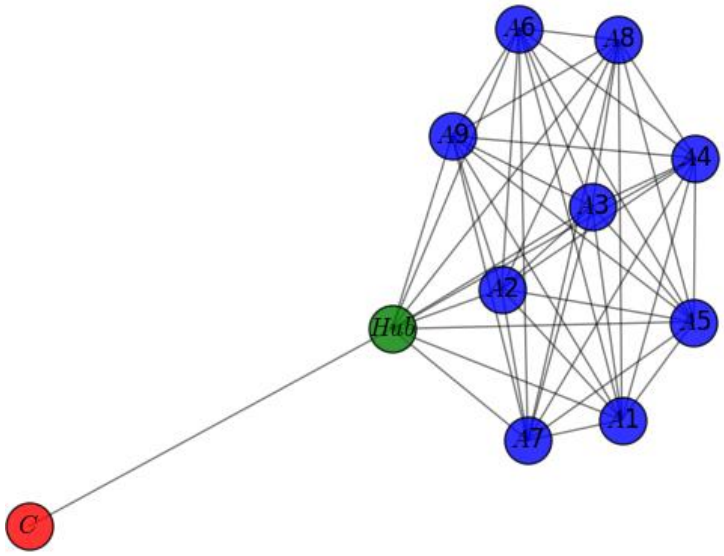
Network Software Modelling Assignment 2

Delivery Agencies Network



Bantra Elpida 16200254

Goyal Shruti 16200726

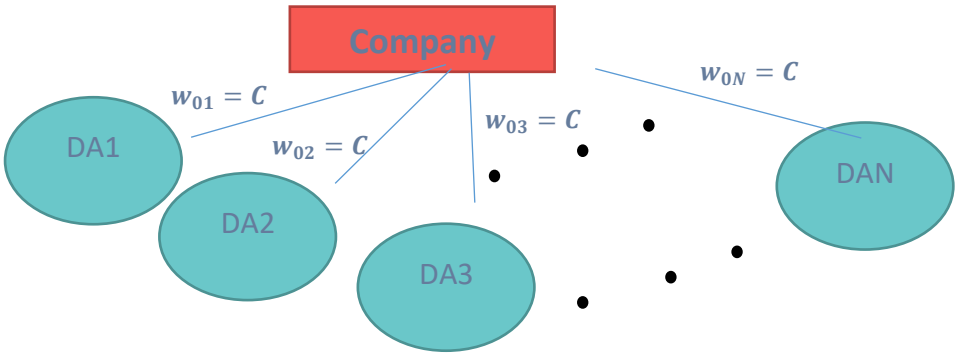
Gupta Deepak 16200660

PROBLEM: Delivery Agencies Selection by Company

We decided to study the delivery agencies problem. This is about how a company after receiving an order from a customer assigns it to one of the delivery agencies which it is collaborating with. Then, the algorithm returns to the company the name of the Delivery Agency and the total cost. We start our analysis based on the following assumptions:

Assumption 1: The company collaborates with N number of delivery agencies and is giving exactly the same amount of money to each of it in order to deliver the order.

We can imagine the assumption one with the following undirected graph, where the order can go only from the company to one of the N Delivery Agencies and the weights of all the edges are the same and denotes the cost C, that the company is going to spend for each order that wants to deliver:



Assumption 2: The company randomly selects one of the N delivery agencies and without concerning for its availability. The company is going to have total cost C for the delivery in any case.

Assumption 3: If the delivery agency is available must undertake the order. The availability of it is denoted by the availability state on the node that symbolises the agency. This state is zero when the agency is not available and is one when it is available.

Assumption 4: Every time, that an ordered is assigned to a delivery agency and it is not available, it is its duty to find a subcontractor (another delivery agency) to give the order.

Assumption 5: Each DA communicates with every other DA. Now, the choice of the subcontractor is based on the weights w_{ij} of the edges between the first, not available for the order, DAi and the subcontractors j (all the other Das). These weights are different, where i=1,…,N a nd j=1,…,N a nd denote the cost of the subcontracts.

