J. Appl. Environ. Biol. Sci., 5(5)73-89, 2015
© 2015, TextRoad Publication

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

A New Method for Optimization of Dynamic Ride Sharing System

Samia Arshad[1], Hamid Turab Mirza[2], Ibrar Hussain[3]

[1, 2]Department of Computer Science, COMSATS Institute of Information Technology, Lahore, Pakistan
[3]Department of Computer Science, COMSATS Institute of Information Technology, Attock, Pakistan

Received: January 1, 2015
Accepted: March 7, 2015

ABSTRACT

Dynamic ridesharing is a profitable way to reduce traffic and carbon emissions by providing an opportunity for a flexible and affordable service that utilizes vehicle seating space. Matching of ride seeker requests with the rides, distributed over the roads is a tedious work. While fulfilling the request of all passengers, the total travel distance of the trip may get increased. Therefore, this article proposes optimal dynamic ridesharing system which matches rides and requests in real time by satisfying multiple participant constraints (e.g. time bounds, availability of empty seat, maximum allowed deviation distance and minimized route ride) to minimize the total travel distance. To efficiently match ride givers and riders we are proposing a novel dynamic ride matching algorithm MRB (Minimal route bi-searching algorithm) considering all above mentioned constraints. We demonstrate working of our algorithm by developing a prototype and evaluated our system on GPS (Global positioning system) trajectories of Lahore city dataset**.** Evaluated results are compared with existing algorithms which shows that our system significantly reduces the travel distance and computation cost in comparison with other recent ride searching methods to maximize efficiency.

KEYWORDS: Dynamic ride sharing system, optimal ride matching algorithm, spatial grid indexing, road networks.

1. INTRODUCTION

Despite the advancement of air and railway services around the world, still roads are the primary source of major traffic and transportation mean. Private cars have become the predominant transport mode globally that steadily worsened traffic congestion and fuel emissions [1]. The burning of fuels adds about 6.3 Giga tons [2] of carbon to the atmosphere each year and twenty-three per cent of world energy-related CO_2 emissions originate from the transport sector.

Usage of public vehicles (e.g. buses and vans) eradicates these environmental issues, however, they boost social issues (i.e. privacy threat, inflexibility in usage (e.g. with whom a traveler wants to share a ride) and unavailability of rides at passenger desired places. To earn high revenue public transport owners usually fill up their vehicles more than their space which is an obstacle, providing comfort and ease to travelers. Researchers spanning over various disciplines such as transportation [3], behavioral, social, environmental psychology [4], economics [5] have identified ridesharing as a good solution to the inefficiency of current transportation models. Therefore, ridesharing is a promising approach to mitigate all these issues.

Ridesharing is the sharing of ride among people with similar itineraries and time schedules by utilizing spare seats in the vehicle [6]. Ridesharing can be either static or dynamic [7], **Static ridesharing** arranges trips that are known in advance, usually one to two days before the departure time while **dynamic ridesharing** uses an automated process to arrange trip on an adhoc basis(e.g. incorporating requests received minutes before the departure time) by using GPS (Global positioning system) on mobile phones [8].

Currently, many mobile phone ridesharing applications are providing their services, like Carticipate (www.carticipate.wordpress.co), EnergeticX/Zebigo (www.zebigo.com), Avego (www.carmacarpool.com) and Piggyback (www.piggyback.com). In such application's, user sets his pickup and destination location by giving a pickup and drop off time. These systems adequately match passengers and rides in real time by considering some constraints (i.e. time bounds, spare seat availability, travel cost reduction and safety etc.) for an optimal solution.

Let's take an example to describe ride matching by considering different constraints.

Corresponding Author: Samia Arshad, Department of Computer Science, COMSATS Institute of Information Technology, Lahore, Pakistan. samiaarshadali@hotmail.com

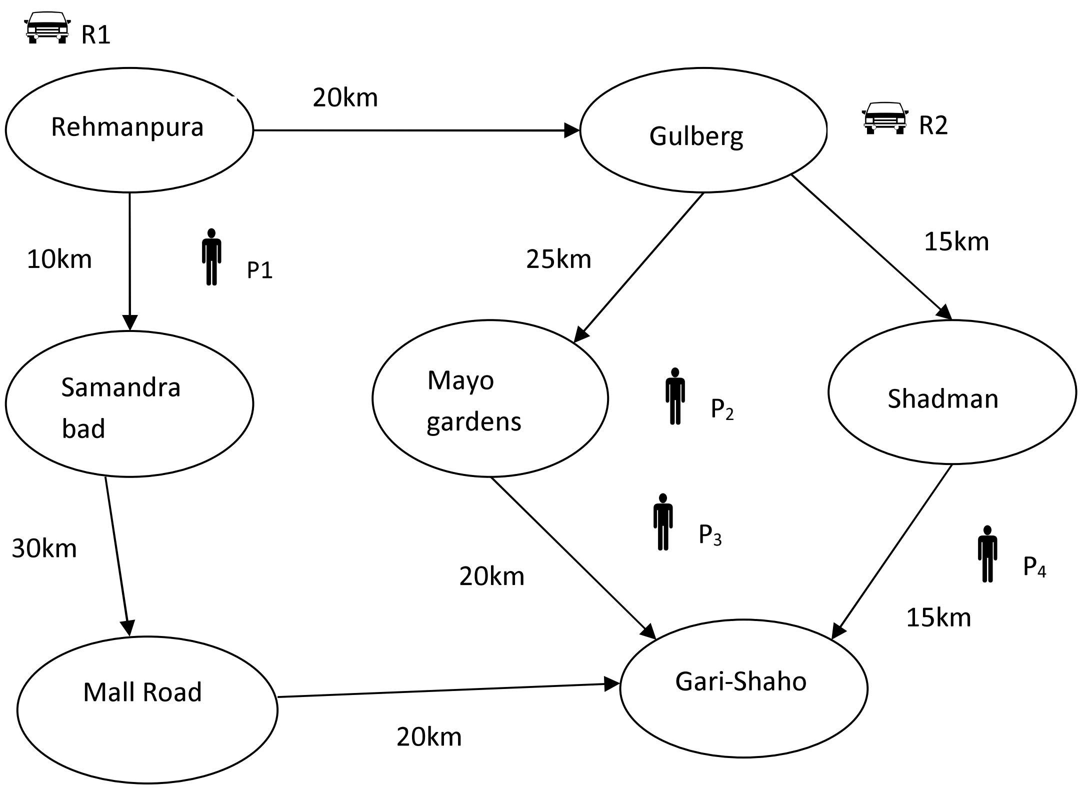

FIGURE 1.Scenario of Ride Matching in Ride Sharing System

Figure 1 exhibits certain scenario: passengers and rides are dispersed on a road network. There are two rides, one is at location Rehmanpura and second is at Gulberg. There are four passengers standing in different locations, i.e., one passenger is at some later location from Rehmanpura, second is at Mayo Gardens, third is at some farther location from Mayo gardens, and fourth is at some farther location from Shadman. All passengers have set their pickup points where they are currently standing and their destination is Gari-Shaho. These passengers are requesting for rides at the same time. Now system has to find out the optimal ride matched with the user's requests under participant's constraints. If the constraint is to maximize passengers, then the ride R2 will be assigned to passengers P2 and P3, if constraint is to minimize the travel distance, then ride R2 will be assigned to the passenger P4, in case constraint is to share a ride with only one passenger then the ride R2 will be assigned to passenger P4, if constraint is to fulfill all passenger requests by allowing 15km deviation then R2 will be assigned to passenger P2, P3 and P4. In this fashion ride matching in a real world scenario is difficult to tackle. The more restrictions a ride share places while searching a ride, the more difficult it will be to find successful matches.

Unlike others [6, 9-11], the goal of this study is to develop a ride searching system considering time, distance, spare seat availability, deviation and minimized route constraints to optimally match rides with rider requests while meeting the service quality guarantee.

The rest of the paper is organized as follows: In section II we sum up the related work and describe our research contributions. Section III formally describes the dynamic ridesharing problem with ride giver and rider constraints and section IV illustrates our proposed service. In section V an optimization scheme is developed for the real-time ride searching system. Section VI presents a simulation environment. Section VII represents the evaluation of our results to investigate how such a ride matching system can minimize total travel distance, considering multiple routes between source and destination by utilizing ride resources more efficiently, compared with conventional ride matching systems. Section VIII concludes and discusses future research directions.

