A Framework for Protecting Worker Location Privacy in Spatial Crowdsourcing

Hien To
Computer Science Dept.
Univ. of Southern California
hto@usc.edu

Gabriel Ghinita
Dept. of Computer Science
UMass Boston
Gabriel.Ghinita@umb.edu

Cyrus Shahabi
Computer Science Dept.
Univ. of Southern California
shahabi@usc.edu

ABSTRACT

Spatial Crowdsourcing (SC) is a 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. In this paper, 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. Extensive experimental results on real-world datasets show that the proposed technique protects workers' location privacy without incurring significant performance metrics penalties.

1. INTRODUCTION

Recent years have witnessed a significant growth in the number of mobile smart phone users, as well as fast development in phone hardware performance, software functionality and communication features. Today's mobile phones are powerful devices that can act as multi-modal sensors collecting and sharing various types of data, e.g., picture, video, location, movement speed, direction and acceleration. In this context, *Spatial Crowdsourcing (SC)* [14] is emerging as a novel and transformative platform that engages individuals, groups and communities in the act of collecting, analyzing, and disseminating environmental, social and other information for which spatio-temporal features are relevant. With SC, *task requesters* outsource their 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. The nature of tasks may vary from environmental sensing to capturing images at social or entertainment events. Typically, requesters and workers register with a centralized *spatial crowdsourcing server (SC-server)* that acts as a broker between parties, and often also plays a role in how tasks are assigned to workers (i.e., scheduling according to some performance criteria). SC has numerous applications in domains such as environmental sensing, journalism, crisis response and urban planning.

Consider an emergency response scenario where the Red Cross (i.e., requester) is interested in collecting pictures and videos of disaster areas from various locations in a country (e.g., typhoon Haiyan in the Philippines in 2013). The requester issues a query to an SC-server, and the request is forwarded to workers situated in proximity to the zones of interest. The workers record photos and videos using their mobile phones, and send the results back to the requester. *Participatory sensing* is another domain where SC is very suitable. Mobile users can leverage their sensor-equipped mobile devices to collect environmental or traffic data.

SC is feasible only if workers and tasks are matched effectively, i.e., tasks are completed in a timely fashion, and workers do not need to travel across very long distances. To that extent, matching at the SC-server must take into account the locations of workers. However, the SC-server may not be trusted, and disclosing individual locations has serious privacy implications [9, 20, 7, 3]. Knowing worker locations, an adversary can stage a broad spectrum of attacks such as physical surveillance and stalking, identity theft, and breach of sensitive information (e.g., an individual's health status, alternative lifestyles, political and religious views). Thus, ensuring location privacy is an essential aspect of SC, because mobile users will not accept to engage in spatial tasks if their privacy is violated.

Several solutions [9, 20, 7] have been proposed to protect location-based queries, i.e., given an individual's location, find points of interest in the proximity without disclosing the actual coordinates. However, in SC, a worker's location is no longer part of the query, but rather the result of a spatial query around the task. In addition, while some work considers queries on private locations in the context of outsourced databases [28, 27], 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.

Articles from this volume were invited to present their results at the 40th International Conference on Very Large Data Bases, September 1st - 5th 2014, Hangzhou, China.
Proceedings of the VLDB Endowment, Vol. 7, No. 10
Copyright 2014 VLDB Endowment 2150-8097/14/06.

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)* [5]. 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 already has access to the worker locations (e.g., through cell tower triangulation). The CSP signs a contract with its subscribers, which stipulates the terms and conditions of location disclosure. Thus, the CSP can release worker locations to third party SC-servers in noisy form, according to DP. However, using DP introduces two difficult challenges, as discussed next.

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, first introduced in [3]. A PSD is a sanitized spatial index, where each index node contains a noisy count of the workers rooted at that node. Specifically, we devise a mechanism to create a *Worker* PSD by extending the *Adaptive Grid (AG)* technique [23]. 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, by the nature of the DP protection model, fake entries may need to be created in the PSD. Thus, the SC-server cannot directly contact workers, not even if pseudonyms are used, as merely establishing a network 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 [22] as means to deliver task requests to workers. Once a PSD partition is identified by the analytical model outlined above, the task request is geocast to all the workers within the partition. Geocast introduces overhead considerations that need to be carefully considered in the framework design.

Our specific contributions are:

(i). We identify the specific challenges of location privacy in the context of SC, and we propose a framework that achieves differentially-private protection guarantees. To the best of our knowledge, this is the first work to study location privacy for SC.

(ii). We propose an analytical model that measures the probability of task completion with uncertain worker locations, and we devise a search strategy that finds appropriate PSD partitions to ensure high success rate of task assignment.

(iii). We introduce a geocast mechanism for task request dissemination that is necessary to overcome the restrictions imposed by DP, and we factor the geocast system overhead in the PSD partition search strategy.

(iv). We conduct an extensive set of experiments on real-world datasets which shows that the proposed framework is able to protect workers' location privacy without significantly affecting the effectiveness and efficiency of the SC system.

The remainder of this paper is organized as follows: Section 2 presents necessary background. Section 3 introduces the proposed privacy framework, whereas Sections 4 and 5 detail the proposed solution. Experimental results are presented in Section 6, followed by a survey of related work in Section 7, and conclusions in Section 8.

