edges, so it is not a greedy algorithm.

We first give some ad hoc definitions.

- If no edge incident with vertex \( v \) has color \( c \), then we say that \( c \) is free at \( v \), or that \( v \) is \( c \)-free.

- Given two colors \( c, d \), a \( cd \)-path is a path in \( G - xy \) whose edges are colored \( c \) and \( d \). If a \( cd \)-path is maximal, then we can invert it by switching the colors \( c \) and \( d \) along the path. The result will still be an edge coloring, because if there were a conflict, then the path would not be maximal.
A fan consists of a vertex \( x \), a sequence \( y_0, \ldots, y_k \) of distinct neighbors of \( x \), and a sequence \( c_1, \ldots, c_k \) of distinct colors, such that \( xy_0 \) is uncolored, and for \( 1 \leq i \leq k \), \( xy_i \) has color \( c_i \) and \( y_{i-1} \) is \( c_i \)-free. We can rotate a fan by recoloring edge \( xy_{i-1} \) with color \( c_i \) for \( i = 1, \ldots, k \), and leaving \( xy_k \) uncolored; the result is still an edge coloring (except for \( xy_k \)).

Given an edge coloring of \( G - xy \) with \( \Delta(G) + 1 \) colors, we begin by constructing a fan based at \( x \) with \( y_0 = y \) as follows. Since we have \( \Delta(G) + 1 \) colors and \( y_0 \) has degree at most \( \Delta(G) \), there is a color \( c_1 \) that is free at \( y_0 \). If there is an edge incident to \( x \) with color \( c_1 \), then we label its other endpoint \( y_1 \). We continue like this: We pick a color \( c_{i+1} \) that is free at \( y_i \), and if possible we pick a \( c_{i+1} \)-colored edge \( xy_{i+1} \) that is not yet in the fan. This terminates when we have a vertex \( y_k \) that is \( c \)-free for some color \( c \), but there is no new edge incident to \( x \) with color \( c \).

Given this fan, there is a color \( c \) that is free at \( y_k \), and there is a color \( d \) that is free at \( x \). We take a maximal \( cd \)-alternating path containing \( x \). Such a path either consists of \( x \), or it starts at \( x \) with a \( c \)-edge, possibly followed by a \( d \)-edge, etc. We invert the path, which gives a new edge coloring.

After this inversion \( x \) is \( c \)-free. We claim that for some \( 1 \leq \ell \leq k \), \( y_\ell \) is \( c \)-free, and \( x, y_1, \ldots, y_\ell \) still forms a fan. If the alternating path was just \( x \), and the inversion did nothing, then we can take the whole fan. Otherwise, the path started with some \( c \)-edge \( xy_{\ell+1} \), and \( y_\ell \) must have been \( c \)-free before the inversion; in other words, \( c_i = c \). If the path did not contain \( y_\ell \), then \( x, y_1, \ldots, y_\ell \) is still a fan. Indeed, the colors \( c_1, \ldots, c_k \) are distinct, so for \( j \leq i \) the inversion did not affect the fact that \( xy_j \) has color \( c_j \) or the fact that \( y_{j-1} \) is \( c_j \)-free. If the path did somehow reach \( y_\ell \), then the inversion made it \( d \)-free, and it colored \( xy_{\ell+1} \) with \( d \). In this case, \( x, y_1, \ldots, y_\ell \) is still a fan, with the only change within the fan being that \( c_\ell \) is replaced with \( d \).

In all cases we have a fan \( x, y_1, \ldots, y_\ell \) in which \( x \) and \( y_\ell \) are \( c \)-free. Now we can rotate the fan, so that the uncolored edge \( xy_0 = xy \) becomes colored, and \( xy_\ell \) becomes uncolored. Then we can color \( xy_\ell \) with \( c \).

\[ \square \]

### 3 Drawing graphs on the plane

Most of the graphs are not planar, and one way to measure, how far is a graph from being planar, is via the crossing number. The crossing number \( cr(G) \) of a graph \( G \) is the minimal number of crossings of edges in a drawing of \( G \) on the plane. (In addition to the requirements that we put on drawings last time, we assume that no three edges cross at the same point.)

In the last exercise sheet, you proved the following bound, given here in a slightly weaker form:

**Proposition 10.5.** \( cr(G) \geq e - 3v \), where \( v \) is the number of vertices of \( G \) and \( e \) is the number of edges of \( G \).