2. RELATED WORK

Development of an optimal ride matching algorithm is a key to ridesharing system optimization. Most state of the art ridesharing systems used different searching algorithms. This is a wide research area which needs to be explored more. Earlier researcher [9] used rolling horizon approach, where plans are made using all known information retrieving ride matches at each execution run, but only a subset of these matches was provided as output before the execution of next optimization request. Their phenomenon for travel distance minimization was that, cost is directly proportional to the travel distance, therefore ridesharing not only reduces cost but also travel distance. They did not use any shortest path technique or other method to reduce distance, did not bother about time bounds also. Moreover, rolling horizon strategy increases computational burden as it provides a lot of matches during the planning horizon process. In contrast to their work, this work uses shortest path technique and spatial factor to reduce travel time and distance.

Afterwards [10] used single side and dual side searching algorithm to obtain best rides matched with the passenger request with minimum increase in travel distance. Single side algorithm finds rides that are spatially and temporally nearest to pickup and destination location. Although, Dual side searching algorithm finds the common ride retrieved from spatial and temporal closeness at both source and destination side. However, dual side searching algorithm may not always minimize the total increase in travel distance, because by utilizing spatial factor they are reducing travel distance. However, this work not only considers spatial factor, but also deviation and minimized route ride constraint to decrease the total travel distance of the participants. Their focus was on minimizing increases in travel distance to fulfill passenger request, however, aim of this work is to decrease the total travel distance of passenger and ride giver including shortening the increase in travel distance.

Practically, from a source to a destination, there exist multiple routes to travel. Finding an optimal minimized route considering different constraints is vital in ridesharing. Limited research has been performed on finding an optimal route in this domain. Researcher [11] used matching route algorithm, which finds the longest common route from passenger and driver routes even when they start and end elsewhere (i.e. passenger and driver routes origin and destination are different) and suggested rides on commonality. However, our work considers the routes of the matched ride (i.e obtained as a result of ride matching algorithm execution) and referred minimized distance route ride to the passenger.

Apart from travel distance, moving unfilled seats of a vehicle raises environmental and economic issues. Earlier recommender systems [12-14] dispatched vacant taxi near to passengers request location on receiving requests without considering ridesharing. Yet [10] and [15] considered rides occupied under full capacity. Although researchers did not consider the spare seat availability in a ride, while execution of searching algorithm. Systems usually checked the availability of empty seats after a ride matching algorithm provides ride matches. Considering time and computations saving, this work is checking vacant seat availability in the ride during searching algorithm execution.

Some researchers [16, 17] proposed systems based on historical trajectories(e.g. recorded log files of rides), offering parking areas for taxi drivers through which they can easily find multiple passengers . Similarly, [18] proposed set of pickup points for a driver to pick up passengers easily and fill up empty seats .While these systems are only designed from the perspective of drivers, however, our system accommodates both driver and passenger needs by providing ridesharing service on passenger desired location for pickup and drop off, and allow deviation to a limited distance, set by the ride giver(i.e. from 0 to 4 km).

Fast retrieval of rides and request processing is very crucial in ridesharing. System usually takes long time to match rider requests with a ride that makes dynamic ridesharing system less feasible to use. For fast processing of passenger request, indexing scheme is appropriate. Earlier work [9, 19] did not use any indexing scheme for query processing. However, in our work, spatial grid indexing method is used for faster request processing.

This research is highly motivated by the work proposed by [10]. On receiving request their system (i.e. T_SHARE) finds a ride match nearer to the request origin and destination location based on spatial closeness (i.e. closest distance) and checks whether it can fulfill the request within the predefined time bounds based on temporal closeness (i.e. minimum travel time). After finding rides at origin and destination side, they found the common ride on both sides and provide its searching results. We extend this concept by considering seat availability check during algorithm execution, obtaining routes of each matched ride and measure deviation distance from the ride original route to passenger source and destination location by using an efficient shortest path technique. Subsequently, system provides the ride that travels on a minimized route from passenger source to destination among all matched rides. T_share [10] system provide matches on basis of shortest pickup and drop off distance, however our proposed system provides ride matches based on total minimized distance from passenger source to destination and also shortens the increase in travel distance of a ride.

Research contributions of this paper are summarized as follows:

- Proposing an approach to consider vacant seat availability during algorithm execution. In case, empty seat is not available during ride matching process it will expand the searching area to find another ride having free seating capacity. Contrary to other systems proposed by [10, 11] that relied on checking ride capacity after ride matching algorithm execution. Checking occupancy of rides during algorithm execution saves time and computations.
- Minimizing the travel distance by considering multiple matched ride routes from request source to destination and recommending a minimized distance route ride among all of them.
- Applying detour distance constraint in contrast to [20] and [10] work where ride fulfilled the passenger request at any detour distance and only take account of time bounds. This work measures the deviation distance using precomputed shortest path technique and provide ride within limited deviation; set by the ride giver (i.e. between 0 to 4 km), to minimize the total increase in travel distance.
- Considering all above factors we are proposing a new algorithm MRB for fast processing of passenger query and finding an optimal ride match on passenger request.

3. PRELIMINARIES AND PROBLEM DEFINITION

Major notations used throughout in this paper are listed below:

Table 1: Major Notations

Notations	Definitions
r	Request for a ride.
r_s	Stream of requests.
r.t	Request generating time.
r.o	Request pickup (source) point.
r.d	Request destination (delivery) point.
r.o.e	Earliest pickup time.
r.o.l	Latest pickup time.
r.d.e	Earliest drop off time.
r.d.l	Latest drop off time.
V	Set of rides (i.e. vehicles) available.
v.id	Ride identifier.
v.l	Ride current location.
v.p	Number of On board passengers in the ride.
G	A grid cell.
g_c	Geographical centre of the cell.
l_g^t	Temporally ordered list of grid cell g.
l_g^d	Spatially ordered list of grid cell g.
l_g^c	Ride list of grid cell g.
V.s	Ride schedule.

Before we formally define the problem, we provide definitions of some basic concepts and terms.

Definition1:- *(Passenger request)* A passenger requests r for a ride associated with r.t (request generating time), r.o (origin or pickup location) and r.d (destination or delivery location), r.o.e (earliest pickup time i.e. when the passenger wants to be picked up), r.o.l

(Latest pickup time i.e. maximum waiting time of the passenger to be picked up), r.d.e (the earliest drop off time i.e. when the passenger wants to be dropped off earliest) and r.d.l (the latest drop off time i.e. maximum travel time of the passenger) as explained in Table 1.

Ridesharing system usually retrieves the origin location r.o and request generating time r.t automatically through the passenger's mobile phone. In this work, we assume that earliest pickup (r.o.e) and earliest drop off (r.d.e) time is equivalent to the request generating time r.t and the latest pickup time is obtained by adding a fixed value e.g. 10 minutes to earliest pickup time. A passenger only needs to explicitly define the drop off location r.d and latest drop off time r.d.l.

Definition 2 :- *(Ride)* From a Set of rides V= $\{v_i\}$ where i=1, 2….n, a ride v_i€ V is comprised of its id v_i.id, location v_i.l, time stamp v_i.t and number of boarded passenger v_i.p.

Definition 3 :- *(Route)* A Route w of a ride is a sequence of road segment w: w_1, w_2……w_n, such that $w_{m+1}.s=w_{m+1}.e$, ($1<=m<n$) where s and e represents starting and ending point of a road segment. Multiple routes w_1, w_2….w_n exist from source r.o to sdestination r.d.

Definition 4:- *(Detour)* Detour distance (dd) is the deviated distance from the original route of the ride. To fulfill passenger request, rides often deviated from the original route. A constraint on maximum deviation is applied by the ride giver (i.e. from 0 to 4km) from the original route of the ride.

Definition 5:- (*Optimal match)* Given a passenger request r and a set of rides V, we say that a ride v_i matches with a passenger request if 1) v_i.p is less than the seat capacity of ride 2) v_i can pick up the passenger before r.o.l and dropped off before r.d.l. 3) v_i can pick up the passenger only if it's at w_m location or within the dd distance 4) v_i has minimum distance from r.0 to r.d.

Problem definition

This work focuses the dynamic ridesharing problem in the following context: aiming to fulfill a stream of passenger requests r_s arranged in an ascending order of their birth time in the queue by dispatching the rides which satisfies the requests r with minimum additional travel distance on road networks and reducing passenger total travel distance by providing a minimized route ride.