2. BACKGROUND

2.1 Spatial Crowdsourcing

Spatial Crowdsourcing SC [14] is a type of online crowdsourcing where performing a task requires the worker to travel to the location of the task (termed *spatial task*). According to the taxonomy in [14], there are two categories of SC, based on how workers are matched to tasks. In *Worker Selected Tasks (WST)* mode, the SC-server publishes the spatial tasks online, and workers can autonomously 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 nearby workers.

WST is the simpler protocol, and it does not require workers to share their locations with the SC-server. However, the assignment is often 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.

In our work, we consider the SAT mode, but we also provide location privacy protection for the workers. Instead of directly disclosing their coordinates to the SC-server, worker locations are first pooled together by a CSP and sanitized according to differential privacy. This introduces significant challenges, as the SC-server has to employ far more complex task assignment strategies that must take into account the uncertain nature of the received location data.

2.2 Differential Privacy

Differential Privacy (DP) [5] 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 access to background information. DP ensures that an adversary is not able to learn from the sanitized data whether a particular individual is present or not in the original data, regardless of the adversary's prior knowledge.

DP allows interaction with a database only by means of aggregate (e.g., count, sum) queries. Random noise is added to each query result to preserve privacy, such that an adversary that attempts to attack the privacy of some individual worker w will not be able to distinguish from the set of query results (called a *transcript*) whether a record representing w is present or not in the database.

DEFINITION 1 (ϵ-INDISTINGUISHABILITY). *Consider that a database produces transcript U on the set of queries $QS = \{Q_1, Q_2, \ldots, Q_q\}$, and let $\epsilon > 0$ be an arbitrarily-small real constant. Then, transcript U satisfies ϵ-indistinguishability if for every pair of sibling datasets D_1, D_2 such that $|D_1| = |D_2|$ and D_1, D_2 differ in only one record, it holds that*

$$\ln \frac{Pr[QS^{D_1} = U]}{Pr[QS^{D_2} = U]} \leq \epsilon$$

In other words, an attacker cannot learn whether the transcript was obtained by answering the query set QS 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 QS, formally defines as:

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

$$\sigma(QS) = \max_{D_1, D_2} \sum_{i=1}^{q} |QS^{D_1} - QS^{D_2}|$$

A sufficient condition to achieve differential privacy with parameter ϵ is to add to each query result randomly distributed Laplace noise with mean $\lambda = \sigma(QS)/\epsilon$ [6].

Typically, the interaction with a dataset consists of a series of analyses (i.e., transcripts) A_i, each required to satisfy ϵ_i-differential privacy. Then, the privacy level of the resulting analysis can be computed as follows:

Theorem 1 (Sequential Composition [19]). *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 [19]). *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.*

2.3 Private Spatial Decompositions (PSD)

The work in [3] introduced the concept of *Private Spatial Decompositions (PSD)* to release spatial datasets in a DP-compliant manner. A PSD is a spatial index transformed according to DP, where each index node is obtained by releasing a noisy count of the data points enclosed by that node's extent. Various index types such as grids, quad-trees or k-d trees [24] can be used as a basis for PSD.

Accuracy of PSD is heavily influenced by the type of PSD structure and its parameters (e.g., height, fan-out). With space-based partitioning PSD, the split position for a node does not depend on worker locations. This category includes flat structures such as grids, or hierarchical ones such as BSP-trees (Binary Space Partitioning) and quad-trees [24]. The privacy budget ϵ needs to be consumed only when counting the workers in each index node. Typically, all nodes at same index level have non-overlapping extents, which yields a constant and low sensitivity of 2 per level (i.e., changing a single location in the data may affect at most two partitions in a level). The budget ϵ is best distributed across levels according to the *geometric allocation* [3], where leaf nodes receive more budget than higher levels. The sequential composition theorem applies across nodes on the same root-to-leaf path, whereas parallel composition applies to disjoint paths in the hierarchy. Space-based PSD are simple to construct, but can become unbalanced.

Object-based structures such as k-d trees and R-trees [3] perform splits of nodes based on the placement of data points. To ensure privacy, split decisions must also be done according to DP, and significant budget may be used in the process. Typically, the exponential mechanism [3] is used to assign a merit score to each candidate split point according to some cost function (e.g., distance from median in case of k-d trees), and one value is randomly picked based on its noisy score. The budget must be split between protecting node counts and building the index structure. Object-based PSD are more balanced in theory, but they are not very robust, in the sense that accuracy can decrease abruptly with only slight changes of the PSD parameters, or for certain input dataset distributions.

The recent work in [23] compares tree-based methods with multi-level grids, and shows that two-level grids tend to perform better than recursive partitioning counterparts. The paper also proposes an *Adaptive Grid (AG)* approach, where the granularity of the second-level grid is chosen based on the noisy counts obtained in the first-level (sequential composition is applied). AG is a hybrid which inherits the simplicity and robustness of space-based PSD, but still uses a small amount of data-dependent information in choosing the granularity for the second level. In our work, we adapt the AG method to address SC-specific requirements.

3. PRIVACY FRAMEWORK

Section 3.1 presents the system model and the workflow for privacy-preserving SC. Section 3.2 outlines the privacy model and assumptions. Section 3.3 discusses design challenges and associated performance metrics.

3.1 System Model

We consider the problem of privacy-preserving SC *task assignment* in the SAT mode. Figure 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 in relative proximity to t. Due to the uncertain nature of the PSD, this is a challenging process which will be detailed later in Section 5. Next, the SC-server initiates a *geocast* communication [22] 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 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 confirming w's availability (alternatively, the consent can be directly sent to the requester). If w is not willing to participate in the task, then no consent is sent, and no information about the worker is disclosed.

