Obfuscation At-Source: Privacy in Context-Aware Mobile Crowd-Sourcing

THIVYA KANDAPPU, ARCHAN MISRA, SHIH-FEN CHENG, RANDY TANDRIANSYAH, and HOONG CHUIN LAU, Singapore Management University

By effectively reaching out to and engaging larger population of mobile users, mobile crowd-sourcing has become a strategy to perform large amount of urban tasks. The recent empirical studies have shown that compared to the pull-based approach, which expects the users to browse through the list of tasks to perform, the push-based approach that actively recommends tasks can greatly improve the overall system performance. As the efficiency of the push-based approach is achieved by incorporating worker's mobility traces, privacy is naturally a concern. In this paper, we propose a novel, 2-stage and user-controlled obfuscation technique that provides a tradeoff-amenable framework that caters to multi-attribute privacy measures (considering the per-user sensitivity and global uniqueness of locations). We demonstrate the effectiveness of our approach by testing it using the real-world data collected from the well-established *TA$Ker* platform. More specifically, we show that one can increase its location entropy by 23% with only modest changes to the real trajectories while imposing an additional 24% (< 1 min) of detour overhead on average. Finally, we present insights derived by carefully inspecting various parameters that control the whole obfuscation process.

CCS Concepts: • **Security and privacy → Privacy protections**; **Usability in security and privacy**; • **Human-centered computing → Empirical studies in collaborative and social computing**; *Empirical studies in ubiquitous and mobile computing*;

Additional Key Words and Phrases: Privacy, Mobile Crowd-sourcing platforms, obfuscation, trajectory, context-aware

ACM Reference Format:

Thivya Kandappu, Archan Misra, Shih-Fen Cheng, Randy Tandriansyah, and Hoong Chuin Lau. 2018. Obfuscation At-Source: Privacy in Context-Aware Mobile Crowd-Sourcing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1, Article 16 (March 2018), 24 pages. https://doi.org/10.1145/3191748

1 INTRODUCTION

In recent years, many operators (such as GigWalk and FieldAgent) have embraced "mobile crowd-sourcing" as a viable commercial paradigm, where individual workers are incentivized to perform a variety of location-specific urban tasks (e.g., reporting on the queuing times at various food outlets, picking up and dropping off packages and checking inventory or prices in retail stores). In such platforms, each crowd-worker typically "pulls" tasks from the entire list of available tasks, often selecting to execute tasks based on a proximity-based filter. More recently, an alternative "push" based approach has been suggested [5], where the crowdsourcing platform

Authors' address: Thivya Kandappu, thivyak@smu.edu.sg; Archan Misra, archanm@smu.edu.sg; Shih-Fen Cheng, sfcheng@smu.edu.sg; Randy Tandriansyah, rtdaratan@smu.edu.sg; Hoong Chuin Lau, hclau@smu.edu.sg, Singapore Management University, School of Information Systems.

2474-9567/2018/3-ART16 $15.00
https://doi.org/10.1145/3191748

proactively recommends tasks to best match an individual crowd-worker's anticipated mobility patterns (e.g., daily commuting behavior).

Experimental deployments have demonstrated [19] that such a mobility profile-driven, globally-coordinated "push" approach can significantly improve the productivity of workers. However, current push-based strategies have one drawback: by requiring detailed individual-specific movement traces from each worker (to generate personalized task recommendations), they heighten potential privacy concerns. Workers may be unwilling to have the mobile crowdsourcing application continually collect their location traces in the background, as such detailed movement traces can be used for tracking (including predicting future movement), stalking (ability to isolate favorite/recently visited locations) and unwanted profiling (exploit the knowledge gained from mobility traces for targeted advertising). This issue has garnered significant attention recently–e.g., Apple introduced some form of differential privacy-driven techniques (on iOS 10 and later versions) to obfuscate the location trajectories collected from a mobile device.

Prior studies have tackled the issue of "location privacy", specially in the context of location-based services (e.g., finding *n*earby restaurants through Google Maps). The proposed solutions span both (a) centralized: *k*-anonymity [26], differential privacy [31], usage of trusted server [16] and, (b) decentralized solutions [3]. In this paper, we consider the introduction of user-controlled obfuscation-based location privacy, specifically tailored to the *push* based mobile crowd-sourcing application. We present a new technique that has two unique features: (a) **No Central Trusted Server:** we do not assume (unlike approaches such as k-anonymity or differential privacy) the existence of any trusted middleman; instead, each worker performs it own location obfuscation independently, without *ever* needing to reveal its true location; and (b) **Obfuscation based on both global and personal movement behavior:** our location obfuscation model allows each worker to adjust his degree of obfuscation on a per-location basis, based on each location's *global popularity* and *personal sensitivity*.

Broadly speaking, our goal is to develop a technique by which a mobile crowd-sourcing client (i.e, App) can avoid revealing location details (especially ones that it deems sensitive), while still being able to take advantage of the push-based crowd-sourcing paradigm. Push-based crowd-sourcing sets up a novel tension between *application utility* and *privacy*, as the platform's ability to recommend low-detour tasks (which maximize a worker's productivity) inherently depends on its awareness of the worker's true movement patterns. Three unique characteristics of mobile crowd-sourcing make the problem more involved: (a) the recommendations for different workers are *coupled*: as the platform seeks to globally coordinate the per-worker task recommendations, the recommended tasks for a specific worker are implicitly affected by the movement trajectories reported by other workers; (b) the impact of obfuscated location reports on the resulting recommendation is also a function of the *spatial distribution* of tasks; and (c) in many scenarios, the price or reward that the platform offers for executing a task is tied to the popularity of the task's location (e.g., tasks in less popular locations have higher rewards [19]), and it requires the server to have an accurate estimate of the aggregated movement patterns.

To achieve the right balance between application utility and privacy, we propose a novel 2-stage obfuscation approach that utilizes the well-known randomized response [34] technique. Under this approach called *Location Obfuscation via 2-Step Randomized Response* or *LORR*, at the first stage, the client reveals its true location (l) to the platform along with $n-1$ other erroneous locations (chosen based on probability p via randomized response). The server aggregates such randomized responses to estimate the overall population-level visit distribution (e.g., to assign appropriate task rewards) for different locations and shares this with worker clients. In the second stage, each client generates an obfuscated location update by choosing one of these n locations (based on both the global popularity and personal sensitivity of these locations), so that the perceived obfuscation benefit is maximized. Key questions that naturally arise include: *(a)* How should the mobile client compute the obfuscation benefit, so as to ensure that its reported trajectory lies close enough to its real one (to ensure that it gets tasks that do not impose high travel detour), while still concealing its most sensitive locations? and *(b)* How does *LORR*

perform–i.e., what is the typical tradeoff between privacy gain (in terms of location entropy) vs. performance degradation (in terms of additional detour overhead) that *LORR* can achieve?

Key Contributions: The paper makes the following key contributions:

- *Multi-attribute Privacy Measure:* To support an obfuscation-based approach, we propose a novel model, where the privacy gain of a candidate obfuscated location is determined by both the location's *global popularity* as well as *user-specific* measures of the *sensitivity* of both the true location and the candidate obfuscated output. This dual determination is based on the observation that individuals are typically more concerned about revealing locations that are either *unique/less popular* (i.e., not many people visit, implying that side-channel information might more easily reveal the worker's identity) or *dominant* (i.e., places such as home or office where the worker usually has high residency times).
- *Tradeoff-amenable 2-Stage Obfuscation Framework:* We describe a per-user location obfuscation framework for mobile crowd-sourcing that considers both the anticipated quality of recommendations generated (i.e., the application utility) and the enhanced degree of location privacy achieved. The unique 2-stage strategy uses a randomized response technique to allow the server to obtain statistically accurate estimates of aggregated movement behavior, while providing each worker client the flexibility to chose the obfuscated location that it eventually reports. More specifically, clients compute the overall *obfuscation gain* of a potential alternative location as a linear combination of both the increased privacy (*entropy gain*) achieved and the likely *quality of the recommendations* that the platform will generate. Different choices of a single coefficient (denoted as q) allow users to tradeoff between these two conflicting objectives.
- *Empirical Validation using* TA$Ker*:* We validate *LORR* using two datasets: the primary one being a real-world campus-level (mostly indoors) deployment of the *TA$Ker* mobile crowdsourcing platform (440 users, more than 25,000 total tasks, duration of 4 weeks), with the secondary one being a city-scale movement pattern of workers (100 users, 4 weeks, over 25,000 trips, derived from commuter transit records). We first show the inherent privacy dangers of the push crowd-sourcing approach, even when users reveal only the locations where they complete tasks: an adversary has a much higher likelihood of recovering the worker's overall movement trace, compared to traditional pull-based crowd-sourcing solutions. We then quantify *LORR's* ability to provide a *linear* tradeoff between privacy gains and worker productivity: Fig. 1 summarizes this observed tradeoff for both the SMU WiFi and commuter transit data, obtained by varying q. Specifically, on the university campus, when $q = 0.5$, *LORR* can increase average location entropy of crowd-workers by 23%, while imposing an additional average 24% (< 1 min) detour overhead. When q=0.9, a user is able to gain 60% uncertainty, while experiencing a relatively modest 54% increase in the travel overhead (detour). Similar tradeoff is observed on the city-scale (outdoors) dataset (depicted in Fig. 1).
- *Comparative Performance:* Based on the *TA$Ker* dataset, we compare *LORR's* performance to prior privacy approaches. In particular, we show that: (a) compared to the differential privacy approach, *LORR* offers a better privacy-vs-detour tradeoff: to achieve similar privacy gain (as a function of change in entropy and adversary's knowledge with respect to the true user location), differential privacy incurs additional 59% of detour (while compromising entropy by 15.3%), and *LORR* gains 30% of entropy while incurring modest 29% increase in detour; (b) *LORR* also outperforms traditional k-anonymity techniques. In particular, k-anoymity is not only unable to provide privacy guarantees to all nodes (in fact, 45% of worker routes are not anonymized when $k = 4$), but also offers lower privacy gain (tradeoff frontier is on right to the tradeoff of *LORR*) as it does not consider the worker-specific sensitivity of different locations.

