IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

Received 20 March 2015; revised 4 July 2015; accepted 2 October 2015. Date of publication 12 October 2015;
date of current version 2 March 2018.

Digital Object Identifier 10.1109/TETC.2015.2490021

Protecting Location Privacy for Task Allocation in Ad Hoc Mobile Cloud Computing

**YANMIN GONG[1], (Student Member, IEEE), CHI ZHANG[2], (Member, IEEE),
YUGUANG FANG[1], (Fellow, IEEE), AND JINYUAN SUN[3], (Member, IEEE)**

[1]Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611 USA
[2]School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China
[3]Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN 37996 USA

CORRESPONDING AUTHOR: Y. GONG (ymgong@ufl.edu)

This work was partially supported by the U.S. National Science
Foundation under Grant CNS-1409797 and Grant CNS-1423165. The work of C. Zhang was supported in part by the Natural Science Foundation of China under Grant 61202140 and Grant 61328208, in part by the Program for New Century Excellent Talents in University under Grant NCET-13-0548, in part by the Innovation Foundation of the Chinese Academy of Sciences under Grant CXJJ-14-S132, and in part by the Fundamental Research Funds for the Central Universities under Grant WK2101020006.

ABSTRACT Mobile cloud computing (MCC) is an emerging cloud-computing paradigm that integrates cloud computing and mobile computing to enable many useful mobile applications. However, the large-scale deployment of MCC is hindered by the concerns on possible privacy leakage. In this paper, we investigate the privacy issues in the ad hoc MCC, and propose a framework that can protect the location privacy when allocating tasks to mobile devices. Our mechanism is based on differential privacy and geocast, and allows mobile devices to contribute their resources to the ad hoc mobile cloud without leaking their location information. We develop analytical models and task allocation strategies that balance privacy, utility, and system overhead in an ad hoc mobile cloud. We also conduct extensive experiments based on real-world data sets, and the results show that our framework can protect location privacy for mobile devices while providing effective services with low system overhead.

INDEX TERMS Mobile cloud computing, location privacy, task allocation, reputation.

I. INTRODUCTION

Nowadays, mobile devices such as smartphones and tablets have gained tremendous popularity. These devices are often equipped with a variety of sensors such as camera, microphone, GPS, accelerometer, gyroscope, and compass. The data (e.g., position, speed, temperature, and heart rate) generated by these sensors enable many useful mobile applications, including location-based services [1], [2], mobile sensing [3], and mobile crowdsourcing [4], [5]. Although improved largely over the past several years, mobile devices are still resource-constrained mainly due to the limited battery lifetime. On the other hand, cloud computing has widely been regarded as the next-generation computing paradigm which provides ‘‘unlimited’’ cloud resources to end-users in an on-demand fashion. The rich cloud resources in cloud computing can be exploited to increase, enhance, and optimize capabilities of mobile devices, leading to the concept of mobile cloud computing (MCC). According to [6], MCC integrates cloud computing technologies with mobile devices to make the mobile devices more capable in terms of computational power, memory, storage, energy, and context awareness.

There are generally two types of mobile clouds in MCC: infrastructure-based and ad hoc [6]. The infrastructure-based mobile cloud consists of stationary computing resources and provides services to the mobile users via the Internet. Alternatively, in the ad hoc mobile cloud, a collection of mobile devices (hereafter referred to as ‘‘mobile servers’’) performs as cloud resources and provides access to local or Internet-based cloud services to other mobile users (hereafter referred to as ‘‘mobile clients’’). In this paper, we focus on the second case, namely, the ad hoc mobile cloud. The main benefit of utilizing ad hoc mobile cloud resources is their distributed and context-awareness features. As explained in [7]–[10], incentivized by the *mobile cloud computing platform* (CCP), individual mobile users contribute their mobile devices as mobile servers in the ad hoc mobile cloud, and these mobile servers can be used to perform location-dependent tasks such as epidemic monitoring, traffic monitoring, image/video capturing, and price checking for mobile clients.

