Lecture I

1. Graph it is defined by its structure, vertices (V) and edges (E), G = (V,E).
2. Adjacency: Two vertices u and v of a graph G are called adjacent if they are connected by an edge (i.e. neighbours) | {u,v} ∈ E which means that edge between u and v exist and belongs in the set of edges E
3. Incident: Two edges are called incident if they share a vertex v;
   Vertex & edge are called incident if the vertex is one of the two vertices that the edge connects.
4. Subset: subset is a list of items of the given set, that this set contains, it is part of it. V' ⊂ V
   (a) Subgraph of G = (V,E) is G' = (V',E') such that
   (b) Dominating set: for a graph G = (V, E) is a subset V' of V such that every vertex not in V' is adjacent to at least one member of V', the neighbours of V' are V'\V.
      i. Induced subgraph: if G' contains all edges of G with endpoints in V'.
5. Degree of a vertex: is the number of edges incident to the vertex (i.e. number of edges that connects given vertex with other vertices.
6. Complete graph: is a simple, undirected graph in which every pair of distinct vertices is connected by a unique edge (i.e. every other vertex of graph G is reachable with one edge from given vertex v)
   (a) Complete graph on n vertices is a K^n (i.e. a K^3 is called a *triangle*).
   (b) Clique: is a subgraph of graph G such that this subgraph is a complete graph
7. Bipartite graph: is a graph whose vertices can be divided into two disjoint sets A and B such that every edge connects a vertex in A to one in B; a bipartite graph is a graph that does not contain any odd-length cycles.

   Proof. by contradiction: Let's assume that a bipartite graph has a odd length cycle, where from definition of bipartite graph v_1 is in A and therefore v_2 must be in B. A has k odd vertices and B has k even vertices. Therefore if cycle would exist it means that there is an edge v_1v_k and this is a contradiction of definition of a bipartite graph.

   □

Lecture II & III

1. Degree of a graph G:
   (a) d(G) is an average degree of a graph G, where d(G) $=\frac{1}{|V|}\sum_{v\in V} d(v)$ or $\in (G) := |E|/|V|$
   (b) $\delta(G)$ is a minimum degree of G
   (c) $\Delta(G)$ is a maximum degree of G
   (d) $\delta(G) \leq d(G) \leq \Delta(G)$
2. $|E| = \frac{1}{2}\sum_{v\in V} d(v) = \frac{1}{2}$ d(G) $\cdot|V|$
   If we sum up all the vertex degrees in G, we count every edge exactly twice: once from each of its ends.
3. Number of odd degree vertex in G is even.
4. Path is a finite sequence of edges which connects a sequence of vertices that are distinct (edge cannot be repeated) from each other
   (a) Two paths are called independent when none of the paths share the same vertex
   (b) Px_i is a path P such that it ends at vertex x_i
   P$x_i = x_0x_1.....x_i$
   (c) x_iP is a path P such that it starts at vertex x_i
   x_iP= $x_ix_{i+1}....x_k$
   (d) x_iPx_j is a path P between two vertices x_i and x_i
   x_iPx_i=$x_ix_{i+1}x_{i+2}....x_{j-1}x_j$
5. Cycle is a path P=$x_0x_1x_2....x_j$ with an edge x_jx_0 (i.e. a closed path/walk)
6. Graph G with δ(G) : then you can always find a path in G of a length δ(G)-1
7. Walk is a finite sequence of edges which connect vertices, no need to be distinct.
   (a) if G has a walk W from u to v, and $u \neq v$, then there is also a path from u to v in G.

   Proof. We use induction on the length of the walk.
   Let W be a walk between u and v.
   Base step: if $|W| = 1$, then W is just the edge uv and it is a $u - v$ path.
   Induction step: Now assume the statement is true for all $u - v$ walks of smaller size than W. If all the vertices in W are distinct, then W is $u - v$ path and we are done. Otherwise, W has a repeated

vertex say x. Let W be the walk obtained by suppressing the section of W between the two repetition of x. Obviously W is $u - v$ walk of smaller length than W. By induction hypothesis, W has $u - v$ path which means that W has $u - v$ path.

□

8. Distance: the distance between two vertices in a graph is the number of edges in a shortest path.

9. Diameter: is the longest shortest path in the graph G

10. Graph G is connected, then $\forall\ xy \in$ V: there is a path P from x to y.

11. Isomorphism of graph G and H is a bijection (i.e. one to one correspondence) between the vertex sets of G and H such that $f : V(G) \to V(H)$ such that any two vertices u and v of G are adjacent in G if and only if *f(u)* and *f(v)* are adjacent in H (i.e. have the same edge structure).

    (a) Denoted by $\simeq$.

    (b) If two graphs G and H are isomorphic then degree sequence it is the same (but even if two graphs have the same degree sequence that does not mean hat they are isomorphic).

    (c) The vertex-edge order of connection is preserved (example: "One way is to count the number of vertices of degree 3 that have 2 neighbors also of degree 3. In your first graph the answer is 4, and in the second graph the answer is 0").

