ENSURING LOCATION DIVERSITY IN PRIVACY PRESERVING SPATIO-TEMPORAL DATA MINING

by
ABDULLAH ERCÜMENT ÇİÇEK

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabancı University
Spring 2009

ENSURING LOCATION DIVERSITY IN PRIVACY PRESERVING SPATIO-TEMPORAL DATA MINING

APPROVED BY

Asst. Prof. Dr. Yücel Saygın …….
(Thesis Supervisor)

Post-doc Research Fellow Mehmet Ercan Nergiz …….
(Thesis Co-Supervisor)

Assoc. Prof. Dr. Albert Levi …….

Assoc. Prof. Dr. Erkay Savaş …….

Assoc. Prof. Dr. Uğur Sezerman …….

DATE OF APPROVAL: …….

ENSURING LOCATION DIVERSITY IN PRIVACY PRESERVING SPATIO-TEMPORAL DATA MINING

Abdullah Ercüment ÇİÇEK

Computer Science and Engineering, MS Thesis, 2009

Thesis Supervisors: Asst. Prof. Yücel SAYGIN and Dr. Mehmet Ercan NERGIZ

Keywords: Data mining, spatio-temporal data, location privacy, anonymization

Abstract

The rise of mobile technologies in the last decade has lead to vast amounts of location information generated by individuals. From the knowledge discovery point of view, this data is quite valuable as it has commercial value, but the inherent personal information in the data raises privacy concerns. There exist many algorithms in the literature to satisfy the privacy requirements of individuals, by generalizing, perturbing, and suppressing data. The algorithms that try to ensure a level of indistinguishability between trajectories in the dataset, fail when there is not enough diversity among sensitive locations visited by those users.

We propose an approach that ensures location diversity named as *(c,p)-confidentiality*, which bounds the probability of visiting a sensitive location given the background knowledge of the adversary. Instead of grouping the trajectories, we anonymize the underlying map structure. We explain our algorithm and show the performance of our approach. We also compare the performance of our algorithm with an existing technique and show that location diversity can be satisfied efficiently.

GİZLİLİĞİ KORUYAN ZAMAN-MEKAN VERİ MADENCİLİĞİNDE MEKAN ÇESİTLİLİĞİN SAĞLANMASI

Abdullah Ercüment ÇİÇEK

Bilgisayar Bilimi ve Mühendisliği, Yüksek Lisans Tezi, 2009

Tez Danışmanları: Yar. Doç. Dr. Yücel SAYGIN ve Dr. Mehmet Ercan NERGİZ

Anahtar kelimeler: Veri madenciliği, mekan-zaman verisi, mekan gizliliği, anonimleştirme

Özet

Son yıllarda, seyyar teknolojilerin yükselişi, büyük miktarlarda kişisel mekan bilgisinin ortaya çıkmasına yol açtı. Bilgi keşfi noktasından bakıldığında, ticari değer içerdiği için çok değerli olan bu veri, yapısında var olan, kişisel bilgiler nedeniyle gizlilik çekincelerini ortaya çıkardı. Literatürde kişilerin gizlilik gereksinimlerini genelleme, bozma ve baskılama metotlarıyla karşılamayı amaçlayan bir çok algoritma bulunmakta. Bu tarzdaki, kullanıcılar arasında belirli bir ayrılamazlık seviyesi yakalamaya çalışan algoritmalar, kullanıcıların ziyaret ettiği mekanlar arasında yeterli çeşitlilik olmadığında başarısız olmaktadırlar.

Bu çalışmada mekan çeşitliliğini sağlayan bir yöntem önerilmektedir. *(c,p)-gizliliği* adı verilen yöntem, kullanıcıların hassas mekanları ziyaret etme olasılığını saldırganın arka plan bilgisine göre sınırlamaktadır. Bu yöntem rotaları anonimleştirmek yerine, altta yatan haritayı anonimleştirmektedir. Çalışmada algoritma açıklamasının yanı sıra, yaklaşımımızın başarımı da gösterilmektedir. Aynı zamanda algoritmamızın başarımı var olan bir teknik ile karşılaştırılmakta ve mekan çeşitliliğinin verimli bir şekilde sağlanabildiği ortaya konmaktadır.

To my fiancée

&

my family

ACKNOWLEDGEMENTS

First of all, I would like to thank my thesis advisor, Asst. Prof. Dr. Yücel Saygın, for helping and supporting me in every way during my masters. His guidance was invaluable for me to broaden my knowledge in the field and to meet some other influential researchers in the world. I would also thank Mehmet Ercan Nergiz for being my co-advisor throughout my masters, and pushing me forward with his ideas, help and inspiration. Asst. Prof. Dr. Murat Kantarcıoğlu was one of the milestones in my career by giving me the chance to visit United States and benefit from his research vision.

I would like to thank my fiancée, for loving, supporting and being beside me, reviewing my thesis for 15 hours, guiding me in my career and for giving meaning to my life. To my mother, for her caring support in every way to bring me where I am today. To my father, who has given his all to bring me up and who is going to be in my heart forever. I would like to thank my mother-in-law and father-in-law for helping and backing me throughout the hard times in my thesis.

I also thank TUBITAK for providing financial support to me during my masters education.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

1. INTRODUCTION

Recent advances in mobile technologies revolutionized human life radically. The need for enhanced abilities of stationed devices for people who change places frequently has initiated the development of various mobile devices. Mobile phones, pagers, GPS devices, PDAs are no longer considered as luxury items; on the contrary they are parts of our everyday life.

As these devices are carried by their owners everywhere, they collect the location information, along with time stamps, forming spatio-temporal data. From the research and commercial point of view, this data is priceless. One example is the transport authority of a city that tries to model the behavior of the vehicles. This model can be used to regulate the traffic flow and to avoid traffic jams in certain areas, using data mining techniques.

Although collecting personal spatio-temporal data has important applications, it does not come for free. The transport authority may not be trusted. Information about location and time for a person is highly confidential, and thus it is subject to privacy concerns. For instance, a person in the dataset may not want others to know that he or she stops by a nightclub frequently.

One obvious method to protect the privacy of individuals is to remove personal identifiers, such as name or car plate and publish the rest of the data. In this case, an adversary (someone who tries to capture information about an individual in the dataset) can see trajectories (a series of spatio-temporal points that belongs to an individual) in the dataset, but he or she cannot know which trajectory belongs to whom. However, this method has been proven to be insufficient due to the existence of external public resources

[3]. An adversary may look up the home and work addresses of an individual from a public telephone directory and try to find a frequent trajectory between these two locations to identify the individual.

To overcome this problem, a technique called *k*-anonymity has been proposed by [1], [2] and [4]. Although this technique was proposed for tabular data, extensions for spatio-temporal data have been proposed in the literature (see Section 2.2). *k*-Anonymity for spatio-temporal data ensures that for each trajectory in the data set there are at least $k - 1$ indistinguishable trajectories. This means that given a *k*-anonymization, an adversary can at best map any person to a group of *k* trajectories. Achieving optimal *k*-anonymity with minimum distortion has been proven to be NP-Hard in [9] and [10], so various methods based on heuristics have been proposed in the literature (see Section 2.1) to *k*-anonymize a dataset via generalizations and suppressions. On the other hand, *k*-anonymity introduces some problems. First of all, grouping *k* trajectories and then generalizing them to the selected group representative is problematic. The group representative often goes through regions that cannot be passed, such as buildings, lakes, etc. Consider the trajectories in the Figure 1.1. Two trajectories going upwards in parallel roads are grouped with *k* requirement of 2, and the midpoints of grouped locations have been released instead of the real trajectories. Notice that the new trajectory goes through ponds, which is impossible. If an adversary has knowledge about the map of the area, he or she would notice that this is not a valid generalization and can easily map the points in the lake to the nearby roads and therefore can learn more than the anonymization itself permits.

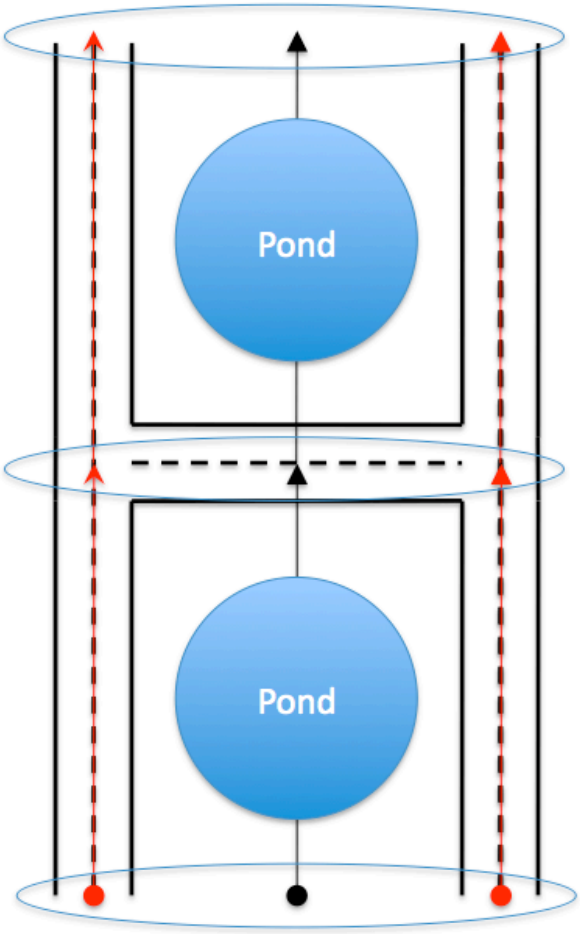


Figure 1.1 A *2*-anonymous but infeasible generalization.

Second problem of *k*-anonymity is that it does not enforce diversity in the sensitive information within the equality groups. In Figure 1.2, one might consider the nightclub as a sensitive region. There are two trajectories that move along the same road and are grouped to provide *2*-anonymity. As for each trajectory, there are exactly *k-1* indistinguishable trajectories in the dataset, without any modifications, the data can safely be released as it is. However, as both trajectories stop by the nightclub, an adversary would be sure that a person he or she knows to be in the dataset visits the nightclub.

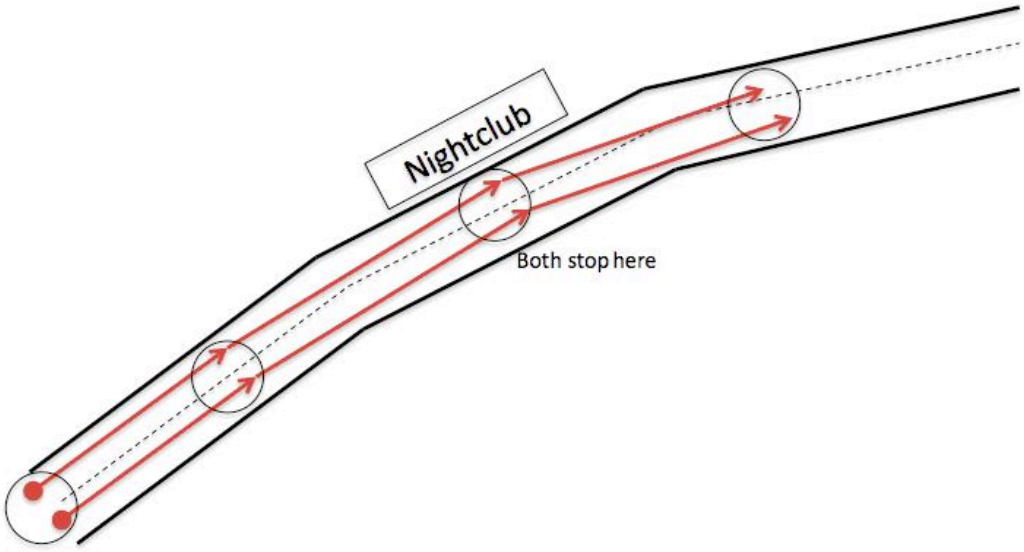


Figure 1.2 *2*-Anonymous generalization that lacks diversity of sensitive location visits.

Yet another problem of using *k*-anonymity for trajectory anonymization is the required level of distortion. Without making a distinction on the locations visited, every single point of the trajectory is considered in the generalization procedure to from groups of *k* identical trajectories. Needless to say, the anonymized data is over-generalized. In spite of the fact that the visits to the non-sensitive areas are not vulnerable, they are generalized to satisfy *k*-anonymity and thus information content is lost unnecessarily. For instance, in Figure 1.2, points other than the visit to the nightclub are not sensitive but have to be grouped.

In this thesis, we propose a technique that relies on map anonymizations, instead of trajectory anonymizations. Our technique addresses the shortcomings of *k*-anonymity explained above. We model the map as a graph where nodes correspond to the locations that can be stopped at, and edges correspond to the roads connecting vertices indicating the direction of the road. Unlike *k*-anonymity, which makes no distinction among the nodes, we specify two types of nodes: *Sensitive* and *Non-Sensitive* nodes. Non-Sensitive nodes are considered as public areas and disclosing that someone stops by these nodes does not cause privacy violation. On the other hand, someone stopping by a sensitive node is considered as

a sensitive information and should be protected. Examples for sensitive nodes are nightclubs, religious or political organizations, hospitals, etc..

We assume that the sensitive nodes on the map have already been specified by the data owner. Based on that, we generate groups around these sensitive nodes in order to create a super node that satisfies our privacy metric. This super node replaces nodes and edges in this group. We introduce privacy parameter p to measure the level of privacy protection of the individuals and introduce a parameter c that limits the background knowledge of the adversary. Using these parameters we introduce our privacy definition: *(c,p)-confidentiality*. To be more precise, p corresponds to the probability that a trajectory stops at a sensitive node in the group using a specific path. Parameter c corresponds to the maximum number of vertices before and after the sensitive node. Visiting those nodes affects the probability of stopping at the sensitive node. Probability is calculated given the current state of the traffic around the sensitive node.

Our contribution can be summarized as follows:

1- We propose a technique that, unlike *k*-anonymity, addresses the map inconsistency problem as it purely works on the map rather than trajectories. The released data is composed of the nodes in the map and therefore we always generate generalizations that are consistent with the real world map.
2- Unlike *k*-anonymity, our technique addresses the diversification problem by introducing the notion of sensitive and non-sensitive nodes. Instead of constraining on the number of individuals following a path, we ensure that the probability of stopping at a sensitive node is bounded by the privacy parameter.
3- Unlike previous approaches, we apply generalizations only on thereabouts of private locations.
4- Our approach can be used in many spatio-temporal applications. For instance, it can release rules for online systems (such as suppressing some routes) and it can generalize and suppress trajectories for offline applications.
5- The complexity of our technique is independent of the density of users.

