Volume 4, Issue 7, July 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Enhanced Driving Based on Local Smoothing Algorithm

D Shobha Rani[1]
Assistant Professor, Dept of CSE,
S V Engineering College for Women,
(Affiliated to JNTU Anantapur),
Tirupati, A.P, India

M Thulasi[2]
Student (M. Tech-CSE),
S V Engineering College for Women,
(Affiliated to JNTU Anantapur)
Tirupati, A.P, India

Abstract-GPS-equipped taxis can be regarded as mobile sensors probing traffic flows on road surfaces, and taxi drivers are usually experienced in finding the fastest (quickest) route to a destination based on their knowledge. In this paper, we mine smart driving directions from the historical GPS trajectories of a large number of taxis, and provide a user with the practically fastest route to a given destination at a given departure time. In our approach, we propose a time-dependent
landmark graph, where a node (landmark) is a road segment frequently traversed by taxis, to model the intelligence of taxi drivers and the properties of dynamic road networks. Then, a Variance-Entropy-Based Clustering approach is devised to estimate the distribution of travel time between two landmarks in different time slots. Based on this graph, we design a two-stage routing algorithm to compute the practically fastest route. We build our system based on a real-world trajectory dataset generated by over 33,000 taxis in a period of 3 months, and evaluate the system by conducting both synthetic experiments and in the field evaluations.

Keywords-Driving directions, time-dependent fast route, taxi trajectories, Local Smoothing Algorithm.

I. INTRODUCTION

Essentially, the time that a driver traverses a route depends on the following three aspects: 1) the physical feature of a route, such as distance, capacity (lanes), and the number of traffic lights as well as direction turns; 2) the time-dependent traffic flow on the route; and 3) a user's driving behavior. Given the same route, cautious drivers will likely drive relatively slower than those preferring driving very fast and aggressively. Also, users' driving behaviors usually vary in their progressing driving experiences.

For example, traveling on an unfamiliar route, a user has to pay attention to the road signs, hence drive relatively slowly. Thus, a good routing service should consider these
three aspects (routes, traffic, and drivers), which are far beyond the scope of the shortest/fastest path computing.

Usually, big cities have a large number of taxicabs traversing in urban areas. For efficient taxi dispatching and monitoring, taxis are usually equipped with a GPS sensor, which enables them to report their locations to a server at regular intervals, e.g., 2-3 minutes. That is, a lot of GPS equipped taxis already exist in major cities, generating a huge number of GPS trajectories every day [2]. Intuitively, taxi drivers are experienced drivers who can usually find out the fastest route to send passengers to a destination based on
their knowledge (we believe most taxi drivers are honest although a few of them might give passengers a roundabout trip. When selecting driving directions, besides the distance
of a route, they also consider other factors, such as the time variant traffic flows on road surfaces, traffic signals and direction changes contained in a route. These factors can be Learned by experienced drivers but are too subtle and difficult to incorporate into existing routing engines. Therefore, these historical taxi trajectories, which imply the intelligence of experienced drivers, provide us with a valuable resource to learn practically fast driving directions.

In this paper, we propose a cloud-based cyber-physical system for computing practically fast routes for a particular user, using a large number of GPS-equipped taxis and the user's GPS-enabled phone. As shown in Fig. 1, first, GPS equipped taxis are used as mobile sensors probing the traffic rhythm of a city in the physical world. Second, a cloud in the cyber world is built to aggregate and mine the information from these taxis as well as other sources from Internet, like web maps and weather forecast. The mined knowledge includes the intelligence of taxi drivers in choosing driving directions and traffic patterns on road surfaces. Third, the knowledge in the cloud is used in turn to serve Internet users and ordinary drivers in the physical world. Finally, a mobile client, typically running in a user's GPS-phone, accepts a user's query, communicates with the
cloud, and presents the result to the user. The mobile client gradually learns a user's driving behavior from the user's driving routes (recorded in GPS logs), and supports the cloud to customize a practically fastest route for the user.

