Algorithms #8

Problem 1.

Given a list $r_1, r_2, r_3 \ldots r_m$, each requirement r_i says you must take k_i courses from s_i (a set of classes).

and a list of classes a given student has taken, determine if the student can graduate.

Algorithm.

1. Build graph G in the following manner:

   V = Δ node for each requirement (R_i)
   Δ node for each class offered (C_i)
   Δ source node
   Δ sink node.

   $E = \{(S, R_i) \text{ s.t. } c(S, R_i) = k_i \ \forall R_i\} \cup$
   $\{(C_i, T) \text{ s.t. } c(C_i, T) = 1 \ \forall C_i\} \cup$
   $\{(R_i, C_i) \text{ s.t. } C_i \in S_i \text{ and } C_i \in L\}$.

2. Run ford-fulkerson on graph G to determine Max Flow.
3. If max flow is equal to the number of classes required, then the student can graduate.

Proof of correctness.

First I will discuss why G is an accurate representation of the problem and then I am going to discuss why Δ student can graduate iff G has max flow F where $F = \sum k_i$

This graph G represents the problem accurately due to the following reasons.

1. Setting the capacity from S to $R_i \ \forall R_i$ to k_i will ensure the fact that k_i classes are taken from R_i
2. Having edges from R_i to C_i only if $C_i \in L$ will make sure that we can

map classes that the student has taken

3. Making all edges c_i to T have $c(c_i) = 1$ ensures that no given class was taken to fulfill two requirements.

Now I claim that student can graduate iff graph G has Max Flow F where $F = \sum k_i$

$\Rightarrow$

If a student can graduate the graph G has Max Flow F.

If a student can graduate then he has completed L classes for which he has fulfilled all requirements without any overlaps. This means that every node R_i will be able to accept its full capacity and send it to c_i's accross the graph. Moreover, at most $\sum k_i$ units of flow are going to reach all c_i's and thus T will receive $\sum k_i$ units of flow.
In other words if a student can graduate we can flow c units of flow through every edge (S, R_i). Therefore, because this is limiting to the amount of flow that can be pushed on G, then the sum of the capacities of all edges (S, R_i) must be the maximum flow of the graph. Because these capacities where k_i then $F = \sum F_i$

$\Leftarrow$ If graph G has flow $F_{(\sum n_i)}$ then a student can graduate.

If graph G has flow $F = \sum k_i$ then it must be that we have maximized the capacity of all edges (S, R_i), the only way to do so is to have sent 1 unit of flow from R_i to corresponding c_i's where by construction there is not a unit of flow from both $R_i \to C_x$ and $R_j \to C_x$ if $i \neq j$ (No overlapping classes). As $c_i \to T$ has capacity 1. Because we maximized $c(S, R_i)$ $\forall R_i$ then it must be that the student also took k_i classes for r_i. Therefore the student can graduate.

So then : this means that the algorithm will correctly determine if the student can graduate.

Running time.

The amount of nodes in the graph is $n = |S| + |B| + |r_1, r_2 \ldots r_n| + |L|$.

The amount of edges in the graph is $m = |r| + |R_1||R_2| \ldots |R_n| + |L|$.

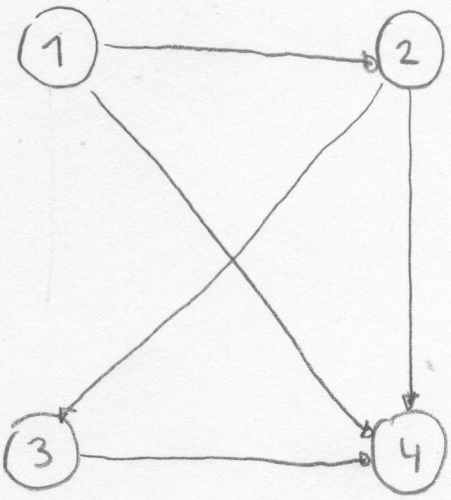
Therefore the running time of creating the graph is $O(n+m)$

Then we need to run a version of ford fulkerson which we proved in class is $O(nm)$.

Therefore the entire algorithm is $O(nm + (n+m))$ which is polynomial.

Problem 3

a)



The degree sequence is realizable.