We believe that our work is the first to demonstrate a technique, which provides a tradeoff between individual-specific location privacy and application utility (detour) in context-aware mobile crowd-sourcing, without assuming any trusted intermediary.

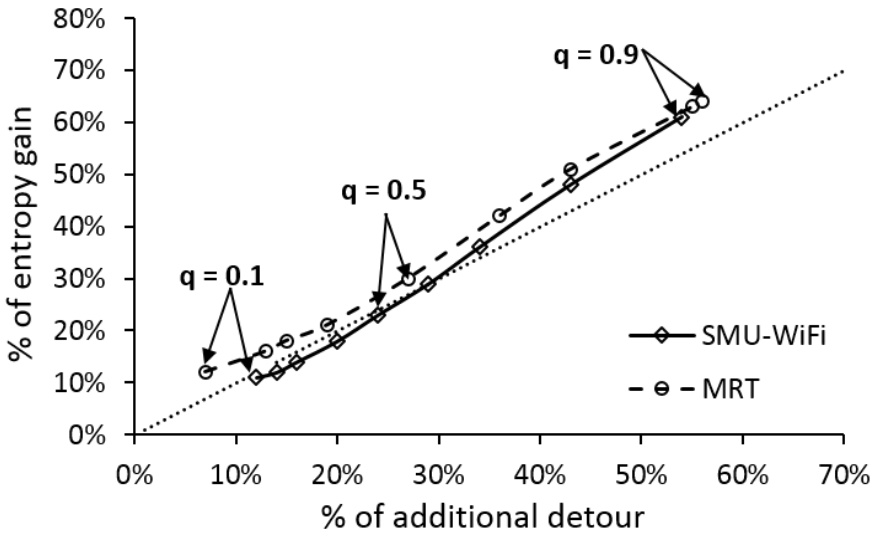


Fig. 1. *LORR* : Achievable Tradeoff between privacy and application utility

2 RELATED WORK

Mobile crowd-sourcing has become an increasingly popular paradigm for executing location-specific tasks in urban areas. Commercial operators, such as FieldAgent (www.fieldagent.net), GigWalk (www.gigwalk.com), and NeighborFavor (www.favordelivery.com) employ the conventional *pull* based models, where, crowd-workers independently browse the list of tasks sorted by proximity to their current locations. In contrary, *push*-based models [5, 6] leverage on the historical mobility patterns of users to predict likely movement behavior and then recommend tasks that lie close to the predicted paths.

Past work on anonymization focused on sanitizing the user data by removing/hiding personally identifiable information (PII). Techniques like k-anonymity [15, 25, 28] and its variants l-diversity [23] and t-closeness [22] were shown to be vulnerable to composition attacks [14], and do not provide desired privacy guarantees for larger k values without distorting the utility of the data [29]. Another major threat with these approaches is that the trusted third party in the middle becomes a single point of attack. Another alternative approach [20] requires workers to collaborate with nearby peers to generate aggregated (and thus non-personally identifiable) queries–this approach is appropriate only for the pull-based model where workers explicitly query for tasks. Alternately, micro-aggregation based techniques [17] (where the reports from multiple workers are aggregated together) have been suggested for concealing the true location of a crowd-worker. However, this model works for participatory sensing applications where workers choose their reporting locations; in contrast, in our model, the platform explicitly specifies the location associated with each available task.

In recent line of work, the concept of differential privacy [11–13] has been studied for mobility-aware applications [30–33]. The key idea here is to perturb the true location to produce an obfuscated location by applying Laplace noise calibrated using pre-defined parameters. To et al [31] used differential privacy specifically for mobile crowdsourcing. In this approach, a trusted intermediary hides individual locations but advertises a noisy estimate of the number of workers within a variable geographic region. The crowdsourcing platform then *geocasts* a task to all nodes within a certain designated region. Note that this approach tries to match each task instantaneously

to workers in the vicinity, unlike our push-based model which tries to recommend individual workers tasks that *may* align with their regular movement trajectory at some future time instant. Moreover, our goal is to empower each worker to make user-controllable, location-specific tradeoffs between privacy and application utility while revealing his trajectory, without relying on any trusted intermediary.

In contrary to the centralized server based mechanisms detailed above, user-controlled release of location data is studied in [4], where the authors assume a *pull* based model of mobile crowd-sourcing. The proposed system utilizes a mobile App, installed by a worker on her mobile device, that alerts the worker about the potential resultant privacy/entropy loss if she actually completed the task. This approach does not recommend tasks to the worker.

Several recent works considered entropy as a measure of privacy by using the frequency of visitations [2–4] to estimate the probability of a user being at a certain location. While these approaches use entropy as a quantifiable anonymity or uncertainty measure, in our work we primarily use it as a canary to calculate the perceived privacy – utility gain and subsequently to decide which location (out of n locations chosen probabilistically) to reveal to the crowd-sourcing platform.

In our work, we consider a push-based crowd-sourcing mechanism, where the continual update of user locations is essential to generate effective task recommendations. We shall develop a 2-stage obfuscation technique that has far better assignment precision, even with the perturbed trajectories. The approach presented in [27] is the closest to our proposed approach, and focuses on accurately estimating the number of people at subway stations and urban/rural area. This system describes the SpotME client that uses the randomized response algorithm (originally introduced in [34]) to mask a user's current locations by choosing multiple other erroneous locations. While *LORR's* first stage is very similar to SpotME, the resulting aggregate popularity count is merely an intermediate step for us, and helps facilitate the second step of user-controlled obfuscation (where the worker chooses to report one of multiplicate candidate locations, based on her sensitivity to the resulting privacy loss vs. quality of task recommendation tradeoff).

3 OVERVIEW

Our privacy-preserving mobile crowd-sourcing platform consists of two major components: (1) the *mobile client* (or simply *client*): i.e., the App installed on user's smartphone, which users use to receive and perform tasks, and which also performs the location sensing and reporting tasks described in *LORR*, and (2) *the back-end server/platform*: receives information from the mobile client, derives and predicts movement patterns, and generates task recommendations. The design of the platform is depicted in Fig. 2.

More specifically, the mobile client aims to obfuscate the current whereabouts of the user that it reports to the server, by considering both (a) the improved location privacy, and (b) the likely loss in the quality of the task recommendations it would receive, as a result of such obfuscation. While the entire crowdsourcing workflow contains 5 stages, the privacy-related actions *which constitute the novel contributions of this paper* are manifested in 2 stages – (1) using Randomized Response to enable the server to derive global counts of workers at different locations, without requiring any client's true location, and (2) the client subsequently reporting a (potentially) obfuscated location that maximizes the obfuscation benefit.

The entire workflow for our context-aware mobile crowd-sourcing platform is described below.

Step 1: Client Query: This query is initiated by the mobile client, prior to obfuscating the user's "true" location. This query helps the client to obtain an estimate of the popularity distribution (i.e., number of devices residing in each region) of locations in its vicinity. In this step, the client uses the well-known randomized response

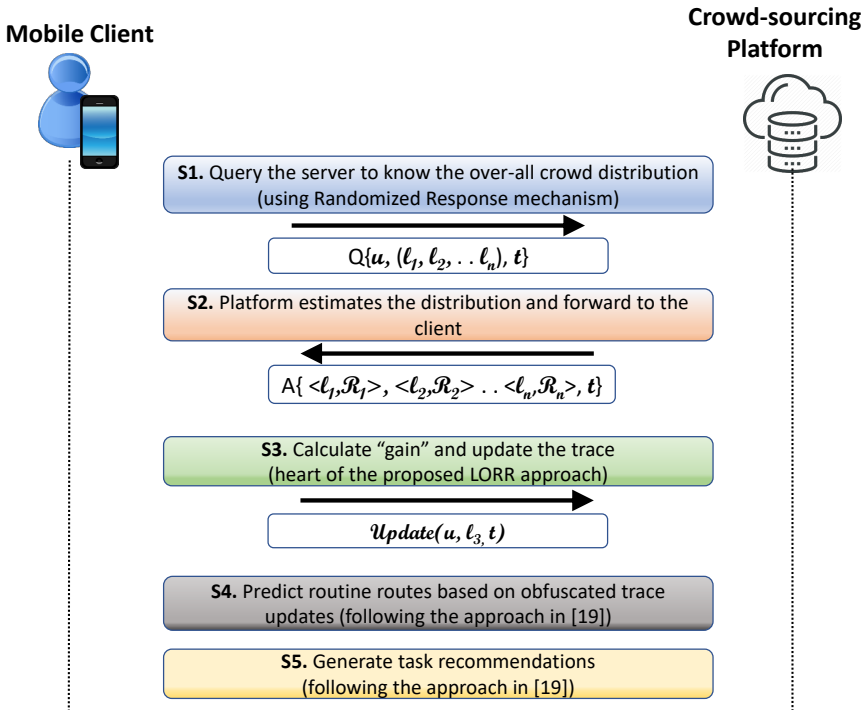


Fig. 2. Proposed System Architecture

technique [34] to choose a varying number of erroneous locations (along with its true location) that it sends to the server, requesting the global residency count of those locations.

Step 2: Platform Response: In this step, the server (i.e., the crowd-sourcing platform) responds with its estimated location count for each of these locations. Note that the randomized response approach allows the server to estimate the true residency counts effectively by utilizing knowledge of p, the probability of coin flipping (see §4.2 for more details), compared to alternate approaches. For example, an approach where clients preferentially choose only popular locations will result in a skewed estimate.

Step 3: Client-side Obfuscation: This stage is the heart of *LORR's* novel privacy strategy. Upon receiving the popularity scores from the platform, the client now picks one of these locations (so as to maximize the perceived obfuscation benefit), and updates the platform with this obfuscated location. As we shall see in §4.3, the selection of the obfuscated location is based on a combination of (a) the location's popularity score (determines *uniqueness* of a location, globally), (b) the location's entropy (measured based on his residency time at each location in the past–this measure captures the client's *dominant* locations), (c) the historical spatial distribution of tasks and (d) the distance from the client's true location.