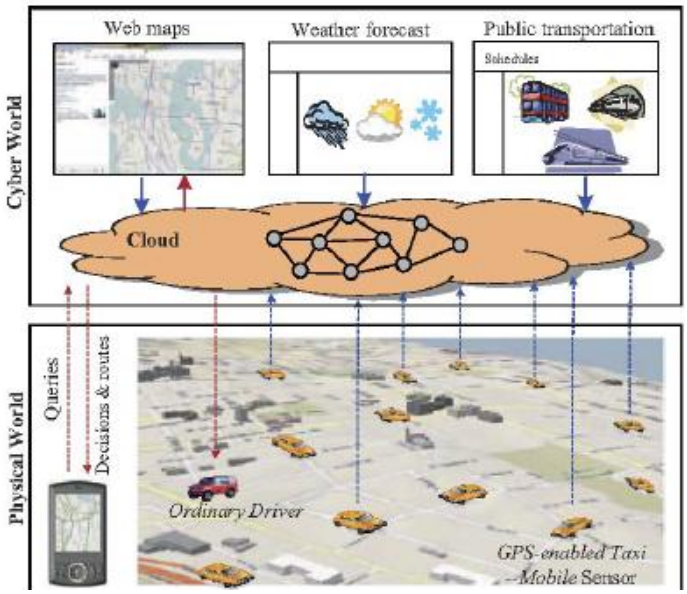


Fig.1. A cloud-based driving directions service

. However, we need to face the following three challenges:
1) Intelligence modeling. As a user can select any place as a source or destination, there would be no taxi trajectory exactly passing the query points. That is, we cannot answer
user queries by directly mining trajectory patterns from the data. Therefore, how to model taxi drivers' intelligence that can answer a variety of queries is a challenge;

2) Data Sparseness and coverage. We cannot guarantee there are sufficient taxis traversing on each road segment even if we have a large number of taxis. That is, we cannot accurately estimate the speed pattern of each road segment; and
3) Low sampling rate problem. To save energy and communication loads, taxis usually report on their locations in a very low frequency, like 2-5 minutes per point. This increases the uncertainty of the routes traversed by a taxi [3]. As shown in Fig. 2, there could exist four possible routes (R1-R4) traversing the sampling points a and b. Since this paper is an extension of our previous publication [1], we summarize the contributions (including that of the previous paper) of our work as follows:

1. In the previous paper [1], we propose the notion of a time-dependent landmark graph, which well models the intelligence of taxi drivers based on the taxi trajectories. We devise a Variance-Entropy-Based Clustering (VE-Clustering for short) method to learn the time-variant distributions of the travel times between any two landmarks.
2. In this extension work:

a. We further improve our routing service by self adaptively learning the driving behaviors of both the taxi drivers and the end users so as to provide personalized routes to the users.
b. We present smoothing algorithms for removing the roundabout part of the original rough routes.
c. We build the improved system by using a real world trajectory data set generated by 33; 000þ taxis in a period of three months, and evaluate the system by conducting both synthetic experiments and in-the-field evaluations (performed by real drivers). The results show that proposed method can effectively and efficiently find out practically
better routes than the competing methods.

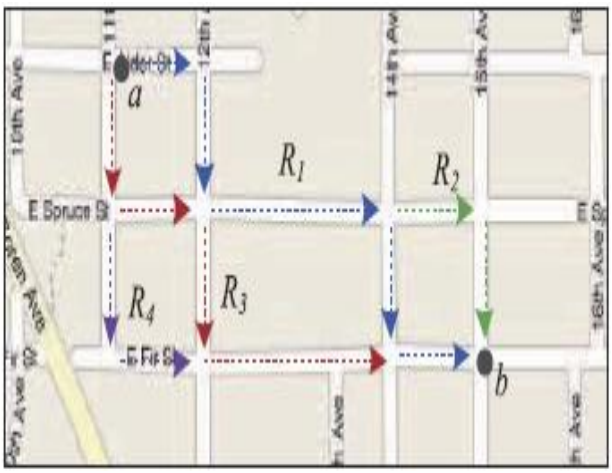


Fig. 2. Low-sampling-rate problem.