12. Euler:

    (a) Eulerian cycle: is a trial in the graph which visits every edge exactly once, which starts and ends at the same vertex

    (b) Eulerian path: is a trial in a graph which visits every edge exactly once

    (c) Eulerian graph: is a graph that contains an Eulerian cycle

    (d) Graph G is Eulerian if and only if the degree of every vertex is even.

    Proof. If G is Eulerian then there is an Euler circuit, P, in G. Every time a vertex is listed, that accounts for two edges adjacent to that vertex, the one before it in the list and the one after it in the list. This circuit uses every edge exactly once. So every edge is accounted for and there are no repeats. Thus every degree must be even. Suppose every degree is even. We will show that there is an Euler circuit by induction on the number of edges in the graph. The base case is for a graph G with two vertices with two edges between them. This graph is obviously Eulerian. Now suppose we have a graph G on $m > 2$ edges. We start at an arbitrary vertex v and follow edges, arbitrarily selecting one after another until we return to v. Call this trail W. We know that we will return to v eventually because every time we encounter a vertex other than v we are listing one edge adjacent to it. There are an even number of edges adjacent to every vertex, so there

will always be a suitable unused edge to list next. So this process will always lead us back to v. Let E be the edges of W. The graph G E has components $C_1, C_2, ..., C_k$. These each satisfy the induction hypothesis: connected, less than m edges, and every degree is even. We know that every degree is even in G E, because when we removed W, we removed an even number of edges from those vertices listed in the circuit. By induction, each circuit has an Eulerian circuit, call them $E_1, E_2, ..., E_k$. Since G is connected, there is a vertex a_i in each component C_i on both W and E_i. Without loss of generality, assume that as we follow W, the vertices $a_1, a_2, ..., a_k$are encountered in that order. We describe an Euler circuit in G by starting at v follow W until reaching a_1, follow the entire E_1 ending back at a1, follow W until reaching a_2, follow the entire E_2, ending back at a_2 and so on. End by following W until reaching a_k, follow the entire E_k, ending back at a_k, then finish off W, ending at v. □

(e) Idea:

Lecture IV

1. Algorithms for finding Euler tours in the Graph G:

   (a) Algorithm 1:

   Result: Algorithm found an Euler tour for the graph G
   W=v_0;
   while *W is not Eulerian tour* **do**
   Find on W a vertex v_0' with incident unused edge;
   Find a closed walk W' from v_0' by iteratively taking an *arbitrary* unused edge at the currently visited vertex by walk W′;
   Prolong walk W by walk W′;
   end

   Algorithm 1: Euler tour I: runtime is $O(m^2)$; does not guarantee finding a Eulerian tour

   (b) Algorithm 2:

   Result: Algorithm found an Euler tour for the graph G
   W=v_0;
   while *W is not Eulerian tour* **do**
   Extend walk W by taking, if possible, an unused edge that is not a bridge in the graph G\∗E(W). If all unused edges are bridges, then take any bridge;
   end

   Algorithm 2: Euler tour II: runtime is $O(m^2)$; guarantee finding a Eulerian tour

2. Bridge: is a edge that if it will be removed from the graph G it cause its disconnection.

3. Hamiltonian:

   (a) Hamiltonian path: is a path in a graph G that contains all vertices of G.

   (b) Hamiltonian cycle: it is a cycle in the graph G that contains all vertices of G plus and edge $v_{start}v_{end}$

   (c) Hamiltonian graph: A graph G is called Hamiltonian if and only if it contains a Hamiltonian cycle.

4. Simple graph: is a graph G such that it does not contain loops or parallel edges

5. Dirac Theorem: A simple graph G with n ≥ 3 vertices and of minimum degree $\delta \geq \frac{n}{2}$ is Hamiltonian.

   Proof. Let P=$p_1p_2, \ldots, p_k$ be the longest path in G. If p_1 is adjacent to some vertex v not in P,

   then the path $vp_1p_2, \ldots, p_k$ would be longer than P, contradicting the choice of P.

The same argument can be made for p_k. So both p_1 and p_k are adjacent only to vertices in P.

Since $deg(p_1) \geq \frac{n}{2}$ and p_1 cannot be adjacent to itself,

k$\geq \frac{n}{2} + 1$.

Claim: There is some value of j ($1 \leq j \leq k$) such that: p_j is adjacent to p_k, and p_j+1 is adjacent to p_1. Suppose that the claim is not true. Then since all vertices adjacent to p_1 or p_k lie on P, there must be at least $deg(p_1)$ vertices on P not adjacent to p_k. Since all the vertices adjacent to p_k and p_k itself also lie on P, the path must have at least $deg(p_1)+deg(p_k)+1 \geq n+1$ vertices. But G has only n vertices: a contradiction.