Despite many promising applications, ad hoc mobile clouds pose several challenges. First, mobile cloud resources in an ad hoc mobile cloud are dynamic and diverse. As a result, some mobile servers may drop the task they are performing and leave the cloud. Some mobile servers may be ''spammers'' that only want to collect rewards and submit arbitrary answers without looking at the specific task. Moreover, some mobile servers may not be powerful enough to provide sensing data at the required accuracy. Therefore, how to allocate tasks to ensure the quality of the service provided by these dynamic mobile servers is challenging. Second, as pointed out by [7], security and privacy of mobile devices as service providers is a critical concern in the ad hoc mobile cloud. In order to allocate tasks and provide effective services, mobile servers in an ad hoc mobile cloud need to share their location data with the CCP, which could reveal a lot of personal information such as a user's identity, health status, personal activities, and political views [11]. Hence, it is mandatory to provide privacy guarantee in order to engage more mobile devices in the cloud. Finally, there is an inherent conflict between quality of service (i.e., utility) and privacy in task allocation. If an ad hoc mobile cloud ensures privacy of mobile servers, it is difficult to guarantee the utility of their MCC service. Finding a solution that ensures privacy while guaranteeing utility for task allocation is a major challenge in such systems.

Several solutions to privacy issues in mobile applications have been proposed. For example, aggregation is a common approach to hiding individual sensitive information when only statistics of users are required [12]. However, this approach only calculates statistics and thus cannot be used to select mobile servers in an ad hoc mobile cloud. Another approach is used in location-based services, where accurate locations are obfuscated in location-based queries, and the service provider returns results based on the obfuscated query [13], [14]. In our scenario, however, the private information is no longer part of a location-based query, but the result of a location-based query regarding the task. Some papers [15], [16] consider queries on private locations in an outsourced database, but they only protect private data from an intermediate service provider while assuming a trust relationship between the data owner and the querying entity. This is not true in our scenario because mobile servers and the CCP may not share an inherent trust relationship. A recent work by To *et al.* [17] has been proposed to protect location privacy of crowdsourcing workers in spatial crowdsourcing. However, their solution does not consider worker reputation, and thus cannot provide any quality control over the final result. Therefore, it can not be easily applied to the mobile cloud computing scenario where service quality is very important.

In this paper, we propose a framework that provides solutions to the above challenges, where both location privacy and service quality are considered. In our framework, the CCP only has access to sanitized location data of mobile servers according to *differential privacy (DP)*. Since every mobile server is subscribed to a *cellular service provider* (CSP) with which it already has a trust relationship, the CSP can integrate mobile server location and reputation information, and provides the data to the CCP in noisy form according to DP. To generate the noisy mobile server data, we adapt the *Private Spatial Decomposition (PSD)* approach proposed in [17] and [18], and construct a new structure called *Reputation-based PSD (R-PSD)*. Since fake points need to be created in the DP model, geocast is used to disseminate tasks to mobile servers to prevent the CCP from identifying these points.

To summarize, our main contributions are as follows:

1) We identify the specific challenges for task allocation in ad hoc mobile clouds, and propose a framework that can achieve differential privacy for mobile server location data while providing high service quality.
2) We introduce a new structure called R-PSD that partitions the space based on both reputation and location information, and develop an efficient search strategy that finds appropriate R-PSD partitions to ensure high quality of service.
3) We use a geocast mechanism when disseminating tasks to mobile servers to overcome the restrictions imposed by DP, and the overhead during this process is incorporated into the design of the search strategy.
4) We conduct extensive experiments based on real-world datasets to show the effectiveness of the proposed framework.

The remainder of this paper is organized as follows. We present background on several techniques we use in Section II. In Section III, we describe the system model for the proposed framework. Section IV and Section V describe the detailed solutions, i.e., R-PSD generation and task allocation based on R-PSD. Thereafter, we discuss the experimental results and evaluate the system overhead in Section VII. Section VIII reviews the related work and Section IX concludes the paper.

II. BACKGROUND

In this section, we introduce background on differential privacy (DP) and Private Spatial Decomposition (PSD).

A. DIFFERENTIAL PRIVACY

The privacy guarantee provided in our framework is ϵ-differential privacy [19], [20]. DP provides protection of datasets against adversaries with arbitrary background information. By sanitizing the data, DP prevents an adversary from knowing whether a certain individual record is present or not in the database. Formally speaking, we have the following formal definition.