II. LOCALSMOOTHING ALGORITHM

Local smoothing. This step aims to find the longest subsequence from the resulting sequence of the global smoothing so as to satisfy the next-nearest principle. It's clear that the brute-force algorithm which checks all the subsequences (whether satisfy Principle 3) takes exponential time. We propose an polynomial time algorithm as shown in Algorithm 1.

Input: a sequence $L = l_1 \rightarrow l_2 \rightarrow l_3 \rightarrow \ldots \rightarrow l_{n-1} \rightarrow l_n$,
$\text{dist}(l_i,l_j)$, $i, j = 1, 2, \ldots, n$
Output: a subsequence (of L)
$L' = l'_1 \rightarrow l'_2 \rightarrow l'_3 \rightarrow \ldots \rightarrow l'_{m-1} \rightarrow l'_m$ that satisfies:
$\forall i = 1, 2, \ldots m-1, \text{dist}(l_i, l_{i+1}) = \min_{j>i}\{\text{dist}(l_i, l_j)\}$

```
1  for i ← 2 to n do
2      for j ← i − 1 downto 1 do
3          if SL(j) == ∅ then
4              Insert the sequence l_j → l_i to SL(j)
5          else
6              Binary search in SL(j) for the largest integer p such
               that dist(l_j,l_{j_p})≤dist(l_j,l_i), if no such value exists,
               p := 0;
               /* where l_j → l_{j_p} → ··· is the p-th
                  sequence in SL(j)                              */
7              Insert l_j → l_i after the p-th sequence of SL(j);
               /* if p == 0, insert l_j → l_i as the first
                  element of SL(j)                               */
8              for w ← 1 to p do
9                  if SL(j)^(w) ⊖^l l_j ⊕^r l_i ∈ SL(j_w) then
                       /* SL(j)^(w) == l_j → l_{j_w} → ··· is the
                          w-th sequence of SL(j)
                          SL(j)^(w) ⊖^l l_j ⊕^r l_i represents that
                          the SL(j)^(w) removes the first
                          landmark l_j from the
                          beginning(left) and adds l_i to
                          the end(right), i.e.,
                          SL(j)^(w) ⊖^l l_j ⊕^r l_i == l_{j_w} → ··· → l_i
                          */
10                     SL(j)^(w) := SL(j)^(w) ⊕^r l_i;
                       /* add l_i after the sequence
                          SL(j)^(w), i.e.,
                          SL(j)^(w) := l_j → l_{j_w} → ··· → l_i   */
11     return The longest sequence L' in {SL(i)|i = 1, 2, ..., n}
```

Fig3:Algorithm1-Local Smoothing Algorithm

Learning Custom Factor

This section describes the process for learning the user's custom factor and providing self-adapted fastest route, which contains five steps as illustrated in Fig. 4:

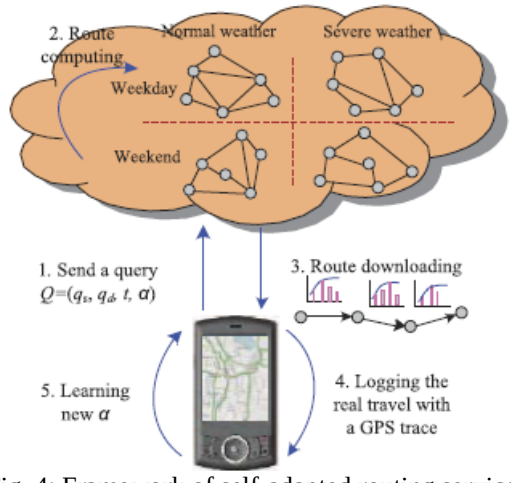


Fig. 4: Framework of self-adapted routing service.

1. Query sending. First, the user sends her query tuple to the cloud, where qs and qd are start point and destination and td is the departure time. The parameter, is the custom factor

2. Route computing. According to the departure time, start and destination point, the cloud chooses a proper landmark graph considering the weather information and whether it's a holiday or a workday. Based on the landmark graph, a two-stage routing algorithm is performed to obtain a time-dependent fastest route based.

3. Route downloading and

4. Path logging. The cloud sends the computed driving routes along with the travel time distributions of the landmark edges contained in the driving route to the phone. Later, the mobile phone logs the user's driving path with a GPS trajectory, which will be used for
Recalculate the user's custom factor. The more a driver uses this system, the deeper this system understands the driver; hence, a better driving direction services can be provided.

