CS 4820, Spring 2018	**Homework 7, Problem 1**
Name: Rongguang Wang	
NetID: rw564	
Collaborators: Siyuao Liu (sl2928); Yihao Chen (yc2288)	

(1) *(10 points)* Suppose we are given an $n \times n$ matrix $A = (a_{ij})$ with non-negative entries, and we are interested in the question of whether it can be expressed as a sum of two matrices R, C such that:

1. The entries of R and C are non-negative.
2. The row sums of R are bounded above by 1.
3. The column sums of C are bounded above by 1.

When there exist matrices R, C satisfying these three constraints, such that $R + C = A$, let us call the pair (R, C) a *row-plus-column (RPC) decomposition of A.*

Design a polynomial-time algorithm that takes a non-negative matrix A as input, and outputs either an RPC decomposition or a set of rows, S, and a set of columns, T, such that

$$\sum_{i \in S, j \in T} a_{ij} > |S| + |T|.$$

(In the latter case it follows easily that there is no RPC decomposition of A, since the row set S and column set T identify an $|S| \times |T|$ submatrix of A whose entries sum up to more than $|S| + |T|$, whereas the entries in the corresponding submatrices of R and C must have sums bounded above by $|S|$ and $|T|$, respectively.)

Solution:

The problem is aiming to find an algorithm which can compute the row-plus-column (RPC) decomposition of a given matrix A. The row-plus-column (RPC) decomposition refers to the matrix pair (R, C) which has the property of $R + C = A$. Additionally, there are there constraints for matrix R and C. Firstly, the entries of R and C are non-negative. Secondly, the sum of each row in R is less than or equals to 1. Finally, the sum of each column of C is less than or equals to 1. The matrix R and C should be outputted if there is a RPC decomposition for matrix A. Otherwise, a set of rows S and a set of columns T should be outputted which have the property $\sum_{i \in S, j \in T} a_{ij} > |S| + |T|$. $|S|$ and $|T|$ corresponding to the size of S and size of T. $\sum_{i \in S, j \in T} a_{ij}$ refers to the summation of the intersection entries of set S and set T.

This problem can be reduced to Ford-Fulkerson Algorithm with the Maximum-Flow Minimum-Cut (Max-Flow Min-Cut) Theorem. A graph G should be created for the Ford-Fulkerson Algorithm. Firstly, a source node s and a sink node t should be created. Then, each entry a_{ij} of the $n \times n$ matrix A corresponding to a node v_{ij} in the graph G. Since we need to satisfy the constraints 2 and 3 ($\sum_{i=1}^{n} a_{ij} \leq 1 |\ j = 1, 2, \ldots, n$ and $\sum_{j=1}^{n} a_{ij} \leq 1 |\ i = 1, 2, \ldots, n$), n row nodes r_i and n column nodes c_j should be created for each row and column. Next, the edges between the nodes should be connected. The source node s should connect to all entry nodes v_{ij} with capacity a_{ij} which corresponding to the entry value in the matrix A. It is noticeable that each entry value of the matrix should be less or equals to 2. Otherwise, there will be definitely no RPC decomposition for it. Therefore, we assume that each entry value of the matrix A is less than or equals to 2. Additionally, each entry node v_{ij} should connect to both the row node r_i and column node c_j in which row and column it located in. The capacities for

either edges are positive infinity ($+\infty$). In this case, each entry of the matrix A was divided into the row sums matrix R and column sums matrix C which corresponding to the flow on the edges between each entry node v_{ij} and the row nodes r_i, and the flow on the edges between each entry node v_{ij} and the column nodes c_j. Finally, each row nodes r_i and each column nodes c_j should connect to the sink node t with capacity 1, which satisfies the constraints 2 and 3 that the sum of each row in R is less than or equals to 1 and the sum of each column of C is less than or equals to 1.

The graph G was created and run the Ford-Fulkerson Algorithm to find the maximum flow f. It is noticeable that all flow value on the edges of the graph G should be positive integers. Otherwise, the algorithm may not terminate. If all the edges between the source s and each entry node v_{ij} in the maximum flow f are saturate (flow value equals to capacity value), then the RPC decomposition was found which are the flow on the edges between each entry node v_{ij} and the row nodes r_i, and the flow on the edges between each entry node v_{ij} and the column nodes c_j. Otherwise, there is no RPC decomposition for matrix A. We should output set of rows S and set of columns T. Since the sum of capacities go out of the source node s is $\sum_{i=1,j=1}^{n} a_{ij}$, the maximum flow f must equals to $\sum_{i=1,j=1}^{n} a_{ij}$. This implies that the flow $\bar{f}$ which is not a maximum flow and matrix A has no RPC decomposition, is less than $\sum_{i=1,j=1}^{n} a_{ij}$. This satisfying the requirement of the output S and T that $\sum_{i\in S, j\in T} a_{ij} > |S| + |T|$. We can find the set of rows S and set of columns T from the minimum-cut (Min-Cut) of the network flow $\bar{f}$. According to the Maximum-Flow Minimum-Cut (Max-Flow Min-Cut) Theorem, the nodes in a graph can be partitioned into 2 subsets x and y with minimum-cut between the two subsets. The edges between the two sets are all saturated and backward edges are not counted in. Additionally, the source node s belongs to set x and sink node t belongs to set y. The entry nodes v_{ij}, row nodes r_i and column nodes c_j can be either in set x or set y. The forward edges between entry nodes v_{ij} to either row node r_i or column node c_j should not be in the cut since the capacities of them are all $+\infty$. There are three kinds of edges inside the cut: forward edges from source node s to entry nodes v_{ij} with capacity a_{ij}, forward edges from row nodes r_i to sink node t with capacity 1 and forward edges from column nodes c_j to sink node t with capacity 1.

