2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Efficient Taxi and Passenger Searching in Smart City using Distributed Coordination

Anmol Agrawal
Dept. of CSE
IIT Roorkee
Roorkee, India
anmol.agrawal17@gmail.com

Vaskar Raychoudhury
Dept. of CSE
Miami University
Oxford, OH, USA
vaskar@ieee.org

Divya Saxena
Dept. of Computing
The Hong Kong PolyU
Kowloon, Hong Kong
divya.saxena.2015@ieee.org

Ajay D Kshemkalyani
Dept. of CSE
Univ. of Illinois at Chicago
Chicago, IL, USA
ajay@uic.edu

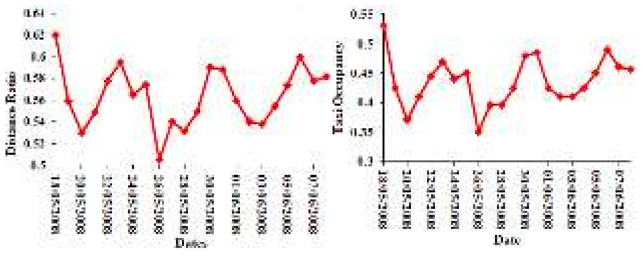
Abstract— **Taxicabs are an important element of urban public transportation. Taxicabs either cruise through city streets in search of passengers or wait at several hotspots (like airports, rail stations, malls, stadiums, taxi stands, etc). Cruising by empty Taxis increases city traffic and carbon footprint while reducing net profit. Alternatively, there might be places where passengers are waiting long for taxis. In order to improve coordination between taxis and passengers with a view to decrease passenger waiting time and to increase taxi profits, we propose a taxi selection algorithm (TSA) as well as a hotspot recommendation approach (HRA). While the proposed *TSA* achieves its objective through distributed coordination among the participating taxis and passengers, the *HRA* uses a clustering approach over a large-scale taxi dataset to pin-point hotspots. The main contribution of this paper lies in extensive experimentation using large-scale taxi dataset of SFO to show that the *TSA* outperforms existing taxi selection algorithms by finding a taxi which can reach the passenger in minimum time with up to 97.59% accuracy. We also evaluate the *HRA* using another taxi dataset from NYC which shows that 60% of the times, a taxi will get a passenger following our recommendation scheme.**

Keywords—Taxi Selection; Hotspot Recommendation; Smart Transportation; Distributed Coordination; Data Analysis

I. INTRODUCTION

Taxicabs provide passengers a comfortable and faster mode of transport with pickups and drop offs at desired place and time. Although there are significant demands of taxis at airports, train stations and stadiums (during matches), it is not hard to find empty taxis in stands (called, waiting) or ploughing city streets (called, cruising) in search of passengers. Many a times, passengers wait for long at designated sites for taxis to arrive. So, in efficient taxi service provisioning, the critical underlying challenge lies in connecting a waiting passenger with a waiting / cruising taxi with minimum possible delay. This challenge emanates from the goal to select the taxi, which can reach the passenger location in shortest time, and this is not possible, if any random taxi is selected. This can reduce waiting time for passenger and waiting / cruising time for taxis. Moreover, it can reduce fuel consumption and traffic congestion (resulting from low-speed cruising) both of which have monetary as well as environmental significance. Also, finding a passenger more efficiently will increase the profit of taxi drivers by increasing overall taxi occupancy time.

According to our analysis done on GPS traces of 535 taxis in San Francisco [7] for a period of 22 days, it can be seen that most of the distance travelled by taxis is while *cruising*. Fig. 1(a) shows the ratio of distance travelled by taxis while it was occupied (i.e., carrying passenger) to the total distance they travelled in the day. The average ratio is 0.56, which is quite low. This means that half of the distance is cruising, which directly reduces their profit. Fig. 1(b) shows the *Taxi occupancy rate* (ratio of the time a taxi is occupied to the total time it was on the street) which also peaks only at 0.53.



(a) Distance Ratio of Taxis in SFO (b) Occupancy Rate of Taxis in SFO

Fig. 1. Preliminary Taxi Data Analysis

In this paper, we aim to address this challenge by developing a distributed taxi searching solution for passengers which works by localized coordination between nearby taxis through a hop-by-hop message transfer. We further enhance profitability of taxi drivers by recommending them nearby locations with higher chances of finding passengers.

Researchers have proposed both centralized as well as distributed solutions for taxi selection. Centralized taxi dispatching solutions finds a taxi, which is nearest to the passenger, or might take shortest time to reach [5], or by suggesting to taxi drivers, the most probable place to get the next passenger [10]. Some researchers used machine-learning algorithms for finding number of vacant taxis in an area based on day, time and weather conditions [12].

Given the scalability concern of centralized systems, distributed taxi selection algorithms have been investigated where passenger requests are propagated in a multi-hop manner until a free taxi is found (or up to a certain number of hops). One critical challenge in distributed taxi selection approach is Blocking Time where unoccupied taxis participating in a selection process are barred to participate in parallel selection processes. Since, only one taxi will be selected finally in a selection process, other unoccupied taxis perceive this blocking period as a wastage of opportunity. Longer the blocking period the taxi is losing profits proportionally. EZCab [2] deploys a Probabilistic Proactive routing protocol for selecting taxis which can reach the passenger in minimum time. However, they actually find the first free taxi assuming that it will reach the passenger in minimum time. Also, they did not address the blocking problem. Sheu, et al. [3] proposed a protocol for selecting a taxi with minimum driving distance by sending replies following the driving path. However, they did not study how much time the selected taxi takes to reach the passenger which determines the passenger waiting time. They also focused on reducing taxi blocking time.

In order to increase taxi occupancy rate, taxi drivers can be suggested best locations (called *hotspots*) to look for passengers. Hotspot recommendation to taxi drivers have been studied by many researchers by analyzing the pickup pattern of passengers [13], by developing recommendation systems [14] or by calculating regions generating high revenues [15]. Some of these approaches [14][15] require continuous GPS traces to evaluate which is costly to collect and to process. Although the aforementioned research works have added significant contribution to the domain, they have not used real dataset to evaluate their proposed schemes. Since, this is an application problem, experimenting with real data helps us to test the system more critically.

