Smarter Outlier Detection and Deeper Understanding of Large-Scale Taxi Trip Records: A Case Study of NYC

Jianting Zhang
Department of Computer Science
The City College of the City University of New York
New York, NY, 10031
jzhang@cs.ccny.cuny.edu

ABSTRACT

Outlier detection in large-scale taxi trip records has imposed significant technical challenges due to huge data volumes and complex semantics. In this paper, we report our preliminary work on detecting outliers from 166 millions taxi trips in the New York City (NYC) in 2009 through efficient spatial analysis and network analysis using a NAVTEQ street network with half a million edges. As a byproduct of large-scale shortest path computation in outlier detection, betweenness centralities of street network edges are computed and mapped. The techniques can be used to help better understand the connection strengths among different parts of NYC using the large-scale taxi trip records.

Categories and Subject Descriptors
H.2.8 [Database Management]: Database applications -Spatial databases and GIS

Keywords
Outlier Detection, Shortest Path, High-Performance, Taxi Trip

1. INTRODUCTION

Probe vehicle data, such as taxi trip records with GPS recorded pickup-up and drop-off locations/times and other related trip data, are valuable in understanding urban traffic and travel patterns. However, such data are often prone to various human, device and information system induced errors. Detecting outliers and subsequently cleaning up the data is a prerequisite for reliable analysis. Existing outlier detection and data clean approaches often use simple rules such as trip locations should be within city boundaries and geometric distances should be within certain thresholds [1]. When intermediate GPS readings are available, it is possible to examine the GPS traces and understand the identified outliers. However, in many cases, due to privacy concerns or data volume concerns or some other policy/management related reasons, the complete GPS traces are not available. Compared with using geometrical distances, shortest path distances can be more accurate and reliable in detecting outliers. However, shortest path computation in large street networks using traditional algorithms can incur unacceptable runtimes and make the computation impractical.

In this study, by integrating an open source package called MoNav [2] that implements an efficient Contraction Hierarchy (CH) based shortest path computation algorithm [3] and a spatial join algorithm to snap pickup and drop-off locations, we are able to compute the shortest paths of the 166 million taxi trip records in NYC in 2009 in less than two hours on a commodity desktop computer. The computed shortest paths not only facilitate outlier detection in a smarter way, but also allow generate edge betweenness centrality maps to understand the connection strengths among different areas of NYC from the taxi trip records. To the best of our knowledge, the approach has not been exploited previously in urban computing research. We believe the new approach can be used to understand the interactions between people and places deeper.

The rest of this paper is arranged as the following. Section 2 briefly introduces related works on outlier detections of taxi trip records, efficient shortest computation on road networks and betweenness centrality computation and applications. Section 3 presents the proposed method, including technique details on snapping taxi pickup and drop-off locations to street network and deriving betweenness centrality maps. Section 4 provides experiment results and performance evaluations. Finally Section 5 is the conclusions and future work directions.

2. BACKGROUND AND RELATED WORKS

The incompleteness, uncertainties and errors associated with taxi trip records have been well recognized in urban computing case studies. A commonly used practice to detect outliers and clean up data is to discard trips that are either too long or too short. For example, the research reported in [1] discarded trips with distances that are greater than 30 km (from one side of the city to the other) or shorter than 200 meters (which are considered unrealistic). A more general approach is to compute the distribution of a measurement (location, distance, duration et al) and then consider those that fall within the unusual ranges as outliers [4]. More domain-specific outlier detection approaches include marking taxi pickup and drop-off locations that are outside certain administrative regions or within regions of certain land use types (e.g. river/lake) as outliers through geospatial analysis such as point-in-polygon testing or nearest neighbor computing.

There are quite a few pioneering works in outlier detections from GPS traces. Some of them snap GPS trajectories to street networks and integrate street network topology into outlier detections [5] while others discretize GPS traces into grid cells and then use grid cells as the basic units for outlier detection [6, 7]. Unfortunately, these techniques can not be applied to our application as complete GPS traces are not available in our dataset and we are limited to use the two ends (Origin/Destination) of GPS trajectories.

In the context of processing GPS data, quite a few techniques have been developed to tackle uncertainty issues related to low GPS sampling rates by using road network
geography and topology [8, 9]. These techniques can be used to detect outliers as well by considering the non-matched GPS locations as outliers. In our study, in addition to applying the threshold-based, distribution-based and spatial analysis based outlier detection techniques, we propose to compute shortest paths and compare the recorded distances with computed shortest distances as a way to detect outliers. This approach can be considered as a network analysis based outlier detection technique. As detailed in the experimental section, in addition to being able to detect device-induced outliers, the approach is able to detect human induced outliers.

