

On the Effectiveness of Removing Location Information from Trajectory Data for Preserving Location Privacy

Amina Hossain¹, Anthony Quattrone², Egemen Tanin², Lars Kulik²

Department of Computing and Information Systems
The University of Melbourne

aminah@student.unimelb.edu.au¹, quattronea, etanin, lkulik@unimelb.edu.au²

ABSTRACT

The ubiquity of GPS enabled smartphones with Internet connectivity has resulted in the widespread development of location-based services (LBSs). People use these services to obtain useful advises for their daily activities. For example, a user can open a navigation app to find a route that results in the shortest driving time from the current location to a destination. Nevertheless, people have to reveal location information to the LBS providers to leverage such services. Location information is sensitive since it can reveal habits about an individual. LBS providers are aware of this and take measures to protect user privacy. One well established and simple approach is to remove GPS data from user data working with the assumption that it will lead to a high degree of privacy. In this paper, we challenge this notion of removing location information while retaining other features would lead to a high degree of location privacy. We find that it is possible to reconstruct the original routes by analyzing turn instructions which could arguably be seen as belonging to the service provider. We evaluated our approach using real road network data and demonstrate the effectiveness of this new attack in a range of realistic scenarios.

CCS Concepts

•Security and privacy → Pseudonymity, anonymity and untraceability; •Networks → Location based services;

Keywords

Trajectory privacy; Navigation privacy; Inference attack

1. INTRODUCTION

Smartphones are capable of retrieving accurate positioning estimates. Combined with the ability to connect to the Internet at high speeds, this has led to the widespread development of location-based services (LBSs). In particular,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IWCTS'16, October 31-November 03 2016, Burlingame, CA, USA

© 2016 ACM. ISBN 978-1-4503-4577-4/16/10...\$15.00

DOI: <http://dx.doi.org/10.1145/3003965.3003966>

LBSs provide important way-finding service for users to assist them in navigation while people are traveling. However, at the same time the location of a user being sent to LBSs can lead to high privacy risks.

Large quantities of location data generated every day by smartphone users during their travels. Location data is private data. Position of users show where a user has been to a high level of accuracy and can reveal sensitive insights about a user upon analysis. For example, a LSP can derive health, habits, and personnel preferences of users.

Techniques demonstrating how to preserve the trajectory privacy of LBS users has been proposed [11, 2, 16, 17, 12]. The overall concept of these existing works is to preserve users' trajectory privacy by removing location information at different levels from the trajectory data that users reveal while they are using LBSs. However, in this paper we show that even removing all the location information from the trajectory data is insufficient to protect users' privacy in a range of scenarios.

LBS data sets consist not only user trajectory information but also relevant advise given to users. This advise can include turn instructions supplied to a user for navigation purposes. We are particularly interested in the turn instructions in this paper. Turn instructions are a primary means to guide a user in order for them to reach to their destination. In [14, 3, 13], turn instructions are defined as instructions on how to follow a route by providing task-oriented specifications of the actions to be carried out to reach a destination. These instructions can easily be argued as belonging to the LSP and could be retained for a long period of time or traded.

We present a new inference attack that uses turn instructions to find the actual route of a user. The reason for utilizing the concept of turn instructions in our paper is twofold: firstly, turn instructions are commonly correlated with the road network environment. Secondly, these instructions may be exchanged by the LBS with third parties without much legal implications. In this paper, we show that by removing location information and retaining just the meta-data including turn instructions is not always privacy preserving in different realistic scenarios.

As a motivational example, consider the road network of Figure 1, a user, Bob wishes to go to a *Hospital* from his current position *S*. He may ask the LBS to provide him a route to the *Hospital* and follows straight, left and right turn directions sequentially (red arrow line). Even when removing the location tracks, just by using the city map and such turn instructions, we can reveal that Bob has started from *S* and

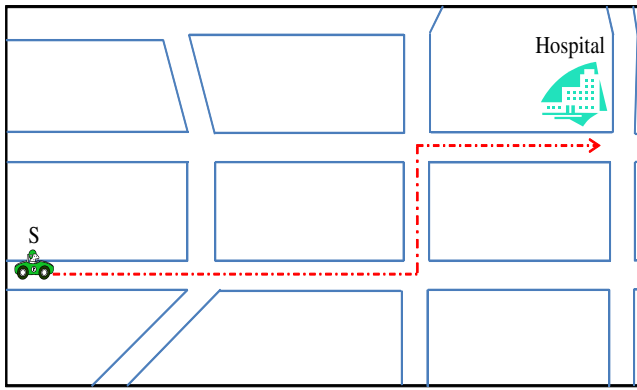


Figure 1: A scenario of trajectory privacy attack by following turn instructions given by a user

ended up at the nominated destination because this sequence is unique for this city. This simple scenario does present an intuitive example of a situation where it is feasible to get high-quality location information with no use of positional information in the first place.

