2009

Anonymity-preserving location data publishing

Girish Lingappa
Iowa State University

Anonymity-preserving location data publishing

by

Girish Lingappa

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Ying Cai, Major Professor
Johnny Wong
Zhao Zhang

Iowa State University

Ames, Iowa

2009

DEDICATION

To my family and friends for all the affection and support.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this thesis. First and foremost, Dr. Ying Cai for his guidance, patience and support throughout this research and the writing of this thesis. His insights and words of encouragement have often inspired me and renewed my hopes for completing my graduate education. I would also like to thank my committee members for their efforts and contributions to this work.

ABSTRACT

Advances in wireless communication and positioning technology have made possible the identification of a user's location and hence collect large volumes of personal location data. While such data are useful to many organizations, making them publicly accessible is generally prohibited because location data may imply sensitive private information. This thesis investigates the challenges inherent in publishing location data while preserving location privacy of data subjects. Since location data itself may lead to subject re-identification, simply removing user identity from location data is not sufficient for anonymity preservation, and other measures must be employed. We provide a literature survey and discuss limitations of related work on this problem. We then propose a novel location depersonalization technique that produces efficient depersonalization of large volumes of location data. The proposed technique is evaluated using simulation. Our study shows that it is possible to guarantee a desired level of anonymity protection while allowing accurate location data to be published.

CHAPTER 1. INTRODUCTION

Today's wireless communication and positioning technology allows wireless service providers to fairly precisely identify geographic location of their users. This capability not only makes it possible to comply with federal E-911 regulations, but also enables service providers to collect large volumes of personal location data. Such data reveals patterns of city population dynamics and therefore can be used in many important applications, including transportation scheduling (1), enterprise and urban planning (2), social interaction and community studies (3), and emergency response (4), just to name a few. While large collections of location data are of great value to a variety of organizations, making them publicly accessible is generally prohibited, since a person's whereabouts may reveal sensitive private information. For example, frequent visits to certain types of locations may be linked directly to one's health condition, lifestyle, and political associations. In particular, unlike other personal data posted on the Internet, location information has the potential to allow an adversary to physically locate a user. Clearly, exposing such information can present significant privacy and security threats to individuals.

This thesis investigates the challenges of publishing location data while preserving the anonymity of data subjects. It may first appear that one can simply replace user identities with randomly generated pseudonyms, but using pseudonyms, or even not using any stated identity at all, is not sufficient for anonymity protection. This is due to the fact that location information by itself can possibly reveal a user's real-world identity. For example, if a location specifies a private address, the subject is likely to be the owner of that address. It may be difficult to link an individual location sample to a subject, but a significant accumulation of location data will eventually reveal a user's true identity. This has been confirmed in a number

of experimental studies by different research groups ((5), (6)).

For anonymity protection, location data must be *depersonalized* before they can be published. The problem of location depersonalization has been investigated in a series of research work ((7), (8), (9), (10), (11), (12)) to support anonymous uses of location-based services (LBS). The basic idea is to reduce location resolution to prevent service providers from deriving their users' true identity based on the location information they submitted in their requests for services. Specifically, when a client node requests LBS, it reports the current position to an anonymity server; the server then computes a *cloaking* box that contains the client node and at least $K - 1$ other mobile nodes. This box is then reported as the client's location to the LBS request. Since any entity inside the cloaking box could be the one that requests the service, this strategy effectively provides K-anonymity protection to the service user.

Authors in (8) present a novel cloaking algorithm that is able to compute a minimal cloaking box with a given anonymity requirement. Making a cloaking box as small as possible, without comprise of anonymity protection, is critical to ensure that the published data remains informative and statistically usable. Until now, (9) is the only source that considers minimizing cloaking resolution. However, the proposed *CliqueCloaking* algorithm must compute a clique graph and therefore is NP-hard, making it impractical for depersonalizing a large set of location data. In contrast, the complexity of the algorithm proposed in (8) is polynomial time.

The existing techniques fall short when applied to publishing large volumes of location data. They are all designed for anonymous uses of LBS. Since a user's location needs to be cloaked only when requested by LBS, these techniques individually depersonalize location samples one at a time. In the case of depersonalizing large volumes of location data, significant savings on disk I/O and CPU time can be achieved through batch processing. Therefore, new algorithms should be designed for efficient location data publishing, and to our knowledge such work has not been described in the literature. The contributions of this thesis are as follows: We propose to periodically publish location data which can be easily collected by wireless service providers. The collected data is periodically refreshed, maintaining a historical

database of location samples. We develop a scalable batch-processing algorithm to efficiently de-personalize large volumes of such location data. Our simulation shows that, by using batch processing, disk I/O and CPU time can be reduced by more than 50% as compared to the process of cloaking location samples one at a time.

The remainder of this thesis is organized as follows. In Chapter 2, we discuss existing work on location depersonalization in more detail. In Chapter 3, we present a system overview and goals, and propose our cloaking algorithms. The performance of the proposed techniques is evaluated in Chapter 4. Finally, we present conclusions in Chapter 5.

CHAPTER 2. RELATED WORK

There has been an exponential growth in the number and variety of data collection activities related to user-specific information (13). Data holders, such as hospitals or banks, must release this information for external access or else there is no practical use of such data. However, there are significant privacy concerns associated with release of such data. Existing works as described in (14) discuss the privacy act and the need for legitimate data access. There have been studies to demonstrate the possibility of enhancing access to federated data while preserving confidentiality (15). These works emphasize the need for privacy protection of users, while making use of their personal data. Researchers have widely discussed the concept of anonymity (16) for protecting user privacy.

