TASK ASSIGNMENT WITH RIGOROUS PRIVACY PROTECTION IN SPATIAL CROWDSOURCING

by

Hien To

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in Computer Science

Viterbi School of Engineering

University of Southern California

Accepted by:

Cyrus Shahabi, Committee Chair

Shahram Ghandeharizadeh, Committee Member

Ulrich Neumann, Committee Member

Aleksandra Korolova, Committee Member

Lian Jian, Committee Member

Li Xiong, Outside Committee Member

(COMPUTER SCIENCE)

April 2016

Contents

List of Tables

List of Figures

Abstract

Spatial Crowdsourcing (SC) is a novel transformative platform that engages individuals, groups and communities in the act of collecting, analyzing, and disseminating environmental, social and other spatio-temporal information. The objective of SC is to outsource a set of spatio-temporal tasks to a set of *workers*, i.e., individuals with mobile devices that perform the tasks by physically traveling to specified locations of interest. However, current solutions require the workers, who in many cases are simply volunteering for a cause, to disclose their locations to untrustworthy entities.

Therefore, we introduce a framework for protecting location privacy of workers participating in SC tasks. We argue that existing location privacy techniques are not sufficient for SC, and we propose a mechanism based on differential privacy and geocasting that achieves effective SC services while offering privacy guarantees to workers. We investigate analytical models and task assignment strategies that balance multiple crucial aspects of SC functionality, such as task completion rate, worker travel distance and system overhead. We address scenarios with both static and dynamic (i.e., moving) datasets of workers.

Chapter 1

Introduction

1.1 Motivation

Smartphones have recently surpassed the PC as the device of choice for accessing the Web, and mobile phone subscription has grown from 5.9 billion worldwide at the end of 2011 [42] to 6.9 billion in 2014, which is 95.5% of the world population. There has also been a significant increase in mobile phone bandwidth: from 2.5G (up to 384Kbps) to 3G (up to 14.7Mbps) and recently 4G (up to 100 Mbps) [54]. The increase in computational and communication performance of mobile devices, coupled with the advances in sensor technology (e.g., video cameras, GPS, motion sensors) lead to an exponential growth in data collection and sharing by smartphones. An individual with a mobile phone can nowadays act as a multi-modal sensor collecting and sharing various types of high-fidelity spatiotemporal data instantaneously (e.g., picture, video, audio, location, time, speed, direction, acceleration).

Exploiting this large volume of potential users and their mobility, a new mechanism for efficient and scalable data collection has emerged, named Spatial crowdsourcing (SC) [31]. With spatial crowdsourcing, the goal is to outsource a set of spatial tasks (i.e., tasks related to a location) to a set of workers who perform the spatial tasks by physically traveling to those locations. Spatial crowdsourcing has applications in numerous domains such as journalism, tourism, intelligence, emergency response and urban

planning. Examples of real-world spatial crowdsourcing applications are TaskRabbit[1], MediaQ[2], iRain[3], etc.

Effective assignment of workers to tasks is an important phase in spatial crowdsourcing, i.e., tasks are completed in a timely fashion, and workers do not incur significant travel cost [31, 32, 12, 61]. To that extent, matching must take into account the location of workers. However, disclosing individual locations has serious privacy implications [24, 43, 20, 29]. Without privacy protection, a malicious adversary can stage a broad spectrum of attacks such as physical surveillance and stalking, identity theft, and breach of sensitive information such as an individual's health issue (e.g., presence in a cancer treatment center), alternative lifestyles, political and religious preferences (e.g., presence in a church). Furthermore, leaking location information may lead to serious consequences. For example, a security flaw in a gay dating app named Grindr reveals precise location of 90% users [6]. This flaw led to serious consequences in countries where homosexuals faced extreme dangers, such as Egypt, Russia, Saudi Arabia. In Egypt, four men were sentenced to up to eight years in prison for homosexual conduct [46]. Thus, ensuring location privacy is an essential aspect of spatial crowdsourcing, because mobile users may not accept to engage in spatial tasks if their privacy is violated.

1.2 Thesis Statement

In this proposal, we identify privacy as the major impediment to the success of any spatial crowdsourcing system. Due to various threats, workers may hesitate to participate in spatial crowdsourcing campaigns if their locations are revealed to untrusted entities (e.g., spatial crowdsourcing company, task requester). Thus, we propose a server-assigned spatial crowdsourcing framework that enables the participation of the workers

[1] www.taskrabbit.com

[2] mediaq.usc.edu

[3] irain.eng.uci.edu

without compromising their privacy. More formally, our goal is to *use* ***differential privacy*** *in* ***task assignment*** *phase of* ***spatial crowdsourcing*** *to hide the location of workers from SC-server with low cost and low overhead without compromising performance.* Privacy of individual is protected with differential privacy, regardless of an adversary's background knowledge. The cost means the travel distance between an assigned worker to a task's location while overhead refers to the communication overhead required to send a task request to the assigned workers. The performance is measured by the number of completed tasks.

1.3 Privacy-Preserving Task Assignment in Spatial Crowdsourcing

Assignment of workers to tasks is a crucial step, with the goals that every task must be completed with high probability, and the travel cost of workers to the task locations must be minimized. Effective and efficient algorithms exist [31, 12, 64, 61, 63] that take as input worker locations and generate worker-task assignments, but they use exact locations, hence they do not provide privacy. In the proposed work, we will focus on performing worker-task assignment while at the same time protecting location privacy of workers. This is a challenging task, as protecting privacy either introduces uncertainty with respect to worker whereabouts, decreasing assignment quality, or creates significant communication overhead.

One may argue that simply removing workers' identity by using fake identity (i.e., pseudonymity) would achieve privacy. However, we argue that hiding users' identity without hiding their locations does not provide privacy. This is because a user's location information can be tracked through several stationary connection points (e.g., cell towers). The user's *location trace* can be easily associated with a certain residence home or office, which reveal the user's identity. This has been referred to as inference attack [37]. This study collects GPS data from volunteer users and finds their home locations with

median error of 60 meters. A study in [23] showed that there is a close correlation between people's identities and their mobility. Furthermore, an article in Nature [11] performed a large-scale study showed that 4 locations uniquely identify 95% individuals. Thus, hiding workers' locations are much more challenging than hiding his/her identity since the workers' locations are needed for effective task assignment.

Location privacy has been addressed before in the context of location-based services. Several solutions [24, 43, 20, 29] have been proposed to protect location-based queries, i.e., given a user's location, find points of interest in the proximity without disclosing the actual coordinates. However, in spatial crowdsourcing, the worker location is no longer part of the query, but rather the result of a spatial query around the task location. This is a considerably more challenging problem, and previous solutions do not offer satisfactory results. In addition, while some work considers queries on private locations in the context of outsourced databases [75, 74, 5], it is assumed that the data owner entity and the querying entity trust each other, with protection being offered only against intermediate service provider entities. This scenario does not apply in SC, as there is no inherent trust relationship between requesters and workers.

We propose a framework for protecting privacy of worker locations, whereby the SC-server only has access to data sanitized according to *differential privacy (DP)* [15]. In practice, there may be many SC-servers run by diverse organizations that do not have an established trust relationships with the workers. On the other hand, every worker subscribes to a *cellular service provider (CSP)* that provides Internet connectivity. The CSP already has access to the worker locations (e.g., through cell tower triangulation), but as opposed to the SC-server, the CSP signs a contract with its subscribers, stipulating the terms and conditions of location disclosure. The CSP collects user locations and releases them to third party SC-servers in noisy form, according to DP. However, using DP introduces three difficult challenges, as follows:

First, the SC-server must match workers to tasks using noisy data, which requires complex strategies to ensure effective task assignment. To create sanitized data releases

at the CSP, we adopt the *Private Spatial Decomposition (PSD)* approach [7]. A PSD is a sanitized spatial index, where each index node contains a noisy count of the workers rooted at that node. To ensure that task assignment has a high success rate, we introduce an analytical model that determines with high probability a PSD partition around the task location that includes sufficient workers to complete the task.

Second, the DP protection model requires fake entries to be created in the PSD. Thus, the SC-server cannot directly contact workers (even if pseudonyms are used) as establishing a connection to an entity would allow the SC-server to learn whether an entry is real or not, and breach privacy. To address this challenge, we propose the use of geocasting [45] as means to deliver task requests to workers. Geocasting is a communication model where a message is delivered to all devices situated within a geographical area, as opposed to conventional communication channels, where recipients are identified by a protocol address. Once a PSD partition is identified by the analytical model outlined above, the task request is geocast to all workers in the partition. Geocast introduces overhead considerations that need to be carefully considered in the framework design.

Third, protecting worker locations across multiple timestamps is notoriously difficult [17]. As workers move, new snapshots of sanitized worker locations must be disclosed, to maintain task assignment effectiveness. However, access to sequential releases gives an adversary more powerful attack opportunities. To counter such threats, differential privacy requires more noise injection, which in the worst case may reach amounts that are proportional to the length of the released location history (i.e., number of disclosed snapshots). Clearly, such large noise would render the data useless, since SC is likely to be a continuously offered service in practice. We study custom-designed mechanisms for differentially-private release of worker locations across consecutive timestamps that preserve location accuracy, such that the task assignment remains highly effective.

Our specific contributions in privacy-preserving task assignment are five-fold. We first identify the specific challenges of location privacy in SC, and we propose a framework that achieves differentially-private protection guarantees. Second, we propose an analytical

model that measures the probability of task completion with uncertain worker locations, and devise a strategy that finds appropriate PSD partitions to ensure high success rate of task assignment. Third, we introduce a geocast mechanism for task dissemination, and we factor the geocast system overhead in the PSD partition search strategy. Forth, we extend our solution to dynamic worker datasets. We investigate privacy budget allocation techniques across consecutive releases, and we employ post-processing techniques based on Kalman filters to reduce the inaccuracy introduced by noise addition. Fifth, we conduct extensive experiments on real-world datasets which show that our framework is able to protect workers' location privacy without compromising performance, while the extra travel cost and communication overhead are tolerable.

1.4 Privacy-Preserving Location Popularity (Proposal)

The proposed framework in Section 1.3 assigns individual task independently without considering their spatial and temporal information of each task, i.e., location and deadline. However, the SC-server may assign a batch of tasks at a particular time interval, e.g., every 1 minute. In addition, the SC-server may have a constraint on how many geocast queries it can request the cell service provider (CSP). In practice, the constraint can capture the communication overhead the SC-server incurs to geocast to selected workers, or the rewards paid to the CSP in order to use the geocast service. Therefore, we would like to maximize the number of completed tasks given the maximum number of geocast queries can be executed over a time period.

Toward that end, we identified task prioritization (i.e., which tasks should be assigned now and which tasks can be deferred to future time period) a key step in task assignment with constraints [31, 67, 61]. In those studies, the "popularity" of a task's location indicates whether the task can be assigned to future workers. The reason is that tasks situated in a "worker-sparse" area (i.e., area with fewer workers visit) is less likely to be performed in the future since there may be no worker visiting such locations. Thus, such

tasks would have higher priority at the current time period. For instance, in Figure 1.1 task 1 locates in a worker-sparse area and thus has higher priority.

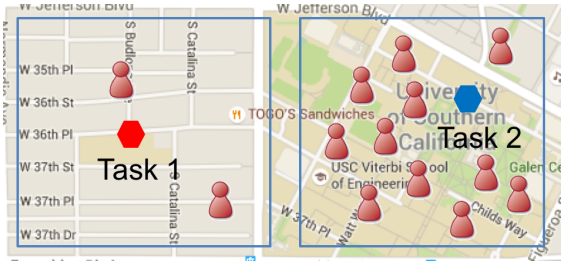


Figure 1.1: Task 1 locates in a less popular area and thus has higher priority in task assignment.

The "popularity" of a task location can be effectively measured using *location entropy* [9], which captures the diversity of visits to that region. A location has a high entropy if many workers visit with equal probabilities. In contrast, the location has a low entropy if there are only a few workers visiting. Figure 1.2 illustrates two locations with the same number of user visits; however, the second location has higher location entropy. For example, a particular user's home would have low entropy, while common areas like a university campus and department of motor vehicles are likely to have high entropy.

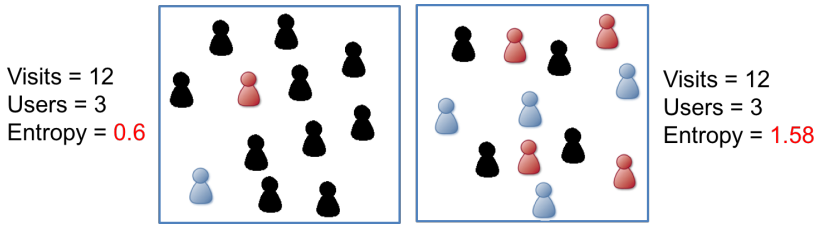


Figure 1.2: Both number of visits and entropy indicate *crowdedness* of a location, but only entropy represent popularity of a location.

Location entropy is an indicator of task priority, which has been used to maximize task assignment in spatial crowdsourcing [61, 67, 31]. However, it is generally computed from workers' location track, which has privacy implication [37]. Therefore, we aim to publish location entropy in differentially private manner while preserving privacy of users. Although there have been various studies on publishing counting query on

location data [72, 71, 7, 52, 53, 60] , differentially-private location entropy has not been yet explored.

Location entropy once published properly will have impacts beyond spatial crowdsourcing applications. It has been shown in [9] that location entropy gains significant improvement (15-20%) in accurately predicting friendship from location trails over simpler models based on only direct properties of the co-location histories, such as the number of co-locations. In [48], location entropy is used to enhance algorithms in computing social strength in geosocial networks, i.e., factoring weight of each location into consideration. The intuition is that two people who meet at a low-entropy location may imply a stronger social connection, compared to the connection between people who meet at a high-entropy location. Furthermore, [48] shows that comparing to the number of unique users visiting or the number of check-ins of a location, location entropy is a better measurement for location popularity.

Release location entropy with differential privacy is challenging due to two main reasons. First, the sensitivity of location entropy query can be high since each user may contribute many visits to a location, thus adding or removing a user with a large number of visits can significantly change the value of location entropy. Another factor contributes to the high sensitivity of location entropy is large variance on the number of visits per user. If users can contribute a very large number of visits (e.g., check-ins) to a particular location, they can have a large influence on location entropy. Second, location visits can be sparse, e.g., majority of locations have only a few users visiting while differentially private algorithms perform worse on sparse datasets. Thus, we need to design algorithms in the presence of skewness and sparseness that exist in real-world datasets.

