Optimized e-Transportation System customized to user need

[1]Ankita Bihani, [2]Pooja Kabra, [3]Supriya Moond

[1]School of Computing Sciences and Engineering, VIT University
[2]School of Computing Sciences and Engineering, VIT University
[3]School of Computing Sciences and Engineering, VIT University
Vellore, India
[1]ankita.bihani@gmail.com
[2]pkabra37@gmail.com
[3]ersupriya13@gmail.com

***Abstract*— This paper presents a web portal that plans the entire trip for the user comfortably in a single web portal. It provides all possible connections between a specified source and destination. The user can prioritize between the cheapest, the fastest and the shortest route available and make online reservations. The system also gives the option of a multi-city round trip. Concepts of graph theory and data structures have been used for optimizing the algorithms. This has reduced the average response time of the system for a user query by 0.2 seconds as compared to the existing system. Besides, the system offers additional functionalities like travel counseling, users' experience sharing forum and hotel booking which makes the system more acceptable to the users.**

Keywords- Web portal, Graph theory, Transportation, Data structure, Algorithms, Multi city round trip

I. INTRODUCTION

In the present scenario, the users need to switch between multiple websites to plan their trip and to decide among the several available means of transportation and/or service providers that suit their needs. This is indeed a time-consuming process without a very satisfactory and accurate result. The main aim of the paper is to address this problem.

The system spans across the various transport facilities available i.e. buses, cabs, trains and flights. The user is given an option to choose between the cheapest, the shortest or the fastest route available between the desired source and destination [1]. By default, the fastest route is selected considering the fact that most users these days opt for fastest travel. However, sometimes users look for cheapest or shortest route possible. Thus, the system caters to the ever-changing user needs and choices, which increases the flexibility of the system.

Besides, the user can avail of reservation facilities based on the specific journey dates. The system gives a comprehensive list of departure and arrival times with the corresponding station names for the entire route. For every connection route – the user can view connection details and/ or make a reservation. The portal has a panel each for the client, admin and service providers [1]. Each panel provides various functionalities that cater to the need of the particular end-user.

- The admin panel has a log in page, provides a detailed user overview, an overview of reserved ticket database, view access to client's personal data. It also enables addition and deletion of client sessions, facilitating ticket reservations and maintaining an updated database of commuting connections.
- The service providers' panel has a log in page, enables them to update the details of their transportation services which include the source and destination, the distance between them, the cost and time taken for each route provided.
- The client panel has various functionalities such as user registration, log in page, access to personal data with edition rights, displaying travel history, making reservations and payments for ticket booking.

II. PROPOSED SYSTEM ARCHITECTURE

The high level architecture of the system that we proposed is shown in the Fig. 2. below:

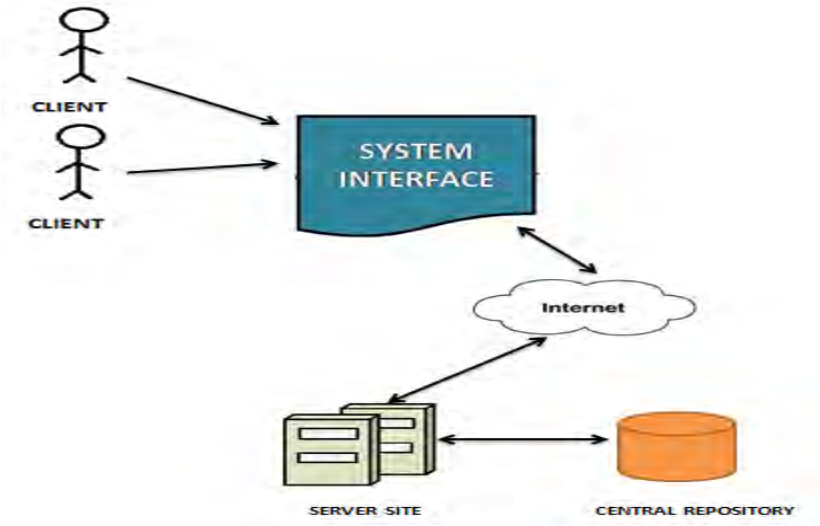

Fig.1. System architecture

In Fig.1. shown above, the System Interface includes functionalities for making ticket reservation for one-way trip or multi-city round trip based on the user's preference, hotel booking facilities, travel counseling facilities, a forum to share the experience with the other users of the system and a feedback collection portal.