In summary, the main contribution of this work are as follows:

- We show that it is possible for a trajectory data set that does not have GPS or other specific location information, to be reverse engineered using turn instructions.
- We present experimental evaluations demonstrating the feasibility of our proposed attack in realistic scenarios.

2. RELATED WORKS

Trajectory privacy is a relatively new research field extending from the area of location privacy. It has recently received considerable attention as it has been discovered that looking at a collection of location points can reveal unique and sensitive insights about people. A trajectory is a path or trace reported by a moving object in a geographical space. It is commonly represented by a set of n time-ordered points where each point consists of a geospatial coordinate pair (x_i, y_i) and timestamp (t_i) .

Extensive research [11, 1, 9, 2, 16, 17, 12] has aimed at protecting trajectory privacy. Methods employed by researchers can be generally classed under two categories: anonymizing methods and spatial cloaking. Anonymizing methods [11, 1, 9] aim to preserve trajectory privacy by ensuring users are indistinguishable from other users. Spatial cloaking methods aim to distort [16, 17, 12] user locations.

In [11], the authors employ the concept of k -anonymity to make a certain trajectory indistinguishable from $k-1$ other trajectories via a generalization-based algorithm. Privacy is ensured by releasing only a randomly generated set of representative trajectories. The authors in [1] proposed an algorithm that ensures k -anonymity from a different angle. It proposed cluster-based algorithms that utilize an uncertainty threshold, which is inherent to trajectory data to group k co-localized trajectories within the same time period to form an aggregate k -anonymized trajectory.

There is a drawback in changing the entire trajectory as a means of preserving privacy; this leads to a more-than-

necessary distorted database and considerably decreases the quality of mining opportunities. On the other hand, some new studies have focused on destination points along a path as the most sensitive parts of a trajectory and proposed solutions to protect these points.

In [9], the authors attempted to preserve sensitive destination points by generalizing them into l -diverse zones, i.e., areas that contain at least l distinct place types such as restaurant, hospital, park and etc. In [9], they treated all the stop points as equal, while in reality, some places like hospitals should be treated as more sensitive compared with other places like shopping malls.

The main idea of spatial cloaking is to preserve users' locations by blurring their locations into spatial regions that satisfy the user-specified k -anonymity level and/or minimum spatial region. Authors in [2, 16, 12, 17] propose solutions based on spatial cloaking techniques.

The basic idea of [2, 17] is that a querying user u forms a group with other $k-1$ nearby peers and blurs u 's location into a spatial area that contains all the group members as a cloaked spatial area. However, this technique may incur high computational overhead for a large area. To address this issue, [12] minimized the size of a cloaked spatial region. An alternative approach proposed in [17] hides the users' real trajectories by using historical footprints and cloaked the spatial region with other $k-1$ predicted footprints. The overarching idea of these existing works is to preserve users' trajectory privacy by suppressing trajectory information from the data that users reveal while they are using LBSs. However, the trajectory data sets contain some location information at the end.

Map matching researches have also exploited turning angles and distances to accurately predict the actual route a user has traveled along. The use of path shapes was proposed in [7] where authors create a shape represented by angles and distances and find similar shapes as derived from the road network to map match the trajectory a user traveled along. This work was extended in [6] to use only angle information to perform successful map matching. The use of turn instructions has been studied in [15] to determine the probability of a user reaching their destination. Coarse turn instructions such as (left, right, straight) can be ambiguous given the road network context. The authors suggest to use instructions such as sharp left or slightly right. In our work, we use very coarse turn instructions to demonstrate that even sharing this information can present a privacy issue as the users actual route can be identified.

3. PROPOSED ATTACK MODEL

In this section, we present an overview of our proposed method. Section 3.1 describes the preliminary steps of our proposed attack. Section 3.2 describes an algorithm for generating turn instructions. Section 3.3 discusses how to find identical sequences of turn instructions on a road network.