This gives a cycle C=$p_{j+1}p_{j+2}, \ldots, p_k p_j p_{j1}, \ldots, p_2 p_1 p_{j+1}$. Suppose GC is nonempty. Then since G is connected, there must be a vertex $v \in G\ C$ adjacent to some p_i. So the path from v to p_i and then around C to the vertex adjacent to p_i is longer than P, contradicting the definition of P.

Therefore all vertices in G are contained in C, making C a Hamilton cycle.

□

6. Every simple graph G with $n \leq 3$ vertices such that $d(u) + d(v) \leq n$ for any two non adjacent vertices u and $v \in$ V is Hamiltonian.

7. Pigeon Hole Principle: states that if n items are put into m containers, with $n > m$, then at least one container must contain more than one item.

Lecture V

1. Circumference of a graph is a length of the longest circle in the graph G.
2. Every edge e can be uniquely determinate by its endpoints uv (i.e. e=uv).
3. Multigraph: is a graph where is possible to have parallel edges.
4. Directed graph: is a graph G such that every edge $\in$E has a direction to the vertex associated with it.
5. Graph T is called a tree if i. T is connected and ii. T has no cycle. Properties:
   (a) Any two vertices are connected by a unique edge

   Proof. Let T be a graph and let there be exactly one path between every pair of vertices in T. So T is connected. Now T has no cycles, because if T contains a cycle, say between vertices u and v, then there are two distinct paths between u and v, which is a contradiction. Thus T is connected and is without cycles, therefore it is a tree. Conversely, let T be a tree. Since T is connected, there is at least one path between every pair of vertices in T. Let there be two distinct paths between two vertices u and v of T. The union of these two paths contains a cycle which contradicts the fact that T is a tree. Hence there is exactly one path between every pair of vertices of a tree. □

   (b) T is minimally connected (i.e. removal of any edge causes disconnection f the entire graph)

   Proof. Let the graph T be minimally connected. Then T has no cycles and therefore is a tree. Conversely, let T be a tree. Then T contains no cycles and deletion of any edge from T disconnects the graph. Hence T is minimally connected. □

   (c) T is maximally acyclic(i.e. T contains no cycle)

   Proof. Let T be a graph without cycles with n vertices and *n1* edges. We have to prove that T is connected. Assume that T is disconnected. So T consists of two or more components and each component is also without cycles. We assume without loss of generality that T has two components, say T_1 and T_2. Add an edge e between a vertex u in T_1 and a vertex v in T_2. Since there is no path between u and v in T, adding e did not create a cycle. Thus T $\cup e$ is a connected graph (tree) of n vertices, having n edges and no cycles. This contradicts the fact that a tree with n vertices has *n-1* edges. Hence T is connected.

   □

   (d) Leaf of T is a vertex with degree 1.

(e) Every tree with $|V| \geq 2$ has a leaf.

Proof. Let v be a vertex with degree $d \geq \Delta(\mathrm{T})$. For every edge vw incident to v , take a longest path starting with vw . By maximality , the last vertex of this path is a leaf. Doing this for each of the d edges incident to v , we get d paths starting at v , which are disjoint except for v (otherwise we would get a cycle). Thus each path gives a different leaf, and we get $d \geq \Delta(\mathrm{T})$ leaves.

□

(f) Rooted tree T is a tree such that t has one designated vertex v such that v is called a root a tree T.

(g) Every tree T with $n = |V|$ has *n-1* edges.

Proof. We prove the result by using induction on n, the number of vertices. The result is obviously true for $n = 1, 2, 3$. Let the result be true for all trees with fewer than n vertices. Let T be a tree with n vertices and let e be an edge with end vertices u and v. So the only path between u and v is e. Therefore deletion of e from T disconnects T. Now, T e consists of exactly two components T_1 and T_2 say, and as there were no cycles to begin with, each component is a tree. Let n_1 and n_2 be the number of vertices in T_1 and T_2 respectively, so that $n_1 + n_2 = n$. Also, $n_1 < n$and $n_2 < n$. Thus, by induction hypothesis,number of edges in T_1 and T_2 are respectively $n_1 1$ and $n_2 1$. Hence the number of edges in T $= n_1\ 1 + n_2 1 + 1 = n_1 + n_2 1 = n1$. □

(h) Creating a new edge between two vertices of T creates a cycle.

6. Search:

(a) Depth First Search: starts at the root and explores as far possible along the branch before backtracking

(b) Breadth First Search: starts at the root and explores all the neighbour nodes at first before going to the next level neighbours.

Lecture VI

1. Matching in a graph G is a set of M⊆E(G) such that no two edges in G have a vertex in common.

2. Complete matching: of a bipartite graph from A to B is a matching M such that every vertex in A has exactly one incident edge in M.

3. Perfect matching: of a bipartite graph is a matching M such that every vertex ∈ V is matched in M.

4. The maximal cardinality of a matching in a bipartite graph G is equal to be minimum cardinality if a vertex cover of its edges.