6- Given that the statistics regarding the state of the traffic are taken from a public source, our technique generates anonymizations resistant to minimality attacks [17], which are made by adversaries who also know the anonymization algorithm.

The rest of the thesis is organized as follows: In Chapter 2, we present the previous research in anonymization. We formulate the problem in Chapter 3. In Chapter 4, we give a detailed explanation of our methodology. In Chapter 5, we experimentally evaluate the performance of the proposed algorithm, and also compare it with the *k*-anonymity approach. Finally, Chapter 6 is dedicated to the conclusion and future work.

2. BACKGROUND AND RELATED WORK

This chapter is organized as follows: First, we dwell upon the general notion of anonymity for tabular data in Section 2.1. Then, we proceed to the anonymity notion in spatio-temporal data in Section 2.2. After that, we consider works in spatio-temporal data in two sub-sections: Section 2.2.1 is dedicated to the previous work on privacy issues regarding Location Based Services (LBS) and finally, Section 2.2.2 explains the related work on releasing static spatio-temporal data.

2.1 Anonymization of Tabular Data

Removing personal identifiers from the microdata (raw data to be published [5]) for data release has been shown to be insufficient for privacy protection [1]. Other attributes such as age, race, sex, etc. can be used to link identities of individuals, to records in the dataset.

Definition 2.1 - *Quasi-identifiers (QI)* are the set of attributes that are not personally identifying, but lead to identification of individuals through linkage of some public resources is called

The *k*-anonymity concept has been proposed as a solution to the problem of linkage through *QI* in [1], [2] and [4]. The approach ensures that for each record there are at least *k-1* other indistinguishable records in the data, with respect to the selected quasi-identifiers.

Definition 2.2 - An *equivalence class* is a set of tuples such that they are indistinguishable with respect to a set of attributes.

Definition 2.3 - A Table *T* is *k-anonymous* with respect to a set of attributes *QI* if and only if for each equivalence class *E,* formed with respect to *QI,* $|E| \geq k$.

Table 2.1 – A public dataset.

Name	Age	Sex	Zip Code
Ben Brown	16	M	44106
Rose East	25	F	44107
George Pry	20	M	44107
Helen West	30	F	44106

Table 2.2 – A dataset without personal identifiers.

Age	Sex	Zip Code	Disease
16	M	44106	Cancer
25	F	44107	Flu
20	M	44107	Cancer
30	F	44106	Cold

Table 2.1 shows an example public dataset with no sensitive information. Using *Age, Sex* and *Zip code*, attributes which are non-sensitive, one can map the tuples in Table 2.2 to specific individuals in Table 2.1, even if Table 2.2 does not contain any personal identifiers.

Table 2.2 shows an example dataset to be released. It is not *k*-anonymous as the *Age* attribute is distinct for each tuple when *k=2*. Table 2.3 shows a *2*-anonymous version of the dataset in Table 2.2. First two tuples and the last two tuples, form equivalence classes, each containing two tuples, and thus the table satisfies the *k*-anonymity requirement.

Table 2.3 – *2*-anonymous version of the dataset in Table 2.2.

Age	Sex	Zip Code	Disease
[16-20]	M	4410*	Cancer
[16-20]	M	4410*	Cancer
[25-30]	F	4410*	Flu
[25-30]	F	4410*	Cold

Achieving optimal *k-anonymity* with minimal distortion has been proven to be NP-Hard in [9], and [10]. There are many methods proposed in the literature based on heuristics. In [2] and [4], Samarati et al. propose heuristics to achieve *k*-anonymity via generalizations and suppressions. Their work assumes generalization hierarchies for *QI* attributes that specify which value can be generalized to which. The proposed technique tries to reach optimal *k*-anonymization with minimal distortion. Iyengar employs a genetic algorithm to find a near-optimal solution to this problem in [7]. The proposed technique makes a distinction between various usages of the anonymized data, such as classification or regression. To achieve better results, the anonymization procedure is altered according to the targeted usage. Later, a simulated-annealing based procedure is proposed by Winkler [8]. Wang et al. propose a fast and scalable, classification specific, bottom-up anonymization scheme in [42]. While methods mentioned above try to generalize the data in a bottom up fashion (from the most specific to the least specific value), Fung et al. specialize the data in a top-down fashion (from the least specific to the most specific value) in [6]. In [12], LeFevre et. al. use the same methodology of [2] and [4], which is called Full-Domain Generalization. They use a domain generalization graph, which is used to move in the search space of possible generalizations of singular attributes.

To address shortcomings of standard *k-anonymity* some new classes of algorithms have been proposed. LeFevre et al. introduce a new multidimensional model for *k*-anonymity and an efficient greedy algorithm to achieve *k*-anonymity [11]. They partition the multi-dimensional space into non-overlapping regions, each containing at least *k* tuples. This way, they consider all attributes in the quasi identifier at the same time.

Machanavajjhala et al. addresses the lack of diversity of sensitive information in the notion of *k*-anonymity and show that the released data is very likely to leak information when all anonymized tuples share the same sensitive value [13]. They also show that with enough background information, which is again very likely to exist, an attacker may deduce more information than expected with the same background knowledge. They propose a method called *l*-diversity to overcome these problems.

Definition 2.4 - A Table *T* is *l*-diverse with respect to a set of attributes *QI* if and only if in each equivalence class *E,* the probability of occurrence of the most frequent sensitive value is less than *1/l*.

Notice that, although Table 2.3 is *2*-anonymous, it is not *l*-diverse when *l=2*. The first equivalence class has no diversity among the sensitive attribute field, as both tuples share the disease of cancer. Thus, an adversary will discover that someone he or she knows to be at the age of 16 and to be in the dataset has the disease cancer even if the table is *2*-anonymous. On the other hand, the anonymization in Table 2.4 satisfies both *2*-anonymity and *2*-diversity.

In [14], Truta et al. similarly point out to the lack of attribute disclosure (disclosure of not the identity, but sensitive information of the record) shortcoming of *k*-anonymity and introduce the *p*-sensitive *k*-anonymity method. Li et al. in [15] have defined *t*-Closeness, which is the extension of *l*-diversity. They enforce that the distribution of a sensitive value in an equivalence class should be close to the distribution of the sensitive value in whole data within a threshold. In [16], Wong et al. provide the concept of (α , *k*) anonymity, which enforces that the ratio of sensitive attributes in a group does not exceed the threshold α, while satisfying *k*-anonymity. Nergiz et al. point out to the inability of previous privacy metrics to preserve the privacy of an individual when the existence or non-existence of an individual in a dataset is private information [19]. The work ensures that, no person should be known to be in the dataset, with certainty greater than δ. Our approach mainly addresses the questions raised in [13] and [14] in the spatio-temporal domain. While diversifying the sensitivity of nodes like in [13], we limit the probability of an individual to stop at a sensitive location.

Table 2.4 – *2*-anonymous and *2*-diverse version of the dataset in Table 2.2.

Age	Sex	Zip Code	Disease
[16-25]	*	4410*	Cancer
[16-25]	*	4410*	Flu
[20-30]	*	4410*	Cancer
[20-30]	*	4410*	Cold

2.2 Anonymization for Spatio-Temporal Data

The concept of anonymity has been used in spatio-temporal data mining frequently. Although spatio-temporal data can be represented in the tabular format, it cannot be anonymized in the same manner, due to the fact that records depend on each other and depend on geographical features. Hence the existing techniques need to be extended. The literature on spatio-temporal data anonymity can be reviewed in two subsections: (1) Location Based Services (LBS), (2) Static Data Release (SDR).

LBS are the services provided to users, using the location information gathered by mobile devices. Asking for the nearest hospital is an example among many uses of LBS. By the nature of the system, LBS providers service users online and therefore need to work on streaming data. The goal of privacy preserving LBS is to provide service to the user, without learning the exact position of the individual.

SDR applications work on already collected, static data. The goal is to publish a trajectory dataset, while keeping identities of the people in the dataset private.

Our work has no restrictions on the type of application; we yield rules for LBS systems, and modify the dataset to be released in the SDR systems, hence it is applicable to both scenarios.

2.2.1 Anonymity in Location Based Services

k-Anonymity concept has been extensively studied in LBS literature. Obfuscation is one of the techniques to satisfy *k*-anonymity. Gruteser et al. ensures *k*-anonymity through spatial and temporal cloaking, which means reducing preciseness of spatio-temporal information and adding uncertainty into sensitive information of the user [20]. They require *k* users to exist in the same area to provide service. Gedik et al., on the other hand, require not just *k* users, but *k* users that make requests to exist in the same area, to forward a request to the service provider [21]. They propose an algorithm called *CliqueCloak* that performs spatio-temporal cloaking and lets user to supply his or her Quality of Service (QoS) choice and privacy level. Bettini et al. has the same interpretation of *k*-anonymity with [20], but unlike [20] and [21], they do not consider every location as sensitive and present Location-Based Quasi-Identifiers (LBQIDs) that are defined by some constraints [22]. They consider the LBQIDs and history of the users' behavior in their approach. Distributed scenarios have been studied in the literature as well. *Privé* is a framework that is proposed by Ghinita et al. which provides *k*-anonymity through a decentralized trusted server [23]. Doing so, they reduce the risk of being disclosed and reduce the bottleneck created by a single service point.

In [24], Mokbel et al. propose a framework called *Casper*, which acts as a personalized privacy provider and a query processor. They represent the area as a grid and organize it in a pyramid structure to perform cloaking in a bottom-up fashion. Like [24], Divanis et al. represent the area as a grid in [25]. They make a distinction between safe and unsafe routes (frequent for the user but not frequent for others) of the user. They provide *k*-anonymity by obfuscation, when a request is made in an unsafe route for a user. Cheng et al. propose an imprecise query engine that evaluates cloaked location information in [26]. In [27], Duckham et al. introduce a framework that seeks a balance between the QoS and the privacy requirements of users. Unlike [24] and [25], they model the spatial region as a graph, like we do.

Anonymity via suppression and perturbation are also popular methods in the literature. In [28], Gruteser et al. make a distinction between sensitive and non-sensitive areas which is similar to our work. They try to avoid disclosure of visits to sensitive areas. They have three methods: (1) Completely suppressing location updates in sensitive areas, (2) Reducing the frequency of location updates, (3) Releasing the request only when it is indistinguishable from *k* sensitive areas the user has visited before. In [29], Beresford et al. propose the *mixed-zone* concept. Mixed zones are regions which act as black-box zones. The trajectory behavior inside these zones are masked. Moreover, there is no link between the trajectories entering and exitting the zone, so that an adversary is less certain about the identity of a trajectory. This is a blurring approach parallel to ours, but the main difference is that we do not break the ties between entering and exiting trajectories. They also provide a sensitivity classification of locations like us, but we differ with the generalization procedure. [30] by Hoh et al. is a similar work to [29]. Instead of suppressing, they perturb trajectories and force them to cross each other. The approaches in [29] and [22] work fine in high-density areas, but fail in low-density areas, since trajectories rarely get close to each other. The density of users in the area does not affect our approach as it focuses on obfuscating the geographical area instead of the trajectories.

2.2.2 Anonymity in Static Data Release

Compared to the literature on LBS anonymity, privacy preserving static trajectory publication is almost an untouched area.

Clustering approach is one of the most popular techniques in anonymization. In [31], Domingo-Ferrer et al. propose different grouping techniques based on the dimensionality of the data. They do not project the multi-dimensional data into single dimension and therefore achieve higher utility unlike the works [32], [33] and [34]. Aggarwal et al. propose another clustering method. They propose release of the cluster centers instead of grouped tuples [35]. Byun et al. propose another clustering method that works in $O(n^2)$ [36]. Based on a different clustering technique that exploits the inherent uncertainty in

spatio-temporal points, Bonchi et al. group trajectories that are close to each other, in [37]. They generalize the grouped trajectories and suppress the outlier points. Nergiz et al. on the other hand, propose a method to generalize clustered trajectories using a generalization based approach, while maximizing the utility metric called: *Log-Cost Metric* in [38].

In [39] Hoh et al. propose a suppression based method that fixes the shortcoming of [22] and [29], with a new privacy metric called time-to-confusion. They specify an path cloaking algorithm that ensures satisfaction of time-to-confusion metric. In [40], Terrovitis et al. consider privacy preservation in vertically partitioned spatio-temporal data (e.g. one site holds the head of the trajectory and the other holds the tail). They do not distinguish between sensitive and non-sensitive nodes. The goal in this work is to limit the prediction of the tail of the trajectory with some predefined probability threshold, given the head. They use suppression to satisfy this probability. Their goal of limiting the conditional probability mentioned is similar to our goal of limiting the ratio of the trajectories that stop by a sensitive node, given the trajectory.

Obfuscation is a method that has been widely deployed in the literature [20],[21],[22],[23],[24],[25],[26] and [27]. All of these works are in LBS context. To the best of our knowledge, ours is the first work that uses spatial region obfuscation in SDR context. This is mainly because of the fact that SDR literature has focused on trajectory anonymization, rather than location anonymization.

It is in the nature of LBS systems to focus on region, as they do not see the whole trajectory while serving a client. They focus on a single location update received from the user and the area that the request has been made in. On the contrary, SDR related algorithms have advantage of accessing the complete data. We approach the problem from an LBS point of view, while exploiting the advantages of complete world knowledge.

3. PROBLEM FORMULATION

In this chapter we give a detailed explanation of the definitions and notations we have used throughout the thesis.