In this lecture we prove something much stronger using the probabilistic method.

**Theorem 10.6.** For any \( G \) satisfying \( e \geq 4v \) we have \( cr(G) \geq \frac{e^3}{6407} \).

Before proving the theorem, let us make the following observation.

**Observation 10.7.** In a drawing of \( G \) minimizing the number of crossings any two edges of \( G \) share at most 1 point in common (either a vertex or a crossing point).


11 Crossing lemma, Ramsey numbers

Lecture 11 – 11.05.2017

1 Crossing lemma

Let us recall the statement of the theorem from the previous lecture.

**Theorem 11.1.** For any $G$ satisfying $e \geq 4v$ we have $cr(G) \geq \frac{e^3}{64v^2}$.

**Proof.** For $e \leq 4v$ the bound in the theorem is weaker than the bound from Proposition [10.5]. Therefore, we may assume that $e > 4v$.

Take the drawing $D$ of $G$, in which $G$ has the least number of crossings. By Observation [10.7], any two edges of $G$ share at most 1 point in common.

Fix a parameter $0 < p < 1$, which will be specified later, and construct a random subgraph $H$ of $G$ by including each vertex of $G$ into $H$ independently with probability $p$. Let $e(H)$, $v(H)$ and $cr(H)$ denote the number of edges, vertices and crossings of $H$, respectively.

By Proposition [10.5], we have

$$cr(H) \geq e(\bar{H}) - 3v(\bar{H}). \quad (1)$$

On the other hand, let us calculate the expected number of vertices, edges, and crossings in $H$. Each vertex has a chance $p$ to be in $H$, therefore, using the linearity of expectation, we have $E[v(\bar{H})] = vp$. Similarly, for an edge to get into $H$, both of its endpoints must be in $H$, which happens with probability $p^2$. Therefore, $E[e(\bar{H})] = ep^2$. The probability that a crossing in the drawing $D$ of $G$ stays in the induced drawing of $H$, is $p^4$. Indeed, both edges must stay, and these two events are independent and have probability $p^2$ (recall that $D$ has no pairs of edges with the same endpoint that cross). Since the drawing of $H$ based on the drawing $D$ is a valid drawing, but may be not optimal, we have $E[cr(H)] \leq p^4 cr(G)$. Therefore, (1) in expectation transforms into

$$p^4 cr(G) \geq E[cr(H)] \geq E[e(\bar{H})] - 3E[v(\bar{H})] = ep^2 - 3vp. \quad (2)$$

Putting $p = 4v/e$ (recall that $e > 4v$, so $p < 1$), (2) transforms into

$$cr(G) \geq \frac{16v^2/e - 12v^2/e}{(4v/e)^4} = \frac{e^3}{64v^2}.$$ 

One conclusion that we can make from this theorem is that both $K_n$ and $K_{n,n}$ have $\Omega(n^4)$ crossings in any drawing on the plane, that is, a positive fraction of pairs of their edges must cross. (We say that $f(n) = \Omega(g(n))$ if there exists a constant $c > 0$ and $n_0$ such that $f(n) \geq cg(n)$ for all $n \geq n_0$.)
2 Ramsey numbers. Upper bounds

Proposition 11.2. If the edges of $K_6$ are colored red and blue, then there is a red $K_3$ or a blue $K_3$.

Proof. Pick a vertex $x_0$. It has five incident edges, among which there must be three red edges or three blue edges. Without loss of generality, assume there are three red edges $x_0x_1, x_0x_2, x_0x_3$. If one of the edges $x_1x_2, x_2x_3, x_1x_3$ is red, then it forms a red $K_3$ together with the two corresponding edges to $x_0$. Otherwise, the edges $x_1x_2, x_2x_3, x_1x_3$ are all blue, so they form a blue $K_3$.

Definition. Given a coloring of the edges of a graph $G$, we say that $G$ contains a monochromatic $H$ if $G$ has a subgraph $H$, all of whose edges have the same color.

Definition. We write $R(H_1, \ldots, H_k)$ for the minimum $N$ such that for every $k$-coloring of the edges of $K_N$ with the colors $c_1, \ldots, c_k$, there is a monochromatic $H_i$ in color $c_i$ for at least one $i$.