Step 4: Trajectory Prediction: In this paper, the platform uses the state-of-the-art route prediction algorithm (previously described in [19]) on these obfuscated location updates (traces), received from all the users, to predict their expected trajectories.

Step 5: Task Recommendation: The platform now generates task recommendations for individual workers, based on client's predicted trajectories, while minimizing the additional detour overhead. The platform is based

on the recommendation engine proposed by Cheng et al. [7], which recommends tasks so as to maximize the expected total rewards (cumulatively across all workers), while adhering to (i) a user-specific detour bound and (ii) stochastic uncertainty in a user's movement trajectory. Note that this framework recommends tasks by optimizing globally, based on future movement trajectories; this is quite distinct from other privacy-aware approaches such as [31], which match tasks to workers on a per-task basis, as soon as each such task arrives.

4 PRIVACY FRAMEWORK

In this section, we describe the core privacy-related components of the *LORR* crowd-sourcing framework–i.e., steps 1-3 of the 5-step workflow for *LORR* outlined in Section 3. This section is organized as follows: in §4.1, we describe how the mobile clients would choose locations to query, in §4.2, we describe how the platform would periodically estimate the true popularity distribution and its accuracy, in §4.3, we describe how the client would obfuscate its current location and update the platform. In addition, for completeness, in §4.4, we summarize the mechanisms for predicting each user's trajectory and for generating trajectory-aware recommendations for each worker.

4.1 Client Query

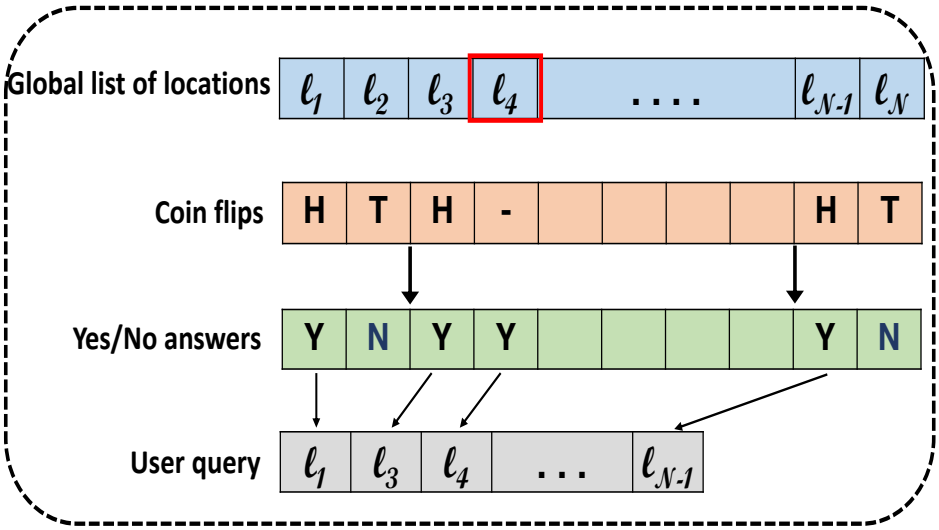


Fig. 3. Randomized Response Mechanism

The mobile client's main goal is to obfuscate its current location (so as to enhance the privacy guarantees without suffering a significant degradation in the quality of the recommendations). However, the client must consider the sensitivity of a location not only based on the user's perspective (i.e., is this a dominant location that reveals key lifestyle details of the user?) but also on its global popularity (i.e., uniqueness). For example, revealing a particular location that is very popular and visited by many other individuals (e.g., a campus cafeteria or a train station) is unlikely to incur significant privacy loss: the user effectively "hides" among the larger pool of users. While the client can obviously determine the individual client's sensitive locations, the need to determine a location's global popularity leads to an apparent catch-22 situation: to establish whether the location is heavily

visited or not, the user has to directly query the platform (this query will implicitly reveal his whereabouts), which in turn needs to have accurate knowledge of the spatio-temporal location distribution of the aggregate set of workers.

To mitigate this situation, we adopt the well-known randomized response technology, based on a biased-coin flipping mechanism. Randomized response offers two benefits: (a) the client queries for the popularity of additional erroneous locations (in addition to its true location), thereby concealing its true location, and (b) it allows the platform to accurately estimate the true aggregate popularity count of workers at each of these distinct queried locations.

The randomized response algorithm works as follows (the strategy is illustrated in Fig. 3):

- The client chooses a set of minimum n locations from the global list (e.g., all the geographic postal-districts in a city) of locations (total of N). Among those n locations, one is the user's actual location (marked in red in Fig. 3).
- For each of the N locations in the global list, randomized response algorithm is executed.
  - The client flips a biased coin (with the probability p of head) and say he is present at the location if *head* appears.
  - Or says the truth if $tail$ appears (with probability $1 - p$).
- The client creates a list of *plausible locations*, consisting of the subset of N locations where he declares himself to be present. The query to the platform will consist of all elements of this plausible location list.

4.2 Estimating Popularity Distribution

The platform first collects all the queries received from the clients and estimates the proportion of users currently present (i.e., footfall) at each of the N locations. If the platform receives a report for location l from a worker, there can be two possibilities: (a) *head* appeared during the coin flip, and the client thus declared *"yes, I'm present at location l"* or (b) $tail$ appeared during the coin flip, and he said declared *"I'm present at location l"*, only if l is his true location. Note that because this flipping is done by the user, the platform does not directly know the true location of the worker.

Let the probability of *head* appearing be p (note: p is assumed to be a common value that is shared by *all* clients).

- Because a client responds with a "no", iff $tail$ appears during the coin flipping, the observed proportion of "no" answers can be formulated by $\widehat{R_{no,l,t}} = (1-p).R_{no,l,t}$, where $\widehat{R_{no,l,t}}$ denotes the observed proportion of users who claim to be *not* at location l at time t (obtained by dividing the number of "no" responses by total number of users) and $R_{no,l,t}$ denotes true proportion of the same.
- Similarly, as the observed proportion of "yes" comprises of the truthful "yes" answers (does not depend on the coin flip) and the responses that falsely report "yes" when *head* appears, we have: $\widehat{R_{yes,l,t}} = R_{yes,l,t} + p.R_{no,l,t}$, where $\widehat{R_{yes,l,t}}$ and $R_{yes,l,t}$ denote the observed and true proportion of the users reported at location l at time t.

From these responses, the true proportion of "no" users ($R_{no,l,t}$)and "yes" users ($R_{yes,l,t}$), at location l and at time t can be estimated as :

$$R_{no,l,t} = \frac{\widehat{R_{no,l,t}}}{(1-p)};\ R_{yes,l,t} = \frac{\widehat{R_{yes,l,t}} - p}{1-p} \tag{1}$$

By periodically estimating the true population scores, the platform is now able to send the respective scores of the queried locations to the clients.

4.3 User Obfuscation

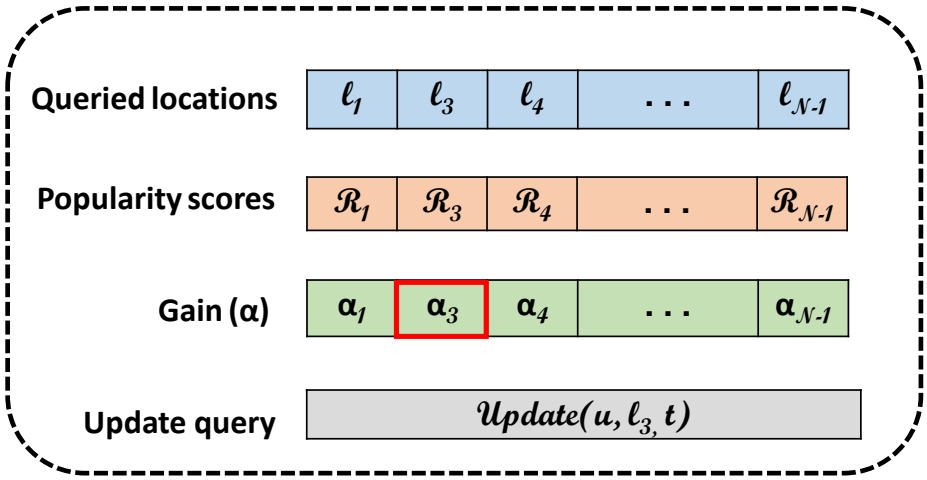


Fig. 4. Strategy for User Obfuscations

This step is the heart of the *LORR* framework. After receiving the respective popularity scores of each location that it queried, the client now has to select one of these candidate locations as its *reported location*–i.e., the one that it will indicate to the platform as its current location. To evaluate the overall benefit of choosing a location (both in terms of the additional privacy achieved and the expected drop in the quality of the recommendations), we propose a novel mechanism that combines the gain factors into a single score, *obfuscation gain*. To express the importance of privacy guarantees and quality-of-recommendations, we introduce a parameter $q \in [0,1]$ *that can be set by each individual agent*. The total *obfuscation gain* of a user u by choosing location i over his true location l is then given by:

$$\alpha_{u,i} = q \cdot R_{yes,i,t} \cdot \frac{H_{u,i}}{H_{u,l}} + (1-q) \cdot F_{i,t} \cdot \frac{D_u}{d(i,l)}, \tag{2}$$

where $H_{u,i}$ captures the sensitivity of the location i of user u (as a measure of entropy) while $H_{u,l}$ represents the sensitivity of his actual location l. Further $F_{i,t}$ considers the historical task distribution of location i, D_u denotes the detour budget of user u and $d(i,l)$ estimates the distance between the true and chosen location l. Note that from the list of n locations queried initially, the user will choose the location that has higher gain values to update the platform (depicted in Fig.4 and the corresponding maximum α is represented in red).

The first operand considers the privacy gain, as a function of (a) the popularity of the location – how many other users are available there (choosing the locations with higher popularity scores will enable the user to hide among many other users), and (b) relative entropy gain (or loss). The second term considers the quality of the recommendations that will be generated based on the obfuscated locations, as a function of (a) historical task distribution of the location of interest (as an indicator of task availability at the obfuscated location), and relative consumption of the detour budget (as each user can only spare D minutes to perform tasks on a day) – higher the fraction, lower the over-all detour. The overall gain is a dimensionless score that takes linear combination of the two gains, weighted by q:

- When $q \rightarrow 0$, the user is primarily concerned about the quality of the recommendations, and will end up choosing locations that are closer to the actual location and that have had higher number of tasks posted in the past.
- Conversely, when $\alpha \rightarrow 1$, the user is less focused on the quality of the recommendations, and ends up choosing locations that yield higher entropy gain (places that he usually does not visit very often) and that are globally more popular, even though these places may be farther away from his current location.