Mainly this work focuses on reducing the total travel distance by considering multiple routes from source to destination and recommending the optimal minimized route to travel. Minimizing distance in ride matching problem is similar to TSP (traveling salesman problem)[21]. This work also considers the time bounds (i.e. earliest and latest pickup and drop-off time). The total travel distance minimization problem with time windows is NP-complete because it has been proved that it is a generalization of the Travelling Salesman Problem with time windows (TSPTW), which has already been proved to be NP-complete [22].Therefore only an optimal solution is possible.

4. FRAMEWORK

The framework of the system is illustrated in Figure 2.

A passenger login to the system and requests for a ride according to definition 1.A ride gets registered in the system when joining the ridesharing system. Ride dynamically uploads its status p. 1) when the passengers entered or leave the ride. 2) When the location and timestamp changes (e.g. after every 1 minute) while connected to the system. Ride giver can also offer a ride for multiple passengers.

System receives a set of requests r_s in a queue and serves them by considering FIFO (first come first out) technique. System maintains the spatial indexes of grid cells for fast processing of passenger request. For each request r system invokes ride searching module to find set of rides that are most likely to satisfy passenger request according to definition 5.If passenger request r is satisfied then system give response to the passenger r_p with the ride id. In case of dissatisfaction, system will provide a spatially and temporally nearest available ride which may pickup or drop off passenger after his r.o.l or r.d.l.

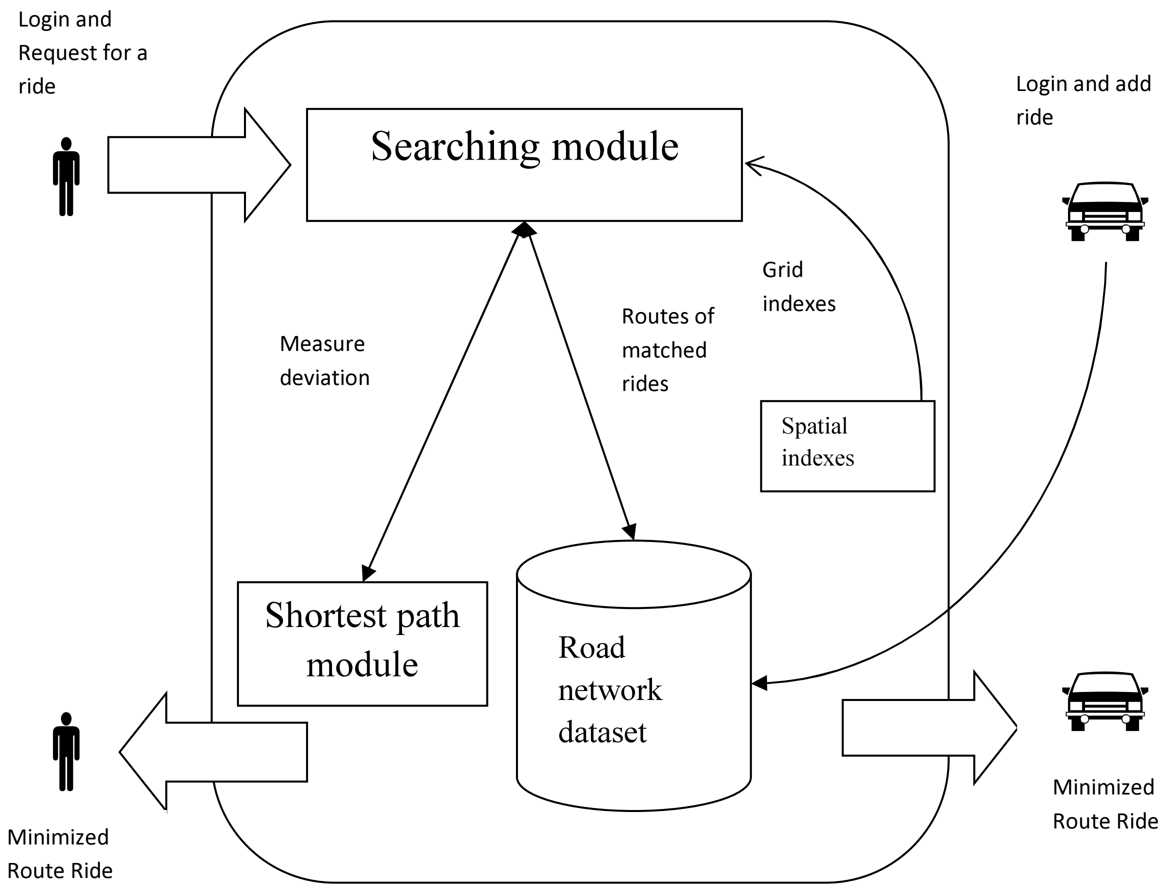

FIGURE 2. System Architecture

Spatial Grid Indexing

In ridesharing main challenge is how to handle a large number of requests as they flow real time in the system [23]. Searching module aims to quickly provide an optimal ride match with the new passenger request with minimum increase in travel distance and follow a minimized route. For this purpose, system will search the rides nearest to the request location. A straightforward strategy is to find distance between the request pickup point and all rides available. After finding minimum distance, selected ride will be assigned to the passenger. While this approach is not practical, because measuring distances between request r and all rides are extremely time consuming. Hence pre computed distances are recommended to approximate the distance of the shortest path. This idea motivated us to use spatial indexing method.

The road network is divided into 5*5 km grid cells where for each grid cell, this work finds out the geographical center of the cell g_c. For each node pair g_{ci} and g_{cj} we used the precomputed distance d_{ij} and precomputed travel time $t_{ij,}$ saved in a matrix. Despite of using advanced travel time estimation techniques, this work only utilizes speed limit on the road network for calculating travel time to lessen the computational load on the system. Now imagine that all points in one grid cell fall to its geographical centre, and then the distance between any two points is equivalent to the distance between two geographical centres of the grid cells. The matrix contained an approximated distance and travel time. Hence, an expensive calculation is avoided by utilizing approximation strategy.

Each grid cell has some data structure for searching rides. System maintains three lists for each cell as demonstrated in Figure-3.

1) **Temporal** ordered grid list l_g^t , containing the list of neighbor grid cells arranged on the basis of shortest travel time t_{ij} to the cell g_i.

2) **Spatial** ordered grid list l_g^d of g_i, containing the list of neighbor grid cells arranged on the basis of shortest travel distance d_{ij} to the cell g_i.
3) **Ride** list l_g^c of the grid cell g_i containing the list of rides which will enter in the cell in near future along with the timestamp t_f when they will enter. Ride list will be dynamic as compared to temporal and spatial lists which are static .Their value changes as the ride moved on from the grid cell g_i or a new ride added in the system.

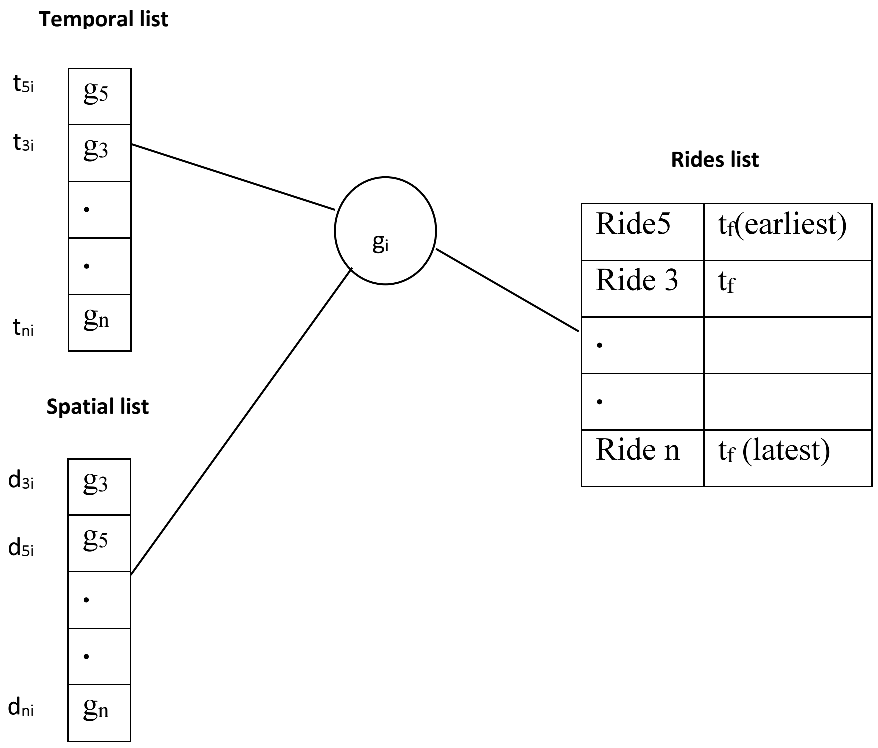

FIGURE 3. Grid Cell Data Structures

Constraints on Matches

This work incorporating many participant constraints i.e. maximum deviation distance, minimized route ride and seat availability constraint to achieve an optimal ride match.