3.1 Preliminaries

We represent the road network as a graph structure. In this section, we describe how to represent the road network as a graph and compute the shortest path between two consecutive locations in the network.

3.1.1 Road Network

A road network R_N can be represented as an undirected

graph $G_{R_N} = (V_G, E_G)$, where V_G is a set of nodes representing intersections and E_G is a set of edges representing roads i.e., $E_G \subseteq V_G \times V_G$ consists of an edge $e_{i,j} = (n_i, n_j)$ where n_i, n_j are adjacent junctions on a road.

A function $w : E_G \rightarrow \mathbb{R}$ is used to associate a weight with each edge e . This weight can be used to model proximity between nodes, including Euclidean distance or travel time. We use $d_{G_{R_N}}(n)$ to denote the degree of a node n with respect to the graph G_{R_N} . A route r_p is a path on G_{R_N} , i.e., a sequence of nodes from V_G . The length l of a route r_p represents the total distance covered along this route. A shortest path S_p is a path that minimizes the total length that needs to be covered from a given source to a destination.

3.1.2 Shortest Path Computation

The shortest path computation is finding a path of minimum weight (distance, time) connecting two specified nodes in a graph. There has been many algorithms to compute shortest paths in networks

Dijkstra’s algorithm [4] is the most well known method to solve this problem and we adopt it for evaluating the shortest distance between two locations in our road network R_N as choice of algorithm has little effect on our attack model. Dijkstra’s algorithm takes a starting node and can find each node n in a G_{R_N} , the shortest path S_p from the start node n_s . It uses a weight function $w : E_G \rightarrow \mathbb{R}$ that determines the cost of traversing an edge e . These costs are represented as the edges’ labels. Starting from the origin n_s , at each step the edge with the lowest costs is selected, which is then marked as *visited*. The costs for reaching all unvisited edges adjacent to the current node n_c are then updated. In this manner a format for reviewing other nodes in the network is generated and maintained.

3.2 Generate Turn Instructions

Fundamentally, turn instructions are a persons actions while they travel on a road. For example, left turn, right turn, continue straight. Based on a known [14, 3, 13] definition, turn instructions refer to instructions on how to follow a route; they are task-oriented specifications of the actions to be carried out to reach a destination.

In our work, we consider three basic types of turn instructions without loss of granularity namely left (δ_l), straight (δ_s) and right (δ_r) (see Figure 3.2). The turn concept at road intersection points and conceptualize actions, i.e., the direction to take at an intersections as opposed to viewing the road network structure.

A small set of trajectory finding primitives suffices to characterize most paths, paths directions and path finding actions [10]. The authors in [10] considered seven directional primitives. But, we use three distinct primitives in this paper to make our point. Commonly, the average degree of a node ($d_{G_{R_N}}$) in a road network is less than four, thus the possible number of upcoming directions is three. For example in Figure 3, Alice wishes to go to a supermarket from her current location after reaching an intersection n_1 . n_1 consists of four routes including the route that Alice is on. Alice can reach her destination by choosing from the three possible ways i.e., left (δ_l), straight (δ_s) and right (δ_r), respectively.

We set an angle range for each direction between two roads to decide on a turn instruction for labeling purposes. We restrict each direction using two reference angles, they are α (right angle) and β (left angle) (see Table 1 and Figure 3),

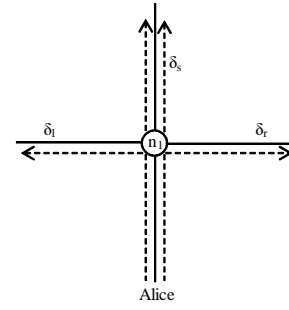


Figure 2: Three Ways of Traveling at a Decision Point

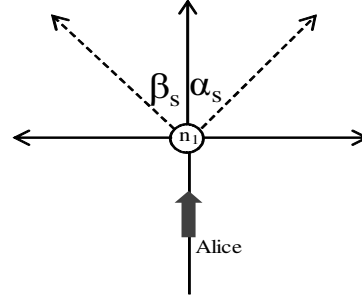


Figure 3: Going Straight

Turn Instruction(T_i)	Reference Angles
δ_l	α_l, β_l
δ_s	α_s, β_s
δ_r	α_r, β_r

Table 1: Turn instructions with their reference angles

we also set the range of the total *antropy* \in for them.