Definition 1: A randomized algorithm $\mathcal{F}$ satisfies ϵ-DP if for any two datasets D_1 and D_2 which differ in only one element, and $\forall O \subseteq \text{range}(\mathcal{F})$, the following inequality holds:

$$\ln \frac{\Pr[\mathcal{F}(D_1) \in O]}{\Pr[\mathcal{F}(D_2) \in O]} \leq \epsilon. \tag{1}$$

In the definition, the parameter ϵ bounds the ratio of probability distributions of two datasets differing on at most one element. It specifies the amount of privacy protection, and a smaller value of ϵ indicates better protection. We call this parameter the *privacy budget*.

In order to achieve ϵ-DP in a dataset, the raw data is sanitized by adding random noise to the released query set QS. The amount of noise is determined by the sensitivity of QS, which is defined as follows:

Definition 2: Given any two datasets D_1 and D_2 which differ in one element, the sensitivity of the released query set QS is

$$\sigma(QS) = \max_{D_1,D_2} \left\| QS_{D_1} - QS_{D_2} \right\|_1 . \tag{2}$$

Given the sensitivity, a sufficient condition to achieve ϵ-DP is to add to each query result randomly distributed Laplace noise with mean $\lambda = \sigma(QS)/\epsilon$ [21].

The results from a database usually involve several stages of analyses M_i. The privacy level of the composition of several stages can be computed by the following results [22]:

Theorem 1 (Sequential Composition): If M_i are a set of analyses, each providing ϵ_i-DP, then their sequential composition satisfies $\left(\sum_i \epsilon_i\right)$-DP.

Theorem 2 (Parallel Composition): If M_i are a set of analyses, each providing ϵ_i-DP, then their parallel composition satisfies $\max_i (\epsilon_i)$-DP.

These theorems enable us to calculate privacy level of an aggregated result based on the privacy level of each individual result.

B. PRIVATE SPATIAL DECOMPOSITION (PSD)

The *Private Spatial Decomposition (PSD)* approach is first introduced in [18] to construct a spatial dataset that achieves DP. A PSD is a spatial index where each index node is associated with a spatial region, and the value for each node is the noisy count of data points (mobile servers in our scenario) in that region. The data structure for spatial index can be grids, k-d trees, or quad-trees [23].

Choice of data structure and its parameters (fan-out and height) can heavily influence the accuracy of PSD. In space-based partitioning PSD such as grids and quad trees, the splitting positions of space is independent of MS locations. Thus privacy budget is only consumed when calculating the noisy count of mobile servers. Typically, index nodes at the same level cover non-overlapping extents, resulting in a low sensitivity of 2 (i.e., the location change of a single MS affects at most 2 cells in a level). The privacy budget ϵ is distributed across levels according to geometric allocation strategy in [18], where leaf nodes are allocated more budget than higher level nodes. Space-based PSD are easy to construct, but they can become unbalanced when mobile servers are not uniformly distributed in space.

On the other hand, object-based structures such as k-d trees [18] split space based on the locations of mobile servers. Since location data are used both for calculating splitting positions and computing noisy counts, the privacy budget should be split between the two processes as well. Object-based structures are expected to be more balanced than space-based PSD; however, they are not very robust in the sense that their accuracy may decrease abruptly with a slight change of the PSD parameters or input dataset distributions.

The work in [24] proposes an adaptive grid (AG) approach with two-level grids. The first-level grid is uniformly divided, and the granularity of the second-level grid depends on the noisy counts obtained in the first-level. AG is a hybrid approach that inherits the simplicity and robustness of space-based approach, but still utilizes some data-dependent information when choosing the granularity for the second-level grid. In this paper, we adapt their approach to construct our PSD.

III. PROBLEM FORMULATION

There are various types of MCC applications with varying system models. In this paper, we consider an emerging MCC system model favored by several recent studies [9], [10], [25]. A unique feature of this system model is that various mobile devices such as smartphones, tablets, and handheld computing devices play the role of servers based on cloud computing principles. Benefits of this system include the proximity of the mobile resources to mobile clients and context-awareness of the mobile resources [7]. In the following, we present the system model in Section III-A, describe the mobile server characteristics in Section III-B, outline our threat model and assumptions in Section III-C, and discuss our performance metrics in Section III-D.