5. Augmenting path with respect to matching M is a path such it does not contain an edge from M. ??

6. Hall theorem/condition: A *complete* matching M between set A and B exist if and only if, for every subset X⊆ A and subset Y⊆B, the vertices in A are connected with at least same number of vertices from B.

   Proof. If such a matching exists, then clearly S must have at least $|S|$ neighbors just by the edges of the matching. We shall prove the other direction by induction on the size of $|A|$. When $|A| = 1$, the result is trivial. For the general case, pick an arbitrary $x \in$ A,x must have at least one neighbor $y \in$ B. We can try to match x to y and find a matching of size $|A|1$ in the graph induced onAx, By by induction. The only case in which this would not work is when there is some set S ⊆ A x that has less than $|S|$ neighbors in B y. Then S must have exactly $|S|$ neighbors in B. Let T denote the neighbors of S. By induction, there is a matching that matches all vertices of S to a vertex of T. For every set S' ⊆AS, since S'∪ S has at least $|S'| + |S| = |S'| + |T|$ neighbors in B, S' must have at least $|S'|$ neighbors in B T. Thus there is a matching that matches all the vertices of S' to B T. □

7. Hall Theorem Lecture proof:

   Proof. by induction:
   Base case: $|A| = 1$, Halls' condition tells that ∀S⊆A: $|N(S)| \geq |A|$. Therefore S=A, there needs to be at least one edge e incident to a. Take M=e.
   Induction Step: Lets assume by Induction Hypothesis, that for all bipartite graphs with $|A|$ <k the implications holds.
   Case 1: $\forall S \subseteq A : |N(S)| \geq |S| + 1$ Is there a complete matching on graph G'=(A'∪B',E')? Yes, because $\forall S' \subseteq A' : |N(S)| \geq |N_G(S')| - 1 \geq (|S'| + 1) - 1 = |S'|$ Hence: complete matching M' in G', plus a,b, forms a complete matching for G.
   Case 2: $\exists S \subseteq A : |N(S)| = (S)$

Take G' included by S and N(S), claim: in G' Hall's condition holds: There exist complete matching M' in G'. So $\forall S' \subseteq S : N_G(S') = N_G(S')$ so, $\forall S' : |N_G(S')| \leq |S'|$
Take any $\bar{S} \subseteq \bar{A} := A \backslash B$. Want to show that $|N_{\bar{G}}(\bar{S}) \subseteq |\bar{S}|$, while we know that $|N_G(\bar{S}) < |\bar{S}|$??

□

Lecture VII

1. Maximum Matching is a matching M such that it contains maximum number of edges.

2. Maximal Matching is a matching M such that it cannot be further be expand.

3. Every Maximum Matching is also a Maximal Matching but not every Maximal Matching is a Maximum.

4. Matching algorithm:

Result: Algorithm found a Maximal Matching of the graph G
M=∅ ;
while *M is not a Maximal Matching* **do**
  Iteratively pick an edge e in G, put it into M, delete all the edges sharing a vertex with edge e, repeat until E(G) is empty.
end

This algorithm will provide Maximal Matching because in the definition of the graph we will start at arbitrary edge and we will delete all the incident edges and move to another one , therefore there will be no more edge to match and algorithm is sufficient.

Algorithm 3: Maximal Matching

5. Vertex Cover (of edges): of a graph G=(V,E) as a set $S \subseteq$ V of vertices such that every edge $u,v \in$ E is covered, i.e., u or v is in S.
Vertex cover of a graph is a set of vertices such that each edge of the graph is incident to at least one vertex of the set.

   (a) Minimum Vertex cover: a Vertex Cover of a minimum size. The vertex cover problem is the algorithmic problem to find, for a given graph G.

6. Minimum vertex cover and Matching relation: From graph G take each vertex of matching M and add it to set VC, which is a set that contains a vertex cover of G.
For every Maximal Matching $|M| \leq |VC| \leq 2 * |M|$

7. Let S_{MVC} be a minimum vertex cover of a graph G. A vertex cover S of G is called a ρ-approximation of a minimum vertex cover, if $\frac{|S|}{|S_{MVC}|}$.

8. There is a polynomial time algorithm that computes, for any graph G, a vertex cover that is, in the worst case, a 2-approximation of a minimum vertex cover.

   (a) Basically the best possible poly-time algorithm is "pick any single vertex..."

9. An independent set in a graph G=(V,E) is a set S$\subseteq$V such that there is no edge between any two vertices in S, i.e., for any $u,v \in$V, $u,v \notin$ E.

10. The complement V\S of any independent set S in a graph G=(V,E) is a vertex cover of G.

11. The complement V\C of any vertex cover C in a graph G=(V,E), is an independent set of G.

12. For a simple graph G with n vertices, n=$|minVC| + |maxIS|$, where min VC is a Minimum Vertex Cover of G, and max IS is a maximum Independent Set of G.