Problem 2 b)

We are given a list of nodes and the outdegrees and indegrees of ~~the~~ each node, want to know if the graph is realizable.

Algorithm:

1. Check if the sum of indegrees equals the sum of outdegrees. if it is not the graph is not realizable, quit.

2. Form a graph G the following way.

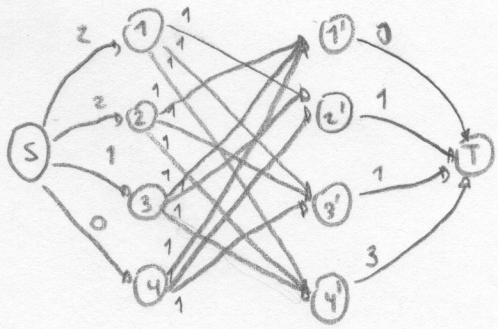
   V = A source node
   A sink node
   Two nodes for each node in the degree sequence. V_i and V_i'

$$E = \{(S, V_i) \forall V_i \text{ st } c(S, V_i) = \text{outdegree}(i)\} \cup$$
$$\{(V_i, V_j') \forall j \neq i \text{ st } c(V_i, V_j') = 1\} \cup$$
$$\{(V_j', T) \forall V_j \text{ st } c(V_j', T) = \text{indegree}(i)\}.$$

for example for

Node	out	in
1	2	0
2	2	1
3	1	1
4	0	3

Graph



3. Use Ford Fulkerson to determine Max Flow F of G.

4. If $F = \sum_{V_i} c(S, V_i)$ then the graph with such degree sequence is realizable.

5. If it is build graph G' where $(i, j) \in E$ if (V_i, V_j') has a unit of flow flowing throught it in G when flow is maximal.

Proof of correctness

First I will discuss why G is an accurate representation of the problem and then I am going to claim that the degree sequence is realizable iff G has a max Flow $F = \sum_{V_i} c(S, V_i)$

We are told to not allow multiedges or cycles so the graph does this by construction by having only 1 edge from a given node v_i to a node v_j' where $c(v_i, v_j') = 1$ and $i \neq j$. The fact that we don't have edges (v_i, v_j') where $i = j$ avoids self loops.

Now I claim that the degree sequence is realizable iff G has max flow $F = \sum_{v_i} c(s, v_i)$

$\Rightarrow$ If a sequence is realizable then G has max flow $F = \sum_{v_i} c(s, v_i)$

If a degree sequence is realizable this means that $\exists$ graph G' where for a particular node v_i it has indegree $= d_{in}(v_i)$ and outdegree $= d_{out}(v_i)$. If this is the case then we can think of graph G as having all these edges have capacity 1 and all other edges having capacity 0. Then it is clear that the max flow across this cut is the number of edges in graph G' (the realizable graph) which in turn is equal to the sum of outdegrees = sum of indegrees. Now because $\sum_{v_i} c(s, v_i)$ = sum of outdegrees is the maximum capacity that can be flowed to all nodes v_i then it is the case that if we have a realizable graph we have maximized all of these capacities and thus no more flow is possible, meaning G has max flow $F = \sum_{v_i} c(s, v_i) =$ sum of outdegrees.

$\Leftarrow$ if G has max flow $F = \sum_{v_i} c(s, v_i)$ then the degree sequence is realizable.

if G has a max flow F then it means that the max flow across the cut $G \setminus S = F$ which is the sum of the capacities of edges (s, v_i). Moreover it must also be that the max flow across the cut $G \setminus T = F$, which is the sum of the capacities of edges (v_j', s). Therefore there must be a way of flowing F flow in 1 units across edges (v_i, v_j') conserving $\sum f_{in}(v_i) - \sum f_{out}(v_i) = 0$. So there must be a realizible graph based on capacities

of the (s, v_i) edges and (v_j', t) edges which by construction is the degree sequence. Thus the degree sequence is realizable.

Altogether the algorithm will determine correctly if a degree sequence is realizable.

Furthermore it follows from ($\Leftarrow$) direction above that after flowing F flow through G, it is clear that those edges (v_i, v_j') with flow 1 are the edges (i, j) in G', the realizable graph.

Running time.

It is clear that the time that it takes us to create the graph is proportional to n and m that is, $O(m+n)$.