Note that the order of the colors is important. For instance, for $R(K_3, K_4)$, the edges of $K_N$ are colored red and blue, and we want to find a red $K_3$ or a blue $K_4$; a blue $K_3$ would not be enough.

Proposition 11.2 thus states that $R(K_3, K_3) \leq 6$. In fact, we have

$$R(K_3, K_3) = 6,$$

because of the following coloring of $K_5$. We take a $C_5$ in $K_5$ and color its edges red; the remaining edges form another $C_5$, which we color blue. There is no $K_3$ in either of the colors.

The Ramsey numbers $R(K_t, K_t)$ have probably been studied most of all. Nevertheless, there is still a mysteriously large gap between the best lower bound and upper bound. In this section we prove lower and upper bounds that are very close to the best known bounds.

Specifically, we will prove that for $t \geq 2$ we have

$$2^{\frac{1}{2}t} \leq R(K_t, K_t) \leq 2^{\frac{1}{2}t}/\sqrt{t}.$$

There have been small improvements to these bounds, but these have not changed the bases of the exponentials on both sides, i.e., $\sqrt{2}$ from below and $4$ from above. It remains a mystery why there is such a big gap between the bounds, and it is big open problem in combinatorics to narrow this gap.

We prove the upper bound and lower bound separately.

Theorem 11.3 (Erdős-Szekeres). For all $t \geq 2$ we have $R(K_t, K_t) \leq 2^{\frac{1}{2}t}/\sqrt{t}$.

Proof. Interestingly, for the proof of this bound for the balanced Ramsey numbers, we will work via the unbalanced Ramsey numbers. We claim that for all $s, t \geq 2$ we have

$$R(K_s, K_t) \leq R(K_{s-1}, K_t) + R(K_s, K_{t-1}). \tag{3}$$

Let $N = R(K_{s-1}, K_t) + R(K_s, K_{t-1})$ and consider a coloring of the edges of $K_N$ with red and blue. Pick a vertex $x$. It has either at least $R(K_{s-1}, K_t)$ red edges or at least $R(K_s, K_{t-1})$ blue edges. The cases are symmetric, so we can assume without loss of generality that $x$ has $R(K_{s-1}, K_t)$ red edges. Consider the 2-colored complete graph on the corresponding $R(K_{s-1}, K_t)$ neighbors of $x$. By definition of $R(K_{s-1}, K_t)$, this graph has either a red $K_{s-1}$ or a blue $K_t$. In the first case we get a red $K_s$, and in the second case we get a blue $K_t$. This proves the claim.
It follows from (3) that for all \( s, t \geq 2 \)

\[
R(K_s, K_t) \leq \binom{s + t - 2}{s - 1}.
\]

(4)

To prove this we use induction on \( s + t \). The bound is easily checked for \( s = 2 \) or \( t = 2 \). Using (3), induction, and the equality \( \binom{m}{n-1} + \binom{m}{n} = \binom{m+1}{n} \), we get

\[
R(K_s, K_t) \leq R(K_{s-1}, K_t) + R(K_s, K_{t-1}) \\
\leq \binom{(s - 1) + t - 2}{(s - 1) - 1} + \binom{s + (t - 1) - 2}{s - 1} \\
= \binom{s + t - 2}{s - 1}.
\]

For \( s = t \), (4) gives \( R(K_t, K_t) \leq \binom{2t - 2}{t - 2} \), and we merely need to approximate this bound. A version of Stirling’s formula gives

\[
2\sqrt{n}ne^{-n} \leq n! \leq 3\sqrt{n}ne^{-n},
\]

which leads to

\[
\binom{2n}{n} = \frac{(2n)!}{(n!)^2} \leq \frac{3\sqrt{2n}(2n)^{2n}e^{-2n}}{(2\sqrt{n}ne^{-n})^2} = \frac{3}{2\sqrt{2}} \cdot \frac{2^{2n}}{n^{2n}}.
\]

Therefore

\[
R(K_t, K_t) \leq \binom{2t-2}{t-2} \leq \frac{3}{2\sqrt{2}} \cdot \frac{2^{2(t-1)}}{\sqrt{t-1}} \leq \frac{2^t}{\sqrt{t}}.
\]

This proves the theorem. \( \square \)

The following simple statement implies that Ramsey numbers are finite for any \( k \) and numbers \( s_1, \ldots, s_k \) and is left for the exercise session:

**Proposition 11.4.** We have \( R(K_{s_1}, \ldots, K_{s_k}) \leq R(K_{s_1}, \ldots, K_{s_{k-2}}, R(K_{s_{-1}}, K_s)) \).

