Problem 1

(a) Proof by contradiction. Suppose both (1) and (2) hold. Then multiply y to both sides of (1) we have $yAx = yb$. (2) provides $yA \geq 0$ and (1) has $x \geq 0$, then we have $yAx \geq 0$, which contradicts with $yb < 0$ in (2). $\square$

(b) (2) can never be infeasible. Notice that the constraint is $yA \geq 0$. We can always set $y := 0$ and obtain $yA = 0 \geq 0$ satisfying the constraint for any A. $\square$

(c) As in (b), the associated linear program is

$$\begin{aligned} \min\ & yb \\ \text{s.t. } & yA \geq 0 \\ & y \text{ UIS}, \end{aligned}$$

then the dual is

$$\begin{aligned} \max\ & 0x \\ \text{s.t. } & Ax = b \\ & x \geq 0, \end{aligned}$$

where we know the primal is either *unbounded* or has a real number as solution.

Therefore, when the primal is unbounded, i.e. (2) is feasible (yb can be arbitrarily small, thus smaller than 0), the dual is infeasible, i.e. (1) is infeasible, because of strong duality.

On the other hand, when the primal is feasible, we know the dual is feasible, i.e. (1) is feasible. Moreover, the feasible solution for primal is *always* $yb = 0$. It can not be negative. This is because part of y would then be non-zero and we must be able to tune the parameter in b to make yb smaller than the feasible solution, as b is unconstrained. Therefore we reach $yb = 0$ being infeasible for (2). $\square$

Problem 2

(a) Putting "polyhedra P and Q has no intersection" as a *dual* of a linear program

$$\begin{aligned} \max\ & 0x \\ \text{s.t. } & Ax \le b \\ & Dx \le e \\ & x \text{ UIS}, \end{aligned}$$

then the primal is

$$\begin{aligned} \min\ & yb + ze \\ \text{s.t. } & yA = 0 \\ & zD = 0 \\ & y \ge 0 \\ & z \ge 0. \end{aligned}$$

If the polyhedras don't have intersection, it means the dual is infeasible, which means the primal is unbounded (i.e., $yb + ze$ can be arbitrarily small). In order words, we know there are $y, z \ge 0$ such that $yA + zD = 0$ and $yb + ze < 0$. $\square$

(b) We show the existence of such a c by showing $c = yA$ is a desirable solution. From (a) we know $c = yA = -zD$, it suffices to show $yAx + zDw < 0$. We know $Ax \le b$ and $Dw \le e$ from the polyhedras. Then we know $yAx + zDw \le yb + ze < 0$, also from (a). Therefore such a c can be a solution, the existence thus holds. $\square$

(c) If the polyhedra does not have a point in common, there exists a separating hyperplane, which can be found using (b). If it has intersecting points, then such c does not exists. Looking back to (a), this equivalently means, if the primal is unbounded, then the dual is infeasible, which means no points in common. Otherwise, if the primal is feasible (it can't be infeasible, because we can always set $y, z = 0$ and get a feasible solution), it means the dual is also feasible. Moreove, $yb + ze = 0$ because otherwise we could have make the value smaller as there are some nonzero entries in b or e (similar idea in Problem 1(c)).

Therefore, the easy-to-verify criteria is, whether or not the primal in (a) is bounded.

Problem 3

Rewrite the flow constraint with P flows and m edges more explicitly, using a $P \times m$ flow-edge matrix, where each column i contains a $0-1$ value indicating if a flow passes through the edge i. The max flow problem (we write out a specific 0-1 matrix as an example) is written as

$$\begin{aligned}
&\text{maximize } \sum_{p=1}^{P} f_p \\
&\text{s.t.} \\
&\begin{bmatrix} f_1 & f_2 & \dots & \dots & f_P \end{bmatrix} \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \dots & 1 \end{bmatrix} \leq \begin{bmatrix} u_1 & u_2 & \dots & u_m \end{bmatrix} \\
&\begin{bmatrix} f_1 & f_2 & \dots & \dots & f_P \end{bmatrix} \geq \begin{bmatrix} 0 & 0 & \dots & \dots & 0 \end{bmatrix}.
\end{aligned}$$

in this example, we can see flow 1 and flow P pass through edge 1, flow 2 passes through edge 2 and flow P passes through edge m. In general the entry M_{ij} is 1 if flow j passes through edge i, 0 otherwise.

