CS157 Homework 9

BY SILAO_XU AND CCWANG

Problem 4

1. (15 points) Consider a version of the "maximum matching" problem, where there are $n \geqslant 100$ people on the left, 100 people on the right, certain pairs of them are willing to get married ("matched"), no one can get married twice, and we want to match the maximal number of people so that we end up with 100 marriages.

   Recall from class that you can set this up as a max-flow problem, by considering the people as a graph with 1 unit of capacity along each edge connecting a person on the left to a person they are willing to marry on the right; additionally, each person on the left has an edge from the source s with capacity 1 (representing the constraint that each person on the left can have at most 1 marriage), and each person on the right has an edge going to the sink t with capacity 1 (representing the constraint that each person on the right can have at most 1 marriage). The algorithm works by first running one of the maximum flow algorithms we have seen in class, and then marrying any two people that have flow between them.

   For this part, modify this construction so that it solves the problem with the additional constraint: the first 40 people on the left *must* get married. Prove that your construction is a correct algorithm.

   Please include a diagram. Your proof should be sure to touch on the following points: 1) any assignment returned by your algorithm is a valid marriage plan; in particular, be sure to point out that integer capacities on edges implies that the flow returned by the algorithms from class will have integer values; 2) any marriage plan could be converted into a flow that satisfies the constraints you describe, proving that your algorithm does not "miss out" on any good marriage plans.

   - Given bipartite graph $B = (A \cup B, E)$ in *Figure 1*, we have the following notation,

     $$\begin{aligned} P &= \text{the 40 vertices denoting people on the left we ensure they } \textit{must} \text{ get married} \\ Q &= \text{the other } n-40 \text{ vertices, representing the rest people on the left} \\ A &= P \cup Q, \text{ represents the } n \text{ people on the left} \\ B &= \text{the 100 vertices, representing the 100 people on the right} \end{aligned}$$

     such that every people in A has at least 1 directed edge going into B with capacity 1 and also every vertex in B has at least one edge directing into it from A with capacity 1.

     We add new vertices s and t, then we add an edge from s to every vertex in P and add an edge from every vertex in B to t. Make all the edge capacities 1. We add new vertice s' and add an edge from s to s' with capacity 60 and edges from s' directing into every vertice in Q with capacity 1. The resulting graph G is shown in *Figure 2*.

     We transform the matching problem to an instance of network flow problem and thus running

Ford-Fugerson algorithm would give us a maximum flow f from s to t.

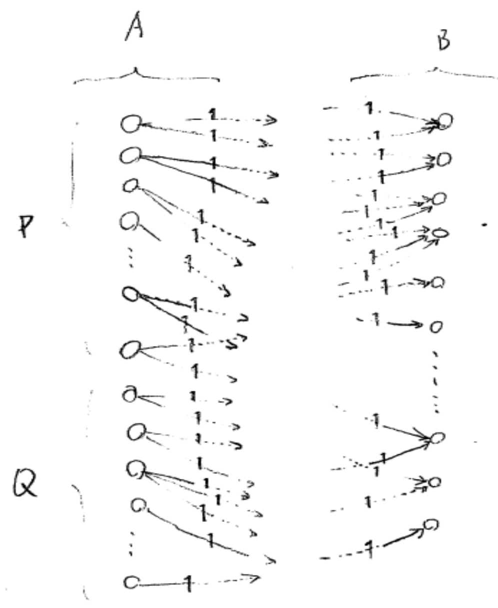

Figure 1.