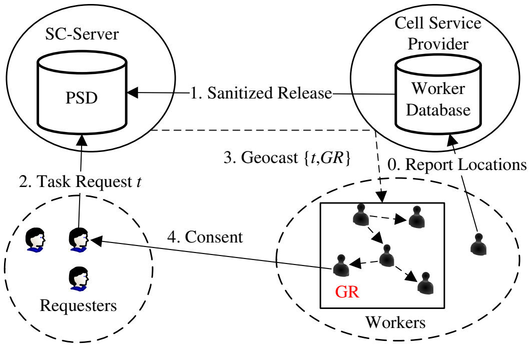


Figure 1: Privacy framework for spatial crowdsourcing

3.2 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.

Focusing on the SC assignment step is important, given the fact that SC workers have to travel to the task location. Mere completion of a task discloses the fact that some worker must have been at that location, and this sort of disclosure 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.

Furthermore, 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 would be 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.

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 the case of public-interest SC applications.

3.3 Design Goals and Performance Metrics

Protecting worker locations significantly complicates 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, a worker may be notified of the task even though she is at a long distance away from the task location, whereas a nearer worker does not receive the request. Finally, in the non-private SAT case, only one selected worker, whose location and identity are known, is notified of the task request. With location protection, many redundant messages may need to be sent, increasing system overhead.

Therefore, we focus on the following performance metrics:

- **Assignment Success Rate (ASR).** Due to PSD data uncertainty, the SC-server may fail to 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[1] to the total number of task requests. The challenge is to keep *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 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 algorithms, 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 *computational overhead* of the matching algorithm, which depends on how many workers need to be notified of a task request.

4. BUILDING THE WORKER PSD

The first step 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*. In this section, we modify the state-of-the-art *Adaptive Grid (AG)* method proposed in [23] to address the specific requirements of the SC framework. Table 1 summarizes the notations used in our paper.

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

[1] *ASR* does not capture worker reliability, tasks may still fail to complete after being accepted. Our focus is on assignment success, reliability is outside our scope.

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 1: Summary of Notations

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

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

where N is the total number of locations and $k_1 = 10$ [23].

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:

$$m_2 = \left\lceil \sqrt{\frac{N' \times \epsilon_2}{k_2}} \right\rceil \qquad (1)$$

where $\epsilon_2 = \epsilon - \epsilon_1$ is the remaining budget, and the constant is set empirically to $k_2 = 5$. Parameter α determines how privacy budget is divided between the two levels.

Figure 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 $\lambda_1 = 2/\varepsilon_1$ to the actual counts of these cells. Second, based on the noisy counts, level-1 cells are further split into level-2 cells. According to Eq. (1), cell D, which has noisy count 200 is partitioned according to a 3x3 grid, while the granularity for other cells is 2x2. Thereafter, AG adds to each level-2 cell (c_i, $i = 1..21$) 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. According to Theorem 2, the sanitized release of AG provides ε-DP.

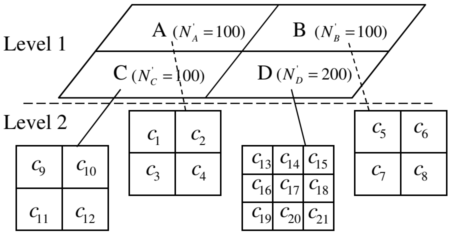


Figure 2: A snapshot of adaptive grid ($\varepsilon = 0.5$, $\alpha = 0.5$)

Although AG was shown to yield good results for general-purpose spatial queries [23], it is not directly applicable to SC, due to its rigidity in choosing its parameters. Specifically, the granularity m_2 of the level-2 grid is too coarse, leading to large geocast areas and high communication overhead, as we show next. According to Eq. (1), the expected number of workers (i.e., noisy count) in a level-2 cell is:

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

Table 2a presents different values of m_2 and $\bar{n}$ when varying total budget ϵ with $\alpha = 0.5$. 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.

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

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.

We increase 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 a $Lap(1/\varepsilon_2)$ distribution, the probability that the noisy count is larger than zero is:

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

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. (1), the corresponding granularity is $m_2 = \left\lceil \sqrt{N'\varepsilon_2/\sqrt{2}} \right\rceil$.

In summary, we modify 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 2b shows that this new setting significantly reduces $\bar{n}$, and as a result ANW. Next, we present a search strategy which groups cells together such that the achieved ASR is above a given threshold.

5. TASK ASSIGNMENT

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 of the SC-server 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.

5.1 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 [21, 14, 1]. The study in [21] 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* [14]. A related study [10] 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:

$$MCD(w_i) = \sum_{j=1}^{n} \frac{d(L_{w_i}, L_{c_j})}{n} \qquad (2)$$

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. (2), we can find the *maximum travel distance (MTD)* that a high percentage of workers are willing to travel. For example, MTD of super-agents in crowdsourcing markets studied in [21] 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 × 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. We assume that all workers are identical and independent of each other in deciding to perform tasks. The study in [21] researches 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 [14], where the workers voluntarily perform tasks, without receiving incentives.

A worker is more likely to accept nearby tasks. To that extent, we model acceptance rate as a decreasing function F 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 the value of s, 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, $p^a = 0$.