Depth First Search (DFS) should be performed to find the boundary between subset x and subset y. The nodes found by DFS defines set x and the rest nodes defines set y. The summation of the forward edges capacity between the two sets defines the Min-Cut. All the row nodes r_i in the subset x defines set of rows S and all the column nodes c_j in the subset y defines set of columns T.

Algorithm:

```
build graph G

create nodes:

  source node s and sink node t
  n*n entry nodes v_ij
  n row nodes r_i
  n column nodes c_j

create edges:

  source node s to entry nodes v_ij with capacity a_ij
  entry nodes v_ij to row nodes r_i with capacity positive infinity
  entry nodes v_ij to column nodes c_j with capacity positive infinity
```

```
  row nodes r_i to sink node t with capacity 1
  column nodes c_j to sink node t with capacity 1

perform Ford-Fulkerson Algorithm to find the maximum flow f

  if the forward edges from sources node to each entry nodes in the flow f are all saturate,
    then output the matrix R and matrix C
    {R = a matrix formed by the flow on the edge from each entry nodes to row nodes}
    {C = a matrix formed by the flow on the edge from each entry nodes to column nodes}

  else
    perform DFS on the maximum flow f to find the subsets x and y
    output the set S and set T
    {S = all row nodes r_i (index of row nodes) in the subset x}
    {T = all column nodes c_j (index of column nodes) in the subset y}
```

Proof of Correctness:

Lemma 1. *The result of the reduction is a valid input for Ford-Fulkerson Algorithm.*

Proof. The input to the Ford-Fulkerson Algorithm is a flow network with a source node, a sink node, intermediate nodes and directed edges with capacities. A graph G was built for this problem described in the algorithm which is a valid input for the algorithm. □

Lemma 2. *Any flow on edges between entry nodes v_{ij} and row nodes r_i or column nodes c_j are entries in the decomposition matrixes of A.*

Proof. Since each entry nodes v_{ij} has an edge points out to the row node r_i and an edge points out to the column node c_j with capacities positive infinity, the output of the algorithm (maximum flow f) divide each entry value a_{ij} into two flow values on the two edges. Each flow values on the edges corresponding to the entries in the decomposed matrixes. □

Lemma 3. *Any entry in the decomposition matrixes of A are flow on edges between entry nodes v_{ij} and row nodes r_i or column nodes c_j.*

Proof. Since the edges point from entry nodes v_{ij} sink in the row nodes r_i and column nodes c_j, the entries in the decomposed matrixes is corresponding to the flow on the edges. □

Lemma 4. *The output of Ford-Fulkerson Algorithm can be transformed into a valid output of the problem.*

Proof. There are only two situations for any maximum flow output from the algorithm. Any maximum flow f with saturated edges between source node s and each entry nodes v_{ij} is a row-plus-column (RPC) decomposition of matrix A. On the other hand, the set of rows S are all row nodes r_i in the set of x with source node s inside and the set of columns T are all column nodes c_j in the set of y with sink node t inside. □

Running Time:

There are $n \times n + 2n + 2 = (n + 2) \cdot n + 2$ which is $O(n^2)$ nodes and $n \times n + 2n + 2n = (n + 4) \cdot n$ which is $O(n^2)$ edges in the graph G. The construction of the network takes $O(1)$ for each node and edge. Finding the maximum flow using Ford-Fulkerson Algorithm takes $O(mV)$ where m is the number of edges and V is the maximum flow value. In this problem, the maximum flow value is bounded by n^2 since there are at most $2n^2$ edges from the source node to the entry nodes layer and each edge has capacity range from 0 to 2 and there are at most $2n^2$ edges from the entry nodes layer to the row nodes layer and the column nodes layer and each edge has capacity 1. So, the cut which has $\{s, r_1, \ldots, r_n\}$ on one sie and all other nodes on the opposite side has capacity n^2. Substituting this upper bound for the max-flow value into the Ford-Fulkerson Algorithm running time bound, and recalling that the number of edges is $O(n^2)$, we find that the running time is bounded by $O(n^2 \times n^2)$ which is $O(n^4)$.