2.1 Traditional Data Anonymity

The need for public data access and associated privacy problems was originally studied for relational data, such as occurs in the release of patient records by hospitals. The Datafly system (17) was created to guarantee anonymity while sharing medical data. Sweeney et al ((13), (18)) studied a formal protection model named K-anonymity to ensure privacy protection for publicly-accessed person-specific data. The idea is to release data such that every record is indistinguishable among at least K-1 other individuals whose records also appear in the released data. The problem of privacy protection and preservation of anonymity has also been studied in the realm of internet communication. Researchers have addressed the need for anonymous communication over the internet by creating a protocol called Hordes (19). An anonymous email communication over an unsecured network using pseudonyms was proposed by Chaum (20).

2.2 Location Anonymity

Privacy protection schemes for location data can be broadly classified into regulatory (21), policy-based, or anonymity-based. Location-based applications initially adopted a policy-based approach for privacy protection (22). In this approach, users must evaluate the policies offered by the provider and trust in a contractual agreement that the data collected by the providers is protected. Protecting the actual identity of users by using pseudonyms instead of actual identities has been studied in ((23), (24)). However, studies have shown ((5), (6)) that pseudonyms are relatively unsafe for location anonymity. Extending on the traditional K-anonymity model, researchers have studied K-anonymity protection for location data as well. Location data is made K-anonymous by replacing exact user locations with a larger area that includes K users. This area, that makes the user indistinguishable among K users, is called a cloaking area. Researchers have proposed various algorithms to find cloaking areas for a given user location and a K-anonymity requirement.

2.3 Spatial and Temporal Cloaking

The problem of location depersonalization on dynamic data was studied by Gruteser and Grunwald (10). The authors in this work introduced middleware architecture, with the use of a trusted location broker service, that provided only anonymous data to the service providers. As an extension of the traditional K-anonymity model (13), they proposed reducing the accuracy of a user's location information along spatial and/or temporal dimensions to produce a certain level of anonymity protection. Specifically, spatial cloaking is used to ensure that every location reported to a service provider is a cloaking area that contains at least K nodes. If the resolution of a location is too coarse for quality services, temporal cloaking is applied, i.e., user's service request is delayed. When more mobile nodes come near the user, a smaller cloaking area can be computed. This basic concept has inspired a series of research publications.

2.4 Personalized Anonymity

In (9), Gedik and Liu considered personalized anonymity. The approach by Grunwald et al provided spatial and temporal cloaking but there was a reduced quality of service either because of a coarse location resolution or a delayed response. Authors in this work allowed users to specify the minimum value of anonymity desired and the maximum spatial and temporal resolution they were willing to tolerate. Authors also developed the CliqueCloak algorithms to minimize the size of the cloaking areas, a factor critical for the quality of LBS. The proposed algorithm, however, has non-polynomial time complexity and is appropriate only when the value of K is small. The techniques proposed in (7) and (25), by Mokbel et al and Kalnis et al, respectively, also support customization of K, but do not minimize the size of the cloaking areas. An important contribution of these two works is their consideration of query processing, i.e., how a location-dependent query can be processed with a location of reduced resolution.

2.5 Single Cloaking

Xu and Cai proposed exploring historical location samples, known as footprints, for location cloaking (8). If a region has been visited by many different people, it is most likely a public area and cannot be linked directly to any specific user. Thus, as long as an area contains a sufficient number of different footprints, it can be used as a cloaking area. While this strategy significantly improves cloaking resolution for a given anonymity requirement, it also makes it possible to support anonymous uses of continuous LBS, wherein users must report their location frequently.

Xu and Cai propose an efficient polynomial time algorithm to find the cloaking area. Given a node N, the algorithm finds its minimal cloaking area , i.e., the minimum bounding circle (MBC) that contains N and $K-1$ other nodes. Let C_{min} denote this MBC, C_a denote a containing circle that contains N and at least $K-1$ other nodes, and C_b denote the bounding circle centered at N with a radius that is twice that of C_a. Also, let $C.R$ denote the radius for a given circle C. These notations are illustrated in Figure 3.1. The following observation supports the authors algorithm for searching C_{min} :

Theorem 1 *C_{min} must be bounded by C_b.*

Proof. By its definition, C_a contains K nodes including N. Since C_a is a candidate of C_{min}, C_{min} must be no larger than C_a, i.e., $C_{min}.R \leq C_a.R$. Since both C_{min} and C_a contain N, the distance between any point in C_{min} and N's position must not be larger than $2 \cdot C_a.R$. As a result, C_{min} must be inside C_b.