Shortest path computation is fundamental in both computer science research and virtually all application areas. A variety of shortest path algorithms, including the classic Dijkstra’s algorithm and the A* algorithms, are available [10]. Several recent shortest path algorithms that are designed specifically for road networks have achieved significant higher efficiencies than the generic ones. Among them, the Contraction Hierarchies (CH) algorithm developed by the Algorithm Engineering group at the KIT of Germany [3] has gained considerable popularities in practical applications. Two open source packages based on the CH algorithm, namely MoNav [3] and OSRM [11], are currently available. We found that the CH implementation in MoNav is easy to understand and easy to integrate in our application. As such, we have extracted the CH implementation module (termed as MoNav-CH hereafter) and used it in our study. As shown in the experiment section, we are able to compute over 25 million unique shortest paths in about 5,952 seconds using a single CPU core. The performance is more than three orders better than a commercial GIS software that we have tested. We also would like to note that a recent Graphics Processing Unit (GPU) based implementation of the CH algorithm called GPHAST [12] has achieved a throughput of less than 2 milliseconds in computing a single-source shortest path tree in an European road network with more than 18 million nodes and 42 million edges. The performance is about three orders faster than the Dijkstra’s algorithm (without including path output overheads). We expect to gain higher performance in shortest path computation after incorporating the GPHAST algorithm. This has been planned for our U2SOD-DB system [13] to provide a comprehensive and integrated data management and data mining platform for large-scale Origin-Destination (OD) data in ubiquitous urban sensing applications, including OD-based taxi trip records.

Node or edge based betweenness centrality has been widely used to study the relative importance of nodes and edges in a network [14]. The betweenness centrality of a node or an edge can be simply defined as the number of shortest paths that pass through the node or edge. As a by-product of computing shortest paths of all taxi trips, the betweenness centrality for all nodes and edges in a road network can be easily computed through simple aggregations on the edges or the nodes. Different from social network applications of betweenness centrality, in this study, we spatially map the numbers of computed shortest paths on all edges for deeper understanding of the connection strengths among different parts of the NYC in a visual manner. While our primary focus on this study has been on computing efficiency and data management sides, we note that betweenness centrality has been recently applied to study traffic flows [15, 16]. Compared with plotting aggregated pickup and drop-off locations separately [1] or examining an OD matrix [17], our approach provides details at the street segment level and relates origin and destination locations simultaneously. The standard map interfaces allows zooming, panning and querying the distributions of the betweenness centrality dynamically which are familiar to many users who use Internet mapping services (such as Google Map) on a daily basis.

3 METHOD AND DISCUSSIONS

Given a taxi trip dataset (T) with the following attributes: pickup location, pickup time, drop-off location, drop-off time and recorded distance, and, a street network (S) with N nodes and M edges, our method for outlier detection has three steps. The first step snaps both the pickup and drop-off locations to their nearest street segments. If a taxi trip has either a pickup location or drop-off location that can not be snapped to a street segment with a reasonable distance threshold (D0), it is considered as a type I outlier. The node in the snapped segment that is closer to the pickup/drop-off locations is then assigned to the trip record as a pickup or drop-off node. The second step computes the unique combinations of the pickup and drop-off nodes of all trips. As demonstrated in the experiment section, computing shortest path of unique node combinations instead of individual taxi trips has reduced shortest path computation workload to nearly 1/7. The third step actually computes the shortest paths using the MoNav-CH module for each unique pickup-up and drop-off node pair. While the shortest paths are being computed, the node and edge centralities are updated for each of the nodes and edges along the computed shortest path. The computed shortest path distances are then compared with the recorded distances. If the computed distances are greater than a threshold D1 and are W times longer than the recorded distances, then the records are marked as type II outliers.

We next briefly discuss some of the implementation details on snapping pickup and drop-off locations. It is obvious that the criteria on snapping point locations to street network segments can be viewed as a combination of window query and a nearest neighbor query which are supported in many Geographical Information Systems (GIS) and Spatial Databases (SDB). However, our experiments have shown that existing GIS and SDB software tools are not designed to handle location data at the scales of hundreds of million points and above. The performance is unacceptable using either ESRI ArcGIS or open source PostgreSQL. While we are aware of proper partitions and configurations can potentially improve performance but we are not sure whether the achievable performance can meet the level that is required by our application. As such, we leverage the P2N-D spatial join module in our U2SOD-DB system for this purpose. While we refer to [13] for more details on this, we would like to note that by adopting a column-oriented, in-memory system with massively data parallel GPU hardware accelerations, the performance of the first step of the proposed method has been significantly improved.