In this paper, we have proposed a taxi selection algorithm, called TSA, with dual purpose – (1) to select the taxi which can reach to passenger's location in shortest time (thereby reduces passenger waiting time), and (2) to reduce the blocking time of unselected taxis (to increase taxi availability). *TSA* works by creating a dynamic tree overlay on the taxi communication network with the passenger as the root. Requests are wirelessly disseminated over the tree for certain pre-specified number of hops. Later, the taxi responses are collected by converge cast and the passenger gets one or more responses among which s/he chooses one taxi. We also proposed a hotspot recommendation scheme, named *TAR* which is similar to [13]. It uses taxi trip information (containing *passenger's pickup and drop locations* and corresponding *time*) as well as pick up pattern of passengers and summarily outperforms the scheme proposed in [13]. We use *Grid Clustering* and *K-means Clustering* algorithms to find the hotspots, and provide a scoring scheme for suggesting a nearby hotspot to a taxi driver. In summary, we make the following contributions in this paper.

- We used the Google API to calculate journey time between two locations, which is highly practical. Other papers calculated journey time based on average taxi speed, which does not give realistic estimates.
- We propose a distributed algorithm to select a taxi which can reach the passengers in minimum time after s/he fires a request. We used real taxi traces to evaluate our algorithm and did a simulation using actual locations of passengers and taxis which, to the best of our knowledge, no other papers did. Using real data gives exact scenario. Significant empirical evaluation over San Francisco [7] taxi GPS traces shows that our algorithm selects the taxi *requiring minimum time* with up to 97.59% accuracy.
- We also propose a hotspot recommendation approach for taxi drivers by applying data mining techniques on the historical taxi trip dataset. Empirical evaluation using 143.35 million trip records from New York City (NYC) [9] Taxi GPS traces show that 60% of the times, a taxi will find a passenger while following our recommendation. No other prior research works used such large data set for evaluation. Furthermore, our experiments with the huge NYC taxi data goes to prove the scalability of our algorithm.

II. Related Works

In this section, we shall discuss research works related to taxi selection and hotspot recommendation to taxi drivers and passengers.

A. Taxi Selection

Two popular methods are *centralized* and *distributed* taxi selection and dispatching approaches. Centralized taxi selection approaches select taxi based on its distance from the requesting passenger or the time it takes to reach the passenger's location [5]. Performance improvements of taxi selection have been achieved by more accurately locating passengers either by providing a route to cruising taxi with high expectation of passengers [10] or using GSM based location detection of passengers while booking a taxi [11]. Phithakkitnukoon et al. [12], used Bayesian classifier with a sequential error-based learning algorithm for finding number of vacant taxis in an area based on day, time and weather conditions. Though performing well, centralized functioning of taxi dispatching systems are less scalable and suffer from a bottleneck.

Distributed taxi selection works by exchanging messages between passenger and nearby taxis using short-range wireless communication. Zhou et al. [2], proposed an application EZCab, which allows passengers to use mobile phones to send a taxi-booking-request which propagates from taxi to taxi until a free taxi is found or up to a maximum of 20 hops. The message routing protocols used in EZCab are flooding, Probabilistic On-Demand, and Probabilistic Proactive where the last one is found to select a taxi, which can reach the passenger in minimum time although it is not guaranteed. Moreover, EZCab did not provided any solution for reducing taxi *b*locking time. Sheu et al. [3], proposed a protocol for selecting a taxi with minimum driving distance by sending replies following the driving path. So, the taxi, from which the reply message is first received, by the passenger, is selected. Due to different reasons, like traffic congestion, it cannot be guaranteed that the physically nearest taxi can reach the passenger in minimum time. They focused on reducing taxi blocking time by sending staying (STA) messages to the blocked taxis which increases communication overhead. D'Orey et al. [4], proposed a distributed taxi selection system for ride sharing, which aims to select a taxi providing minimum trip fare. The afore-mentioned algorithms do not provide any solution for selecting a taxi which reaches in minimum time to passenger's location.

B. Taxi Recommendation Systems

Researchers analyze passenger pick up patterns to identify top pick up points and suggest such locations to the taxi drivers in order to improve their occupancy and profitability. Lee et al. [13], used K-means [20] clustering to analyze pickup pattern of passengers across 100 clusters. Yuan et al. [14], built a recommendation system for both taxis and passengers using the Beijing taxi dataset, and proposed an algorithm for finding possible parking places for empty taxis. Powell et al. [15], developed a Spatio-Temporal Profitability Map for taxi drivers by dividing a region into grids and calculating profitability of a location using historical dataset. Zhang et al. [16], have used L1 SVM for finding (and differentiating between) efficient and inefficient passenger searching strategies for taxi drivers and provided a correlation between service strategies and revenues. Ge et al. [17], proposed an energy-efficient recommendation approach, which provides taxi drivers with a sequence of locations such that the driving distance to get a passenger is minimized. Moreira-Matias et al. [18], proposed an online recommendation to choose taxi stands for taxi drivers using time-series forecasting technique to predict spatio-temporal distribution of passengers in real time.

While [13] used dataset containing trip information (*passengers' pickup and drop location*, and *time*) of taxis to find the locations where the probability of finding a taxi is high, other recommendation models [14][15] used continuous and dense GPS traces which are costly to collect and to maintain. Our recommendation model is different from others as it uses real dataset having taxis' trip information and we analyze passenger pick up patterns across weekdays and weekends considering both peak and non-peak hours in order to decide the hotspots for both taxis and passengers.

III. Proposed Work: Taxi Selection Algorithm

In this section, we present the details of our taxi selection algorithm. Depending on whether the taxi is having passenger or not, taxi can be in one of the following states: *Occupied*, *Unoccupied* or *Blocked*. Fig. 2 shows the transition among these taxi states. All the variables and messages used in our algorithm are listed in Table I and Table II, respectively.

