Solution for Homework 2

Prob 1. Similar to edit distance problem, we use LCS(i, j) to denote the length of LCS between $A[1..i]$ and $B[1..j]$. Then the formula is

$$\text{LCS}(i,j) = \begin{cases} 0 & i = 0 \text{ or } j = 0 \\ \text{LCS}(i-1, j-1) + 1 & a_i = b_j \\ \max(\text{LCS}(i-1, j), \text{LCS}(i, j-1)) & \text{otherwise} \end{cases}$$

and LCS(n, m) is what we want.

For the second problem, denote B be the reverse of A and use previous formula to solve the LCS between A and B. One can check LCS$(i, n-i)$ and LCS$(i, n-i-1)$ to derive what we want.

Prob 2. For simplicity we add another edge from sink t to source s with no capacity constraint. Thus every node should satisfy the flow conservation condition.

$$\begin{aligned} \max_{f} \quad & f_{t,s} & \\ \text{s. t.} \quad & f_{u,v} \le c_{u,v} & \forall (u,v) \in E \text{ except } (t,s) \\ & \sum_{w} f_{w,u} - \sum_{v} f_{u,v} = 0 & \forall u \in V \\ & f_{u,v} \ge 0 & \forall (u,v) \in E \text{ and } (t,s) \end{aligned}$$

Prob 3. By definition the dual is

$$\begin{aligned} \min_{p,y} \quad & \sum_{(u,v) \in E \text{ except } (t,s)} y_{u,v} c_{u,v} & \\ \text{s. t.} \quad & p_v - p_u + y_{u,v} \ge 0 & \forall (u,v) \in E \text{ except } (t,s) \\ & p_s - p_t \ge 1 & \\ & y_{u,v} \ge 0 & \forall (u,v) \in E \text{ and } (t,s) \end{aligned}$$

If this linear programming has an optimal solution where every variable is 0 or 1 then it's quite clear a min-cut problem.

Given an $0, 1$-solution p^* and y^*, if we remove all the edges whose $y_{u,v}$ are 1. Then all the edges remained should satisfy $p_v - p_u + y_{u,v} \ge 0$, but

$y_{u,v} = 0$ implies $p_v - p_u \geq 0$ and thus every $v \in V$ reachable from u must have a greater p value. But $p_s - p_t \geq 1$ then there is no path from s to t.

For another direction, given an $s - t$ cut we can construct a $0, 1$-solution to this LP. Just let $y_{u,v}$ be a $0, 1$-indicator of whether an edge belongs to the cut. Then set $p_u = 1$ for every u still reachable from s otherwise just set $p_u = 0$. Then it's clear a $0, 1$-solution to the LP.

Prob 4. Denote $\#(A)$ be the number of such (i, j) pairs in an integer array A. Suppose we break A into two halves L and R. Clearly that $\#(A) = \#(L) + \#(R) + c(L, R)$ where $c(L, R) = |\{(l, r) : l \in L, r \in R, l > r\}|$. For $\#(L)$ and $\#(R)$ we can solve it recursively.

To calculate $c(L, R)$, suppose we have L and R sorted. Then we use the merge procedure in merge sort to get A sorted. Each time when we pick en element in R, we will find k such pairs where k is the number of elements remained in L.

So to summarize, we can use merge sort to sort A, and the summation of all k during the merge procedure is the answer.