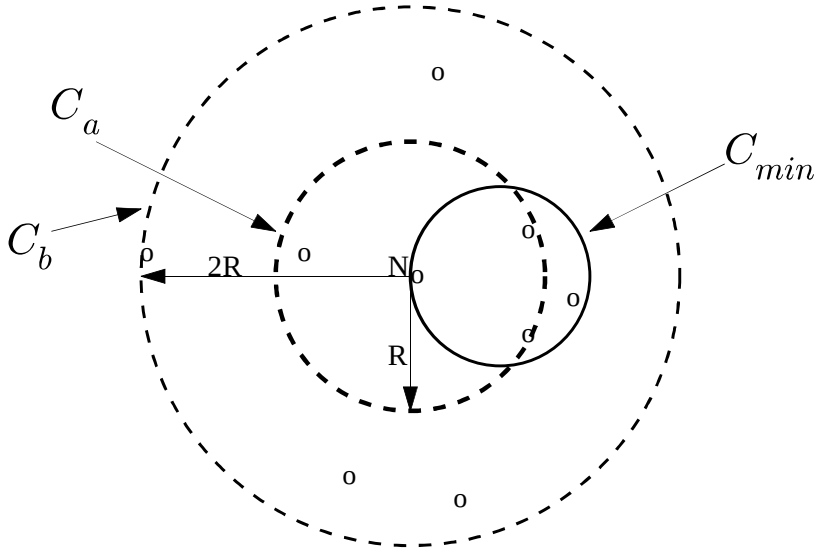

Figure 2.1 C_{min} must be inside C_b ($K = 4$)

The problem is how to find a C_a with a small radius. Location samples are indexed using a quad-tree, and the following simple approach is used to find a C_a. First, find the cell where N locates and mark this cell as the searching box. If the number of nodes inside the searching box is less than K, then expand the searching box by including its adjacent cells. This process is repeated until the searching box contains at least K nodes. Among these nodes, find $K-1$ nodes that are nearest to N and set C_a to be the MBC that bounds these $K-1$ nodes and N. This step costs $O(K)$.

After locating a C_a, determine C_b and retrieve all nodes inside C_b. Let S be the set of these nodes and $|S|$ the number of nodes in the set. As the area of C_b is 4 times that of C_a, the number of nodes inside C_b is $O(K)$. Given C_b and the set of nodes inside it, construct the candidates for C_{min} and then select the one that has the smallest radius as C_{min}. Since C_{min} is the minimum circle that contains N and at least $K-1$ other nodes, the circumference of C_{min} contains at-least two nodes. C_{min}'s candidates are classified into two categories.

A candidate in the first category has exactly two nodes on its circumference. In this case, the two nodes must form a diameter of the candidate. Such candidates can be enumerated by considering all possible pairs of nodes inside C_b. Given a pair of nodes, construct the circle with these two nodes as its diameter. The circle is a valid candidate if it contains N and at least $K-1$ other nodes. Among all valid candidates, find the one that has the smallest diameter. Let this candidate be C. Given a set of nodes S, there are totally $\binom{|S|}{2}$ different pairs of nodes. The computational cost in this step is $O(K^2)$.

A candidate in the second category has at least three nodes on its circumference. Note that any three nodes can form a triangle in a two-dimensional domain (as long as they are not on the same line), and this triangle can form only one circumscribed circle. Enumerate all possible triple nodes in S. For each triple, construct the circum-circle of the triangle formed by these three nodes. If the circle contains N and at least $K-1$ other nodes, it is a valid candidate. Again, among all valid candidates, find the one that is the smallest. Let this candidate be C'. The computational cost in this step is $O(K^3)$.

Finally, compare C with C', and designate the smaller one as C_{min}. Since the total cost of the entire process is $O(K) + O(K^2) + O(K^3) = O(K^3)$, the above algorithm finds C_{min} in polynomial time.

Despite the differences of these techniques in cloaking computation, they are all designed for anonymous uses of LBS, and they depersonalize users' locations one at a time when their requests arrive.

CHAPTER 3. LOCATION DATA PUBLISHING

3.1 System Overview

We assume a large number of mobile users with their locations sampled periodically. Each location sample is represented as a two-dimensional point. The location samples collected within one cycle are stored in a location table. For efficient retrieval of locations in a given region, we assume a quad-tree is used to index location samples from each location table. Over a period of time, many such location tables are created and old location tables might contain only a user's footprint. A footprint is defined to be a user's location sample collected at some point of time although the user might have physically moved from the location. Footprints help reduce the anonymizing area for location data published for a given time interval. Also, footprints enhance security, because the more footprints in a given region the less likely one can be correlated to successfully identify a subject. We also consider using footprints as location samples while publishing anonymized location data. From this point on we will use the terms location sample and footprint interchangeably.

Each user can specify a value of K (i.e., a desired level of anonymity protection) that can be included in a user's profile. Given a user's location (x, y) and the corresponding value of K, we want to identify a *K-Anonymity Area* (KAA), defined to be a circular region [1] that contains the point (x, y) and at least $K - 1$ other users' locations sampled at the same time period. Thus, given a location table with n users' positions, we want to convert it into a new location table with n KAA. The new table can then be released to the general public.

To make published location data as useful as possible, each KAA should be as small as

[1] A rectangular region can also be used as a KAA. We choose to use a circular rather than rectangular region because different rectangles can have the same area. Nevertheless, our algorithm can also generate a rectangular KAA.

possible. In the following subsections, we discuss how to compute KAA for a number of nodes in one batch, by extending the Single Cloaking algorithm proposed in (8). Our proposed batch-cloaking algorithm takes advantage of the proximity of nodes to anonymize a set of nodes per iteration of the single cloaking algorithm. We also extend this random batching of nodes, by proposing a sweep technique. The proposed sweep technique attempts to maximize the number of nodes in each batch.