For example, straight instruction is established with reference angles α_s and β_s in Figure 3, if any line or edge lie in this range (called \in) the direction is considered to be straight.

To derive turn instructions in a road network, we propose an algorithm for generating turn instructions based on the concept above. Given a source and a destination, and the initial orientation of a driver, a sequence can be trivially generated using the shortest path algorithm and enumerating the turn instructions per junction with the above angle range concept.

Algorithm 1 describes how to generate the turn instructions using our approach.

Algorithm 1: GENERATETURNINSTRUCTIONS: Generate a sequence of turn instructions

Input: Starting point n_0 , destination point n_1 and a road network graph $G_{R_N} = (V_G, E_G)$

Output: A sequence of turn instructions TS

```

1  $TS \leftarrow \emptyset$ 
2  $r_p \leftarrow \text{shortestPath}(n_0, n_1, G_{R_N})$ 
3 for  $i \leftarrow 0$  to  $|r_p| - 1$  do
4    $n_0 \leftarrow r_i, n_1 \leftarrow r_{i+1}$ 
5    $\delta \leftarrow \text{calculateTS}(n_0, n_1)$ 
6    $TS \leftarrow TS \cup \delta$ 
7 return  $TS$ 

```

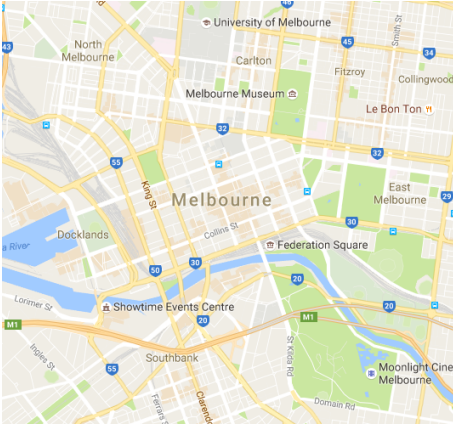


Figure 4: Melbourne Downtown Area

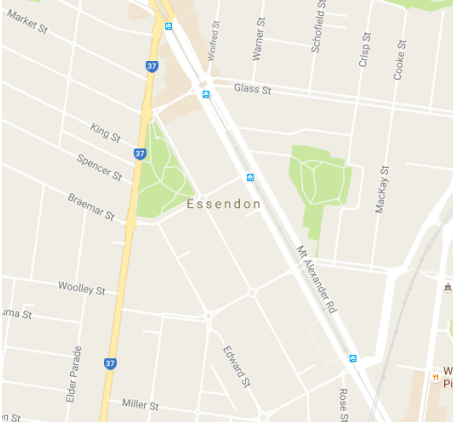


Figure 5: Essendon Inner Suburb

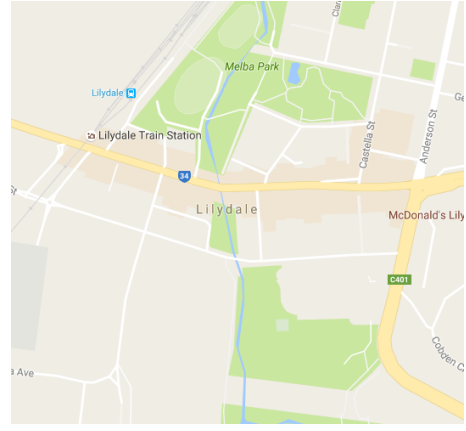


Figure 6: Lilydale Outer Suburb

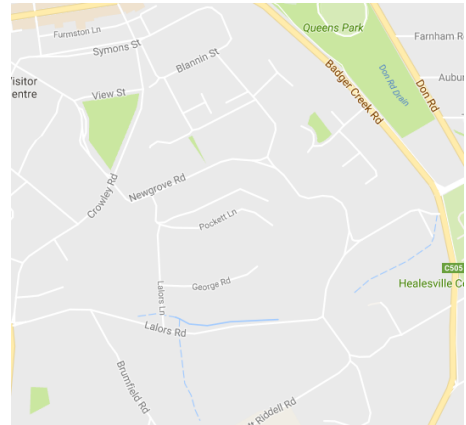


Figure 7: Healesville Rural Area

3.3 Interpretation of Instructions

In this section, we illustrate how to interpret turn instructions on a real road network i.e, we discuss the algorithm for translating turn instructions into possible paths during traveling on road.

