CSCI 570 Spring 2020 Xinrui Ying (USCID 1092350440)
Homework 4 Problem

Solution:

1. The max-flow for the left graph is 7. The minimum $s-t$ cut are

   - {S} and { A,B,C,D,T }
   - {S,A,B,D} and {C,T}

   The feasible flows are

   - S->A->C->T with flow of 2
   - S->A->B->C->T with flow of 1
   - S->B->D->T with flow of 3
   - S->B->C->T with flow of 1

   The max-flow for the right graph is 7. The minimum $s-t$ cut are

   - {S} and {A,T,B}
   - {S,B} and {A,T}

   The feasible flows are

   - S->A->T with flow of 3
   - S->B->T with flow of 3
   - S->B->A->T with flow of 1

2. We would first construct the graph G of the grid. For each location in the grid, we will create two unique vertices v_{in} and v_{out}, and add an directed edge of weight 1 from v_{in} to v_{out}. For each pair of neighbor vertices (v_1, v_2) in the grid, we will add an edge from v_{2out} to v_{1in} and an edge from v_{1out} to v_{2in}. We would also have a starting vertex s, and an ending vertex t. s will have an outgoing edge of weight 1 to all the p persons' v_{in}. For all the boundary vertices' v_{out}, we will have an an outgoing edge of weight 1 to a new ending vertex t.

   The time of constructing the graph is $O(|V| + |E|)$ and Using Ford-Fulkerson algorithm to find the max-flow, the time complexity would be $O(|F|(|V| + |E|)) = O(|V||E|)$.

   Proof: We want to show that the max-flow is the same as the number of disjoint paths.
   If there exists p disjoint paths, it's simple to find p feasible flow of 1 in the graph by just following the paths.
   If the max-flow of graph G is p, if any of the two flows share a vertex, then because the edge between v_in and v_out is 1, the two flows will have sum of 1 but not 2, means there are p disjoint paths.

3. We would first construct the graph G, we would create a set of vertex v_i $\forall i \in N$. An edge from v_i to v_j with weight of τ_{ij}, and an edge from v_j to v_i with weight of τ_{ji}. Create a starting vertex s, and have outgoing edges to all v_i with weight of $q_i' + p_i$. Create an ending vertex t, and all v_i will have outgoing edges to t with weight of $q_i + p_i'$. Find the min $\{s, t\}$-cut and take the product in the first company that is in s set and take the product in the second company that is in t set.

   Proof:
   Let the optimal selection is A and B with all the selections from first company in A and all selection from second company in B. Based on the optimal selection, our s, t-cut must be the same, otherwise we will be able to change the cut and form a less overall flow.

4. We would first construct the graph G for this problem. For each lotus leave i we build a pair of vertex (v_{in}, v_{out}), and we add an directed edge of weight m_i from v_{in} to v_{out}. For different lotus leave i and j, if the distance between them is less than d, we create an edge of weight N from i's v_{out} to j's v_{in}, and we create an edge of weight N from j's v_{out} to i's v_{in}.

   If we want to prove whether all frogs can party at i's lotus leave, we simply make i's v_{in} be t, we create a new vertex s and add outgoing edges to all other j's v_{in} with weight of n_j for each edge. Afterward, we are looking for the max-flow from s to t, if it's equal to $N - n_i$, then lotus leave i can hold a party. We loop through 1 to N and create the output.

   Total time complexity will be $O(N(O(N) \times O(N^4))) = O(N^6)$.
   The proof: If lotus leave i can hold the party, then there exist a path for each frog in other lotus leave to jump to lotus leave i. Graph G follows the jump distance and also create constrain for each lotus leave, therefore, there will be corresponding flows in graph G.
   If there is flows of the graph matches the desired result, then all frogs from other lotus leave will jump to lotus leave i because each frog will only has flow of 1 given the construction.

5. We would construct the graph G first. For each course i, we will create a vertex c_i for it. For each $[s_i, e_i]$ $\forall i \in N$, we separate the interval to small intervals of size 1, if the small interval doesn't have a representing vertex, we will build a vertex k for it, and k will have an outgoing edge of size 1 to c_i. We also will create a starting vertex s and s will have outgoing edges of size 1 to all the k vertices we built. After this, we will create an ending vertex t and all the c_i will have an outgoing edge of weight w to t.

   Now, let number of k vertices be K, because we have N subjects, so the best grade we will have will be K/N, we will set the $w = K/N$. We will calculate the max-flow of graph G and if the max-flow is equal to $w \times K$, then the weight is the result; if it's not, we will change w to $w - 1$ and continue the process.

   The total time is $O(|V|^2|E|^2)$.
   Proof:
   If the maximum grade is f, when we set the $w = f + 1$, there will be some c_i cannot reach the flow of $f + 1$, therefore max-flow won't equal to $w \times K$. When $w = f$, we will have $w \times K$ otherwise the maximum grade cannot be f.
   If when we set $w = f$ and it finally reached that max-flow equal to $w \times K$. Then it means all c_i have flow value of f, similar saying maximum grade can be f.

6. For each row we will create a vertex r_i and for each column we will create a vertex called c_j. We create a starting vertex of s, s will have outgoing edges to all r_i with value of sum of row r_i. There should also be an ending vertex t, all c_j will have outgoing edges to t with value of sum of column c_j. For each r_i, r_i will have an outgoing edge to c_j, the weight of the edge will be either the floor or the ceiling of the value at $M[i, j]$. t will have an outgoing edge of weight infinity to s, and thus form a circulation. Find if the max-flow equal to the total sum and the path of each flow will be the rounding value.

   The proof:
   If there exist a rounding, then we follow the rounding and just set the flows with value of the rounding and the total sum will match the max-flow.
   If the total sum equal to the max-flow, we simply find the flows for each path and because of the option between row node and column node are the number after rounding, the result will be a valid rounding.

7. We first find the edges of min-cut. Adding 1 to these edges one by one and if it occurs an augmenting path, then the edge should belong to S.

   Finding min-cut sets can be found by creating the residual graph through the process of Ford-Fulkerson or Edmonds-Karp. The complexity for this process is $O(|E|^2|V|)$. Adding 1 to the the edges in the min-cut one by one and see if it creates an augmenting path afterwards, so overall it takes at most $O(E)$ iterations. Therefore, overall polynomial.

   Proof: We know that min-cut is equal to max-flow. If we have augmenting path remaining in the graph, meaning we can have extra flows. Then, the min-cut will increase as well. So, if the augmenting path exists after increases the edge weight, min-cut will increase and thus the edge is in S.

   ■