Chapter 2

Background

2.1 Spatial Crowdsourcing

Crowdsourcing is the process of outsourcing tasks to a distributed group of individuals. The difference between crowdsourcing and ordinary outsourcing is that a task or problem is outsourced to an undefined public rather than a specific body, such as paid employees. Spatial Crowdsourcing [31] is defined as a type of online crowdsourcing where performing an outsourced task is only possible if the worker is physically at the location of the task (termed spatial task). Spatial crowdsourcing is enabled by mobile devices that allow users to perform tasks at a certain location and submit the results to a spatial crowdsourcing server.

In conventional (i.e., non-privacy-preserving) spatial crowdsourcing [31], tasks can be assigned to workers in two ways. In Worker Selected Tasks (WST) mode, the SC-server publishes the spatial tasks and workers can choose any tasks in their vicinity without the need to coordinate with the SC-server. In Server Assigned Tasks (SAT) mode, online workers send their location to the SC-server, and the SC-server assigns tasks to selected nearby workers. The advantage of WST is its simplicity, but the assignment is sub-optimal, as workers do not have a global system view. Workers typically choose the closest task to them, which may cause multiple workers to travel to the same task while many other tasks remain unassigned. The SAT mode incurs the overhead of running complex matching algorithms at the SC-server, but the best-suited worker is selected for a task. This requires the SC-server to know the workers' locations, which poses a privacy threat.

2.2 Differential Privacy

Differential Privacy (DP) [15] has emerged as the de-facto standard in data privacy, thanks to its strong protection guarantees rooted in statistical analysis. DP is a *semantic* model which provides protection against realistic adversaries with background information. Releasing data according to differential privacy ensures that an adversaryâĂŹs chance of inferring any information about an individual from the sanitized data will not substantially increase, regardless of the adversary's prior knowledge. DP ensures adversary do not know whether an individual is present or not in the original data. DP is formally defined as follows.

Definition 1 (ϵ-indistinguishability) *[16]*

Consider that a database produces set of query results $\hat{D}$ on the set of queries $Q = \{q_1, q_2, \ldots, q_{|Q|}\}$, and let $\epsilon > 0$ be an arbitrarily small real constant. Then, transcript U produced by a randomized algorithm A satisfies ϵ-indistinguishability if for every pair of sibling datasets D_1, D_2 that differ in only one record, it holds that

$$\ln \frac{Pr[Q(D_1) = U]}{Pr[Q(D_2) = U]} \leq \epsilon$$

In other words, an attacker cannot learn whether the transcript was obtained by answering the query set Q on dataset D_1 or D_2. Parameter ϵ is called *privacy budget*, and specifies the amount of protection required, with smaller values corresponding to stricter privacy protection. To achieve ϵ-indistinguishability, DP injects noise into each query result, and the amount of noise required is proportional to the *sensitivity* of the query set Q, formally defined as:

Definition 2 (L_1-Sensitivity) *[16] Given any arbitrary sibling datasets D_1 and D_2, the sensitivity of query set Q is the maximum change in the query results of D_1 and D_2*

$$\sigma(Q) = \max_{D_1, D_2} ||Q(D_1) - Q(D_2)||_1$$

There are multiple ways to achieve DP. One approach to achieving DP is adding random noise to each query result to preserve privacy, such that an adversary that attempts to attack the privacy of some individual w will not be able to distinguish from the set of query results whether a record representing w is present or not in the database [16]. An essential result from [16] shows that a sufficient condition to achieve differential privacy with parameter ϵ is to add to each query result randomly distributed Laplace noise with mean 0 and scale $\lambda = \sigma(Q)/\epsilon$.

Typically, the interaction with a dataset consists of a series of analyses A_i, each required to satisfy ϵ_i-differential privacy. Then, the privacy level of the resulting analysis is computed as follows:

Theorem 1 (Sequential Composition [41]) *Let A_i be a set of analyses such that each provides ε_i-DP. Then, running in sequence all analyses A_i provides $(\sum_i \varepsilon_i)$-DP.*

Theorem 2 (Parallel Composition [41]) *If D_i are disjoint subsets of the original database, and A_i is a set of analyses each providing ε_i-DP, then applying each analysis A_i on partition D_i provides* $\max(\epsilon_i)$*-DP.*

Chapter 3

Related Work

3.1 Spatial Crowdsourcing

Spatial Crowdsourcing (SC) has recently been attracting attention in both the research communities (e.g., [31, 12, 64, 61, 63]) and industry (e.g., TaskRabbit, Gigwalk [44]). Various aspects of spatial crowdsourcing have been studied, including task assignment [31, 10, 50, 68, 67, 77, 69, 61], task scheduling [12, 13], trust [32, 4], scalability [1], incentives [44, 57, 58, 62] and applications [3, 36, 66, 59]. A recent survey in this area can be found in [67], which distinguishes SC from related fields, including crowdsourcing, participatory sensing, volunteered geographic information.

3.2 Location Privacy

Location privacy has been studied first within the model of spatial k-anonymity [24, 34, 43, 18, 73, 14, 47], where the location of a user is hidden among k other users. Other studies focused on space transformation to preserve location privacy [33, 76]. Such techniques assume a centralized architecture with a trusted third party, which is a single point of attack. More recent work removes this assumption and provides cryptographic-strength protection [19]. While location privacy has largely been studied in the context of location-based services, only a few works have studied privacy in participatory sensing (PS) [30, 27, 28, 8]. The focus of [30] is to privately assign a set of spatial tasks to each worker while other works [27, 28] focus on preserving privacy in a PS campaign during the data contribution (i.e., how participants upload the collected data to the server without revealing their identities). Closest to our work is the approach from [8],

in which a privacy-preserving framework in WST mode is proposed, and the participants collect data in an opportunistic manner without the need to coordinate with the server. However, as discussed in Section 3.1, the WST mode yields poor results in practice. Furthermore, the privacy model in [8] does not provide formal privacy guarantees. In our work, we focus on the SAT case in the context of differential privacy, the de-facto standard for data publication.

3.3 Location Privacy in Spatial Crowdsourcing

Several recent works addressed the topic of location privacy in spatial crowdsourcing [51, 65, 21, 21, 78, 26, 38]. A recent survey in this area can be found in [49]. Particularly, the approach in [22] proposes a framework that can protect the workers' location privacy when allocating tasks to the workers. The solution adopts the privacy model used in our work, i.e., it assumes a trusted CSP and differentially private location sanitization. Similar to our study published in [64, 63], they develop analytical models and task allocation strategies that balance privacy, utility, and system overhead. The CSP can integrate mobile server location and reputation information.

The work in [78] studies reward-based spatial crowdsourcing that enables task assignment with optimized reward allocation. The authors also reuse our privacy framework introduced in [64], in which the SC-server and workers are connected by a trusted Cell Service Provider (CSP). The PSD receives exact locations from workers and releases a sanitized view to the SC-server, which is used to perform task assignment. The approach in [21] introduces a framework for protecting privacy of workers using location obfuscation techniques. This study assumes a trusted proxy to aggregate exact context (e.g., location) information from workers and output sanitized contexts to the SC-server. The authors develop an optimization model for task selection that maximizes the total expected utility of the server given constraints on privacy and efficiency. However, only worker statistics are protected under differential privacy, whereas locations

are obfuscated using cloaking techniques. Location cloaking is vulnerable to background knowledge attacks, and does not provide sufficient protection strength. In contrast, our work provides the strong semantic protection guarantees of differential privacy. Finally, the recent work in [38] addresses both location privacy and incentives in mobile sensing systems by proposing two credit-based privacy-aware incentive schemes. One of them assumes an online trusted third party, similar to our CSP-centered approach.

Other studies [51, 26] use cloaking technique (i.e., not distinguish among other k workers) thus do not provide rigorous privacy protection. Cloaking technique not only reveals workers' identities but also is prone to homogeneity attack [40] when all k workers are at the same location. The study in [38] uses cryptography, which is computationally expensive; thus, it is not designed for real-time spatial crowdsourcing applications.

Chapter 4

Privacy-Preserving Task Assignment in Spatial Crowdsourcing

4.1 Privacy Framework for Spatial Crowdsourcing

Section 4.1.1 presents the system model and the workflow for privacy-preserving SC. Section 4.1.2 outlines the privacy model and assumptions. Section 4.1.3 discusses design challenges and associated performance metrics.

4.1.1 System Model

We consider the problem of privacy-preserving SC *task assignment* in the SAT mode. Figure 4.1 shows the proposed system architecture. Workers send their locations (Step 0) to a trusted *cellular service provider (CSP)* which collects updates and releases a PSD according to privacy budget ϵ mutually agreed upon with the workers. The PSD is accessed by the *SC-server* (Step 1), which also receives tasks from a number of *requesters* (Step 2). For simplicity, we focus on the single-SC-server case, but our system model can support multiple SC-servers.

When the SC-server receives a task t, it queries the PSD to determine a *geocast region (GR)* that encloses with high probability workers close to t. Due to the uncertain nature of the PSD, this is a challenging process which will be detailed later in Section 4.3. Next, the SC-server initiates a *geocast* communication [45] process (Step 3) to disseminate t

to all workers within GR. According to DP, sanitizing a dataset requires creation of fake locations in the PSD. If the SC-server is allowed to directly contact workers, then failure to establish a communication channel would breach privacy, as the SC-server is able to distinguish fake workers from real ones. Using geocast is a unique feature of our framework which is necessary to achieve privacy protection. Geocast can be performed either with the help of the CSP infrastructure, or through a mobile ad-hoc network where the CSP contacts a single worker in the GR, and then the message is disseminated on a hop-by-hop basis to the entire GR. The latter approach keeps CSP overhead low, and can reduce operation costs for workers.

Upon receiving request t, a worker w decides whether to perform the task or not. If yes (Step 4), she sends a *consent* message to the SC-server (or requesters) confirming w's availability. If w is not willing to participate in the task, then no consent is sent, and no information about the worker is disclosed.

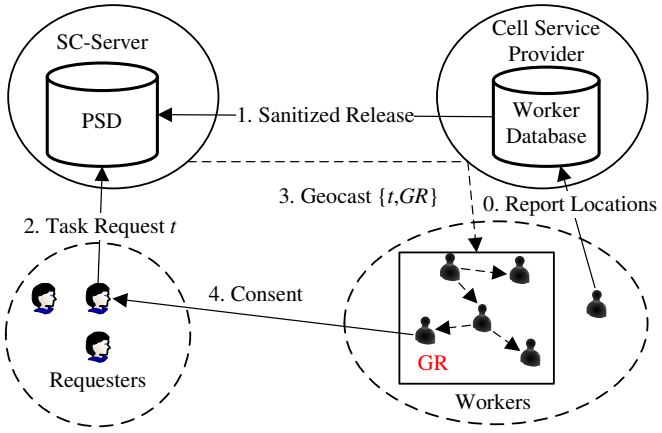


Figure 4.1: Privacy framework for spatial crowdsourcing

4.1.2 Privacy Model and Assumptions

Our specific objective is to protect both the *location* and the *identity* of workers *during task assignment*. Once a worker consents to a task, the worker herself may directly disclose information to the task requester (e.g., to enable a communication channel between

worker and requester). However, such additional disclosure is outside our scope, as each worker has the right to disclose his or her individual information. Our focus is on what happens prior to consent, when worker location and identity must be protected from *both* task requesters and the SC server.

We emphasize that focusing on the assignment step is the correct approach, given that SC workers have to physically travel to the task location. Mere completion of a task discloses the fact that some worker must have been at that location, and this is unavoidable in SC. To protect her location after consent, a worker can still enjoy some form of identity protection (e.g., using pseudonyms and anonymous routing), for which solutions are already available (e.g., TOR). On the other hand, no solution exists to date for the more challenging problem of privacy-preserving task assignment, hence we direct our efforts in this direction. Focusing on task assignment also makes sense from a disclosure volume standpoint. During assignment, all workers are candidates for participation, therefore locations of all workers are exposed, absent a privacy-preserving mechanism. On the other hand, after task request dissemination, only few workers will participate in task completion, and *only if* they give their *explicit consent*.

Workers cannot trust the SC-server, especially as there may be many such entities with diverse backgrounds, e.g., private companies, non-profits, government organizations, academic institutions. On the other hand, the CSP already has a signed agreement with workers through the service contract, so there is already a trust relationship established, as well as mutually-agreed upon rules for data disclosure. Furthermore, the CSP already knows where subscribers are, e.g., using cell tower triangulation, so worker location reporting does not introduce additional disclosure. In addition, having the CSP expose a PSD release of the user location dataset can benefit applications beyond crowdsourcing. For instance, the PSD can be shared with law enforcement agencies for public safety, or with commercial organizations to increase the revenue of the CSP. Therefore, there is sufficient motivation for the CSP to provide such a location sanitization service.

However, the CSP has no expertise, and perhaps no financial interest, to host an SC service, which needs to deal with a diverse set of issues such as interacting with various task requester categories, managing profiles (e.g., some workers may only volunteer for environmental tasks), etc. The role of the CSP is to aggregate locations from subscribed workers, transform them according to DP, and release the data in sanitized form to one or more SC-servers for assignment. As multiple SC-servers can use the same PSD, it is practical for the CSP to provide PSDs for a small fee, e.g., a percentage of the workers' payment, or a tax incentive in case of a public-interest SC applications.

4.1.3 Design Goals and Performance Metrics

Protecting worker location complicates significantly task assignment, and may reduce the effectiveness and efficiency of worker-task matching. Due to the nature of DP, it is possible for a region to contain no workers, even if the PSD shows a positive count. Therefore, no workers (or an insufficient number thereof) may be notified of the task request. The task may not be completed. Alternatively, the GR may comprise workers who are at a long distance away from the task location, whereas nearer workers are not included. Finally, in the non-private SAT case, only one selected worker, whose location and identity is known, is notified of the task request. With location protection, redundant messages need to be sent, increasing overhead. We focus on the following performance metrics:

- **Assignment Success Rate (ASR).** Due to PSD data uncertainty, the SC-server may incorrectly assign workers to tasks (e.g., no worker is reached, or task is too far and workers do not accept it). ASR measures the ratio of tasks accepted by a worker to the total number of task requests. The challenge is to maintain ASR close to 100%.