We finish this section with the following cute corollary.

**Corollary 11.5** (Schur). For every \( k \) there exists an \( N \) such that if \( \{1, \ldots, N\} \) is \( k \)-colored, then there is a monochromatic solution of \( x + y = z \).

**Proof.** By Theorem [11.3] we can choose \( N \) large enough so that every \( k \)-coloring of the edges of \( K_N \) contains a monochromatic \( K_3 \). Consider a complete graph \( K_N \) on vertex set \( \{1, \ldots, N\} \), and color the edge \( ij \) with the color of the integer \( |i - j| \). There must be a monochromatic \( K_3 \) in this graph, which means there are three integers \( i < j < k \) such that \( j-i, k-j, k-i \) have the same color. But \( k-i = (k-j)+(j-i) \), so \( (x, y, z) = (j-i, k-j, k-i) \) is a monochromatic solution of \( x + y = z \). \( \square \)

For instance, for \( k = 2 \) we can take \( N = 5 \). Indeed, suppose \( \{1, 2, 3, 4, 5\} \) is 2-colored. We can assume that 1 is colored red. If 2 is also red, then \( 1+1 = 2 \) is a monochromatic solution, so we can assume that 2 is blue. Since \( 2+2 = 4 \), we can assume that 4 is red. Then since \( 1+4 = 5 \), we can assume that 5 is blue. Now if 3 is red, then \( 1+3 = 4 \) is monochromatic, while if 3 is blue, then \( 2+3=5 \) is monochromatic.
3 Ramsey numbers. Lower bounds

We proceed to the lower bound. Its proof is one of the first examples of the probabilistic method in combinatorics.

**Theorem 11.6.** 1. If for some $t, n \in \mathbb{N}$ we have

$$2^{1 - \binom{t}{2}} \binom{n}{t} < 1,$$

then $R(K_t, K_t) > n$.

2. If for some $n = n(t)$ and $t \to \infty$ we have

$$2^{1 - \binom{t}{2}} \binom{n}{t} = o(n),$$

then $R(K_t, K_t) > (1 - o(1))n$.

**Proof.** Color the edges of $K_n$ randomly and independently of each other into red and blue (both possibilities have probability $1/2$). Then the probability that on a given set of $t$ vertices we got either a red or a blue clique is $2^{1 - \binom{t}{2}}$. Therefore, by the linearity of expectation, the expected number of monochromatic cliques is

$$X := \binom{n}{t} 2^{1 - \binom{t}{2}}.$$

There exists a graph $G$ which has at most $X$ monochromatic cliques.

For the first part of the theorem, we have $X < 1$, and, since the number of monochromatic cliques is integer, it must be equal to 0 in $G$.

For the second part of the theorem, the number of cliques in $G$ is $o(n)$. Deleting one vertex from each clique gives us a graph on $(1 - o(1))n$ vertices without any cliques. □

While the second approach gives better bounds, in this particular problem it doesn’t make much difference.

12 Ramsey numbers, graphs with high girth and high chromatic number

Lecture 12 – 18.05.2017

1 Ramsey numbers

Let us obtain a numerical corollary from Theorem 11.6.

**Corollary 12.1** (Erdős). For all $t > 3$ we have $R(K_t, K_t) \geq 2^{t/2}$. 
Proof. We wish to choose \( n = n(t) \), so that \( 2^{1 - \left( \frac{t}{2} \right)} \binom{n}{t} < 1 \). Let us make some estimates:

\[
\binom{n}{t} \leq \frac{n^t}{t!} \leq \frac{n^t}{2^{1+t/2}},
\]

where the last inequality holds for \( t > 3 \). Then

\[
2^{1 - \left( \frac{t}{2} \right)} \binom{n}{t} < 2^{1-t(t-1)/2-1/2}n^t = 2^{-t^2/2}n^t.
\]

This is smaller than 1 if \( n < 2^{t/2} \).

\[\square\]

2 Graphs with high girth and high chromatic number

In this section we will show, using probabilistic techniques, that there exist graphs with no short cycles and with high chromatic number. This means that the chromatic number of a graph is not based on local properties of a graph: locally such a graph will look like a forest, but globally it has a very complicated structure, as measured by the chromatic number.