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. The utility depends on the AR and on the worker count $\bar{w}$ estimated to be enclosed within GR. A SC-server will typically establish an *expected utility* threshold EU which is the targeted success rate for a task. 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: $P(X = \mathit{True}) = p^a$ and $P(X = \mathit{False}) = 1 - p^a$. Assuming w independent workers, $X \sim \mathit{Binomial}(w, p^a)$. We define the *utility* of a geocast region covering w workers as:

$$U = 1 - (1 - p^a)^w \qquad (3)$$

U measures the probability that at least one worker accepts the task. The utility definition can be extended for the case of redundant task assignment, where multiple workers are required to complete a task. In such a case, $U = 1 - \sum_{i=1}^{k} \binom{w}{i} (p^a)^i (1 - p^a)^{w-i}$, where k is the number of workers required to perform the task. Although redundant task assignment is required in some cases [15], in this work we focus on single-worker task assignment.

5.2 Geocast Region Construction

Given task t, the geocast region 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 keeps expanding 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. (3)
7: If $c_i = Nil$, return $GR$ /*geocast region 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. (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. The updated utility is the probability that a worker in either the current geocast region, or the newly added cell, or in both, performs the task:

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

Line 11 computes the new neighboring cells that are not in GR, and they are not situated farther than MTD. These cells are added to Q according to their respective 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, in Section 5.3 we investigate a finer-grained solution search space by allowing partial cell inclusion; second, in Section 5.4 we consider the effect that the GR shape has on hop-by-hop task request dissemination.

5.3 Partial Cell Selection

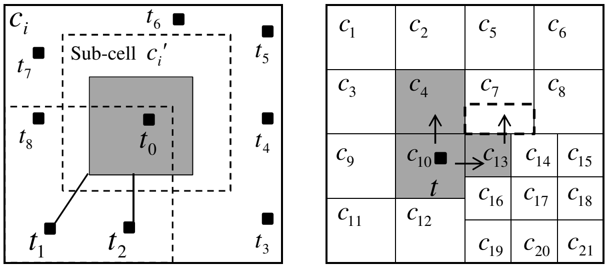
The adaptation of AG proposed in Section 4 significantly reduces the granularity of level-2 cells, but the number of workers can still be large, and the resulting ANW can lead to high overhead. Such workers may be unnecessarily included in the GR, even if the required EU could be achieved with far fewer workers. We propose an optimization that allows partial inclusion in the GR of a level-2 cell.

Before finalizing 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 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 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.