Lecture IX

1. Koning's Theorem: In bipartite graphs $|minVC| = |maxM|$

   Proof. $|minVC| \leq |maxM|$ yo show this we need to show that there exist a VC, with one vertex from every $e \in$M, where we know that in the bipartite graphs G=(A∪B,E), one of the set A or B always forms a VC for G.

   (a) pick from ∀ e=u,v∈M exactly one endpoint to S

   (b) we show that S in VC.

   (c) Decision:

      i. for e=u,v∈M

      ii. take u into S, if there exist an altering path starting in A, end ending in v.

      iii. else, put u into S.

   Now, S is a Vertex Cover

      i. Take any edge ab∈E,a∈A,b∈B
      Case 1: ab∈M
      Edge ab is alone an alternating path ending in b, b is matched → b is in S.
      Case 2: ab∉M
      when a is matched:
      If there is no VC on a then b' is placed to S → There is an alternating path P from A to b'.
      b lies on P:
      Pb cannot be augmenting → b is matched → b is chosen for S.

   □

   Proof. Let G be a minimal counterexample. Then G is connected, is not a circuit, nor a path. So, G has a node of degree at least 3. Let u be such a node and v one of its neighbors. If (G\ v) <(G), then, by minimality, G\ v has a cover W' with $|W'| <$ (G). Hence, W'∪ v is a cover of G with cardinality (G) at most. Assume, therefore, there exists a maximum matching M of G having no edge incident at v. Let f be an edge of G\M incident at u but not at v. Let W' be a cover of G\f with $|W'| =$(G). Since no edge of M is incident at v, it follows that W' does not contain v. So W' contains u and is a cover of G.

   □

2. Alternating path P in H with respect to matching M: starts at an unmatched vertex and then contains, alternatively, edges from E\M, and from M.

3. Augmenting Path P in G with respect to matching M: alternating path that ends in the unmatched vertex(thus why P uses at least one edge, non-matching edge).

4.

Result: Algorithm found a Maximum Matching of the graph G
M=$\emptyset$;
while $\exists$ *augmenting path P in G with respect to M* **do**
  (a) Find such P
  (b) Augment M along P ("Flip edges")
end
This algorithm will provide Maximum Matching of G $\Rightarrow$ Berg theorem.

Algorithm 4: Maximal Matching

5. Berge's Theorem states that a matching M in a graph G is maximum (contains the largest possible number of edges) if and only if there is no augmenting path (a path that starts and ends on free (unmatched) vertices, and alternates between edges in and not in the matching) with M.

   Proof. Necessity was shown above so we just need to prove sufficiency. Let us assume that M is not maximum and let M' be a maximum matching. The symmetric difference Q = MΔM' is a subgraph with maximum degree 2(because if this would have more then it means that it is connected with other vertices). Its connected components are cycles and paths where the edges of M and M' alternate. Hence, the cycles have even length and contain as many edges of M and of M'. Since M' is greater than M, Q contains at least one path P that contains more edges of M' than of M. Therefore, the first and the last edges of P belong to M' , and so P is M-augmenting. □

6. Augmenting Path in the bipartite graph G=(A$\cup$B,E) every edge is of odd length, with one endpoint in A, and one endpoint in B, if and only if exist always starts in A without loss of generality.

   (a) Such path P goes "right","left","right"....
   Every such path uses
   - "to the right" only non-matching edges;
   - "to the left" only matching edges.

   (b) $\exists$ an augmented path P in G=(V,E) with respect to M $\Leftrightarrow$ $\exists$ a directed path in $\overrightarrow{G}$(M) that starts and ends in an unmatched vertex.

7. In bipartite graph G=(A$\cup$B, E), one set it is always a vertex cover(i.e. set A of the vertices makes a vertex cover).

Lecture X

1. Weighted graph G=(V,E) together weights on edges, given by a mapping w: E$\rightarrow$ $\mathbb{R}$.
2. Shortest path problem is the problem of finding a path between two vertices in the graph such that the sum of the weights of its constituent edges is minimized.
3. Shortest Path can be found only if there are no negative cycles.
4. $\forall w(e) \geq 0$. Let P be a shortest path between u and v, P=u=$v_0, v_1, v_2, ..., v_k = v$, then $\forall i < j : v_i P v_j$is a shortest path between v_i andv_j.
5. Dijkstra Algorithm:

Result: Algorithm found a Shortest Path in the graph G
-$\forall$ v: d[v]=∞;
-Π[v]=NULL;
-S=$\emptyset$;
while $S \neq V$ **do**
  (a) u$\leftarrow$ vertex with $min_{v \in V \setminus S}$d[v]
  (b) S=S$\cup$u
  (c) $\forall$ neighbour v of u outside S:
    if d[u]+w(u,v,) ¡ d[v] **then**
    d[v]= d[u]+w(u,v)
    Π[v]:=u
end
The running time of the algorithm is $O((n+m)log_n)$

