CORGI: An interactive framework for Customizable and Robust Location Obfuscation

Primal Pappachan
Pennsylvania State University, USA
primal@psu.edu

Vishnu Sharma Hunsur Manjunath
Pennsylvania State University, USA
vxh5104@psu.edu

Chenxi Qiu
University of North Texas, USA
chenxi.qiu@unt.edu

Anna Squicciarini
Pennsylvania State University, USA
acs20@psu.edu

Hailey Onweller
Pennsylvania State University, USA
hso5021@psu.edu

2023 IEEE 39th International Conference on Data Engineering (ICDE) | 979-8-3503-2227-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICDE55515.2023.00294

***Abstract*—Customizing the location obfuscation functions generated by existing systems can result in weakening the privacy guarantees offered by these functions as they are not robust against such updates. In this demo, we present a new framework called, CORGI, i.e., CustOmizable Robust Geo Indistinguishability. The demonstration platform is a web application which is built on top on a real world dataset (Gowalla). The user-friendly interface of the demo allows participants to easily specify their customization preferences and generate a customizable and robust location obfuscation function. They can also examine the trade-offs among privacy, utility, and customization; visualized on a map for comparison between CORGI and a state of the art baseline.**

I. Introduction

In light of the increased number of location-based services, many location obfuscation mechanisms have been successfully proposed as simple but effective privacy mechanisms [1]. The key idea underlying these approaches is to protect users' privacy while ensuring the quality of service by transforming users' actual locations into obfuscated locations. *Geo-Indistinguishability (Geo-Ind)* is one of the most popular privacy criteria used in location obfuscation mechanisms [2]. It extends the well known *Differential Privacy (DP)* [3] paradigm to protect location privacy in a rigorous fashion. To satisfy Geo-Ind, if two locations are geographically close, their reported obfuscated locations will have similar probability distributions. In other words, given the obfuscated location it is hard for an adversary to distinguish a true location among nearby ones.

The obfuscation function is formulated as a Linear Programming (LP) problem, after discretizing an area of interest. As this problem has a large number of Geo-Ind constraints, it is solved at a cloud server instead of users' devices which have limited computation capability [4], [5], [6]. These obfuscation functions, generated using such a workflow, tend to be monolithic as it provides the same obfuscation range and the granularity of location sharing for all users. The obfuscation range is a set of locations from which an obfuscated location is chosen, and the granularity of the location determines the size/semantics of the location being shared (e.g., lat-long pairs, block, county). Prior work [7], [8] on customizing DP focused on statistical releases of data and not point queries that are used for sharing location data.

We propose *CORGI* (CustOmizable Robust Geo Indistinguishability), a framework for generating location obfuscation with strong privacy guarantees (based on *Geo-Ind*) that effectively allows users to balance the trade-off among privacy, utility, and customization. CORGI uses a semantic representation of a given region, in the form of a tree structure. This location tree helps users in specifying their required obfuscation range and granularity of location sharing, in the form of preferences. As the server and communication channel are untrusted, these preferences are evaluated on the user side and selectively shared (only the number of locations to be removed from the obfuscation range and not the exact locations are shared) with the server so as to protect the privacy of the user. CORGI relies on the untrusted server for performing the computationally heavy task of generating the obfuscation function by taking into account the user needs for customization. The robust obfuscation function generated by CORGI is customizable, while satisfying Geo-Ind requirements, with only minimal loss in utility compared to the traditional non-robust approaches.

During the demo, the participants interact with a web application in order to customize a location obfuscation function. Using a real dataset containing user check-ins (Gowalla), we have created different simulation scenarios where participants can quickly experiment with different settings without having to share their real data. The web application visualizes the results of location obfuscation in an intuitive manner so that participants can learn about the privacy guarantees of Geo-Ind as well as the impact of customization on these guarantees. Using this application, a demo participant will be able to

- ▷ Gain an intuition for location obfuscation functions based on Geo-Ind used for Location Privacy
- ▷ Select different ready made scenarios and different customization parameters to generate customizable and robust obfuscation functions
- ▷ Compare the performance of CORGI against a state-of-the-art non-robust baseline in terms of utility and privacy guarantee, while supporting customization

2375-026X/23/$31.00 ©2023 IEEE
DOI 10.1109/ICDE55515.2023.00294

The core elements of our demo include: 1) a set of representative users along with their metadata, 2) multiple customization settings pane that the demo participant can change, and 3) a map on which participants input their locations and the output of obfuscation functions are visualized in the form of a heat map. This demo illustrates how CORGI can be a meaningful tool for individuals to customize location obfuscation functions and examine the impact of these parameters on both privacy and utility.

II. Overview

In this section, we briefly summarize the key concepts used in CORGI and present a brief overview of the CORGI framework (see Figure 1).