4.4 Trajectory Prediction & Task Recommendation

Once the platform receives the intermittent, potentially-obfuscated, location updates from the individual workers, it is then responsible for predicting each individual's movement trajectory, and then generating appropriate task recommendations. The route prediction algorithm first transforms the raw data to routes. It extracts *reference locations*–i.e., the location where the user spends the largest amount of time in any given time segment (for the evaluation results in this paper, we consider 30 minute time segments for the campus dataset, and 60 mins for the outdoor commuting dataset). The algorithm then formulates the movement pattern of a user in a given time window as a *transition graph* and finds the best probable k routes (represented as a sequence of reference locations), based on the past traces of movement trajectory of the worker. Given a specific route (i.e., ordered list of reference locations), the real movement path is constructed by computing the standard shortest path between each consecutive pair of reference locations.

Subsequently, the task recommendation engine utilizes the following inputs: (1) the topology of the environment: represented as a graph, where stay points are denoted as nodes and links represent travel path, (2) set of clients: each client is characterized by the detour time limit and the routine trajectories inferred from the reported locations (routine trajectories can be probabilistic), and (3) set of tasks: each task is characterized by its location, validity time window, execution and reward. Task recommendation is formulated an integer linear program, which is essentially a special variant of the routing problem, with possibly additional side constraints–e.g., mandating that a task be recommended to exactly η workers or not at all (if not possible). Because the exact optimization problem is computationally intractable, a Lagrangian relaxation algorithm, described in Cheng et al. [7], is utilized to derive a heuristic solution for practical, large-scale crowd-sourcing scenarios. This setup has been proven in our past field experiments [18] to be robust and reliable, and we can directly adopt it without major changes.

5 EVALUATION

The clients in the system obfuscate user locations (before revealing them to the server) in such a way that the quality of the recommendations they receive (made to match their trajectories) is not greatly diminished, while still getting desirable privacy guarantees. To empirically evaluate the *LORR* framework, and understand the performance tradeoffs associated with various parameter choices, we will be measuring the following metrics.

- *Location Fidelity:* How accurately does the trajectory defined by the set of *reported locations* (obfuscated) match with the user's actual mobility patterns?
- *Obfuscation Tradeoffs:* How effectively can the worker obfuscate his true location while considering the desired trade-offs? In particular, we study how the parameter q controls the tradeoffs between privacy and utility, on a per-worker basis, and the impact that q has on the following:
  - Gain in entropy as a result of the obfuscation strategy
  - Increase in detour overheads incurred as a resultant of recommendations generated based on obfuscated traces

  - Change in worker productivity (measured by the average amount of rewards earned in a unit of time – $ earned per minute)
- *Randomized Response Performance:* How accurately can the platform infer the true popularity distribution of a location using the randomize response-based queries periodically issued by individual workers? More specifically, we are interested to see the impact of the probability p on the following:
  - Accuracy of the popularity distribution estimation by the platform.
  - Vulnerability of the users to inference attacks.

5.1 Experiment Study Details

To study the above mentioned points, we run experiments on two different datasets: (a) WiFi based traces of worker movement while using a mobile crowd-sourcing platform on the SMU campus, and (b) traces of commuting using Singapore's public transport data, based on trip transaction records on public buses and trains (MRT). The SMU WiFi data is our primary dataset, and will be used not just to study the effectiveness of *LORR*, but also to compare its performance to prior privacy approaches used in mobile crowd-sourcing. The Bus/MRT is used as a secondary, lower-fidelity dataset, primarily as a means to validate our key findings–i.e., to ensure that our findings on SMU's primarily-indoor campus are replicated even under a hypothetical wide-area, city-scale deployment.

SMU WiFi data: Our primary dataset involves large-scale traces of indoor movement captured at Singapore Management University– an urban campus consisting of 4 distinct academic buildings, 1 library, and 1 administrative building. As part of a large-scale experimental mobile testbed [24] (which comprises a pool of over 3,500 undergraduate students), the university operates a passive indoor location service [21]. The location data is obtained using server-side Wi-Fi fingerprinting techniques and offers a median accuracy of 6-8 meters, with a latency of 5 seconds.

For finer-grained localization of users in such an indoor environment, we consider a collection of landmarks in each floor level of the building and group them as a section (each level can have more than one section). Each section is called a *region* or *location* throughout the rest of the paper. During the first stage of the randomized obfuscation process, each user will choose n locations from the building where they are currently located. For example, if the true location is on level 3 of building A, then all the erroneous locations chosen will be constrained by the same building A. For the purpose of this experiment, we consider the indoor-localization mobility data for the entire month of March, 2017.

To see the efficacy of the two-level obfuscation on the quality of the recommendations received, we consider the existing mobile crowd-sourcing platform called *TA$Ker* that has been operationally deployed on our urban campus for 2.5 years, with a participation base of more than 1,000 users [18, 19]. *TA$Ker* utilizes the above mentioned historical mobility traces (unobfuscated) of the users to generate trajectories and then match the available list of tasks to the users' predicted trajectories so as to maximize the task completion in expectation. These tasks are location specific and require the user to navigate to the task location in order to perform and submit the response – e.g., report us on the status of the rubbish bin near level 2 lift lobby of SIS building, check whether the seminar room 3.1 of SIS is occupied. In this paper, we use the mobility traces collected from the *TA$Ker* platform as the data source for an empirical, data-driven study to understand how the quality of the recommendations and user efficiency will be affected (compared to the *TA$Ker* baseline) if users utilize the *LORR* framework to modify the location data that they reveal to the crowd-sourcing platform.

Bus/MRT data: We also use the public transport data (bus and MRT/train) of 5.1 million commuters of Singapore, collected during the month of August 2013. This data consists of comprehensive records of the tap-in (entry) and tap-out (exits) details (i.e., bus stop or train station) of approx. 127 million trips made by commuters during that

month, using both public buses and the train network. The average trip distance and sojourn times are 7.4 km and 20.03 minutes, respectively. The data spans 4608 bus stops and 131 MRT stations. Each trip record captures the unique card ID of a user, tap-in time, boarding station/bus stop, tap-out time and alighting station/bus stop.

Using the data, we first construct the network map as a graph where each bus stop and MRT station is a node, and an edge between 2 nodes exists if at least one bus or MRT directly connects them. Now we hypothesize a scenario, where residents of Singapore are using a *TA$Ker* like App to report on the status on various municipal services related facilities. We assume the residents can report on the following: (a) historical sites, monuments and museums (e.g., checking the condition of these heritage buildings), (b) heritage trees (e.g., report on fungal infestation), and (c) bicycle parking racks (e.g., check and report on number of vacant bicycle stands). Based on these assumptions, we identify 805 unique locations of above mentioned facilities/resources as task locations spread across the city.

To set the baseline (no privacy) we deduce the origin-destination pairs of each trip as stay locations and generate recommendations for 100 most frequent travellers. We assume that the user performs the tasks either before boarding or after alighting from the bus/train, and that a user will be willing to spend a maximum of 15 minutes or 1km worth of walking distance (the detour budget) in performing tasks. During the obfuscation phase the user first chooses n erroneous stations/stops that lie within ± 3 hops along the same transport service (the locations lie along the same line of MRT/bus route) of the current trip. He then updates his obfuscated location to the crowd-sourcing platform based on previously described *LORR* steps. For brevity, we only provide the performance trade-off results derived based on this dataset in Section 5.4.

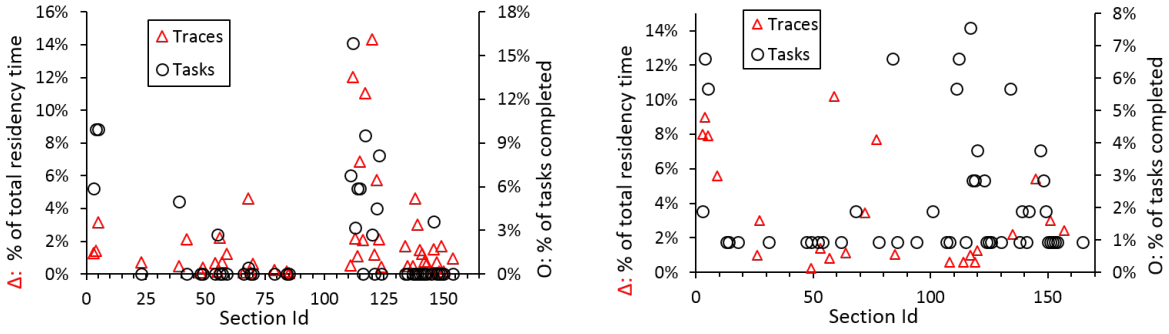
5.2 Implicit Revelation of Task Locations

Before we evaluate the efficacy of the proposed privacy mechanism for our mobile crowd-sourcing system, in this section we consider another potential breach of privacy that occurs when the users implicitly reveal their locations while performing and submitting the tasks' responses. More specifically, we are interested to see whether an adversary can reproduce the trajectories (or patterns of movements) by only observing the task completion locations in space and time. Our investigation is shown in Fig. 5a and 5b, where we depict the percentage of the total residency time and submitted reports at each location on our campus for a particular *push* and *pull* class user of *TA$Ker* (we consider the most active user/highest performer from each category). The X-axis denotes unique section ID of all the sections on our campus. While the primary Y-axis captures the percentage of total residency time (deduced from the true mobility traces) and the secondary Y-axis depicts the percentage of tasks performed (calculated based on the submitted responses).