There are explicit constructions of such graphs, but they are much more complicated and were found later.

The construction in the theorem is based on a random graph \( G(n, p) \). The graph \( G(n, p) \) has \( n \) vertices, and each edge is included in \( G(n, p) \) independently of others and with probability \( p \), \( 0 < p < 1 \). Strictly speaking, \( G(n, p) \) is not a graph, but a probability space, which assigns each (labeled) graph \( G = (V, E) \), \( |V| = n \), probability, equal to \( p^{|E|}(1-p)^{\binom{n}{2}-|E|} \). We will need only basic things about \( G(n, p) \) for our purposes.

Any graph characteristics of \( G(n, p) \) is a random variable, and we would be interested in expected values of such characteristics. Let us start with the expected number of edges. Each edge appears in \( G(n, p) \) with probability \( p \), and there are \( \binom{n}{2} \) potential edges, therefore, by the linearity of the expectation, the expected number of edges is \( p \binom{n}{2} \).

How do we find the expectation of \( X_S \)? In the first problem set (Exercise 9) we showed that there are \( \frac{(k-1)!}{2} \) different cycles of length \( k \) in a complete graph \( K_k \). Each of those has probability \( p^k \) to “survive” in \( G(n, p) \). Therefore,

\[
E[X_k] = \frac{(k-1)!}{2} p^k,
\]

which concludes the proof.
Let us recall Markov inequality. For any random variable $X$, which satisfies $X \geq 0$, we have

$$\Pr[X \geq \alpha E[X]] \leq \frac{1}{\alpha}$$

for any $\alpha > 1$.

**Theorem 12.3.** For any $k, l \in \mathbb{N}$ there is a graph $G$ with no cycles of length $\leq k$ and $\chi(G) \geq l$.

**Proof.** Choose a sufficiently large $n = n(k, l)$ and fix $p = n^{-1 + \frac{1}{k+1}}$. Consider a graph $G(n, p)$. By Proposition 12.2, the expected number of cycles of length $s$ for $s \geq 3$ in such a graph is

$$\binom{n}{s} \frac{(s-1)!}{2^s} p^s < n^s p^s = n^{\frac{1}{k+1}}.$$

We used the inequality $\binom{n}{s} \leq \frac{n^s}{s!}$ for the inequality above.

Therefore, the expected number of cycles of length less than $k$ is at most

$$C := \sum_{s=3}^{k} n^{s^{\frac{1}{k+1}}} = O(n^{\frac{k}{k+1}}) = o(n).$$

On the other hand, using the inequality $(1-p) \leq e^{-p}$, valid for any $p > 0$, the expected number of independent sets of size $x$ in $G$ is

$$\binom{n}{x} (1-p)^{\binom{x}{2}} < n^x e^{-p(x)} = e^{x(\log n - \frac{(p-1)x}{2})}.$$

This expression is $o(1)$, provided that $\log n = o(px)$, which holds e.g. for

$$x = x_0 := n^{1-\frac{1}{k+1}} \log^2 n.$$

Therefore, using Markov’s inequality, we may conclude that

$$\Pr[G(n, p) \text{ has more than } 2C \text{ cycles of length } \leq k] < \frac{1}{2},$$

$$\Pr[G(n, p) \text{ has at least } 1 \text{ independent set of size } x_0] < \frac{1}{2}$$

for sufficiently large $n$.

It implies that there exists a graph $G'$ which has at most $2C$ short cycles and $\alpha(G') < x_0$. Take such a graph and delete one vertex out of each short cycle, obtaining a graph $G$. We deleted $o(n)$ vertices, and the resulting graph $G$ has no cycles of length $\leq k$. On the other hand, $\alpha(G) \leq \alpha(G') \leq x_0$. Therefore,

$$\chi(G) \geq \frac{|V(G)|}{\alpha(G)} \geq \frac{(1-o(1))n}{x_0} = (1+o(1)) \frac{n^{\frac{1}{k+1}}}{\log^2 n} \to \infty,$$

as $n \to \infty$. The graph $G$ is the desired graph. \qed
1 Triangle-free graphs

Let $G$ be a graph on $n$ vertices that does not contain any triangle as a subgraph (in other words, $G$ is $K_3$-free). What is the maximum number of edges that $G$ can have?