3.2 Batch Cloaking - Random

Given a set of location samples, the Single Cloaking algorithm depersonalizes them one at a time. This process can be improved through batch processing. Specifically, in the process of finding a C_{min} for a particular node N with an anonymity requirement of K, we can also find C_{min} for other nodes that are nearby. C_a is defined as the containing area with K nodes associated with the node being anonymized. Our key observation here is that the containing area would remain the same for all nodes within C_a, and having an anonymity requirement less than or equal to K. Let C_a be a bounding circle that contains N and at least $K-1$ other nodes, where K is N's anonymity requirement, and C_B the circle centered at N with a radius that is three times that of C_a. Let N' be any node inside C_a with an anonymity requirement K', where $K' \leq K$, and C'_{min} be its minimum KAA. These notations are illustrated in Figure 3.1. We have the following claim for C_{min} of any node in C_a:

Theorem 2 *C'_{min} must be bounded by C_B.*

Proof. By its definition, C_a contains K nodes including N' and is therefore a candidate of C'_{min}. As N' is inside C'_{min}, its distance to any point in C'_{min} must not be larger than $2 \cdot C_a.R$. Since the largest distance between N' and N is $C_a.R$, the distance between N and any point in C'_{min} must not be larger than $3 \cdot C_a.R$. Thus, C'_{min} must be inside C_B.

Once we determine C_a for a particular node, we can find C_{min} for all nodes inside C_a with an anonymity requirement no larger than K. We will refer to these nodes as *batch nodes* hereafter. However, with batch cloaking the search scope increases to C_B (with a radius of

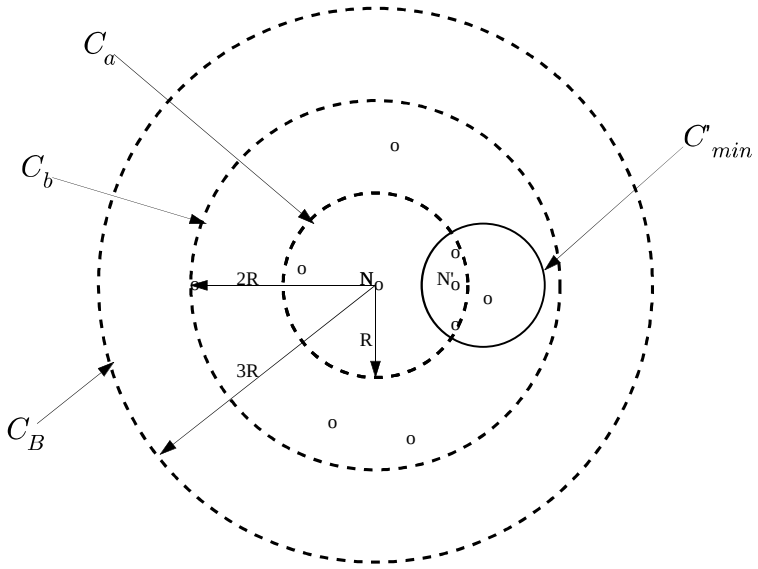

Figure 3.1 C'_{min} must be inside C_B ($K = 4$)

$3 \cdot C_a.R$), as compared to C_b (with a radius of $2 \cdot C_a.R$). This means batch cloaking is more expensive if we find C_{min} for one node at a time. Whether or not we should apply batch processing depends on whether there are a sufficient number of batch nodes. Suppose there are n nodes inside C_a and the number of batch nodes is n_b. Assuming the nodes are uniformly distributed, the number of nodes inside C_b and C_B can be estimated as $4n$ and $9n$, respectively. If we cloak each node one by one, the total number of C_{min} candidates we need to construct is $n_b \cdot (\binom{4n}{2} + \binom{4n}{3})$. On the other hand, if we apply batch cloaking, then the total number of C_{min} candidates in C_B is $\binom{9n}{2} + \binom{9n}{3}$. Thus, if the number of batch nodes is larger than $\frac{\binom{9n}{2}+\binom{9n}{3}}{\binom{4n}{2}+\binom{4n}{3}}$, then batch processing can be applied to improve performance.

In light of the above observation and analysis, we can devise a batch cloaking algorithm. Supposing that we allocate a buffer B in main memory to hold the location samples to be cloaked, we have the following algorithms.

BatchCloaking

1. Fill B with the location samples to be depersonalized;
2. Find the location sample, say N, in B with the highest value of K;
3. Find a bounding circle C_a that contains N and at least $K-1$ other nodes;
4. Let n be the number of nodes inside C_a (these nodes are all batch nodes since their anonymity requirement must be no larger than K);
5. If $n < \frac{\binom{9n}{2}+\binom{9n}{3}}{\binom{4n}{2}+\binom{4n}{3}}$, then do single cloaking as follows:
   - Compute C_b;
   - Load all nodes inside C_b;
   - Construct all C_{min} candidates;
   - Set C_{min} to be the one with the smallest radius that contains N and at least $K-1$ other nodes;
   - Output N and its C_{min};
   - Remove N from B.
6. Otherwise, do batch cloaking as follows:
   - Compute C_B;
   - Load all nodes inside C_B;
   - Construct all possible circles with these nodes (remove one immediately if its radius is larger than C_a);
   - For each node inside C_a, find its C_{min} from the candidates constructed above;
   - Output all these nodes and their corresponding C_{min};
   - Remove all these nodes from B.
7. Fill B with the remaining nodes on the disk and repeat the above process until all nodes are depersonalized.

