CS 381 Introduction to the Analysis of Algorithms
Fall 2021 Simina Branzei and Alexandros Psomas HW 5

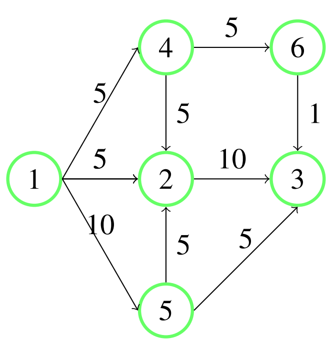
Due Fri November 12 at 11:59PM

1. 10 points Prove using induction that the recurrence $T(n) = T(n-1) - T(n-2) + n^2 - 2n + 3$ with $T(1) = 1$ and $T(2) = 4$ exactly equals n^2 for $n \geq 3$.

2. 15 + 4 + 3 + 3 points Answer the following questions for the graph given below:

   - Run the Ford-Fulkerson algorithm on the graph. You need to present the residual graph after each step i.e., after you send the flow along a specific path. To keep solutions consistent, assume Ford-Fulkerson uses DFS prioritizing lower numbers to choose its augmenting paths. (e.g. if the only valid paths are 1463 and 153, it will choose 1463)
   - State the max flow for the given graph.
   - If the edge from vertex 4 to vertex 6 has its weight changed from 5 to 10, what would the maximum flow be?
   - If the edge from vertex 6 to vertex 3 has its weight changed from 1 to 2, what would the maximum flow be?

   The vertex 1 is the source, while the vertex 3 is the sink.



3. 25 points

   In a certain hospital, there are $2n$ doctors and in the next month there are $3n$ surgeries planned. Each doctor i has reported an array S_i of the surgeries she is qualified to perform. For example, if doctor 1 can perform surgeries 1, 3, and 4, but not any others, then $S_1 = [1,3,4]$. Each surgery requires 2 qualified doctors. Each doctor only has time to participate in at most 3 surgeries.

   Find and describe a max-flow-based algorithm to find an assignment of doctors to surgeries. If not every surgery can be fully staffed with 2 qualified doctors, fill as many slots as possible (i.e. two surgeries with one qualified doctor each is exactly as good as one fully staffed surgery and one completely unstaffed surgery). Then analyze and prove your algorithm's runtime and correctness.

   You may use max flow as a subroutine without explaining how it works (though you do need explain what the input graph is, how to interpret the output, etc.). If every edge capacity is an integer, you may assume max flow uses an integer amount of each edge. Assume that max flow requires $O(|E|C)$ time, where $|E|$ is the number of edges in the input graph and C is the value of the max flow.

   Here's an example:
   $n = 2$
   $S_1 = [1,2,3]$
   $S_2 = [2,5]$
   $S_3 = [1,2,4]$
   $S_4 = [1,2,3,4,5,6]$

   One correct answer would be as follows:

Surgery	Doctors assigned to it
1	1,3
2	1,2
3	1,4
4	3,4
5	2,4
6	

   with total 10 slots filled, since 11 is not possible. Another correct answer would be

Surgery	Doctors assigned to it
1	1,3
2	1,2
3	1
4	3,4
5	2,4
6	4

   since it also has 10 slots filled.

4. 25 points

   There is a large convention which wants to coordinate assignments of its $100n$ attendees to n affiliated hotels. Each attendee wants to stay at one hotel. Each hotel can accommodate 100 attendees. Each attendee i reports her preference list a_i; i.e. she prefers $a_i[1]$ over $a_i[2]$ over $a_i[3]$ ... over $a_i[n]$. Each hotel j reports its preference list h_j in a similar way.

   Describe a modification of the Gale-Shapely algorithm to find a stable assignment of attendees to hotels; i.e.

   - Every attendee is assigned to one hotel
   - Every hotel has 100 attendees assigned to it
   - There does not exist an attendee i and hotel j such that
     - i is not assigned to j
     - i prefers j over the hotel she was assigned to
     - j prefers i over its least preferred assigned attendee

   Then analyze and prove your algorithm's runtime and correctness.

5. 5 + 5 + 5 points

   Consider the Reverse-Delete algorithm for finding an MST:

   (a) Sort all edges by weight

   (b) For each edge e in descending weight order:

   (c) If removing e would not disconnect the graph, delete e

   (d) After considering all edges, return the remaining graph

   Prove the following statements relating to the Reverse-Delete algorithm for finding the minimum spanning tree. You may assume the graph is undirected, has distinct edge weights, and is connected. You may use the property that (in graphs with distinct edge weights) no MST contains the heaviest edge in a cycle. You may use each statement as part of your proofs for the statements after it.

   - If an edge belongs to a cycle, then deleting it will not disconnect the graph
   - The Reverse-Delete algorithm will delete the heaviest edge from every cycle
   - Reverse-Delete always returns an MST