Computer Network project's work report

Project number 3

Author: Amir Hamidi

Student Number: 9907794

January 2022

Table of Contents

Introduction

In this project, we have implemented the Distance Vector algorithm. we avoid the repetition of describing this algorithm as it is fully described in text book and project description.

All parts of this project are completely implemented and we now point out the important features of this project. In this project you can choose an input (an adjacency matrix of a graph) through IO and the Emulator will calculate the distance vector of that graph.

After that You will be able to see the distance vector of any node of the graph in each iteration and at the end, you will be able to update the cost of a link in the graph and the Emulator will calculate the distance vector of the graph again. Of course, you can see the distance vectors (for old iteration(before update) and new iteration(after update)) again and this cycle continues until you exit the program.

Code Explanation

Unlike its name, we don't want to get into details of coding in this section as it is not the goal of this work report. but here we notice some of the most important aspect of overall codes and files that has been written and created for this project.

Files (.py)

Node.py: This file contains the Class "Node" in this project. As it was recommended, we have use OOP in this project. Objects of this class (class Node) have three methods and countless attributes. These methods are vector_update() which updates the distance vector of that node, link_update() which updates the link specified by the user and dv_show() which shows the distance vector of all iterations.

DV_Emulator.py: this python file is playing the role of network. As it was mentioned in the project description, nodes are in different places and it is the network that has the responsibility of delivering the necessary information for distance vector algorithm in real world. this file is exactly doing that by delivering the distance vector information to each and every node.

TXT Folder

In this folder, we have several inputs, Input 1 is the sample input in the project description, Input 2 is question 3 in the 10^{th} home work of this course, Input 3 is an example in the internet and finally, Input 4 is the example in the textbook. After completion of this project, the DV_Emulator.py was able to obtain the distance vector of all 4 inputs. It was also able to update distance vectors correctly, after a link update (again, for all 4 input).

Walkthrough for Running the Project

This program has three stages and I explain them in order of occurrence. This helps you to run the project easier.

First Stage:

In the first stage the program will automatically update the distance vector of the initial input after you run the DV_Emulator file. like all other projects, you can give your input to the program through arguments (In the first index of the arguments).

Second Stage:

In the second stage, the program asks you whether you want to see the distance vector of any node or not. At this point, you should choose between **y** and **n** as your answer (it **not** case sensitive). If you choose n as your answer, the program proceeds to the next stage and it doesn't show you the distance vector of any node. Now consider you have chosen y as your answer. (which mean you want to see distance vector of some nodes) The program then asks you to select a node and you should choose the node that you want to see Its distance vector.

Let's say you want to see the distance vector of node 2 and this means you should enter this node by simply type 2 and then press enter. The program then shows you the distance vector of node 2 for all iteration. After showing the distance vector of node 2, the program asks you to enter another node. this process continues until you enter -1 as a node (which doesn't exist) and this number mean that you are now finished and you want to go to the next stage.

Stage three:

In this stage, you'll be asked to enter your request and you should choose between ***LU*** and ***Exit*** as an answer(it **not** case sensitive). If you choose exit, the program stops and you should run it again in order to use it. If you choose LU on the other hand, you will be able to update a link cost. In order to use the LU command, you should first type LU, following by the number of first node, following by the number of second node and then, the new cost of the link (the link which has the first node and second node as its two sides). Here is an example of how you can change the cost of the link between node 1 and 2, to the new value 5.

lu 1 2 5

as it can be seen, there is one space between each two value.

After you have updated the link cost to its new value, the program returns to the first stage in order to calculate the distance vector of each node based on new cost of the updated link.

It's also worth mentioning that mistakes are predicted and if you answer any of the questions in any of the stages with inappropriate answer, the program will response with a note that will guide you to a suitable answer.

NOTE: Besides this work report, there is Presentation, video file, right next to this PDF. In case of any confusion with reading this work report, feel free to check this video file as I have run the entire project by myself.