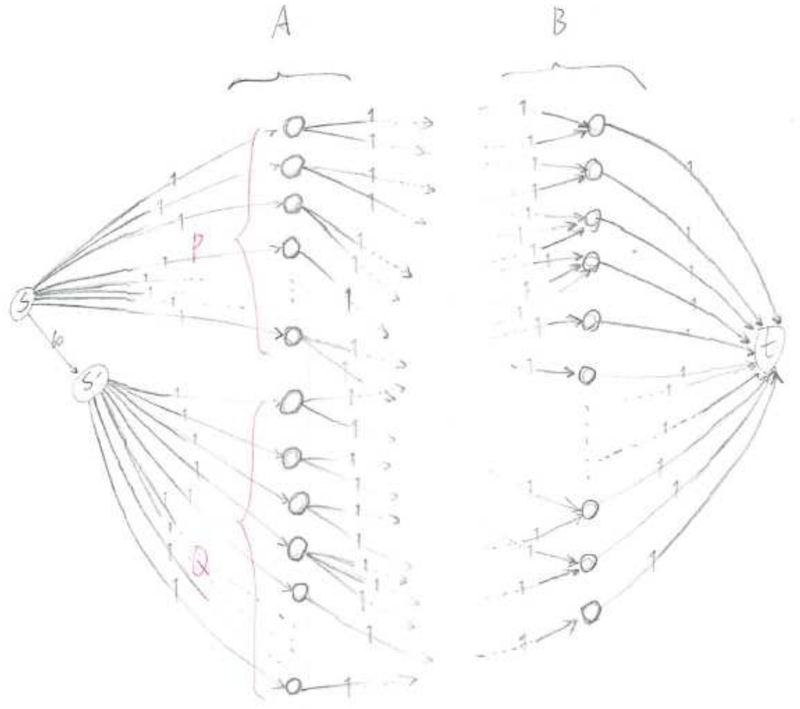

Figure 2.

- Here we are going to prove that the assignment returned by the algorithm is a valid marriage plan.

  Because the capacities are integers, our flow will be integral. Because the capacities are all 1, we will either use an edge completely or not use an edge at all.

  Let M be the set of edges going from A to B that we use. We will show that (i) M is a matching, (ii) M is the largest possible matching.

  i. M is a valid marriage plan

     We can choose at most one edge leaving any vertex in A. We can choose at most one edge reaching vertex in B. If we chose more than 1, we couldn't have balanced flow between A and B.

  ii. M is the largest possible matching

     The correspondence between flows and matchings is — if there is a matching of k edges between A and B, the resulting flow f, starting from s and sinking into t, would be of value k, where f has 1 unit of flow across each of the k edges and $\leqslant 1$ unit leaves (from A) and enters (into B) each node; if there is a flow f of integral value k, there is a matching with k edges between A and B.

     We find the maximum flow f with k edges and this corresponds to a matching M of k edges. If there were a matching with $>k$ edges, we could have found a flow with value $>k$, contradicting that f was maximum. Hence M is the maximum matching.

- Now we are going to prove that after solving the maximum network flow problem on graph G, if we get the maximum 100 marriages the first 40 people on the left *must* get married.

  Here we prove by contradiction.

  The contradiction hypothesis is that there is one of the 40 people on the left cannot get married.

  According to the correspondence between flows and matchings in (ii), then there would be no edge going into this node from s and the value going out of s would be less than 100 so now min cut in G is less than 100, this contradicts with the *Theorem* that the maximum s-t flow value cannot exceed the minimum s-t cut.

  So the first 40 people on the left must get married.

2. (10 points) Consider the (apparently very different though actually very similar) problem of scheduling weekly TA office hours in a course. Suppose there are n TAs on the course staff and m non-overlapping one-hour slots throughout the week during which TAs may hold office hours; however, no more than one TA can be assigned to each hour. Each TA i is available only during some subset $T_i \subseteq \{1, ..., m\}$ of the weekly slots; further, each TA has to hold at least 2 hours every week. Your task is to assign TA staff to TA hours in a way that satisfies all aforementioned constraints.

   Hint: Design your algorithm as an adaptation of your algorithm for the previous part, using a max-flow algorithm as a subroutine (how do each of the elements of this problem correspond with elements from the problem of the previous part?). You might want to first consider the problem without the constraint that each TA must cover at least two hours.

   Include a proof of correctness, though feel free to make use of concepts from the previous part to avoid repeating yourself.

Also, analyze the run time of your algorithm here in terms of n and m. You might want to check **http://en.wikipedia.org/wiki/Maximum_flow_problem#Solutions** for reference on the running time of max-flow algorithms.