A. SYSTEM MODEL

The system model for our proposed privacy-preserving framework is shown in Fig. 1. There are mainly four parties in the system: the CCP, the CSP, *mobile servers*, and *mobile clients*. The CCP has a pool of mobile servers as its cloud resources, and assigns a reputation score to each of these mobile servers based on its historical task completion performance. Mobile clients send requests to the CCP to utilize

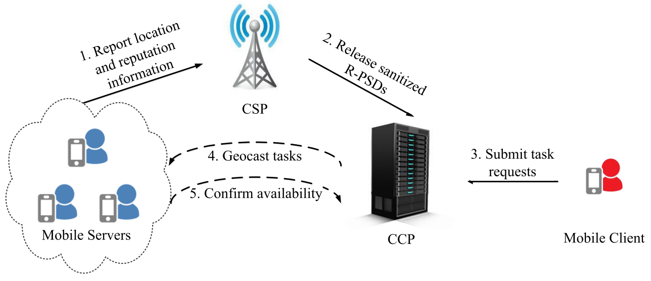


FIGURE 1. **Privacy-preserving framework for task allocation in MCC.**

its cloud resources for completing various tasks. The CSP is a trusted third-party that guarantees privacy of mobile servers while enabling efficient MCC services.

Our system works as follows. First of all, mobile servers send their locations extracted from their GPS sensors, and reputation scores learned from the CCP to the CSP (Step 1) who then collects updates and releases a *Reputation-based Private Spatial Decomposition (R-PSD)* according to the privacy budget ϵ agreed upon with mobile servers (Step 2). When the CCP receives a task request from mobile clients (Step 3), it uses the R-PSD to decide a geocast region that contains mobile servers in close proximity to the task and with high reputation level. Then, the CCP initiates a geocast communication process [26] to all mobile servers within the geocast region (Step 4). Geocast is used in our framework to keep the CCP from directly contacting mobile servers. Note that when sanitizing a dataset based on DP, it is required to create some fake locations in the R-PSD. If allowed to contact mobile servers directly, the CCP can easily identify these fake points, and therefore breach privacy whenever it fails to establish a communication channel with some mobile servers. After receiving the task, a mobile server decides whether to accept the task or not. If the mobile server decides to accept the task, it replies with a message confirming its availability to the CCP (Step 5). Otherwise, it does not reply and remains invisible to the CCP.

In such an MCC system, we focus on protecting location privacy of mobile servers. The most challenging issues in the proposed framework are the following:

1) How to incorporate the reputation into the traditional PSD to construct the R-PSD?
2) How to choose an appropriate geocast region in order to ensure high service quality given the uncertain nature of the R-PSD?
3) How to disseminate tasks to mobile servers in the chosen geocast region with low overhead?

As shown later, we propose novel and efficient approaches to address all of these challenges.

B. TASK AND MOBILE SERVER CHARACTERISTICS

Tasks considered in the system are location-dependent, i.e., they must be performed at specific locations. Typical examples include sensing tasks and those in location-based services. In many cases, the mobile server needs to travel physically to the location associated with the task. Therefore, most mobile servers that perform a task will be located in close proximity to the task location. Furthermore, it is not uncommon that some tasks need to be performed by more than one mobile server.

Mobile servers are mobile devices that have diverse computing, communication, and sensing capabilities. They could be either voluntary resources that are donated by mobile users [27] similar to that in participating sensing, or recruited by the CCP using some incentives similar to that in crowdsourcing [28]. Due to their heterogeneity, mobile servers may have different performances when completing a task. We use reputation as a means to evaluate the quality of results received from mobile servers. It reflects the trustworthiness of the returned results. Here, we assume that the CCP has already implemented a reputation system that assigns reputation scores to each mobile server. Several reputation systems [29]–[32] have been proposed in the literature, and they can be used in our system. Similar to [31], the reputation score of a mobile server is a number in the range of 0 and 1. A mobile server with a high reputation indicates that it has been providing reliable results for past tasks. Note that the reputation of a mobile server is updated periodically by the CCP based its past performance.

