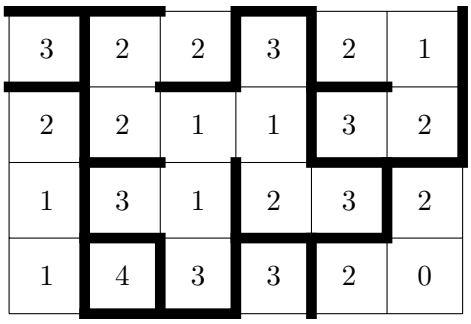
CS 4820, Spring 2018	**Homework 7, Problem 3**
Name: Rongguang Wang	
NetID: rw564	
Collaborators: Siyuao Liu (sl2928); Yihao Chen (yc2288)	

(3) *(10 points)* Consider a puzzle in which you are given an n-by-n square grid, with an integer in the range $\{0, \ldots, 4\}$ written inside each grid cell. You are asked to select a subset F of the edges of the grid, such that for each grid cell the number written inside the cell matches the number of elements of F that belong to the cell's boundary. The following figure shows an example of a puzzle and a valid solution to the puzzle.

3	2	2	3	2	1
2	2	1	1	3	2
1	3	1	2	3	2
1	4	3	3	2	0



Design a polynomial-time algorithm that, when given such a puzzle, either outputs a valid solution or decides (correctly) that there is no valid solution.

Solution:

This problem is aiming to find an algorithm which can select a subset of F of edges which contains the boundary edges that matching with the integer (from 0 to 4) inside each cell of the $n \times n$ grid. A valid solution should be outputted or there is no valid solution for this problem. Basically, this problem can be solved by reducing to the Ford-Fulkerson Algorithm. A floe network G should be built for the algorithm. Firstly, we create a source node s and a sink node t for the flow network G. Then, we should consider the edge type corresponding to each cell in the given grid. There are three kinds of edges in the grid: adjacent edges between cells; boundary edges of the four side boundary cells which are $\{v_{i,j} \mid for\ i = 1\ or\ i = n\ or\ j = 1\ or\ j = n\}$, except the four corner cells $\{v_{1,1}, v_{1,n}, v_{n,1}$ and $v_{n,n}\}$, with one boundary edge; and corner edges of the four corner cells which are $\{v_{1,1}, v_{1,n}, v_{n,1}$ and $v_{n,n}\}$ with two boundary edges. Next, to deal with the adjacent edges case, the grid, which can be treated as a chessboard consists of white cells and black cells, should be divided into two subsets of cells. Since the subset of white cells and the subset of black cells on the chessboard have no adjacent edges with each other which means any cells in either white cells set and black cells set have no neighbor cells, we can just divide the grid into two sets based on whether they belongs to the white cells or the black cells. Now, we can create nodes for both white cells subset and black cells subset. Suppose in the flow network graph G, we place $n^2/2$ black cell nodes in vertical line on the left hand side and place $n^2/2$ white cell nodes in vertical line on the right hand side. We create edges pointing out from the source node s to the black cell nodes. The capacity of each edge is the integer value corresponding to each black cell node. Similarly, create edges pointing out from the white cell nodes to the sink node t. The capacity of each edge is the integer value corresponding to each white cell node. The capacity of each edge is a limitation for each cell which satisfies the requirement of the problem. Although the white cells and black cells were separated into two sets, they still have adjacent edges in reality. If one edge between a white cell and a black cell was selected, it should effect both of the cells. Thus, we connect

each black cell nodes in the left hand side subset to the white cell nodes in the left hand side if they share the same edges. There should be two, three or four edges point out from each black cell node to their adjacent white cell node with capacity 1.