- We can transform the maximum matching problem to the network flow problem where the graph is shown in *Figure 3* with the notation that

  $$\begin{aligned} A &= \text{the } n \text{ vertices representing the } n \text{ TAs} \\ B &= \text{the } m \text{ vertices, representing the } m \text{ non-overlapping one-hour time slots} \end{aligned}$$

  Each edge between s and any node in A is of capacity 2. Each edge between t and any node in B is of capacity 1. Any edge between A and B is of capacity 1.

  We run the *Ford-Fulkerson* algorithm in the graph G we get the maximum s-t flow f.

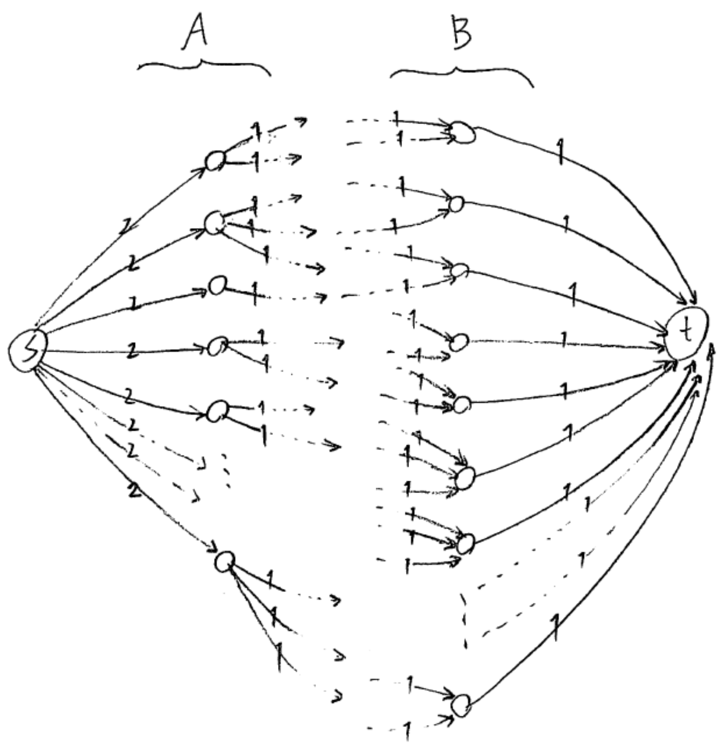

Figure 3.

- Here we are going to prove that the resulting flow could be converted into a valid TA hourse assignment plan.

  Because the capacities are integers, our flow will be integral. Because the capacities of edges between A and B are all 1, we will either use an edge completely or not use an edge at all.

  Let M be the set of edges going from A to B that we use. Using the same method as 4.1 we can show that (i) M is a matching, (ii) M is the largest possible matching.

- Now we are going to prove that after solving the maximum network flow problem on graph G in *Figure 3*, if we get the maximum $2n$ TA hours matching every TA would have exactly 2 hours assigned.

  We intend to prove by contradiction.

The contradictary hypothesis is that there is one of the TAs cannot get 2 hours assigned.

According to the correspondence between flows and matchings in (ii), then there would be an edge going into this corresponding vertex from s and the total value going out of s would be less than $2n$ so now min cut in G is less than $2n$, this contradicts with the *Theorem* that the maximum s-t flow value cannot exceed the minimum s-t cut.

- The running time of *Ford-Fulkerson* is $O(|E| \cdot |f|)$ where $|E|$ is the number of edges of G and $|f| = \sum_{e \text{ leaving } s} c_e = |A| = 2n$. There would be at most $n+m+nm$ edges in G. So the running time is $O(n^2 + nm + n^2m)$.

3. For 5 points extra credit, solve problem 4.2 with the *additional* constraint that the TAs, collectively, must hold exactly H hours per week. ($H = 2n$ is the easy case; $H < 2n$ is trivially impossible; $H > 2n$ is, in some sense, analogous to problem 4.1)

- For *Case 1*, $H = 2n$, the algorithm we above for 4.2 would give us exactly $2n$ TA hours assigned to n TAs. For *Case 2*, $H < 2n$, there will be TAs cannot be assigned at least 2 hours and thus is impossible.