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: $U_{required} = \frac{U - U_{curr}}{1 - U_{curr}}$
6: Worker count needed to achieve $U_{required}$, $w_{required} = \log^{1-p^a_{sub}}_{1-U_{required}}$
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
```

Figure 3a depicts the 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 square. Subsequently, depending on the relative position of the task to the shaded circle (i.e., eight possibilities t_1-t_8), we determine the sub-cell's center. For example, the closest sub-cell's center of t_1 is the left bottom corner of the shaded square.



(a) Case 1: splitting c_i

(b) Case 2: splitting cell 7

Figure 3: Examples of partial cell selection

Figure 3b presents the case when the GR comprises of multiple cells {4,7,10,13}. This example is a flat version of the AG in Figure 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) from which the splitting cell (i.e., 7) is expanded. 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.

5.4 Communication Cost

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

- *Infrastructure 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.
- *Infrastructureless 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 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 regions have equal areas.

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 [18]. One widely accepted measure proposed in [17] 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 [26] 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:

$$Hop\ count = \frac{Farthest\ distance\ between\ two\ workers}{2 \times Communication\ range} \quad (4)$$

The worker network needs to be connected for the ad-hoc based geocast to succeed. In other words, 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.

6. PERFORMANCE EVALUATION

We evaluate experimentally the performance of the proposed framework for worker location protection in SC. We present the experimental methodology in Section 6.1, followed by results and discussions in Section 6.2.

6.1 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. (2), and 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, and a user review is equivalent to accepting a SC task. The MTD for Yelp is 13.5km.

Name	**#Tasks**	**#Workers**	**MTD (km)**
Gowalla	151,075	6,160	3.6
Yelp	15,583	70,817	13.5

Table 3: Dataset Characteristics

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 values $\epsilon \in \{0.1, \mathbf{0.4}, 0.7, 1\}$, 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 $MAR \in \{\mathbf{0.1}, 0.4, 0.7, 0.9\}$. Default values are shown in boldface. For Zipf acceptance rate decrease function, skew parameter s is set to 1. Wireless communication range is 100 meters.

We randomly generated $1,000$ tasks and measured the performance of the proposed solution with respect to ASR, ANW, and WTD. For WTD, we consider two scenarios: in WTD-NN, the SC-server collects all consents and chooses the closest worker to the task site, whereas in WTD-FC the first consenting worker is assigned to the task. We also measure the average hop count HOP required for geocast, according to Eq. (4). To compute ASR, we simulate a binomial model as discussed in Section 5.1, and each worker flips a biased coin and decides whether to accept a received task request or not, based on personalized threshold p^a (recall that p^a takes into account distance to task). A task is considered accepted if at least one worker agrees to perform it. 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 to better understand the underpinnings of the proposed solution. All measured results are averaged over ten random seeds.

6.2 Experimental Results

6.2.1 Evaluation of GR Construction Heuristics