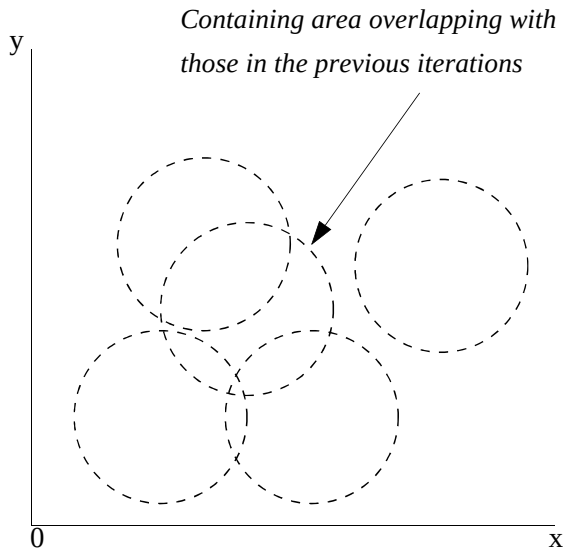

Figure 3.2 Containing circle samples in random batching

3.3 Batch Cloaking - Sweep

The batch-cloaking algorithm proposed above works significantly better than the single-cloaking algorithm. However, the approach used, to group nodes into one batch is greedy. The drawback to this approach is that it does not consider the distribution of nodes and does not follow any order in de-personalizing nodes and hence might result in picking redundant nodes for batch processing. A redundant node in our case would be a location sample which is already de-personalized. For example, with the random-batching technique we pick a node with the highest value of anonymity requirement, K, and find a containing circle. We then construct the bounding circle and identify the KAA for all nodes within this containing circle. In subsequent iterations we follow the same approach, picking nodes with higher value of K among the remaining unprocessed nodes. It is possible that the containing area constructed here overlaps with one of the containing areas from previous iterations. This reduces the number of unprocessed nodes that can be added to the current batch. Figure 3.2 depicts one such scenario. The containing circle designated in the figure will result in fewer batching nodes for the current iteration. The algorithm would require more iterations to complete processing

all nodes and hence higher computational cost. Hence, it is desirable to increase the number of batching nodes per iteration.

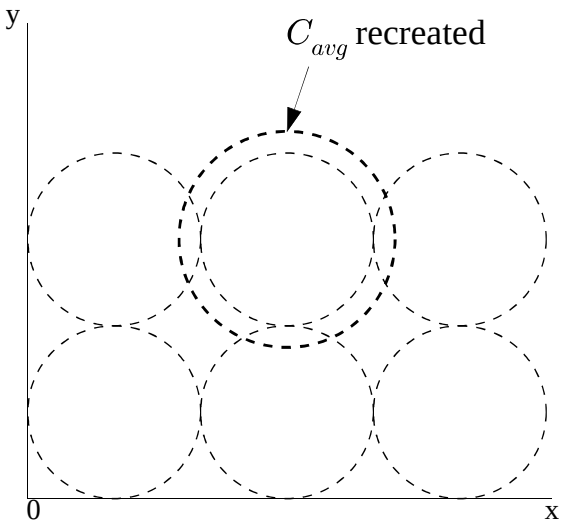

Figure 3.3 Containing circle samples in sweep batching

We propose an improvement over random batching, called a sweep technique. The sweep technique is a simple approach which considers the density of location sample distribution. The location samples are represented on a quadrant formed by x and y axes, and the idea is to sweep over the quadrant starting at one end and covering the entire area, as shown in Figure 3.3. We estimate the average density, D_{avg} of the location samples (footprints) and the average anonymity requirement value, K_{avg}. We now construct an imaginary containing area C_{avg} based on K_{avg} and D_{avg}. In the first iteration the x and y co-ordinates of the centre of the containing circle are set to x = y = r, where r is the radius of C_{avg}. All nodes within this containing area are included for batch processing and if there is any node with an anonymity requirement value greater than K_{avg}, the K value is reset to this higher value for the current iteration. Finally, we reconstruct the containing area for the current iteration with centre (r,r) and the latest K value for the current iteration. We then follow the approach discussed in the previous section to find KAA for all nodes in the batch. For the next iteration, we roll C_{avg} to

the next position by retaining the y-co-ordinate but resetting the x-co-ordinate to the previous value + 2*r, r being the radius of C_{avg}. We continue the sweep until we hit the boundary on the x-axis and then reset x to r and y to previous value + 2*r. We stop once the entire grid is covered and we then use our random batching to process any remaining nodes.

CHAPTER 4. PERFORMANCE STUDY

We have implemented the proposed techniques and evaluated their performance with simulated collections of location samples, each having its own anonymity requirement K. Given a set of location samples, we index them using a quad-tree. Each internal node of the tree has pointers linking its four child nodes, and each external (leaf) node is a fixed-size buffer that occupies one disk page. For each set of location samples, we depersonalize them using these approaches, *single cloaking* , *random batching* and *sweep batching*. The former depersonalizes location samples one at a time while the latter two do so in batch. We plot the results for all three techniques in the same graphs. We observe that random batching outperforms single cloaking, and the sweep technique shows further improvement in performance over random batching. We choose two performance metrics:

- *CPU cost*: Given a set of location samples, we record the total CPU time used by each technique to depersonalize them all, and report this time as a technique's CPU cost.

- *Disk I/O cost*: Given a large set of location samples, it may be too large to be entirely loaded into main memory for processing. In this case, disk I/O becomes the performance bottleneck, and cache-replacement policies have a significant impact on the total time spent in depersonalization. To avoid assuming some specific cache-replacement policy, we count the number of different leaf nodes accessed in processing each location sample, and use this count as the estimation of disk I/O cost.