- **Worker Travel Distance (WTD).** The SC-server is no longer able to accurately evaluate worker-task distance, hence workers may have to travel long distances to

Symbol	Definition
ε, ε_i	Total privacy budget and level-i budget
α	AG budget split, $\alpha = 0.5$ means $\varepsilon_1 = \varepsilon_2$
N	Total number of workers
N'	Noisy worker count of level-1 cells
$m_i \times m_i$	Level-i grid granularity
$\bar{n}$	Expected noisy worker count of a level-2 cell
t	A task or its location, used interchangeably
c_i	A level-2 cell
n_{c_i}	Noisy worker count of c_i
$p^a_{c_i}$	Acceptance rate of workers within c_i
c'_i	Sub-cell of cell c_i

Table 4.1: Summary of Notations

tasks. The challenge is to keep the worker travel distance low, even when exact worker locations are not known.

- **System Overhead.** Dealing with imprecise locations increases the complexity of assignment, which poses scalability problems. A significant metric to measure overhead is the average number of notified workers (**ANW**). This number affects both the *communication overhead* required to geocast task requests, as well as the *computation overhead* of the matching algorithm, which depends on how many workers need to be notified of a task request.

4.2 Sanitizing Workers' Locations using Adaptive Grid

The first step in the proposed framework consists of building a PSD (at the CSP side) to be later used for task assignment at the SC-server. Building the PSD is an essential step, because it determines how accurate is the released data, which in turn affects *ASR*, *WTD* and *ANW*. We compared various PSD, including with k-d trees, quadtree and variations [7], adaptive grid [52] and h-tree [60]. We found h-tree structure (a hybrid between grid and k-d tree) can obtain good accuracy for skewed data; however, on average, the two-level grid performed better. Thus, in this section, we extend *Adaptive Grid (AG)* method to address the specific requirements of the SC framework. Table 5.1 summarizes the notations used in our presentation.

PSDs based on uniform grids treat all regions in the dataset identically, despite large variances in location density. As a result, they over-partition the space in sparse regions, and under-partition in dense regions. AG avoids these drawbacks by using a two-level grid and variable cell granularity. At the first level, AG creates a coarse-grained, fixed-size $m_1 \times m_1$ grid over the data domain. AG uses a data-independent heuristic to choose level-1 granularity as

$$m_1 = max(10, \frac{1}{4}\sqrt{\frac{N \times \epsilon}{k_1}})$$

where N is the total number of locations (this is considered known, but a high-precision estimate can also be found using a small fraction of ϵ). $k_1 = 10$ is suggested in [52].

Next, AG issues m_1^2 count queries, one for each level-1 cell, using a fraction of the total privacy budget: $\epsilon_1 = \epsilon \times \alpha$, where $0 < \alpha < 1$. AG then partitions each level-1 cell into $m_2 \times m_2$ level-2 cells, where m_2 is adaptively chosen based on the noisy count $N'$of the level-1 cell:

$$m_2 = \sqrt{\frac{N' \times \epsilon_2}{k_2}} \tag{4.1}$$

where $\epsilon_2 = \epsilon - \epsilon_1$ is the remaining budget, and the constant is set empirically in [52] to $k_2 = 5$. The budget parameter α determines how privacy budget is divided between the two levels, and a setting of $\alpha = 0.5$ is recommended [52] .

Figure 4.2 shows a snapshot of an adaptive grid, with four level-1 cells A,B,C,D. Constructing a differentially private AG requires two steps. First, the noisy counts N' of A,B,C,D are computed by adding random Laplace noise with mean $\lambda_1 = 2/\varepsilon_1$. Second, based on the noisy counts, level-1 cells are further split into level-2 cells. According to Eq. 4.1, cell D, which has noisy count 200 is partitioned according to a 3×3 grid, while the granularity for other cells is 2×2. Thereafter, AG adds to each level-2 cell (c_i, $i = 1 \ldots 21$) random Laplace noise with mean $\lambda_2 = 2/\varepsilon_2$. Finally, their corresponding noisy counts n_{c_i} together with the structure of the AG are published.

Although AG yields good results for general spatial queries [52], it is not directly applicable to SC, due to its rigidity in choosing parameters. Specifically, the granularity

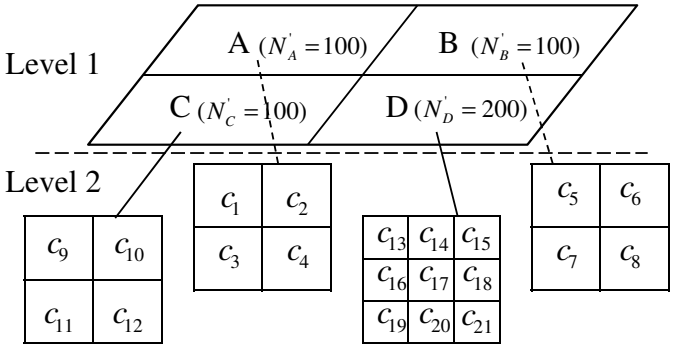


Figure 4.2: A snapshot of adaptive grid ($\varepsilon = 0.5$, $\alpha = 0.5$)

m_2 of the level-2 grid is too coarse, leading to large geocast areas and high communication overhead. According to Eq. (4.1), the expected number of workers (i.e., noisy count) in a level-2 cell is:

$$\bar{n} = N'/{m_2}^2 = k_2/\epsilon_2$$

Table 4.2a presents different values of m_2 and $\bar{n}$ when varying total budget ϵ with $\alpha = 0.5$.

ε	ε_2	m_2	$\bar{n}$
1	0.5	3	11
0.5	0.25	2	25
0.1	0.05	1	100

(a) Original AG ($k_2 = 5$)

ε	ε_2	m_2	$\bar{n}$
1	0.5	6	2.8
0.5	0.25	5	5.6
0.1	0.05	2	28.2

(b) Modified AG ($k_2 = \sqrt{2}$)

Table 4.2: Granularity m_2 and average count per cell $\bar{n}$ ($N' = 100$)

Note that, the values of $\bar{n}$ are rather large, especially for more restrictive privacy settings (i.e., lower ϵ). For $\epsilon = 0.1$, $\bar{n}$ is 100. In practice, a geocast region is likely to include multiple PSD cells, hence 100 is a lower bound on the ANW, while its typical values can grow much higher, leading to prohibitive communication cost.

We propose a more suitable heuristic for choosing k_2. Recall that the primary requirement of SC task assignment is to achieve high ASR. To that extent, we want to ensure that the task request is geocast in a non-empty region, i.e., the real worker count is strictly positive. According to the Laplace mechanism of DP, each PSD count is the sum of noisy and real counts. Given the level-2 privacy budget ϵ_2, we can also quantify the

distribution of added noise, which has standard deviation $\mu = \sqrt{2}/\epsilon_2$. Therefore, if the PSD count is larger than μ, then with high probability there will be at least one worker in the level-2 cell.

Our objective is to increase the granularity m_2 in order to decrease overhead, but only to the point where there is at least one worker in a cell. Denote by $count_{PSD}$ the value reported by PSD for a certain level-2 cell. Given the $Lap(1/\varepsilon_2)$ distribution, the probability that the noisy count is larger than zero is expressed as

$$p_h = 1 - \frac{1}{2} exp(-\frac{count_{PSD}}{1/\epsilon_2})$$

Furthermore, we want to have the PSD count larger than the noise, i.e., $\bar{n} = k_2/\varepsilon_2 \geq \sqrt{2}/\varepsilon_2$, so at the limit we set $k_2 = \sqrt{2}$. The resulting probability of having non-empty cells is $p_h = 1 - \frac{1}{2} exp(-\sqrt{2}) = 0.88$. According to Eq. (4.1), the corresponding granularity is $m_2 = \left\lceil \sqrt{N'\varepsilon_2/\sqrt{2}} \right\rceil$.

In summary, we adapt AG by carefully reducing the granularity threshold at level-2 such that ANW is reduced, while the probability for each level-2 cell to contain a real worker is at least 88%. Table 4.2b shows that this new setting significantly reduces $\bar{n}$, and as a result ANW.

4.3 Performing Task Assignment on Sanitized Data

When a request for a task t is posted, the SC-server queries the PSD and determines a geocast region GR where the task is disseminated. The goal is to obtain a high success rate for task assignment, while at the same time reducing the worker travel distance WTD and request dissemination overhead ANW.

4.3.1 Task Localness and Acceptance Rate

Travel distance is critical in SC, as workers need to physically visit the task locations. Workers are more likely to perform tasks closer to their home or workplace [44, 31, 2].

The work in [44] shows that 10% of all workers, denoted as *super-agents*, perform more than 80% of the tasks. Among super-agents, 90% have daily travel distance less than 40 miles, and the average travel distance per day is 27 miles. This property is referred to as *task localness* [31]. A related study [25] addresses the localness of contents posted by Flickr and Wikipedia users, and proposes a *spatial content production model (SCPM)* that computes the *mean contribution distance (MCD)* of each worker as follows:

$$MCD(w_i) = \sum_{j=1}^{n} \frac{d(L_{w_i}, L_{c_j})}{n} \tag{4.2}$$

where $L(w_i)$ is the location of worker w_i, and L_{c_j} are the locations of its n contributions.

Based on Eq. (4.2), we can find the *maximum travel distance (MTD)* that a high percentage of workers are willing to travel to perform their assigned tasks. For example, *MTD* of super-agents in crowdsourcing markets studied in [44] is 40 miles with 90% cumulative ratio of contributors. Besides communication overhead, task localness is thus another reason to impose an upper bound on geocast region size. Intuitively, the maximum geocast region is a square area with side size equal to $2 \times MTD$. Hereafter, we refer to *MTD* as both the maximum travel distance and the maximum geocast region size.

We denote by *acceptance rate (AR)* the probability $p^a (1 \leq p^a \leq 1)$, that a worker accepts to complete a task for which s/he receives a request. We assume that all workers are identical and independent of each other in deciding to perform tasks. The work in [44] studies reward-based SC labor markets and shows that super agents have an average AR of 90.73% while other agents have an acceptance rate of 69.58%. Acceptance rate is much smaller in self-incentivized SC [31], where the workers voluntarily perform tasks, without receiving incentives.

A worker is more willing to accept nearby tasks, so we model acceptance rate as a decreasing function of travel distance. We consider two cases: *(i) linear*, where AR decreases linearly with distance starting from an initial *MAR (Maximum AR)* value

(when the worker is co-located with the task) and *(ii) Zipf*, where acceptance rate follows Zipf distribution with skewness parameter s. The higher s is, the faster p^a drops. p^a is maximized when the worker is co-located with the task and becomes negligible at MTD. If the distance is larger than MTD, $p^a = 0$.

4.3.2 Analytical Utility Model

We develop an analytical *utility* model that allows the SC-server to quantify the probability that a task request disseminated in a certain GR is accepted by a worker. Intuitively, the utility depends on the AR and on the worker count $\bar{w}$ estimated to be enclosed within GR. An SC-server will typically establish an *expected utility* threshold EU which is the targeted success rate for a task (this is a system goal, rather than an outcome). Generally, EU is considerably larger than an individual worker's p^a, so the GR must contain multiple workers.

We define X as a random variable for the event that a worker accepts a received task: $P(X = True) = p^a$ and $P(X = False) = 1 - p^a$. Assuming w independent workers, $X \sim Binomial(w, p^a)$. We define the *utility* of a geocast region covering w workers as:

$$U = 1 - (1 - p^a)^w \tag{4.3}$$

U measures the probability that at least one worker accepts the task. The utility definition can be extended to the case of redundant task assignment, where multiple workers are required to complete a task (refer to Section 4.3.5).

4.3.3 Geocast Region Construction

Given task t, the GR construction algorithm must balance two conflicting requirements: determine a region that *(i)* contains sufficient workers such that task t is accepted with high probability, and *(ii)* the size of the geocast region is small. The input to the

algorithm is task t as well as the worker PSD, consisting of the two-level AG with a noisy worker count for each grid cell.

The algorithm chooses as initial GR the level-2 cell that covers the task, and determines its U value. As long as utility is lower than threshold EU, it expands the GR by adding neighboring cells. Cells are added one at a time, based on their estimated increase in GR utility. Following the task localness property, we take into account the distance of each candidate neighboring cell to the location of t, and give priority to closer cells. The algorithm stops either when the utility of the obtained GR exceeds threshold EU, or when the size of GR is larger than MTD, hence utility can no longer be increased. The GR construction algorithm is a greedy heuristic, as it always chooses the candidate cell that produces the highest utility increase at each step.

Algorithm 1 Greedy Algorithm (GDY)

1: Input: task t, MTD, $0<EU<1$
2: Output: geocast region GR
3: MTD is the square of size $2 \times MTD$ centered at t
4: Init $GR = \{\}$, utility $U = 0$
5: Init max-heap $Q = \{$level-2 cell that covers $t\}$
6: Remove $\{c_i, U_{c_i}\} \leftarrow Q$, U_{c_i} is computed from Eq. (4.3)
7: If $c_i = Nil$, return GR {GR is larger than MTD}
8: $GR = GR \cup c_i$
9: If $U_{c_i} \geq 0$, $U = 1 - (1 - U)(1 - U_{c_i})$
10: If $U \geq EU$, return GR
11: Find $neighbors = \{\{c_i's\ neighbors\} - GR\} \cap MTD$
12: $Q = Q \cup neighbors$
13: Goto Line 6

The pseudocode of the greedy algorithm is depicted in Algorithm 1. In Line 5, Q is a heap of cells $\{c_i\}$, sorted decreasingly according to cell utility U_{c_i}. U_{c_i} is computed according to Eq. (4.3), namely $U_{c_i} = 1 - (1 - p^a_{c_i})^{n_{c_i}}$, where n_{c_i} is the noisy worker count of c_i, and $p^a_{c_i}$ is the acceptance rate of the workers inside c_i. Since worker locations within a cell are not known, we assume they all have the same acceptance rate. Moreover, we assume the worker-task distance is equal to the average distance between the task and each four corners of cell c_i.

