

## Notes for Tutorial 10: Nov 28

### Chapter 7-Problem 19

- (a) To solve the problem we model the problem as a network flow problem.

We construct a graph with the following vertices  $v_1, \dots, v_k$ , representing the doctors.  $u_1, \dots, u_n$  representing the days, and  $s$  and  $t$  as the source and sink.

We have three sets of edges:

- (a)  $E_1$  is the set of edges from source to  $v_j$ . We add edges of capacity  $\infty$  from  $s$  to  $v_j$  for every  $j$ . (we do not have any restriction on the amount work that a doctor can do.)
- (b)  $E_2$  is the set of edges from  $u_i$  to  $t$ . We add edge of capacity  $p_i$  from  $u_i$  to  $t$ . (This restrict the number of doctors that must be present at the day  $i$ .)
- (c) finally  $E_3$ , we apply the lists  $L_j$ . We add an edge of capacity 1, from  $v_j$  to  $u_i$  iff  $i \in L_j$ . This shows that doctor  $j$  can work on the day  $i$  and the capacity 1 is because doctor  $j$  should not be counted twice in the day  $i$ .

Now, assume the flow  $f_e$  is the flow in edge  $e$  in the maximum flow from  $s$  and  $t$ . Let  $F$  be the amount of flow from  $s$  to  $t$ . We have  $F = \sum_{e \in (u_i, t)} f_e$ .

Now if  $c_e = f_e$  for every edge  $e$  from  $u_i$  to  $t$ , then we can assign the doctor  $j$  the working days  $L'_j = \{i | f_{(v_j, u_i)} = 1\}$ . These sets satisfy that on the day  $i$  exactly  $p_i$  doctors work, and each doctors job is assigned from its list  $L_j$ . But, if there is no such flow it means there is no set of list  $L'_1, \dots, L'_k$  that satisfies the properties (A) and (B). because if lists  $L_j$  exists that satisfies (A) and (B) then we could assign the flow  $f_e$  to the edge  $(v_j, u_i)$  iff the  $i \in L'_j$ , and assign the flows to  $E_1$  and  $E_3$  accordingly. This way, we could have a larger flow which is contradiction with our assumption of  $f_e$  being maximum flow.

- (b) For this part we add another vertex set  $v'_j$  to our vertices. We keep the edge sets  $E_1$ ,  $E_2$ , and  $E_3$ . We add edges of capacity  $c$  from  $v_j$  to  $v'_j$ . We also add edges of capacity 1 from  $v'_j$  to  $u_i$  for every  $i$  such that  $i \notin L_j$ .

Now if  $c_e = f_e$  for every edge  $e$  from  $u_i$  to  $t$ , then we can assign the doctor  $j$  the working days  $L'_j = \{i | f_{(v_j, u_i)} = 1 \wedge f_{(v'_j, u_i)} = 1\}$ . We still have the property that no doctor is assigned to the same day more than once. We also have that each doctor is assigned to at most  $c$  working days outside of its list  $L_j$ , because we had  $c_{(v_j, v'_j)} = c$ .

But, if there is no such flow it means there is no set of list  $L'_1, \dots, L'_k$  that satisfies the properties (A\*) and (B), because if lists  $L'_j$  exist that satisfy (A\*) and (B) then we could assign the flow  $f_e$  to the edge  $(v_j, u_i)$  iff the  $(i \in L'_j) \wedge (i \in L_j)$  and  $(v'_j, u_i)$  iff  $(i \in L'_j) \wedge (i \notin L_j)$ , and assign the flow to the other edges accordingly. This way we could have a larger flow which is contradiction with our assumption of  $f_e$  being maximum flow.

example for part a:

$$p_1 = 2, p_2 = 2, p_3 = 1 \quad L_1 = \{1, 2\}, L_2 = \{2, 3\}, L_3 = \{3\}$$

construct the graph and show the relation between the flow and the graph.

example for part b:

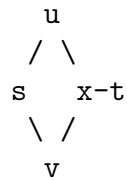
$p_1 = 2, p_2 = 3, p_3 = 3 \quad L_1 = \{1, 2\}, L_2 = \{1, 2\}, L_3 = \{3\} \quad c = 1$  construct the graph and show the relation between the flow and the graph.

### Chapter 7-Problem 10

To solve this problem first we check that whether  $f(e^*) = c(e^*)$  or not. If it was not equal the amount of flow will remain the same. But, if the  $f(e^*) = c(e^*)$  then we have to change the flow. First we deduct 1 from the  $f(e^*)$ . after that, we find a paths  $p_1$  and  $p_2$  from  $s$  to  $u$  and  $v$  to  $t$ , such that for every edge in those paths the amount of flow in that edge is non-zero (we use BFS on the graph where the edges are the edges of with  $f(e) > 0$ ). Then we reduce the amount of flow in each edge  $e$  in  $p_1$  and  $p_2$  by one. Note that,  $p_1 \cap p_2 = \emptyset$  because the flow is acyclic. Now, we have a valid flow where the amount of flow coming to every vertex is equal to the amount of flow going out of that vertex. The total flow is  $F - 1$  now, while we had a flow of size  $F$  in the first flow. This flow might not be the maximum flow, thus we try to find an augmenting path from  $u$  to  $v$ . If there does not exist an augmenting path the amount of maximum flow will remain  $F - 1$ , otherwise we have a flow of size  $F$  and because the maximum flow in the first graph was  $F$  this flow is also maximum.

In this algorithm we use BFS twice and we try to find an augmenting path once, and the other parts of the algorithm is constant time. Thus, the total time of the algorithm is  $\Theta(m + n)$ .

example:



In this example the capacity of all edges are 1 and edges are directed from left to right. The flow from  $s$  to  $t$  is  $s \rightarrow u \rightarrow x \rightarrow t$ . in this example reduce the capacity of edge  $(u, x)$ .