Additionally, we should consider the boundary edges case. Here, we dealing with boundary cells (both black and white cells) without considering the corner cells. Each boundary cells have only one boundary edge which does not share with any other cells. Thus, we connect each black boundary cell nodes $\{v_{i,j} \mid for\ i = odd\ integer\ \&\ j = 1\ or\ n;\ i = 1\ or\ n\ \&\ j = odd\ integer\}$ to the sink node t with capacity 1. Similarly, connect the source node s to each white boundary cell nodes $\{v_{i,j} \mid for\ i = even\ integer\ \&\ j = 1\ or\ n;\ i = 1\ or\ n\ \&\ j = even\ integer\}$ with capacity 1. Finally, we consider the corner edges case. There are four corner cell nodes $\{v_{1,1}, v_{1,n}, v_{n,1}$ and $v_{n,n}\}$ in the grid. Each corner cells have two boundary edges which do not share with any other cells. Thus, we connect the corner cell node(s) who belongs to the black cell nodes set to the sink node t with capacity 2. Similarly, connect the source node s to the corner cell node(s) who belongs to the white cell nodes set with capacity 2. Now, the flow network graph G was built. Run the Ford-Fulkerson Algorithm to find a 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 forward edges pointing from the source node s to the black cell nodes are saturated, which means the flow value equals to the capacity of each edge, and all the forward edges pointing from the white cell nodes to the sink node t are saturated, then we found the subset F. Subset F containing edges with positive flow in the maximum flow f except the two kinds of edges used to check for saturation. Otherwise, there is no valid solution for the problem.

Algorithm:

```
given a grid with integer number in each cell,

build graph G:

create nodes:

  create one source node and a sink node
  partition the cells on the grid into black subset and white subset in O(n)
  create the black cells nodes
  create the white cells nodes

create edges:

  connect edges pointing from each black cell nodes to their adjacent white cell nodes
  with capacity 1

  connect edges pointing from source node to black cell nodes with capacity of the integer
  in the corresponding black cell

  connect edges pointing from white cell nodes to sink node with capacity of the integer
  in the corresponding white cell

  connect edges pointing from each black boundary cell nodes (except the corner nodes)
  to the sink node with capacity 1
```

```
  connect edges pointing from source node to each white boundary cell nodes
  (except the corner nodes) with capacity 1

  connect edges pointing from each black corner cell nodes to the sink node
  with capacity 2

  connect edges pointing from source node to each white corner cell nodes with capacity 2

run Ford-Fulkerson Algorithm and find the maximum flow f

if all edges pointing from the source node to each black cell nodes are saturated
&& all edges pointing from each white cell nodes to the sink node are saturated,

  output edge subset F
  F = {edges with positive flow in the maximum flow f except the two kinds of edges
  used to check for saturation}

else

  output null
```

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. *If all edges pointing from the source node to each black cell nodes are saturated and all edges pointing from each white cell nodes to the sink node are saturated, the maximum flow f is a valid solution for the problem.*

Proof. Basically, there are two kinds of edges in the flow network G: edges of each cell from the grid and flow limitation from the integer in each cell. The grid edges was connected after considering the adjacent edges case, the boundary edges case and the corner edges case. The limitation edges are connected according to the requirement of the problem. Once all the limitation edges are saturated, the requirement of the problem for each cell was satisfied. Then, a valid solution was found. □

Lemma 3. *The output of Ford-Fulkerson Algorithm can be transformed into a valid output of the problem.*

Proof. The output of the algorithm is a maximum flow f and we need a subset of edges F that have been selected by each cell. The set F is a subset of the flow f. Edges with positive flow in the maximum flow f except the two kinds of edges used to check for saturation are elements in set F. □

Running Time:

There are $n^2 + 2$ which is $O(n^2)$ nodes and $(n-1) \times (n-1) + n + 4n - 4$ which is $O(n^2)$ edges in the flow network G. The partition of the black and white nodes costs $O(n)$ time. 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 $n/2 \times n/2$ which is n^2 since there are at most n^2 edges from the black cell nodes layer to the white cell nodes layer and each edge has capacity 1. So, the cut which has $\{s, v_{b1}, \ldots, v_{b\frac{n}{2}}\}$ 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 $O(n^2 n^2)$ which is $O(n^4)$.