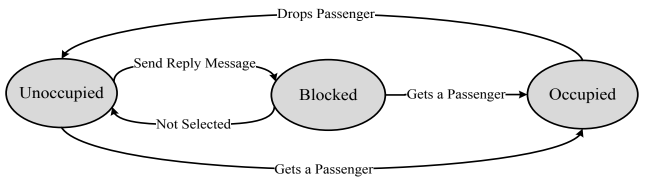


Fig. 2. Taxi State Change Diagram

TABLE I. Variables Used

Variable	Significance
T_i	Taxi $T_i \in \{T\}$ where T is set of all Taxis
T^o	Set of all Occupied Taxis
T^u	Set of all Unoccupied Taxis
P_i	Passenger $P_i \in \{P\}$ where P is the set of all passengers
$Succ_x$	Set of all neighbor taxis from Taxi/Passenger x
P_id_i	Id of Passenger P_i
T_id_j	Id of Taxi T_j
L_{Pi}	Location (Latitude, Longitude) of Passenger P_i
L_{Ti}	Location (Latitude, Longitude) of Taxi T_i
ΔT_i	Tolerance Time (MaxWaiting Time) given by Passenger P_i
PAR_t	Parent of Taxi t in the Diffusion Tree
PRT_t^P	Passenger Reaching Time (Time for Taxi t to reach P)
Max_Hop _Count	Maximum hops to which P_i 's request will be forwarded

TABLE II. Message Table

Messages	Significance
REQ (*P_id*, L_P, ΔT, *Hop_Count*, *PRT*)	Request sent/forwarded by Passenger/taxis
REP (*P_id*, *T_id*, L_P, L_T, *PRT*)	Taxis send reply message to their parent
ACCEPT (*P_id*, *T_id*)	Passenger/Taxi sends Accept message to its child with minimum PRT
CONF (*P_id*, *T_id*)	Confirmation sent by Passenger to a selected taxi

A. Distributed Taxi Selection Algorithm

Problem: Given a passenger P_i at location L_{Pi} and a set of taxis T located 'near' (within a radius d of) P_i, select a taxi $T_j \in T$ at location L_{Tj} such that $PRT_{\mathrm{Tj}}^{Pi} \le PRT_{\mathrm{Tk}}^{Pi}$ where, $T_k \in (\{T\} - T_j)$.

```
                         Initialization:
1.  $\forall t_o \in T^o$, $State_{to} \leftarrow Occupied$
2.  $\forall t_u \in T^u$, $State_{tu} \leftarrow Unoccupied$
                  Tree Creation Phase at $P_i$:
3.  $\forall t \in Succ_{Pi}$, send REQ to $t$
4.  Start $RT_{Pi}$, $ST_{Pi}$
                  Tree Creation Phase at $T_j$:
5.  Receives REQ from $P_i/T_i$ AND set $PAR_{Tj} \leftarrow P_i/T_i$
6.  Start $RT_{Tj}$, $ST_{Tj}$
7.  if State $T_j = Unoccupied$,
8.      Calculate $PRT_{Tj}^{Pi}$ , send REP to $PAR_{Tj}$
9.      Start $AT_{Tj}$
10. if $TTL \neq 0$,
11.     Decrement $TTL$ by 1
12.     if $PRT_{Tj}^{Pi} < PRT_{Req}$
13.         Update $PRT_{Req} = PRT_{Tj}^{Pi}$
14.     $\forall t \in Succ_{Pj}$, send REQ to $t$
                  Convergence Phase at $T_j$:
15. if ACCEPT received before $AT_{Tj}$ expires
16.     Wait for CONF from $PAR_{Tj}$
17. else
18.     $State_{Tj} \leftarrow Unoccupied$
19. $\forall t \in Succ_{Tj}$, wait to receive REP from $t$ until $RT_{Tj}$ expires
20.     Send ACCEPT to t with min. $PRT_t^{Pi}$
21. if $ST_{Tj}$ expires
22.     $\forall t \in Succ_{Tj}$, Send REP with min $PRT_t$ to $PAR_{Tj}$
23. When CONF message $C_m$ received
24.     if $C_m \in T_j$,
25.         $State\ T_j \leftarrow Occupied$
26.         Head towards $P$
27.     else
28.         Forward $C_m$
                  Convergence Phase at $Pi$ :
29. $\forall t \in Succ_{Pi}$, wait to receive REP from $t$ until $RT_{Pi}$ expires
30. Send ACCEPT to $t$ with min $PRT_t^{Pi}$
31. if $ST_{Pi}$ expires
32.     Send CONF to $t$ with minimum $PRT_t^{Pi}$
```

Fig. 3. *TSA* Algorithm

Our algorithm (Fig. 3) starts when a passenger (P_i) looking for a taxi broadcasts a REQ message from his smartphone. Each of the neighboring taxis receiving the REQ, sets the sender as its parent and decides on its course of action depending on its current state. If a duplicate REQ is received, that is simply ignored. When an unoccupied taxi T^u receives a REQ (from P or any other intermediate node), it calculates the time (PRT_{Tu}^{Pi}) to reach P_i's location and iff that is less than the PRT mentioned in the REQ, then T^u replaces the original PRT with PRT_{Tu}^{Pi} and forwards the REQ message to neighbors except the parent node and starts the Selection

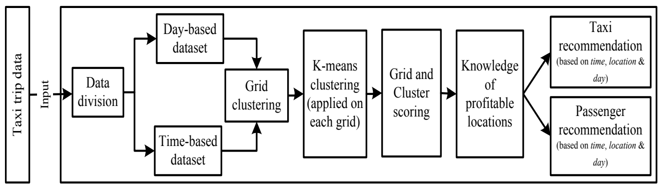
(ST) and Rejection (RT) timers. If T^u is at the maximum permissible distance from P, it is a leaf node and it does not forward the REQ message any further and instead sends a REP back to its parent iff it's PRT_{Tu}^{Pi} is less than the PRT received through REQ and starts an Accept timer (AT). Otherwise, the leaf node unblocks itself. When an intermediate node receives one or more REP messages from its successors until its RT expires, it chooses the child which sent the REP with minimum PRT and sends an ACCEPT message. All child nodes which sent the REP to their parent but did not receive an ACCEPT message before the AT expires will unblock themselves. In this way, REP messages converge towards the tree root and finally the passenger chooses the minimum PRT node among its children and sends a *CONF*.