We also note that using shortest path as a proxy of actual trip paths is approximate in nature, especially when computing shortest paths purely based on distances. As many previous studies have shown that, experienced taxi drivers usually take quite a few factors into consideration in choosing travel paths and it is not surprising that some researchers suggest use experiences of taxi drivers for online navigation purposes [18, 19]. Nevertheless, we believe that when the recorded trip distances deviate significantly from the computed shortest distances by setting D1 and W properly, it can be an effective filtering mechanism to identify true outliers. In addition, the
computed betweenness centrality measurements are aggregated values where the inaccuracies and uncertainties have a good chance to be canceled by each other and can be used as a visualization tool to understand the overall travel and traffic patterns in metropolitan areas such as NYC.

4 EXPERIMENTS AND RESULTS

4.1 Data and Experimental Setting

We use approximately 166 million taxi trips in NYC in 2009 in this study. While there are more than 20 attributes from the original database dump, we use only the five attributes as listed in previous section and we refer to [13] for more details regarding to the data management aspects of both taxi trip data and infrastructure data in U'SOD-DB. For illustration purposes, we have plotted a distribution map of aggregated numbers of taxi pickup locations using a spatial resolution of 16 feet (~5 meters) and a grid dimension of 8192*8192. As shown in the right part of Fig. 1, clearly, the majority of taxi pickup locations are in the midtown and downtown Manhattan area. When the taxi pickup locations in the area are aggregated at 0.5 feet resolution, the effect of road network becomes apparent as shown in the left part of Fig. 1. The similar trends have been observed for the drop-off locations.

The street network in the NYC area provided by NAVTEQ through its University Program not only includes the five boroughs in NYC but also includes three neighboring counties, i.e., Westchester, Nassau and Suffolk. The expanded spatial coverage allows examine taxi trips with either pickup locations or drop-off locations outside the NYC boundary. Significant preprocessing work has been carried to bridge between the map data format used by NAVTEQ and the graph data format used by MoNav. While we could have extracted both nodes and edges in the streets shapefile according to NAVSTREETS data format documentation, we have decided to incorporate intermediate nodes that are separately stored in zlevels for the purpose of snapping taxi locations to street segments more accurately. The resulting network has 434,521 nodes and 501,756 nodes. As the network is very sparse, the CH-based algorithm is especially suitable as many natural shortcuts exist.

Fig. 1 Aggregated Pickup Locations in NYC in 2009 with Details in Manhattan

Fig. 2 Histograms of Distributions of Trip Distance, Time, Speed and Fare
4.2 Distributions of Trip Distances, Time, Speed and Fare

In addition to the spatially aggregated pickup and drop-off counts at different grid scales, we have also computed the histograms of travel distances, travel times, travel speeds and travel fares from the 2009 trip records and they are shown in Fig. 2. Note that the first bins in the histograms represent invalid values and the last bin represents excessive values that are larger than the pre-defined thresholds and their counts can be larger than the neighboring bin counts. From the figure we can see that there are quite some short trips (travel distance <1.0 miles and/or travel time <5 minutes) in NYC which may warrants further studies. While the distributions of both travel distances and travel times are heavily tailed, the speed distribution exhibit a nice normal distribution centered at 10 MPH (Miles per Hour) which may reflect heavy traffic congestion in NYC. The start/end overheads of short trips may also contribute to the low average speed. Despite waiting times are counted towards fares, the distribution of fares closely resembles the distribution of travel speed which may suggest that trip distance is still the dominate factor of fares. It might be interesting to separate midtown and downtown Manhattan area with the rest of the NYC and look into the differences of their travel speed distributions. The same idea can be applied to the time dimension to compare the differences of distributions during peak/off-peak time slots, weekdays/weekends, normal/special events days and their combinations.

4.3 Results on taxi trip outlier detection

Using the method discussed in Section 3, we have set \( D_0 = 200 \) feet, \( D_1 = 3 \) miles and \( W = 2 \) for our experiments in outlier detections. Among the 166 million taxi trip records, approximately 2.5 millions (1.5%) whose pickup or drop-off locations can not be snapped to a street segment of the NAVTEQ street map dataset and are identified as Type I outliers. While the majority of these outliers could be induced by GPS device errors, some of them may be associated with incompleteness of street networks, e.g., picking up and dropping off at private land parcels, and thus requires further investigations. Fig. 3 plots all the street segments symbolized by the numbers of computed shortest paths that pass through them (c.f. Section 2 and Section 4.4) overlaid with the map of the NYC Community District data [20]. We can clearly see that quite some taxi trips cross the boundary of NYC while the majority of the trips are in the midtown and downtown Manhattan area (also see Fig. 1 and Fig. 5).