We evaluate the performance of the greedy algorithm for GR construction from Section 5.2 and its variations. GDY refers to the algorithm running using the original AG PSD from [23], whereas G-GR uses our customized granularity 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 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). This is due to the fact that the optimization is applied only to the last visited cell. Compared to GDY, $G\text{-}GP$ reduces ANW by up to a factor of 5, and the improvement is more significant when privacy budget is low. 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, $G\text{-}GP$ also yields reduced WTD and HOP by up to a factor of 8 and 7, respectively.

On the other hand, $G\text{-}PA$ obtains lower ASR than the expected utility of 90%, particularly for small ϵ. This can be explained based on the fact that applying partial cell selection tends to reduce aggressively the number of workers included in the GR, which may result in under-provisioning (i.e., an insufficient number of workers receive task requests). All other methods achieve close to the target EU of 90%, but most often this is a result of over-provisioning, which in turn increases ANW.

Figure 5 captures in more detail the effect of $G\text{-}GP$ and grid granularity on ASR, as well as the under/over-provisioning tendencies. With coarser-grained grids (i.e., large k_2) over-provisioning occurs, whereas finer-grained grids suffer from excessive noise-to-real-count ratio, resulting in under-provisioning. Note that our choice of $k_2 = \sqrt{2} \sim 1.41$ achieves a good trade-off: it results in about 90% utility and also reduces ANW, WTD and HOP.

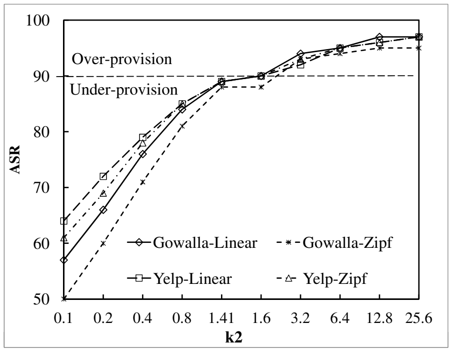


Figure 5: Average ASR over $\epsilon \in \{.1, .4, .7, 1\}$, varying k_2

6.2.2 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 6 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 ($G\text{-}GP\text{-}Hybrid$) 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.

6.2.3 Overhead of Achieving Privacy

We compare the proposed solution with the non-private algorithm for task assignment, presented in Section 6.1. Figure 7 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.

	ANW	HOP	$WTDNN$	$WTDFC$
Gow.-Linear	161%	54%	25%	18%
Gow.-Zipf	103%	30%	22%	23%
Yelp-Linear	202%	92%	19%	20%
Yelp-Zipf	132%	41%	17%	25%

Table 4: Average relative increase in percentage of different performance metrics compared to non-private case

We also observe that privacy does not significantly increase WTD, proving that the greedy GR construction algorithm does a good job in selecting nearby workers for a task. Table 4 summarizes the variation of considered metrics when adding privacy. Note that, the travel distance, which is perhaps the most important factor in SC, is not considerably impacted by privacy. We also observed that the overhead incurred is generally higher for the sparser Yelp data, which is not surprising, as it is a well-known fact that differentially private algorithms perform better on dense datasets.

Table 4 also shows the effect of different acceptance rate functions. Zipf incurs lower overhead compared to Linear. The reason is that with Zipf distribution, the acceptance rate of the workers drops faster for the same distance to the task compared with the linear case. The smaller acceptance rate leads to larger ANW in both private and non-private cases; however, ANW increases at a faster rate in the non-private case.

6.2.4 The Effect of Varying MAR and EU

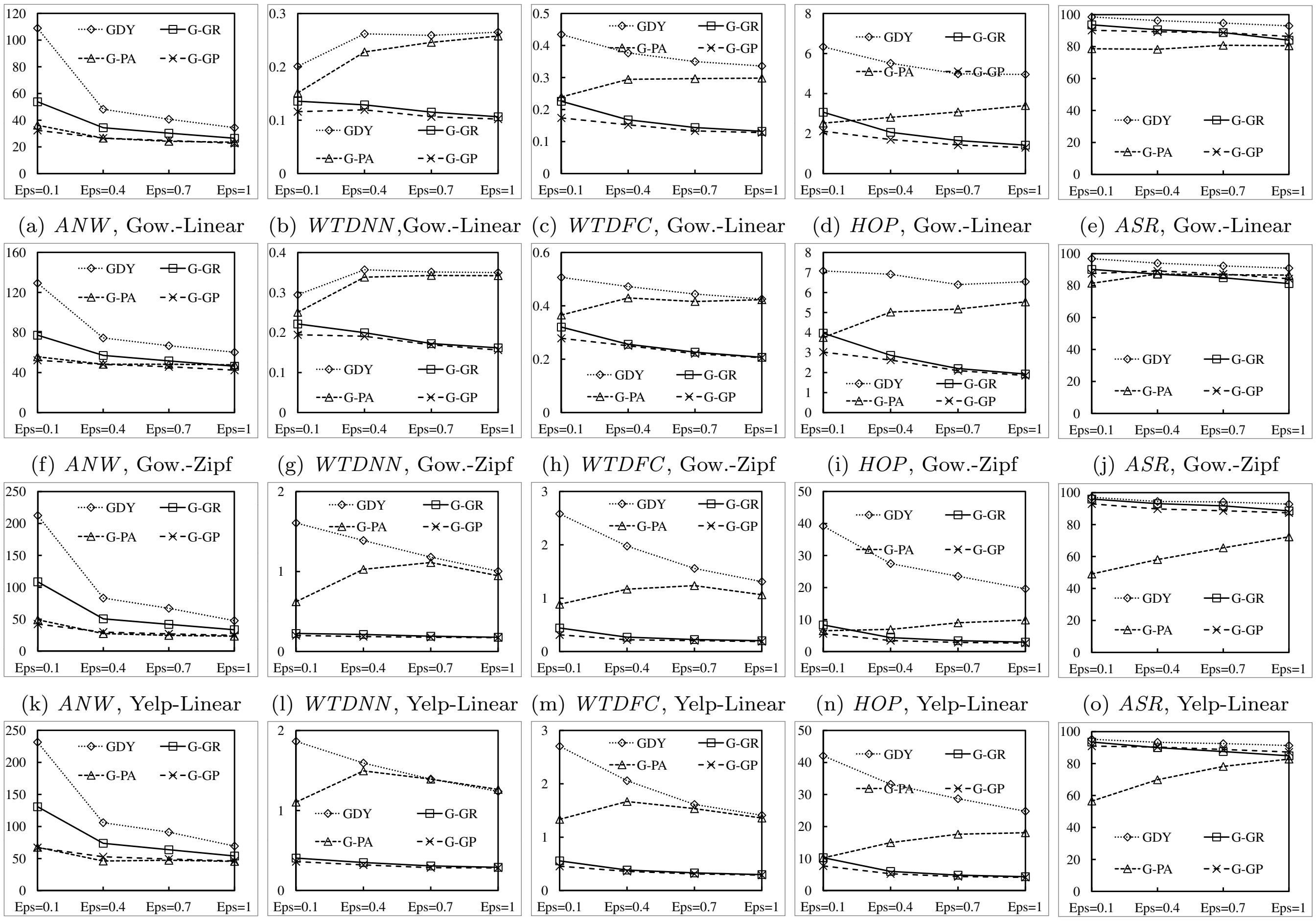
We evaluate the performance of $G\text{-}GP\text{-}Hybrid$ on the Yelp dataset by varying the maximum acceptance rate (MAR) and the expected utility EU (similar trends were observed for Gowalla). Figure 8a shows the results when varying MAR. As expected, a higher acceptance rate yields lower overhead and shorter travel distance, as workers are more willing to accept tasks. The GR size is also smaller, thus leading to a smaller network diameter and HOP value.