The dual[1] of the problem becomes

$$\begin{aligned}
&\text{minimize } c_1u_1 + c_2u_2 + ... + c_mu_m \\
&\text{s.t.} \\
&\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \dots & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 1 \end{bmatrix} \\
&\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}
\end{aligned}$$

[1] We actually set the original max-flow as the dual, and construct the min-cut problem as primal here

Using the concise expression as in the problem, this becomes

$$\begin{aligned} \text{minimize } & \sum_{e=1}^{m} c_e u_e \\ \text{s.t.} & \sum_{e \in P} c_e \geq 1 \qquad \text{for all path } P, \\ & c_e \geq 0 \qquad\quad \text{for all edge } e. \end{aligned}$$

The dual problem is essentially the min-cut problem.

The first constraint encodes "distance" of all paths where c_e is like a distance of each edge. This means for each path, the distance is at least 1. The second constraint says the edge has non-negative length. The objective is using the length of edge as a "weight" so as to minimize the weighted sum of edge capacity.

Intuitively, the optimal solution is to assign the length = 1 for edges on the min-cut boundary and 0 for other edges. Then, in the max flow, each path will cross the boundary once therefore has distance 1, satisfying the constraints. Because of the min-cut property, we know the objective is minimized. Fractional solution is sub-optimal or equals to min-cut solution as the linear problem has extreme point as the solution.

Problem 4

(a) The first constraint ensures flow conservation on each node in the graph. The last constraint means flow being non-negative. The second constraint basically means there needs to be flow in the graph. Therefore, if the problem is feasible, there is a circulation in the graph, as each node obeys conservation law and there is flow through.

The objective is to minimize the cost of total flow on the edges. Running this optimization will "delegate" the flow from circulation (as in the constraint) to the *minimum mean cycle* (as in the objective). This is because the circulation can be decomposed into cycles. For the cycles that have a higher mean cycle (i.e., doing the delivery work there earns less efficiently than some other cycle), the flow on those cycles will be delegated to the one with minimum mean cycle, so as to minimize the objective.

For formally, consider two cycles with length l_1, l_2 and total cost c_1, c_2, suppose $c_1/l_1 < c_2/l_2$. Then $c_1 f/l_1$ is strictly smaller than $c_2 f/l_2$, where f is the *total* sum of flow along each edges in the cycle (sum of f_{il} for the cycle). Therefore the optimal solution distribute all flows to the *minimum mean cycle*.

(b) The dual is

$$\begin{aligned} &\max \lambda \\ &\text{s.t. } p_i - p_j + \lambda \le c_{ij} \qquad \forall i, j \\ &\quad p_i, \lambda \text{ UIS.} \end{aligned}$$

(c) Consider the constraint to be $p_j - p_i + c_{ij} - \lambda \ge 0$. The optimization problem says "find largest λ such that feasible prices $p_i's$ exists". The largest possible λ is actually *average cost of minimum mean cycle over the edge length.*

For *any* cycle in the graph, we can telescope around the cycle and cancel out all $p_i's$ and obtain $\sum(c_{ij} - \lambda) \ge 0$. Now, for all these cycles, at least one of them has to saturate, because otherwise we can increase λ and still satisfy all constraints, contradicting with maximization of λ. For the cycle that $\sum(c_{ij} - \lambda) = 0$, we have $\sum c_{ij} = k\lambda$, where k is the cycle length. Therefore λ is "upper bounded" by the minimum mean cycle $\sum c_{ij}/k$ and the optimization reaches it.

(d) We combine binary search and min-cost max-flow, which are both combinartorial algorithms to find λ. First, notice that $\lambda \in [0, \sum |c_{ij}|]$, which is a finite region. Then we binary search λ in the region. At each step t and region $[l^t, r^t]$, we subtract λ from all costs in the edges, and run min-cost max-flow algorithm to see if there is still cycles. Notice that in (b), the constraint itself with a given λ is a problem finding feasible prices in the graph with modified edge cost. If there is negative cost cycle, it means feasible price does not exist. Therefore, if the flow algorithm

returns a cycle, we need to reduce λ and therefore search in the smaller half of the region ($l^{t+1} = l^t, r^{t+1} = (l^t + r^t)/2$); otherwise, search in the larger half of the region ($r^{t+1} = r^t, l^{t+1} = (l^t + r^t)/2$).