B. *Greedy Taxi Selection Algorithm*

We also develop the greedy approach of *TSA* (G-TSA) where a taxi will only forward REQ further if its passenger reaching time is greater than the tolerance time set by the passenger. It is similar to the EZCab approach where they try to find the first free taxi and only ACCEPT timer is used to unblock the taxis. Taxi sends the REP as soon as it receives REQ, and it is forwarded immediately towards the root. The passenger sends CONF to the taxi whose REP arrived first. So, according to this approach, an *Occupied* taxi with minimum Haversine distance receives the REQ first and sends the REP. Hence, it selects the taxi with minimum Haversine distance assuming that a taxi can reach fast if its REP is received first.

IV. Hotspot Recommendation Model for Taxis & Passengers

In this section, we first outline the model framework and the data mining techniques used followed by the detailed working description of the model.



Taxi trip data
Input
Data division
Day-based dataset
Time-based dataset
Grid clustering
K-means clustering (applied on each grid)
Grid and Cluster scoring
Knowledge of profitable locations
Taxi recommendation (based on *time*, *location* & *day*)
Passenger recommendation (based on *time*, *location* & *day*)

Fig. 4. Recommendation Model Framework

A. *HRA Model Framework*

Fig. 4 illustrates the framework of the Recommendation model built by using NYC taxi dataset (yellow taxis) [9], which consists 143,352,415 (≈ 143.35 million) entries where each entry contains data for one trip. Each trip has *passenger's pickup* and *drop location* (geographic coordinates), and corresponding times. We use 8 months' (from January to August) containing 97,991,425 (≈ 98 million) (66.66% of the whole dataset) trip records as training dataset, while we use rest as the testing dataset. The dataset is divided into subsets based on the trip *time* and *day of the week*. We further apply data mining approaches on those subsets for finding the various profitable locations in the city. Below, we briefly discuss the various steps involved in building the model and its working.

B. *Data Division*

Fig. 5 illustrates average number of passenger requests arriving on different days of the week and different times of the day for the month of August, 2016 in NYC [9]. Fig. 6 illustrates the passenger count at different time of the days for a period of 22 days in San Francisco.

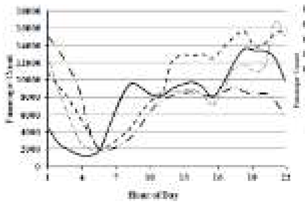


Fig. 5. Average Passenger Count in NYC (August, 2016)

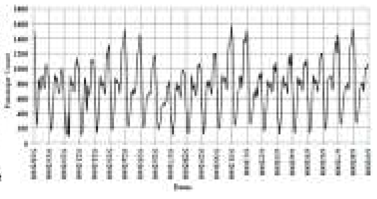


Fig. 6. Passenger Count in San Francisco

The passenger patterns mostly changes on weekends but does not vary much on weekdays. During Friday nights, most number of requests are generated. Hence, data is divided into four categories: *Friday*, *Saturday*, *Sunday* and *Weekdays* (remaining days). Then there is a huge variation on different times of a day. The number of requests is more in the morning and evening than any other time of the day on weekdays. Thus, the dataset is further divided according to the *time*.

C. *Data Mining Approaches*

After dividing the dataset into subsets, the datasets are clustered based on the trip's pickup location using Grid Clustering, so that, the city is divided into square grids (areas) of size 1 sq. km (Fig. 7). Fig. 8 shows pickup points clustered in a grid. Since, a grid area is large, we further classified it into small clusters using K-means clustering which provides cluster centroids that can be used to define the cluster's location for the recommendation. In this algorithm, we need to provide the number of clusters (K) in advance. Fig. 9 shows the clusters formed inside a grid. We used these clustering techniques because of their linear time complexities.

Fig. 7. NYC Grid Structure

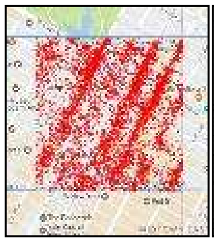


Fig. 8. Pickup Points Clustered in a Grid

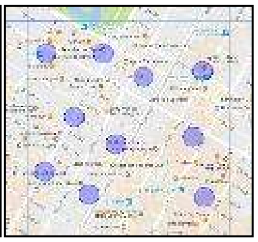


Fig. 9. Clusters formed inside a Grid by K-means

1) The value of K is decided such that the maximum number of clusters inside a grid does not go beyond a number G_k. The value of K is calculated using the formula given in Eq 2.

$$G_k = \frac{\sqrt{N_{max}/2}}{\mathrm{R}} \quad \ldots (1) \qquad K_i = \frac{\sqrt{N_i/2}}{\mathrm{R}} \quad \ldots (2)$$

R is the number which is calculated using the value of G_k and, N_{max} is the number of points present in the grid with maximum points in it. N_i is the number of points present in the grid G_i.

2) Clustering Output: After dividing the data, we perform clustering of data of each time slot of all the days. Hence, we have a set of clusters for each hour of the four different days

(Ref. *Section IV(B)*). Fig. 10 and 11 shows the heatmap of trips' pickup location in the morning time on weekdays and evening time on Friday, respectively. This represents that some locations have more trips than others. Fig. 12 and 13 represent the clusters formed during the weekdays' morning (9 AM to 10 AM) and evening (6 PM to 7 PM) time, respectively. These two figures clearly show that change in time changes the locations from where more passengers' request originates. Similarly, Fig. 14 represents the clusters formed during Saturday's morning and evening time. Fig. 15 represents the clusters formed in the evening time of weekdays and Saturday. This figure shows even at the same time, different days (Weekdays and Saturday) have different clusters representing changing behavior of passenger requests with days. The value of threshold used to calculate *GS* and *CS* is the mean value of grid and cluster in a time slot, respectively. The maximum clusters inside a grid is taken as 16 (each cluster size will be greater than 250 sq. meters).

