BLG336-E PROJECT REPORT

Project No	3
Student	Alperen KANTARCI – 150140140
Instructor	Zehra Çataltepe
Date	11.05.2018

Important reminder: In order to compile the code you should use this command g++ -std=c++11 main.cpp -o thomassShops

1-) My implementation to disconnecting graphs is based on Menger's theorem with max-flow algorithm. Menger states that finding the maximum number of internally disjoint paths from source to sink will give you minimum number of nodes to disconnect the graph into two parts. This theorem can be proven by max-flow min-cut algorithm and from this intuition I implemented the Menger's theorem to this problem. I used this logic because the thing that we want to the split the graph into two, yet we do not have certain sources and sinks in our problem. So I calculate for the all possible source-sink pair to choose the best among them. Another sub problem that I encountered is in our splitting there should be at least 2 shops in each sides but Menger's theorem doesn't guarantee that property. So I check the each side if there are shops that do not trade with another shop, if there is a node like that then I do not count that partition as successful partition.

In my implementation I change the original graph before running Ford-Fulkerson Algorithm to find max-flow. First since graph is undirected, I change the graph as directed and give them capacity of 1. After that in order to apply Menger's theorem I split all nodes except source and sink into two different nodes. I call these two splitted nodes as n_input and n_output for node n. n_input nodes accepts only coming edges to node n. On the other hand n_output only accepts out edges from node n. And there are 1 connection between n_input and n_output which coming from n_input into the n_output and have capacity of 1. This enables us to converting max-flow algorithm from edge based to vertex based. Reverting the internal edge between n_input and n_output means that path is blocked. This implementation make easier to calculate internally disjoint paths from source to sink. So max-flow algorithm will directly give us maximum internally disjoint paths which means that minimum number of nodes that should be removed.

You can see how I transform the original graph on Figure 1. The transformed graph is on Figure 2. You can see the capacities are 5 and internal arcs of the splitted nodes are 1. Also you can see that source and sink nodes are not splitted and I calculate for every possible pairs.

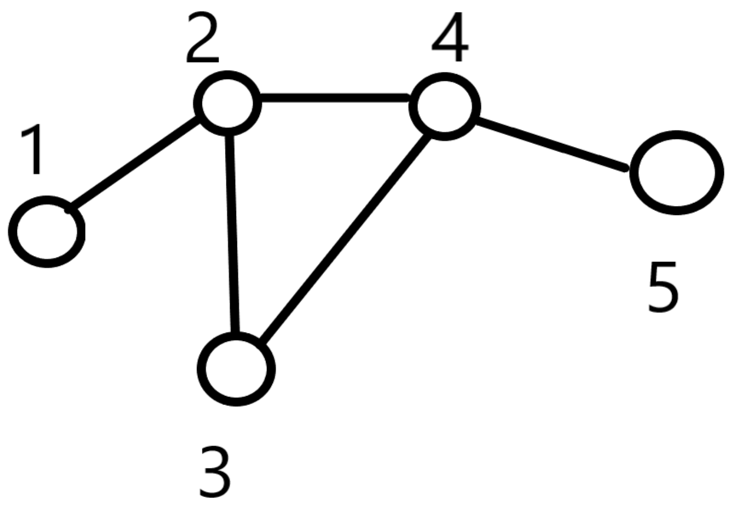

Figure 1: Original graph

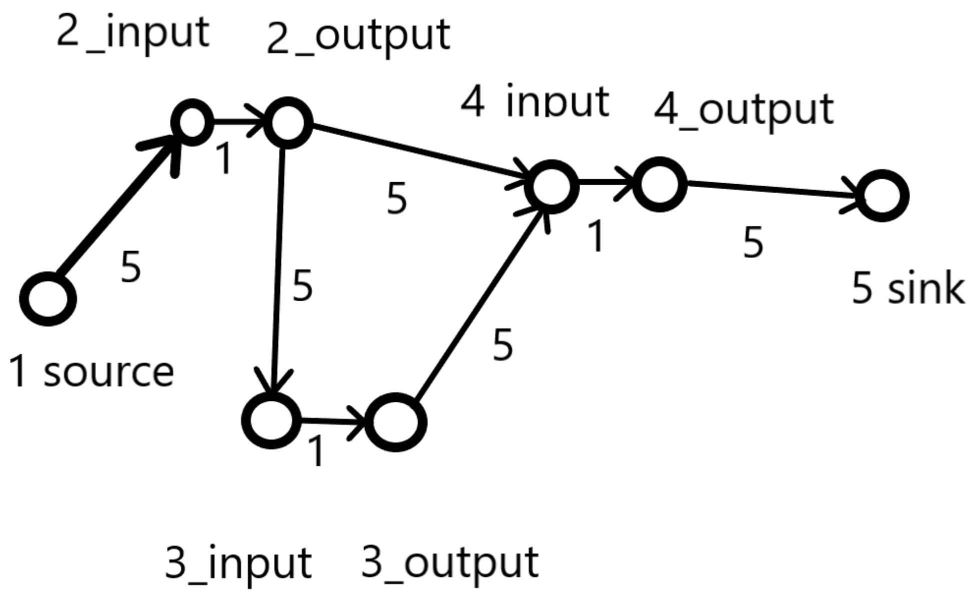

Figure 2: Transformed graph

2-) Residual graph after the Ford-Fulkerson algorithm gives us a graph that has no path between source and sink. Because of at every flow we reverse some edges after a certain point there won’t be any edge to flow from source to sink. So actually back edges will correspond to the cut of the graph. These back edges shows us there is no more augmenting path exists. I actually used them in my minimum cut calculation. In order to assure there should be atleast 2 shops in Thomas’s shop problem, i check the minimum cut set by searching back edges. So by using that i find the vertexes that belongs to minimum cut if i remove some nodes. Since there are internal arcs in my transformed graph. Reversed edge with 1 capacity means that this node will be removed for disconnecting this graph.