For *push* class users (Fig. 5a), we can see that there are strong correlations between task submission locations and the locations where the user spent large amount of time (the correlation score for residency time & percentage of tasks performed is 0.65). This is not surprising as recommendations are generated by considering predicted dwell locations. On the other hand, for the *pull* class user (who has to browse and choose his own tasks), the correlation is absent (a mere 0.04), as shown in Fig. 5b. This may be due to the opportunistic nature of the *pull* class user, who performs tasks spontaneously at the locations while being on the move. Although the figures illustrate only a single user for each class, we have confirmed that these observations are representative: similar correlation scores were observed across all the users in the respective pools (push: 0.54–0.68 and pull: 0.03–0.08).

These results indicate that while the *push − based* mobile crowd-sourcing offers compelling benefits, it does pose potential privacy concerns, even with just the task performance locations. This alone is not entirely a weakness, as we have shown that the efficiency of the push-based mobile crowd-sourcing is derived from the fact that recommendations are generated based on highly accurate trajectory predictions. Instead of scrapping all the benefits that a push-based system would bring, we thus recommend an active privacy management scheme:

sacrificing some less-sensitive location traces, reaping the benefits of push-based system in those places, while protecting the most sensitive locations.



(a) Push class

(b) Pull class

Fig. 5. The true trajectory and completed task locations of (a) *push* and (b) *pull* class user

5.3 Stage-01: Client Query via Randomized Response

5.3.1 Frequency of Client Query: The accuracy of the ground-truth estimation depends on two parameters, (a) the frequency in which the client queries the platform, and, (b) the probability p of head appearing in coin flipping. One may expect that higher frequency will lead to accurate popularity estimation. But the trade-off between the accurate estimation and localizing the user (i.e., privacy invasion) naturally occurs when the frequency is high and the user is stationary – the passive adversary (who observes the periodic location queries and tries to infer/localize the user based on the possible transitions of the user between subsequent queries) can easily localize the user by matching two subsequent queries when the user is stationary. On the other hand, choosing coarser-grained frequency will render the platform to be unable to capture transitions of users. Hence, choosing the right frequency is important to balance both privacy guarantees and accurate ground-truth estimation.

In this paper, we study how the mobility pattern of a student evolves on our urban campus by translating the observed raw traces to trajectories. We use the Wi-Fi indoor localization data described earlier. We define a user's residency period in a location as an *episode*. These spatio-temporal occurrences of residency episodes are observed longitudinally, per user basis, to derive the stay-time distribution. We eliminated the devices (or users) that are extremely stationary (such as laptops) by avoiding any stay episode that is longer than 4 hours (typical class duration is 3 hours and 15 minutes). We also eliminated episodes that are shorter than 5 minutes in order to remove very transient users. Our goal here is to see whether even after this filtering, the episodes show the generic pattern of user movements on our campus. These insights are used to deduce a right location update frequency that captures the majority of transitions that users typically make, without compromising their location details. We report the average stay time distribution (per user, per day) measured in minutes in Tab. 1. We consider 3 classes of users: (1) all the users who turned *ON* their WiFi interface, (2) the users enrolled with the *TA$Ker* platform, and (3) relatively larger but more representative group of users who are signed up with the LiveLabs testbed [24]. From the study, it suggests that choosing the frequency = 5 minutes (the value we chose earlier to filter out the transient users) does not distort the mobility patterns. As a sanity check we report the distributions for 3 different types of locations (1) the whole campus, (2) food-courts, and (3) classrooms. It rightly

captures the average stay distributions (a typical stay duration in a classroom would span for 3 hours in a day vs. a user would spend nearly an hour at food-courts during the lunch and tea breaks) across various locations of interest. Throughout the rest of the analyses, we set the frequency of querying at 5 minutes.

Table 1. Stay time distribution – per user, per-day (in minutes)

Location	All				*TA$Ker*				LiveLabs			
	Min	Max	Avg	Med	Min	Max	Avg	Med	Min	Max	Avg	Med
Whole Campus	5	527	158	137	5	486	188	173	5	512	184	169
Food-court	5	405	30	16	5	171	25	13	5	174	28	15
Classrooms	5	476	78	58	5	288	68	521	5	236	75	58

5.3.2 Accuracy of the Ground-truth Estimation: Studying the accuracy of estimation of the true popularity distribution is important because the user's decision to choose a location to obfuscate mainly depends on how many active users are available there. To capture the accuracy metric, we calculated the error (denoted by e_l) in estimation (obtained and forwarded to the client by the platform using Eq. 1) as a difference from the actual ground-truth (which can be obtained for our studies using the non-obfuscated/original traces collected by the *TA$Ker* system):

$$e_l = \frac{\frac{\sum_{\forall t \in T}(R_{l,t} - \overline{R}_{l,t})}{\overline{R}_{l,t}}}{|T|}, \tag{3}$$

where $\overline{R}_{l,t}$ denotes the actual ground-truth and T represents the collection of timestamps in which the platform estimated the popularity scores at location l in a day. Note that e_l is a location specific metric (in contrary to the conventional metric that considers the over-all error – averaged across all the locations, at a given time), meaning we're observing it as a per location score. The motivation behind this way of capturing the error is due to the fact that different locations exhibit their own intrinsic nature of occupancy – e.g., places like library and food-courts have relatively smoother traffic while classrooms may have bursty arrival of students (occupied only during the class timings). We shall show how this intrinsic nature of location occupancy affects the error in estimation.

The probability p determines the number of locations to be queried (higher the p value, more the locations to be queried). While higher values of p enable the users to mask themselves among multiple other locations, lower values of p allow the platform to predict the population density more accurately (mainly because the number of people who falsely report their presence at a location is smaller, compared to higher values of p). To see the effect of p on prediction error, we plot in Fig. 6, the error in estimation (calculated using Eq. 3) for various p values. As mentioned earlier, to see how the intrinsic nature of the locations (i.e., temporal variations on ingress and egress flow of users of a location) affect the error in prediction, we consider three different locations: (1) food-court (we can expect to have a smoother traffic here as it's accessed by the users commonly), (2) classrooms (where a bursty traffic is expected during the scheduled class time), and (3) the whole campus. For sanity checking, we plot the fraction of distinct users appearing at each minute (obtained by dividing the number of users appear in a minute by total number of unique users observed at the location on the same day) at the food-court (in Fig. 7a) and a classroom (seminar room 2.2 in one of the academic building – depicted in Fig. 7b) on a typical Tuesday in March (note that each data point is an average of 4 Tuesday's user availability at that location). We observe from Fig. 7a that the flow of traffic in food-court is smoother than the flow in classroom – the traffic flow nature we observed here is validated against the publicly available campus class timetable – this particular seminar room

2.2 is occupied on Tuesdays during the first semester of 2017 between 08:15 and 11:30 and again between 15:30 and 18:45.

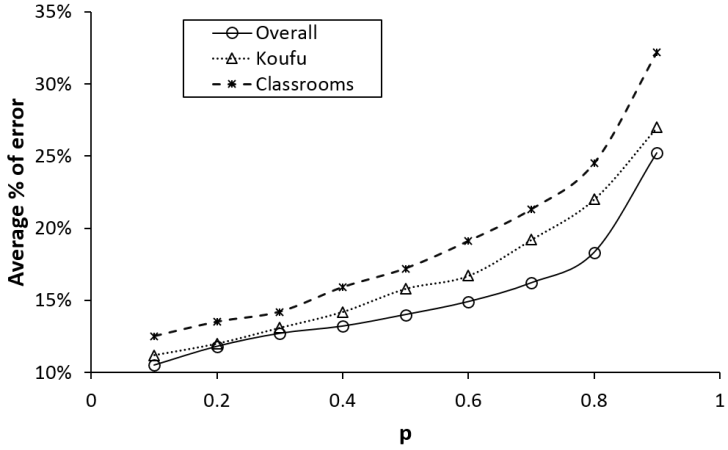
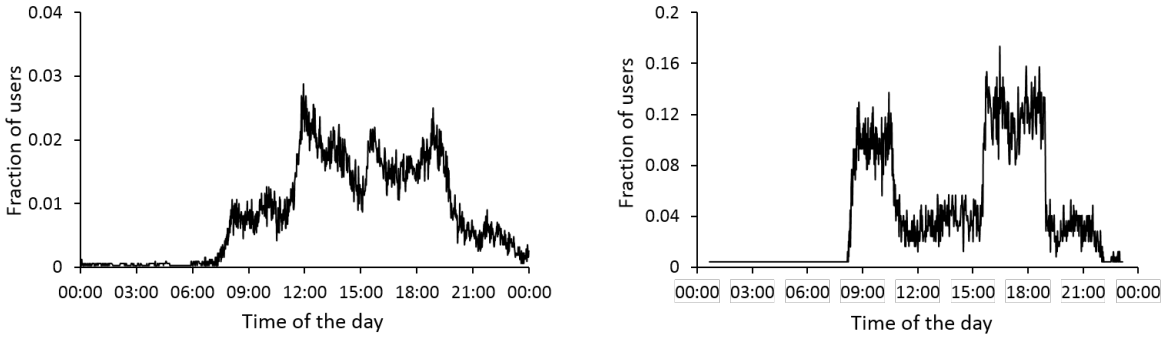


Fig. 6. Error vs. p

From Fig. 6, and using the auxiliary details we found about various locations (depicted in Fig. 7a and Fig. 7b), we conclude:

- The estimation error increases with the probability p, regardless of the traffic flow nature of the locations.
- The typical user flow in a specific location also affects the estimation accuracy. In particular, the traffic-fluctuations of a location cause the platform to overestimate the population density, especially in situations where the location has a very small set of real workers. More quantitatively, locations with high occupancy fluctuations bear higher estimation error – 28% more compared to locations with smoother traffic (i.e., the whole campus) when the users prefer stronger privacy protection while querying (e.g., p=0.9).



(a) Food-court (b) Classroom

Fig. 7. Traffic flow of (a) food-court, and (b) classroom on a typical Tuesday in March 2017.

5.3.3 Inference Attack: Though the client is not explicitly revealing its true location to the platform (by querying along with $n-1$ other erroneous locations), a passive adversary can observe subsequent queries and the time gap between them to infer the potential whereabouts of the user. For example, assume the user is querying the platform to know the popularity distribution of locations (l_1, l_2, l_3 and l_4) and t minutes later, he queries again for the locations (l_5, l_6 and l_7). By collating the list of locations that can be reached from l_1–l_4 within t minutes (inter-query time gap) with the locations the user queried (l_5–l_7), the adversary can easily localize the user.