3.1 Notation

Definition 3.1 – *G* is the graph model used to represent the underlying geographical area, which can be considered as a representation of the map.

$$G = (V, E)$$

Definition 3.2 – A *vertex v* (called node or point throughout the thesis as well) is a 2D structure that corresponds to a point in the spatial region.

$$v = [x,y]$$

Definition 3.3 – The set of vertices *V* consist of locations where user location information is sampled. These are the points of interest, on which users can stop.

$$V = \{v_1, .. , v_f\}, |V| = f$$

Definition 3.4 – *E* is the set of edges that connect vertices in *V*.

$$E = \{e_1, .. , e_r\}, |E| = r$$

Definition 3.5 – Each directed edge *e,* connects two vertices, indicating the direction of the road.

$$e_i = [v_x - v_y] \text{ s.t. } v_x, v_y \in V \text{ and there is a link from } v_x \text{ to } v_y$$

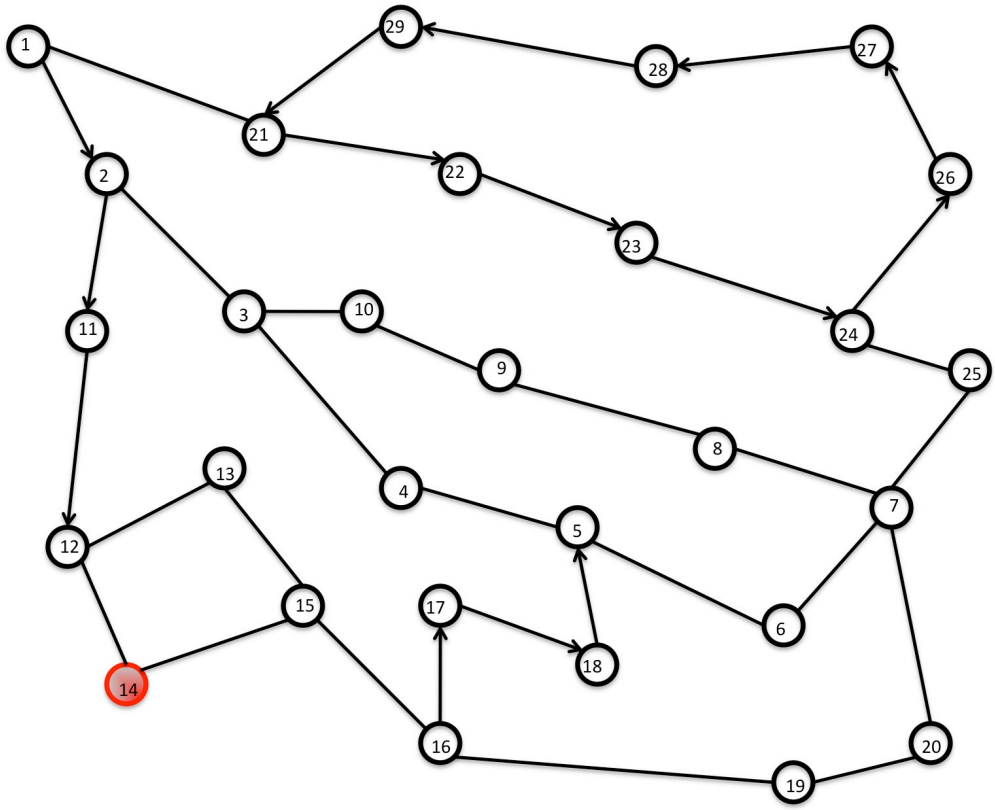


Figure 3.1 Graph representation of the geographical area.

Definition 3.6 - A *sensitive node* correspond to a location where stopping by may be considered as a privacy breach.

Definition 3.7 - Set *S is the* collection of sensitive nodes.

$$S = \{v_1, .. , v_t\}, |S| = t \text{ such that } S \subseteq V, \quad \forall v_i \in S, v_i \text{ is sensitive.}$$

Figure 3.1 shows a graph representation of the geographical area. Numbered nodes are connected via edges. Undirected edges correspond to two directed edges, which means that there is a two-way connection between two vertices. We mark vertex *14* to denote that it is a sensitive node.

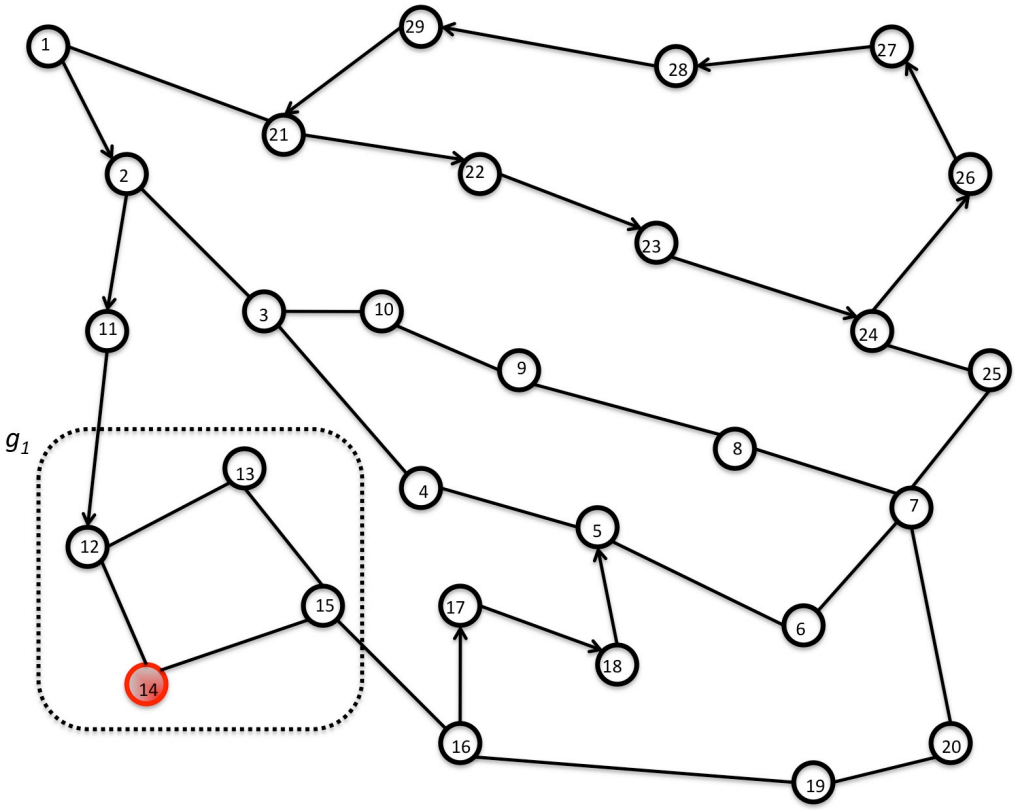


Figure 3.2 Example grouping for the example in Figure 3.1.

Definition 3.8 - A *group g* is a connected subgraph of *G*.

$$g_a = (V_a, E_a) \text{ where}$$

$$V_a \subseteq V,$$

$$E_a \subseteq E \text{ s.t. } e_i \in E_a \text{ and } e_i = [v_x - v_y] \text{then } v_x \in V_a \text{ and } v_y \in V_a,$$

$$\text{and } g_a \text{ is connected}$$

Definition 3.9 - *Gr* is the set of discrete groups in *G.*

$$Gr = \{g_1, .. , g_c\}, |Gr| = c$$

Definition 3.10 - Given *Gr,* generalization of graph *G, G'* with respect to *Gr* is a new graph in which all nodes belong to the group g_i has been replaced by a super node n_i representing g_i and edges are modified accordingly.

$$G'=(V',E') \text{ with respect to } Gr = \{g_1, .., g_c\} \text{ is a generalization of } G=(V,E) \text{ s.t.}$$
$$V'=(V-\cup_i V_i)+ \cup_i n_i,\ E'=(E - \cup_i E_i)+A+B,$$
$$\text{where } A=\{[v_x - n_i] \mid [v_x - v_y] \in E \wedge v_y \in V' \wedge v_x \in V \wedge v_x \notin V'\},$$
$$B=\{[n_i - v_x] \mid [v_y - v_x] \in E \wedge v_y \in V' \wedge v_x \in V \wedge v_x \notin V'\},$$
$$y \in \{1..c\}$$

Figure 3.2 depicts an example grouping on the map in Figure 3.1. Due to a probable violation of privacy, vertex *14* is generalized (the method is going to be explained in Chapter 4) to g_1. We denote the set of sensitive nodes *S* in a group *g* using the dot (.) notation (e.g. *g.S* for group *g*). This instance can be summarized as follows:

$$g_1 = (V_1, E_1)$$
$$V_1=\{12, 13, 14, 15\}$$
$$E_1=\{[12\text{-}13], [13\text{-}15], [15\text{-}14], [14\text{-}12], [12\text{-}14], [14\text{-}15], [15\text{-}13], [13\text{-}12]\}$$
$$g_1.S=\{14\}$$

Definition 3.11 - An *Entrance* to a group g_a, in a generalization *G',* is defined as a node that is not in *g* but has an outgoing link to a node in *g.*

$$\text{Vertex } v \text{ is an entrance node iff } e \in E, \text{ s.t. } e = [v_x\text{-}v] \text{ where } v \in V_a,\ v_x \in V \text{ and } v_x \notin V_a$$

Definition 3.12 - An *Exit* from a group *g,* in a generalization *G',* is defined as a node that is not in *g* but has an incoming link from a node in *g.*

$$\text{Vertex } v \text{ is an exit node iff } e \in E, \text{ s.t. } e = [v\text{-}v_y] \text{ where } v \in V_a,\ v_y \in V \text{ and } v_y \notin V_a$$

Definition 3.13 – *En* is the set of entrances to the group g_a, in a generalization *G'*. It is denoted by the dot notation (e.g. *g.En* for group *g*).

$$En = \{v_1, .. , v_h\},\ |En| = h,\ s.t.\ En \subseteq V_a,\ v_i \in En,\ v_i \text{ is an entrance node}$$

Definition 3.14 – *Ex* is the set of exits from the group *g,* in a generalization *G'*. It is denoted by the dot notation (e.g. *g.Ex* for group *g*).

$$Ex = \{v_1, .. , v_z\},\ |Ex| = z,\ s.t.\ Ex \subseteq V_a,\ v_i \in Ex,\ v_i \text{ is an exit node}$$

According to the definitions above, for the example in Figure 4, the *Ex* and *En* sets are formed as follows:

$$g_1.En = \{12, 15\} \text{ and } g_1.Ex = \{15\}$$

Definition 3.15 -A *Route* specifies an adversary's view of a trajectory's movement in the group. Since, the group is an obfuscation mechanism, an adversary may only deduce the entrance and the exit of a trajectory in and out of the group. Therefore a group may have four types of routes:

1. A trajectory enters from an entrance and exits from an exit.
2. A trajectory enters from an entrance but does not exit.
3. A trajectory starts inside the group and therefore does not enter the group but exits from an exit.
4. A trajectory starts and then ends in the group, therefore neither enters nor exits.

$$r = \langle v_1, v_2 \rangle \ s.t.\ (v_1 = null \wedge v_1 \in En) \vee (v_2 = null \wedge v_2 \in Ex)$$

Definition 3.16 - *R* is the set of all routes in group *g,* in a generalization *G'*. It is denoted by the dot notation (e.g. *g.R* for group *g*).

$$R = \{r_1, .. , r_d\},\ |R| = d$$

Considering the example in Figure 5, route list $g_1.R$ is the following:

$$g_1.R=\{ <12, 15>, <15, 15>, <12, null>, <15, null>, <null, 15>, <null, null>\}$$

For instance a trajectory following the vertices *11→ 12→ 13* is considered to be taking the route *<12,null>* as the trajectory enters the group from vertex *12* and stops at vertex *13* and thus, does not exit the group. Likewise, a trajectory following the vertices *15→ 13*, is considered to be taking route *<null,null>* as it neither enters the group, nor exits the group.

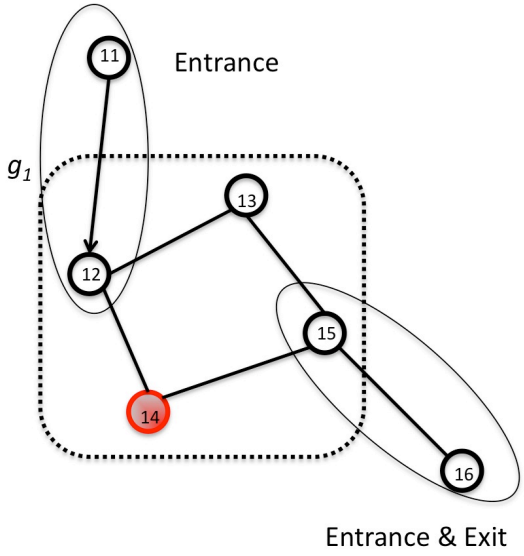


Figure 3.3. Entrance and exits of group g_1.

Definitions regarding trajectories are similar to the definitions in [39].

Definition 3.17 - A point *p* in the spatio-temporal domain is a triple with two dimensions for spatial representation and one dimension for temporal representation.

$$p=[x,y,t]$$

For the sake of simplicity, we do not show the time dimension in the figures throughout the thesis and show 2D figures, which are easier to comprehend.

Definition 3.18 - A trajectory *tr* is an ordered series of spatio-temporal points. A point is referred using dot notation (e.g. $tr.p_i$ for *trajectory tr*).

$$tr_i = \{p_1 , .. , p_m\}, |tr_i| = m$$

Definition 3.19- A trajectory dataset *T* is collection of trajectories:

$$T = \{tr_1 , .. , tr_n\}, |T| = n$$

3.2 Problem Definition

We assume that we have the graph representation of the geographical area as explained in Section 3.1. For the SDR context, we have a static and complete trajectory dataset without personal identifiers. For LBS context, we assume that the background information about user movements on the map is obtained.

Our adversary model can be considered as a strong one. The knowledge of the adversary is listed below:

1. The adversary knows which points are considered sensitive in the map.
2. The adversary has background information about the target individual, such as home and work addresses, so that it is possible for him or her to identify the target despite the fact that there are no personal identifiers on trajectories.
3. The adversary has background information about general user behavior on a map (See definition 3.20), just as the data owner. This might be public information, or the adversary might have gained this information simply by observation.

Definition 3.20 – Strong Adversary Background: Given a trajectory the adversary can guess the probability of the trajectory stopping at a node. Thus, adversary has the following prior belief:

$$P(\text{a trajectory } t \in T \text{ stops at a vertex } v \in V \mid t)$$

In the ideal case each single location the trajectory visits, affects the probability of stopping at a point. However, not every point has a significant effect on the probability. A node far away from a set of possible stopping locations will not contribute much on adversary's belief on the exact stopping location. In this work we assume that for a trajectory, probability of stopping at a vertex *v*, given the whole trajectory is equal to probability of stopping at a vertex given the portion of the trajectory that is close to *v:*

$$P(\text{a trajectory } t \in T \text{ stops at a vertex } v \in V \mid t) \approx P(\text{a trajectory } t \in T \text{ stops at a vertex } v \in V \mid t) \text{ where each point } t.p_i \in t \text{ is 'close' to } v$$

The closeness measure (called *c* from now on) is a parameter to our algorithm. We consider the vertices that are within *c* nodes distance from the target node (without considering the direction of the edges). Collections of these vertices and the edges that connect them form the *neighborhood* of the target node. We assume that only the vertices in the neighborhood of the target node *v* are effective on an adversary's posterior belief on an individual stopping at *v*.

Definition 3.21 – *c-Neighborhood* of a group *g is* $g^c = (V^c, E^c)$ such that $v_i \in V^c$ is at most *c* hops away from *v,* where *v* is the initiating vertex in *g* and *c* is a positive integer. Initiating vertex is the vertex that grouping starts with.