When a candidate cell is removed from Q (Line 6), it is added to GR (Line 8), and GR utility is updated in Line 9. Updated utility U' is the probability that a worker in either the current geocast region or the newly added cell performs the task:

$$U' = U(1 - U_{c_i}) + (1 - U)U_{c_i} + UU_{c_i} = 1 - (1 - U)(1 - U_{c_i})$$

Line 11 computes the new neighboring cells that are not in GR, and are not situated farther than MTD. These cells are added to Q according to their utilities. If a cell resides partially outside MTD, it is pruned to its fraction contained within the MTD, and its noisy count is updated proportionally to the pruned area.

In summary, the geocast region construction algorithm greedily expands the GR by choosing to include at each step the grid cell that results in the highest estimated increase in utility. Cell utility takes into account the noisy worker count, as well as the distance between the cell and the task location. Next, we consider two refinements to the heuristic: first, we investigate a finer-grained solution search space by allowing partial cell inclusion in Section 4.3.4; second, we consider the effect that the GR shape has on hop-by-hop task request dissemination in Section 4.3.6.

4.3.4 Partial Cell Selection

Even though the adaptation of AG proposed in Section 4.2 significantly reduces the granularity of level-2 cells, the number of workers can still be rather large, and the resulting ANW can lead to high task request dissemination costs. Such workers may be unnecessarily included in the GR, even if the required EU could be achieved with far fewer workers. We propose an optimization that allows partial inclusion in the GR of a level-2 cell.

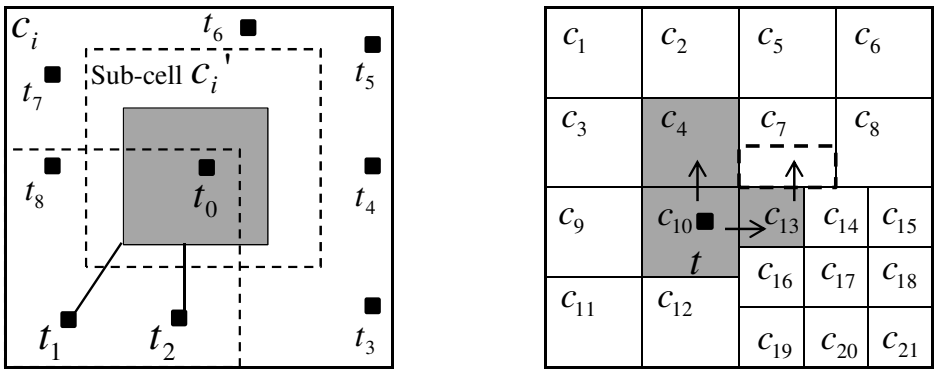
Before adding a new cell c_i to the GR (Line 10 of Algorithm 1), the optimization checks whether the utility increase provided by c_i will exceed the required utility EU. If so, the algorithm computes a sub-region of c_i whose utility is sufficient to reach EU.

The pseudocode of the heuristic is depicted in Algorithm 2, which includes two steps. First, it computes the percentage of c_i's area (Lines 3-7) that is likely to enclose sufficient users. Next, it finds a sub-cell with that area (Lines 8-9) which is uniquely determined by its shape and location. The optimization in Algorithm 2 can be inserted as a function call before Line 10 in the main Algorithm 1. To compute a sub-cell, two constraints

Algorithm 2 Partial Cell Selection Heuristic

1: Input: task location t, last cell c_i, current utility U_{curr}
2: Output: *sub-cell* c_i' of c_i
3: $dist = distance(t, c_i)$
4: $p^a_{sub} = acc\_rate(dist)$
5: $U_{required} = \frac{EU - U_{curr}}{1 - U_{curr}}$
6: Worker count needed to achieve $w_{required} = \log^{1-U_{required}}_{1-p^a_{sub}}$
7: Area $percentile = w_{required}/w_{c_i}$
8: If c_i covers t, find *sub-cell* given area *percentile*
9: Otherwise, find *sub-cell* adjacent with current region

need to be satisfied. First, the sub-cell needs to be completely inside the parent cell. Second, the sub-cell must be adjacent with the current GR to form a continuous region. Therefore, depending on whether or not the current GR contains one or multiple cells, we use two strategies to find the sub-cell.



(a) Case 1: splitting c_i (b) Case 2: splitting cell 7

Figure 4.3: Examples of partial cell selection.

Figure 4.3a depicts the first case where the GR includes only one grid cell c_i (i.e., the task t_0 is inside c_i, the parent cell). Intuitively, to cover the closest workers to the task, the shape of the sub-cell c_i' (dashed line) must be a square. The boundary of cell

c_i' can therefore be completely determined given its area. To satisfy the first constraint, the center of c_i' needs to be in the shaded square, whose center is the same as that of c_i, and its size is equal to the difference between the side lengths of c_i and c_i'. In addition, the position of c_i' is such that the distance between its center and the task is minimized. The distance is zero when the task (e.g., t_0) is inside the shaded region (the task is co-located with c_i''s center). Otherwise, if the task is outside the shaded square, its closest sub-cell's center must be on the border of the shaded square. Subsequently, depending on the relative position of the task to the shaded circle (i.e., eight possibilities t_1-t_8), we can find the corresponding sub-cell's center. For example, the closest sub-cell's center of t_1 is the left bottom corner of the shaded square.

Figure 4.3b presents the second case, in which the *GR* comprises of multiple cells {4,7,10,13}. This example is a flat version of the AG in Figure 4.2. The arrows in the figure depict the expansion process of the geocast algorithm. For example, cells 4 and 13 are expanded from cell 10 while cell 7 is expanded from cell 13. To ensure the *GR* is a continuous region, we require the long edge of the sub-cell (dashed rectangle) to be adjacent to the neighbor cell (i.e., 13) that the splitting cell (i.e., 7) is expanded from. When its long edge is fixed, the sub-cell is uniquely specified given its area. The rationale behind this choice is to ensure the continuity constraint.

4.3.5 Redundant Task Assignment

In some spatial crowdsourcing applications, workers cannot always be trusted [32]. Thus, we extend our technique to tackle the issue of trust by having tasks performed redundantly by multiple workers. Equation 4.3 can be extended as follows:

$$U = 1 - \sum_{i=1}^{K} U^i = 1 - \sum_{i=1}^{K} \binom{w}{i} (p^a)^i (1 - p^a)^{w-i} \tag{4.4}$$

where K is the number of workers required to perform the task and $U^i = \binom{w}{i}(p^a)^i(1 - p^a)^{w-i}$ is the probability that exactly i workers perform the task.

The geocast algorithm can be extended to the case where at least K workers in either the current geocast region GR or the newly added cell c_i perform the task. The updated utility can be computed based on the probability of having at most $K-1$ workers perform the task. Particularly, Line 9 of Algorithm 1 can be updated as follows:

$$U' = 1 - \sum_{j=0}^{K-1} U_{c_i}^{j} \sum_{l=0}^{K-j-1} U^{l} \tag{4.5}$$

where $U_{c_i}^{j} = \binom{n_{c_i}}{j}(p^{c_i})^{j}(1-p^{c_i})^{n_{c_i}-j}$ is the probability that exactly j workers in c_i perform the task and $\sum_{l=0}^{K-j-1} U^{l}$ is the probability that at most $K-j-1$ workers in GR perform the task. Note that $U_{c_i}^{j} = 0$ if $j > n_{c_i}$.

The partial cell selection heuristic can be modified such that a sufficient number of workers $w_{required}$ will be selected from c_i to ensure that the updated utility in Equation 4.5 is larger than or equal to the expected utility $U' \geq EU$. Particularly, Lines 5 and 6 in Algorithm 2 can be placed within a loop to evaluate the updated utility: **for** w **from 1 to** n_{c_i} **do:** $evaluate(U')$, where w is the number of workers in c_i to be selected.

4.3.6 Communication Cost

Dissemination of a task request within the GR can be implemented in two ways:

- *Infrastructure-based Mode.* In this mode, the CSP sends an individual message to each worker within the GR. The cost is proportional to ANW, which may be large.
- *Infrastructure-less Mode.* Workers within the GR can relay the task request hop-by-hop, using a mobile ad-hoc network protocol over WiFi or Bluetooth. In this case, the CSP only needs to send several messages to workers (one single message may suffice if the worker network is connected).

Geocasting using hop-by-hop communication is an attractive alternative. The SC-server does not know the actual worker placement, so the GR construction strategy cannot rely on detailed routing information, but fortunately, the shape of the GR is

often a good predictor of ad-hoc routing performance. Intuitively, it is cheaper to geocast within a shape with less skew, such as a circle or a square, as opposed to skewed regions such as line-shaped areas, which have large network diameter. For instance, in Figure 4.3b, the region of cells {1,2,3,4} is more favorable for geocast than {2,4,5,6}, despite the fact that the two areas have equal size.

We assume that the geocasting cost is proportional to the minimum bounding circle that covers the *GR*. Thus, the more *compact* the *GR*, the lower the cost. Several measures of compactness for two-dimensional shapes are discussed in [39]. One widely accepted measure proposed in [35] is the *Digital Compactness Measurement (DCM)*, which measures region compactness as the ratio between the area of the region and the area of its smallest circumscribing circle. An efficient solution to find the smallest enclosing circle is a randomized algorithm [70] that runs in linear time to the number of data points in the region. The maximum value of *DCM* is 1 when the shape is a circle.

We modify Algorithm 1 to choose new cells to add to *GR* based on compactness, instead of utility. At each iteration, the cell that increases the compactness of the *GR* most is chosen from the list of candidates. Due to the inclusion of the new cell, the potential compactness increase of all other candidates may need to be re-computed, to account for the change in shape. We also consider a hybrid method that factors in both utility and compactness in cell selection. The merit function of the hybrid is a linear combination of the resulting *GR* utility and compactness.

To evaluate the effectiveness of using compactness in the *GR* search strategy, we use as metric an estimation of the hop count required to disseminate the task request to all workers, given the communication range of the wireless network (e.g., 50-100 meters for WiFi). We approximate the hop count as the diameter of the network divided by the communication range:

$$Hop\ count = \frac{farthest\ distance\ between\ two\ workers}{2 \times communication\ range} \tag{4.6}$$

In practice, the worker network needs to be connected for the ad-hoc geocast to succeed. A message from any worker (i.e., seed) should be able to reach any other in the GR, using hop-by-hop wireless communication. Otherwise, if the network contains multiple disconnected components, the task cannot be sent to all workers from a single seed. In the latter case, the CSP would need to send the task to multiple seeds within the ad-hoc network. However, this level of detail goes beyond the scope of our work, and we restrict ourselves to using the hop count metric as an estimation of geocast cost, in conjunction with ANW.

4.4 Protection for Dynamic Workers' Locations

Spatial crowdsourcing systems receive continuously requests for task assignment. Hence, it is important to keep track of whereabouts of *moving* users and to release a *sequence* of worker PSDs that allow effective spatial task assignment over multiple timestamps. In this section, we extend our solution for single-snapshot PSDs to the case of dynamic worker datasets. First, in Section 6.1, we formulate the problem, outline a baseline solution and highlight its limitations. Next, in Section 6.2 we propose a technique for effective multiple-snapshot spatial task assignment.

4.4.1 Problem Statement and Baseline Solution

We assume a discrete time model, where the SC server receives and processes task requests at T discrete timestamps, $\{t_1, \ldots, t_T\}$. The SC generates a sequence of snapshot PSDs $\{PSD_1, \ldots, PSD_T\}$, one for each timestamp t_k, and processes each task assignment request received at timestamp t_k according to private decomposition PSD_k.

Since many (or all) of the workers appear in multiple PSDs at distinct timestamps, an adversary has more opportunities to breach the location privacy of workers by correlating information from consecutive PSDs. In the context of differential privacy, this increased

knowledge available to the adversary is modeled as an increase in L_1-sensitivity. Specifically, in the worst case, the sensitivity grows from 2 for a single snapshot to $2 \times T$ for a set of T released PSDs. On the other hand, the amount of privacy budget ϵ remains unchanged. Hence, the problem of privacy-preserving spatial task assignment for multiple snapshots is significantly more challenging than its single-snapshot counterpart. Formally, the problem we solve is defined as:

Problem Statement. Given a set of discrete timestamps $\mathcal{T} = \{t_1, \ldots, t_T\}$, $T > 1$, and a privacy budget ϵ, release a set of private spatial decompositions $\{PSD_1, \ldots, PSD_T\}$ that satisfies ϵ-differential privacy and supports effective spatial task assignment as measured by performance metrics ASR, WTD and ANW.

A baseline solution, denoted as *BasicD* (*basic dynamic*), is to repeatedly apply our single-snapshot PSD algorithm at every timestamp, disregarding correlation between data at different timestamps. At each timestamp, only a fraction $\frac{\epsilon}{T}$ of the entire privacy budget is used, which guarantees that the overall algorithm satisfies ϵ-differential privacy due to the sequential composition theorem (Section 2.2). The *BasicD* pseudocode is presented in Algorithm 3. However, as T grows, the amount of budget received by the PSD at each timestamp decreases, resulting in a high amount of noise that needs to be added to each worker count. Consequently, the released sanitized data become unusable, and the quality of task assignment decreases significantly. Next, we propose a technique that addresses the limitations of *BasicD*.

Algorithm 3 BasicD Algorithm

1: **Input:** T, worker location datasets $\{D_1, ..., D_T\}$, budget ϵ
2: **Output:** $\{PSD_1, \ldots, PSD_T\}$
3: For $k = 1$ to T:
4: Publish k^{th} PSD of D_k with budget ϵ/T

4.4.2 Multiple-Snapshot Worker PSD

We build upon our single-snapshot solution and augment it with an adaptation of FAST [17], a recently-proposed approach to continuously release private counts on top of

adaptive grids. To characterize the dynamicity of the data, FAST constructs an internal process model that captures the temporal correlation of aggregated location traces. It improves accuracy of released data at each timestamp by performing posterior estimation to reduce perturbation error. Also, when T is large, FAST samples the time series of aggregate data to reduce overall perturbation cost.