C. THREAT MODEL AND ASSUMPTIONS

We aim to protect the location privacy of mobile servers *before they accept a task*. Note that once a mobile server accepts a task, it may by itself reveal its information to the CCP or the client who requests the task. However, information disclosure at this stage is out of our scope. We only focus on privacy leakage before the mobile server accepts a task due to two reasons. First, before accepting a task, all mobile servers are candidates for real-time task allocation, and thus their location information is monitored continuously. The volume and timescale of information exposure make privacy protection mechanisms a necessity. On the other hand, only a few mobile servers will accept the task and expose their information at later stage. Second, a mobile server explicitly consents to reveal its location information after accepting a task, which is unavoidable in our scenario. The influence of such leakage can be mitigated by hiding its identity, which is done in Tor [33]. On the other hand, few papers consider the more challenging problem of preserving privacy at early stage.

We consider the CCP not trustworthy when protecting servers' privacy. Any information learned by the CCP may be leaked to adversaries when it is compromised. On the other hand, the CSP has already established a trust relationship with mobile servers through cellular service contracts and mutually agreed rules regarding information disclosure. Moreover, the CSP already has access to the locations of mobile servers through localization techniques such as cell tower triangulation. Hence, reporting mobile server locations discloses no additional information. Although the CSP is trusted to learn mobile server locations, we could not rely on it when allocating tasks among mobile servers. The reason is obvious: task allocation involves multiple issues such as profile management and mobile client categorization, and the CSP has no expertise or incentive to get involved with such services. Therefore, in our framework, the CSP is considered as a trusted third-party that helps protecting privacy of mobile servers.

We also assume secure, reliable, and authenticated communication channels among the CCP, the CSP, and mobile servers.

D. PERFORMANCE METRICS

This section presents a task allocation model that effectively allocates tasks among mobile servers in the MCC system while providing differential location privacy for mobile servers. Adding privacy protection to task allocation greatly complicates the problem since the CCP can no longer allocate a task among mobile servers based on their exact locations. Due to the uncertain nature of DP, it is possible that there is no mobile server in a geocast region, even if the noisy count shows positive. Thus the task may not be completed as no, or an insufficient number of mobile servers are actually notified. Also, if the task is allocated to mostly mobile servers with low reputation scores, the result may not satisfy the quality-of-service requirement for the task. Finally, the CCP may need to contact more mobile servers than what is needed in the privacy-oblivious case, increasing system overhead.

Therefore, we focus on the following performance metrics for the proposed framework:

- *Acceptance Rate.* Due to the uncertainty of R-PSD, the CCP may fail to find enough mobile servers for a task. The acceptance rate of task allocation captures the ratio of accepted tasks to all task requests.
- *Service Quality.* Even if a task is accepted by some mobile servers based on their location information, they may not fulfill the task successfully to meet the quality-of-service requirement for the task. High service quality could be achieved by selecting only mobile servers with high reputation score; however, this may result in low acceptance rate. Therefore, one needs to balance service quality and acceptance rate when choosing the geocast region.
- *System Overhead.* Additional system overhead is incurred due to privacy protection, as more mobile servers would be contacted when the CCP does not know their precise locations. We use the average number of notified mobile servers to quantify the communication overhead of notifying mobile servers in the task dissemination process and the computational overhead of geocast region determination.

IV. CONSTRUCTING THE R-PSD

In this section, we solve the first challenging issue, that is, how to incorporate the reputation to build the R-PSD. We first describe the method of building a PSD without considering any reputation information and then, add reputation levels into the previous method to construct an R-PSD.

A. CONSTRUCTING PRIVATE SPATIAL DECOMPOSITIONS

To build a PSD based on the locations of mobile servers, we follow the state-of-the-art Adaptive Grid (AG) approach proposed in [17] and [24] due to the advantages mentioned in Section II-B. The AG approach overlays a two-level grid onto a region. At the first level, the location domain is uniformly partitioned into $m_1 \times m_1$ cells. To minimize errors due to DP partition, the level-1 granularity m_1 is chosen as

$$m_1 = \max\left(10, \left\lceil \frac{1}{4}\sqrt{\frac{N \times \epsilon}{k_1}} \right\rceil\right), \tag{3}$$

where N is the total number of mobile server locations, ϵ is the total privacy budget, and k_1 is a small constant depending on the datasets. This heuristic method is data-independent, and thus does not consume any privacy budget. Extensive experiments in [24] suggest that $k_1 = 10$ works well for different sizes of datasets and different privacy budget ϵ.