Figure 3.4 shows g_1^2 which is the 2-neighborhood of g_1. We assume that the portion of a given trajectory in this neighborhood effectively changes adversary's posterior belief

and therefore we do not consider the rest of the trajectory in our calculations. We now revisit the definition of the background knowledge of the adversary after this assumption.

Definition 3.22 – Effective Adversary Background: Given the portion of the trajectory in a neighborhood of a target node, the adversary knows the probability of the owner of the trajectory stopping at the target node. Here we assume that the adversary has information only on the statistics of the neighborhood of the target node.

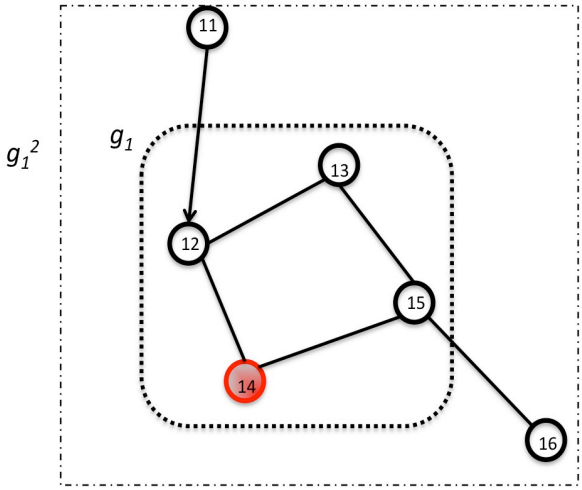


Figure 3.4 2-Neighborhood of group g_1.

Definition 3.23 – A *Path* pt_i in a *c*-neighborhood g^c is a series of locations belonging to a portion of a trajectory in the neighborhood. A path may or may not enter the corresponding group g. The potion of the trajectory within the group g is replaced with the route it is taking. All paths in a c-neighborhood form the set Pt^c. Given a group g_1 and c-neighborhood g_1^c:

$$Pt_1^c = \{pt_1, .. , pt_n\} , |Pt_1^c| = \text{n},$$

$$pt_i = \{a_1, .. , a_m\} \text{ where } a_q = v \in V_1^c \text{ or } a_q \text{ is a } route \text{ in } g_1.$$

Definition 3.24 – An equivalence class *eq* is the set of trajectories in a c-neighborhood g^c that follow the same path in g^c. All equivalence classes in a c-neighborhood g^c *form the set* Eq^c. Given a group g_1 and c-neighborhood g_1^c*:*

$$Eq_1^c = \{eq_1, .. , eq_n\} , |Eq_1^c| = n,$$
$$eq_i = \{tr_1, .. , tr_m\} \text{ where each } tr_x \text{ follow the same path } pt_i \in Pt_1.$$

Considering all trajectories that enter into the c-neighborhood of the target vertex *v* in a group g, we define a new term called *probability of disclosure* for paths and for groups. These effectively change the posterior belief of the adversary given a generalization, thus we try to bound these probabilities in the following sections.

Definition 3.25 – *Probability of disclosure* of a path, *pt* in c-neighborhood g^c, considering a target vertex *v,* is the ratio of the size of the equivalence class belonging to trajectories that take *pt* and that stop at *v*, to the size of the equivalence class belonging to trajectories that take *pt* and that stop at some location in the group *g*.

Definition 3.26 – *Probability of disclosure* of a group is the biggest *probability of disclosure* of a path in the c-neighborhood of that group.

Given the above-mentioned capabilities of the adversary, the goal of this work is protecting a user's privacy via bounding the *probability of disclosure* of the group that he or she visits with user defined level of privacy called *p.* Now we are ready to define *(c,p)-confidentiality.*

Definition 3.27 - Given a graph *G (with sets S and Gr),* a trajectory dataset *T* and given events A and B, *(c,p)-confidentiality* is satisfied iff $P(A|B) \leq p$ and therefore $P(AB)/P(B) \leq p$ where A is the event that a trajectory $t \in T$ stops a vertex $v \in V_i$ & *v* is sensitive*,* B is the event that *t* follows path pt_j and t stops in *g,* $g_i \in Gr$ with corresponding

g_i^c and $pt_j \in Pt_i$, p is a user determined level of privacy and c is the user defined measure of closeness.

Definition 3.28 – Problem of *(c,p)-confidentiality*: Given a graph G *(*with sets S and Gr*)* and a trajectory dataset T, find a generalization G' of G satisfying *(c,p)-confidentiality.*

4. GENERALIZATION & SUPPRESSION SCHEME

The main methods used to satisfy *(c,p)-confidentiality* is generalization and suppression. However, there are four phases of the algorithm:

1. Collection of Statistics: The collection of statistics from internal or external data.
2. Generalization: Enlarging groups when the *(c,p)-confidentiality* is not satisfied.
3. Suppression: Trashing paths and corresponding trajectories, when *(c,p)-confidentiality* cannot be satisfied.
4. Output: Modification of the trajectory dataset in SDR context and release of rules in LBS context.

The utility mechanism to measure the success of the procedure is simply the level of generalization, in other words it is the average group size. The rate of the suppressed trajectories is a supporting indication of the information content. The goal is to satisfy the privacy condition with minimum average group size. Suppression mechanism acts as a fuse to ensure that utility is kept within an acceptable range, and we do not generalize too much.

4.1 Collection of Statistics

The algorithm needs some background information to base calculations on. Given a group and the c-neighborhood, we need to calculate *probability of disclosure* of each path. This information may be gathered from a trajectory dataset, or through observation.

For the SDR context, the statistics can be obtained from the data itself. As the data is static and complete, before running the algorithm the data owner may extract such statistics from the data and use it. In LBS case, we do not have such a complete dataset. The data may be collected offline from a training data or it may be gathered from external sources.

As most algorithms seek utility along with anonymity, they try to find the generalization that is minimal. While trying to maximize utility, the choices of the algorithm may leak information to the adversary. An algorithm is *minimality attack* resistant if information leakage is independent of the public availability of the algorithm, which is defined in detail in [17]. In our work, if background statistics are gathered from an external source, or some portion of the data is used as training data, then there is no relation between the anonymized data and the choices of algorithm. Hence our approach is minimality attack resistant when external background information is used.

4.2 Generalization Procedure

The goal of anonymization is to limit the *probability of disclosure* of each path in the c-neighborhood of a group. The nodes are grouped to satisfy this condition. Adding neighboring nodes to the group further blurs the information seen by the adversary.

```
Input: G, S, Gr = {}, Cutoff_Limit, User Stats St
Output: G'
Anonymize()
Begin

    For each v_i ∈ S

        Create a new group g;
        Add v_i to g;
        GetNeighborhood(c);
        GetRoutes(g);              //Figure 4.2
        GetEquivalenceClasses(St);   //Figure 4.7

      While !Check_cp_Confidentiality(g, S, St) & !willBeSuppressed(Cutoff)
                                                        //Figures 4.5 and 4.8

          Vertice tobeAdded = get_Neighboring_Vertex();
          Add tobeAdded to g;
          GetRoutes(g);
          GetEquivalenceClasses(St);

       End While

    End For

     If LBS then
        ReleaseRules(Gr);  //Figure 4.13
     else
        G' =ModifyDataset(G, Gr);     //Figure 4.11
     End If

     return G';

End
```

Figure 4.1 Pseudocode for Anonymization.

4.2.1 The Algorithm Flow

The algorithm starts by collecting the sensitive nodes in the graph *G* to form the set *S*. For each sensitive node in the set *S*, a group with a single node (the initiating node) is

formed (see Figure 4.1). Given the entrances and exits to the group, the sets *En* and *Ex* are generated (see Figure 4.2). The c-neighborhood of the group is determined according to the given *c* value and the equivalence classes are obtained from the user statistics based on the sets *En* and *Ex* (see Figure 4.7). For each equivalence class in the c-neighborhood, *(c,p)-confidentiality* is tested for the given *p* value (see Figure 4.5). If there is no violation of *(c,p)-confidentiality,* then the group is released as a super-node in the graph. If some path violates *(c,p)-confidentiality* then a neighboring node is added into the group as well (which node is added is going to be explained in Section 4.2.2), and the algorithm performs same checks for the new group with the new node (see Section 4.3 for a relaxed version based on suppression). Note that another sensitive node may be added into the group. In this case, the algorithm checks *(c,p)-confidentiality* for each path and each sensitive node in the group. However, the neighborhood of the group is the same as the neighborhood of the initiating sensitive node.

```
Input: Group g
Output: Route List R
GetRoutes()
Begin
      Clear R;
      For each en ∈ En of g
        For each ex ∈ Ex of g
            Create a new route r with entrance en and exit ex;
            Add r to R;
         End For
       End For

      For each en ∈ En of g
          Create a new route r with entrance en with no exit;
          Add r to R;
      End For

      For each ex ∈ Ex of g
            Create a new route r with exit ex with no entrance;
            Add r to R;
      End For

     Create a new route r with no entrance and no exit;
     Add r to R;
     return R;
   End
```

Figure 4.2 Pseudocode for generating routes.

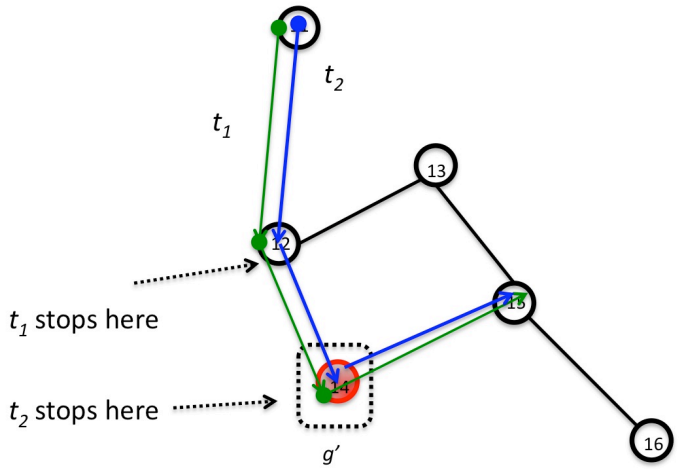


Figure 4.3 Group *g'* with just one vertex and two trajectories.

Figure 4.3 depicts an example generalization using the graph in Figure 3.1, with *p* value of 0.5 and *c* value of 2. The only sensitive vertex is vertex *14* and it is grouped as *g'*. There are two trajectories passing over this node: t_1 *and* t_2. Trajectory t_1 just passes by, but t_2 stops in the group (on vertex *14*). Table 4.1 shows the paths in the c-neighborhood and the ratios of sensitive stops to any stops in the group for each path in the c-neighborhood. Notice that there is a single path in this example. As this path violates *(c,p)-confidentiality* (1.0 > 0.5), the group has to be generalized.

Table 4.1 Equivalence class of group *g'* shown in Figure 4.3.

Paths	Stops at a sensitive node	Stops in group	Ratio
11,12,<14,14>,15	1	1	1

Based on the selection criteria, lets assume that vertex *12* is added to the group to form *g''*, as shown in Figure 4.4. Table 4.2 shows the new paths in the c-neighborhood and corresponding statistics. As t_2 stops on vertex *12* and the route *<14,14>* is eliminated with

the new group layout, *(c,p)-confidentiality* is satisfied for the corresponding path and the group is released.

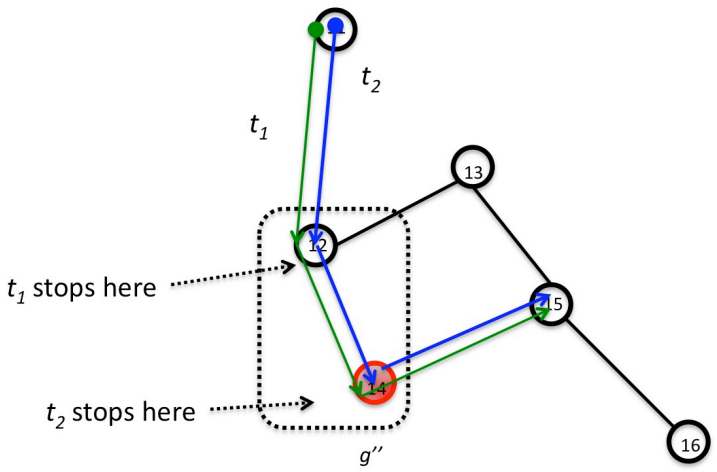


Figure 4.4 Group *g''* after inclusion of vertex *12*.

Table 4.2 Group *g''* after the addition vertex *12*.

Paths	Stops at a sensitive node	Stops in group	Ratio
11,<12,14>,15	1	2	0.5

Input: Group *g, Sensitive Nodes List S, Statistics about user behavior St*
Output: *true if (c,p)-confidentiality* is satisfied, *false* otherwise
Check *cp Confidentiality*()
Begin
boolean isValid = *true*;
For each $eq_i \in Eq$

If (ratio of number of people who are in eq_i and stops in a sensitive node to the number of people who are in eq_i and stops in g) $\geq p$ then

isValid = *false;*
Mark eq_i as a violating equivalence class;

End if

End For

return *isValid;*

End

Figure 4.5 Pseudocode for Checking *(c,p)-confidentiality.*

4.2.2 Node Selection

There are two approaches to select which node to add into the group.

1. Breadth First Search (BFS) Method
2. Violating Routes Method

Breadth First Search Method traverses all nodes in a c-neighborhood using BFS approach, starting from the private node that is considered to be at the centre of the group. The goal is to position the node as the physical centre as well. Figure 4.6 shows the order of nodes to be included into the group in order to satisfy *(c,p)-confidentiality* using BFS approach.

A *violating* route in a group is a route that a path with *probability of* disclosure greater than p uses. The goal of Violating Routes Method is to get rid of routes that are problematic, by adding nodes before the entrance or after the exit. A violating route in the c-neighborhood violates *(c,p)-confidentiality* in the next iteration if not modified. We enclose this route with new entrance and exits that are taken into the group using the paths in the c-neighborhood that use the problematic routes. Lets assume that *p* value is 0.2 and *c* is again 2 for the graph in Figure 4.4. As *p* is not satisfied the next node to be included is either node *11* or node *15* to enclose the route *r* = <*12, 14*>.

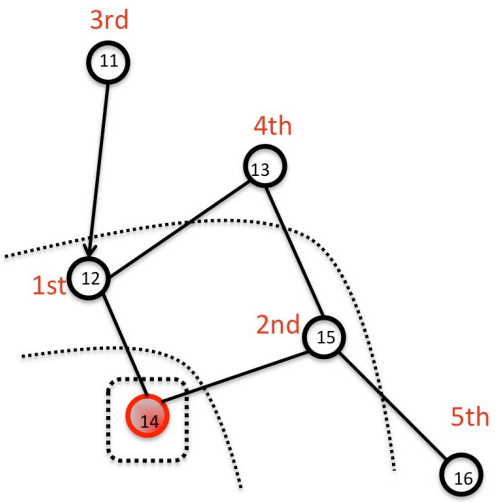


Figure 4.6 BFS vertex inclusion order.