Our first study investigates how the techniques performance is affected by the number of location samples. In this study, we generated a number of different sets of location samples, ranging from 10,000 to 100,000. The level of anonymity requirement K of these location samples range from 5 to 20 with an average of 10. The disk page size (quad-tree leaf node) is

set to be 4K bytes. We ran both single and batch approaches on these sets of location samples and plotted their performance results in Figure 4.1 and Figure 4.2 . As the number of location samples increases, both batch and single take more CPU time and disk I/O cost to complete depersonalization. However, batch outperforms single by about 50% in all settings, showing a significant advantage of depersonalizing as many location samples as possible per iteration.

Our second study investigates how the performance of the two techniques is affected by the level of anonymity requirement. In this study, we generated 50,000 location samples, but varied their average anonymity requirement K from 5 to 20. The size of the quad-tree leaf node is again fixed at 4KB. The performance results of both single and batch are illustrated in Figure 4.3 and Figure 4.4. For CPU cost and Disk IO, when the value of K is small (from 5 to 10), the performance of the techniques is about the same, but as K increases (beyond 10), the performance gap increases as well due to the fact that when K is smaller, the number of batch nodes included in C_a is less. When this number is not sufficient, there is actually no batch processing. When the number K becomes larger, there is more chance for batch processing, which in turn significantly improves the overall system performance. This study indicates that batch processing can perform much better when there is sufficient memory allocated for depersonalization. In reality, users tend to choose a larger value of K for their privacy protection, making batch processing highly desirable.