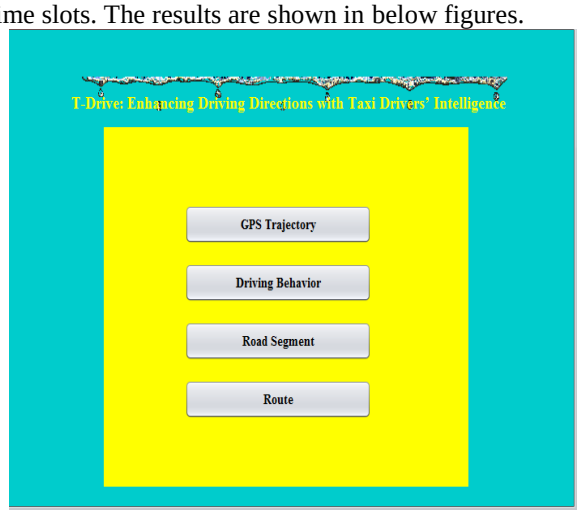
5. Adapting the custom factor. The custom factor of a given user can be learned in an self-adaptive way.

III. EVALUATION ON ROUTING

For evaluating the effectiveness of the routes suggested by different methods (say methods A and B), we use the following two criteria: Fast Rate 1 (FR1) and Fast Rate 2 (FR2) where method B is used as a baseline.

$$\begin{cases} \text{FR1} = \dfrac{\text{Number(A's travel time} < \text{B's travel time)}}{\text{Number(queries)}} \\ \text{FR2} = \dfrac{\text{B's travel time} - \text{A's travel time}}{\text{B's travel time}}. \end{cases}$$

FR1 represents how many routes suggested by method A are faster than that of baseline method B, and FR2 reflects to what extent the routes suggested by A are faster than the baseline's. Meanwhile, we use SR to represent the ratio of method A's routes being equivalent to the baseline's.
We generate 1,200 queries with different geo-distances of origin-destination pairs and departure times. The geo distances range from 3 to 23 km and follow a uniform distribution. The departure times range from 6 am to 10 pm and are generated randomly in different time slots. The results are shown in below figures.