Then running Ford Fulkerson takes time $O(nm)$

And constructing the graph G' takes time $O(m)$, to find all the edges (v_i, v_j') st the flow on the edge is 1 unit.

Altogether this algorithm solves in poly time if a degree sequence is realizible and moreover, if it is, it returns graph G'.

Problem 3 a)

Algorithm.

1. Make graph G' such that

$V' = V \cup S' \cup T'$

$E' = E \cup \{(s, v_i) \text{ s.t. } d_{v_i} < 0 \text{ and } c(s, v_i) = -d_{v_i}\} \cup \{(v_i, T) \text{ s.t. } d_{v_i} > 0 \text{ and } c(v_i, T) = d_{v_i}\}$.

2. Use Ford Fulherson to find the max flow F of G'

3. If $F = \sum d_{v_i} \mid d_{v_i} > 0$ then G has a circulation.

Proof of correctness.

let S be the set of nodes s_i s.t. $d_{v_i} < 0$
let T be the set of nodes t_i s.t. $d_{v_i} > 0$

Proposition in order for G to have a feasible circulation f we must have that

$$\sum_{s_i \in S} -d_{s_i} = \sum_{t_i \in T} d_{t_i} = F.$$

And then the original graph G has a circulation iff G' has a max flow F.

$\Rightarrow$ if G has a circulation then G' has a max flow F

if G has a circulation then it means that we are able to send $-d_{s_i}$ units of flow along every edge (s', s_i) and d_{t_i} units of flow along every edge (t_i, T') on G' because it must be that $\forall$ s_i the graph G' was capable of supplying $-d_{s_i}$ units of flow through the graph and $\forall$ t_i t_i received an additional d_i units of flow. So Because there is a circulation in G, the max flow from S' to T' is $F = \sum_{s_i \in S} -d_{s_i} = \sum_{t_i \in T} d_{t_i}$

$\Leftarrow$ if G' has a max flow F then G has a circulation

if G' has a maximum flow F then we must have maximized the capacities of all edges (s', s_i) and (t_i, T') so it must be that if we remove these edges G was able to circulate the flow through the graph

So it is clear that G has a circulation iff G' has $\text{flow}^{\max} F = \sum_{t_i \in T} d_{t_i} = \sum_{s_i \in S} -d_{s_i}$
and it must be that if G has a circulation then $\sum_{s_i \in S} -d_{s_i} = \sum_{t_i \in T} d_{t_i}$.

Therefore our algorithm will correctly say if G has a circulation

Running Time

It is clear that the construction of G' take $O(n)$ to iterate over all nodes. and takes at most $O(n)$ new edges.

So FF will take at most $O(n'm')$ time to run where $n' = O(n)$ and $m' = O(m+n)$. Therefore this algorithm is polytime.

b)

Algorithm.

1. Construct graph G' st $l'(e) = 0$ $c'(e) = c(e) - l(e)$ and $\forall v_i$
   $d'_{v_i} = d_{v_i} - M_v$ where $M_v = \sum_u l(u, v_i) - \sum_u l(v_i, u)$.
2. Run the algorithm of part a on G'. Answer as it would.

Proof of Correctness.

Need to show that G' is an accurate representation of G, if it is then if G' has a circulation then G does as well

What happens if we push $f_0(e) = l(e)$ on every edge? Then the remaining capacity $c'(e) = c(e) - f_0(e) = c(e) - l(e)$ left in the edge.

The problem of pushing $l(e)$ on each edge is that the demand on each node v_i is then $\sum_u l(u, v_i) - \sum_u l(v_i, u) = M_{v_i}$ it could be that $M_{v_i} \neq d_{v_i}$ thus f_0 is not a valid circulation. But this means that the remaining demand on every node is in fact $d_{v_i} - M_{v_i}$

So by pushing $l(e)$ ~~on~~ e we have fulfilled the lower bound on every edge! so if we now have G' where $\forall e$ $c'(e) = c(e) - l(e)$ and $d'_{v_i} = d_{v_i} - M_{v_i}$ we have a graph G' with NO lower bounds (ie $l'(e) = 0$ $\forall e$) so we can just run the algorithm of part a on G' to see if there is a circulation on G.

Running Time.

Clearly the construction of G' takes time $O(n+m)$ and we know solving 2. take poly time. Therefore this algorithm is poly time.