Assumption 6: The DA decides the subcontractor according to the weights, apparently choose the node with the smaller weight in order to pay less for the subcontract, but ignores once again if the delivery agency is available or not. These weights are changing in the algorithm, but initially are subjective to some constraints. These constraints are:

1) Initially, the total sum of the weights of every path, that starts from one delivery agency i and passing through all the different delivery agencies once finishes to the delivery agency j, cannot be greater than the cost C. Since, every node is connected to all the nodes once, the above constraint can be secured if the following statement is true:

$$\max_{\forall i} \sum_{j=1}^{N} w_{ij} \leq C \quad (1)$$

2) Every time that a delivery company DAk returns availability state zero, all the edges with direction from every other node to this company's node take weight equal to the addition of its previous weight and of the cost C. So, when availability_state_k = 0, then

$$w_{ik} = w_{ik} + C \quad (2)$$

Like it is declared that we can return back to this company only when all the other companies are unavailable and it has the cheaper subcontract.

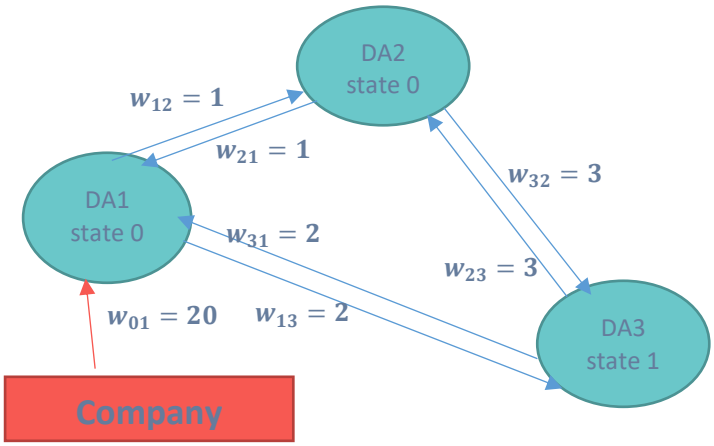
3) The weights are initially different because of different reasons such as distance, alliances between the companies etc.

We can describe this logic as an initially Greedy algorithm, which decides according to the weights and without knowing the availability state which neighbour is cheaper and assign the work to it. But after the visit to one node if this agency is available the algorithm stops, otherwise, the information about its unavailability passes to the weights of the edges that lead to it in order to be sure of two things:

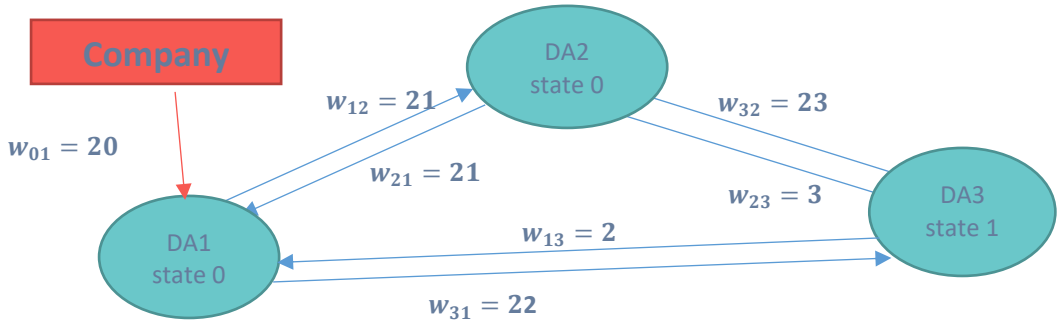
1. The algorithm will not come back to an unavailable agency while there are available agencies.
2. Adding as second stopping criterion (the first one is to stop when the DA is available and to return as total cost C) to stop when the DA is unavailable and we reached with an edge with weight strictly greater than C, we know that all the DAs were unavailable and then the company must pay bigger total cost than C€ (we set a parameter M here where MC will give the total cost in this case).

Below we study different cases for N=3, C=20€, M=1.4, weights that obey the above constraint and different availability states.

Case1: We start with an available DA. If the first chosen DA from the company is the DA3 the algorithm stops and is returned as total cost € 20 and the name of the DA3.



