IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 5, Ver. I (Sep. – Oct. 2015), PP 26-30
www.iosrjournals.org

Public conveyance system for shortest path nding for real road network

Agam Mathur[1],Mayuresh Jakhotia[2],Anish Lavalekar[3],Nikita Magar[4]
[1];[2];[3];[4]Computer Department,VIIT,Pune University

Abstract: *This article analyzes problems of determining the shortest path and optimal route amongst the given stoppages. The model of the problem is presented as a directional graph, where nodes are pickup points (termed as stoppage point in database) and crossings outside stoppage points and edges are roads among stoppage points and crossings. Each node has some information attached to it: stoppageId, stoppageName, latitude, longitude and numberOfPassengers of the stop, maintenance organizations, and mark(s) of the crossing(s). All pickup points are connected by roads. These roads are considered as the edges of the graph. Edges also have information attached to it: roadId, source, destination, distance, time etc.We have selected Floyd Warshall algorithm to nd the shortest path between two stoppages. This algorithm works in two stages: in rst stage, it nds the shortest path between all stoppages, and in second stage it nds optimized route to visit some of these stoppages. The solution is displayed in the form of shortest distance and time between two locations. The program is written in java language. It uses 3 tables as input from database : nodes, vehicle details and road. This paper gives implementation outcome of Floyd Warshall algorithm to solve the all pairs shortestpath problem for directed road graph system. We have considered an example of a map of Pune.*
Keyword: *Adjacency matrix,intermediate path,optimal route, shortest path,transitive closure*

I. Introduction

Now days in the current scenario companies and schools provide transportation facilities to their employees and students from their organizations to their respective houses and vice versa. The purpose of this article is to present an interface between a traveling agency and its traveller so that, the subscribers can keep a track of the service they have subscribed.A key problem in public conveyance system is the computation of shortest paths between di erent locations for a given region. Sometimes this computation has to be done in real time. During the literature survey we found how di cult it is to nd the shortest path covering all pickup points and how to allocate available transport vehicles to these speci c routes manually. There is always a need to nd optimal path for their operational viability. So, just the idea that our system will help in e cient allocation of vehicles and shortest route creation to save time and fuel consumption. The system scope is to design such system for a small area of the city.

II. Algorithms To Find Shortest Path

There are various algorithms[9] present in literature to nd shortest path between two points from source location to destination location such as Dijkstra's, Bellman ford, A*, Johnson's and Floyd-Warshall algorithm. After conducting extensive research on the existing shortest path algorithms, it was observed that Floyd - Warshall shortest path algorithm is the most appropriate for calculating short-est paths in real-road networks since it involves calculation of shortest path between all pair vertices. Also, this is the fastest and simplest algorithm.

III. Methods

1.The Floyd Warshall algorithm

The Floyd - Warshall algorithm[2];[4];[5];[7];[9] (also known as Floyd's algorithm, Roy - Warshall algorithm, Roy - Floyd algorithm, or the WFI algorithm).This algorithm is a graph analysis algorithm, which is $^{(0)}=w_{ij}$ used for nding transitive closure of a relation R and also for nding shortest paths in a weighted graph. Weighted graph may be with positive or negative edge weights (but with no negative cycles).

It is a all pair shortest path algorithm ,so single execution of the Floyd Warshall algorithm nds lengths of all the shortest paths between each and every pair of vertices present in the graph ,thus it does not returns details of the path themselves. To nd all pair of vertices in a graph Floyd Warshall algorithm will be used. This algorithm is competitive for dense graphs and uses adjacency matrices as opposed to adjacency lists

Consider an instance for TSP is given by W as, Represent the directed, edge weighted graph in adjacency matrix form.

	$^w 11$	$^w 12$	$^w 13$	
W= matrix of weights =[2]	$^w 21$	$^w 22$	$^w 23$	3
4	$^w 31$	$^w 32$	$^w 33$	5

: w_{ij} is the weight of edge (i, j), or in nity if there is no such edge.