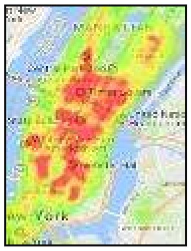


Fig. 10. Heatmap of Pickup Locations on Weekday Mornings

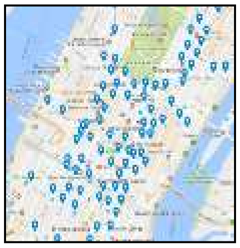


Fig. 12. Clusters for Weekday Mornings

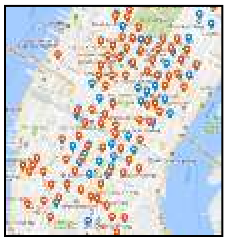


Fig. 14. Clusters for Saturday Mornings (Blue) and Evenings (Red)

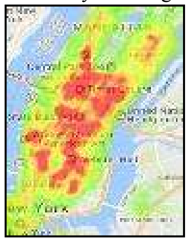


Fig. 11. Heatmap of Pickup Locations on Friday Evenings

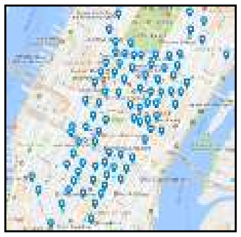


Fig. 13. Clusters for Weekday Evenings

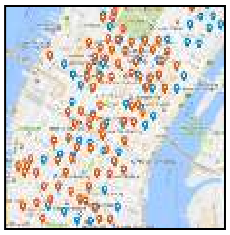


Fig. 15. Clusters for Weekday (Blue) and Saturday (Red) Evenings

D. Grid and Cluster Scoring

After obtaining the clusters, we calculate their profitability using *Grid Score* (GS) and *Cluster Score* (CS), and finally these scores are used to provide a recommendation. We need to normalize both *GS* and *CS*, so that they do not dominate the other one while calculating the final score for the recommendation.

1) Cluster Score (CS): It is calculated by dividing the total number of points lying inside the cluster by the cluster threshold. Cluster threshold is taken as the mean of points present inside the clusters as shown in Eq 3. The cluster score calculation is given by the Eq 4.

$$T_c = \frac{\sum N_i^C}{C} \quad \ldots (3) \qquad CS_i = \frac{N_i}{\mathrm{T_c}} \quad \ldots (4)$$

where, T_c is the cluster threshold, N_i^C is the number of points in cluster i, C is the total number of clusters, and CS_i is the *CS* of cluster i.

2) Grid Score (GS): It is calculated similar to *CS* where a cluster is replaced by a grid. Grid threshold is taken as the mean of points present inside the grids. Eq 5 and 6 represents the calculation of Grid Score.

$$T_G = \frac{\sum N_i^G}{G} \quad \ldots (5) \qquad GS_i = \frac{N_i}{T_G} \quad \ldots (6)$$

where, T_G is the grid threshold, N_i^G is the number of points in grid i, G is the total number of clusters, and GS_i is the GS of grid i.

E. Taxi/Passenger Recommendation

Taxi recommendation request by passengers contains information about their current location (Lat., Long.), recommendation time and day of the week. In this way, they can also make requests for a later time also and not only for present time. On the basis of all these parameters, nearby locations (w.r.to their current location) are suggested to them.

Recommendation for taxi drivers is not straightforward like the passengers'. We develop a technique to address two concerns – (1) taxis should be directed uniformly to different locations instead of a small set of locations, and (2) distance to the recommended locations should not be very large. So, we need to strike a balance between taxi profitability and taxi availability. In order to achieve this objective we recommend the centroids of the clusters obtained by K-means clustering. Selection of top locations is done by giving a final score to each location based on the distance from taxi's location to the recommended location, *CS* and *GS* (grid to which cluster belong). Eq. 8 shows the calculation of final Score.

$$DS_L = \mathrm{S}_{max} - \frac{\mathrm{D(T,\ L)}\,.\,S_{max}}{\mathrm{R}} \quad \ldots (7)$$

$$FS_L = \alpha . CS_L + \beta . GS_L + \mu . DS_L \quad (8)$$

where L belongs to the set of locations (cluster centroid) within range. FS_L is the final Score of L, CS_L is the *CS* of location L, GS_L is the *GS* of location L, DS_L is the Normalized Distance Score (DS). *DS* is calculated using Eq. 7. S_{max} is the maximum distance score, D (T, L) is the distance from taxi's current location to the selected location, and R is the range within which locations are selected, i.e., the maximum allowable distance.

From these equations, we can see that least distance will have maximum score, i.e., nearby locations are given a better score than locations far from the taxi. α, β, μ are the weightage given to each score. This scoring system will prevent recommending all the taxis to the same location. By adding *DS*, the final score will change as the distance of every taxi will be different from the same location. Taxis which are nearby, however, may get same recommendation.

V. Experiments and Results

In this section, we describe the performance of our proposed algorithms.

A. Performance Analysis of TSA

1) Dataset: We use the taxi dataset of *San Francisco* [7] to check the working of *TSA*. The dataset contains GPS traces of 535 taxis over 22 days (from 18 May, 2008 to 8 June, 2008). After cleaning and pre-processing , we found 414,865 passenger requests among which we select 16,500 passenger requests occurred on 19 - 20 May, 2008.

2) Performance Metrics: Below, we formally define our performance metrics for TSA:

- **Selection Time**: It is the time elapsed between origination of a passenger request and a corresponding confirmation is received by a selected taxi.
- **Average Blocking Time:** It is the average amount of time during which *Unoccupied* taxis were in *blocked* state while executing the TSA.
- **Number of Messages (NM):** It is the total number of messages exchanged between taxis and a passenger during the execution of the TSA.
- **Selection Accuracy Percentage (SAP):** It is the percentage of times a taxi with the shortest PRT is selected.