A. Preliminaries

Location Tree: We build a hierarchical index over a given spatial region for location representation. We design a tree-like structure, called *location tree*, where each level of the tree represents a particular granularity of location data, and lower levels of the tree increase granularity. This representation of locations is intuitive and makes it easier for users to specify the granularity of location sharing they are comfortable with. The location tree is generated using Uber's H3[1] hexagonal hierarchical spatial index which takes as input as a region (longitude and latitude) and resolution (between 0 and 15, with 0 being coarsest and 15 being finest), and outputs a hexagonal grid index for the region.

Obfuscation matrix. Given an area of interest, the obfuscation range is represented as a finite discrete location set $\mathcal{V} = \{v_1, ..., v_K\}$. Given this set of K locations, an obfuscation function can be represented as a stochastic matrix $\mathbf{Z} = \{z_{i,j}\}_{K \times K}$ [9]. Here, each $z_{i,j}$ represents the probability of selecting $v_j \in \mathcal{V}$ as the obfuscated location given the real location $v_i \in \mathcal{V}$. For each real location v_i (corresponding to each row i of $\mathbf{Z}$), the *probability unit measure* needs to be satisfied:

$$\sum_{j=1}^{K} z_{i,j} = 1, \forall i = 1, ..., K, \tag{1}$$

i.e., the sum probability of its obfuscated locations is equal to 1. From the attacker's perspective, the user's actual and reported locations can be described as two random variables X and Y, respectively. We apply *Geo-Indistinguishability (Geo-Ind)* [2] as the privacy criterion for location privacy guarantees and (ϵ-Geo-Ind). For a given the obfuscation matrix $\mathbf{Z}$ that covers a set of locations $\mathcal{V}$ at the same granularity level in the location tree, $\mathbf{Z}$ satisfies ϵ-Geo-Ind if only if for each pair of real locations $v_i, v_j \in \mathcal{V}$ and any obfuscated $v_l \in \mathcal{V}$

$$\frac{\Pr\left(X = v_i \,|Y = v_l\right)}{\Pr\left(X = v_j \,|Y = v_l\right)} \leq e^{\epsilon d_{i,j}} \frac{p_{v_i}}{p_{v_j}}, \forall v_i, v_j, v_l \in \mathcal{V} \tag{2}$$

where p_{v_i} and p_{v_j} denote the prior distributions of v_i and v_j, respectively, $\epsilon > 0$ is predetermined constant called privacy parameter/budget, and $d_{i,j}$ denotes the distance between v_i and v_j which can be implemented using spherical, euclidean, or hop distances. We define *obfuscation in a location Tree* $(\mathcal{V}, \prec)$ as a function that maps a given real location $v_i \in \mathcal{V}^n$ to another location $v \in \mathcal{V}^n$ that is at the same level with v_i.

In Location Based Services (LBS), a user shares their location in order to receive a target location at which the required service is available (e.g., restaurant, gas station). The utility of location obfuscation is computed as quality loss based on a given target location. Given that user's real location is v_i, the reported location is v_j, and the target location is v_n, the quality loss is given by

$$QL(v_i, v_j, v_n) = \mid d_{v_i, v_n} - d_{v_j, v_n} \mid . \tag{3}$$

User Customization: Users specify their customization needs using policies. A policy consists of three parameters: 1) *Privacy level* determines the obfuscation range i.e., set of locations/nodes from the location tree from which users' obfuscated location is selected. A higher privacy level implies a wider range of obfuscated locations to select for users. 2) *Precision level* specifies the exact granularity at which the user reports their locations. For example, if a user requires the precision level to be 1, then their reported location/node is restricted to the set of nodes in level 1 of the location tree[2]. 3) *Customization Preferences* specify users' preferred options for location selection and further narrows down the obfuscation range and therefore reduces the number of locations/nodes in the obfuscation matrix. An example of a policy modeled using these 3 attributes is as follows: <*privacy_l = 3*, *precision_l = 0*, *user_preferences = [distance* $\leq$ *5 miles*] This customization policy states that the nodes from level 3 (privacy_l = 3) represent the maximum obfuscation range. From this set of possible locations, any of them which has a distance higher than 5 miles from their real location should not be considered for reporting (user_preferences). Finally, when generating their obfuscated location, they would like it to be at the granularity of level 0 (precision_l = 0) which are the leaf nodes.

B. Framework

Our problem setting is that of *LBS* where users share their privatized locations with a server in order to receive service provisioning (e.g. Uber, Lyft, Yelp, Citizen Science). There are three main actors in our setting: *users*, *LBS*, and a *server*. *Users* wish to share their locations in a privacy-preserving manner with LBS applications. *LBS* use the privatized locations shared by the user for providing services to the user. Finally, we have the *server* which runs on the cloud with whom non-sensitive portions of the policies are shared and it takes care of computationally heavy operations. Users do not trust neither the LBS applications nor the server with their sensitive location information or policies. Figure 1 introduces the flow of CORGI and interactions among these three actors. Please refer to the full version of our paper for more details [10].
① The server generates a location tree for an area of interest.
②③ The location tree is shared with the users to allow them

[1]https://www.uber.com/blog/h3/

[2]As the privacy level is the maximum possible granularity for location sharing, the precision level is always lower than the privacy level.