: Return a matrix D, where each entry d_{ij} is d(i,j). Could also return a predecessor matrix, P, where each entry p_{ij} is the predecessor of j on the shortest path from i.Adjacency may consist of two types of values any number for weighted edge value,zero value for self loop of a node and some in nity value which shows there is no path present between two nodes in the graph. Consider intermediate vertices of a path:

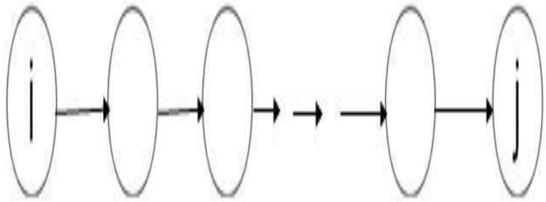

Figure 1: Intermediate vertices of path.

Consider we know the length of the shortest path between nodes i and j whose intermediate vertices are only those with numbers 1, 2, ..., k-1. Now to extend this from k-1 to k.Here we need to nd path which will be the shortest path between vertices i and j. we can use following two possible ways:

Two possibilities:
1. Going through the vertex k does n^0t help the path through vertices 1...k-1 is still the shortest.
2. There is a shorter path consisting of two sub paths, one from i to k and one from k to j. Each sub path passes only through vertices numbered 1 to k-1 Thus, $d_{ij}^{(k)}$=min($d_{ij}^{(k\ 1)}$,$d_{ik}^{(k\ 1)}$+$d_{kj}^{(k\ 1)}$)

Also, d_{ij} (since there are no intermediate vertices.) When k = jVj, we ^{0}re done. Let n be jVj, the number of vertices. To nd all n^2 of shortestPath(i,j,k) (for all i and j) from those of shortestPath(i,j,k 1) requires $2n^2$ operations. Since we begin with shortestPath(i,j,0) = edgeCost(i,j) and compute the sequence of n matrices shortestPath(i,j,1), shortestPath(i,j,2),:::, shortestPath(i,j,n), the total number of operations used is n : $2n^2 = 2n^2$. Therefore, the complexity of the algorithm is (n^3).[5],[6]

Here Floyd Warshall algorithm consider all nodes as the index of matrix and take adjacency matrix as a input considering elements in the matrix as the values of edges of directed graph.Then it

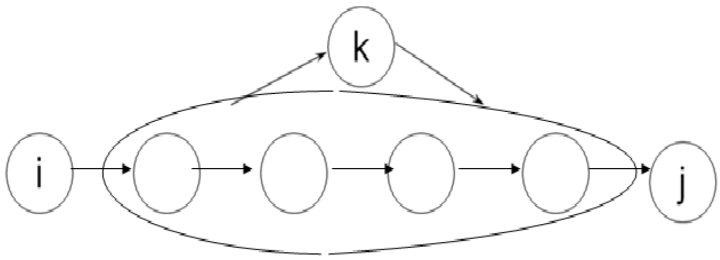

Figure 2: Alternative Intermediate vertices of path using Floyd Warshall algorithm.

will nd transitive closure for a given adjacency matrix which will consist of shortest path betweeen each and every vertices present in the graph.

IV. Implementation And Results

The Floyd Warshall algorithm has been tested on an example map of particular region considering one common source. E.g.Pune city with consideration of DPS school as common source for all destina-tions. We have started the search from DPS school, covering all nodes in the map of Pune city which are present in our database.

STOPPAGEID	STOPPAGENAME	LATITUDE	LONGITUDE	NO_OF_STUDENT
0	DPS	18.465617	73.92336	15
1	NIBM	18.4669588	73.9015676	4
2	Mohmadwadi	18.4705325	73.913577	3
3	Kondhwa	18.4567714	73.8759506	5
4	Fakhari society	18.5122306	73.88601	3

Figure 3: Table of nodes.

ROADID	SOURCE	DESTINATION	DISTANCE	TIME
0	0	1	3.1	7
1	0	2	2.9	8
2	1	0	3.1	7
3	1	4	2	4
4	2	0	2.9	8
5	2	3	6	11
6	3	2	6	11
7	3	4	1	3
8	4	1	2	4
9	4	3	1	3

Figure 4: Table of road.

VEHICLE_ID	REGISTRATION_NO	SEATING_CAPACITY
0	MH12 KJ 2741	4
1	MH12 FN 7093	5
2	MH12 LD 5757	6
3	MH12 SC 8372	7
4	MH12 OC 7232	7

Figure 5: Table of vehicle details.