Deviation

The amount by which a ride moved away from their original route is called deviation. Each vehicle has a predefined route, but is allowed to deviate from this route to pick-up and drop-off passengers at preferred locations within a certain service area. *Deviation* services accommodate spontaneous unscheduled requests. However, by fulfilling all passenger requests at any distance, total travel distance of the ride gets increased. To accommodate this tradeoff, this work is providing flexibility to the ride giver to set deviation constraint within 4km and decide whether to deviate or not from the original route by reducing increase in travel distance. In this way without increasing much distance most of the requests will be fulfilled. However, some requests remain unfulfilled by utilizing this constraint.

Minimized Route

From a source to destination, there exist many routes in reality, finding the minimized route to travel will give us an optimal solution. Our work finds rides moving across a given source to the destination within time bounds. Searching results provide a list of matched rides. This work will find the distance of all resulting rides from passenger source to destination and provide minimized route ride to the passenger for saving the total travel distance of the passenger.

Vacant Seat Availability Check

Availability of seat is an important factor which needs to be checked in ride sharing system. Existing systems checked the seat availability after executing the matching algorithm. Resultant rides list was usually large, which wasted a lot of time by increasing computation cost. Our work focuses on checking vacant seat availability in a ride while executing ride matching algorithm. Whenever an algorithm finds a ride fulfilling time bound constraint, then it will check seat availability, if ride doesn't have vacant seat, system will expand the searching area to find another ride fulfilling empty seat available constraint. This method reduces execution time as compared to existing systems.

5. The Proposed Searching Algorithm :- (Minimal Route Bi-Searching Algorithm MRB)

To provide an optimal solution to the dynamic ride matching problem, we are proposing a new searching algorithm including all above mentioned constraints, named minimal route bi-searching algorithm (MRB).

Algorithm: Minimal Route Bi-Searching Algorithm (MRB)
Input: Passenger request r with request generating time t_{cur}, destination location r.d, latest delivery time r.d.l.
Output: Best matched ride.

```
1       $g_o$<-Passenger pickup location
2       $g_d$<-Passenger delivery location
3       $S_0$<-Set of rides entered in $g_o$ before r.o.l
4   For each ride in $S_o$ check spare seat availability
5     If rides have vacant seat                do
          $S_0$ <- rides having empty seats.
      else
          $S_0$ <- $\varphi$
6       $S_d$<-Set of rides entered in $g_d$ before r.d.l
        Repeat step 5 for $S_d$.
7       S <- $S_0 \cap S_d$
8         $M_0$ <- $\varphi$
9         $M_d$ <- $\varphi$
10        If(S contains rides)
          Break and jump to step 19
          else
11       For grid cell $g_i$ in $g_O.l_g^t$     do
12         If   $t_{cur}+ t_{io} \leq$ r.o.l   then $M_0$ <- gi
           else break
13        For grid cell $g_j$ in $g_d.l_g^t$      do
14           If    $t_{cur} + t_{jd} \leq$ r.d.l    then $M_d$ <- $g_j$
             else break
15    While (S is empty)           do
16      For all grid cells $g_i$ in $l_g^d$          do
            $S_o$ <- Find rides whose $t_f$ is no later than r.o.l-$t_{cur}$.
            Repeat step 5 for $S_o$
17       For all grid cells $g_j$ in $l_g^d$      do
            $S_d$ <- find rides whose $t_f$ is no later than r.d.l-tcur.
            Repeat step 5 for $S_d$
18          S<- $S_o \cap S_d$
19       For each ride in S              do
            S.route [] <- calculate stopping points (analyzing coordinates along the path)
20        If any point in S.route [] = $g_o$ and any point in S.route []= $g_d$  do
             MD= {selected ride}
          Else
             Deviation distance← allowed predefined ride deviation distance.
21           Measure distance from $g_o$ to S.route [] points $\leq$ r.o.l and distance from
             $g_d$ to S.route [] points $\leq$ r.d.l
22        If (distance ($g_o$, S.route [] ) $\leq$ deviation distance) and
                          distance ($g_d$, S.route []) $\leq$ deviation
          Distance)
     ———— MD= {selected ride}
          Else break
23        For all rides in MD            do
     |        Measure distance between passenger pickup and delivery point
     |        Distance (r.o, r.d) on that ride route
     |——      Shortest distance among all distances will be selected and saved in MD
24   |        Return MD
```

Working of Algorithm

To understand the working of algorithm, suppose a request r comes at time t_{cur}, r source is at g_6 grid cell and destination is at g_4.

Algorithm working is divided into two sections described as follows:

Section 1

MRB algorithm is a bidirectional searching algorithm that searches rides both from source and destination side simultaneously. Detailed working of MRB algorithm as shown in Figure 4 is given below:

First cell selected by the searching algorithm is g_6 form source and g_4 from destination side. The algorithm scans the ride list of the grid cells g_6 and g_4 and selects rides entering in g_6 before r.o.l and in g_4 before r.d.l as shown in equation 1 and 2.

$$t_f \le r.o.l \quad \text{(for source } g_6) \quad 1)$$

$$t_f \le r.d.l \quad \text{(for destination } g_4) \quad 2)$$

Resulting ride list of cell g_6 satisfying the equation 1 is stored in a set S_o and g_2 resulting rides list satisfying equation 2 stored in the set S_d . Algorithm then checks the spare seat availability for all rides in the set S_O and S_d. Rides having spare seats selected from S_0 set will be stored in S_0.In case, no spare seat is available in any ride of set S_0, S_0 will become empty and same process will be repeated for S_d. Afterwards, algorithm finds the intersection among both sets. If any common ride found, it will jump to section 2 and find an optimal minimized route ride.

In other case, if intersection returns the empty set then the algorithm will expand the searching area by scanning the temporal l_g^t list of grid cell g_6 and g_4 at the same time and determines the grid cells which satisfy these equations.

$$t_{cur} + t_{i6} \le r.o.l \quad \text{(for } g_6) \quad 3)$$

$$t_{cur}+t_{i4} \le r.d.l \quad \text{(for } g_4) \quad 4)$$

To select rides with minimum increase in travel distance the algorithm then checks the spatial ordered l_g^d list of both g_6 and g_4 and selects the cells which satisfy equation 3 and 4. Algorithm then traverse the ride list of each selected cell g_s and selects those rides that satisfy these equations:

$$t_f \le r.o.l\text{-}t_{s6} \quad \text{(for } g_6) \quad 5)$$

$$t_f \le r.d.l\text{-}t_{s4} \quad \text{(for } g_s) \quad 6)$$

Vacant seat availability will be checked again for all rides in the set S_0 and S_d and satisfactory rides will be stored in both sets. After taking intersection of source S_o and destination sets S_d, the resulting rides list S will be passed towards section 2.