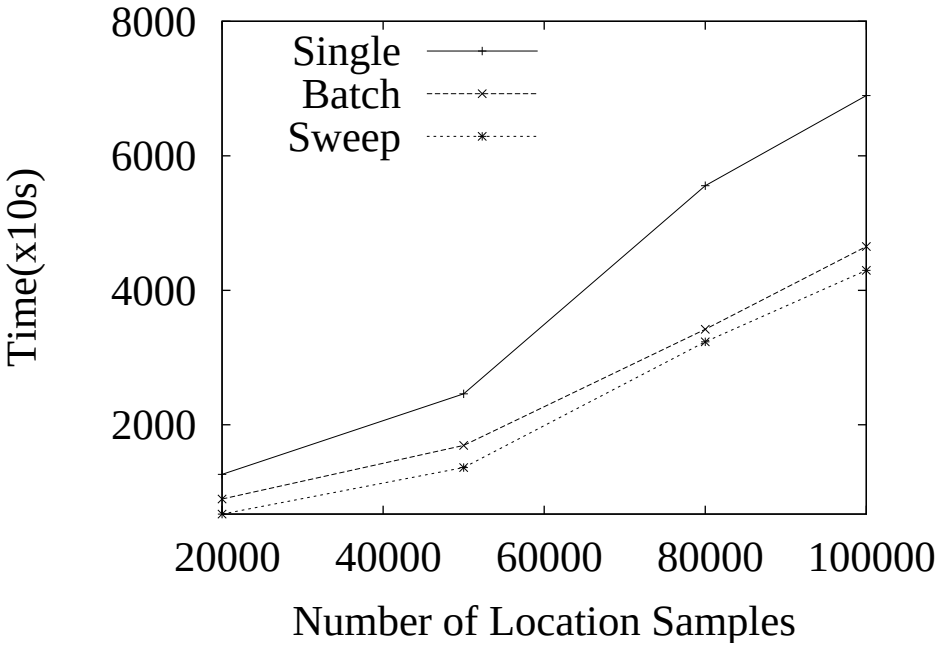

Figure 4.1 Effect of the number of location samples

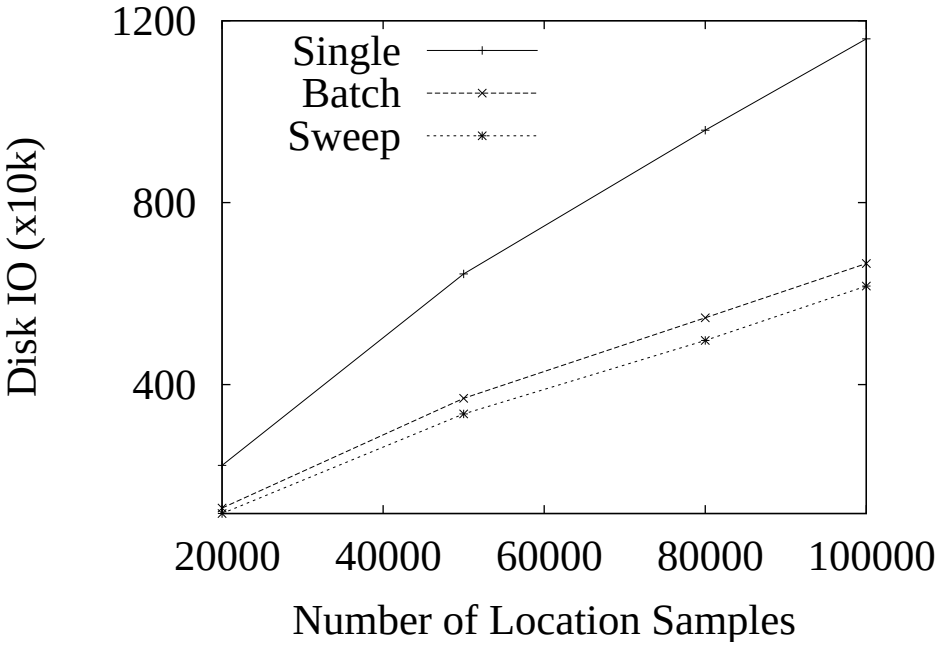

Figure 4.2 Effect of the number of location samples

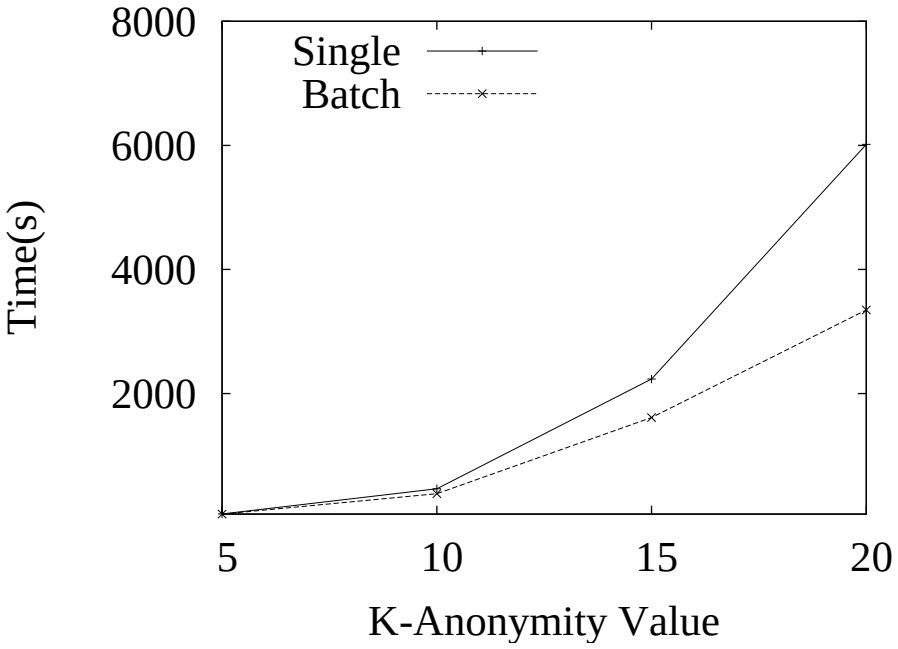

Figure 4.3 Effect of anonymity requirement

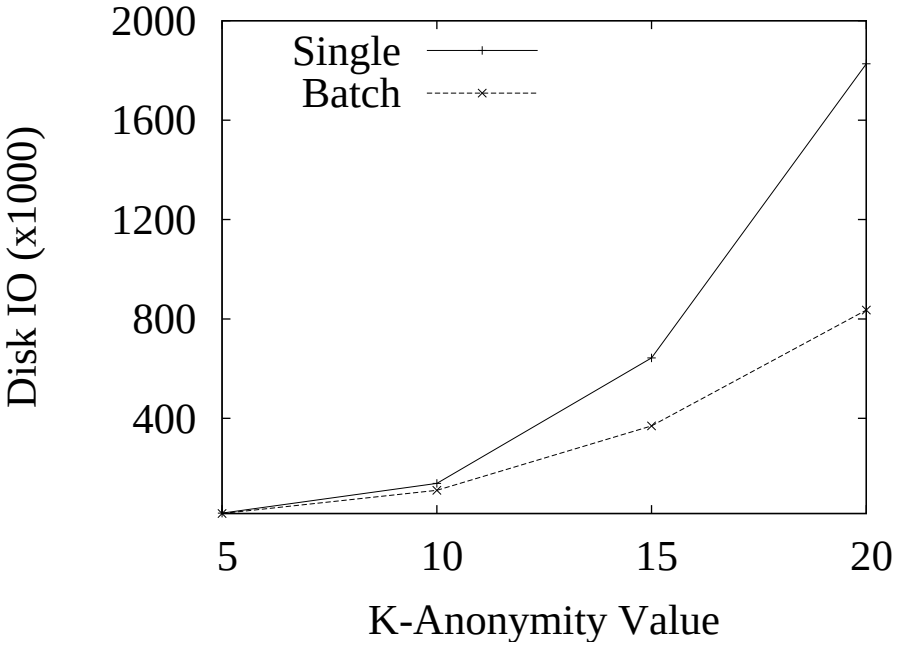

Figure 4.4 Effect of anonymity requirement

CHAPTER 5. SUMMARY AND DISCUSSION

Today's wireless service carriers have accumulated large collections of location data that are of great value to many organizations. However, making such data accessible to the general public would present significant privacy and security threats to individuals. To the best of our knowledge, this paper is the first that investigates the challenges of anonymity-preserving location data publishing. Our research goal was to allow location data to be published as accurately as possible, yet prevent them from being used for subject re-identification with a level of guarantee that is user-specified. We have presented a novel location depersonalization technique for efficient depersonalization of large volumes of location data. We have evaluated the performance of the proposed techniques through simulation, and our results show that the batching technique is highly efficient in publishing large volumes of location data.

BIBLIOGRAPHY

[1] Francisca Rojas and Francesco Calabrese and Filippo Dal Fiore and Sriram Krishnan and Carlo Ratti, *Real Time Rome*, MIT senseable lab. Illustrates practical use of location data, available at `http://www.holcimfoundation.org/Portals/1/docs/F07/WK-Inf/F07-WK-Inf-ratti02.pdf`. Accessed Jan 2009

[2] MapInfo, *What is Location Intelligence?*,Products. Illustrates practical use of location data, available at `http://mapinfo.com`. Accessed Jan 2009

[3] Changqing Zhou and Dan Frankowski and Pamela Ludford and Shashi Shekhar and Loren Terveen (2004). Discovering Personal Gazetteers: An Interactive Clustering Approach *GIS '04*, *1*pages 266-273.