The main idea of the proposed dynamic WorkerPSD algorithm is to spend a fraction of the privacy budget to build the structure of the adaptive grid in the first time instance. During subsequent time instances, the AG structure is unchanged, and only the counts of the level-2 cells are updated, according to the new configuration of the dataset. Figure 4.4 illustrates how FAST is integrated within the proposed WorkerPSD framework, and Algorithm 4 provides the pseudocode. At timestamp t_1, the algorithm releases the noisy counts of level-2 cells using the Laplace mechanism. In each of the following timestamps, the workerPSD structure aggregates the new *true count* and the old *published counts* of every level-2 cell as the input to the FAST component.

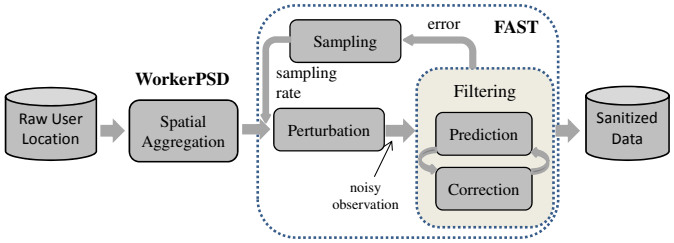


Figure 4.4: Workflow for Dynamic Worker PSD Computation

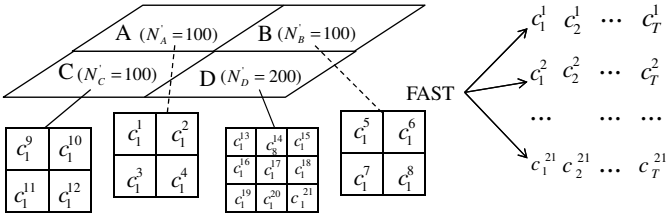


Figure 4.5: Two-level Grid for Dynamic WorkerPSD

Figure 4.5 illustrates the two-level grid decomposition representing the worker PSD. At the first timestamp, a fraction $\epsilon_1 = f \times \epsilon$ of the privacy budget is used to determine the

geometry of the level-2 grid (Line 3 in Algorithm 4). The resulting grid decomposition is represented by the cells with continuous borders in Figure 4.5. The remaining budget, amounting to $\epsilon_2 = (1 - f) \times \epsilon$, is used to release noisy counts for timestamps t_1 to t_T, according to the FAST approach (Lines 4-8). With adaptive grid, level-2 cells do not overlap, so according to [7], each set of counts corresponding to the same cell at distinct timestamps receives budget ϵ_2 (Line 7).

Algorithm 4 Dynamic Worker PSD

1: **Input:** T, locations $\{D_1, ...D_T\}$, budget ϵ
2: **Output:** *PSD {each second-level cell maintains a list of noisy counts}*
3: Build *PSD* structure of D_1 given budget $\epsilon \times f$
4: For c^i in second-level cells of *PSD*:
5: For $k = 1$ to T:
6: c_k^i = actual count of c^i in D_k
7: Apply FAST to $\{c_1^i, c_2^i, ..., c_T^i\}$ given budget $(1 - f) \times \epsilon$
8: *PSD* $\leftarrow$ update noisy counts $\{n_{c_1^i}, n_{c_2^i}, ..., n_{c_T^i}\}$ of *PSD*

Choosing an appropriate f value. A larger f value allocates more budget for PSD construction in the initial timestamp, which increases the number of level-2 cells, providing finer granularity. However, a larger f also reduces the budget for later timestamps, which may lead to high noise-to-real-count ratio. To balance these two factors, we propose a simple analytical model to select f.

We assume a simplified version of Algorithm 4 where the noisy counts of individual PSD cells are obtained by adding Laplace noise directly with privacy budget $\frac{(1-f)\epsilon}{T}$. Denote by $count_{PSD}$ the value reported by PSD for a certain level-2 cell. Given the $Lap(\frac{2T}{(1-f)\epsilon})$ distribution, denote the probability that the expected noisy count is larger than zero by

$$p_h = 1 - \frac{1}{2} exp(-\frac{count_{PSD}(1 - f)\epsilon}{2T})$$

Our goal is to have PSD count larger than the expected noise. From Section 4.2, the average number of workers in a level-2 cell is $\bar{n} = \sqrt{2}/\varepsilon_2 = 2\sqrt{2}/(f\epsilon)$. Thus,

$$p_h = 1 - \frac{1}{2} exp(-\frac{\sqrt{2}(1 - f)}{Tf})$$

Figure 4.6 illustrates the analytical dependence of p_h on f. Note that p_h asymptotically decreases to 0.5 when f increases. This result suggests that to avoid excessive noise-to-real-count ratio, especially when T is large, f should be set to a relatively small value, e.g., $f =\in [0.1, 0.2]$. In Section 4.5, we explore the effect of varying f on system accuracy and performance.

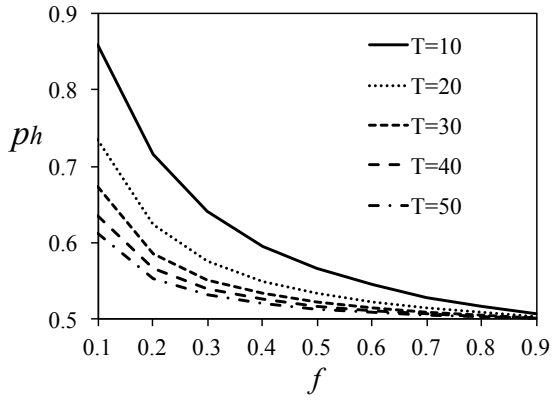


Figure 4.6: The probability p_h is greater than zero when varying f.

4.5 Performance Evaluation

4.5.1 Experimental Methodology

We use two real-world datasets: Gowalla and Yelp. Gowalla contains the check-in history of users in a location-based social network. For our experiments, we use the check-in data in the area of San Francisco, California. We assume that Gowalla users are the workers of the SC system, and their locations are those of the most recent check-in points. We also model each check-in point as a task that was previously accepted for execution by a worker. Based on this model, we determine the mean contribution distances ($MCDs$) according to Eq. (4.2), and we compute maximum travel distance (MTD) as the 90% MCD percentile value, leading to a value of $3.6km$. The Yelp data corresponds to the greater area of Phoenix, Arizona, and includes locations of $15,583$ restaurants, $70,817$ users and $335,022$ user reviews. We use restaurant locations as tasks,

Name	Tasks	Workers	MTD	Workers/km^2
Gowalla (Go.)	151,075	6,160	3.6	35
Yelp (Ye.)	15,583	70,817	13.5	4

Table 4.3: Dataset Characteristics

and a user review is equivalent to accepting a SC task. Table 4.3 presents the details of the datasets.

For moving users experiments, we divide each dataset into T time instances based on the time of activities (i.e., review time for Yelp and check-in time for Gowalla). Using this methodology, 10-30% of the worker locations are updated at each timestamp. If a user has no activity at a time instance, his location remains the same as in the previous instance.

To evaluate the overhead of privacy, we compare our proposed solution with a non-private algorithm that has access to exact worker locations. Given a task and the *actual* worker locations, the algorithm keeps adding nearby workers one by one ($1NN$, $2NN$, etc.) until the obtained utility exceeds threshold EU, or until the size of the GR is larger than MTD. The geocast query is the minimum bounding circle of the nearest workers.

We consider privacy budget $\epsilon \in \{0.1, .., \mathbf{0.5}, .., 1\}$, ranging from strict to loose privacy requirements. We set the expected utility $EU \in \{0.3, 0.5, 0.7, \mathbf{0.9}\}$ and the maximum acceptance rate $MAR \in \{\mathbf{0.1}, 0.3, 0.5, 0.7\}$. We vary the number of time instances $T \in \{50, 60, ..., \mathbf{100}\}$ and the budget percentile for grid structure $f \in \{\mathbf{0.1}, 0.2, .., 1\}$. We vary the number of workers required to perform a task $K \in \{\mathbf{1}, 2, 3, 4, 5\}$. Default values are shown in bold. For Zipf acceptance rate, skew parameter s is set to 1. Wireless communication range is 50 meters.

We randomly generated $1,000$ tasks and measured the performance of the proposed solution with respect to the metrics in Section 4.1.3: ASR, ANW, and WTD. To compute ASR, we simulate a binomial model as discussed in Section 4.3.1. Each worker flips a biased coin and decides whether to accept or not a received task request, based on personalized threshold p^a (recall that p^a takes into account distance to task). A task is accepted if at least one worker agrees to perform it. We consider two WTD variations:

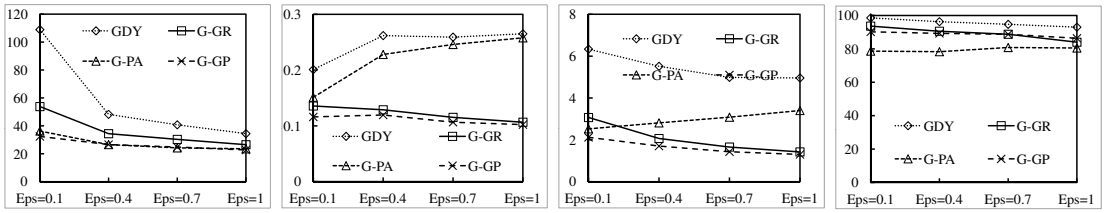
WTD_{NN} (default case) and WTD_{FC}. In the former case, the metric value is determined as the distance from the task to the nearest worker that accepts the task, whereas in the latter, the distance to the first worker that accepts the task is considered (i.e., first-come). We also measure the average hop count *HOP* required for geocast, according to Eq. (4.6). Finally, we also show the results obtained for the average number of cells in a *GR* (*CELL*) and the compactness of the *GR*. Although these metrics are not directly perceived by the end users, they help us to better understand the underpinnings of the proposed solution. All measured results are averaged over ten random seeds.

4.5.2 Experimental Results

Evaluation of Single-snapshot PSD

Evaluation of GR Construction Heuristics: We evaluate the performance of the greedy algorithm for *GR* construction from Section 4.3.3 and its variations. *GDY* refers to the algorithm using the original AG PSD from [52], whereas *G-GR* uses our customized AG solution. The optimization allowing partial cell selection is denoted by *G-PA*, and the combination of both *G-GR* and *G-PA* by *G-GP*. Figure 4.7 illustrates the results.

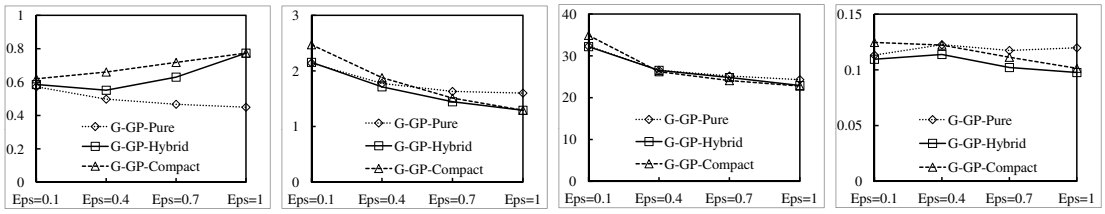
G-GP generally performs best in terms of minimizing *ANW*, *WTD* and *HOP* in all combination of datasets (Gowalla, Yelp) and acceptance rate functions (Linear, Zipf). Moreover, by comparing *G-GP* and *G-PA* with *GDY* and *G-GR*, we observe that customized AG granularity contributes mostly to the improvements. Partial cell selection proves useful mostly when the privacy budget is small (i.e., resulting grid is coarse). Compared to *GDY*, *G-GP* reduces *ANW* by up to a factor of 5, and the improvement is more significant when privacy budget is low. Specifically, increasing ϵ provides a more accurate estimation for the worker counts in the PSD, and also the granularity of the level-2 AG grows. As a result, *ANW* can be more tightly controlled. Moreover, *G-GP* also yields reduced *WTD* and *HOP* by up to a factor of 8 and 7, respectively.



(a) *ANW*, Go.-Linear (b) *WTD*,Go.-Linear (c) *HOP*, Go.-Linear (d) *ASR*, Go.-Linear

Figure 4.7: Comparison of *GR* construction heuristics by varying ε.

Evaluation of Compactness-Based Heuristics: We evaluate the effect of the compactness-guided heuristic for *GR* construction. For brevity, we only include Gowalla results (Yelp dataset shows similar trends). As shown in Figure 4.8, the compactness-based approach (*G-GP-Compact* significantly increases the compactness measure compared to its utility-based counterpart (*G-GP-Pure*). The hop count is also reduced, by up to 36%, particularly when the privacy budget is large. However, the compactness-only approach does not fare that well for lower privacy budgets. On the other hand, the hybrid heuristic that combines utility and compactness in the ranking of candidates (*G-GP-Hybrid*) manages to perform better than its counterparts for all ϵ values. We conclude that such a balanced approach is the best solution for *GR* construction.

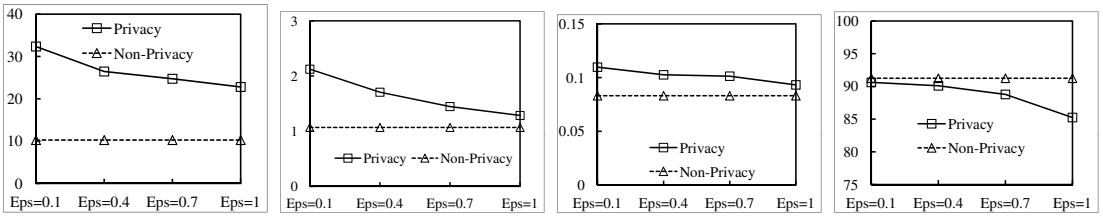


(a) *CMP*, Go.-Linear (b) *HOP*, Go.-Linear (c) *ANW*, Go.-Linear (d) *WTD*,Go.-Linear

Figure 4.8: Comparison of compactness-based heuristics by varying ε.