To see the role of parameter p on adversary's ability to localize the user, we plot the fraction of feasible locations (over all locations) versus the number of queries made (see Fig. 8). As expected, fraction of feasible locations are roughly the same as the p values initially, yet as the user issues more queries, the fraction decreases; this is most evident for small p values (e.g., for $p = 0.1$). Though our study is conducted in a smaller urban campus, it's worthwhile to note that the results corroborate well with large scale studies conducted with subway data in [27].

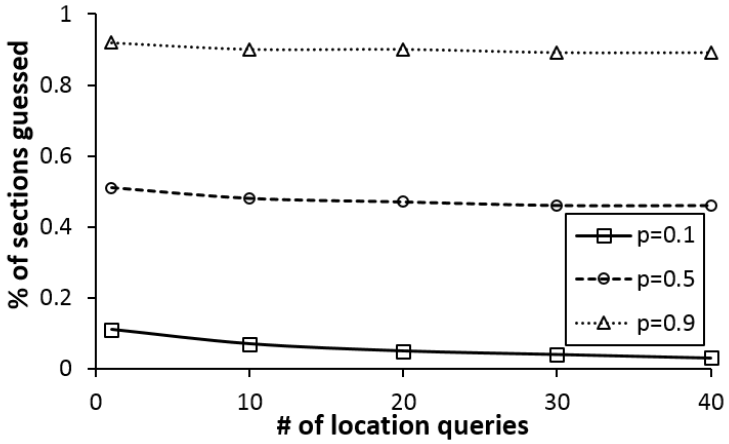


Fig. 8. Knowledge gained by an adversary over consecutive queries.

5.4 Stage-02: User Obfuscation

5.4.1 Entropy gain and q: Once the location is updated by the user, the platform generates trajectories (based on these obfuscated traces) in order to make task recommendations. In order to do so, the platform uses the same technique described in §4.4 (adopted from the state-of-the-art algorithm described in [19]) – first extracts the stay episodes for each time segment and form a transition graph to list all the possible routes. For the analysis, we considered top 5-routes per user – aiming to capture majority of all the possible paths.

To see the impact of the parameter q (of Eq. 2), we first examine how close the obfuscated trajectory of a user is from the true one. In Fig. 9, we plot an aggregate distance metric (averaged over all the users) as a histogram for 3 different q values: (a) 0.1, (b) 0.5, and (c) 0.9. Our motivation is to understand the severity of the location obfuscation needed to achieve desirable privacy guarantees. By considering top-5 trajectories per user, the average distance between true and the corresponding obfuscated trajectory is calculated (initially between a pair of true and obfuscated trajectory and averaged over all 5 pairs) across a single time window. This metric is averaged across multiple time-windows and then across multiple days to obtain a per-user based distance metric across our study period. From Fig. 9, we see that when lower values of q are chosen, the distribution is skewed towards "left", implying that the locations that are closer to the original one was given priority while updating the trajectories –

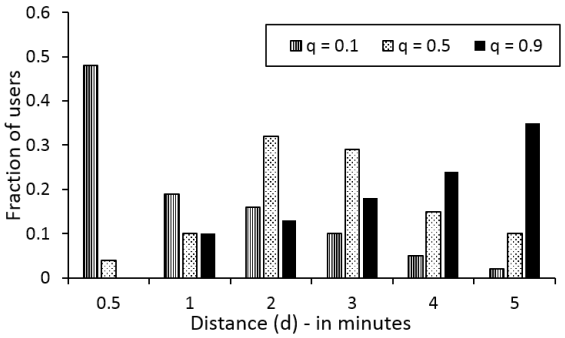


Fig. 9. Histogram of distance values between the true and obfuscated trajectories

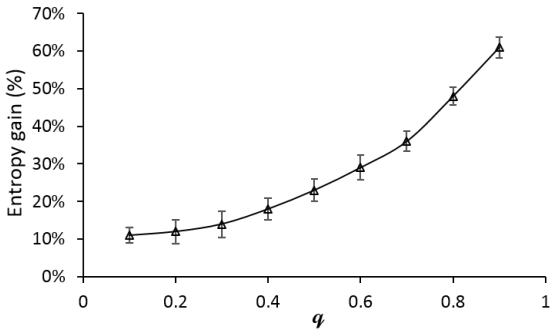


Fig. 10. Impact of q on Entropy Gain

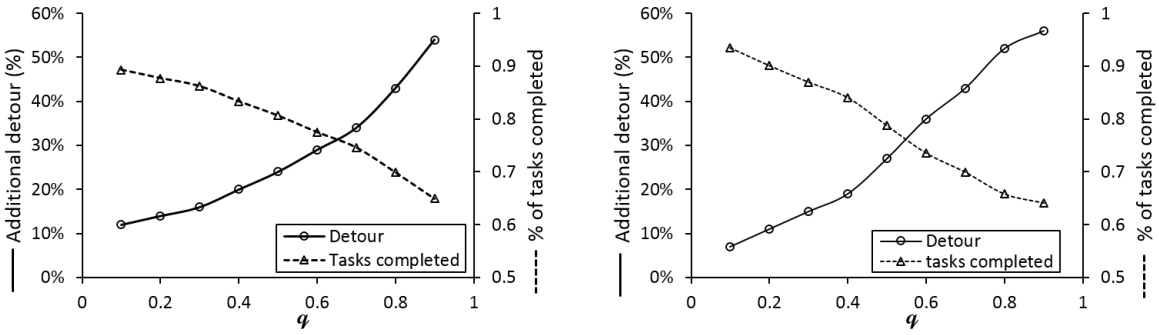
at least 67% of the chosen locations are within the same level of the building (distance $\leq$ 1 minute). On the other hand, higher q values make the skew towards "right" – letting the user to choose and update locations that yields higher privacy gain (entropy) while eliminating the locations that are close-by – 59% of the chosen locations are at least 3-4 levels away from the true location (distance $\geq$ 4 minutes).

Next, to see the achieved privacy guarantees by obfuscation, we use entropy as a measure of privacy. For each user, first we estimate the probability of appearing at location l as an aggregate residency score observed in the history – by averaging the scores at each location (where score of user u is calculated per user, per location basis and given by $score_{u,l}$ = stay time at location l/total stay time across all the locations). This estimation is repeated on both true (T) and obfuscated ($\acute{T}$) trajectories to calculate entropy values of the same (denoted by $H(T) = -\sum_{l \in T} Pr(l) log(Pr(l))$, and $H(\acute{T}) = -\sum_{\acute{l} \in \acute{T}} Pr(\acute{l}) log(Pr(\acute{l}))$. We use the difference between $H(\acute{T})$ and $H(T)$ to determine the entropy gain, $\Delta(H)$: the improvement in unpredictability of trajectories. In Fig. 10, we present the entropy gain (in Y-axis) while choosing different q values (depicted in X-axis). We see that as expected, the gain in entropy increases with q value (lower q values imply that users choose locations that are closer to the original one, while being oblivious to the privacy parameters). Even with the moderate balancing between the privacy and utility (i.e., q=0.5), the user is able to gain 23% in entropy, while not distorting the trajectory greatly – by choosing q=0.5, a user's obfuscated trajectory will be 2 minutes away from the true one (refer Fig. 9). This suggests that even a "neutral" user (one who balances the privacy and quality-of-recommendation objectives) would receive reasonable privacy guarantees, without severely obfuscating the entire trajectory.

5.4.2 Quality of the Recommendations: In Fig. 11a, on the primary Y-axis, we depict the performance loss observed in SMU WiFi dataset – measured by the difference in detour when obfuscated vs. non-obfuscated trajectories were used to generate recommendations, for various q values. We note that, for smaller q values ($\leq$ 0.5), the privacy-imposed obfuscation does not significantly affect the user detour (not more than 24%). This increase is, however, significant at higher values of q. We see that with q = 0.9, a user will incur 54% more detour, on average. Note that, at this point, the system effectively provides 61% more uncertainty (higher entropy) in user trajectories.

In Fig. 11a, on secondary Y-axis, we illustrate the average number of tasks a user can complete with a detour budget of 30 minutes. We plot the percentage of recommended tasks completed when different q values are chosen as compared to the non-private system. In the hypothetical private scenario, we assumed users complete the tasks in the order of recommendation. We see that as q grown larger (or being more privacy-centric), the completion

rate drops. This is because of the fact that larger q values tend to choose the location that yield more privacy guarantees while avoiding the locations that are too close-by. For example, even with more conservative privacy measures (q=0.9), the users are able to complete 65% of the tasks that he would complete normally when the true trajectories were exposed. This means that, the user can achieve greater level of privacy without incurring prohibitive increase in detour or loss in rewards.



(a) SMU WiFi Dataset

(b) Outdoor Dataset

Fig. 11. Impact of q on Detour Incurred and % of Tasks Completed – (a) SMU WiFi and (b) Outdoor dataset

We plot the same measures observed in Outdoor dataset in Fig. 11b. We observed similar trend – as we traverse through q, the additional detour incurred becomes progressively larger, specially for higher values of q. This in turn affects the productivity of the worker – with strict privacy measure of q=0.9, the users are able to complete 64% of the tasks that he would complete normally (when the user reveals his true location).

5.5 Comparison Against Existing Approaches

In this section we provide a detailed comparison of our approach against two alternative, previously-proposed location privacy preserving techniques, namely, $k-$anonymity and Differential privacy. Note that neither k-anonymity nor differential privacy has been explicitly explored for "push-based" mobile crowd-sourcing; accordingly, we adapt the schemes (as explained below) to best fit our operating model. Moreover, both these models assume a *trusted intermediary*, unlike *LORR*, where clients do not reveal their true locations to anybody.