The search terminates when the search region is smaller than the smallest possible difference of two cycles in the graph. As we assume the costs are all integers, the cost of two cycles c_1, c_2 differ at least by 1. Then the smallest possible difference of *mean cycle* is bounded by $c_1/k_1 - c_2/k_2 = (c_1k_2 - c_2k_1)/(k_1k_2) \leq 1/(k_1k_2) \leq 1/m^2$, where m is total number of edges. Therefore, it suffices to terminate the binary search when the region size is smaller than $1/m^2$.

Problem 5

(a) Consider a standard LP problem

$$\begin{aligned} \min\ & c^T x \\ \text{s.t. } & Ax = b \\ & x \geq 0, \end{aligned}$$

and its dual

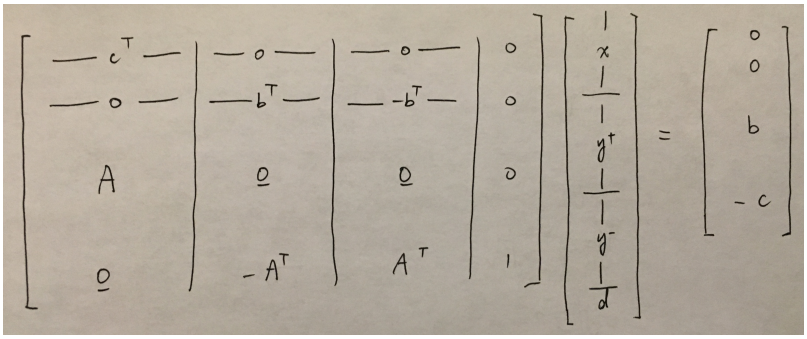
$$\begin{aligned} \max\ & b^T y \\ \text{s.t. } & A^T y \leq c. \end{aligned}$$

We know if the primal is feasible, so does the dual. From the strong duality, the min of the primal equals to the max of the dual. Therefore,

$$\begin{aligned} c^T x &= b^T y \\ Ax &= b \\ x &\geq 0 \\ A^T y &\leq c, \end{aligned}$$

which is a legitimate LP and can be transform to the standard form. Notice that here we don't have a objective to optimize.

Specifically, changing to $c^T x - b^T y = 0$ and $-A^T(y^+ - y^-) + d = c$, with $y^+ \geq 0, y^- \geq 0, d \geq 0$.



(b) The dual is

$$\begin{aligned} \max\ & b^T y \\ \text{s.t. } & A^T y \leq 0 \\ & y \ \text{ UIS.} \end{aligned}$$

(c) The solution is 0. This is because when the primal is feasible, dual is feasible and bounded. If $b^T y$ is less than 0, then we could have set $y = 0$ satisfying the constraint but make the objective 0 larger than the negative value. If $b^T y$ is larger than 0, then we could have *double* the value of y, still satisfying the constraints but *double* the objective and make it bigger.

(d) Using (c) we can pass to the algorithm with optimal solution 0 of dual in (b). Then in $O((m+n)^k)$ time if the algorithm returns and it checks the feasibility in (a), we know the optimal value of original problem is obtained.

If the return from the algorithm[2] does not pass the feasibility check in (a), it means the original problem is either infeasible or unbounded.

To check the feasibility, we use the same trick that removes the objective of the original problem and looks into its dual. The dual maximization is 0 if the primal is feasible. Therefore we invoke the algorithm on the modified original problem with 0 passed to its dual. If the returned result checks the feasibility test, the original problem is feasible (therefore unbounded because of above), otherwise it is infeasible.

The runtime is asymptotic in $O((m+n)^k)$ because the two calls of the algorithm each has $O(c(m+n)^k) = O((m+n)^k)$ and the checking of feasibility is also in $O((m+n)^k)$.

[2]the algorithm may not terminate in $O((m+n)^k)$ when the "solution" to the dual passed into the algorithm is wrong, in this case we return a random result.