Input: User Stats St
Output: Equivalence classes *Eq*
GetEquivalenceClasses()
Begin
Get portions of trajectories in the neighborhood;
Group portions that follow identical locations in the same order and use same routes when they are in the group;
return all equivalence classes;
End

Figure 4.7 Pseudocode for Equivalence Class generation.

4.3 Suppression Procedure

Suppression mechanism is useful in the following situations:

1. There may be some small sub-graphs that are not connected to the main graph. To be more precise, the algorithm may run out of nodes to be added to the group, before *(c,p)-confidentiality* is satisfied.
2. The group might have included all points in the c-neighborhood but might not have satisfied *(c,p)-confidentiality* yet.
3. The group may grow enormously because of a small number of paths violating *(c,p)-confidentiality*, hammering down utility of the generalized data.

When there is no room for the group to grow, as *(c,p)-confidentiality* is not satisfied, it is compulsory for the algorithm to suppress these violating equivalence classes and corresponding trajectories. Thus, any trajectory taking that path is removed from the released data in the SDR case. This does not mean a privacy violation for the users that are removed because the adversary does not know whether that user was in the raw dataset or not. On the other hand, in the LBS case, any request made in the suppressed routes is not forwarded to the service provider (See Section 4.4.2 for the system behavior in LBS context).

Input: *cutoff_limit*
Output: *true if violation is less than the cutoff_limit, false otherwise*

willBeSuppressed()
Begin

Sum_of_All_Stops = 0;
Sum_of_Sensitive_Stops = 0;

For each $eq_i \in Eq$

Sum_of_All_Stops += nb. of people who are in eq_i and stops in *g;*
Sum_of_Sensitive_Stops += nb. of people who are in eq_i and stops in a sensitive node ;

End For

If p – (Sum_of_Sensitive_Stops / Sum_of_All_Stops) ≤ *cutoff_limit*
or
there is no room for the group to grow
then

If *SDR* then
For each $eq_i \in Eq$ and eq_i is marked as violating;
Remove all $t \in T$ that are in $eq_{i;}$;
End For

End If

return *true;*

End if
return *false;*
End

Figure 4.8 Pseudocode for checking if *Cutoff Limit* is satisfied.

The data owner may want to release a group, which does not satisfy *(c,p)-confidentiality*. A group may violate *(c,p)-confidentiality* but it might be close enough, so that when an acceptable number of paths are suppressed then *(c,p)-confidentiality* is satisfied.

The user-defined range that *(c,p)-confidentiality* can be ensured via suppression is called *Cutoff Limit* (See Figure 4.8). If, the *probability of disclosure of a path* plus *Cutoff Limit*, is smaller than or equal to *p,* then the group is released after all violating routes have been suppressed (all trajectories in the corresponding equivalence class in SDR context).

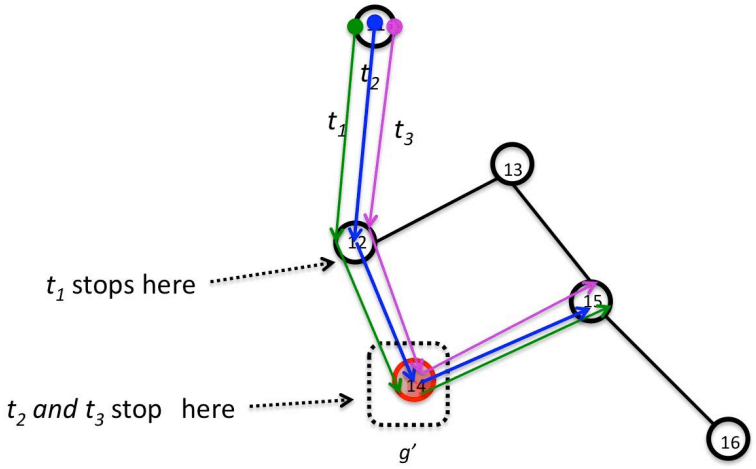


Figure 4.9 Group *g'* with one vertex and three trajectories.

We change the example in Figure 4.3 a little bit and add a new trajectory t_3 that is identical to t_1 to visualize the suppression operation. We use *Cutoff Limit* of 0.2. Considering paths in the c-neighborhood of *g',* we will obtain Table 4.3, which violates *(c,p)-confidentiality* . As, $p = 0.5$ and $0.5 + 0.2 < 1.0$, we cannot suppress the violating paths and need to generalize the group.

Table 4.3 Ratio of sensitive stops to any stops in the group *g'* in Figure 4.9.

Paths	Stop at a sensitive node	Stop in group	Ratio
11,12,<14,14>,15	2	2	1

When vertex *12* is added to the group to form *g'',* Table 4.4 is obtained from the group illustrated in Figure 4.10. This time the violation is rate is 0.66, and $0.5 + 0.2 > 0.66$,

which means the suppression rate is satisfied. The algorithm suppresses this path and all trajectories that exist in the corresponding equivalence class.

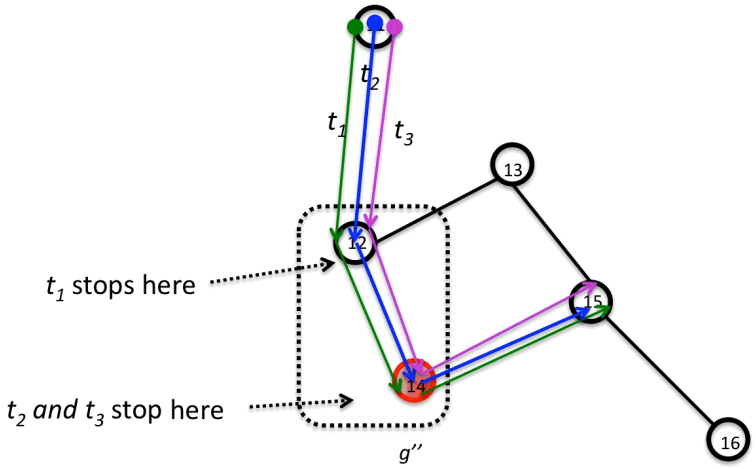


Figure 4.10 Group *g''* with after addition of vertex *12.*

Table 4.4 2 Routes of group *g''* shown in Figure 4.10.

Paths	Stop at a sensitive node	Stop in group	Ratio
11,<12,14>,15	2	3	0.66

4.4 Output

The output of the algorithm is different based on the type of the application. The methodology for SDR is explained in Section 4.4.1 and the methodology for LBS is explained in Section 4.4.2.

4.4.1 SDR Output

As SDR systems work on complete and static data, we anonymize the raw data according to the anonymized graph. Since all trajectories follow nodes in the graph, all the algorithm needs to do is to modify points in trajectories according to generalization performed.

In each trajectory, we search for a series of nodes that correspond to a route in some group, in the generalized map. All the nodes in the series found are removed and replaced by a single node that corresponds to the specific group. The timestamp is set to the timestamp of the first node replaced, which indicates the time of entrance to the group.

The example in Figure 4.12 shows a trajectory following a path on the graph, which is depicted in Figure 3.1. Vertices *12*, *13* and *15* belong to g_1, they are replaced with the group g_1. The timestamp is set to the timestamp of vertex *12* as it is the moment of entering into the group. If the trajectory starts within the group, then the timestamp is set to the timestamp of the first node visited in the group.

The algorithm ensures that for each group, *probability of disclosure* is smaller or equal to the user determined value *p* based on the background information used, which is determined by the parameter *c*.

Input: Graph *G,* Group List *Gr*
Output: *Graph G'*
ModifyDataset()
Begin

For each $t \in T$
For each $g \in Gr$
For each series of points in *t* that are in *g*
Replace the series with a new point *n=[x,y,t]* where
x and *y* represent the centre of *gr* anf *t* is the timestamp
of first node visited in *g;*
End For
End For
End For
return G;

End

Figure 4.11 Pseudocode for modifying the trajectory data in SDR context.

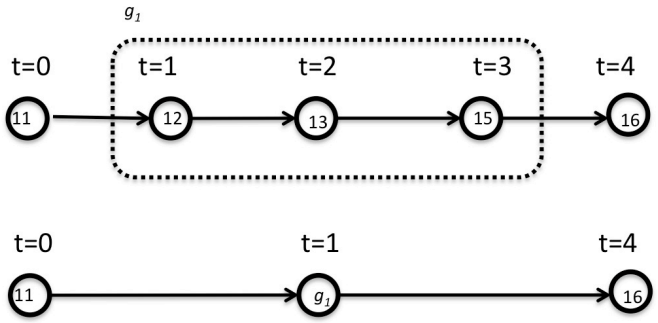


Figure 4.12 Trajectory generalization.

4.4.2 LBS Output

There are two types of requests a LBS provider may receive in the anonymized map: Requests made in a grouped node and requests made in a non-grouped node. The requests that are made in a non-grouped node do not raise any privacy concerns. On the other hand, requests made in a grouped node may be problematic if there are some suppressed paths in the group.

```
Input: Group List Gr
Output:
ReleaseRules()
Begin
        Suppress each request made in a path that is marked;
End
```

Figure 4.13 Pseudocode for LBS rule generation.

The goal of LBS systems is to service a user as the request arrives given the privacy constraints. As LBS provider cannot guess the location of the next request from a specific user, it can only act based on what has been received so far. Consider a user that has followed a portion of a suppressed path in the c-neighborhood of the sensitive node and this portion (which is a path itself) is not suppressed. In this case the algorithm suppresses requests made by this user, although he or she is not in a suppressed path. This is because there is a chance that this user visits the remaining vertices on this path while getting service during the trip.

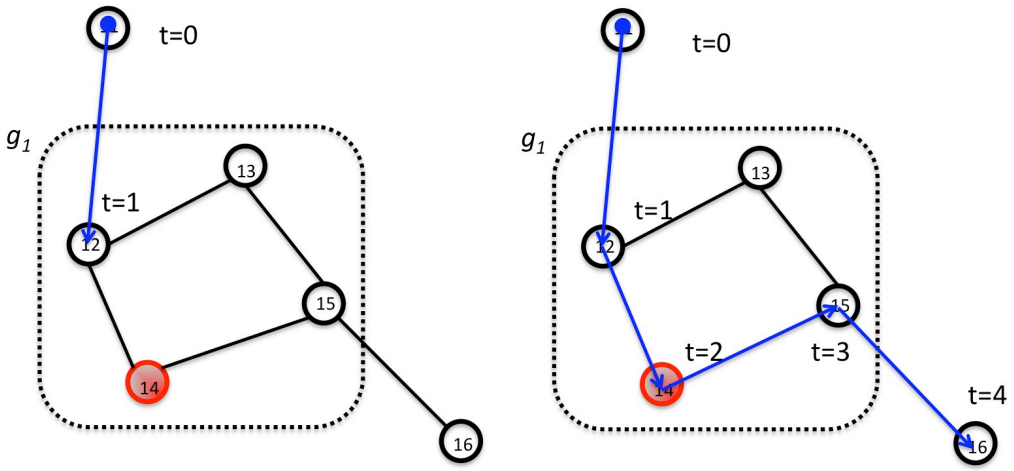


Figure 4.14 LBS suppression example.

The Figure 4.12 depicts an example of a trajectory travelling along the group g_1 where $c = 2$. The trajectory starts at vertex *11*, at time t = 0. It goes forward making a request at each node visited. It enters the group using the vertex *12*, stops there and makes a service request at time t =1. The path *pt* this trajectory taken so far is *pt ={11, <12,null>}* which is not suppressed. This user may go on and exit from vertex *15,* taking the path *pt'={11, <12,15>, 16}*, which is suppressed. Although path *{11, <12,null>}* is not suppressed we need to suppress requests made in this path as the user may exit from vertex *15* and end up in a suppressed path getting service all the way, which is a violation of privacy.

5. PERFORMANCE EVALUATION

We have simulated our approach by coding in Java and using Eclipse as our IDE. The tests were run on a MacBook Pro with Intel Core 2 Duo Processor and 2GB memory.

5.1 Map Structure

We used OpenStreetMap (OSM) [43], as the source of map structure. We have downloaded map of Milano with the following coordinates: Minimum latitude:"45.54665", Minimum Longitude:"9.17288", Maximum latitude:"45.55799", Maximum longitude: "9.19472". We parsed the XML file and extracted vertex and edge information in accordance with our graph definition. There are 4073 nodes and 4321 edges in the graph.

We have drawn the map using Java to visualize the actions of the approach. The dots correspond to the nodes in the map and the lines correspond to the edges that connect the nodes. Figure 5.1 shows the map we used to evaluate our approach.

Figure 5.1 Milano map visualized using Java.

5.2 Trajectory Data

We generated a synthetic dataset of 3000 trajectories. To begin with, we ran Floyd-Warshall all-pairs shortest path algorithm [44] on Milano map, in order to obtain shortest path from any node in the graph to any other node. Then, we picked two random nodes: n_1 and n_2 in the graph. If they are connected (there exists a road from the first node to second node), we picked all the nodes on the shortest path from n_1 *to* n_2 and created a trajectory. Obviously, all trajectories follow the nodes and edges of the graph. The Figure 5.2 shows sample trajectories on the map that are represented by connected empty circles. Sensitive nodes are shown as big standalone dots.