Our database consists of three tabular entities nodes(gure.3), road(gure.4) and vehicle details(Figure.5). The table nodes consist of the details about the stoppages. It consists of the elds stoppage id, stop-page name, latitude, longitude and no of student. The table road consist of the information about the road which consist of elds road id, source id, destination id, distance and time in which source and destination are referred to stoppage id in nodes table. The third table vehicle details consist of the information about the vehicles. It consist of vehicle id, registration no and seating capacity.

After creating a database we nd the shortest path between all stoppages using Floyd Warshall algorithm. This algorithm needs adjacency matrix as input considering all stoppages as vertices. Then we use this matrix for further requirement of Floyd Warshall. In next step Floyd Warshall create transitive closure matrix which consist of shortest distance between all stoppages as it is an all-pair shortest path algorithm. After nding shortest path between required pair of stoppages, system is able to display the shortest path from required source for di erent vehicles. There may be the situation occurs that the same stoppage consist of more than one routes with same distance from common stoppage point. In such situation, instead of distance we consider the time as a factor to nd the shortest path from the current stoppage point. Algorithm Floyd Warshall only gives the shortest path between two stoppages but we required the shortest intermediate path between any two vertices so we perform bubble sort on the Floyd Warshall output and stored the shortest intermediate path in the queue.

```
Checking disance, time and path before floyd warshall
 ( 0.0 , 0.0 , -1 )    ( 3.1 , 7.0 , 0 )    ( 2.9 , 8.0 , 1 )    ( 999.0 , 999.0 , null )    ( 999.0 , 999.0 , null )
 ( 3.1 , 7.0 , 2 )    ( 0.0 , 0.0 , -1 )    ( 999.0 , 999.0 , null )    ( 999.0 , 999.0 , null )    ( 2.0 , 4.0 , 3 )
 ( 2.9 , 8.0 , 4 )    ( 999.0 , 999.0 , null )    ( 0.0 , 0.0 , -1 )    ( 6.0 , 11.0 , 5 )    ( 999.0 , 999.0 , null )
 ( 999.0 , 999.0 , null )    ( 999.0 , 999.0 , null )    ( 6.0 , 11.0 , 6 )    ( 0.0 , 0.0 , -1 )    ( 1.0 , 3.0 , 7 )
 ( 999.0 , 999.0 , null )    ( 2.0 , 4.0 , 8 )    ( 999.0 , 999.0 , null )    ( 1.0 , 3.0 , 9 )    ( 0.0 , 0.0 , -1 )
Checking disance, time and path after floyd warshall
 ( 0.0 , 0.0 , -1 )    ( 3.1 , 7.0 , 0 )    ( 2.9 , 8.0 , 1 )    ( 6.1 , 14.0 , 0 -> 3 -> 9 )    ( 5.1 , 11.0 , 0 -> 3 )
 ( 3.1 , 7.0 , 2 )    ( 0.0 , 0.0 , -1 )    ( 6.0 , 15.0 , 2 -> 1 )    ( 3.0 , 7.0 , 3 -> 9 )  ( 2.0 , 4.0 , 3 )
 ( 2.9 , 8.0 , 4 )    ( 6.0 , 15.0 , 4 -> 0 )    ( 0.0 , 0.0 , -1 )    ( 6.0 , 11.0 , 5 )    ( 7.0 , 14.0 , 5 -> 7 )
 ( 6.1 , 14.0 , 7 -> 8 -> 2 )    ( 3.0 , 7.0 , 7 -> 8 )  ( 6.0 , 11.0 , 6 )    ( 0.0 , 0.0 , -1 )    ( 1.0 , 3.0 , 7 )
 ( 5.1 , 11.0 , 8 -> 2 )    ( 2.0 , 4.0 , 8 )    ( 7.0 , 14.0 , 9 -> 6 )    ( 1.0 , 3.0 , 9 )    ( 0.0 , 0.0 , -1 )
```

Figure 6: output of Floyd Warshall.

After creating shortest path we allocate vehicles according to the capacity of vehicles and number of students present at each stoppage point. If number of student at particular stop is zero then that place will not consider as stoppage. If number of stop at particular stop is greater than the capacity of bus then system will allocate two buses for same stoppages. Then system will display the total amount of distance and time required to cover shortest path for each vehicle.