Suppose we have a set of turn instructions T_s . A turn instruction is represented as a tuple comprised of a turn primitive tp (left, right, straight) and a distance measure representing the distance traveled since the last junction dt . Thus, T_s contains a set of tuples (tp, dt) . This is the most common method that is available in online and in care navigation systems.

For each possible source in a road network, we first traverse each connected edge and derive the turn primitive and distance. Once derived, provided it is a match to the first element of the set, we then traverse the connected edges of the corresponding edge and look for a match to the second element in the set. This process is repeated until the number of junctions followed is equal to the length of the given turn instructions set. Each time the traversal leads to a complete match, a identical path counter is incremented. The number of resulting identical paths indicates the privacy level of a user's traveled route as the user could have traveled on any one of them.

If only one candidate path is found, then it is known for certain what a route a targeted individual traveled along. If only a small number of paths are found, it is very likely that an adversary will still be able to infer the correct path by

using additional background knowledge. This attack appears to be infeasible as given a set of instructions, in a real large city, we imagine that there could be many matches to a given trip when stripped down to its turn instructions especially in the grid like downtown areas.

Algorithm 2 describes our approach to find matching paths given a sequence of turn instructions.

Algorithm 2: FINDMATCHINGPATHS: Returns the matching candidate paths given a set of turn instructions

Input: A sequence $TS = \{(tp_1, dt_1), (tp_2, dt_2), \dots\}$, a road network graph $G_{RN} = (V_G, E_G)$ and a counter c

Output: A list of paths with matching turn instructions to TS

```

1  $P \leftarrow \emptyset$ 
2 if  $c > |TS|$  then
3   return  $P$ 
4 for all vertices  $v_0 \in G_{RN}$  do
5    $C \leftarrow \text{connectedEdges}(v_0, G_{RN})$ 
6   for all connected vertices  $(v_0, v_1) \in C$  do
7      $\delta_0 \leftarrow TS_c, \delta_1 \leftarrow \text{calculateTS}(v_0, v_1)$ 
8     if  $\delta_0 \equiv \delta_1$  then
9        $P \leftarrow P \cup \text{FindMatchingPaths}(TS, G_{RN}, c+1)$ 
10 return  $P$ 

```

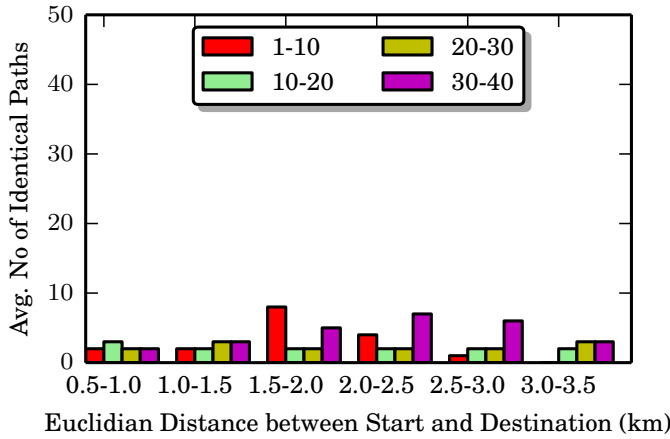


Figure 8: Melbourne Downtown - Avg. No of Identical Paths found for Generated Paths by The Number of Turn Instructions (1-10, 10-20, 20-30, 30-40)

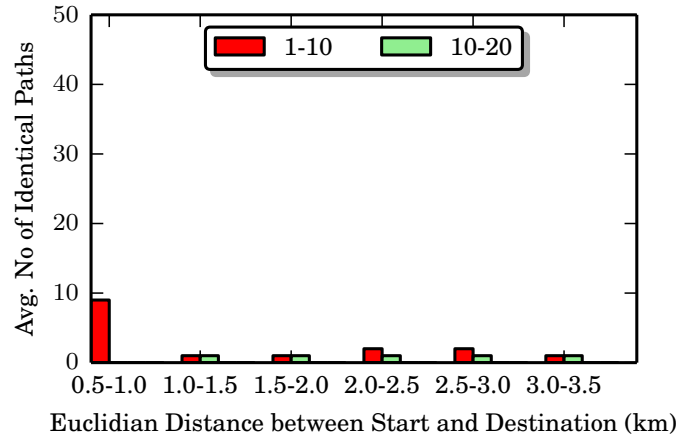


Figure 10: Lilydale - Avg. No of Identical Paths found for Generated Paths by The Number of Turn Instructions (1-10, 10-20)