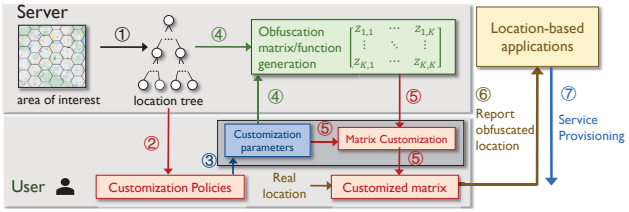

Fig. 1. Overview of CORGI framework.

to specify their policies.

④ CORGI evaluates the policies and shares the relevant customization parameters with the server, using which the server generates the obfuscation function. This function is represented by a set of probability distributions in an *obfuscation matrix* and provides strong location privacy guarantees based on Geo-Indistinguishability [2].

⑤ Users receive the obfuscation function/matrix and customize it based on their needs.

⑥ ⑦ This customized obfuscation function is utilized to determine the user's obfuscated location, to be reported to LBS applications for the purpose of service provisioning.

III. Demonstration Plan

The demo is instantiated using the Gowalla dataset [11]. Gowalla is a location-based social networking website where users share their locations via check-ins. Each check-in entry has the following attributes: *[user, check-in time, latitude, longitude, location id]*. We sampled the user check-ins from the San Francisco (USA) region. We chose this region because it has a dense distribution of user check-ins distributed over a large area. Overall, this sample includes 38,523 check-ins. We discretized the area into hexagonal partitions at different resolutions using the H3 library. We generated the root node, which covers the entire region at resolution 6 followed by the children for this root node at resolution 7. We repeated the process twice and generated a tree of height 3 with a total of 343 leaf nodes. The initial location tree is pre-generated, but the participant has the option to update it as necessary, as explained later.

To provide a meaningful experience to the demo participants, CORGI will be preloaded with data from five prototypical users. For each of the five prototypical users, we have extracted their check-ins and built their profiles accordingly. Precisely, we assigned the location where they spent the most time during work hours as the office and the location where they spent the most time during off hours as home. We also analyzed the number of check-ins per location in order to identify metadata about locations such as which ones are popular during the day and night. The representative users with their metadata and location metadata are used to set up customization preferences, allowing participants to quickly explore the functionality of CORGI and examine its effectiveness in customizing location obfuscation.

The demo also includes an approach that computes obfuscated locations using *linear programming (LP)* [12], [9], [4]. This is our *baseline* approach, as it is not robust against the removal of locations from the obfuscation range on the user side. The steps in interacting with the demo are as follows. The participant can learn about the different options by clicking on the question mark button next to them. "Area" is by default set to San Francisco.

① The participant selects a representative user from the "user" drop-down menu and the application highlights their home and office location on the map. The participant can also click on any of the options under customization preferences and the corresponding set of locations (e.g., popular places) will be highlighted on the map.

② The participant inputs the current location of the user by clicking and dropping the red pin on the map. participant also inputs the target location(s) which is highlighted on the map with a green pin.

③ Under system settings, the participant has the option to update the privacy parameter ϵ (by default set to 5), and the total number of nodes in the tree (by default set to 49).

④ Under user settings, the participant selects the obfuscation range (privacy level - drop-down menu) and granularity of location sharing (precision level - drop-down menu).

⑤ The participant can customize the obfuscation function further by selecting specific locations to be omitted from the obfuscation range. Under *customization preferences*, the participant can select which location(s) should not be considered for reporting to a location service by checking the boxes next to it. Examples are home, office, popular, and deserted places (number of locations shown next to it).

⑥ Participant clicks on "Report my location" which simulates the reported location on the map (highlighted using a blue pin). This location is output from the obfuscation function generated based on input parameters. The map also shows the locations that violate the Geo-Indistinguishability constraint using a heat map. The "results" box shows for baseline and CORGI: 1) the "quality loss" (computed using Eq. 3), 2) the percentage of all entries in the obfuscation function that satisfy the Geo-Ind constraint (Eq. 2).