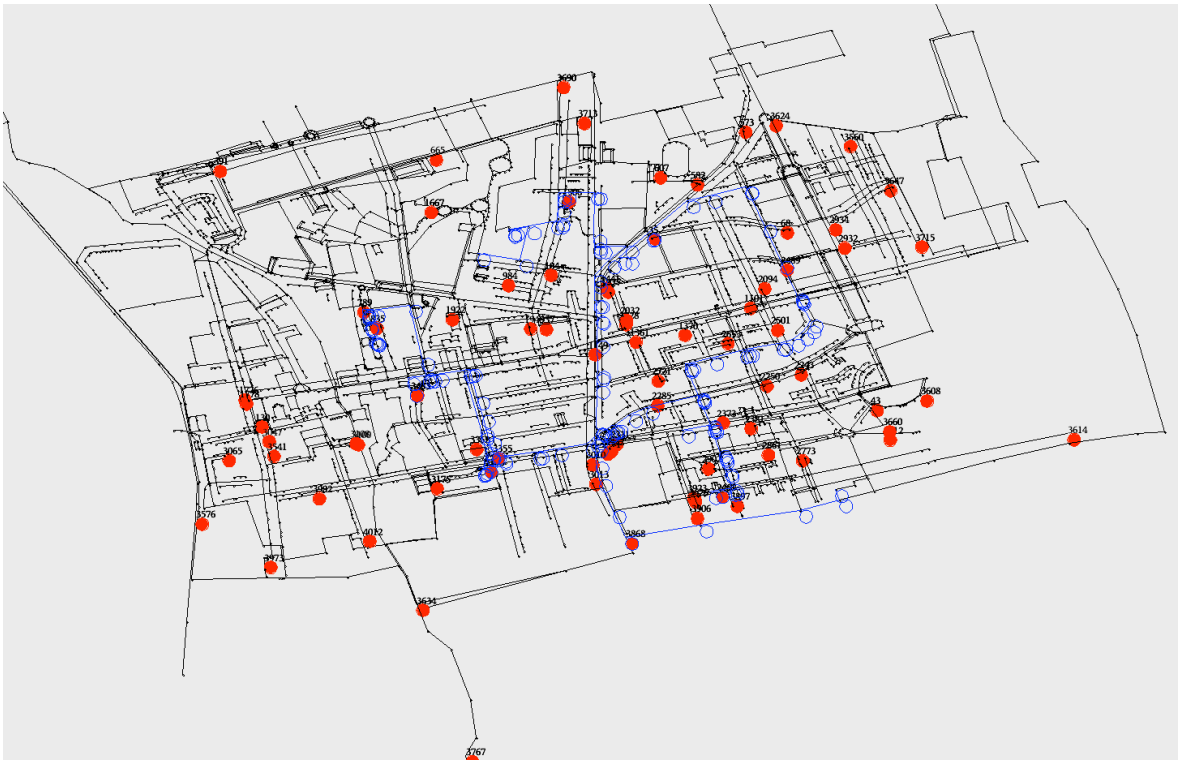


Figure 5.2 Milano map with sensitive nodes and trajectories.

The trajectories are assumed to be stopping on the start and the end points and a trajectory is generated such that it stops on a node with 1/6 probability.

Figure 5.3 A group with sensitive and non-sensitive nodes.

Figure 5.3 shows an example grouping where *vertex 112* is a sensitive node and the other three vertices in the rectangle are grouped. These four nodes are going to act as a super-node in the anonymized map.

5.3 Experiments

To measure the performance of our approach, we randomly picked 36 nodes and marked them as sensitive nodes.

We took three different *p* values: 0.5, 0.33 and 0.25 to see the performance of our approach when the privacy level gets stricter. We used *c* values: 3, 6 and 9 to see the effect of an increase in the background knowledge of the adversary. Finally, we have used two *Cutoff Limit* values: 0.1 and 0.05

5.3.1 Effect of the values of *c, p* and *Cutoff Limit*

As explained in Section 3.2, there are three parameters in our algorithm: *c, p* and the *Cutoff Limit.* In this set of experiments, we show the effect of a change in one of these parameters using the *violating routes* node selection approach.

Parameter *p* adjusts the desired level of privacy in the resulting anonymization. When we reduce *p*, privacy protection of the individuals increase as percentage of the number of people taking a path decrease. Parameter *c* determines the level of background knowledge of the adversary. As *c* increases, the neighborhood of the target node enlarges and adversary becomes more powerful as he or she is capable of considering far away nodes. In Figure 5.4, we show the effect of the change in parameters *c*, *p* and *Cutoff Limit* on the average group size, which is the utility metric we use.

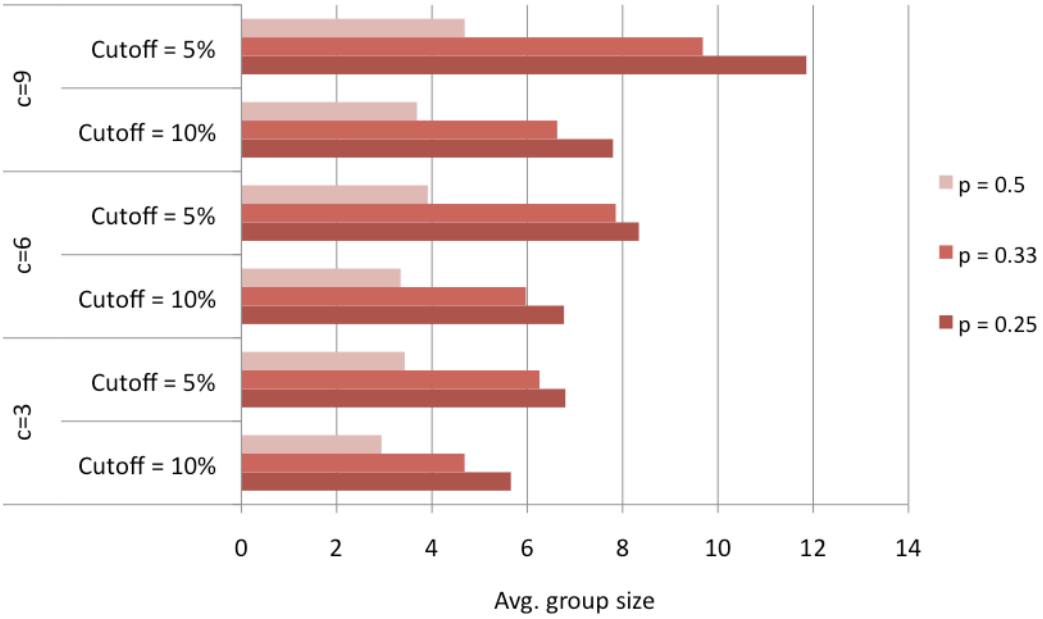


Figure 5.4 Average group sizes for the parameters *c, p* and *Cutoff Limit.*

In all cases a decrease in p results in an increase in the average group size. Notice that, p=0.5 means at most 1 out of 2 trajectories on a path that stop in the group, can stop on a sensitive node, whereas $p = 0.2$ means that at most 1 out of 5 trajectories that stop in the group taking a path, can stop on a sensitive node, which is more restrictive.

Similarly, increase in c results in an increase on the average group size in all cases. As the c-neighborhood is widened, number of equivalence classes increase. Thus the algorithm needs to consider more options and needs to add more nodes into the group.

Finally, as the *Cutoff Limit* increases, the algorithm is able to stop the operation earlier and can suppress more trajectories. Therefore the average group size decreases. In the extreme case when *Cutoff Limit* is 100, the algorithm needs to satisfy *(c,p)-confidentiality* for all paths in the c-neighborhood and is not allowed to make a shortcut.

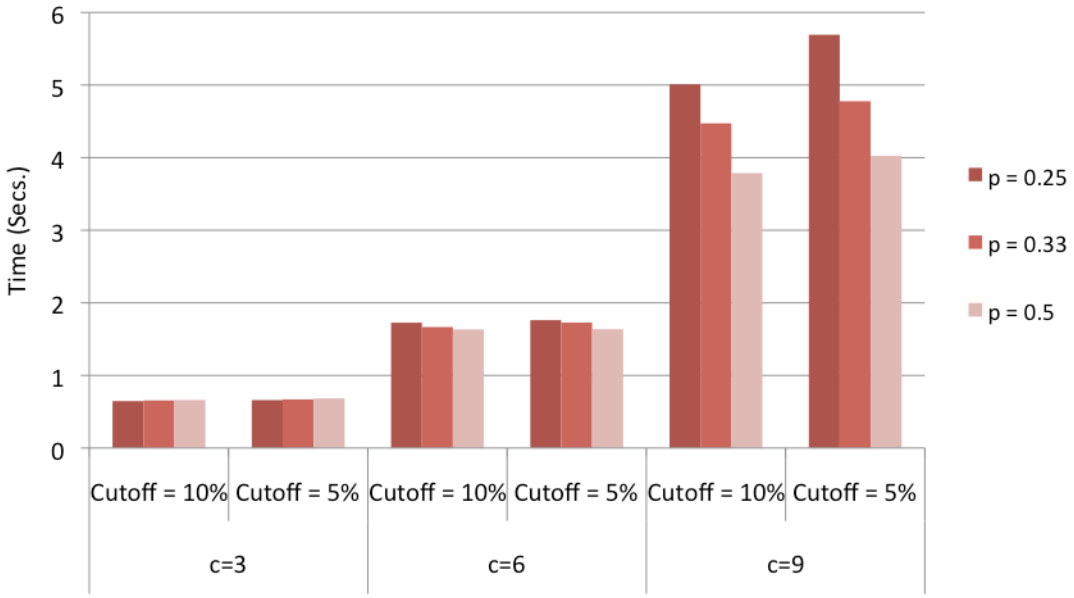


Figure 5.5 Time performance for the parameters *c, p* and *Cutoff Limit.*

Figure 5.5, shows the time performance of the algorithm given the parameters. Increase in *Cutoff Limit* result in a decrease in the time required because the algorithm is more free to suppress trajectories rather than adding vertices to the group and spending time. The difference becomes more obvious when both *c* and *p* increase.

The increase in background knowledge of the adversary clearly forces the algorithm to spend more time to satisfy *(c,p)-confidentiality*. The bigger the c-neighborhood is, the bigger the number of paths to be considered and equivalence classes to be formed. As this number increases, time requirement of the algorithm increases as well. This makes sense, as we are facing a thougher adversary and we need more resources to satisfy the privacy requirement of an individual.

Similary as *p* gets stricter, the algorithm needs more time. When *p* is smaller, the number of paths that violate *(c,p)-confidentiality* increases and the algorithm has to add more nodes to the group which takes more time. This again makes sense as the data owner is more conservative and desires a higher level of privacy. Again we need more time to satisfy the needs of the user.

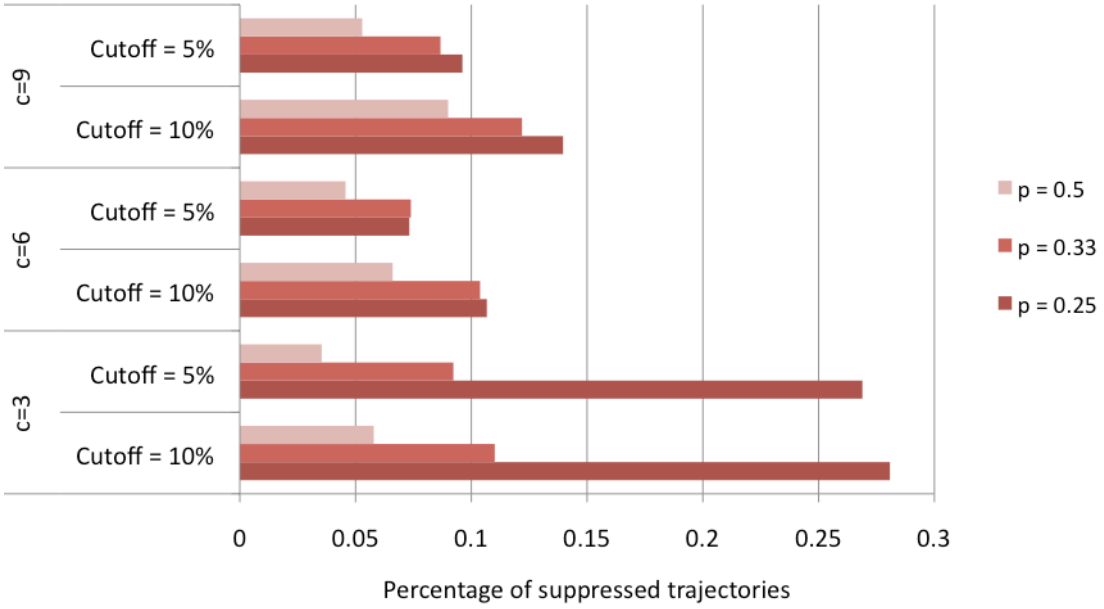


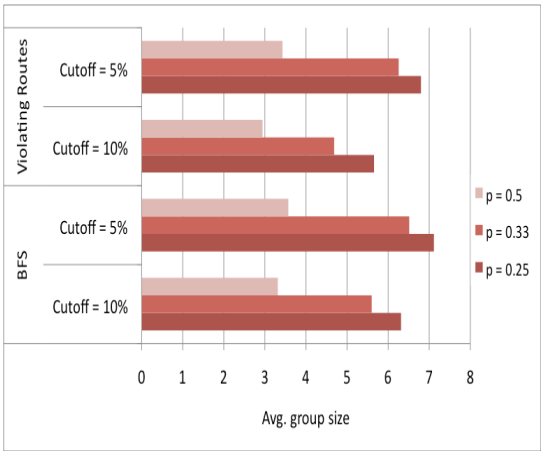
Figure 5.6 Suppression rate comparison for the parameters *c, p* and *Cutoff Limit.*

Figure 5.6 shows the percentage of the number trajectories that are suppressed to the number of all trajectories that enter the group. This is an indication of the utility achieved. As the *Cutoff Limit* increases, the algorithm is allowed to suppress more routes and hence the suppression rate increases. Similarly when *p* is stricter, *(c,p)-confidentiality* is satisfied by suppressing more trajectories.

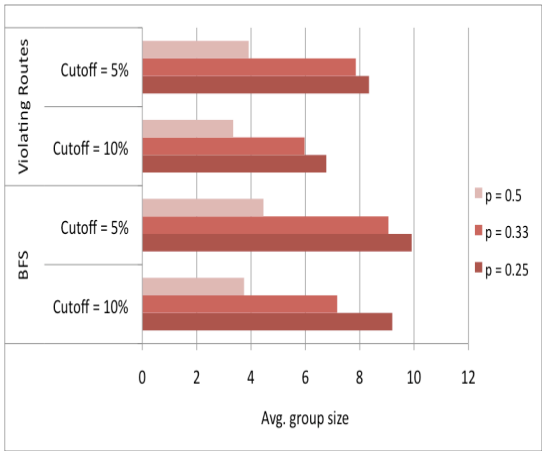
There is no clear relation between the value of *c* and the percentage of the trajectories suppressed because *c* affect this value both negatively anf positively. Increase in *c,* broadens the group and decreases the suppression amount required. On the other hand, the increase in the number of paths, forces algorithm to suppress more trajectories.