We particularly consider 3 metrics to evaluate the performance tradeoff across multiple privacy techniques: (a) entropy gain/loss– $\Delta(H)$ (private vs. non-private), (b) radius of gyration– r (distance between the true and obfuscated location) and (c) additional detour incurred due to the privacy mechanism. To comprehend the total gain in privacy we consider $\exp(\Delta(H)) * r$ as a combined privacy gain after employed the privacy technique. *This is particularly important in scenarios, where obfuscation is performed by choosing a location of coarser granularity (by combining multiple neighbouring locations) which will eventually lead to loss in entropy (i.e.,* $\Delta(H) \leq 0$*).*

5.5.1 k-Anonymity. k-anonymity is a trajectory anonymization technique that groups k co-localized trajectories (i.e., the trajectories that share the same spatial locations within the same time span) to form a k-anonymized aggregated trajectory. We specifically adopt the clustering-based anonymization approach [1], with the anonymization approach consisting of three phases [8]:

- Pre-processing phase: In this phase, the algorithm first identifies the trajectories that share the same (start, end) timings and groups them together. A crucial step during this phase is to prune the head or tail of the trajectories to make sure that the resulting set of co-localized trajectories can have a common duration (i.e. the same (start, end) times).
- Clustering phase: A greedy approach is then employed to create k-anonymity sets of at least k co-localized trajectories–i.e., given 2 trajectories T_a and T_b sharing the same time span between $[t_s, t_e]$, they are defined as co-localized if the Euclidean distance between each pair of location samples of the trajectories is less than or equal to the threshold d). It does so by first choosing pivot trajectories, one per each group defined in the pre-processing step, as centers of the anonymity clusters. It will keep adding trajectories that are co-localised with the center until the cluster has k members.
- Space Transformation phase: Once all the trajectories are grouped into respective k-anonymity clusters, an aggregate (or representative) trajectory that resides within the cluster's bounding volume is generated and released. This representative trajectory acts as a proxy for the corresponding k trajectories, in subsequent data analysis.

Although k-anonymity's use of a trusted broker implies a fundamental difference with *LORR's* obfuscation-at-client paradigm, we utilize it as a representative approach for location anonymity. We shall show that k-anonymity becomes vulnerable to privacy attacks even in the relatively small geographical space of our university campus, and also that it is unable to consider the individualized location sensitivity (captured by per-worker entropy).

To study clustering-based k-anonymity on our WiFi-based indoor location data, we define two trajectories to be *co-localized* if they share spatial samples within the same building (i.e., d = 5 minutes) over the same interval. To bootstrap the algorithm, we randomly choose a trajectory within each group (that share same start and end timing) and find $k-1$ other co-located trajectories. After the initial phase of clustering, we prune either the head or tail of the currently non-clustered trajectories (pruning is restricted to the initial or final 10% of each trajectory's length), and repeat the process, so as to maximize the number of trajectories that get anonymized. Table 2 illustrates the results for different values of k (varying from $2-5$).

Table 2. k-anonymity

k	Fraction of users clustered
2	74%
3	63%
4	55%
5	42%

Results:

- Based on the experiments conducted using indoor localization data, we consider one more additional metric specially for the case of k-anonymity along with the 3 measures mentioned earlier in this section. We use the fraction of trajectories that could not be clustered – this helps us to understand the uniqueness of the trajectories that caused the extreme case of non-clustering. First, we plot the trade-off between additional detour and weighted privacy gain for various k values in Fig. 12. We observe similar trend in k-anonymity as compared to our approach *LORR* – the additional detour and gain in privacy increases as k becomes progressively larger. However, the gain in privacy is significantly lesser – if a user wants to conceal his location within the distance of 3.5 minutes away from his true location, he incurs 25% of additional detour with *LORR* and 31% with k-anonymity (when k = 4).

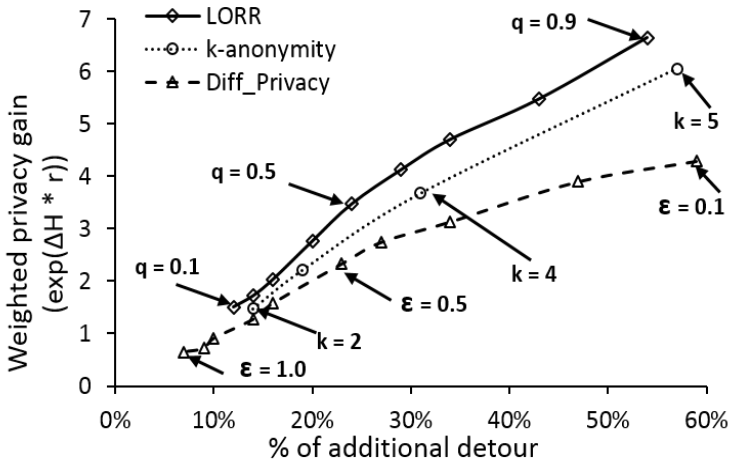


Fig. 12. Comparison of Privacy Techniques: Tradeoff between privacy and application utility

- We also find out that the fraction of trajectories that are non-clustered is considerably high, specially when k grows larger (refer to Table. 2). This higher fraction also raises a question of users being re-identified due to the uniqueness [10] and low-diversity of their trajectories (similar to the diversity issue reported in an urban campus environment in [29]). Even with modestly larger k (=4), 45% of worker routes are not anonymized. However, in the approach we proposed, direct obfuscation of each location sample in client-side reduces the uniqueness of the original trajectory (unique locations or lower density locations are not favoured in the stage 2 of our approach, and replaced by the locations that are less sensitive and highly occupied).

5.5.2 Differential Privacy. Differential privacy [11, 13] is a well explored concept, with both centralised and de-centralised implementations. The mathematical framework associated with this technique helps a user to quantify the *worst-case* privacy loss incurred due to the presence of an individual user's record in a dataset, even when the adversary has access to the auxiliary information about that specific user. In this paper we consider the recent approach presented in [31] where the authors use differential privacy to protect user location data from the crowd-sourcing platform. Their model consists of 3 main entities: (a) users, whose true location details are known to the cell service provider, (b) cell service provider, a trusted broker who collects user location details, and (c) crowd-sourcing platform, who disseminates the task requests to the users without knowing their actual whereabouts. More specifically their approach consists of the following steps:

- Release of private spatial decomposition [9]: In this phase the cell service provider releases the spatial dataset in a differentially private manner to the crowd-sourcing platform, by adding carefully calibrated noise (drawn from a Laplace or Gaussian distribution, such that the noise satisfies the differential privacy constraint) to the true count of users available in a particular region/grid. It partitions the spatial map adaptively (based on the location density) into 2 levels and add noise to the count of workers in each level-2 cell.
- Estimate the Geocast region: When the crowd-sourcing platform receives a new task request, it uses the released (noisy) spatial map to estimate the geocast region, over which the task request is to be disseminated to the available users. The proposed approach in [31] uses a greedy selection algorithm, initially choosing

the grid that contains the task location and progressively expanding by adding neighbouring grids that yield the maximum increase in utility. The utility is measured by the probability that a user in the chosen geocast region accepts the disseminated task (it can be modeled as a Binomial distribution).
- Dissemination of the task: The service provider then disseminates the task to all the users in the geocast region chosen in the above step.

Results:

- In Fig. 12, we plot the privacy – utility tradeoff of differential privacy technique for various ϵ values (ranging from 0.1 to 1.0). We see that *LORR* enables better tradeoff than the differential privacy technique – to achieve same weighted privacy gain, differential privacy incurs 30% more additional detour as compared to *LORR* . This is due to the fact that, though, differential privacy offers lower detour (by greedily starting from the task location and expand the geocast region by adding neighbouring nodes), the perceived change in entropy is negative – when the geocast region is growing larger, a user will lose his entropy (as he may have more than one residency episode in the geocast region). Hence the resultant weighted privacy gain is comparatively lesser.
- Further this technique also imposes the risk of (a) service provider colluding with the crowd-sourcing platform and exposes the true trajectories, and (b) ability to estimate the users' true location from the task reports the platform received.

Key Takeaways: The comparative results show two key capabilities of *LORR*:

- *Personalization:* Unlike k-anonymity and differential privacy approaches, which focus on only aggregate counts/statistics, *LORR* is able to personalize the obfuscation steps. In particular, *LORR* allows each client to choose an obfuscated location, based on both the global occupancy counts and its *own sensitivity to revealing this location.* As a result, the average entropy gain in *LORR* is demonstrably higher than the alternative approaches.
- *Robust Tradeoffs: LORR* enables a better tradeoff between privacy gain and loss of crowd-sourcing productivity (as a result of higher detours) than the other approaches. More specifically, in our datasets, *LORR* provides an almost-linear tradeoff between these two competing metrics. In contrast, the k-anonymity approach of enabling this tradeoff (by varying k) also significantly increases the fraction of non-anonymized workers (as k increases), whereas the differential privacy tradeoff parameter (ϵ) generates a tradeoff frontier that is strictly worse (to the right of) *LORR's* tradeoff curve.

6 DISCUSSION

While our results demonstrate the promise of the *LORR* framework, there are several future directions of investigation.

6.1 Randomized Response and Differential Privacy

It is well studied that randomized response can be represented as (ϵ, δ)–differential privacy, providing stronger mathematical validation and quantification of privacy loss to the probabilistic randomized response approach [13]. However, we did not choose ϵ to quantify the privacy loss for the following reasons: (a) even for relatively smaller values of coin flipping probability p the equivalent ϵ will be considerably larger – for example, when p = 0.1, the resultant ϵ will be more than 2, and (b) since each coin flip per user is a randomized responses technique, when the user attempts to query to obfuscate single location (by doing n flips) his additive ϵ will be significantly larger (note that preferred ϵ value is <1).

6.2 Heterogeneous Location Obfuscation

In this paper, we consider homogeneous obfuscation by the clients in the system: in our trace-driven studies, we assume all the clients use the same parameter q during the second-stage of the user-controlled obfuscation strategy. However, in reality, user perceptions may vary and different users may prefer different values of q. The resulting dynamics needs to be studied carefully to derive additional insights. For example, it is unclear how the performance of the proposed approach will be affected when various proportions of users adopt different gain co-efficient (q).

More interestingly, this heterogeneous nature may be perceived as the prisoner's dilemma, specially when 2 highly-correlated users (in terms of mobility patterns) obfuscate with different q values. It is possible that users choices of q may be modeled as non-cooperative games.