3) Simulation Details: We assume that the location of passenger request origin is same as the passenger's pickup location. We use Omnet++ 5.0 [21] along with INetworkNode (INet) framework [22] for the simulation purpose. To create the scenario, we select the passenger and all the taxis within 1 km range of selected passenger, then we use their geographic locations to set their starting point in the simulation. The simulation parameters are given in Table III.

TABLE III. PARAMETERS

Parameters	Values		
Transmission Range (meters)	100	150	200
Bit Rate	1 Mbps		
Number of Hops	5		
Taxi Speed	20 km/hr		
Taxi Mobility	Random		

One scenario performs the selection process for one passenger request. So, our selection time is same as execution time. We use Google's Distance Matrix API to calculate the time required by the taxis to reach the passenger's location and the actual road distance. We calculate the results using three different transmission ranges. For the selection accuracy, we compare, *TSA*, EZCab [2] and *G-TSA* (similar to [3]).

4) Results: The performance of *G-TSA* and EZCab is same for all the metrics except Blocking time and SAP. Therefore, we compare the performance of *TSA* with EZCab only for the Blocking time. For the remaining parameters, we compare the performance of *TSA* with *G-TSA*.

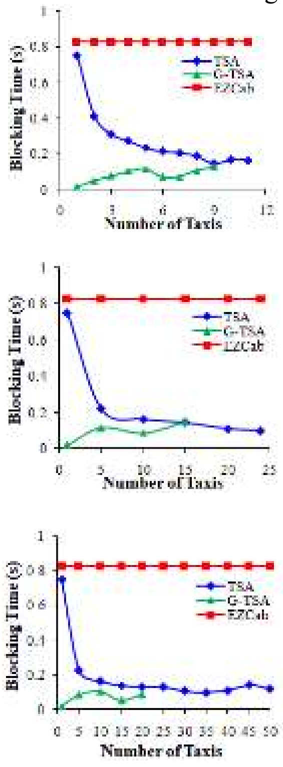


Fig. 16. Blocking Time w.r.to Number of Taxis

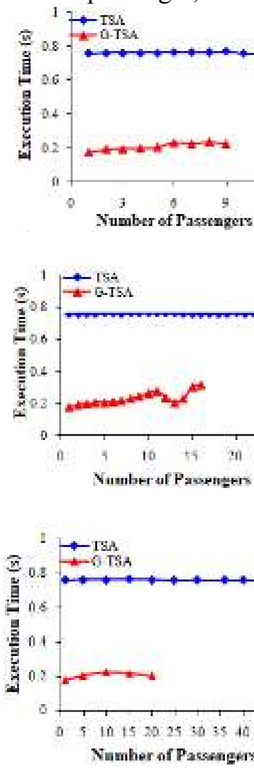


Fig. 17. Execution Time w.r.to Number of Taxis

(a) 100 meters

(b) 150 meters

(c) 200 meters

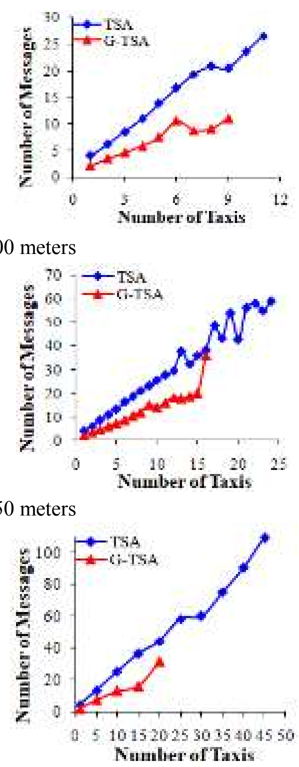


Fig. 18. NM vs Number of Taxis

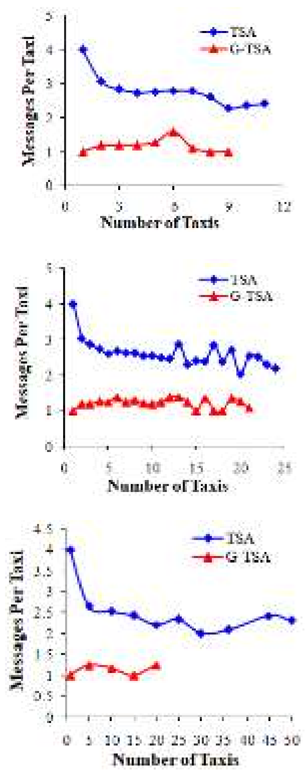


Fig.19. Average NM Per Taxi w.r.to Number of Taxis

Blocking Time: Fig. 16 shows the comparison of average blocking time w.r.to the number of available taxis. In EZCab, all the *Unoccupied* taxis who receives request are blocked for a fixed period of time. While in *TSA*, we use an ACCEPT message to unblock the other taxis if a taxi with shorter time is available. So, *TSA* and *G-TSA* perform better in reducing blocking time. We can notice that the Blocking time is high for *TSA* if the numbers of taxis are very less which reduces with the increase in number of taxis.

Execution Time: Fig. 17 (a), (b) and (c) show the execution time w.r.to the number of available taxis with variable transmission ranges. The execution time of the *TSA* is more than *G-TSA* as it waits for the REP of all the taxis and selects the one with shortest time. *G-TSA*, on the other hand, selects the first free taxi without waiting for any other replies.

Number of Messages: Fig. 18 and 19 depicts the total messages sent and average number of messages sent per taxi, respectively. In *G-TSA*, an *Unoccupied* taxi do not forward the REQ and a taxi who received REP message do not send the ACCEPT message to any of its children, hence the messages sent are less in *G-TSA* as compared to *TSA*.