5.3.2 BFS vs. Violating Routes

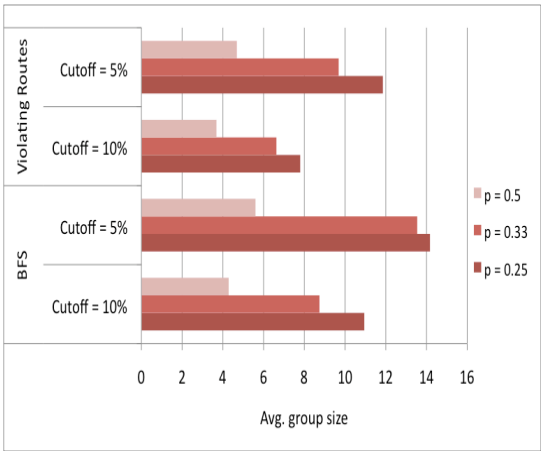
We compare the performances of two node selection approaches that have been discussed in Section 4.2.2.



c = 3



c=6

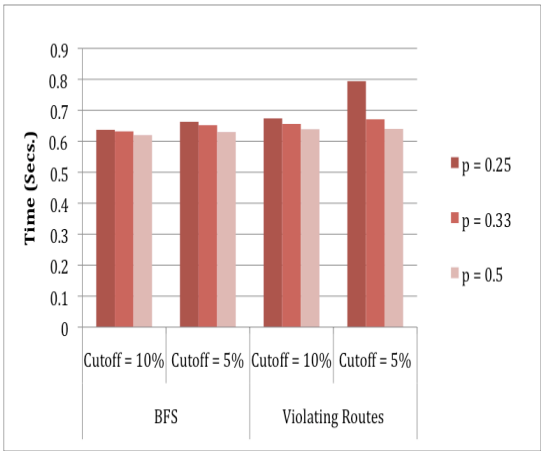


c =9

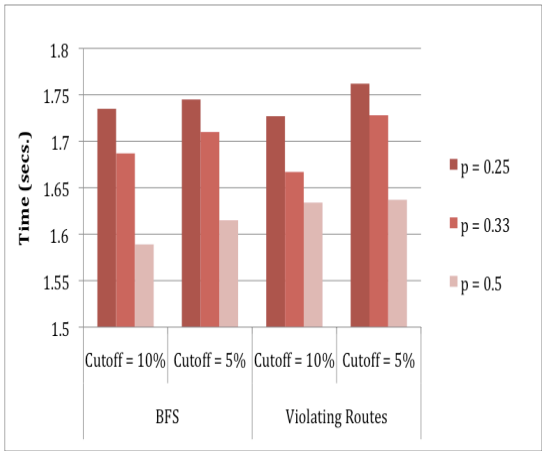
Figure 5.7 Average group size comparison of BFS and Violating Routes approaches.

Figure 5.7 shows average group sizes for each *c* value. The average group size is shown against the corresponding *p* value and *Cutoff Limit.* In all cases, *Violating Routes* approach performs better than BFS approach as it satisfies *(c,p)-confidentiality* using fewer nodes, which preserves utility. This makes sense, since *Violating Routes* approach works on the problematic routes, whereas BFS approach only tries to surround the sensitive node with non-sensitive nodes geographically.

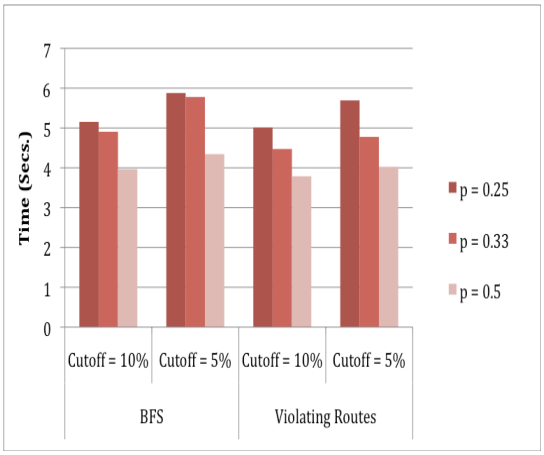
Increases in *c* and *p* values, increase the group sizes as explained before, but it also widens gap between performances of two approaches more significant.



c = 3



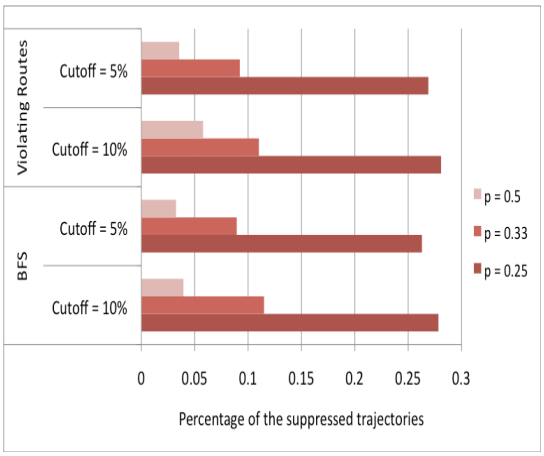
c=6



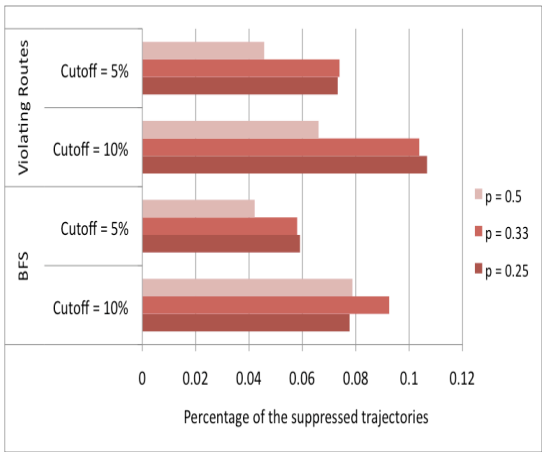
c =9

Figure 5.8 Time performance comparison of BFS and Violating Routes approaches.

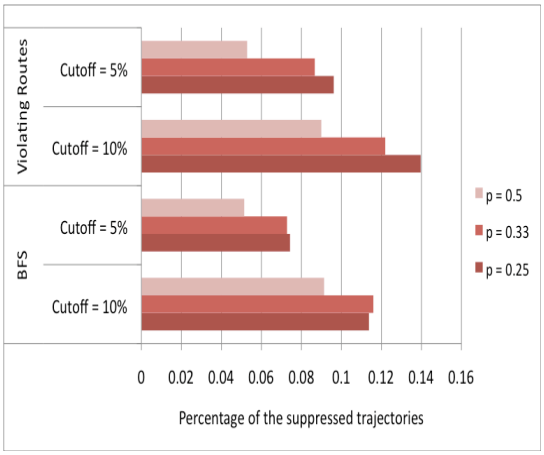
Figure 5.8 shows the time performances of two approaches. When the width of the c-neighborhood is small, BFS approach performs better than *Violating Routes* approach. This is because of the fact that the group can satisfy *(c,p)-confidentiality* easily. BFS can find the node that *Violating Routes* would choose as the number of candidates is small. When the c-neighborhood grows, the number of candidates to choose from grows as well. Although BFS makes a decision within a small period of time, *Violating Routes* makes correct choices and uses less time overall, by adding a smaller number of nodes into the group as shown in Figure 5.8.



c = 3



c=6



c =9

Figure 5.9 Suppression rate comparison of BFS and Violating Routes.

Figure 5.9 shows the suppression rate comparison of BFS and *Violating Routes* approaches. In almost all cases BFS approach seems to suppress fewer trajectories. As BFS adds more vertices into the group, it needs less suppression, but the difference is not so significant.

When we consider the results so far, node selection method represents a trade-off between suppression and generalization. If BFS is chosen, then generalization (average group size) rate increases and if *Violating Routes* is chosen, then the suppression rate increases. However we are going to use *Violating Routes* approach in the rest of the experiments as the main utility metric we use is the average group size.

5.3.3 *(c,p)-confidentiality* vs. *k*-anonymity

We have mentioned the inability of *k*-anonymity to satisfy the diversity of the visited sensitive places throughout the thesis. We seek the answer to the question: "Is *k*-anonymity able to satisfy *(c,p)-confidentiality*?" and compare the performances of the two methods: *(c,p)-confidentiality* and *k*-anonymity.

First, we give a definition of *k*-anonymity for spatio-temporal data. The methods used in [37] and [38] offer *k*-anonymity via clustering trajectories. As mentioned earlier, our approach does not need to perturb trajectories that move on non-sensitive areas. Thus distortion rate is smaller. Hence, it is unfair to compare our approach against such methods. Rather we adopt a definiton similar to [21].

Definition 5.1 - A group g_1 is *k*-anonymous if for each path in g_1^c that enters g_1, there exists none or there exists at least *k* trajectories that stop in the group g_1 taking that path.

$$\forall\, eq_i \in Eq, \text{ and corresponding } pt_i \text{ that enters the group } g$$
$$|eq_i| = 0 \text{ or nb. of } tr_z \in eq_i \text{ s.t. } tr_z \text{ stop in } g \geq k$$

Notice that in [21], authors require *k* users to perform requests in the region to satisfy *k*-anonymity (*k* users that do not make requests is not enough). Similarly *k* users that exist in the group is not enough in our approach, and we need *k* users that stop in the group.

We use the *c* and *p* values selected in Section 5.3.1 and test them against *k* values of 2, 4, 6 and 8. Figure 5.10 shows the average group size needed to achieve *(c,p)-confidentiality* and *k*-anonymity for given parameters *c,p* and *Cutoff Limit.*

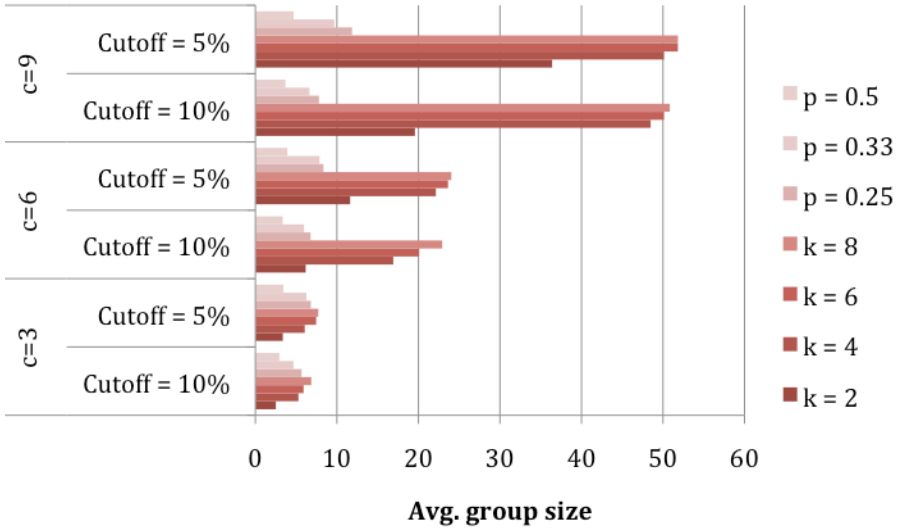


Figure 5.10 Average group size comparison of *(c,p)-confidentiality* vs. *k*-anonymity.

The required average group size increases exponentially for *k*-anonymity. It fills up its quota of nodes that can be added into the group, which is restricted by c-neighborhoods of the groups. For $c = 9$, the average group size stays the same while *k* increases. This is because there is no available node to be added into the group within the c-neighborhood and algorithm suppresses the violating paths. This effect can be seen in Figure 5.11, which shows the suppression rates. However, the required number of nodes for *(c,p)-confidentiality* seems to increase linearly and average group sizes are so small compared to the requirement of *k*-anonymity.

We show the suppression rates of *(c,p)-confidentiality* and *k*-anonymity for the given parameters in Figure 5.11. Obviously *k*-anonymity suppresses a lot more than *(c,p)-confidentiality*. Especially when c-neighborhood is wide and *Cutoff Limit* is low, *k*-anonymity needs to suppress more than 60% of the trajectories, whereas the suppression rate of *(c,p)-confidentiality* does not exceed 30% in the extreme case.

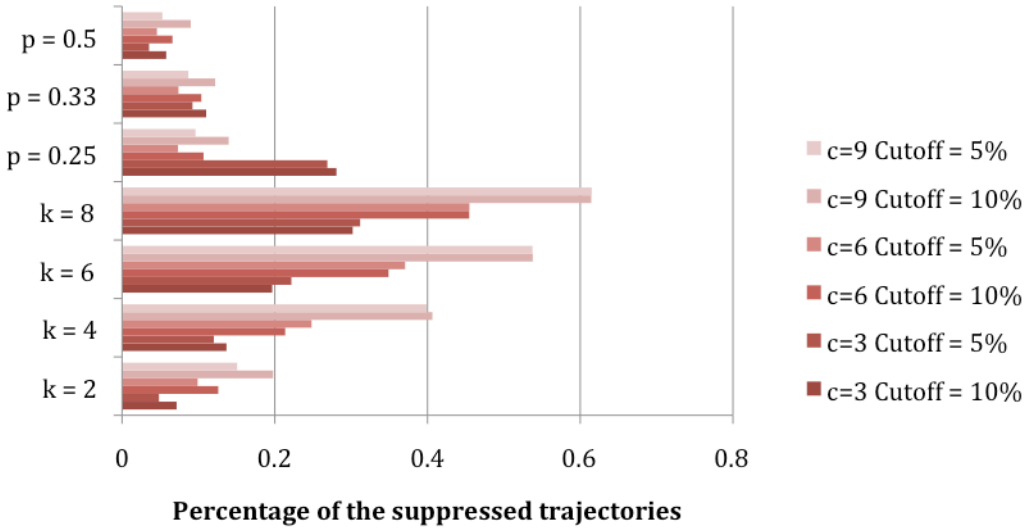
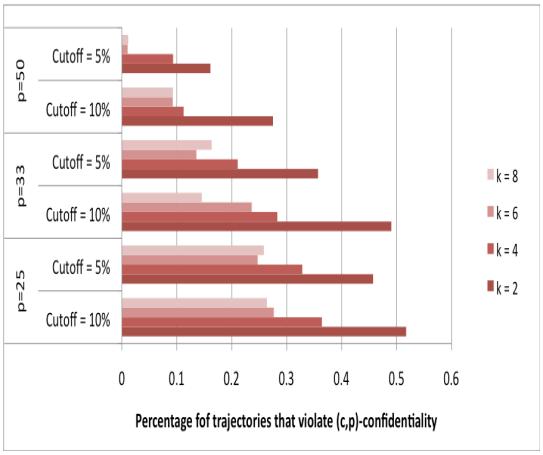
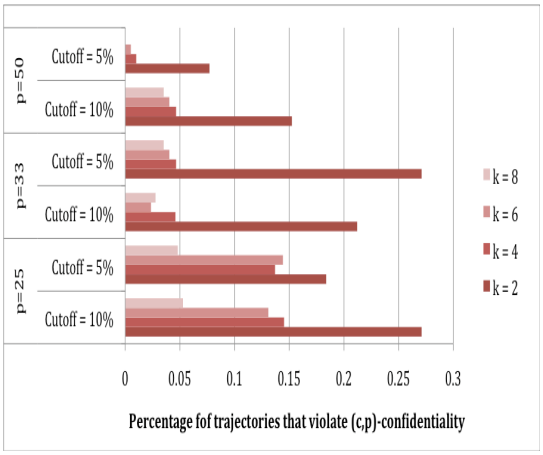


Figure 5.11 Suppression comparison of *(c,p)-confidentiality* vs. *k*-anonymity.