Interestingly, Figures 8c and 8d show that MAR has a significant effect on decreasing WTD. This effect is more pronounced than the drop due to increase in privacy budget ϵ, as observed in previous experiments. Figure 8e shows that the number of grid cells in the GR drops as MAR increases, due to increased utility of each cell. For the largest MAR value, a single cell is sufficient as GR, so $CELL = 1$.

Figure 9 measures the impact of increasing EU. To obtain a higher probability of task acceptance, the GR construction algorithm will generate a larger geocast region, leading to increased overhead, as measured by ANW, HOP and WTD.

7. RELATED WORK

While crowdsourcing has been present in both the research community (e.g., [25]) and industry (e.g., oDesk and Amazon Mechanical Turk), spatial crowdsourcing only recently received attention (e.g., [21], [15] and [14]).



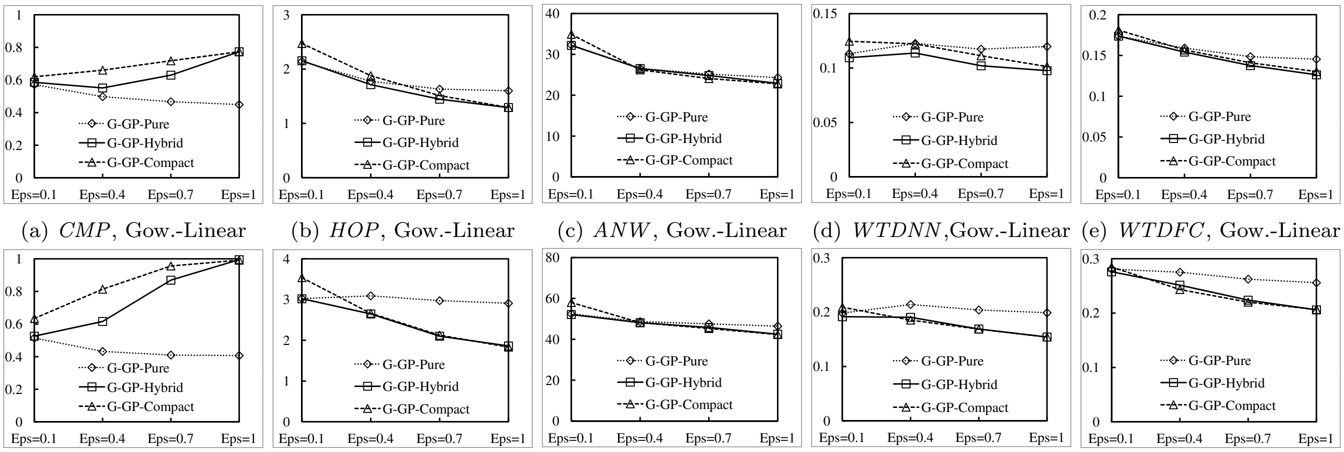
(a) *ANW*, Gow.-Linear (b) *WTDNN*,Gow.-Linear (c) *WTDFC*, Gow.-Linear (d) *HOP*, Gow.-Linear (e) *ASR*, Gow.-Linear

(f) *ANW*, Gow.-Zipf (g) *WTDNN*, Gow.-Zipf (h) *WTDFC*, Gow.-Zipf (i) *HOP*, Gow.-Zipf (j) *ASR*, Gow.-Zipf

(k) *ANW*, Yelp-Linear (l) *WTDNN*, Yelp-Linear (m) *WTDFC*, Yelp-Linear (n) *HOP*, Yelp-Linear (o) *ASR*, Yelp-Linear

(p) *ANW*, Yelp-Zipf (q) *WTDNN*, Yelp-Zipf (r) *WTDFC*, Yelp-Zipf (s) *HOP*, Yelp-Zipf (t) *ASR*, Yelp-Zipf

Figure 4: Comparison of *GR* construction heuristics by varying ε

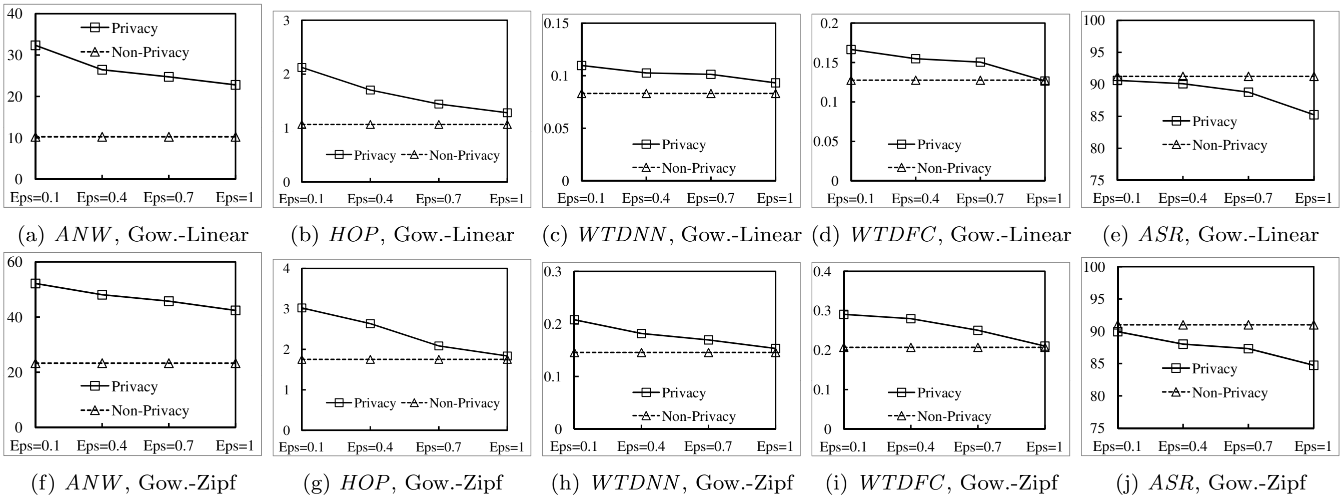


(a) *CMP*, Gow.-Linear (b) *HOP*, Gow.-Linear (c) *ANW*, Gow.-Linear (d) *WTDNN*,Gow.-Linear (e) *WTDFC*, Gow.-Linear

(f) *CMP*, Gow.-Zipf (g) *HOP*, Gow.-Zipf (h) *ANW*, Gow.-Zipf (i) *WTDNN*, Gow.-Zipf (j) *WTDFC*, Gow.-Zipf

Figure 6: Comparison of compactness-based heuristics by varying ε

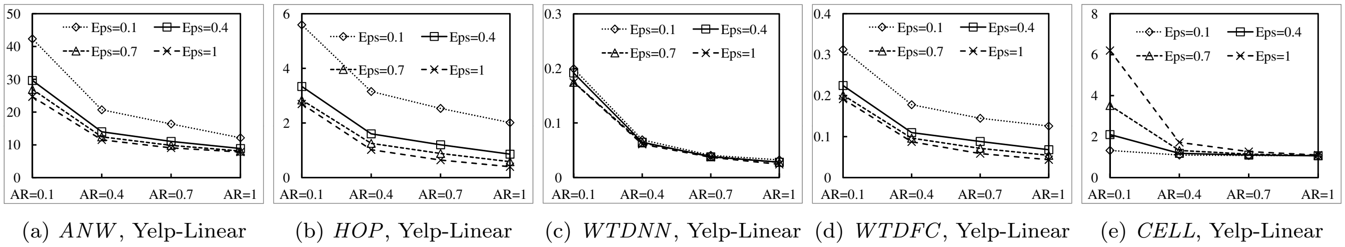
Location privacy has been studied extensively. Some techniques focus on evaluating queries in a transformed space [16, 28] which hides the real placement of data points. Other techniques make use of cryptographic protocols such as private information retrieval [7]. Another category of methods focuses on location cloaking, e.g., using spatial k-anonymity [9, 20], where the location of a user is hidden among k other users. The mentioned techniques assume a centralized ar-



(a) ANW, Gow.-Linear (b) HOP, Gow.-Linear (c) $WTDNN$, Gow.-Linear (d) $WTDFC$, Gow.-Linear (e) ASR, Gow.-Linear

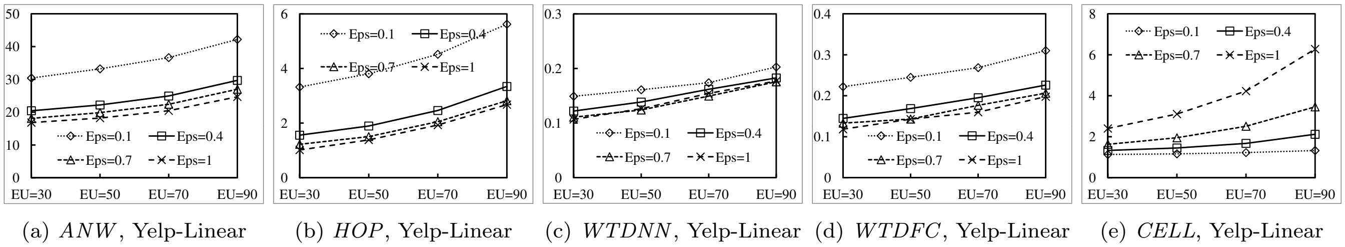
(f) ANW, Gow.-Zipf (g) HOP, Gow.-Zipf (h) $WTDNN$, Gow.-Zipf (i) $WTDFC$, Gow.-Zipf (j) ASR, Gow.-Zipf

Figure 7: Overhead of privacy (*G-GP-Hybrid*) compared to non-private algorithm



(a) ANW, Yelp-Linear (b) HOP, Yelp-Linear (c) $WTDNN$, Yelp-Linear (d) $WTDFC$, Yelp-Linear (e) $CELL$, Yelp-Linear

Figure 8: Performance of geocast algorithm (e.g., *G-GP-Hybrid*) by varying MAR (Yelp-Linear)



(a) ANW, Yelp-Linear (b) HOP, Yelp-Linear (c) $WTDNN$, Yelp-Linear (d) $WTDFC$, Yelp-Linear (e) $CELL$, Yelp-Linear

Figure 9: Performance of geocast algorithm (e.g., *G-GP-Hybrid*) by varying EU (Yelp-Linear)

chitecture with a trusted third party, known as location anonymizer. A major issue with these approaches is that the anonymizer becomes a single point of attack. Thus, some techniques focus on peer-to-peer anonymization [2, 8].

While location privacy has largely been studied in the context of location-based services, only a few studies focused on privacy for participatory sensing (PS) [13, 11, 12, 4]. The focus of [13] is to privately assign a set of spatial tasks to each worker while other works [11, 12] focus on preserving privacy in a PS campaign during data collection (i.e., how participants upload the collected data to the server without revealing their identities). The closest work to ours is [4], 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. In contrast, we assume the SAT mode which has far better assignment precision.

8. CONCLUSION