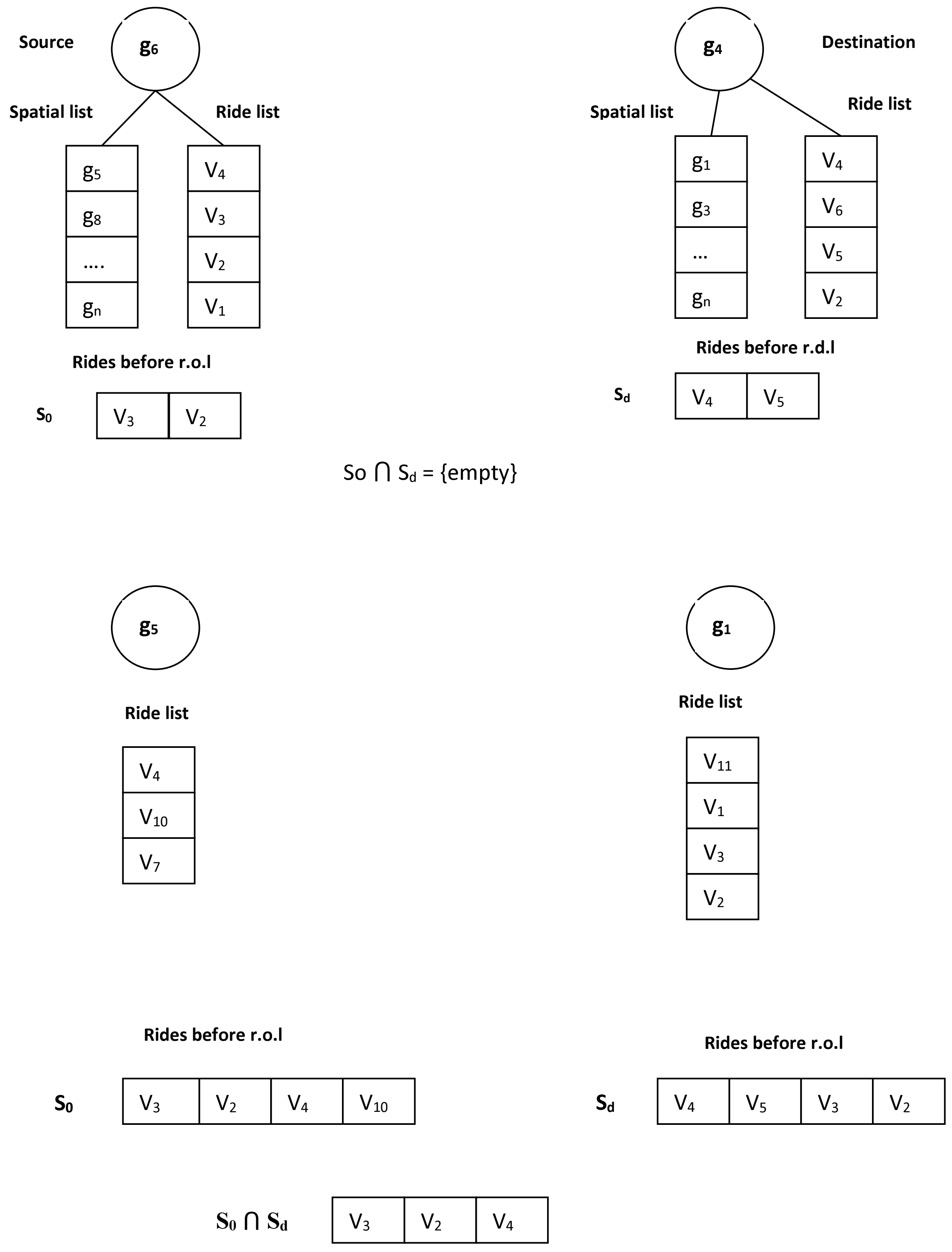

FIGURE 4. Calculation of Ride Set in MRB

Algorithm finds the routes of the each ride in the resulting set S and determines whether passenger is on the ride original route or away from the route as shown in Figure 5. This algorithm finds the commonality of rides at both source and destination side therefore only a small number of rides obtained in the resulting set. Accordingly, if ride contains g_0 and g_d points within its route points (i.e. coordinates) then its deviation is 0 and passenger is on the way of resulting rides routes, algorithm then retrieves the distance from passenger source to destination location for each ride. Consequently minimized distance route ride will be assigned to the passenger. In other case, if passenger is not within the selected ride route, algorithm will measure the deviation distance between the pickup point of r and the routes stopping points of less than r.o.l , and drop off point of r and the routes stopping points of less than r.d.l by using precomputed shortest path technique. If distance is less than the ride giver allowed deviation, then ride will be selected .On contrary; ride will be omitted from the searching space. For all selected rides fulfilling deviation constraint, system will compute the total travel distance between r.o pickup and delivery point r.d including deviation distance. Correspondingly, minimized route ride will be assigned to the passenger.

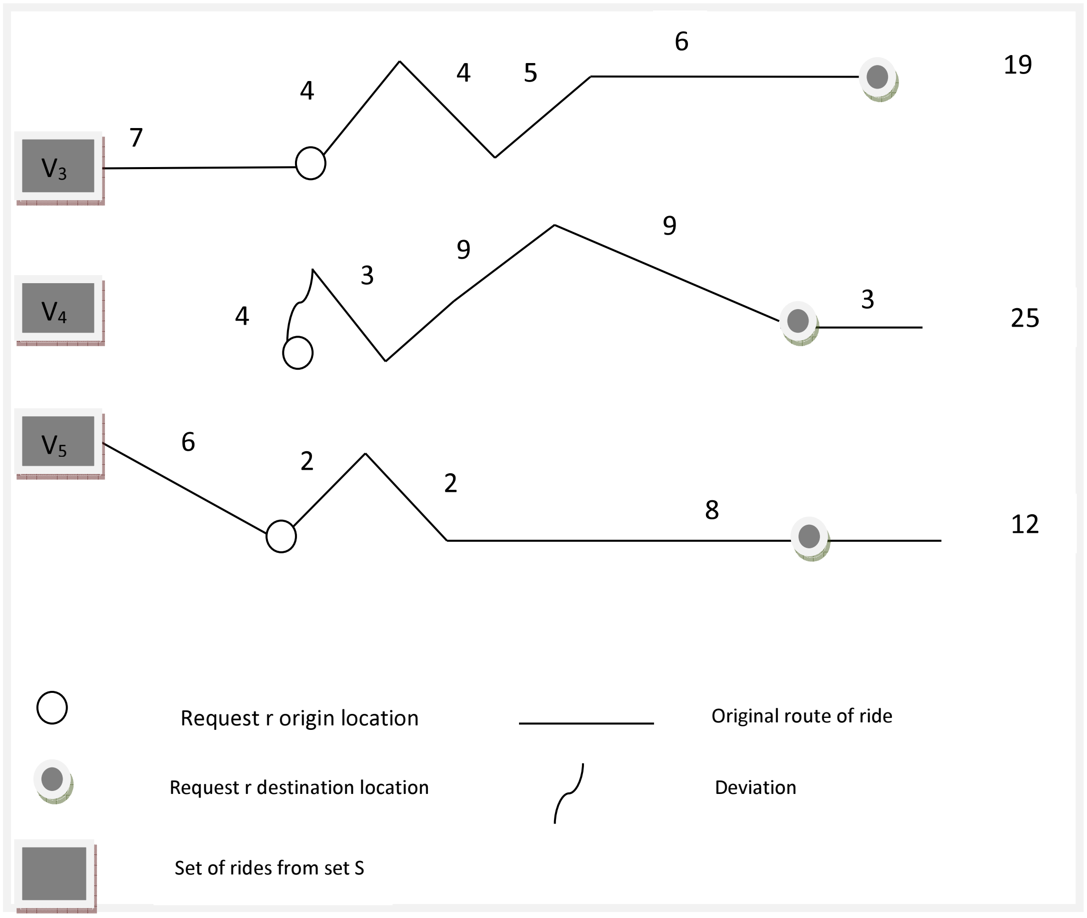

FIGURE 5. Multiple Routes of Ride form set S

Shortest Path Method

In ridesharing for each request shortest path algorithm needs to be executed many times, which will increase the computational burden on the system. Therefore, we are using pre-computed shortest path distance matrix approach. In this scheme, shortest path is computed between every pair of centroids (i.e. geographical centre) and stored in a distance matrix. Whenever we need to calculate the shortest distance between two locations, system will search the distance matrix and retrieves the shortest path that reduces heavy computations. Recently retrieved distances will be added in the cache, to eliminate relooking of the matrix if the same request received again. This scheme is efficient for both small and large road networks compared with invoking shortest path algorithm several times for a single request.

Data Set

Since there exist a lot of adequacy and deficiency of transport planning in Pakistan's most urbanizing city Lahore. Therefore, for developing a sustainable urban transport system in the city, we developed Lahore city ride trajectory dataset. Ride trajectory contains mobility patterns of a ride having 60 sec time interval. Trajectories are recorded by Comsats university Lahore students, faculty members and doctors of Jinnah and General hospital using mobile phones having GPS feature. Trajectories are recorded by of 150 - 200 rides over a period of 20 days in the year of 2014. Therefore, each file contains ride route on a certain path having predefined schedule. The total distance of the data set is more than 500 kilometers.