Among the 166,384,464 taxi trips whose pickup and drop-off locations are successfully snapped to NAVTEQ street network segments, about 18,000 are identified as type II outliers. For illustration purposes, the top 9 outliers in January 2009 are listed in Fig. 4 ordered by the decreasing ratio of calculated distance (calc_dist) over recorded distance (record_dist). The computed shortest paths clearly suggest that the recorded distances are incorrect. An explanation is that the metering devices, which may be separate from GPS devices, were reset during the trips for various reasons. As the number of Type II outliers is fairly small, our outlier detection technique has made it possible to examine these trip records individually in a GIS (as exemplified in Fig 4). We have tried to decrease \( D_1 \) to identify more outliers. However, for short trips, it is likely that taxi drivers may not follow shortest paths for various reasons and thus the proposed approach may increase the chances of false positives on outlier detection. We are trying to incorporate complete GPS traces and learn the differences between real trip paths and computed shortest paths.

4.4 Results on Betweenness Centrality

The 166 million taxi trips have resulted in approximately 25 million unique combinations of NAVTEQ street network nodes and it took 5,952 seconds to compute the shortest paths, including aggregating the hourly and overall centrality measurements for both nodes and edges. The result is encouraging in the sense that MoNav-CH has achieved a throughput of roughly 0.2 second per 1000 pair shortest path computation on a reasonably large street network with more than half a million edges using a single CPU core. While computing overhead has been considered as a major hurdle for urban computing in understanding human mobility [21], our results have demonstrated the feasibility of large-scale computation through efficient algorithm engineering.

The aggregated edge centrality map in NYC area using the 2009 data is shown in Fig. 5. We can see that there are strong connections between the midtown and downtown Manhattan area with the two major airports (LGA at top-right
and JFK at bottom-right) in NYC. The bridges, tunnels and highways are extensively used for the connections. We can also see that while the connections among different parts of Manhattan are even stronger, a few street segments, such as Broadway and the 5th/6th/7th avenues, have much higher centralities (and hence are more important) according to the taxi trips as shown in the left part of Fig. 5.

![Fig. 5 Mapping Betweenness Centralities (All hours)](image)

To further compare the dynamics of the street segment centralities across times, we have generated hourly centrality maps for the majority of the NYC area. The bi-hourly maps in the midtown and downtown Manhattan area are show in Fig. 6 where the widths of the street segments are symbolized using three levels of shortest path counts as indicated by the legend located at the bottom of the figure. From Fig. 6 we can see that many of the street segments in the area have more than 10^5 trips in 2009 (i.e., ~250 trips per day in an hour) for all hours. There are very few changes across hours from 08H to 22H except a slight recession at 16H (mid-afternoon). However, we do see the connection strengths decrease significantly during 00H-08H compared to the rest of the day. The connection strengths reach a minimum point around 04H where only Broadway and a few other streets still maintain the highest level of connection strength. While NYC has a well-known nickname, i.e., “The City That Never Sleeps” [22], the taxi trip records reveal that it does take a long nap in the early morning and a short nap in the afternoon.

**CONCLUSION AND FUTURE WORKS**

Large-scale taxi trip records are error-prone due to a combination of device, human and information system induced errors. In the cases that accesses to complete taxi trip trajectories are limited due to either policy, privacy or technical constraints, outlier detections based on pickup/drop-off locations and times and trip distance/duration information are especially challenging. In this study, by leveraging the U^2SOD-DB system that we have developed to facilitate efficiently managing large-scale OD data and the MoNav open source package for efficient shortest path computation, we are able to detect outliers that can not be snapped to street segments and/or have significant differences between computed shortest distances and recorded trip distances. The derived edge betweenness centralities can be used to understand connection strengths among different parts of metropolitan areas such as NYC.

For future work, first, we plan to develop a more comprehensive framework for outlier detections and data cleaning by incorporating pickup and drop-off times, trip duration and fare information. Second, we would like to generate dynamics of betweenness maps at different traffic conditions, e.g., peak/non-peak, morning/afternoon and weekdays/weekends. Finally, it is desirable to further improve the efficiencies of shortest path computation by exploiting parallel computing on distributed systems, multi-core machines and GPU devices.

**REFERENCES**

11. http://project-osrm.org/

Fig. 6 Mapping Bi-Hourly Edge Betweenness Centrality in the Midtown and Downtown Manhattan Area