Instructions.

- Please write your answers concisely.

Que 1. Consider the flow network in Figure (1a) with a source $s$ and a sink $t$. The edge capacities are shown on the edges. Figure (1b) shows an $s$-$t$ flow in the same network with flow values indicated on the edges.

(a) A flow network with capacities on the edges
(b) A flow network, with current flow values shown on the edges.

Figure 1: Network Flow

(a) [1 mark]. Suppose the initial flow is given by Figure (1b). Construct the residual graph with respect to this initial flow. Recall that any forward/backward edges with zero capacity are not kept in the residual graph.

(b) [1 mark]. Is there a path from $s$ to $t$ in the residual graph? Describe the path. What is the bottleneck $b$ (minimum capacity of an edge) on this path?

(c) [1 mark]. Suppose we push $b$ units of flow along this path. Show the flow network with new flow values.

(d) [1 mark]. Construct the residual graph with respect to the new flow. Is there a path from $s$ to $t$ in this residual graph?

(e) [2 mark]. What is the current outgoing flow from $s$? Construct an $s$-$t$ cut in the network whose outgoing capacity is equal to the current outgoing flow from $s$.

Que 2. Consider the problem of TA (teaching assistants) allocation to courses. A TA may be suitable for a subset of courses and not suitable for other courses. Suppose we are given the suitability information by a $m \times n$ matrix with \{0, 1\} entries, where $m$ is the total number of TAs and $n$ is the number of courses. We are also given the required number of TAs for each course, say, $r_1, r_2, \ldots, r_n$. Naturally, a TA can only be assigned to a course for which they are suitable. And a TA can be assigned to at most one course. We want to find out whether it’s possible to meet the TA requirement for every course.

For example, suppose for course $X$ the suitable TAs are \{1, 2, 3\}. And for course $Y$, the suitable TAs are \{2, 3\}. Suppose $X$ and $Y$ both require two TAs each. Then it is not possible to meet the requirement for every course.

We plan to solve this problem using the network flow algorithm as a subroutine. For any given input for TA allocation (suitability matrix and $r_i$’s), we want to first build an appropriate flow network. We should design the network in a way that the value of the maximum flow should tell us whether it’s possible to meet the TA requirement for every course.
Building the flow network: We create one vertex for every TA and one vertex for every course. We add a (directed) edge from a course to a TA if and only if that TA is suitable for that course. Additionally, we also create a source vertex $s$ and a sink vertex $t$.

(a) [1 mark]. Where will you put the outgoing edges from $s$ and what will be their capacities?

(b) [1 mark]. Where will you put the incoming edges to $t$ and what will be their capacities?

(c) [1 mark]. What should be the capacity of the edges which are going from a course to a TA?

(d) [1 mark]. If it is possible to meet the TA requirement for every course, then what will be the maximum flow in your network?

(e) [1 mark]. If it is not possible to meet the TA requirement for every course, then what can you say about the maximum flow in your network?

(f) [4 marks]. Prove that when it is not possible to meet the TA requirement for every course, then there must be a subset $C$ of courses such that the total number of TAs who are suitable for some course in $C$ is less than the required number $\sum_{i \in C} r_i$. You can directly use the max flow min cut theorem (but not other theorems like Hall’s theorem, Dilworth’s theorem).

Hint: Say $U$ is a minimum $s$-$t$ cut ($U$ contains $s$). Consider the set of courses in $U$.

Que 3. Damerau-Levenshtein distance between two strings (over any alphabet) is defined as the minimum number of valid operations required to convert one string into another, where the valid operations are – deletion of a character, insertion of a character, substituting one character with another, and swapping of two adjacent characters. There is no restriction on the order in which these operations are performed. For example, the two strings from and north have distance 4, as shown below.

from $\leftrightarrow$ form $\leftrightarrow$ norm $\leftrightarrow$ norh $\leftrightarrow$ north

We want to design a dynamic programming algorithm to compute the Damerau-Levenshtein distance between two given strings. Let the two given strings be $a_1a_2\cdots a_n$ and $b_1b_2\cdots b_m$, where each $a_i$ and $b_j$ belong to an alphabet $\Sigma$.

Let $d(i,j)$ denote the distance between the substrings $a_1a_2\cdots a_i$ and $b_1b_2\cdots b_j$, for $0 \leq i \leq n$ and $0 \leq j \leq m$ ($i = 0$ or $j = 0$ just means empty substring).

We will build an $(n+1) \times (m+1)$ table $D$ whose $(i,j)$ entry is supposed to be $d(i,j)$ for $0 \leq i \leq n$ and $0 \leq j \leq m$. Finally, $D(n,m)$ will be our answer.

Set $D(0,i) = i$ and $D(j,0) = j$ for each $i$ and $j$.

To fill the rest of the table we use the following equation

\[
D(i,j) = \min \begin{cases} 
D(i-1,j) + 1 & \text{(deletion)} \\
D(i,j-1) + 1 & \text{(insertion)} \\
D(i-1,j-1) & \text{considered only if } a_i = b_j \\
D(i-1,j-1) + 1 & \text{(substitution)} \\
D(i-2,j-2) + 1 & \text{considered only if } i,j \geq 2, \text{ and } a_i = b_{j-1} \text{ and } a_{i-1} = b_j \text{ (swapping)}. 
\end{cases}
\]

It turns out that this approach is wrong. 😞

Find an example of two strings, where this algorithm will output a larger number than the Damerau-Levenshtein distance (2 marks). Fill up the table $D$ completely for your example, as per the above equation (2 marks). Show the correct distance between the two strings in your example, via a sequence of valid operations (1 mark).

Hint 1: A similar approach would have worked correctly, if we didn’t have the swapping operation in our distance definition.

Hint 2: There is an example where both the strings have length at most 3.