6. DEMONSTRATION SCENARIO

To practically evaluate our proposed approaches, we develop a prototype system based on realistic Lahore ride trajectories dataset. Requests are generated randomly from the trajectory dataset. To start experimenting, we need current status of a ride (i.e. rides current location), in this work we determine the current status by slicing historical trajectory to a certain time stamp T_b to determine whether a ride is occupied or not? All rides passing through timestamp T_b are considered to be occupied and others are non occupied. To provide feasibility and considering more rides in our experiment we use a temporal parameter. The rationale behind this idea is that: If T_b is set to be 10:00am and some rides don't exactly pass through 10:00am , however, it passes through 9:58am and then 10:03am , then system will not consider it. Therefore, a temporal parameter of 5 minutes is added. In case any ride is passing through within 5 minutes of before and after 10:00 am then system will consider that ride for the experiment also.

On the basis of above mentioned settings we evaluated requests on the prototype, whose screenshot is shown in Figure 6. MRB algorithm always selected one minimized route ride. A red polyline in the screen shot stands for the original route of the ride. Blue points represent the request pickup and drop off points respectively.

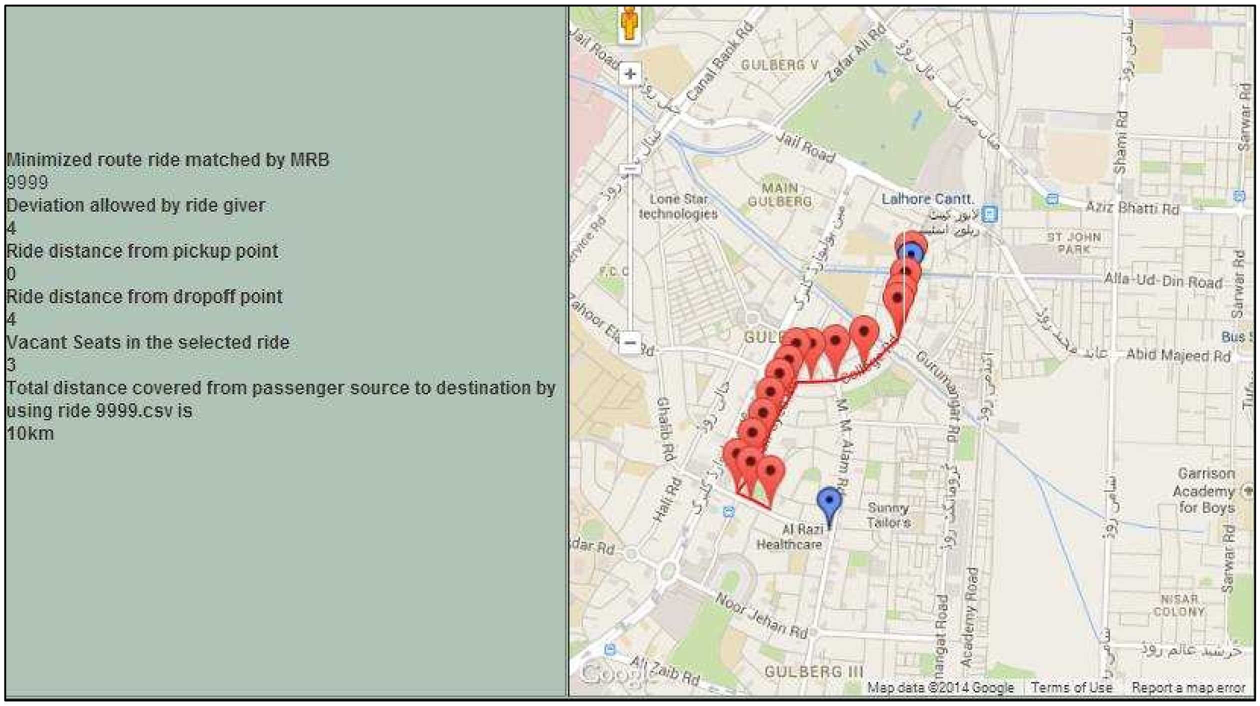
Minimized route ride matched by MRB
9999
Deviation allowed by ride giver
4
Ride distance from pickup point
0
Ride distance from dropoff point
4
Vacant Seats in the selected ride
3
Total distance covered from passenger source to destination by using ride 9999.csv is
10km

FIGURE 6. Screen Shot of Simulation

7. EXPERIMENTAL EVALUATION AND RESULTS

To measure the number of requests generated for a ride on different roads of Lahore. We scanned our collected real world GPS Lahore trajectory dataset and analyze it by mapping each ride route on Google maps, through which we conclude that most visited roads of Lahore are:

1) Mall Road.
2) Raiwind Road.
3) Ferozpur Road.
4) Maulana Shaukat Ali Road.

Figure 7 shows the fluctuation of requests generated during different time periods of a day on four busiest roads of Lahore.

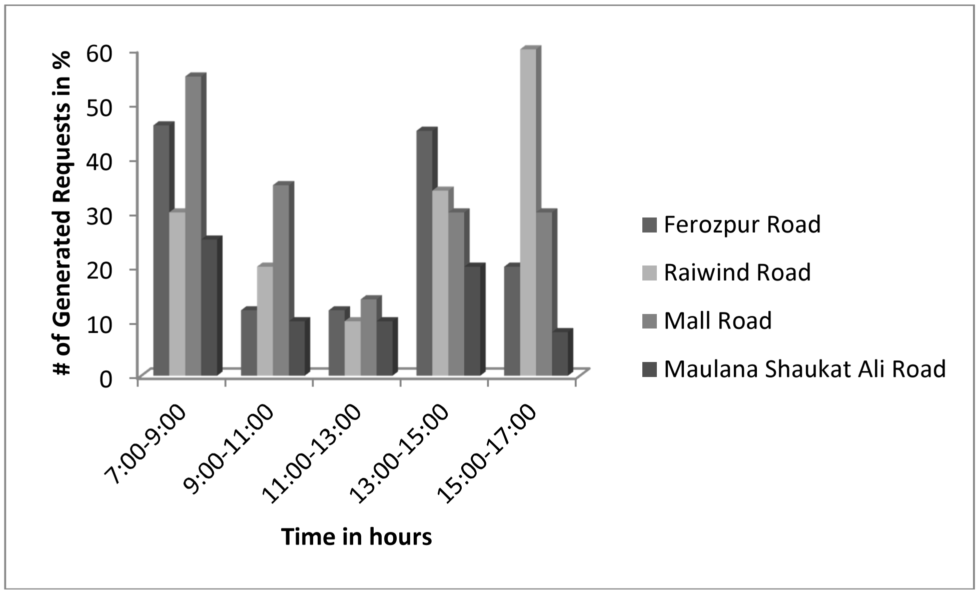

FIGURE 7. Request Generated During Time of a Day

It can be concluded from the above figure that: highest request rate is during (3:00 pm to 5:00 pm) on Raiwind road Lahore.

Furthermore, we analyze our simulation results on the basis of distance and computation cost to optimize the dynamic ride sharing system such that total travel distance gets minimized and computation cost of the ride sharing system becomes low. To measure the effectiveness of our findings we compare our proposed system with T-Share system [10].T_SHARE system uses two matching algorithms, single side and dual side searching algorithms where single side searching algorithm searches spatially closed ride either at source or destination side and found a ride that satisfies its temporal bounds, although dual side algorithm finds spatially and temporally closed ride common at both source and destination side as described in related work section II.

We conduct experiments on ten passenger requests. To measure the Computation cost of our proposed system in comparison with most popular T_share system, we carry out experiments based upon above proposed setting as mentioned in section VI on the following parameters:

- Number of road nodes accessed per query.
- Number of rides accessed per query.
- Vacant seat availability constraint during algorithm execution.