Overhead of Achieving Privacy: We compare the proposed solution with the non-private algorithm for task assignment, described in Section 4.5.1. Figure 4.9 presents the overhead incurred by privacy when varying ϵ (for brevity, we only show Gowalla results). As expected, when ϵ increases, the PSD offers more accurate data, and the overhead (in terms of *ANW*, *WTD* and *HOP*) decreases. Interestingly though, *ASR* drops in value. This can be explained through significant over-provisioning that occurs

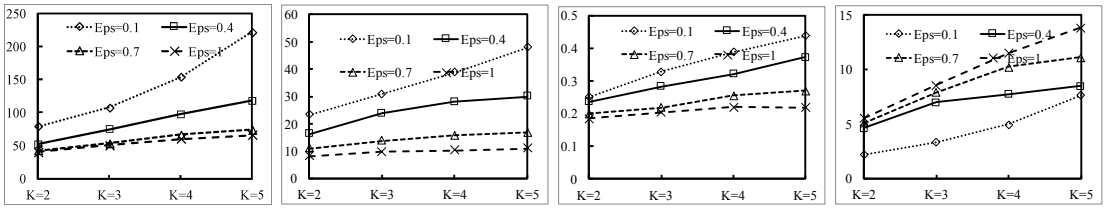
for lower budgets, when the greedy heuristic enlarges the GR in the quest for achieving the desired EU. As a result, more workers are notified, and the chances of task acceptance are higher. However, overhead is also much higher. We also observe that privacy does not significantly increase WTD, proving that the greedy GR algorithm does a good job in selecting nearby workers.



(a) *ANW*, Go.-Linear (b) *HOP*, Go.-Linear (c) *WTD*, Go.-Linear (d) *ASR*, Go.-Linear

Figure 4.9: Overhead of privacy (*G-GP-Hybrid*) compared to non-private algorithm.

Effect of Varying K: We evaluate the performance of *G-GP-Hybrid* on the Yelp dataset by varying the number of workers required to complete a task. Figure 4.10 shows the results. As expected, higher K yields higher overhead as more workers are required to perform a task. Also, we observe that as the privacy budget ϵ gets larger, the impact of K on the performance metrics attenuates.



(a) *ANW*, Ye.-Linear (b) *HOP*, Ye.-Linear (c) *WTD*, Ye.-Linear (d) *CELL*, Ye.-Linear

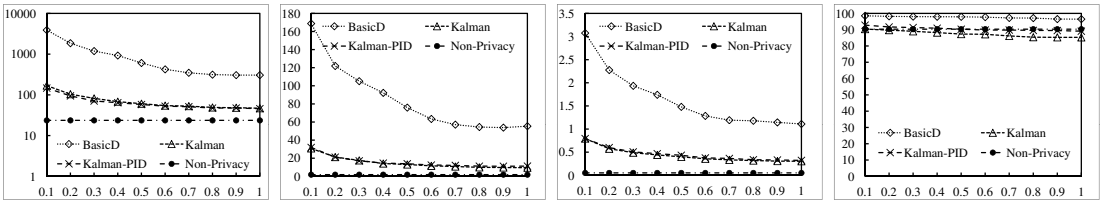
Figure 4.10: Performance of the geocast algorithm (*G-GP-Hybrid*) when varying the number of workers required to complete a task (Ye.-Linear).

Evaluation of Multiple-snapshot PSD

Effect of Varying ϵ: We evaluate the performance of the proposed algorithms for dynamic WorkerPSD from Section 4.4. In our implementation, we use the *G-GP-Hybrid* as a single-snapshot base for the dynamic case, and we consider two multiple-snapshot

instances: the first uses the Kalman filtering without sampling (denoted as *Kalman*) and the second uses Kalman filtering in conjunction with the PID adaptive sampling (denoted as *KalmanPID*). We compare the obtained results against two benchmarks: the non-private case, and the *BasicD* baseline introduced in Section 6, which divides privacy budget equally among all time instances. For brevity, we present the results only for linear acceptance rate function, as similar results have been observed for the zipf case (the focus of this experiment is on user movement, so the choice of acceptance rate function does not have a significant impact).

Figure 4.11 presents the performance metrics measurements of the considered methods when varying privacy budget ϵ. As expected, when ϵ increases, the PSD offers more accurate data, and the overhead (in terms of *ANW*, *WTD* and *HOP*) decreases. We observe that both Kalman-based algorithms are superior to *BasicD* by a significant amount. Furthermore, their performance is not far behind that of the non-private approach. Interestingly, the *BasicD* baseline obtains a high *ASR*, but this occurs due to excessive over-provisioning at lower budgets. In such cases, the greedy heuristic enlarges the *GR* to achieve the desired utility, and the overhead is very high.



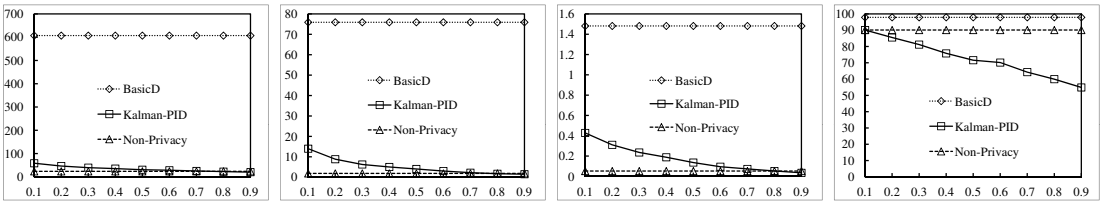
(a) *ANW*, Ye.-Linear (b) *HOP*, Ye.-Linear (c) *WTD*, Ye.-Linear (d) *ASR*, Ye.-Linear

Figure 4.11: Performance Overhead for Multiple-snapshot WorkerPSD when varying privacy budget ε.

Effect of Varying f: Next, we investigate the performance of the multiple-snapshot algorithm when varying the privacy budget split captured by parameter f. Recall from Section 6 that a fraction f of the budget is used in the first timestamp to determine the adaptive grid structure. A higher f results in a more accurate initial structure, but in the detriment of accuracy when later determining noisy counts for level-2 cells. Conversely, a

small value of f may impact negatively the ability of the structure to capture the initial worker density, but may result in higher accuracy for individual cell counts.

Figure 4.12 shows a decreasing trend for ANW, HOP and WTD as f increases. As expected, a higher f yields lower overhead (ANW) and shorter travel distance (WTD) in *KalmanPID* due to finer-grained grids. The GR size is also smaller, thus leading to a smaller network diameter and HOP value. However, Figure 4.12d shows 36% drop in utility with respect to EU. To achieve the expected utility $EU = 0.9$, f must be set to lower values, such as 0.1.



(a) ANW, Ye.-Linear (b) HOP, Ye.-Linear (c) WTD, Ye.-Linear (d) ASR, Ye.-Linear

Figure 4.12: Varying budget split f.

Chapter 5

Privacy-Preserving Location Popularity (Proposal)

Despite the fact that location entropy has various applications in spatial crowdsourcing [31, 67, 61] and geosocial networks [9, 48], calculating location entropy requires users' locations, which is a privacy threat. Therefore, we propose to publish location entropy using differential privacy. The remainder of this chapter is organized as follows: Section 5.1 defines the problem and discuss its challenges. Section 5.2 presents necessary background. Section 5.3 proposes a baseline algorithm to publish differentially private location entropy. Section 5.4 discusses future work.

5.1 Problem Definition

In this section, we will give the notations and the formal definition of the problem. We then discuss its challenges that need to be addressed in our algorithm design.

5.1.1 Definition

Each location l is represented by a point in two-dimensional space and a unique identifier l ($-180 \leq l_{lat} \leq 180$) and ($-90 \leq l_{lon} \leq 90$)[1]. From now on, l refers to both the location and its unique identifier. For a given location l, let O_l be the set of visits to location l. Thus, $c_l = |O_l|$ gives the total number of visits to l. Also, let U_l be the set of distinct users that visited l, and $O_{l,u}$ be the set of visits that user u has made to the location l. Thus, $c_{l,u} = |O_{l,u}|$ denotes the number of visits of user u to location l.

[1] l_{lat}, l_{lon} are real numbers with ten digits after the decimal point

l, L	a particular location and the set of all locations
$H(l)$	location entropy of location l
$\hat{H}(l)$	noisy location entropy of location l
ΔH_l	sensitivity of location entropy for one location
ΔH	sensitivity of location entropy for multiple locations
O_l	the set of visits to location l
u, U	a particular user and the set of all users
M	maximum number of locations contributed by a user
U_l	the set of distinct users who visits l
$O_{l,u}$	the set of visits that user u has made to the location l
c_l	the total number of visits to l
$c_{l,u}$	the number of visits that user u has made to the location l
C	maximum number of visits a user contributes to a location
$p_{l,u}$	the fraction of total visits to l that belongs to user u
P	the maximum of $p_{l,u}$ across all users and locations

Table 5.1: Notations used in differentially private location entropy.

The probability that a random draw from O_l belongs to $O_{l,u}$ is $p_{l,u} = \frac{|c_{l,u}|}{|c_l|}$, which is the fraction of total visits to l that belongs to user u. The location entropy for l is computed from Shannon entropy [55] as follows:

$$H(l) = H(p_{l,u_1}, p_{l,u_2}, \ldots, p_{l,u_{|U_l|}}) = -\sum_{u \in U_l} p_{l,u} \log p_{l,u} \tag{5.1}$$

In our study, the natural logarithm is used. The aim of our study is to publish location entropy of all locations $L = \{l_1, l_2, ..., l_{|L|}\}$, where each location is visited by a set of users $U = \{u_1, u_2, ..., u_{|U|}\}$ while preserving the privacy of users. Table 5.1 summarizes the notations used in our work.

5.1.2 Challenges

The first challenge is due to a large number of visits contributed by a user per location. If the maximum number of visits of a user to a location is not limited, the change of location entropy when a user is added or removed may be unbounded. This challenge implies that the sensitivity of location entropy should depend on the maximum number of visits of a user to a location. Increasing the sensitivity decreases the utility of differentially-private algorithms. Figure 5.1 shows the maximum number of visits contributed by a user per location in Gowalla dataset that we will use for evaluation.

Although most users contribute one and only one visit, the sensitivity of location entropy is determined by the worst case scenario – the maximum number of visits. This suggests that users tend not to check-in to their homes because if users tend to check-in to their homes (where they often spend most of their time), the peak of the graph would not be at 1.

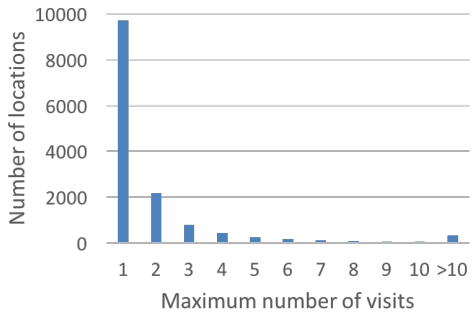


Figure 5.1: The largest number of visits by a user per location (Gowalla, New York).

The second challenge comes from the fact that a user may contribute to many locations. Thus, adding or removing this user may change entropy values of all locations that the user visits, which further increases the sensitivity as a whole. Figure 5.2 shows the number of locations visited by each user. The figure confirms that there are many users who contribute to more than 10 locations.

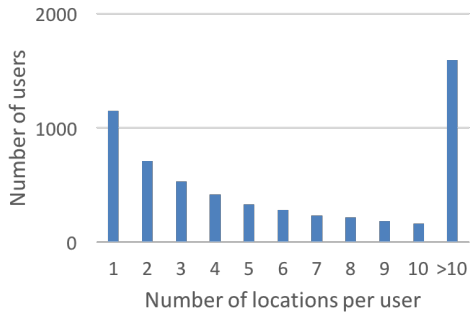


Figure 5.2: A user may contribute to many locations (Gowalla, New York).

The third challenge is due to the sparsity of real-world datasets. For example, Figure 5.3 summarizes the number of users contributing visits to each location in a check-in

dataset. The figure shows that most locations have check-ins from fewer than ten users. The small number of users per location, the smaller location entropy (Equation 5.3), which may leads to excessive noise-to-real-count ratio caused by noise adding mechanism in differential privacy.

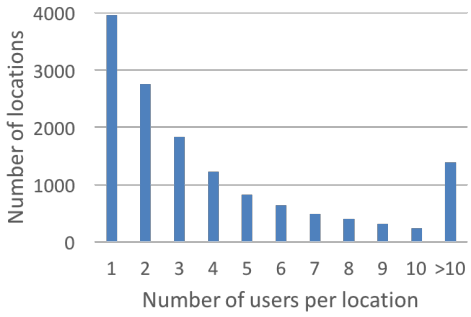


Figure 5.3: Sparsity of location visits (Gowalla, New York).

5.2 Background on Shannon Entropy

In this section, we present some properties of Shannon entropy and differential privacy that will be used throughout the rest of the work.

5.2.1 Shannon Entropy

Shannon [55] introduces entropy as a measure of the uncertainty in a random variable with a probability distribution $U = (p_1, p_2, ..., p_{|U|})$:

$$H(U) = -\sum_i p_i \times \log p_i \tag{5.2}$$

where $\sum_i p_i = 1$. $H(U)$ is maximal if all the outcomes are equally likely:

$$H(U) \leq H(\frac{1}{|U|}, ..., \frac{1}{|U|}) = \log |U| \tag{5.3}$$

We also have:

$$H(p, 1-p) \leq H(q, 1-q) \tag{5.4}$$

where $0 \leq p, q \leq 1$ and $\max\{p, 1-p\} \geq q$.

Entropy Update:

Let U_1 and U_2 be non-overlapping partitions of a database U including users who contribute visits to a location l, and ϕ_1 and ϕ_2 are probabilities that a particular visit belongs to partition U_1 and U_2, respectively. Shannon discovered that using logarithmic function preserves *additivity* property of entropy:

$$H(U) = \phi_1 H(U_1) + \phi_2 H(U_2) + H(\phi_1, \phi_2) \tag{5.5}$$

Then, adding a new person u into U changes its entropy to:

$$H(U^+) = \frac{c_l}{c_l + c_{l,u}} H(U) + H\Big(\frac{c_{l,u}}{c_l + c_{l,u}}, \frac{c_l}{c_l + c_{l,u}}\Big) \tag{5.6}$$

where $U^+ = U \cup u$ and c_l is the total number of visits to l and $c_{l,u}$ is the number of visits to l that is contributed by user u. Equation (5.6) can be derived from Equation (5.5) if we consider U^+ includes two non-overlapping partitions u and U with associated probabilities $\frac{c_{l,u}}{c_l + c_{l,u}}$ and $\frac{c_l}{c_l + c_{l,u}}$. Noting that entropy of a single user is zero, i.e., $H(u) = 0$.

Similarly, removing a person u from U changes its entropy as follows:

$$H(U^-) = \frac{c_l}{c_l - c_{l,u}} \Big(H(U) - H\Big(\frac{c_{l,u}}{c_l}, \frac{c_l - c_{l,u}}{c_l}\Big)\Big) \tag{5.7}$$

where $U^- = U \setminus u$.