6.3 Adaptive Privacy Control

The novel user-controlled data obfuscation approach we introduced in this paper allows the user to tweak the gain coefficient longitudinally to toggle between the states of being privacy conscious and quality oriented. For example, if a user finds out he's revealing more sensitive details to the platform (e.g., if he observes that he is receiving a large number of recommendations closer to his dominant locations), he can now modify the parameter q (to be higher) to increase the level of obfuscation. This feature is not studied in this paper and allocated for future work.

6.4 Pricing Effects

The obfuscation of individual trajectories not only affects the estimation of occupancy levels of different locations, but can also implicitly affect the prices (rewards) associated with different tasks. Many practical crowd-sourcing platforms assign higher rewards to tasks located at less popular locations. However, current popularity-sensitive models for setting task rewards may not be effective when the underlying movement traces (of each individual worker, as well as at an aggregate level) do not represent the true movement behavior. It is possible that the added amount of privacy also results in an additional cost (in terms of reward prices) on the overall platform. Further studies are needed to quantify this possibility.

7 CONCLUSIONS AND FUTURE WORK

Recent empirical studies have shown the advantages of *push*-based mobile crowd-sourcing, where users receive recommendations that are tailored based on their predicted movement behaviors. However, it comes with various privacy threats imposed to the users: when the raw traces are exposed to the system, it can be exploited for tracking and stalking purposes. In this paper, we propose a novel, two-stage obfuscation mechanism for push-based mobile crowd-sourcing platforms while allowing the user client to balance the tradeoff between privacy and recommendation accuracy. During the stage one, the client allows the platform to estimate accurate measure of popularity scores by claiming that he is present at n locations (one of those is the true location) chosen based on randomized response mechanism. In stage two, we provide a parameter for user clients to control the privacy loss, by obfuscating locations while maximizing the its perceived benefits. We empirically validate our proposed approach by using *TA$Ker* :

- By utilizing the data collected from the *TA$Ker* platform, we first exhibit that even task completion locations provide a sufficiently strong prediction on user's dwell time and location. This illustrates the needs for location obfuscation in *push*-based mobile crowd-sourcing.

- We show how a user can effectively control the parameter q to achieve desired privacy and/or application utility guarantees. More quantitatively, we show that with same preference for both privacy gain and quality of recommendation objectives, we can increase the location entropy of crowd-workers by 23% while imposing an additional 24% detour overhead.
- We further explore how the overall system performance is affected by different parameters, such as the probability p, intrinsic properties of locations and obfuscation coefficient q.

ACKNOWLEDGMENTS

This material is supported partially by the National Research Foundation, Prime Minister's Office, Singapore under its International Research Centers in Singapore Funding Initiative, and partially by the U.S. Army International Technology Center Pacific (ITC-PAC) under Contract No. FA5209-17-C-0006.

REFERENCES

[1] O. Abul, F. Bonchi, and M. Nanni. 2008. Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. In *IEEE International Conference on Data Engineering (ICDE)*.
[2] Alastair R. Beresford and Frank Stajano. 2003. Location Privacy in Pervasive Computing. *Pervasive Computing* (2003).
[3] Ioannis Boutsis and Vana Kalogeraki. 2013. Privacy Preservation for Paricipatory Sensing Data. In *IEEE International Conference on Pervasive Computing and Communications (PerCom)*.
[4] Ioannis Boutsis and Vana Kalogeraki. 2016. Location Privacy for Crowdsourcing Applications. In *ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*.
[5] Cen Chen, Shih-Fen Cheng, Aldy Gunawan, Archan Misra, Koustuv Dasgupta, and Deepthi Chander. 2014. TRACCS: Trajectory-Aware Coordinated Urban Crowd-Sourcing. In *2nd AAAI Conference on Human Computation and Crowdsourcing*. 30–40.
[6] Cen Chen, Shih-Fen Cheng, Hoong Chuin Lau, and Archan Misra. 2015. Towards city-scale mobile crowdsourcing: Task recommendations under trajectory uncertainties. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
[7] Shih-Fen Cheng, Cen Chen, Thivya Kandappu, Hoong Chuin Lau, Archan Misra, Nikita Jaiman, Randy Tandriyansiyah, and Desmond Koh. 2017. Scalable urban mobile crowdsourcing: Handling uncertainty in worker movement. *ACM Transactions on Intelligent Systems and Technology* (2017), to appear.
[8] C.Y Chow and M.F. Mokbel. 2011. Trajectory privacy in location-based services and data publication. *ACM SIGKDD Explorations Newsletter* 13, 1 (2011).
[9] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu. 2012. Differentially Private Spatial Decompositions. In *IEEE International Conference on Data Engineering (ICDE)*.
[10] Yves-Alexandre de Montjoye, Cesar A Hidalgo, Michael Verleysen, and Vincent D Blondel. 2013. Unique in the Crowd: The privacy bounds of human mobility. *Scientific Reports* 1376 (2013).
[11] Cynthia Dwork. 2006. Differential Privacy. In *International Colloquium on Automata, Languages and Programming*.
[12] Cynthia Dwork. 2008. Differential Pricacy: A Survey of Results. In *Conference on Theory and Applications of Models of Computation*.
[13] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2008. Calibrating Noise to Sensitivity in Private Data Analysis. In *Conference on Theory and Applications of Models of Computation*.
[14] Srivatsava Ranjit Ganta, Shiva Kasiviswanathan, and Adam Smith. 2008. Composition Attacks and Auxiliary Information in Data Privacy. In *Knowledge Discovery and Data Mining*.
[15] M. Gruteser and D. Grunwald. 2003. Anonymous Usage of Location Based Services Through Spatial and Temporal Cloaking.. In *The First International Conference on Mobile Systems, Applications, and Services*.
[16] Takamasa Higuchi, Paul Martin, Supriyo Chakraborty, and Mani Srivastava. 2015. AnonyCast: Privacy Preserving Location Distribution for Anonymous Crowd Tracking Systems. In *ACM International Joint Conference on Pervasive and Ubiquitous Computing*.
[17] K. L. Huang, S. S. Kanhere, and W. Hu. 2009. Towards Privacy-sensitive Particpatory Sensing.. In *Pervasive Computing and Communications*.
[18] Thivya Kandappu, Nikita Jaiman, Randy Tandriyansiyah, Archan Misra, Shih-Fen Cheng, Cen Chen, Hoong Chuin Lau, Deepthi Chander, and Koustav Dasgupta. 2016. TASKer: behavioral insights via campus-based experimental mobile crowd-sourcing. In *2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*.
[19] Thivya Kandappu, Archan Misra, Shih-Fen Cheng, Nikita Jaiman, Randy Tandriyansiyah, Cen Chen, Hoong Chuin Lau, Deepthi Chander, and Koustav Dasgupta. 2016. Campus-Scale Mobile Crowd-Tasking Deployment & Behavioral Insights. In *The 19th ACM Conference on Computer-Supported Cooperative Work and Social Computing*.

[20] Leyla Kazemi and Cyrus Shahabi. 2011. A privacy-aware framework for participatory sensing. *ACM SIGKDD Explorations Newsletter* 13, 1 (2011), 43–51.
[21] Azeem J. Khan, Vikash Ranjan, Trung-Tuan Luong, Rajesh Krishna Balan, and Archan Misra. 2013. Experiences with performance tradeoffs in practical, continuous indoor localization.. In *IEEE WOWMOM*. 1–9.
[22] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. 2007. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *International Conference on Data Engineering*.
[23] Ashwin Machanavajjhala, Johns Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. 2006. l-Diversity: Privacy Beyond k-Anonymity. In *International Conference on Data Engineering*.
[24] Archan Misra and Rajesh Krishna Balan. 2013. LiveLabs: Initial Reflections on Building a Large-scale Mobile Behavioral Experimentation Testbed. *SIGMOBILE Mobile Computing and Communications Review* 17, 4 (2013), 47–59.
[25] M. F. Mokbel, C. Y. Chow, and W. G. Aref. 2006. The New Casper: Query Processing for Location Services without Compromising Privacy.. In *The First International Conference on Mobile Systems, Applications, and Services*.
[26] Ben Niu, Qinghua Li, Xiaoyan Zhu, Guohong Cao, and Hui Li. 2014. Achieving k-anonymity in Privacy-Aware Location Based Services. In *IEEE INFOCOM*.
[27] Daniele Quercia, Ilias Leontiadis, Liam McNamara, Cecilia Mascolo, and Jon Crowcroft. 2011. SpotME If You Can: Randomized Responses for Location Obfuscation on Mobile Phones. In *IEEE 31st International Conference on Distributed Computing Systems*.
[28] Pierangela Samarati and Latanya Sweeney. 1998. Generalizing Data to Provide Anonymity when Disclosing Information. In *Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*.
[29] Kaixin Sui, Youjian Zhao, Dapeng Liu, Minghua Ma, Lei Xu, Li Zimu, and Dan Pei. 2016. Your trajectory privacy can be breached even if you walk in groups. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*.
[30] Hien To, Ghinita Gabriel, and Cyrus Shahabi. 2015. PrivGeoCrowd: A Toolbox for Studying Private Spatial Crowdsourcing. In *IEEE International Conference on Data Engineering*.
[31] Hien To, Gabriel Ghinita, Liyue Fan, and Cyrus Shahabi. 2016. Differentially Private Location Protection for Worker Datasets in Spatial Crowdsourcing. *IEEE Transactions on Mobile Computing* (2016), 1–14.
[32] Hien To, Gabriel Ghinita, and Cyrus Shahabi. 2014. A framework for protecting worker location privacy in spatial crowdsourcing. *Very Large Data Base Endowment* 7, 10 (2014), 919–930.
[33] Leye Wang, Daqing Zhang, Dingqi Yang, Brian Lim, and Xiaojuan Ma. 2016. Differential Location Privacy for Sparse Mobile Crowdsensing. In *IEEE International Conference on Data Mining*.
[34] S. L. Warner. 1965. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *J. Amer. Statist. Assoc.* (1965), 63–69.

Received May 2017; Revised November 2017; Accepted January 2018