VEHICLE_ID	REGISTRATION_NUMBER	VEHICLE_SEATING_CAPACITY	IS_REQUIRED	OCCUPIED_VEHICLE_CAPACITY	VEHICLE_STOPS	VEHICLE_PATH	VEHICLE_DISTANCE	VEHICLE_TIME	OCCUPIED_CALCI
2	MH12 LD 5757	6	1	3	(0->4)||(4->0)	(0->3)||(8-> 2)	(5.1)||(5.1)	(11.0)||(11.0)	(3)
0	MH12 KJ 2741	4	1	3	(0->2)||(2->0)	(1)||(4)	(2.9)||(2.9)	(8.0)||(8.0)	(3)
1	MH12 FN 7093	5	1	4	(0->1)||(1->0)	(0)||(2)	(3.1)||(3.1)	(7.0)||(7.0)	(4)
3	MH12 SC 8372	7	1	5	(0->3)||(3->0)	(0->3->9)||(7->8->2)	(6.1)||(6.1)	(14.0)||(14.0)	(5)
4	MH12 OC 7232	7	0	0	-	-	-	-	-

Figure 7: Vehicle allocation according to shortest path and capacity of bus.

Before starting search for shortest path we allocate students according to vehicle route id and vehicle capacity etc. After testing our system it is found that it is not only capable of nding multiple routes but also shortest path between any pair of stoppages by taking consideration of factors like time and distance which can a ect the system.

V. Conclusion

With the help of Floyd Warshall algorithm,it is possible to conduct transportation analysis con-cerning given geographical region.Sometimes, this type of analysis has to be completed in real time. As a consequence, these analysis tasks demand high performance shortest path algorithms, that run fastest on real road networks.

This system provides user friendly interface.It not only gives the shortest path between the stoppages but also provides the details about the distance,time etc.required to reach from source to desti-nation. Additionally, it provides the information about the vehicle like vehicle route , capacity of vehicle etc. according to the number of passengers it allocates to the vehicle.

Acknowledgement

We take this opportunity to express our deepest sense of gratitude and sincere thanks to those who have helped us in completing this article. We express sincere thanks to our guide Prof. V. A. Mishra who has given us valuable suggestions, excellent guidance, continuous encouragement and taken keen interest in the completion of this work. Her kind help and constant inspiration will even help us in future. We wish to express true sense of gratitude towards Prof. Y.V.Dongre who at very discrete step in study of this paper contributed his valuable guidance and help to solve every problem that arose.

References

[1]. Fast Shortest Path Algorithms for Large Road Networks Faramroze Engineer Department of Engineering Science University of Auckland New Zealand

[2]. Heuristic shortest path algorithms for transportation applications: State of the art L. Fua,*, D. Sunb, L.R. Rilettc aDepartment of Civil Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1 bCollege of Automation, Chongqing University, Chongqing, 400044, China cMid-America Transportation Center, University of Nebraska-Lincoln, W339 Nebraska Hall, P.O. Box 880531, Lincoln, NE 68588-0531, USA

[3]. Mining the Shortest Path within a Travel Time Constraint in Road Network Environments Eric Hsueh-Chan Lu, Chia-Ching Lin, and Vincent S. Tseng Department of Computer Science and Information Engineering National Cheng-Kung University Tainan, Taiwan, R.O.C.

[4]. Design and implememntation of multiparameter Dijkstra0 s(MPD) algorithm:A shortest path algorithm for real-road networks.(September 2011) 1Nishtha keswani,2Dinesh Gopalani 1asistant professor,central universal of Rajsthan, India 2asistant professor, Malviya National Institute Of Technology, Jaipur. (IJAER) 2011, Vol. No. 2, Issue No. III, Septem-ber 2011

[5]. Shortest path algorithms,Wikipedia, the free encyclopedia en.wikipedia.org/wiki/Shortest path problem

[6]. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. tein.MIT Press/McGraw-Hill, ISBN: 0-262-03293-7.I Chapter 25 All-Pairs Shortest Paths

[7]. Algorithm lectures note on shortest path Je Erickson http://www.cs.uiuc.edu/ je e/teaching/algorithms/ I Lecture 20 All-pairs shortest paths

[8]. Open Shortest Path First (OSPF) Conformance and Performance Testing

[9]. Shortest Path Finding Algorithms for Real Road Network. Agam Mathur, Mayuresh Jakhotia, Anish Lavalekar, Nikita Magar 1, 2, 3,4Computer Department, VIIT, Pune University IJLTEMAS Volume III, Issue X, October 2014