5.3 Baseline Solution

In this section, we present a baseline algorithm to publish entropy of a location using Laplace mechanism [16] (Section 5.3.1). In order to appropriately add Laplace noise to location entropy, we first present a bound for the sensitivity of location entropy.

5.3.1 Baseline Algorithm

The baseline algorithm tries to address two challenges, a user may contribute a large number of visits to a location and a user may contribute to many locations. The following lemma shows that the global sensitivity of location entropy is a function of C and M, which linearly increases with plurality of M and monotonically increases with plurality of C. We prove this lemma in Appendix .0.1.

Lemma 1 (Sensitivity of LE depends on C) *Sensitivity of location entropy is* $\Delta H \leq max\left\{\frac{C}{n_0}, \ln 2\right\}M$, *where* $n_0 + C = n_0 \ln n_0$.

Figure 5.4 shows that sensitivity bound monotonically increases when increasing C. This means that the sensitivity can be reduced by having small C. Note that the line graph is non-smooth since C and M are integer values. Similarly, the sensitivity can also be reduced by having small M.

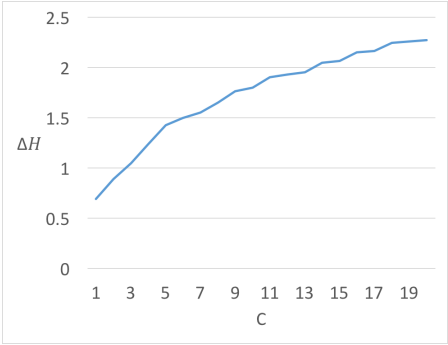


Figure 5.4: Sensitivity bound when varying C.

M can be as large as the number of locations $|L|$ while C can potentially unbounded. Therefore, we propose to limit user activity both across locations and per location. First, to limit the number of locations per user to M, we keep the first M locations' visits of the users who visits more than M locations and throw away the rest of locations' visits. As a result, adding or removing a single user in the data affects at most M locations. Second, to limit the number of visits of every user to a location to C, we set the number of visits of the users who contribute more than C visits to a particular location to C.

At a high-level, the baseline algorithm (Algorithm 5) works as follows. Line 4 limits user activity across locations while Line 8 limits user activity per location. Line 4 has two effects on the published data. First, the number of users $|U_l|$ in some locations may be reduced, which alters the value of location entropy. Second, some locations may be thrown away and their location entropy will not be published. Furthermore, Line 8 also alters the value of location entropy but by trimming the number of locations' visits. The actual entropy of location l (after limiting user activity) is computed in Line 9. The noisy location entropy is published in Line 10, where $Lap(\Delta H/\epsilon_e)$ denotes a random variable drawn independently from Laplace distribution with mean zero and scale parameter $\Delta H/\epsilon_e$.

Algorithm 5 Baseline Algorithm

1: Input: privacy budget ${\epsilon_e}^2$, the set of locations $L = \{l_1, l_2, ..., l_{|L|}\}$, maximum number of visits of a user to a location C, maximum number of locations a user visits M
2: Output: differentially private location entropy of each location $\hat{H}(l)$
3: For each user u in U
4: Limit user activity across locations by keeping the first M locations' visits of the users who visits more than M locations
5: Compute global sensitivity ΔH from Lemma 5.3.1
6: For each location l in L
7: Count #visits each user made to l, $c_{l,u}$
8: Limit user activity per location $\bar{c}_{l,u} = \min(C, c_{l,u})$, from which compute $\bar{p}_{l,u}$
9: Compute $\bar{H}(l) = -\sum_{u \in U_l} \bar{p}_{l,u} \log \bar{p}_{l,u}$
10: Publish noisy location entropy $\hat{H}(l) = \bar{H}(l) + Lap(\frac{\Delta H}{\epsilon_e})$

The performance of Algorithm 5 depends on how we set C and M. There are two conflicting effects regarding the choice of value for C and M. Large C and M require more noise to be injected (Equation .19); however, location entropy can be computed more accurately (Line 9). On the other hand, small C and M would lead to large approximation error in Line 9 but less Laplace noise. Therefore, we need to find C and M that balance perturbation error injected by Laplace noise and the approximation error introduced by limiting individual's activity.

5.3.2 Privacy Guarantee for Baseline Algorithm

The following theorem shows that Algorithm 5 is differentially private.

Theorem 3 (Baseline Algorithm) *Algorithm 5 satisfies $\epsilon-$differential privacy.*

5.4 Future Work

Table 5.5 shows timeline of my proposed work. Particularly, to realize the proposal, our actions are follows.

(i. Reduce the global sensitivity by limiting the number of locations each user visits.

(ii. Evaluate the algorithms in terms of the proposed measurements on real-world geospatial datasets, e.g., Gowalla[3].

(iii. Consider application-dependent metrics, e.g., the number of performed tasks in spatial crowdsourcing.

	2016							2017			
	Jun	July	Aug	Sept	Oct	Nov	Dec	Jan	Feb	Mar	Apr
Developing Algorithms											
Conducting Experiments											
Writing Paper											
Writing Dissertation											

Figure 5.5: Timeline for the proposed work.

[3]https://snap.stanford.edu/data/loc-gowalla.html

.0.1 Sensitivity of Location Entropy

Sensitivity depends on P:

We now prove a sensitivity bound for location entropy ΔH based on the maximum visits of a user to a location. For convenience, let $n_l = |U_l|$. First, we prove a sensitivity bound that depends on P and n_l, where $P = \max_{u \in U_l} p_{l,u}$.

Location entropy of location l, with $c_{l,u}$ visits from user u_i is computed as follows.

$$H(l) = -\sum_{u \in U_l} p_{l,u} \times \ln(p_{l,u}) \tag{.8}$$

where $p_{l,u} = \frac{c_{l,u}}{c_l}$ and $c_l = \sum_{u \in U_l} c_{l,u}$. U_l is the set of users who visits at location l.

U_l and $U_l^+ = U \cup u$ are neighboring user datasets. From Equation 5.6, we have:

$$H(U_l^+) = \frac{c_l}{c_l + c_{l,u}} H(U_l) + H\Big(\frac{c_{l,u}}{c_l + c_{l,u}}, \frac{c_l}{c_l + c_{l,u}}\Big) \tag{.9}$$

Thus,

$$\begin{aligned} H(U_l) - H(U_l^+) &= \frac{c_{l,u}}{c_l + c_{l,u}} H(U_l) - H\Big(\frac{c_{l,u}}{c_l + c_{l,u}}, \frac{c_l}{c_l + c_{l,u}}\Big) \\ &= p_{l,u}^+ H(U_l) - H(p_{l,u}^+, 1 - p_{l,u}^+) \end{aligned}$$

where $p_{l,u}^+ = \frac{c_{l,u}}{c_l + c_{l,u}}$

Sensitivity of entropy is

$$\begin{aligned} \Delta H(l) &= \max_{u \in U_l^+} |H(U_l) - H(U_l^+)| \\ &= \max_{u \in U_l^+} |p_{l,u}^+ H(U_l) - H(p_{l,u}^+, 1 - p_{l,u}^+)| \\ &\leq \max_{u \in U_l^+} \Big\{ p_{l,u}^+ H(U_l), H(p_{l,u}^+, 1 - p_{l,u}^+) \Big\} \end{aligned}$$

A study [56] proved the following inequality of $H(U_l)$ that is relative to the maximum probability P_l defined in Table 5.1.

$$H(U_l) \leq H\Big(\frac{1-P_l}{n_l-1}, ..., \frac{1-P_l}{n_l-1}, P_l\Big) \tag{.10}$$

where $\frac{1-P_l}{n_l-1}$ appears (n_l-1) times.

Since $p^+_{l,u} \leq P_l$, if $P_l \leq 0.5$, then $H(p^+_{l,u}, 1-p^+_{l,u}) \leq H(P_l, 1-P_l)$. If $P_l > 0.5$ and $p^+_{l,u} \neq P_l$, then $p^+_{l,u} \leq 1-P_l \leq 0.5$. So $H(p^+_{l,u}, 1-p^+_{l,u}) \leq H(1-P_l, P_l) = H(P_l, 1-P_l)$.

Thus, sensitivity of location entropy is bounded as follows.

$$\Delta H(l) \leq \max\Big\{P_l \cdot H\Big(\frac{1-P_l}{n_l-1}, ..., \frac{1-P_l}{n_l-1}, P_l\Big), H(P_l, 1-P_l)\Big\} \tag{.11}$$

where $\frac{1-P_l}{n_l-1}$ appears (n_l-1) times.

The Proof of Lemma 1

We now show that above sensitivity bound can be relaxed to be a function of C as follows. From Equation 5.3, we have:

$$\Delta H(l) \leq \max\Big\{\frac{c_{l,u}}{c_l + c_{l,u}} \ln n, H(P_l, 1-P_l)\Big\} \tag{.12}$$

Note that $c_{l,u}$ is the number of visits after being restricted to C. Equality happens when $P_l = 1/n_l$ or all users visit the location with equal probability.

It can be shown that $\frac{c_{l,u}}{c_l+c_{l,u}} \leq \frac{C}{c_l+C} \leq \frac{C}{n+C}$ (equality happens when visit counts of all users are 1, except the one with maximum visits C). Also $H(P_l, 1-P_l) \leq \ln 2$. Thus,

$$\Delta H(l) \leq \max\left\{\frac{C \ln n}{n+C}, \ln 2\right\} \tag{.13}$$

Taking derivative $\Delta H(l)$ with respect to variable n, we have:

$$\begin{aligned}(\Delta H(l))'_n &= C\frac{\frac{n+C}{n} - \ln n}{(n+C)^2} \\ &= C\frac{n+C-n\ln n}{n(n+C)^2}\end{aligned} \tag{.14}$$

$(\Delta H(l))'_n = 0$ at n_0 that satisfies:

$$n_0 + C = n_0 \ln n_0 \tag{.15}$$

Moreover:

$$(n+C-n\ln n)'_n = 1 - \ln n - n\frac{1}{n} = -\ln n \leq 0 \tag{.16}$$

So $(\Delta H(l))'_n$ decreases when n increases. Hence, $(\Delta H(l))'_n > 0$ when $n < n_0$ and $(\Delta H(l))'_n < 0$ when $n > n_0$. So $\Delta H(l)$ increases when $n < n_0$, decreases when $n > n_0$, is maximized at $n = n_0$, and we have:

$$\Delta H(l) \leq \max\left\{\frac{C}{n_0}, \ln 2\right\} \tag{.17}$$

Figure 6 shows the value of function $\frac{C \ln n}{n+C}$ when fixing $C = 10$ and varying n.

To find n_0 from Equation .15, we first limit the value of n_0 in the range $(1, \max(e^2, C)]$. $n_0 \leq \max(e^2, C)$ and $n_0 \geq 1$ since $n_0 - n_0 \ln n_0 + C$ achieves maximum value of $C + 1$ at $n_0 = 1$.

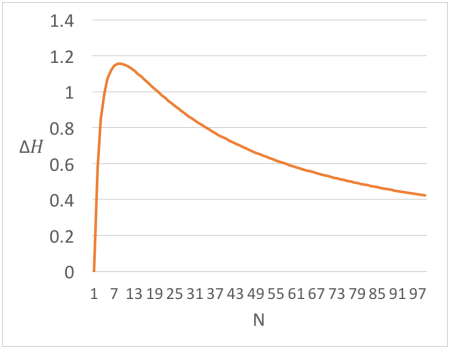


Figure 6: The value of $\frac{C \ln n}{n+C}$ when varying n (C=10). ΔH achieves maximum at $n = n_0$ that satisfies Equation .15

Taking derivative $\Delta H(l)$ with respect to variable C, we have:

$$\begin{aligned}(\Delta H(l))'_C &= \frac{n + C - C}{(n + C)^2} \ln n \\ &= \frac{n}{(n + C)^2} \ln n \end{aligned} \tag{.18}$$

So $\Delta H(l)$ increases when C increases.

The maximum number of locations a user visits is $M \leq |L|$; thus, removing a user would change entropy across all locations by an amount bounded by:

$$\Delta H \leq \max \left\{ \frac{C}{n_0}, \ln 2 \right\} M \tag{.19}$$

The Proof of Lemma 2

In this section, we will bound the sensitivity ΔH based on k and C. Figures 5.4 and 6 suggest that ΔH can be reduced by decreasing C and/or increasing k (i.e., decreasing P), where k is the minimum number of users at a location.

We have:

$$\Delta H(l) \leq \max \Big\{ - P_l^2 \ln P_l - P_l(1 - P_l) \ln(\frac{1 - P_l}{n_l - 1}), H(P_l, 1 - P_l) \Big\} \tag{.20}$$

Set $\Delta H_1 = -P^2 \ln P - P(1-P)\ln(\frac{1-P}{n-1})$ and $\Delta H_2 = H(P, 1-P)$. Figure 7 shows that the sensitivity bound ΔH_1 achieves maximum at a particular value of $P = p_0$ and keeps increasing until $P = p_0$. Next, we show that $p_0 \geq 0.5$.

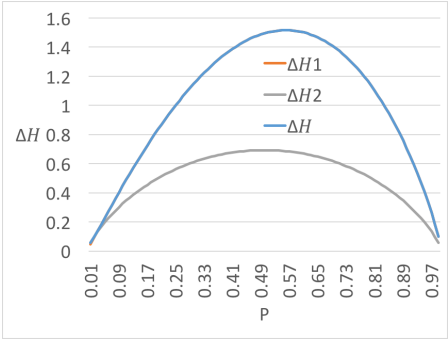


Figure 7: Sensitivity bound when varying P ($n = 100$).