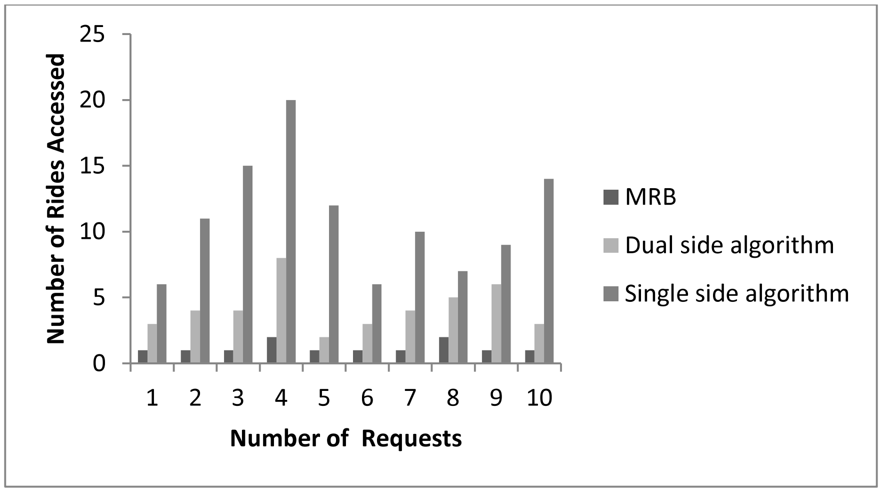

FIGURE 8. Number of Road Nodes Accessed

Since MRB algorithm always proposes a minimized route ride. Therefore, according to the Figure 8, we can determine that MRB accesses one or maximum two rides at each request even if large numbers of rides are available near to request pickup and drop off points as depicted in the case of request 4. However as dual side algorithm finds common rides at both source and destination side therefore it accesses relatively more rides than MRB, single side algorithm accesses a large number of rides due to finding the spatially and temporally closed rides near to request pickup and drop off points .

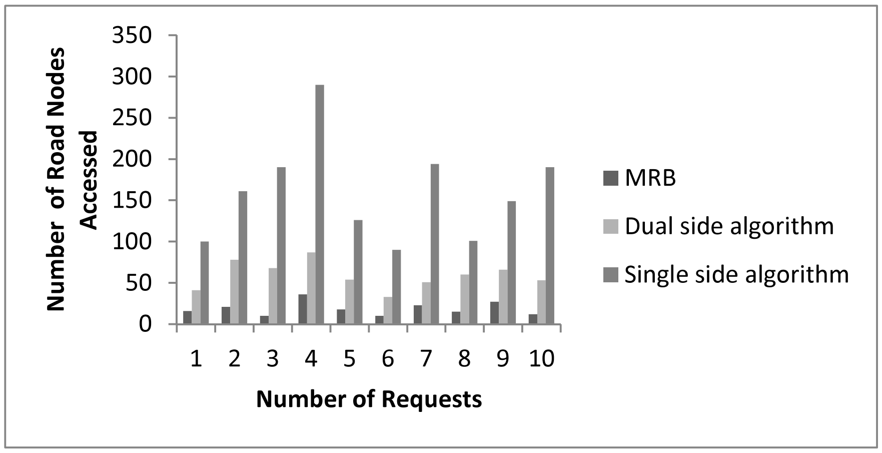

FIGURE 9. Number of Rides Accessed

By analyzing the result shown in **Figure 9** we observe that MRB traverses less number of road nodes since it accesses less number of rides, hence perform less computations.

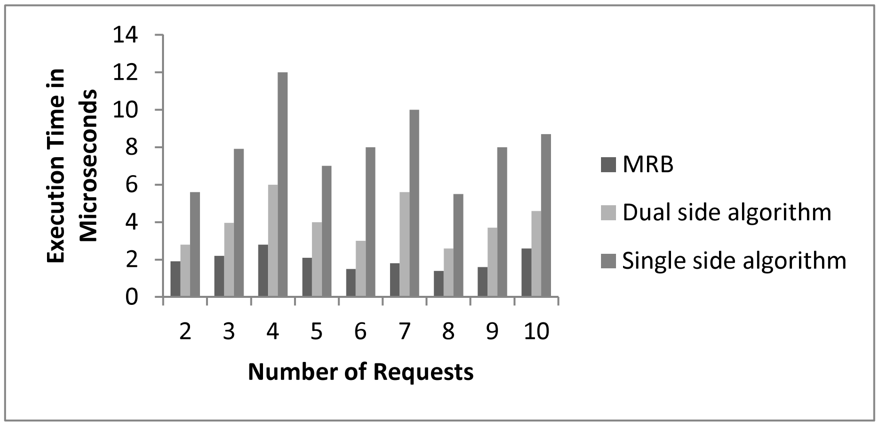

FIGURE 10. Running Time of Algorithms Using Seat Availability Constraint

We compute execution time of the matching algorithms using seat availability constraint during and after algorithm execution. Since in contrast to single and dual side algorithms, MRB integrates seat availability check during algorithm execution, hence we observe from the result shown in Figure 10 that computation cost of the MRB algorithm is significantly lesser than the other two algorithms.

To verify the optimality of our proposed service, we compare the travel distance of the proposed service using MRB algorithm with the single and dual side searching algorithms. Since single and dual side algorithms provide a ride that can pickup and drop off passenger with minimum increase in travel distance, however our system provides the ride with minimum total travel distance of passengers while fulfilling the request .To measure the increase in travel distance of the ride giver and total travel distance of passenger we are using two parameters:

- Deviation.
- Minimize route.

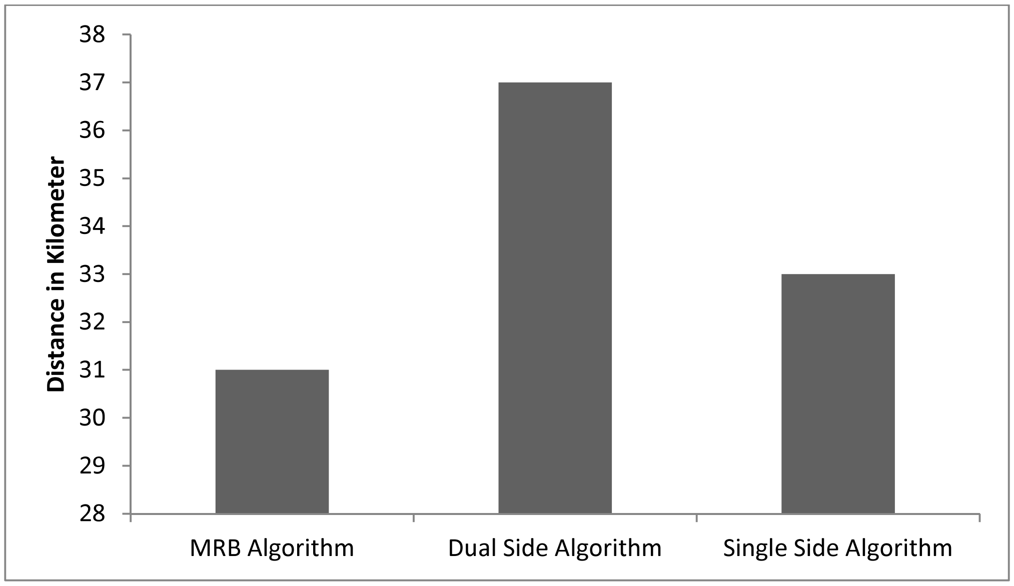

FIGURE 11. Increase in Travel Distance of the Ride Giver

Evaluated results demonstrated that by applying deviation constraint, proposed system saves increase in travel distance as compared to T_share searching mechanisms that allowed deviation at any distance, however, number of missed passenger by applying this constraint is one out of ten passengers. Since, in contrast to T_SHARE service aim of our proposed service is not maximizing the passengers. So we will neglect this effect.