In this paper, we introduced a novel privacy-aware framework for spatial crowdsourcing, which enables the participation of workers without compromising their location privacy. We identified geocasting as a needed step to ensure that privacy is protected prior to workers consenting to a task. We also provided heuristics and optimizations for determining effective geocast regions that achieve high task assignment rate with low overhead. Our experimental results on real data demonstrated that the proposed techniques are effective, and the cost of privacy is practical.

As future work, we will extend our framework to also protect privacy of task locations. Another challenging problem is to address PSD in the context of multiple time snapshots. Finally, we will focus on finding more sophisticated PSD structures that provide better accuracy than AG.

9. ACKNOWLEDGMENTS

H. To and C. Shahabi have been supported in part by NSF grant IIS-1320149, the USC Integrated Media Systems Center (IMSC), and unrestricted cash gifts from Google and Northrop Grumman. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of any of the sponsors such as NSF.

10. REFERENCES

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

[2] C.-Y. Chow, M. F. Mokbel, and X. Liu. Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. *GeoInformatica*, 15(2):351–380, 2011.

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

[4] 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.

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

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

[7] 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.

[8] G. Ghinita, P. Kalnis, and S. Skiadopoulos. Mobihide: a mobilea peer-to-peer system for anonymous location-based queries. In *Advances in Spatial and Temporal Databases*, pages 221–238. Springer, 2007.

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

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

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

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

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

[14] L. Kazemi and C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing. In *ACM SIGSPATIAL GIS*, pages 189–198, 2012.

[15] L. Kazemi, C. Shahabi, and L. Chen. Geotrucrowd: Trustworthy query answering with spatial crowdsourcing. In *ACM SIGSPATIAL GIS*, pages 314–323, 2013.

[16] A. Khoshgozaran, C. Shahabi, and H. Shirani-Mehr. Location privacy: going beyond k-anonymity, cloaking and anonymizers. *Knowledge and Information Systems*, 26(3):435–465, 2011.

[17] C. E. Kim and T. A. Anderson. Digital disks and a digital compactness measure. In *Proc. of ACM symposium on Theory of Computing*, pages 117–124, 1984.

[18] 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.

[19] F. McSherry and I. Mironov. Differentially private recommender systems: building privacy into the net. In *Proc. of ACM Intl. Conf. on Knowledge Discovery and Data Mining*, pages 627–636, 2009.

[20] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The New Casper: Query Processing for Location Services without Compromising Privacy. In *Proc. of VLDB*, 2006.

[21] 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.

[22] J. C. Navas and T. Imielinski. Geocastgeographic addressing and routing. In *Proc. of ACM Intl. Conf. on Mobile Computing and Networking*, pages 66–76, 1997.

[23] W. Qardaji, W. Yang, and N. Li. Differentially private grids for geospatial data. In *Proc. of IEEE Intl. Conference on Data Engineering (ICDE)*, 2013.

[24] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1990.

[25] J. Wang, G. Li, T. Kraska, M. J. Franklin, and J. Feng. Leveraging transitive relations for crowdsourced joins. In *Proc. of ACM SIGMOD*, pages 229–240, 2013.

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

[27] B. Yao, F. Li, and X. Xiao. Secure Nearest Neighbor Revisited. In *Proc. of ICDE*, 2013.

[28] M. L. Yiu, G. Ghinita, C. S. Jensen, and P. Kalnis. Enabling search services on outsourced private spatial data. *The VLDB Journal*, 19(3):363–384, 2010.