III. WORKING OF THE SYSTEM

In order to make a choice, one has to select the source and the destination from a drop down menu. On selection confirmation, a list of all possible routes between the source and destination will be displayed on the screen, starting with the ones that match with the user's choice followed by the lesser relevant options. In case of cheapest connections, connections are sorted by ticket price while in case of fastest connections; they are sorted by connection duration. Similarly, if the user's choice is shortest path [5], they are sorted by connection lengths. To make the user interface more interactive, a graphical calendar is displayed for the user to choose the dates of journey.

The system is designed to improve itself based on feed-back received from end-users using a *self-learning mechanism*. The system provides an interface between the service providers and the client by which the client can register complaints specific to a particular agent/ company.

For making the payments, a payment gateway is used which links the website to the processing network and the merchant account.

Along with this, the major tourist attractions of the destinations are offered. To ease the stay, a list of hotels available at the destination is displayed sorted according to the tariff. This plans the entire trip for the client in a single portal. In addition, there is a forum where users can share their experience about the service providers, their travel experience and the places worth visiting which will benefit the other users to decide and plan their travel.

In addition to the aforesaid functionalities, the system supports travel counseling facility which will provide the client complete information about the climatic conditions, disease outbreaks and necessary medical precautions depending on the chosen destination(s).

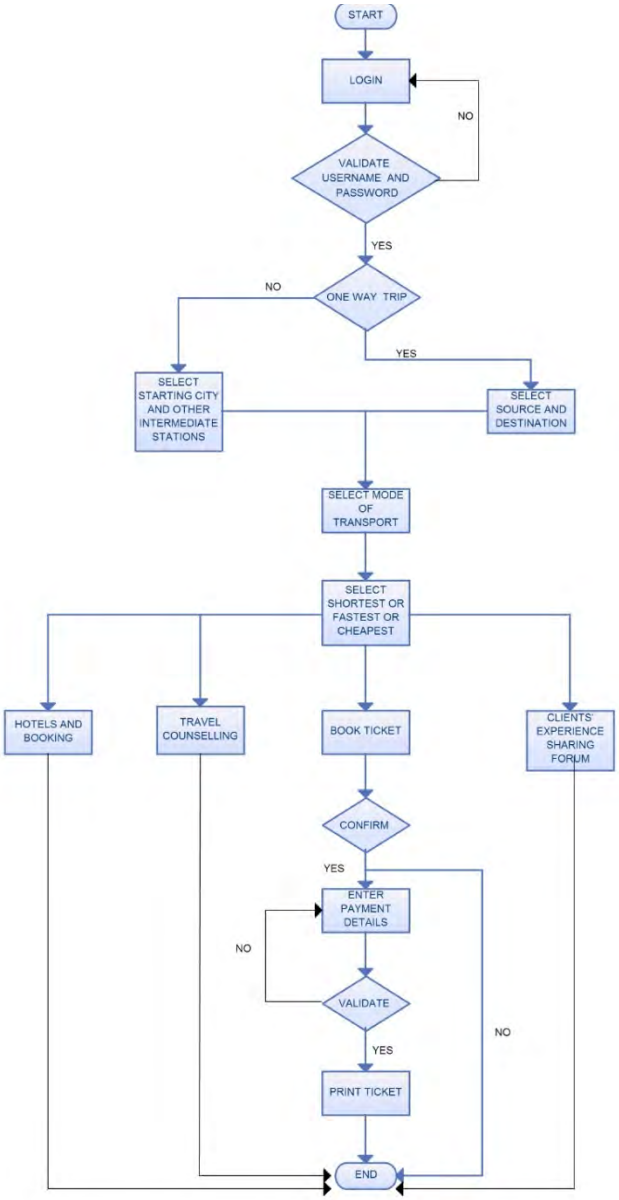

Fig.2. Flow of control of the system

Fig.2. shows the complete flow of control of the system from the point where the user logs into the system.