With a bit of thinking one can arrive at the conjecture that $K_{\lfloor n^2/4 \rfloor, \lceil n^2/4 \rceil}$ is optimal, i.e., the answer is $\lfloor n^2/4 \rfloor \cdot \lceil n^2/4 \rceil$. This is indeed the case, although the proof below looks simpler than it actually is.

**Theorem 13.1** (Mantel, 1907). A $K_3$-free graph on $n$ vertices contains at most $\lfloor n^2/4 \rfloor$ edges.

**Proof.** Let $v$ be a vertex in $G$ of maximum degree $\Delta$, and let $S = N(v)$ be its neighborhood (so $|S| = \Delta$). Note that there is no edge with both endpoints in $S$, otherwise we would get a triangle with $v$. So every edge of $G$ touches a vertex in $V(G) \setminus S$. Also, every vertex touches at most $\Delta$ edges, so the total number of edges is at most $\Delta |V(G) \setminus S| = \Delta (n - \Delta) \leq \lfloor n^2/4 \rfloor$.

(At the inequality in the middle, we used the easy fact that $ab \leq (a+b)^2/2$.)

As an application, we answer the following question.

Let $a_1, \ldots, a_n \in \mathbb{R}^d$ be vectors such that $|a_i| \geq 1$ for each $i \in \{1, \ldots, n\}$. What is the maximum number of pairs satisfying $|a_i + a_j| < 1$?

**Proposition 13.2.** There are at most $\lfloor n^2/4 \rfloor$ such pairs.

**Proof.** Define the graph $G$ on $\{1, \ldots, n\}$ where $ij$ is an edge iff $|a_i + a_j| < 1$. It is enough to show that $G$ is triangle-free. But this is indeed the case, since for any $i, j, k \in [n]$,

$$|a_i + a_j|^2 + |a_j + a_k|^2 + |a_k + a_i|^2 = |a_i + a_j + a_k|^2 + |a_i|^2 + |a_j|^2 + |a_k|^2 \geq 3,$$

so at least one of $|a_i + a_j|^2, |a_j + a_k|^2, |a_k + a_i|^2$ is at least 1.

\[ \square \]

2 Cliques

Instead of triangles, we can ask the same question for arbitrary graphs: For a given graph $H$, what is the maximum number of edges that an $H$-free graph on $n$ vertices can have?

**Definition.** The extremal number or Turán number of $H$, $ex(n, H)$, is the maximum value of $|E(G)|$ among graphs $G$ on $n$ vertices containing no $H$ as a subgraph.

As a generalization of the triangle-free case, notice that dense graphs not having $K_{r+1}$ as a subgraph can be obtained by dividing the vertex set $V$ into $r$ pairwise disjoint subsets $V = V_1 \cup \cdots \cup V_r$, $|V_i| = n_i$, $n = n_1 + \cdots + n_r$, joining two vertices if and only if they lie in distinct sets $V_i, V_j$. We denote the resulting graph by $K_{n_1, \ldots, n_r}$ (this is called a complete $r$-partite graph). It has $\sum_{i<j} n_i n_j$ edges. Assuming $n$ is fixed, we get the maximum number of edges among such graphs when we divide the numbers $n_i$ as evenly as possible, that is
$|n_i - n_j| \leq 1$ for all $i, j$. Indeed, suppose $n_1 \geq n_2 + 2$. By shifting one vertex from $V_1$ to $V_2$, we obtain $K_{n_1-1,n_2+1,...,n_r}$, and that contains $(n_1 - 1)(n_2 + 1) - n_1n_2 = n_1 - n_2 - 1 \geq 1$ more edges than $K_{n_1,...,n_r}$. In particular, if $r$ divides $n$, then we may choose $n_i = \frac{n}{r}$ for all $i$, obtaining

$$\binom{r}{2} \left(\frac{n}{r}\right)^2 = \left(1 - \frac{1}{r}\right) \frac{n^2}{2}$$

edges. Turán’s theorem states that this is an upper bound for the number of edges in any graph on $n$ vertices without an $(r + 1)$-clique. This result is considered by many to be the starting point of extremal graph theory. There are many different proofs known; the one we give here is a generalization of our proof for Theorem 13.1.

**Definition.** We call the graph $K_{n_1,...,n_r}$ with $|n_i - n_j| \leq 1$ the Turán graph, denoted by $T(n, r)$.

**Theorem 13.3** (Turán, 1941). Among all the $n$-vertex simple graphs with no $(r + 1)$-clique, $T(n, r)$ is the unique graph having the maximum number of edges.