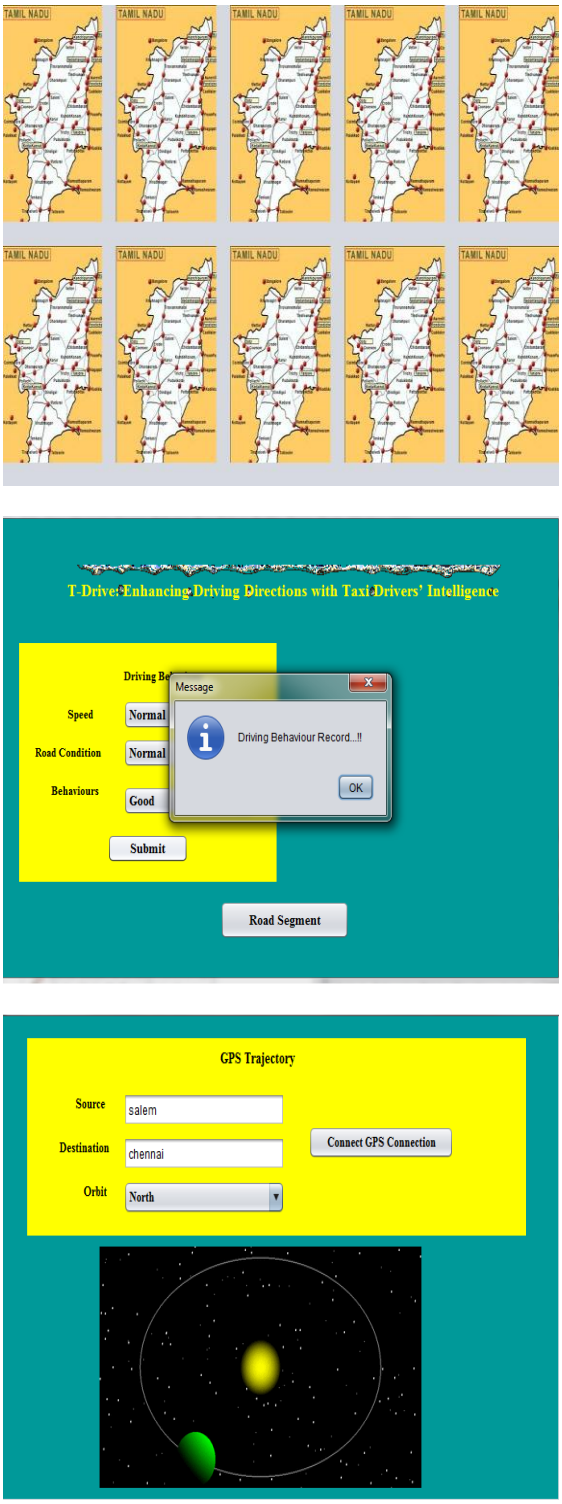


Fig:5 Verified Results

IV. CONCLUSION

This paper presents an approach that finds out the practically fastest route to a destination at a given departure time in terms of taxi drivers' intelligence learned from a large number of historical taxi trajectories. In our method, we first construct a time-dependent landmark graph, and then perform a two-stage routing algorithm based on this graph to find the fastest route. We build a real system with real world GPS trajectories generated by over 33,000 taxis in a period of 3 months, and evaluate the system with extensive experiments and in-the-field evaluations. The results show that our method significantly outperforms both the speed- constraint-based and the real-time-traffic-based method in the aspects of effectiveness and efficiency.

REFERENCES

[1] R. Chhikara and L. Folks. The inverse Gaussian distribution: theory, methodology, and applications. 1989.
[2] K. Cooke and E. Halsey. The shortest route through a network with time-dependent internodal transit times. J.Math. Anal. Appl, 14(492-498):78.
[3] B. C. Dean. Continuous-time dynamic shortest path algorithms. Master's thesis, Massachusetts Institute of Technology, 1999.
[4] B. Ding, J. Yu, and L. Qin. Finding time-dependent shortest paths over large graphs. In Proc. EDBT, pages 205{216. ACM, 2008.
[5] S. Dreyfus. An appraisal of some shortest-path algorithms. Operations Research, 17(3).
[6] Fayyad and Irani. Multi-interval discretization of continuous-valued attributes for classification learning. Proc. IJCAI, pages 1022{1027, 1993.
[7] H. Gonzalez, J. Han, X. Li, M. Myslinska, and J. Sondag. Adaptive fastest path computation on a road network: A traffic mining approach. In Proc. VLDB, 2007.
[8] A. G□uhnemann, R. Sch□afer, K. Thiessenhusen, and P. Wagner. Monitoring traffic and emissions by outing car data. Institute of transport studies Australia, 2004.
[9] E. Kanoulas, Y. Du, T. Xia, and D. Zhang. Finding fastest paths on a road network with speed patterns. In Proc. ICDE, 2006.
[10] J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. LNCS, 4206:243{260.
[11] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang. Map-matching for low-sampling-rate GPS trajectories. In Proc. GIS. ACM, 2009.
[12] A. Orda and R. Rom. Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. JACM, 37(3):625, 1990.
[13] D. Pfoser, S. Brakatsoulas, P. Brosch, M. Umlauft, N. Tryfona, and G. Tsironis. Dynamic travel time provision for road networks. In Proc. GIS. ACM, 2008.
[14] J. Yuan, Y. Zheng, C. Zhang, and X. Xie. An interactive-voting based map matching algorithm. In Proc MDM, 2010.
[15] Y. Zheng, Y. Chen, Q. Li, X. Xie, and W. Ma. Understanding transportation modes based on GPS data for Web applications. ACM Transactions on the Web, 4(1):1{36, 2010.
[16] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W. Ma. Understanding mobility based on GPS data. In Proc. Ubicomp, pages 312{321, 2008.
[17] Y. Zheng, L. Liu, L. Wang, and X. Xie. Learning transportation mode from raw gps data for geographic applications on the web. In Proc.