IV. IMPLEMENTATION PHASE

A. *Algorithms Used*

To implement the idea of presenting the user a wide spectrum of commutation options to choose from, we use several algorithms which are discussed below:

Depth First Search [3] [10] is used for finding routes between the specified source and destination. The algorithm is described as follows:

1. Declare the graph [8] G with vertices V defined by an array.
2. Declare an empty stack.
3. Push all the vertices into the stack and start tracing it. If an unvisited vertex arrives, then pop that vertex from the stack, else display the visited vertex using Boolean array.
4. Push the neighbor vertex of the unvisited vertex into the stack and repeat the process again.

To sort the routes, we have to travel across each and every station in the transportation network. Hence we use the Depth First Search algorithm for this purpose. This ensures that no station is visited twice and keeps the record of every unvisited station using a stack.

Heap sort [3] is used to sort the routes between the stations based on cost, distance or time depending on user's choice. The pseudo code is as follows:

```
Heapsort(X)
{
Build_Max_Heap(X)
for i <- length(X) to 2
 {
swap X[1] <-> X[i]
heapsize[X]<- heapsize[X] -1
Max_Heapify(X, 1)
}
Build_Max_Heap(X)
{
Heapsize[X]<- length(X)
for i <- floor( length[X]/2 ) to 1
do_Max_Heapify(X, i)

}
Heapify(X, i)
{
   l<- left(i)
   r<- right(i)
if((l<=heapsize[X])and (X[l]>X[i]))
then greatest<- l
else
greatest<- i
if(r<=heapsize[X])and
(X[r]>X[greatest])
then greatest<- r
if (greatest != i)
 {
then swap X[i] <-> X[greatest]
Max_Heapify(X, greatest)
 }
}
```

Where left (i) returns 2i and right (i) returns 2i+1.

The heap sort algorithm is used to get three sorted graphs for determining the shortest, the fastest and the cheapest paths.

The three heaps are built according to:

- Time taken to travel from one station to another for finding the fastest path.
- Distance between two stations for finding the shortest path.
- Cost of travel for finding the cheapest path.

The most striking functionality of the portal is to allow the client to plan a round trip that starts from a city, goes to a number of cities and returns to the starting city. This feature helps the users to plan their trips efficiently by providing the shortest or the fastest or the minimal cost route. The Christofides algorithm (used for solving Travelling Salesman Problem [6] [9]) is used to implement this idea. The algorithm is described as under:

1. Find the minimum spanning tree [2] [9] using Kruskal's algorithm.
2. Find the perfect matching M from cities with odd degree.
3. Combine the edges of M and the spanning tree to form graph G.
4. Find the Euler cycle [4] in G by skipping the cities that are already visited.

The Kruskal's algorithm [3] (used in step1 of Christofides algorithm) proceeds as follows:

1. The starting city is considered first.
2. The edge with the minimum cost or distance or time from the starting city to the next city is found.
3. Then the starting city is connected to the next city using the above found edge.
4. Similarly, the least cost edge is found for the next city and so on till all the cities in the user chosen route are visited exactly once.

To ensure that there is always an even number of odd degree vertices to do the matching (in step 2 of Christofides algorithm), Handshaking Lemma comes to play. The Handshaking lemma states that in any graph, *the sum of all the vertex-degrees is equal to twice the number of edges.*

The Euler [4] cycle (to be found in step 4 of Christofides algorithm) is a path through a graph which starts and ends at the same vertex and includes every edge exactly once**.**

Fleury's algorithm is used to find the Euler cycle. The algorithm is explained as under:

Let G be an Eulerian graph [4].

1. Choose any vertex v of G and set current vertex equal to v and current trail equal to the empty sequence of edges.
2. Select any edge e incident with the current vertex but choosing a bridge only if there is no alternative.
3. Add e to the current trail and set the current vertex equal to the vertex at the 'other end' of e. [If e is a loop, the current vertex will not move.]
4. Delete e from the graph. Delete any isolated vertices.
5. Repeat steps 2 – 4 until all edges have been deleted from G. The final current trail is an Eulerian trail [7] in G.

V. RESULTS AND DISCUSSION

The comparative study between the existing methodology and our proposed solution is given as under:

Table I. Comparative study between existing method and proposed solution

Basis of comparison	**Existing Method**	**Proposed solution**
Sorting algorithm	Bubble sort Time complexity: O(n^2)	Heap Sort Time complexity: O(nlogn)
Multicity round trip	No	Christofides algorithm for solving Travelling Salesman problem
Average Response time for a user query	0.10s	0.8s

We propose a single efficient portal that maps the user requirements across all the possible commutation options. Hence, the user is provided with a much better range of options to choose from. It improves the acceptability of the system. Moreover, searching and sorting algorithms have been optimized by using Depth First Search and by using Heap sort instead of Bubble sort for sorting. This will increase the efficiency of the system leading to lesser delay in service response time. The faster the system is – the more it will be preferred by the users.