- For *Case 3*, $H > 2n$, similar to 4.1, we can transform the maximum matching problem to the network flow problem where the graph is shown in *Figure 4* with the notation that

$$\begin{aligned} A &= \text{the } n \text{ vertices representing the } n \text{ TAs} \\ B &= \text{the } m \text{ vertices, representing the } m \text{ non-overlapping one-hour time slots} \end{aligned}$$

Each edge between s and any node in A is of capacity 2. Each edge between s' and any node in A is of capacity $H - 2n$ and also the edge from s to s'. Each edge between t and any node in B is of capacity 1. Any edge between A and B is of capacity 1.

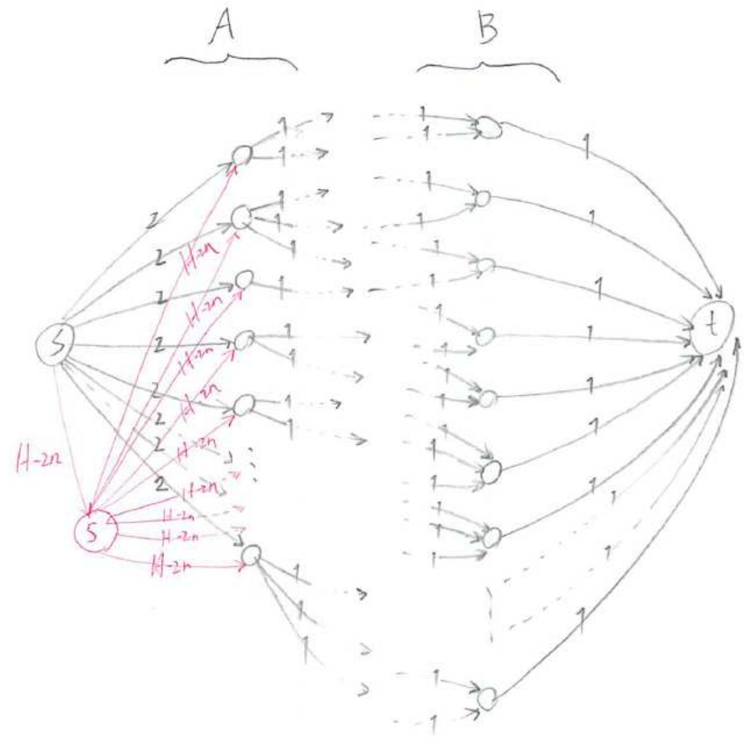

Figure 4.

- We need to prove that the resulting flow could be converted into a valid TA hours assignment plan such that let M be the set of edges going from A to B that we use, (i) M is a matching, (ii) M is the largest possible matching.

  The proof is similar to *Case 1* illustrated in 4.2 above.

- Now we are going to prove that after solving the maximum network flow problem on graph G in *Figure 4*, if we get the maximum H TA hours matching, every TA would have exactly at least 2 hours assigned satisfying the constraints.

  We intend to prove by contradiction.

  The contradictary hypothesis is that there is one of the TAs cannot get 2 hours assigned.

  According to the correspondence between flows and matchings in (ii), then there would be at most 1 edge going into this vertex from either s or s'.

  (1) If there is no edge directing into it, the total value going out of s would be at most $2n - 2 + (H - 2n)$, that is $H - 2$, so now min cut in G is less than H. This contradicts with the *Theorem* that the maximum s-t flow value cannot exceed the minimum s-t cut.

  (2) If there is only one edge directing into it from s', there would be no edge going from s and thus the out-going value of s would be at most $2n - 2 + (H - 2n)$, which is less than H for sure, in the same manner as (1) it contradicts with the *Theorem*.

  (3) If there is only one edge directing into it from s, the total out-going value of s would be at most $2n - 1 + (H - 2n)$, which is less than H and thus contradicting the *Theorem* the same as (1) and (2).

  So our hypothsis doesn't hold and every TA would have at least 2 hours assigned.