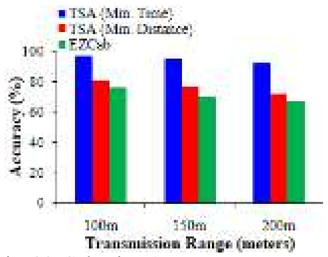


Fig. 20. Selection Accuracy Percentage

Selection Accuracy Percentage: Fig. 20 shows the selection accuracy of all three algorithms with varied transmission ranges. *TSA* selects the taxi with shortest time more than 90% times. It sometimes fails due to message loss and transmission delay.

B. Performance Analysis of HRA

In this section, we describe the datasets used for *HRA* followed by our clustering methods and results.

1) Dataset: We use two different datasets to check the working of *HRA* Model. First is San Francisco dataset (Ref. Section *V(A)*), where we use 9 days' data to train the model and the next 13 days' data to test it. We further used the NYC dataset, which consists of 12 months' of trip records in the year 2015. Another dataset used is *NYC taxi dataset* (yellow taxis) [9] as described in *Section IV(A)*. In both the datasets, the testing data is divided based on pickup day - *Friday*, *Saturday*, *Sunday* and *Weekdays* (See Section *IV(B)*). Then each dataset is further divided according to hour of a trip, and such 24 sets are made – one for each hour.

2) Performance Metrics: To measure the working of *HRA*, we use the following performance metrics.

- **Taxi Availability Ratio (TAR):** TAR represents the number of times the passengers get a taxi when requested. This is calculated as the ratio of number of taxis available to the number of passenger requests made across all the time slots, for all clusters in that time slot and all days in testing data. Eq. 9 shows the calculation of TAR. High ratio represents the less waiting time of passengers.

$$TAR = \frac{\sum_t \sum_c \sum_d \min(\,Taxis_c^{d,t}, Passengers_c^{d,t})}{\sum_t \sum_c \sum_d Passengers_c^{d,t}} \quad \ldots (9)$$

  where t is a time slot, c is a cluster ($c \in$ Cluster list) formed in time slot t, d is a day in testing data, $Taxis_c^{d,t}$ is the number of available taxis in a given cluster, date and time slot, and $Passengers_c^{d,t}$ is the number of passenger requests made in a cluster given cluster, date and time slot.

- **Hit Ratio:** Hit ratio represents the number of taxis who received a passenger during a time slot. It is given by the number of passenger requests made by the number of available taxis in all the time slots, for all clusters in that time slot and all days in testing data. Eq. 10 shows the calculation of Hit Ratio. High ratio represents a good recommendation to taxis, and reduced waiting and cruising time of taxis.

$$Hit\ Ratio = \frac{\sum_t \sum_c \sum_d \min(Taxis_c^{d,t},\ Passengers_c^{d,t})}{\sum_t \sum_c \sum_d Taxis_c^{d,t}} \quad (10)$$

  where t is a time slot, c is a cluster ($c \in$ Cluster list) formed in time slot t, d is a day in testing data, $Taxis_c^{d,t}$ is the number of available taxis at given cluster, date and time slot, and $Passengers_c^{d,t}$ is the number of passenger requests made at a given cluster, date and time slot.

3) Results: In this section, we show the results obtained for San Francisco dataset. Due to space constraints we omit the results of using NYC dataset which is also very similar.

San Francisco Dataset: To evaluate the increase in selection rate after recommendation, we select top 20 clusters on Friday, Saturday, Sunday and Weekdays, and calculated TAR for each day. TAR will show the number of requests fulfilled when it was made resulting in a decrease in waiting time of passengers. Table IV shows the TAR obtained for individual day category before the recommendation and after recommendation. The result shows an overall increase of 8.725% after recommendation.

TABLE IV. TAR FOR SAN FRANCISCO DATASET

Days	TAR (Before recommendation)	TAR (After recommendation)
Friday	0.725078819146	0.854437788336
Sat	0.730122746876	0.89643551523
Sun	0.772526501767	0.810496236409
Weekdays	0.818628175171	0.887917389010
Overall	0.7860343954961428	0.8732912994307143

TABLE V. HIT RATIO FOR SAN FRANCISCO DATASET

Algorithm/ Hit ratio	Set 1 (α = 0.34, β = 0.33, μ = 0.33)	Set 2 (α = 0.5, β = 0.3, μ = 0.2)	Model created using K-means [13]
Weekdays	0.607778336286	0.435972205582	0.4483678912
Friday	0.586644832529	0.417380929407	0.3982883721
Saturday	0.575202722322	0.400828539605	0.4103714374
Sunday	0.604897785342	0.428763305370	0.4072697065
Overall	0.599254108663	0.426831453986	0.4299144401

We calculate hit ratio for all the clusters formed after clustering. The weightage given to *CS* (α), *GS* (β) and *DS* (μ) are 0.34, 0.33 and 0.33, respectively. Table V shows the Hit ratio achieved after recommendation for all the day categories. The result shows an overall Hit ratio (for all the days) of 0.449. Then we compare Hit ratio of San Francisco using two different value sets for score weights and compared their result with the model built using K-means and obtained a 17% improvement.

NYC Dataset: We calculate Hit ratio of NYC dataset using *HRA* model and compared it with another model which is created by applying K-means [13] on the whole dataset without making any data division which we did in our approach. We calculate Hit ratio for two different sets of values for the weightage given to *CS* (α), *GS* (β) and *DS* (μ). In set 1, equal weightage of 0.33 is given to all, while in set 2, the values of $\alpha, \beta,$ μ are taken as 0.5, 0.3 and 0.2, respectively. Table VI shows the hit ratio of HRA for both sets and the model created using K-means algorithm. It is found that set 1 gives better results than set 2 and model using the model generated using K-means algorithm. The result of set 2 is almost same as the results obtained from K-means algorithm, which shows the selection of weightage parameters is important.