Algorithm 5: Dijkstra Algorithm

6. Proof of Dijkstra: $\forall$u: d[u]=DIST(s,u)

Proof. We prove: "At the start o each iteration of the while loop, d[v]=DIST(s,v)$\forall$v$\in$S.
Easy hit: $S = \emptyset$; S=(s)... this invariant holds d[s] was set to 0.
We proceed by mathematical induction to prove that the invariant holds in every iteration.
CLAIM: d[u]=DIST(s,u)

- going along P, we need to leave S
- we know that d[y]=$dist(s,y)_{(1)}$ (because when x was added to S, we updates d[y])
- Algorithm picked u:d[u]$\leq d[y]_{(2)}$
- P is the shortest path: DIST(s,y)$\leq DIST(s,u)_{(3)}$

Thus: DIST(s,t)$\leq$d[u]$\leq_{(1)}$d[y]$=_{(2)}$DIST(s,y)$\leq_{(3)}$DIST(s,u) $\square$

Lecture XI

1. With negative edges Dijkstra does not work.
2. For the graph G with negative edges we can use Bellman-Ford algorithm, which is based on the existence of negative cycles in the graph G.
3. Bellman-Ford algorithmic approach: What is the shortest path from s to t using at most k edges, where k=1,2,3,...,n-1.
4. Bellman-Ford algorithm:

Result: Algorithm found a Shortest Path in the graph G
-∀ v: d[v]=∞;
-Π[v]=NULL;
-d[s]=0;
for *k=1 TO n-1* **do**
  ∀ edge (u,v)∈E
    if d[v]¿d[u]+w(u,v) **then**
    d[v]= d[u]+w(u,v)
    Π[v]:=u
end

Algorithm 6: Bellman-Ford Algorithm

5. Bellman-Ford algorithm can be modified to detect a negative cycles.

   After the original algorithm finishes, check if ∃ (u,v)∈E such that d[v]>d[u]+w(u,v) if yes ten there is a negative cycle in the graph G.

6. Bellman-Ford:

Proof. By induction on the number of edges k in a path. The base case is correct since$D_s = 0$. For all v∈V\s, on each step a shortest (s, v) path with up to k + 1 edges must consist of a shortest (s, u) path of up to k edges followed by a single edge (u, v). Therefore if we take the minimum of these we get the overall shortest path with up to k + 1 edges. For the source the self edge will maintain $D_s = 0$. The algorithm can only proceed to n rounds if there is a reachable negative-weight cycle. Otherwise a shortest path to every v is simple and can consist of at most n vertices and hence n-1 edges.

□

Lecture XII

1. There is always an Independent Set IS of size $\frac{n}{2}$ in any tree T.
2. To find a Independent Set IS in T:
   (a) we can use the definition of the rooted tree, where we have one vertex which is designated as a root of a tree, and we choose Odd layers for IS and Even no.
   (b) Take v of the largest degree. Put N(v) into set S. Remove from T v and N(v) and all vertices blocked by N(v). ← not optimal
   (c) Take leaf f(s). Adjust T (remove leaf and its neighborhood).
3. Maximal IS:

Result: Algorithm found a Maximum Independent Set
S=$\emptyset$;
while *T has at least one vertex* **do**
  l← a vertex of minimum degree in T;
  S← $S \cup \{l\}$;
  T← $T \backslash (\{l\} \cup \{N(l\})$;
end

Algorithm 7: Independent Set Algorithm

4. A treewidth of a connected graph G , denoted by *tw*(G), is the smallest number $t \in \mathbb{N}$ such that there exist *bags*$B_j \subseteq$V, j =1,...,n', each of size at most t+1, and
   (a) V are covered by the bags, i.e. V $\subseteq \cup_j B_j$.
   (b) For every edge {u,v}∈E(G) there is a bag containing u and v.
   (c) The graph X with bags B_j as vertices, and created as follows is a tree: every two bags B_j and Bj' that share a vertex of V form and edge in X.
   (d) For any vertex $v \in$V, the bags containing v induce a connected subgraph of X.
5. Spanning subgraph G' of graph G is a subgraph G'=(V',E') such that V'=V.
6. A spanning graph G' of G such that G' is a tree is called a *spanning tree* of G.
7. Let G=(V,E) be a connected graph with edge weights w(e), e∈E. A minimum spanning tree of graph G is a spanning tree T of minimum weight w(T):=$\sum_{e \in E(T)} w(e)$.
8. For the problem given above, natural greedy works just fine.

Lecture XIII

1. Traveling Salesman Problem: Given a complete graph G=(V,E) with edge-weights w(e), the traveling salesman problem is to find a cycle C that visits every vertex and has minimum weight w(C):=$\sum_{e \in C}$w(C) among all cycles containing all vertices. In the traveling salesman problem the graph is always complete, therefore you can always find a cycle.