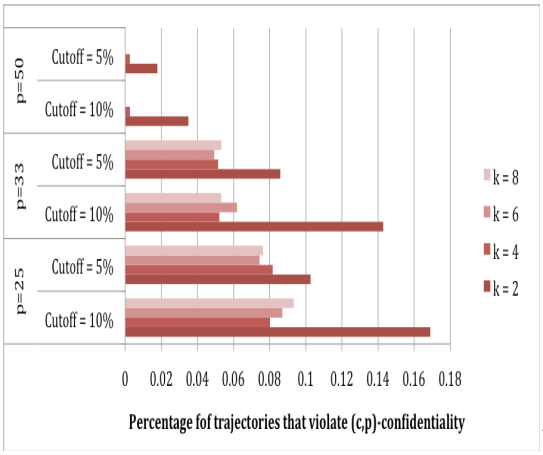
Figure 5.12 shows the results of the tests where we check if *k*-anonymous sets satisfy *(c,p)-confidentiality*. This is going to show the value of *k* that is needed to satisfy the requirement of given *c* and *p*. We show the percentage of trajectories that stop in the group and violate *(c,p)-confidentiality* . This percentage is used as the measure of closeness to satisfy *(c,p)-confidentiality*. The figure clearly depicts that *k*-anonymity is not able to satisfy *(c,p)-confidentiality*.



c = 3



c=6



,

c =9

Figure 5.12 Percentage of trajectories that violate *(c,p)-confidentiality* in *k*-anonymous *sets.*

Given loose values of *c*, *p and Cutoff Limit,* as *k* increases*, k*-anonymity gets close to satisfying *(c,p)-confidentiality*. For instance, when *c = 9, p = 50* and *Cutoff Limit = 0.05*, *10-anonymity* satisfies *(3, 50)-confidentiality*. However, the group size of 55 and suppression rate of 60% to satisfy the constraint makes it infeasible to use *k*-anonymity for location diversity because the data is distorted so much. Needless to say, the decrease in *p* results in more violation as it is harder to satisfy *(c,p)-confidentiality* with a stricter level of privacy.

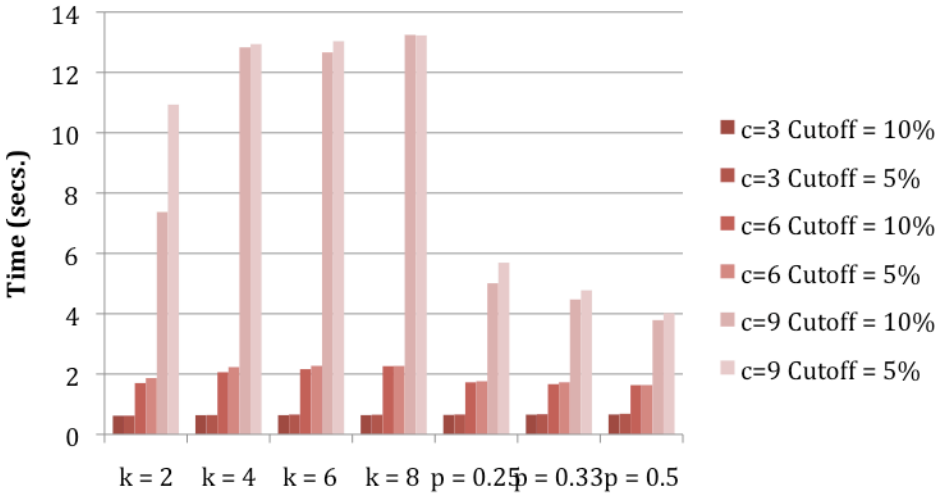


Figure 5.13 Time comparison of *(c,p)-confidentiality* and *k*-anonymity.

Figure 5.13 compares time performances of *k*-anonymity and *(c,p)-confidentiality*. Both approaches require more time as their constraints are restricted. As *k*-anonymity gets close to the upper bound of nodes, its time requirement stabilizes. Nevertheless, time requirement of *(c,p)-confidentiality* increases linearly, just as the average group size.

All tests in this section show that *k-anonymity* is insufficient to provide location diversity. On the other hand, *(c,p)-confidentiality* has proven to be successful in ensuring location diversity within very small time periods and small average group sizes.

6. CONCLUSIONS AND FUTURE WORK

In this thesis, we point out to the importance of diversity, in privacy preserving spatio-temporal data mining. We show that existing privacy preservation mechanisms such as *k*-anonymity, fail to capture information leaks mentioned in Chapter 1. We propose an approach called *(c,p)-confidentiality,* that protects the privacy of the users by diversification of locations.

We have make a binary distinction of locations. Nodes are either sensitive or non-sensitive. This is for the sake of simplicity and this distinction can be extended without losing generality (such as sensitivity rates). We limit the probability of users that stop on a sensitive node with probability *p.* We model the map as a graph and generalized the nodes to form super-nodes which satisfy *(c,p)-confidentiality*. For the outlier trajectories that violate *(c,p)-confidentiality,* we use suppression method.

We consider paths taken by the users in a c-neighborhood as equivalence classes. The c-neighborhoods are determined by the parameter *c*. We provide a block-box abstraction for the movements of trajectories inside a group. Only thing known for a trajectory moving inside a group is its route that corresponds to, the entrance and exit vertices within the group.

We use two node selection procedures, to decide on which node to include in the group while generalizing. The method that tries to enclose problematic routes has performed better than the BFS approach which tries to geographically centralize the sensitive node in the group.

Focusing on anonymizing the map, instead of the trajectories gives us the advantage of distorting just the sensitive portions of the trajectory in SDR context, rather than

perturbing all clustered trajectories like in [36]. Also, focusing on the map proves our algorithm to be independent of the density of trajectories in the dataset. As works in [29] and [30] tries to perturb/suppress trajectories that are in close proximity to each other, they fail in low-density areas unlike our approach.

We point out to the infeasible generalizations released by *k*-anonymity that do not consider geographical structure and the inherent lack of diversity of the *k*-anonymous groups. Then, we tested our approach against *k*-anonymity with a similar definition to [21] and the results have shown that *k*-anonymity alone is insufficient to provide *(c,p)-confidentiality* and causes information leaks. In terms of utility and time performance, *(c,p)-confidentiality* is more efficient. It generates smaller groups that satisfy *(c,p)-confidentiality* within negligible time periods. Thus, it keeps utility high and computation costs very low. The trajectory trash rate supports these results as well. Another advantage of our algorithm is that, it is minimality attack resistant when the statistical background information used is independent of the anonymized data. This ensures that, the algorithm's choices are independent of the data anonymized. Hence, it does not leak information to the adversary in this manner.

Finally, our approach is adaptable to both Static Data Release context and Location Based Services, although the literature so far is split into two discrete areas. We anonymize the released data with the groups to be released, and we release rules for the LBS systems to restrict the places where a user can be provided with service.

Future direction of this work can be focused on LBS context. Current state of the work suppresses each request made in a path that has the chance to end up in a suppressed path, even if it is not in a suppressed path for the time being. This method over suppresses requests and decreases utility. Some probabilistic methods that can limit the probability of ending in a suppressed path may have been used to overcome this problem.

We also plan to enhance the adversary background knowledge definition in the future and face with a tougher version, in which the adversary is capable of calculating the

probability of disclosure for the c-neighborhood of each node within the c-neighborhood of the target node in the graph, whereas our current approach is target node centered. In this case, the approach will have to consider more constraints to protect the privacy of individuals, but is going to be more protective.

REFERENCES

[1] P. Samarati. Protecting respondents' identities in microdata release. In IEEE Transactions on Knowledge and Data Engineering, 2001.

[2] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, CMU, SRI, 1998.

[3] L. Sweeney. Uniqueness of simple demographics in the U.S. population. Technical report, Carnegie Mellon University, 2000.

[4] L. Sweeney. k-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5):557–570, 2002.

[5] Willenborg, L and DeWaal, T. Statistical Disclosure Control in Practice. Springer-Verlag, 1996.

[6] B. C. M. Fung, K. Wang, P. S. Yu, "Top-Down Specialization for Information and Privacy Preservation", Proceedings of the 21st International Conference on Data Engineering, 2005.

[7] V. Iyengar. Transforming data to satisfy privacy constraints. In ACM Special Interest Group on Knowledge Discovery and Data Mining, 2002.

[8] W. Winkler. Using simulated annealing for k-anonymity. Research Report 2002-07, US Census Bureau Statistical Research Division, 2002.

[9] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In Proc. of the ACM Symp. on Principles of Database Systems, June 2004.

[10] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Anonymizing tables. In International Conference on Database Theory, pages 246–258, 2005.

[11] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity." in *Proc. of the 22nd IEEE Int. Conf. on Data Engineering 2006*.

[12] K. LeFevre, D.DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *ACM SIGMOD*, 2005.

[13] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In Proceedings of the 22nd IEEE International Conference on Data Engineering, 2006.

[14] T. M. Truta, B. Vinay, "Privacy Protection: p-Sensitive k-anonymity Property", Proceedings of the 22nd IEEE International Conference on Data Engineering, 2006.

[15] N. Li and T. Li. t-closeness: Privacy beyond k-anonymity and l-diversity. In Proceedings of the 23nd International Conference on Data Engineering (ICDE '07), Istanbul, Turkey, Apr. 16-20 2007.

[17] R. C.-W. Wong, A. W.-C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In VLDB '07: Proceedings of the 33rd International conference on Very large data bases, pages 543–554. VLDB Endowment, 2007.

[18] R. C.-W. Wong, J. Li, A. W.-C. Fu, and K. Wang. (α, k)-anonymity: An enhanced k-anonymity model for privacy-preserving data publishing. In Proceedings of ACM KDD, Philadelphia, Pennsylvania, USA, August 20-23 2006.

[19] M. E. Nergiz, M. Atzori, and C. Clifton. Hiding the presence of individuals in shared databases. In SIGMOD'07: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, Beijing, China, June 11-14 2007.

[20] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In Proceedings of the 1st International Conference on Mobile Systems, Applications, and Services, 2003.

[21] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In The 25th International Conference on Distributed Computing Systems (ICDCS'05), 2005.

[22] C. Bettini, X. S. Wang, and S. Jajodia. Protecting privacy against location-based personal identification. In Secure Data Management, pages 185–199, 2005.

[23] G. Ghinita, P. Kalnis, and S. Skiadopoulos. PRIVE: Anonymous Location-based Queries in Distributed Mobile Systems. In *Proceedings of International Conference on World Wide Web*, pages 371–380, 2007.

[24] M. F. Mokbel, C. Y. Chow, andW. G. Aref. The New Casper: Query Processing for Location Services without Compromising Privacy. In *Proceeding of Very Large Databases*, VLDB 2006.

[25] V. S. V. Aris Gkoulalas-Divanis. A free terrain model for trajectory k-anonymity. In 19th International Conference on Database and Expert Systems Applications - DEXA '08, pages 49–56, 2008.

[26] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar. Preserving user location privacy in mobile data management infrastructures. In 6th Workshop Privacy Enhancing Technology Workshop, pages 393–412. Springer, 2006.

[27] M. Duckham and L. Kulik. A formal model of obfuscation and negotiation for location privacy. In Pervasive, pages 152–170, 2005.

[28] M. Gruteser and X. Liu. Protecting privacy in continuous location-tracking applications. IEEE Security and Privacy, 02(2):28–34, 2004.

[29] A. R. Beresford and F. Stajano. Location Privacy in Pervasive Computing. IEEE Pervasive Computing, 2(1):46–55, 2003.

[30] B. Hoh and M. Gruteser. Protecting location privacy through path confusion. In SECURECOMM ’05: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM’05), pages 194–205, Washington, DC, USA, 2005. IEEE Computer Society.

[31] J. Domingo-Ferrer and J. M. Mateo-Sanz, “Practical data-oriented microaggregation for statistical disclosure control.” *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 1, pp. 189–201, 2002.

[32] N. Anwar, “Microaggregation – Small Aggregates Method”, Internal Report, Luxemburg: Eurostat, 1993.

[33] D. Defays, P. Nanopoulos, “Panels of Enterprises and Confidentiality: Small Aggregates Method”. Proceedings 1992 Symposium Design and Analysis of Longitudinal Surveys, pp. 69-78, 1995.

[34] D. Defays, N. Anwar, “Microaggregation: A Generic Method”, Proceedings of Second International Symposium on Statistical Confidentiality, pp. 69-78, 1995.

[35] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu, "Achieving anonymity via clustering." in *Proceedings of the 25th ACM Symposium on Principles of Database Systems, PODS 2006*.

[36] J. W. Byun, A. Kamra, E. Bertino, and N. Li, "Efficient k-anonymization using clustering techniques." in *Proc. of the 12th International Conference of Database Systems for Advanced Applications, DASFAA 2007*.

[37] F. Bonchi, O. Abul, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In Proceedings of the 24nd International Conference on Data Engineering, ICDE 2008, Cancun, Mexico, Apr. 7 2008.

[38] Mehmet Ercan Nergiz, Maurizio Atzori, Yucel Saygin, Baris Guc. Towards Trajectory Anonymization: a Generalization-Based Approach. Transactions on Data Privacy, Vol. 2, No. 1, 2009.

[39] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Preserving privacy in GPS traces via density-aware path cloaking. In ACM Conference on Computer and Communications Security (CCS), VA, USA, Oct. 29 2007.

[40] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories, 9th International Conference on Mobile Data Management 2008, MDM 2008, pages 65–72, April 2008.

[41] T. Dalenius, "Finding a Needle in a Haystack or Identifying Anonymous Census Record", Journal of Official Statistics, 2(3), 329-336, 1986.

[42] K. Wang, P. S. Yu, and S. Chakraborty. "Bottom-up generalization: A Data Mining solution to privacy protection", IEEE International Conference on Data Mining 2004, pages 249-256.

[43] OpenStreetMap (OSM). July 1, 2004. Retrieved May 1, 2009, from OpenStreetMap Website: http://www.openstreetmap.org.

[44] Cormen, T. H., Leiserson C. E., Rivest R. L., Stein C. (1990). *Introduction to Algorithms.* MIT Press and McGraw-Hill.