VI. CONCLUSION AND FUTURE SCOPE

We have provided an optimized system to the users to plan their entire trip on a single web portal. Searching algorithm uses Depth First search which ensures that there is no unvisited vertex in the user chosen route while sorting algorithm used has increased in efficiency from O(n^2) to O(nlogn) where n stands for the number of stations(nodes). The user has an option to plan a multicity round trip. This uses the optimal Christofides algorithm for solving the Travelling Salesman problem. In addition, there are several functionalities that guide the user in the process of planning the travel and stay. Thus Optimized e-Transportation System customized to user need aims to make the user's life easier and faster.

We plan to extend the system by providing an option to the users to go for self- planned road-trips. This would entail giving the user a choice between the shortest and the fastest route available. Moreover, the system can be made available as a mobile app wherein Global Positioning System can be incorporated to guide the users throughout the travel.

The system can be expanded such that the user is not restricted to choose a single means of transport for the entire journey. The flexibility to switch between the different means of transportation for different lengths of the journey based on the optimal criterion chosen by the user will be made available.

VII. ACKNOWLEDGEMENTS

We would like to acknowledge our guide, Prof. Selvakumar R, Senior Professor, VIT University, Vellore (India) for his guidance and support. Our heartfelt gratitude to Prof. Muhammad Rukunuddin Ghalib, Assistant Professor (Senior), VIT University, Vellore (India) and Prof. Siva Rama Krishnan Somayaji, Assistant Professor, VIT University, Vellore (India) for their constructive criticism throughout the research work.

VIII. REFERENCES

[1] Junru Cao, Zhenbai Song, Van Wang. Ting Wang," *Analysis and Application of the Optimal Path Based on ArcGIS Geodatabase' Network Model*" ,2nd Conference on Environmental Science and Information Application Technology, 2010

[2] Anna Kopka, Wojciech Zabierowski, Andrzej Napieralski, "*Graph Theory and Web Technologies Application for Train Timetable Database Handling*", pp.20-24, Polyana, UKRAINE, 2007

[3] John Harrison," *A Short Survey of Automated Reasoning*", Lecture Notes in Computer Science, Volume 4545, pp. 334-349, 2007

[4] Kenneth Sörensen, Gerrit K. Janssens,"*An algorithm to generate all spanning trees of a graph in order of increasing cost*", Pesqui. Oper, vol.25 no.2, 2005

[5] Thomas H. Cormen, "*Introduction to algorithms*." MIT Press, Cambridge, MA, USA. ISBN 0-262-03293-7, 2001

[6] J. E. Yukich, "*Probability Theory of Classical Euclidean Optimization Problems*," vol.1675 of Lecture Notes in Mathematics, Springer-Verlag, 1998

[7] Boris V. Cherkassky,Andrew V. Goldberg,Tomasz radzik ,"*Shortest Paths Algorithms: Theory and Experimental Evaluation*",pp. 129-174, 1996

[8] David S. Johnson, Lyle A. McGeoch," *The Traveling Salesman Problem: A Case Study in Local Optimization*", pp- 1-94.,1995

[9] Robin J. Wilson," *An Eulerian trail through Königsberg"* Journal of Graph Theory**,** vol. 10, issue. 3, pp. 265–275, 1986

[10] Biggs, N., Lloyd, E., Wilson, R. "*Graph Theory*", pp. 1736–1936, Clarendon Press, Oxford,1976

[11] J. B. Kruskal, "*On the shortest spanning subtree of a graph and the traveling salesman problem*" Proceedings of the American Mathematical Society, vol. 7, no. 1, pp. 48–50,1956

[12] Christopher Beck and Laurent Perron, "*Discrepancy-Bounded Depth First Search*", ILOG SA 9, rue de Verdun, BP85, 94253 GentillyCedex, France.

[13] Claude Le Pape, Laurent Perron, Jean-Charles Regin, and Paul Shaw," *Robust and Parallel Solving of a Network Design Problem*",2002