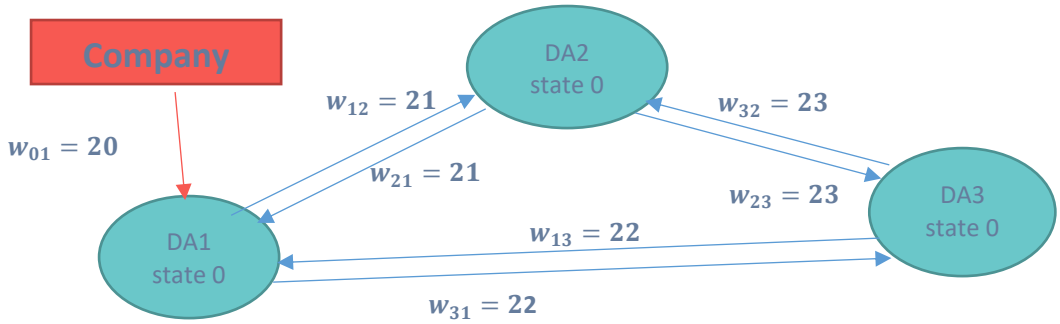
Case2: We start with an unavailable DA but there is at least one available DA. If the first chosen DA from the company is the DA1 the weight from the previous node (company) is equal and not greater than C, so the algorithm doesn't stop, $w_{21} = w_{21} + C = 21$, $w_{31} = w_{31} + C = 22$ and the next node is selected. As $w_{12} \leq w_{13}$, we visit node 2 next. DA2 has state 0 but $w_{12} \leq C$. So, the weights change $w_{12} = w_{12} + C = 21$, $w_{32} = w_{32} + C = 23$ and we have these weights now:



As $w_{23} \leq w_{21}$ we go to the node 3, which has state 1 and the algorithm stops, returning DA3 and total cost 20€.

Case3: There are no available DAs.

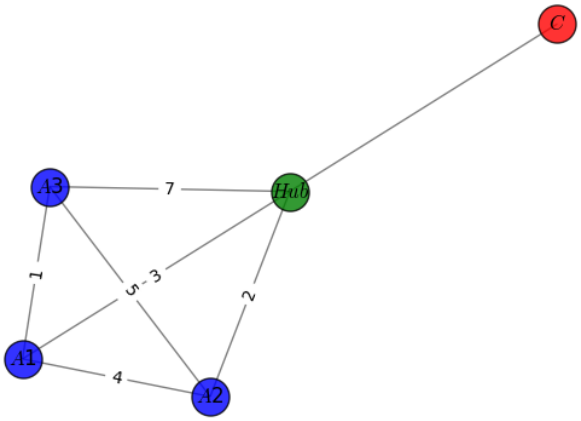
If the first chosen DA from the company is the DA1 the weight from the previous node (company) is equal and not greater than C, so the algorithm doesn't stop, $w_{21} = w_{21} + C = 21$, $w_{31} = w_{31} + C = 22$ and the next node is selected. As $w_{12} \leq w_{13}$, we visit node 2 next. DA2 has state 0 but $w_{12} \leq C$. So, the weights change $w_{12} = w_{12} + C = 21$, $w_{32} = w_{32} + C = 23$ and as $w_{23} \leq w_{21}$ we go to the node 3. DA3 has state 0 and $w_{23} \leq C$ so the algorithm doesn't stop, $w_{13} = w_{13} + C = 22$, $w_{23} = w_{23} + C = 23$ and the next node is selected. Then, $w_{31} \leq w_{32}$ so we go to node 1.



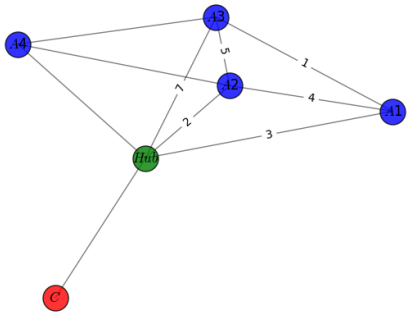
Now, we are in a node with state 0 but $w_{31} = 22 > 20 = C$ and the algorithm stops returning DA1 as the delivery agency that has to deliver and total cost 20x1.4=28€.

Unit Test Cases:

Test 1: Randomly selected node or initial agent which is selected is in the available state, so it will deliver the product directly without traversing.



Test Case 1 and 3



Test Case 2

Test 2: Starting agent selected by default is A3 and only agent A2 is available (state ==1) and all other agents A1, A3, A4 state is unavailable (state == 0). First, we go to selected node A3 then we will find a neighbour of A3 with minimum cost on edges, then we will go to A1 then to A2, which will finally deliver the product.

Test 3: All agents are in state == 0 i.e. No Agent is available. Program will not traverse in graph and print message "No Delivery Agents are Available"

Conclusion:

Simulation of Delivery Agent network has been successfully designed, developed and tested using various test cases. Random complete graph has been generated using networkx library. Communication between agents via edges and based on their state. Further, this problem can be implemented to optimize the delivery network of companies like JustEat.ie, Deliveroo.ie, so that companies can save time and money to provide fast delivery to customers.