For the heat map, the heat/intensity of a location is determined based on the number of pairwise violations of Eq. 2 that the location is involved in. The locations with the largest number of violations are shown in red and the ones with the least violations are shown in green. If the user is at a location with a large number of violations, then an adversary with knowledge of the prior probability distribution of user check-ins in the area (from publicly available information) will be able to eliminate this location from the search range and hence increase the location identification accuracy. These violations occur due to customization, thus weakening the privacy guarantee provided by location obfuscation mechanisms. The quality loss with CORGI is slightly higher than the baseline however the number of violating locations for CORGI will be much lower than the non-robust baseline. This is because CORGI generates a robust obfuscation function taking into account the user's customizability needs whereas the baseline is generated with no room for customizability.

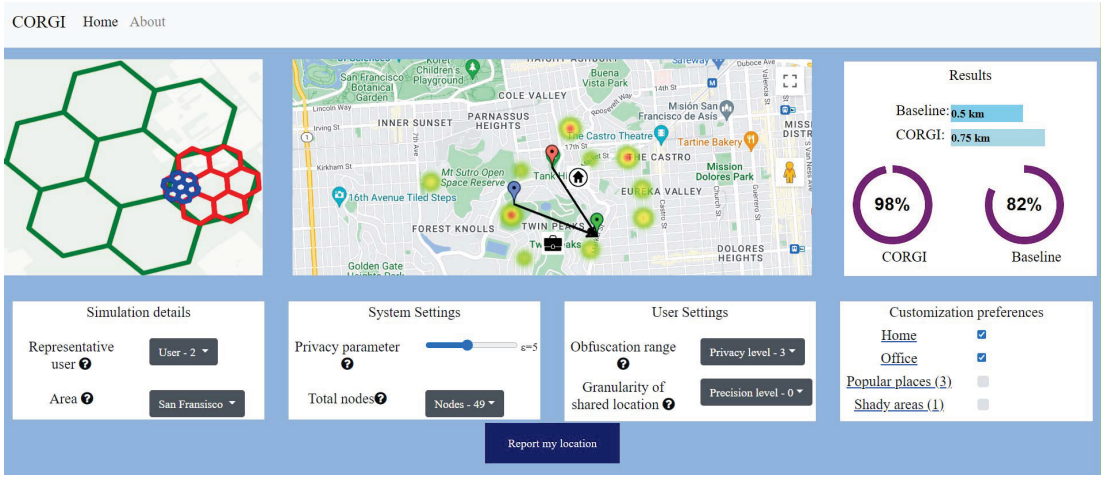

Fig. 2. Interface of CORGI

The participant can further interact with CORGI by updating any of the system settings, user settings, and customization preferences. After updating the settings, when a user clicks "Report my location" the application will generate a new obfuscation location and report a new location based on it.

IV. Conclusions

Customization of location obfuscation functions by end users is a challenging problem. CORGI is a first attempt at making the obfuscation functions based on Geo-Ind less monolithic and more user-centered. Using this demo, a participant can gain a deeper understanding of location obfuscation functions and examine the impact of customization on utility and privacy guarantees. In the future, we would like to use CORGI to handle trajectory data and protect against adversaries not just with prior knowledge but also with posterior knowledge.

V. Acknowledgement

This work was supported by NSF grant under CNS-2136948. We thank the reviewers for their comments which helped to improve the paper.

References

[1] J. W. Kim, K. Edemacu, J. S. Kim, Y. D. Chung, and B. Jang, "A survey of differential privacy-based techniques and their applicability to location-based services," *Computers & Security*, p. 102464, 2021.

[2] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 901–914.

[3] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy." *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3-4, pp. 211–407, 2014.

[4] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma, "Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation," in *Proc. of ACM WWW*, 2017, pp. 627–636.

[5] C. Qiu, A. C. Squicciarini, Z. Li, C. Pang, and L. Yan, "Time-efficient geo-obfuscation to protect worker location privacy over road networks in spatial crowdsourcing," in *Proc. of ACM CIKM*, 2020.

[6] R. Shokri, G. Theodorakopoulos, C. Troncoso, J. Hubaux, and J. L. Boudec, "Protecting location privacy: Optimal strategy against localization attacks," in *Proc. of ACM CCS*, 2012, pp. 617–627.

[7] D. Kifer and A. Machanavajjhala, "Pufferfish: A framework for mathematical privacy definitions," *ACM Transactions on Database Systems (TODS)*, vol. 39, no. 1, pp. 1–36, 2014.

[8] X. He, A. Machanavajjhala, and B. Ding, "Blowfish privacy: Tuning privacy-utility trade-offs using policies," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 1447–1458.

[9] C. Qiu, A. C. Squicciarini, C. Pang, N. Wang, and B. Wu, "Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[10] P. Pappachan, C. Qiu, A. Squicciarini, and V. S. H. Manjunath, "User customizable and robust geo-indistinguishability for location privacy (under revision)," in *Proceedings of the 26th International Conference on Extending Database Technology (EDBT)*. OpenProceedings.org, 2023.

[11] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 1082–1090.

[12] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proc. of ACM CCS*, 2014, pp. 251–262.