2. Triangle inequality: for every edge $\{uv\}$ and every vertex x, $w(uv) \leq w(ux)+w(xv)$.

   Proof. Easily; starting with a walk of length d_G(u, v) from u to v, and appending a walk of length d_G(v, w) from v to w, results in a walk of length d_G(u, v) + d_G(v, w) from u to w. This is an upper bound on the real distance from u to w. □

3. A metric traveling salesman problem is a traveling salesman problem given by a graph where the edge-weights satisfy the triangle inequality.

4. Minimum Spanning Tree and Traveling salesman problem are related.

   (a) Both are subgraphs of the original graph G.

   (b) Both have similar number of vertices(n in case of TSP and n-1 in case of the MSP)

   (c) Deleting one edge from C_{TSP} would create a MSP.

5. Given Graph G=(V,E) with edge-weights w(e)≥0,e∈E, and given set of required vertices R⊆V, a Stainer Tree for vertices R is a subtree of G that contains all vertices of R. A Minimum Stainer Tree is a Stainer Tree of minimum weight w(T) among all Steiner trees.
   Given graph G does not contain non-negative edges.

6. The Steiner tree problem is the computational problem that asks for a minimum Steiner tree for a given edge-weighted graph G and set of required vertices R.

7. The metric Stainer tree problem is the Stainer tree problem where G is a complete graph, and the edge-weights satisfy the triangle inequality.

Lecture XIV

1. Let

   - e = edge in the graph.
   - c(e) = capacity of the given edge e
   - f(e) = amount of flow going through given edge e

   The theorem states:

   If c(e) for all edges, e, in graph are integers, then there exists a max flow f for which every flow value f(e) is an integer. The theorem only says the upper bound of f(u,v) are integers.. it does not say anything about its lower bound.. therefore flow can be any number between 0 and c(u,v)..

2. Maximum cardinality of matching in G= value of max flow in G.

   Proof. $\leq$

   - Given a max matching M of cardinality k.
   - Consider flow f that sends 1 until along each of k paths.
   - f is a flow and has value k

   $\geq$

   - Let f be a ma flow in G' of value k'.
   - Integrality theorem $\Rightarrow$ k is integral and can assume f is 0-1
   - Consider M= set of edges from L to E with f(e) = 1
     - each node in A and B participates in at most one edge in M.
     - $|M|$=k: consider cut (A$\cup$S,R$\cup$t).

   □

3. k-regular bipartite graph: a graph G=(A$\cup$B,E) such that $|A| = |B|$, and δ(G) = Δ(G) (i.e. all the vertices have equal degree).

   Proof. We prove this by induction on k. If k = 1, then the graph G itself is a perfect matching. For the inductive step, let G have bipartition (A, B). We know $|A| = |B|$. Now suppose S $\subseteq$ A and $N_G(S)| < |S|$. The sum of the degrees of the vertices in $|S|$ is k$|S|$, so by Pigeon-Hole Principle, there is a vertex in $N_G(S)$ with degree greater than k. This contradicts that G is k-regular. Thus, for any S $\subseteq$ A, $|N_G(S)| \leq |S|$. From Hall's Theorem, this means G has a perfect matching. Now delete every edge from this perfect matching. The resulting graph is (k-1)-regular, so the result now follows by our inductive assumption.

   □

4. Two paths are edge-disjoint if they have edge in common.

5. Max number edge-disjoint s⇝t paths equal value of the max flow.

   Proof. $\leq$

   - Suppose there are k edge-disjoint s⇝t paths $P_1, ..., P_k$.
   - f(e)=1 if e participates in some path P_j; else set f(e)=0
   - Since paths are edge-disjoint, f is a flow of value k.

   $\geq$

   - Suppose max flow value is k.
   - Integrality theorem $\Rightarrow$ there exist flow f of value k.
   - Consider edge (s,u) with f(s,u)=1
     - by conservation, there exist an edge (u,v) with f(u,v)=1
     - continue until reach t, always choosing new edge
   - Produces k (not necessarily simple) edge-disjoint paths.

   □

6. The max number of edge-disjoint s⇝t paths is equal to the min number of edges whose removal disconnects t from s.

   Proof. $\leq$

   - Suppose the removal of F⊆E disconnects t from s, and $|F|$ =k. (where F is a set of all edges whose removal disconnects t from s).
   - Every s⇝t path uses at least one edge in F.
   - Hence, the number of edge-disjoint oaths is $\leq$ k.

   $\geq$

   - Suppose max number of edge-disjoint paths is K.
   - Then value of max flow is k.
   - Max-flow min-cut theorem $\Rightarrow$ there exist a cut (A,B) of capacity k.
   - Let F be set of edges going form A to B
   - $|F|$ =l and disconnects t from s.

   □

7. Given graph G=(V,E) with nonnegative edge capacities $c(e)$ and node supply demands $d(v)$, a circulation is a function that satisfies:

   - For each e ∈E: $0 \leq f(e) \leq$ c(e) (capacity)
   - For each v ∈V: $\sum_{e \ in \ to \ v} f(e) - \sum_{e \ out \ of \ v} f(e) = d(v)$ (conservation)

8. From max-flow formulation and integrality theorem for max flow it follows: If all capacities and demands are integers, and there exist a circulation, then there exist one that is integer-valued.