[4] Craig Harvey and Amy Zeller, *Emergency Response Data Sharing Solution*, ERDAS Inc and NVision Solutions, available at Illustrates practical use of location data `http://gisdevelopment.net/proceedings/mapafrica/2008/maf08_19.pdf`. Accessed Jan 2009

[5] A. R. Beresford (2003). Location Privacy in Pervasive Computing. *IEEE Security and Privacy Pervasive 2003, Volume 2*, *1*pages 46-55.

[6] John Krumm (2007). Inference Attack on Location Tracks *Pervasive 2007 Fifth International Conference on Pervasive Computing*,

[7] M. F. Mokbel and C.-Y. Chow and W. G. Aref (2006) The New Casper: Query Processing for Location Services without Compromising Privacy *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB'06)*, *1*pages 763-774.

[8] Toby Xu and Ying Cai (2008) Exploring Historical Location Data for Anonymity Preservation in Location-based Services *Proceedings of IEEE INFOCOM, to appear*,

[9] B. Gedik and L. Liu (2005). A Customizable k-Anonymity Model for Protecting Location Privacy *ICDCS '05*, *1* pages 620-629.

[10] M. Gruteser and D. Grunwald (2003). Anonymous Usage of Location-based Services through Spatial and Temporal Cloaking *ACM MobiSys '03*, *1* pages 31-42.

[11] Caludio Bettini and X. Sean Wang and Sushil Jajodia (2005). Protecting Privacy Against Location-Based Personal Identification *SDM 2005, LNCS 3674*, *1* pages 185-199.

[12] Chi-Yin Chow and Mohamed F. Mokbel (2007). Enabling Private Continuous Queries For Revealed User Locations. *SSTD 2007, LNCS 4605*, *1* pages 258-275.

[13] Latanya Sweeney. k-Anonymity:A Model For Protecting Privacy *Fuzziness and Knowledge Based Systems,10(5)* , *1* pages 557-570.

[14] Boruch, Robert F and Cecil, Joseph S. The Privacy Act of 1974 and the Social Sciences Need for Access to Data. *International Conference on Emerging Data Protection and the Social Sciences Need for Access to Data, Germany (1978)*, *1* pages 23.

[15] G. Duncan and R. Pearson (1991). Enhancing access to data while protecting confidentiality: prospects for the future. *Statistical Science, Volume 6, Number 3 (1991)*, *1* pages 219-232.

[16] Andreas Pfitzmann and Marit Koehntopp. Anonymity, unobservability, and pseudonymity a proposal for terminology. *Proceedings of the International Workshop on Design Issues in Anonymity and Unobservability. volume 2009 of LNCS. Springer, 2000*,

[17] L. Sweeney. Guaranteeing anonymity when sharing medical data, the Datafly system.. *Proceedings, Journal of the American Medical Informatics Association. Washington, DC: Hanley and Belfus, Inc., 1997.*,

[18] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. *Technical Report SRI-CSL-98-04, SRI International, 1998*,

[19] Clay Shields and Brian Neil Levine (2000). A protocol for anonymous communication over the internet. *In Proceedings of the 7th ACM conference on Computer and communications security, (2000)*, *1*pages 33-42.

[20] David L. Chaum (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM, 24(2): 1981*, *1*pages 8490.

[21] European Union. Data protection directive (95/46/ec). *Official Journal of the European Communities, L. 281:31, November 1995*, available at Illustrates regulatory policies for privacy protection `http://www.cdt.org/privacy/eudirective/EU_Directive_.html`. Accessed Jan 2009

[22] Sastry Duri, Marco Gruteser, Xuan Liu, Paul Moskowitz, Ronald Perez, Moninder Singh, and Jung-Mu Tang. Framework for security and privacy in automotive telematics. *In Proceedings of the second international workshop on Mobile commerce. ACM Press, 2002*, *1*pages 2532.

[23] Satyajayant Misra, Guoliang Xue. Efficient anonymity schemes for clustered wireless sensor networks. *International Journal of Sensor Networks, Volume 1, 2006*, *1*pages 50-63.

[24] Heesook Choi, Thomas F. La Porta, and Patrick McDaniel. Privacy Preserving Communication in MANETs *Proceedings of Fourth Annual IEEE Communications Society Conference on Sensor, Mesh, and Ad Hoc Communications and Networks, June 2007,CA*.

[25] P. Kalnis and G. Ghinita and K. Mouratidis and D. Papadias (2006). Preserving Anonymity in Location Based Services. *Technical Report TRB6/06, Dept. Of Computer Science, National University of Singapore*,

[26] Kristen LeFevre and David J. DeWitt and Raghu Ramakrishnan (2005). Incognito: Efficient Full Domain K-Anonymity *SIGMOD '05*, *1* pages 49-60.

[27] A.R. Beresford (2005). Location Privacy in ubiquitous computing *University of CAMBRIDGE, Technical Report*, *1* pages 98-102.

[28] Sarah F. Frisken and Ronald N. Perry (Nov 2002) Simple and Efficient Traversal Methods for Quadtrees and Octrees

[29] N. Roussopoulous and S. Kelly and F. Vincent (1995). Nearest Neighbor Queries *ACM SIGMOD '95*, *1* pages 71-79.