**Proof.** We apply induction on $r$. The case $r = 1$ is trivial (actually, the case $r = 2$ is). Now assume $r \geq 2$ and let $G$ be a graph that has the maximum number of edges among all graphs on $n$ vertices containing no $K_{r+1}$. Let $v$ be a vertex of maximum degree $\Delta = \Delta(G)$ and let $S = N(v)$ be the neighborhood of $v$ and $T = V(G) \setminus S$ be its complement. Then $|S| = \Delta$. As $G$ contains no $(r + 1)$-clique and $v$ is adjacent to all vertices of $S$, we note that $G$ contains no $K_r$ with all vertices in $S$.

We now construct another $K_{r+1}$-free graph $H$ on $V(G)$ that has at least as many edges as $G$. We define $T$ to be an independent set of $H$, but add all edges between $S$ and $T$ to $H$. Finally, on $S$, let $H$ be isomorphic to $T(\Delta, r - 1)$. Then $H$ is a complete $r$-partite graph, so it clearly is $K_{r+1}$-free. To see that $H$ has no fewer edges than $G$, let $e_T$ denote the number of edges in $G$ touching $T$, and let $e_S$ denote the number of edges not touching $T$ (i.e., with both endpoints in $S$). Then $|E(G)| = e_T + e_S$.

On the other hand, we know that $e_T \leq \Delta |T|$ because each vertex of $T$ has degree at most $\Delta$ in $G$, and we also know that $e_S \leq |E(T(\Delta, r - 1))|$ using induction and the fact that $G$ is $K_r$-free on $S$. But $H$ contains exactly $\Delta |T|$ edges touching $T$ and $|E(T(\Delta, r - 1))|$ edges not touching $T$, so indeed $|E(G)| = e_T + e_S \leq |E(H)|$.

This argument shows that $|E(G)| \leq |E(H)|$ for a complete $r$-partite $H$. But we have seen that $T(n, r)$ maximizes the number of edges among complete $r$-partite graphs, so in fact we have $|E(G)| \leq |E(H)| \leq |E(T(n, r))|$, as needed.

To prove uniqueness, note that equality can only hold in our previous bound if $S$ induces the complete $r - 1$-partite graph $T(\Delta, r - 1)$ (using the induction hypothesis), and $T$ touches exactly $\Delta n_r$ edges in $G$. But the latter can only happen if $T$ is an independent set in $G$. Indeed, the sum of the degrees in $T$ counts each edge spanned by $T$ twice, and each edge connecting $T$ and $S$ once. As $\Delta$ is the maximum degree in $G$, the sum of degrees is at most $\Delta n_r$, so $T$ can only touch this many edges if it spans none of them. But then $G$ is $r$-partite and since it has the maximum number of edges, $G = T(n, r)$. \hfill \square

Turán’s theorem shows that $ex(n, K_{r+1})$ is essentially equal to $(1 - \frac{1}{r}) \frac{n^2}{2}$ (they are equal when $n$ is divisible by $r$ and they differ by a constant when $n$ is not divisible by $r$). But what happens for other graphs? What is $ex(n, H)$? Surprisingly, the answer pretty much only depends on the chromatic number of $H$ (at least when $\chi(H) \geq 3$):
Theorem 13.4 (Erdős-Stone-Simonovits). Let $H$ be a graph of chromatic number $\chi(H) = r + 1$. Then for every $\varepsilon > 0$ and large enough $n$,

$$\left(1 - \frac{1}{r}\right) \frac{n^2}{2} \leq \text{ex}(n, H) \leq \left(1 - \frac{1}{r}\right) \frac{n^2}{2} + \varepsilon n^2.$$ 

3 Bipartite graphs

For bipartite $H$, the Erdős-Stone-Simonovits theorem only says that $\text{ex}(n, H) = o(n^2)$. It is one of the biggest questions in graph theory to determine the order of magnitude of the extremal number of bipartite graphs. There are very few graphs for which we know the answer. One such example is $H = C_4 = K_{2,2}$.