9. Given (V,E,c,d) there does not exist a circulation if and only if there exist a node partition(A,B) such that $\sum_{v \in B}$d(v) > cap(A,B) (i.e. demand by nodes in B exceeds supply of nodes in B plus max capacity of edges going from A to B.).

10. A circulation is a function that satisfies:

    - For each $e \in$ E: l(e) $\leq$ f(e) $\leq$ c(e), where l(e) is a lower bound for each edge $e \in$E. (capacity)
    - For each $v \in$V: $\sum_{e \ in \ to \ v} f(e) - \sum_{e \ out \ of \ v} f(e) = d(v)$, where d(v) are demands for the each node. (conservation)

Lecture XVI

1. A *coloring* of a graph G is an assignment c:V$\rightarrow \mathbb{N}$ of colors (integers) to vertices, such that $\forall\{$ u,v,$\} \in$E: c(u)$\neq$c(v).
   - k-coloring is a coloring using k colors
   - Graph G is k-colorable, if $\exists$ k-coloring c of the graph G
   - chromatic number of G: χ(G) is the smallest number such that G is k-colorable.
2. We can provide a good bound on the χ(G)
3. Basically a coloring induces a partition of V into $V_1, V_2, ..., V_k$, where the set V_i is the set vertices v with c(v) = i and every V_i is an independent set of G.
   - $\forall_{i,j}$ there needs to be at least one edge between V_i and V_j, because then we can merge this two sets into one (independent set) and use one color less.
4. For each K_n we need n colors
5. Brooks: χ(G)$\leq \Delta$(G) for every G not being a clique or odd-length cycle.
6. Bound: χ(G)$\leq \Delta$(G)+1
   - Tight on complete graphs
   - Tight for odd-length cycles
7. 6 colors theorem: Any map can be colored with at most 6 colors.

   Proof. Trivial with planar graph with at most 6 vertices. We can easily produce a 6 coloring with one color for each vertex.

   Now, suppose that G is simple planar with n vertices, and that all simple planar graphs with n-1 vertices are 6 colorable. G must contain a vertex v of degree at most 5. If we delete v and it's incident edges, the remaining graph has (n-1) edges and is thus 6-colorable. A 6-coloring of G is obtained by coloring v with a different color from the (at most 5) vertices adjacent to v. □

   Proof. Lemma: Every standard map has at least one country (face) with five or fewer edges.

   Proof: We will prove this by contradiction. Suppose the conclusion of the lemma is false. Then every face has at least 6 edges. If v is the number of vertices, e is the number of faces, and f is the number of faces, notice that 6 f $\leq$ 2 e (each face generates at least six vertices but each vertex gets counted twice). Also notice that since we have a standard map we have 3 v = 2 e (each vertex generated exactly three edges but each edge gets

counted twice). Euler's formula says that $1 = v - e + f$. But $v = 2/3\ e$ and $f \leq 1/3\ e$ and so $v - e + f \leq 2/3\ e - e + 1/3\ e = 0$. Thus we conclude that $1 \leq 0$ which is clearly false. So it must be the case that there at least one face with five or fewer edges. QED □

8. Planar Graph: a graph G such that can be drawn in the plane so that edges do not cross each other.
9. Dense graph is a graph in which the number of edges is close to the maximal number of edges.
10. Sparse graph is a graph in which the number of edges is close to the minimal number of edges.
11. Euler characteristic of the planar graphs:
    - n- number of vertices
    - m - number of edges
    - number of faces of drawing of G (regions bounded by edges, including the outer, infinitely large region)
    - $\Rightarrow$ n+f-m=2
12. Any connected planar graph has Euler characteristics n+f-m=2.

    Proof. Proof by induction.
    Base case: n = 1
    - if n = 1 then the graph consist of m loops
    - the number of faces of the graph is f = m + 1
    - therefore we indeed have n+f-m=2

    Induction step: n > 1
    - Since G is connected, there exist an edge a,b which is not a loop.
    - We contract a,b but without merging parallel edges
    - We obtain a graph G' with n-1 vertices, m-1 edges and f faces.
    - By the applying the induction hypothesis on G', we have n-1+f=e-1+2
    - Increasing both side by 1 completes the proof

    □
13. Let G be a connected planar simple graph with n vertices, where $n \geq 3$ and m edges. Then $m \leq 3n - 6$.

    Analogy with the walls, so every face in the graph has at least 3 walls, therefore $W \leq 3f$, but every edge has two sides, so number of walls must equal $W = 2m$, therefore $\Rightarrow$ $\mathbf{2m \geq 3f}$,
    and by the Euler formula n+f-m=2, so f=2+m-n, if we will substitute f in the previous walls formula we have: $2m \geq 3(2+m-n)$ and after we get $m \leq 3n - 6$.