Graphs

Nariman Haidar

October 2022

1 Introduction:

In this assignment we are going to learn about graphs. A Graph is a non-linear data structure consisting of vertices and edges. There is a lot of types of graphs like a tree and linked list. To understand the graphs very well we are going to work with something realistic in our assignment like the railroad network of Sweden. So in this map we are going to description of all the connections and turn this into a graph and then find the shortest paths between cities in Sweden using trains. So we are going to understand how a graph data structure works and time complexity for this algorithm.

2 Method:

the graph:

The graph that we have in this assignment is about 52 cities and 75 connections and to represent this graph we are going to read the file that has all information and then add bidirectional connections to the graph. To do that we are going to create two classes, city and connection. The class city is to connect all the cities with each other has string name and an array that is connections, then writ the arguments string name and an array of neighbours then write the connect method that has two parameter city next and int destination then it will go through all the neighbours and if the neighbour of i is null then make a new object of class connection with two parameter of next and destination. Then I write the class connection that have two arguments city and destination.

the map:

Now we have the city class defined then we are going to implement the map class. With the help of the object "city" we can define a map. A map consists of a collection number of cities, as we will represent 54 cities and 75 connections. Then I write the look up method that we are going to use it to lookup the object that represents a city given the name of a city.It is going to lookup procedure is only used when we add new cities to the map and when we start to traverse. To

implement the look up method that has a parameter string name we are going to hash the name and while it is true if the index of array cities is null then break and if the name of index of cities is equal to the name so return the index of cities then set the index to the index plus one mod then set the new object of city and then set the cities index to the city and then return the city.

hashing to our resque:

Then I write the method hash that takes a string as argument. I used the hash method that is in the assignment. We use the hash method to be able to place all the cities in a vector and the hash method help us to save memory.

Shortest path from A to B:

After that we implement the map so now we can use this to find route between a given city and another. So first we are going to write a method that measure the time that takes between two cities and then we are going to implement the method that is depth first and this method is using recursively as, so to implement this method that take three parameter first is of type city from and the second is of type city to and the third is of type integer max, then we are going through all the neighbours then If the start node has neighbors, the shortest path is searched recursively. Then if the distance is not null so the value shrt is updated. Then I write some bench mark to know the shortest n shortest way between different cities by measure the time by minutes that took between two cities and then measure the time that took from my program to measure the time between the cities. We can see the result in the table down. We know is not the optimal solution, because when searching for a particular trips, it took a very long time to get the answer, because the cities that the program visited are not saved, and we want a better and faster solution, so we must think of an alternative plan that is better.

table

detect loops:

We need to improve our code to be better in terms of response time. So we are going to change the loop that we wrote in the previous code to a new loop that do not strictly need the max value since we will not end up in an infinite loop anyway and it can remember which cities it has visited and that way, you won't have to pass them again when searching. We can see the results down in the table that is better than the result in the previous table.

table

Now we are going to find the shortest path from "Malmö" to "Kiruna" using a max value of 10.000 and we can see the result in the table down. This table

From-To	The shortest way	The programs time (ms)
Malmö-Göteborg	153	3
Göteborg-Stockholm	211	7
Malmö-Stockholm	273	3
Stockholm-Sundsvall	327	67
Stockholm-Umeå	517	36418
Göteborg-Sundsvall	538	14486
Sundsvall-Umeå	190	90
Umeå-Göteborg	728	1
Göteborg-Umeå	no path	no path

Table 1: This table to measure the time of the shortest way between the cities and measure the programs time.

From-To	The shortest way	The programs time (ms)
Malmö-Göteborg	153	0.022
Göteborg-Stockholm	211	0.033
Malmö-Stockholm	273	0.05
Stockholm-Sundsvall	327	1.3
Stockholm-Umeå	517	1.5
Göteborg-Sundsvall	538	1
Sundsvall-Umeå	190	2.3
Umeå-Göteborg	728	0.07
Göteborg-Umeå	728	1.3

Table 2: This table to measure the time of the shortest way between the cities and measure the programs time.

shows that when we have the max value set to null, we get an answer faster compared to if the max value is high.

table

Things to ponder

Our solution is a good solution when the maps are small, but when it comes to large maps and looking for the shortest path there, our solution is ineffective there. Google Maps is a great and very effective example. For example, if we wanted to search for the shortest route for a train journey from Malmö to Athens, the result was a search time of 18 milliseconds, which is a very good thing compared to our results that we have done.

From-To	**The shortest way**	**The programs time(ms)**
Malmö-Kiruna	1162	32
Malmö-Kiruna	1162	27

Table 3: This table to measure the time of the shortest way between the cities and measure the programs time.