Theorem 13.5. If a graph $G$ on $n$ vertices contains no $K_{2,2}$, then

$$|E(G)| \leq \left\lfloor \frac{n}{4} (1 + \sqrt{4n - 3}) \right\rfloor.$$ 

Proof. Let $G$ be a graph on $n$ vertices without a 4-cycle. Let $S$ be the set of “cherries”, i.e., pairs $(u, \{v, w\})$ where $u$ is adjacent to both $v$ and $w$, with $v \neq w$:

$$u \quad \quad \quad \quad \quad v \quad \quad \quad w.$$ 

We will count the elements of $S$ of in two different ways. Summing over $u$, we find $|S| = \sum_{u \in V(G)} \binom{d(u)}{2}$. On the other hand (and this is the crucial observation): every pair $\{v, w\}$ has at most one common neighbor (because $G$ is $K_{2,2}$-free), so $|S| \leq \binom{n}{2}$.

The rest of the proof is just calculations. So far we have

$$\sum_{u \in V} \binom{d(u)}{2} \leq \binom{n}{2}$$

or equivalently,

$$\sum_{u \in V} d(u)^2 \leq n(n - 1) + \sum_{u \in V} d(u). \quad (5)$$

Now applying the Cauchy-Schwarz inequality to the vectors $(d(u_1), \ldots, d(u_n))$ and $(1, \ldots, 1)$, we get $(\sum_{u \in V} d(u))^2 \leq n \sum_{u \in V} d(u)^2$. This, together with (5), implies

$$\left(\sum_{u \in V} d(u)\right)^2 \leq n^2(n - 1) + n \sum_{u \in V} d(u).$$

Here the sum of the degrees is $2|E(G)|$, so we get $4|E(G)|^2 \leq n^2(n - 1) + 2n|E(G)|$. Or equivalently,

$$|E(G)|^2 - \frac{n}{2} |E(G)| - \frac{n^2(n - 1)}{4} \leq 0.$$ 

Solving this quadratic equation yields the theorem. $\square$

This theorem shows that $\text{ex}(n, K_{2,2}) = O(n^{3/2})$. This upper bound is tight in the sense that there are $K_{2,2}$-free graphs on $n$ vertices with $\Omega(n^{3/2})$ edges. One example is the following.
Example. Let $p \geq 3$ be a prime, and $G_0$ be the graph on the vertex set $\mathbb{Z}_p \times \mathbb{Z}_p$ where $(x,y)$ and $(x_1,y_1)$ are connected by an edge and only if $x + x_1 = y y_1$. (Technically this is a multigraph as it has loops).

Note that $G_0$ is $p$-regular. Indeed, for every $x,y,y_1$ there is a unique choice of $x_1$ such that $(x,y)$ and $(x_1,y_1)$ are adjacent. Also, loops correspond to solutions of the equation $2x = y^2$. There is therefore one loop for every choice of $y$, giving $p$ loops in total. Let $G$ be the graph we obtain by deleting the loops from $G_0$.

Now, $G$ has $n = p^2$ vertices, and $\frac{1}{2}(np - p) = (\frac{1}{2} + o(1))n^{3/2}$ edges. It also has no $K_{2,2}$s. Indeed, for any $(x_1,y_1)$ and $(x_2,y_2)$, a vertex $(x,y)$ adjacent to both of them satisfies $x + x_1 = y y_1$ and $x + x_2 = y y_2$, so $x_1 - x_2 = y(y_1 - y_2)$. If $y_1 = y_2$ then $x_1 = x_2$, so if our chosen vertices $(x_1,y_1)$ and $(x_2,y_2)$ were distinct then $y_1 - y_2 \neq 0$. Then $y$ is uniquely determined from the last equation, and this defines $x$, as well.

It is not too hard to generalize the proof of Theorem 13.5 to arbitrary $K_{s,t}$, and get the following theorem:

**Theorem 13.6 (Kővári-Sós-Turán).** For any integers $1 \leq r \leq s$, there is a constant $c$ such that $\text{ex}(n, K_{r,s}) \leq cn^{2 - \frac{1}{r}}$.

However, lower bound constructions where the number of edges has the same order of magnitude is only known for $r = s = 2$ (see above) and $r = 2, s = 3$.

For even cycles, the following result is the best known upper bound.

**Theorem 13.7.** For any integer $2 \leq k$, there is a constant $c$ such that $\text{ex}(n, C_{2k}) \leq cn^{1 + \frac{1}{k}}$.

Once again, matching constructions are only known for $k = 2, 3, 5$. 