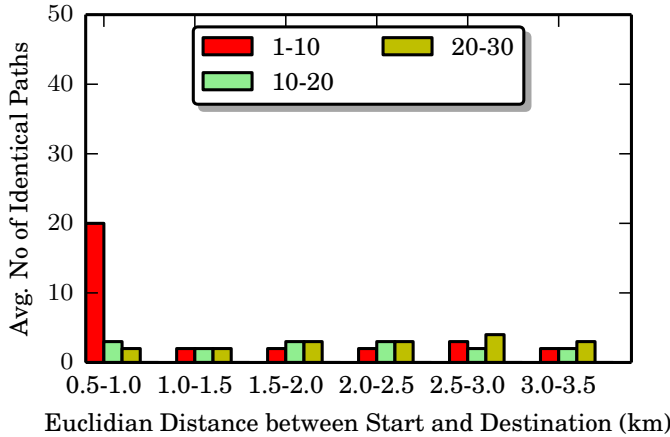


Figure 9: Essendon - Avg. No of Identical Paths found for Generated Paths by The Number of Turn Instructions (1-10, 10-20, 20-30)

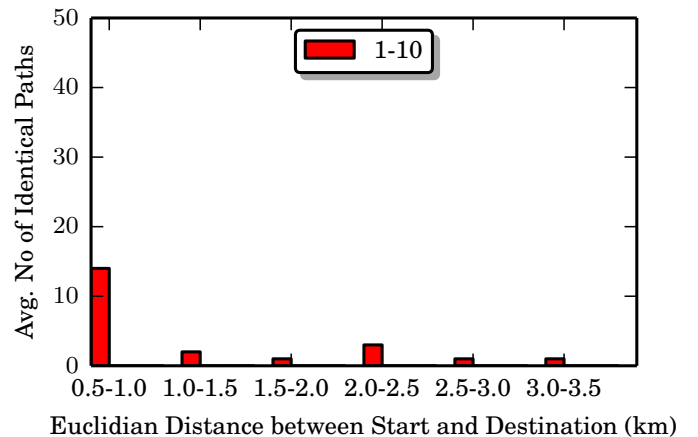


Figure 11: Healesville - Avg. No of Identical Paths found for Generated Paths by The Number of Turn Instructions (1-10, Note: Paths with longer turn instructions could not satisfy the road network)

4. EVALUATION

In our evaluation, we consider the privacy level to be based on the measurement of finding the number of paths with identical turn instructions of a specified path given a road network. We considered a real road network. The aim of our evaluation was to determine the conditions that can lead to turn instructions exposing the route of a user traveled and what circumstances allow for turn instructions to be safely exchanged. We found that there are many situations where user privacy is exposed if turn instructions are shared.

The city we considered suitable for testing was Melbourne, Australia. This is due to the road network in the inner city approximating a grid while in the suburbs the road network is more sparse which is common in many cities around the world. The actual trajectory of a user can be indirectly inferred in instances where only a few identical paths can be identified. Conversely, the privacy of a user is preserved when a large number of paths have the same turn instructions.

Paths residing in Melbourne's inner city and surrounding were generated with turn sequences and used to search for identical sequences. The road network data was sourced

from the OpenStreetMap [8]. A bounding box with an area of 4km^2 was placed around areas of consideration. We generated 100 paths within the test areas. Paths were generated randomly selecting a source and destination pair. The shortest path is then found using Dijkstra's algorithm and the turn sequences are derived.

The Melbourne downtown is arranged in a grid like manner with many intersections leading into main arterials as well as smaller streets and lanes as can be seen in Figure 4. Our results demonstrated in Figure 8 that even when a seemingly large number of combinations possible, user paths are still unique on average.

Similar results were found in the inner city suburb of Essendon and suburbs in its extended area in Figure 9. Essendon is arranged in a grid like structure as can be seen in Figure 5 and has a reasonably high population density for an inner city suburb.

