Answer the questions below. For the questions involving both programming and analysis, please submit your code through Udacity and your analysis through T-square.

1. Let be a network with integer capacities. The problem is to find an s-t cut in G of minimum capacity that has the smallest number of edges among all minimum capacity cuts of . Show how to modify the capacities of to create a new network also with integer capacities such that any minimum capacity s-t cut in is a minimum capacity s-t cut in with a minimum number of edges.

Implement your algorithm here:

<https://www.udacity.com/course/viewer#!/c-ud557/l-1209378918/m-3410948546>

1. Let and be two vectors of positive integers such that for some ,
   1. Implement an algorithm to find a matrix , if it exists, with 0/1 entries, rows and columns such that it has exactly ones in the ith row and exactly ones in the jth column. Return None if no such exists. Submit your answer here.

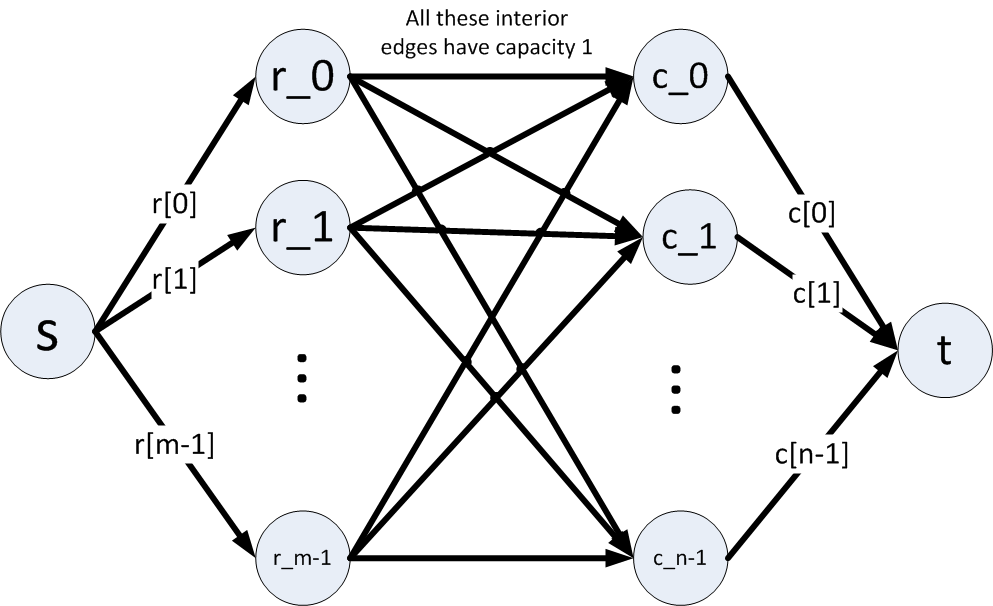
<https://www.udacity.com/course/viewer#!/c-ud557/l-1209378918/m-3373348543>

Hint: Construct an appropriate flow network and find a maximum flow in it.

* 1. Prove the correctness of your algorithm.

The graph being constructed is shown below. From an intuitive standpoint, this graph encompasses everything needed to solve this problem. We have the flow from the source to the row nodes capped at the values of vector r and the flows from the column nodes to the sink are capped by the values of vector c. In the middle we have every row vertex connected to every column vertex with a maximum flow of 1 along each of these edges. In the solution, it is this flow that we are concerned about. In the output matrix, each row can contribute at most one unit of “flow” to each column. The conservation of flow both into the row vertices and out of the column vertices ensure that the matrix output will not exceed the values of the rows and columns.

However, because the maximum flow between the interior vertices can be less than that desired in the output matrix, a check is still required to see that the output matrix is correct.



* 1. Prove that its running time is the Why does it run in the claimed time bound? Hint: What will the value of the augmenting flows be? How many augmenting flows will be required?

Using the graph above, since r and c both sum to K, we have K possible augmenting flows from the source to the row nodes and from the column nodes to the sink node. Note that each of these augmenting flows is only counted once – a flow from s to r\_i to c\_j to t will result in one augmenting flow. There are K of these augmenting flows.

We have a potential for m\*n augmenting flows among the interior nodes. Note that there are m\*n edges in the interior, each with a unit capacity.

So in total, we have a maximum m\*n\*K augmenting flows, so the running time is O(mnK).

Note: Using Edmonds-Karp takes O(|E|2|V|) time. In this case, that would be O( (mn + m + n)2(m+n+2)), but we can reduce that to O( (mn)2(K) ) because

but because this graph has a special structure, the bound on the running time can be made tighter.

1. A graph is said to be k-regular if the degree of each vertex is exactly k. Let be a -regular bipartite graph.
   1. Prove that .
   2. Prove that any such has a perfect matching. Hint: Use Frobenius-Hall.
   3. Prove that the edges of a k-regular bipartite graph can be partitioned into k perfect matchings.
2. Let be an undirected graph with edge costs . Let denote the set of s – t paths for two vertices s, t in . Let denote the set of cuts that separate s and t (each cut is a set of edges). Prove that .