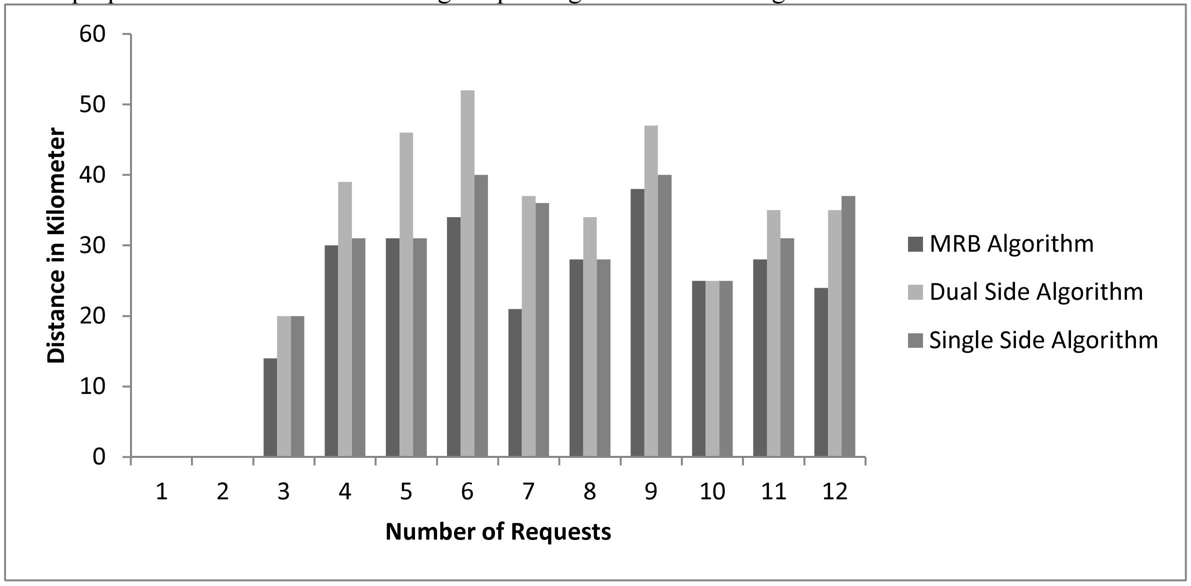

FIGURE 12. Total Travel Distance of the Passenger

Evaluated results demonstrated that by applying minimized route constraint, proposed service significantly reduces the travel distance as compared to the T_Share system.

By summing up all results on computation cost and distance, we can describe our findings in the following table:

Table 2:Results

	Percentage saved by MRB Algorithm over Single Side Algorithm			**Percentage saved by MRB algorithm over Dual Side Algorithm**		
Computation Cost	*No of Road Nodes Accessed*	*No of Rides Accessed*	*Total Execution Time*	*No of Road Nodes Accessed*	*No of Rides Accessed*	*Total Execution Time*
	79%	82%	71%	60%	66%	47%
	Percentage saved by MRB Algorithm over Single Side Algorithm			**Percentage saved by MRB algorithm over Dual Side Algorithm**		
Distance	*Increase in Travel Distance of the Ride Giver*	*Total Travel Distance of the Passenger*		*Increase in Travel Distance of the Ride Giver*	*Total Travel Distance of the Passenger*	
	6%	15%		16%	26%	

Therefore, we can state that our system is efficient and provides an optimal solution for finding an efficient ride match.

8. CONCLUSION AND FUTURE WORK

This article was dedicated to propose a more optimal searching mechanism by incorporating many participant constraints to reduce the total travel distance of the ride partners (i.e. ride taker and ride giver).

In operational aspect, we develop MRB algorithm by modifying dual side algorithm to solve the specific case of the problem (i.e. optimization). In feasibility aspect, we provide ride matches by satisfying many rider and driver constraints (i.e. time, vacant seat available, deviation set by ride giver, minimized route).

The effectiveness and efficiency of our proposed mechanisms are carefully investigated by a series of experiments on realistic Lahore city GPS trajectory dataset in different metrics. By analyzing experimental results that compare different methods, with the focus on reducing computations to minimize the total travel distance of the ride partners, we conclude that our mechanism provides an optimal solution. For instance, in case of ten requests, our service saved on average 26% travel distance and 47% computations as compared to dual side algorithm and 71% computations and 15% travel distance as compared to single side algorithm. Suppose a ride consumes 2 liter gasoline per 25km and lets average distance a rideshare traveled in a day is 100 km, then our proposed system saves 62% gasoline per month. On average, our service can answer a request in 1.4ms, i.e., it can serve 2810k queries per hour.

This study motivates a number of important directions for further research. Our study proposes a dynamic ridesharing system by integrating four constraints (i.e. time bound, vacant seat availability check during algorithm execution, deviation constraint set by ride giver and minimized route ride).However a more robust solution can be acquired by incorporating more participant preferences i.e. safety preferences, gender preferences, window seat availability to obtain a more user friendly system.

In future this work can be extended to accommodate multihop ridesharing (i.e. share a ride with multiple drivers). Advanced travel time estimation techniques to improve the prediction of ride travel time can be considered further.

ACKNOWLEDGMENTS

This research paper is made possible through the help and support of Mr. Sol Ma who guided us at every step of this research. Moreover, we dedicate our acknowledgments of gratitude towards our family and friends.

REFERENCES

[1] Downs, A. 2004. Why Traffic Congestion is Here to Stay.... and Will Get Worse. UCTC's Annual Student Research Conference

[2]http://www.unep.ch/etb/publications/Green%20Economy/Reducing%20emissions/UNEP%20Reducing%20emissions%20from%20private%20cars.pdf, Accessed 14 May 2014

[3] Morency, C, 2007.The ambivalence of ridesharing. Transportation.,34(2):239-253

[4] Nordlund, A.M., and Garvill, J, 2003. Effects of values, problem awareness, and personal norm on willingness to reduce personal car use. Journal of environmental psychology., 23(4): 339-347

[5] Brownstone, D., and Golob, T.F, 1992. The effectiveness of ridesharing incentives: Discrete-choice models of commuting in Southern California. Regional Science and Urban Economics ., 22(1): 5-24

[6] Wang, X.2013. Optimizing ride matches for dynamic ride-sharing systems. Dissertation, Georgia Institute of Technology.

[7] Ma, S., and Wolfson, O. 2013. Analysis and evaluation of the slugging form of ridesharing. Proceedings of the

21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 64-73

[8] Agatz, N., Erera, A., Savelsbergh, M., and Wang, X.2012. Optimization for dynamic ride-sharing: A review. European Journal of Operational Research., 223(2):295-303

[9] Agatz, N., Erera, A.L., Savelsbergh, M.W., and Wang, X. 2011. Dynamic ride-sharing: A simulation study in metro Atlanta. Procedia-Social and Behavioral Sciences., (17): 532-550

[10] Ma, S., Zheng, Y., and Wolfson, O.2013. T-share: A large-scale dynamic taxi ridesharing service. IEEE 29th International Conference , pp. 410-421

[11] Haddad, Y., Cohen, Y., and Goldsmith, R.2013. A Dynamic Real Time Car Sharing System. International Journal of Soft Computing and Software Engineering., (3): 872-888.

[12] Yamamoto, K., Uesugi, K., and Watanabe, T.2008.Adaptive routing of cruising taxis by mutual exchange of pathways. Knowledge-Based Intelligent Information and Engineering Systems, pp. 559-566

[13] Lu, J.-L., Yeh, M.-Y., Hsu, Y.-C., Yang, S.-N., Gan, C.-H., and Chen, M.-S.2012. Operating electric taxi fleets: A new dispatching strategy with charging plans. Electric Vehicle Conference (IEVC), 2012 IEEE International, pp. 1-8

[14] Yuan, N.J., Zheng, Y., Zhang, L., and Xie, X.2013. T-finder: A recommender system for finding passengers and vacant taxis. Knowledge and Data Engineering, IEEE Transactions on, 25(10): 2390-2403

[15] Tian, C., Huang, Y., Liu, Z., Bastani, F., and Jin, R. 2013. Noah: A dynamic ridesharing system. Proceedings of the 2013 international conference on Management of data, pp. 985-988

[16] Yuan, J., Zheng, Y., Zhang, L., Xie, X., and Sun, G.2011. Where to find my next passenger. Proceedings of the 13th international conference on Ubiquitous computing , pp. 109-118

[17] Yuan, J., Zheng, Y., Zhang, C., Xie, X., and Sun, G.-Z.2010. An interactive-voting based map matching algorithm. Mobile Data Management (MDM), 2010 Eleventh International Conference on, pp. 43-52

[18] Ge, Y., Xiong, H., Tuzhilin, A., Xiao, K., Gruteser, M., and Pazzani, M. 2010. An energy-efficient mobile recommender system. Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 899-908

[19] Thaler, D., and Teredesai, A.2009. Maximizing Ridesharing Matches for Dynamic Carpooling. Intelligent Environments , pp. 489-492

[20] Herbawi, W.M., and Weber, M. 2012 .A genetic and insertion heuristic algorithm for solving the dynamic ridematching problem with time windows. Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference , pp. 385-392

[21] Greco, F. 2008. Traveling Salesman Problem, InTech.

[22] Savelsbergh, M.W. 1985. Local search in routing problems with time windows. Annals of Operations research., 4 (1): 285-305

[23] Amey, A.M.2010. Real-time ridesharing: exploring the opportunities and challenges of designing a technology-based rideshare trial for the MIT community, Dissertation, Massachusetts Institute of Technology.