In the outer suburbs, it is easier to infer a path a user takes due to the graph being less interconnected. Consider the suburb of Lilydale in Figure 6, it can be seen that there

are sections of the road network that are not interconnected. The results charted in Figure 10 indicate that at most nine identical turn sequences can be derived and this reduces down to two identical sequences for longer turn instruction sequences.

Rural areas unlike city areas have simplified road networks. Take the map of Healesville, a semi-rural town. It can be seen in Figure 7 that the road network for Healesville is simplified compared to the areas closer to the city. Our results given in Figure 11.

In summary, many paths even in reasonably populated areas with grid like infrastructure can be inferred easily without the need of additional background information. Our findings indicate that data providers that wish to exchange turn direction information in a privacy preserving manner need to consider adding noise using techniques such as differential privacy [5] to ensure privacy is protected.

5. CONCLUSIONS

In this paper, we have highlighted that turn instructions are sufficient to infer a path without access to the original GPS data.

We described our approach for representing a road network, finding the shortest path and methods to derive the turn instructions between a source and destination and reverse engineered them.

The proposed attack we described uses a set of turn instructions, traverses every node in a road network and incrementally expands until a match could be found. Extensive experiments over real road networks have been conducted to evaluate the effectiveness of our proposed attack. LSPs need to be carefully exchanging directions with partners as the original path of a user can be inferred in a majority of cases despite the fact that the data is sanitized from GPS traces such as source and destination locations.

In future work, we plan to determine the metrics on turn instructions and build a system to advise users about potential privacy violations. We also want to investigate methods that can allow for the exchange of these turn directions safely. One possible method we want to explore is differential privacy by adding noise to the instructions.

6. REFERENCES

- [1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 376–385.
- [2] C.-Y. Chow and M. F. Mokbel. Enabling private continuous queries for revealed user locations. In *International Symposium on Spatial and Temporal Databases*, pages 258–275, 2007.
- [3] M. Denis, F. Pazzaglia, C. Cornoldi, and L. Bertolo. Spatial discourse and navigation: An analysis of route directions in the city of Venice. *Applied cognitive psychology*, 13(2):145–174, 1999.
- [4] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
- [5] C. Dwork. Differential privacy. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.
- [6] S. Funke, R. Schirrmeyer, S. Skilevic, and S. Storandt. Compass-based navigation in street networks. In *Web and Wireless Geographical Information Systems - 14th International Symposium, W2GIS 2015, Grenoble, France, May 21-22, 2015, Proceedings*, pages 71–88. Springer, 2015.
- [7] S. Funke and S. Storandt. Path shapes: an alternative method for map matching and fully autonomous self-localization. In *19th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2011, November 1-4, 2011, Chicago, IL, USA, Proceedings*, pages 319–328. ACM, 2011.
- [8] M. M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4):12–18, 2008.
- [9] Z. Huo, X. Meng, H. Hu, and Y. Huang. You can walk alone: trajectory privacy-preserving through significant stays protection. In *Database Systems for Advanced Applications*, pages 351–366, 2012.
- [10] A. Klippel, H. Tappe, L. Kulik, and P. U. Lee. Wayfinding choremes—a language for modeling conceptual route knowledge. *Journal of Visual Languages & Computing*, 16(4):311–329, 2005.
- [11] M. E. Nergiz, M. Atzori, and Y. Saygin. Towards trajectory anonymization: a generalization-based approach. In *Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*, pages 52–61, 2008.
- [12] X. Pan, X. Meng, and J. Xu. Distortion-based anonymity for continuous queries in location-based mobile services. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 256–265, 2009.
- [13] K. Schweizer, S. Katz, and G. Janzen. Orientierung im raum-kognitive grundlagen und sprachliche realisierung. *Tourismus Journal*, 4(1):79–101, 2000.
- [14] B. Tversky and P. U. Lee. Pictorial and verbal tools for conveying routes. In *Spatial information theory. Cognitive and computational foundations of geographic information science*, pages 51–64. 1999.
- [15] M. Westphal and J. Renz. Evaluating and minimizing ambiguities in qualitative route instructions. In *19th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2011, November 1-4, 2011, Chicago, IL, USA, Proceedings*, pages 171–180. ACM, 2011.
- [16] T. Xu and Y. Cai. Location anonymity in continuous location-based services. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*, page 39, 2007.
- [17] T. Xu and Y. Cai. Exploring historical location data for anonymity preservation in location-based services. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008.