Taking derivative ΔH_1 with respect to variable P, we have:

$$\begin{aligned}(\Delta H_1)'_P &= -2P\ln P + 2P\ln(1-P) - 2P\ln(n-1) - \ln(\frac{1-P}{n-1}) \\ &= -2P\ln P - (1-2P)\ln(\frac{1-P}{n-1}) \end{aligned} \tag{.21}$$

$(\Delta H_1)'_P = \ln 2$ at $P = 0.5$ while $(\Delta H_1)'_P = -0.26 + (1-2P)\ln(n-1) < 0$ at $P = 0.75$; therefore, ΔH_1 obtains maximum at $p_0 > 0.5$ or ΔH_1 monotonically increases when increasing P and $P < 0.5$. Thus,

$$\begin{aligned}\Delta H_1(l) &\leq -P_l^2 \ln P_l - P_l(1-P_l)\ln(\frac{1-P_l}{n_l-1}) \\ &\leq (C_l/n_l)^2 \ln(C_l/n_l) - \frac{C_l(n_l-C_l)}{n_l^2}\ln(\frac{n_l-C_l}{n_l(n_l-1)})\end{aligned}$$

Across all locations (all possible values of n_l and C_l), our hypothesis is that $\Delta H_1 \leq (C/k)^2 \ln(C/k) - \frac{C(k-C)}{k^2}\ln(\frac{k-C}{k(k-1)})$, where $k = \min\{n_l\}$ and $C = \max\{C_l\}$.

Similarly, $(\Delta H_2)'_P = -\ln P + \ln(1-P)$, $(\Delta H_2)'_P$ obtains maximum at $p_0 = 0.5$ and $(\Delta H_2)'_P > 0$ when $P < 0.5$. Therefore, ΔH_2 monotonically increases when increasing P as long as $P < 0.5$. Thus, $H(P_l, 1-P_l) \leq H(P, 1-P)$.

Consequently, we can limit ΔH by limiting C and k (i.e., publish only entropy of cells with at least k visitors and throw away cells with fewer than k users and/or decreasing C).

Reference List

[1] A. Alfarrarjeh, T. Emrich, and C. Shahabi. Scalable spatial crowdsourcing: A study of distributed algorithms. In *Mobile Data Management (MDM), 2015 16th IEEE International Conference on*, volume 1, pages 134–144. IEEE, 2015.

[2] F. Alt, A. S. Shirazi, A. Schmidt, U. Kramer, and Z. Nawaz. Location-based crowdsourcing: extending crowdsourcing to the real world. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, pages 13–22. ACM, 2010.

[3] Z. Chen, R. Fu, Z. Zhao, Z. Liu, L. Xia, L. Chen, P. Cheng, C. C. Cao, Y. Tong, and C. J. Zhang. gmission: A general spatial crowdsourcing platform. *Proceedings of the VLDB Endowment*, 7(13):1629–1632, 2014.

[4] P. Cheng, X. Lian, Z. Chen, R. Fu, L. Chen, J. Han, and J. Zhao. Reliable diversity-based spatial crowdsourcing by moving workers. *Proceedings of the VLDB Endowment*, 8(10):1022–1033, 2015.

[5] S. Choi, G. Ghinita, H.-S. Lim, and E. Bertino. Secure knn query processing in untrusted cloud environments. *Knowledge and Data Engineering, IEEE Transactions on*, 26(11):2818–2831, 2014.

[6] J. Cook. Security flaw in gay dating app grindr reveals precise location of 90August 2014.

[7] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu. Differentially private spatial decompositions. In *ICDE*, pages 20–31, 2012.

[8] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos. AnonySense: privacy-aware people-centric sensing. In *Intl. Conf. on Mobile systems, applications, and services*, pages 211–224, 2008.

[9] J. Cranshaw, E. Toch, J. Hong, A. Kittur, and N. Sadeh. Bridging the gap between physical location and online social networks. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 119–128. ACM, 2010.

[10] H. Dang, T. Nguyen, and H. To. Maximum complex task assignment: Towards tasks correlation in spatial crowdsourcing. In *Proceedings of International Conference on Information Integration and Web-based Applications & Services*, IIWAS '13, pages 77:77–77:81, New York, NY, USA, 2013. ACM.

[11] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific Reports*, 3:1376, 2013.

[12] D. Deng, C. Shahabi, and U. Demiryurek. Maximizing the number of worker's self-selected tasks in spatial crowdsourcing. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 324–333. ACM, 2013.

[13] D. Deng, C. Shahabi, and L. Zhu. Task matching and scheduling for multiple workers in spatial crowdsourcing, 2015.

[14] R. Dewri, I. Ray, and D. Whitley. Query m-invariance: Preventing query disclosures in continuous location-based services. In *Mobile Data Management (MDM), 2010 Eleventh International Conference on*, pages 95–104. IEEE, 2010.

[15] C. Dwork. Differential privacy. In *Automata, languages and programming*, pages 1–12. Springer, 2006.

[16] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pages 265–284. Springer, 2006.

[17] L. Fan and L. Xiong. An adaptive approach to real-time aggregate monitoring with differential privacy. *Knowledge and Data Engineering, IEEE Transactions on*, 26(9):2094–2106, Sept 2014.

[18] B. Gedik and L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *Mobile Computing, IEEE Transactions on*, 7(1):1–18, 2008.

[19] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location based services: anonymizers are not necessary. In *SIGMOD*, pages 121–132, 2008.

[20] G. Ghinita, P. Kalnis, and S. Skiadopoulos. Prive: anonymous location-based queries in distributed mobile systems. In *Proceedings of the 16th international conference on World Wide Web*, pages 371–380. ACM, 2007.

[21] Y. Gong, L. Wei, Y. Guo, C. Zhang, and Y. Fang. Optimal task recommendation for mobile crowdsourcing with privacy control. *Internet of Things Journal*, 2015.

[22] Y. Gong, C. Zhang, Y. Fang, and J. Sun. Protecting location privacy for task allocation in ad hoc mobile cloud computing. *IEEE Transactions on Emerging Topics in Computing*, 2015.

[23] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, 2008.

[24] M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *USENIX MobiSys*, 2003.

[25] B. J. Hecht and D. Gergle. On the localness of user-generated content. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 229–232. ACM, 2010.

[26] J. Hu, L. Huang, L. Li, M. Qi, and W. Yang. Protecting location privacy in spatial crowdsourcing. In *Web Technologies and Applications*, pages 113–124. Springer, 2015.

[27] L. Hu and C. Shahabi. Privacy assurance in mobile sensing networks: go beyond trusted servers. In *Pervasive Computing and Communications*, pages 613–619, 2010.

[28] K. L. Huang, S. S. Kanhere, and W. Hu. Towards privacy-sensitive participatory sensing. In *Pervasive Computing and Communications*, pages 1–6, 2009.

[29] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *Knowledge and Data Engineering, IEEE Transactions on*, 19(12):1719–1733, 2007.

[30] L. Kazemi and C. Shahabi. Towards preserving privacy in participatory sensing. In *Pervasive Computing and Communications*, pages 328–331. IEEE, 2011.

[31] L. Kazemi and C. Shahabi. GeoCrowd: enabling query answering with spatial crowdsourcing. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. ACM, 2012.

[32] L. Kazemi, C. Shahabi, and L. Chen. Geotrucrowd: trustworthy query answering with spatial crowdsourcing. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 314–323. ACM, 2013.

[33] A. Khoshgozaran and C. Shahabi. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In *Advances in Spatial and Temporal Databases*, pages 239–257. Springer, 2007.

[34] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. In *Pervasive Services, 2005. ICPS'05. Proceedings. International Conference on*, pages 88–97. IEEE, 2005.

[35] C. E. Kim and T. A. Anderson. Digital disks and a digital compactness measure. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 117–124. ACM, 1984.

[36] S. H. Kim, Y. Lu, G. Constantinou, C. Shahabi, G. Wang, and R. Zimmermann. Mediaq: mobile multimedia management system. In *Proceedings of the 5th ACM Multimedia Systems Conference*, pages 224–235. ACM, 2014.

[37] J. Krumm. Inference attacks on location tracks. In *Pervasive Computing*, pages 127–143. Springer, 2007.

[38] Q. Li and G. Cao. Providing privacy-aware incentives in mobile sensing systems. *IEEE Transactions on Mobile Computing*, 2015.

[39] W. Li, M. F. Goodchild, and R. Church. An efficient measure of compactness for two-dimensional shapes and its application in regionalization problems. *International Journal of Geographical Information Science*, (ahead-of-print):1–24, 2013.

[40] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[41] F. McSherry and I. Mironov. Differentially private recommender systems: building privacy into the net. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 627–636. ACM, 2009.

[42] mobiThinking. Global mobile statistics 2014 part a: Mobile subscribers; handset market share; mobile operators. May 16 2014.

[43] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new Casper: query processing for location services without compromising privacy. In *Proceedings of the 32nd international conference on Very large data bases*, pages 763–774. VLDB Endowment, 2006.

[44] M. Musthag and D. Ganesan. Labor dynamics in a mobile micro-task market. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 641–650. ACM, 2013.

[45] J. C. Navas and T. Imielinski. Geocast: geographic addressing and routing. In *Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking*, pages 66–76. ACM, 1997.

[46] R. Noackg. Could using gay dating app grindr get you arrested in egypt? September 2014.

[47] B. Palanisamy and L. Liu. Mobimix: Protecting location privacy with mix-zones over road networks. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pages 494–505. IEEE, 2011.

[48] H. Pham, C. Shahabi, and Y. Liu. Inferring social strength from spatiotemporal data. *ACM Trans. Database Syst.*, 41(1):7:1–7:47, Mar. 2016.

[49] L. Pournajaf, D. A. Garcia-Ulloa, L. Xiong, and V. Sunderam. Participant privacy in mobile crowd sensing task management: A survey of methods and challenges. *SIGMOD Record*, 44(4):23, 2015.

[50] L. Pournajaf, L. Xiong, and V. Sunderam. Dynamic data driven crowd sensing task assignment. *Procedia Computer Science*, 29:1314–1323, 2014.

[51] L. Pournajaf, L. Xiong, V. Sunderam, and S. Goryczka. Spatial task assignment for crowd sensing with cloaked locations. In *Mobile Data Management (MDM), 2014 IEEE 15th International Conference on*, volume 1, pages 73–82. IEEE, 2014.

[52] W. Qardaji, W. Yang, and N. Li. Differentially private grids for geospatial data. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 2013.

[53] W. Qardaji, W. Yang, and N. Li. Understanding hierarchical methods for differentially private histograms. *Proceedings of the VLDB Endowment*, 6(14):1954–1965, 2013.

[54] M. Sauter. Beyond 3g: Bringing networks, terminals and the web together: Lte, wimax, ims, 4g devices and the mobile web 2.0. 2011.

[55] C. E. Shannon and W. Weaver. A mathematical theory of communication, 1948.

[56] I. J. Taneja. On generalized information measures and their applications. *Advances in Electronics and Electron Physics*, 76:327–413, 1989.

[57] R. Teodoro, P. Ozturk, M. Naaman, W. Mason, and J. Lindqvist. The motivations and experiences of the on-demand mobile workforce. In *The 17th CSCW*, pages 236–247. ACM, 2014.

[58] J. Thebault-Spieker, L. G. Terveen, and B. Hecht. Avoiding the south side and the suburbs: The geography of mobile crowdsourcing markets. In *The 18th CSCW*, pages 265–275. ACM, 2015.

[59] H. To, M. Asghari, D. Deng, and C. Shahabi. SCAWG: A toolbox for generating synthetic workload for spatial crowdsourcing. In *CROWDBENCH 2016*. IEEE.

[60] H. To, L. Fan, and C. Shahabi. Differentially private h-tree. 2015.

[61] H. To, L. Fan, L. Tran, and C. Shahabi. Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints. *In proceedings of the IEEE International Conference on Pervasive Computing and Communications*, 2016.

[62] H. To, R. Geraldes, C. Shahabi, S. H. Kim, and H. Prendinger. An empirical study of workers' behavior in spatial crowdsourcing. 2016.

[63] H. To, G. Ghinita, L. Fan, and C. Shahabi. Differentially private location protection for worker datasets in spatial crowdsourcing. In *IEEE Transactions on Mobile Computing (IEEE TMC 2016)*. IEEE.

[64] H. To, G. Ghinita, and C. Shahabi. A framework for protecting worker location privacy in spatial crowdsourcing. *Proceedings of the VLDB Endowment*, 7(10):919–930, 2014.

[65] H. To, G. Ghinita, and C. Shahabi. PrivGeoCrowd: A toolbox for studying private spatial crowdsourcing. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*, pages 1404–1407, April 2015.

[66] H. To, S. H. Kim, and C. Shahabi. Effectively crowdsourcing the acquisition and analysis of visual data for disaster response. In *Big Data (Big Data), 2015 IEEE International Conference on*, pages 697–706. IEEE, 2015.

[67] H. To, C. Shahabi, and L. Kazemi. A server-assigned spatial crowdsourcing framework. *ACM Transactions on Spatial Algorithms and Systems*, 1(1):2, 2015.

[68] U. ul Hassan and E. Curry. A multi-armed bandit approach to online spatial task assignment. In *11th IEEE International Conference on Ubiquitous Intelligence and Computing UIC 2014)*, 2014.

[69] U. ul Hassan and E. Curry. Efficient task assignment for spatial crowdsourcing: A combinatorial fractional optimization approach with semi-bandit learning. *Expert Systems with Applications*, 2016.

[70] E. Welzl. *Smallest enclosing disks (balls and ellipsoids)*. Springer, 1991.

[71] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. *Knowledge and Data Engineering, IEEE Transactions on*, 23(8):1200–1214, 2011.

[72] Y. Xiao, L. Xiong, and C. Yuan. Differentially private data release through multidimensional partitioning. In *Secure Data Management*, pages 150–168. Springer, 2010.

[73] M. Xue, P. Kalnis, and H. K. Pung. Location diversity: Enhanced privacy protection in location based services. In *Location and Context Awareness*, pages 70–87. Springer, 2009.

[74] B. Yao, F. Li, and X. Xiao. Secure nearest neighbor revisited. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 733–744. IEEE, 2013.

[75] M. L. Yiu, G. Ghinita, C. S. Jensen, and P. Kalnis. Outsourcing search services on private spatial data. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*, pages 1140–1143. IEEE, 2009.

[76] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu. Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 366–375. IEEE, 2008.

[77] H. Yu, C. Miao, Z. Shen, and C. Leung. Quality and budget aware task allocation for spatial crowdsourcing. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1689–1690. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[78] L. Zhang, X. Lu, P. Xiong, and T. Zhu. A differentially private method for reward-based spatial crowdsourcing. In *Applications and Techniques in Information Security*, pages 153–164. Springer, 2015.