TABLE VI. HIT RATIO FOR NYC DATASET

Days	Hit Ratio
Friday	0.424431968047
Saturday	0.450935241604
Sunday	0.463739376771
Weekdays	0.452065704331
Overall	0.449082502799

VI. CONCLUSION AND FUTURE WORKS

In this paper, we focused on better coordination between passengers and taxi cabs, as public transport facilities, which can reduce passengers' waiting time while improving taxi drivers' profit by increasing taxi occupancy rate. First, we proposed a Taxi Selection algorithm which aims to find a taxi that can reach the passenger's location in least possible time. We also proposed a Hotspot Recommendation Model which suggests taxi drives (passengers) nearby locations where the chances of finding passengers (taxis) are high. We have carried out extensive empirical evaluation of our proposed schemes using two large scale taxi datasets of San Francisco and New York and have established the strength of our schemes. In future, we plan to introduce a cognitive model which updates the solution using current information in an adaptive manner and based on current passenger movements. This will be useful for locations which have recently become popular because of the opening of a new landmark and was not included in the old model, and it also eliminates the locations where from the passenger requests have reduced.

VII. ACKNOWLEDGMENT

Research work reported in this paper is partially funded by the Alexander von Humboldt Foundation through the post-doctoral Fellow Dr. Vaskar Raychoudhury. Also, we acknowledge the critical comments and advices provided by Dr. Rajdeep Niyogi, Associate Professor, Department of CSE, IIT Roorkee, India, during the initial stages of this work. We also thank Ms Shrawani Silwal from Miami Univ., Ohio, USA for her help to improve the paper.

REFERENCES

[1] M. Batty, K. Axhausen, F. Giannotti, A. Pozdnoukhov, A. Bazzani, M. Wachowicz, G. Ouzounis, and Y. Portugali, "Smart cities of the future," *The European Physical Journal Special Topics*, vol. 214, no. 1, 2012, pp. 481-518.

[2] P. Zhou, T. Nadeem, P. Kang, C. Borcea and L. Iftode, "EZCab: A Cab Booking Application Using Short-Range Wireless Communication," *In proceedings of theThird IEEE International Conference on Pervasive Computing and Communications*, 2005, pp. 27-38.

[3] J. P. Sheu, G. Y. Chang, and C. H. Chen, "A Distributed Taxi Hailing Protocol in Vehicular Ad-Hoc Networks," *In proceedings of the 2010 IEEE 71st Vehicular Technology Conference*, 2010, pp. 1–5.

[4] P. M. d'Orey, "Empirical evaluation of a dynamic and distributed taxi sharing system," *In proceedings of the 15th IEEE Conference on Intelligent Transportation Systems*, 2012, pp. 140–146.

[5] D. Lee, H. Wang, R. Cheu, and S. Teo, "Taxi Dispatch System Based on Current Demands and Real-Time Traffic Conditions," *Transportation Research Record: J. Transportation Research Board*, vol. 1882, no. 1, 2004, pp. 193-200.

[6] New York City Report. http://www.techinsider.io/why-new-york-city-looks-like-it-does-2015-9 [accessed on April,2016]

[7] Mobility traces of taxi cabs in San Francisco, USA. http://crawdad.org/epfl/mobility/20090224/cab/[accessed on March, 2017]

[8] New York City Taxi and Limousine Commission. Taxi of Tomorrow Survey Results, Feb 2011. http://www.nyc.gov/ [accessed on April, 2016]

[9] NYC Taxi and Limousine Commission (TLC) Trip Record Data.http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml [accessed on April, 2017]

[10] K. Yamamoto, K. Uesugi, and T. Watanabe, "Adaptive Routing of Cruising Taxis by Mutual Exchange of Pathways," *In proceedings of the 12th International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, 2010, pp. 559-566.

[11] K. T. Voon and K. C. Yow, "GSM Positioning- Based Taxi Booking and Dispatch System," *In proceedings of the 10th WSEAS International Conference on Applied Informatics and Communications, and 3rd International Conference Biomedical Electronics and Biomedical Informatics*, 2010, pp. 25-30.

[12] S. Phithakkitnukoon, M. Veloso, C. Bento, A. Biderman, and C. Ratti, "Taxi-Aware Map: Identifying and Predicting Vacant Taxis in the City," *In proceedings of the First International Joint Conf. Ambient Intelligence (AMI)*, 2010, pp. 86-95.

[13] J. Lee, I. Shin, and G. Park, "Analysis of the passenger pick-up pattern for taxi location recommendation," *In proceedings of the 4th International Conference* on *Networked Computing and Advanced Information Management*, 2008, pp. 199–204.

[14] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun, "Where to Find My Next Passenger?" *In proceedings of the 13th International Conference on Ubiquitous Computing*, 2011, pp. 109-118.

[15] J. Powell, Y. Huang, F. Bastani and M. Ji, "Towards reducing taxicab cruising time using spatio-temporal profitability maps," *In proceedings of the 12th International Conference on Advances in Spatial and Temporal Databases*, 2011, pp. 242–260.

[16] D. Zhang, L. Sun, B. Li, C. Chen, G. Pan, S. Li, and Z. Wu, "Understanding taxi service strategies from taxi GPS traces," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, 2015, pp. 123–135.

[17] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. Pazzani. "An energy-efficient mobile recommender system," *In proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010, pp. 899–908.

[18] L. Moreira-Matias, R. Fernandes, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "An online recommendation system for the taxi stand choice problem (Poster*)," In proceedings of the IEEE Vehicular Networking Conference*, 2012, pp. 173–180

[19] H. S. Kim, S. Gao, Y. Xia, G. B. Kim and H. Y. Bae "DGCL: An Efficient Density and Grid Based Clustering Algorithm for Large Spatial Database*", In proceedings of the 7th International Conference, WAIM*, 2006, pp. 362-371.

[20] J. Han and M. Kamber, "Cluster Analysis: Basic Concepts and Methods", *in Data Mining: Concepts and Techniques*, 3rd Edition, India: Academic Press, 2011.

[21] OMNeT++ Home Page. https://omnetpp.org/ [accessed on April, 2017]

[22] OMNeT++ INet Framework Home Page. https://inet.omnetpp.org/ [accessed on January, 2017]

[23] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Large Clusters in Large Spatial Databases with Noise," *In proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226-231.