After the level-1 grid is determined, the CSP issues a noisy count query for each level-1 cell using a small portion of the privacy budget: $\epsilon_1 = \alpha \times \epsilon$, where parameter $\alpha \in (0, 1)$ decides how the privacy budget is divided between two levels. Then the CSP further partitions each level-1 cell into $m_2 \times m_2$ level-2 cells, where m_2 is chosen as:

$$m_2 = \left\lceil \sqrt{\frac{N' \times \epsilon_2}{k_2}} \right\rceil, \tag{4}$$

where $\epsilon_2 = \epsilon - \epsilon_1$ is the remaining privacy budget for noisy count in each level-2 cell, N' is the noisy count of mobile server locations in the level-1 cell, and k_2 is a constant depending on datasets. In [17], To et al. argue that the PSD count should be small to provide fine granularity, but should be larger than the standard deviation of added noise. Here we set $k_2 = \sqrt{2}$, which provides a good balance between these two considerations according to [17].

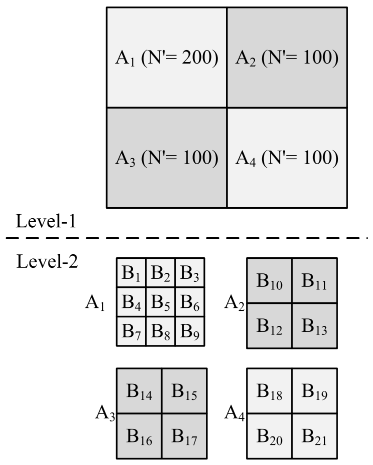


FIGURE 2. **Example of an adaptive grid.**

An example of an adaptive grid is illustrated in Fig. 2. There are 4 level-1 cells, A_1, A_2, A_3, A_4, in the level-1 grid, which is determined by (3). The noisy count of each level-1 cell is calculated by adding random Laplace noise with mean $1/\epsilon_1$ to the actual count. These level-1 cells are further divided into level-2 cells based on the noisy count of mobile servers in each cell. The number of level-2 cells in each level-1 cell is $m_2 \times m_2$, where m_2 is calculated by (4). In the figure, the noisy count for the level-1 cell A_1

is 200 and A_1 is divided into a 3×3 grid. In the same way, other level-1 cells, A_2, A_3, A_4, are all divided into 2×2 level-2 grids. The AG approach finally gives noisy counts of mobile servers for each level-2 cell, which are calculated by adding random Laplace noise with mean $1/\epsilon_2$. Finally the CSP publishes these noisy counts together with the structure of AG. The AG approach provides ϵ-DP guarantee according to Theorem 1.

B. CONSTRUCTING REPUTATION-BASED PRIVATE SPATIAL DECOMPOSITION

Note that the PSD constructed before does not contain the reputation information of mobile servers, and therefore the CCP cannot control the quality of the answers. Integrating reputation information into our framework, however, is non-trivial. First, incorporating reputation into the PSD adds a new data dimension, which increases the difficulty of applying DP. Specifically, the CSP should both provide a noisy count of mobile servers following DP and reveal the reputation associated with each mobile server. Thus we design a new data structure, which is a unique feature of our framework. Second, to ensure service quality, the CCP prefers mobile servers with high reputation score. However, MCC resources may be scarce, and it may take a long time to wait for a mobile server with high reputation score. This is impractical for many time-critical tasks. As a result, the CSP should balance between acceptance rate of task allocation and service quality. In the following, we first propose a new data structure, R-PSD, that integrates location and reputation information without compromising mobile server location privacy. Task allocation based on the proposed R-PSD will be detailed in Section V.

The R-PSD consists of multiple layers of sub-PSDs. The range of mobile server reputation scores is first divided into several reputation levels, and mobile servers whose reputation scores fall into the same level are grouped together. The number of levels reflects the granularity of reputation that a specific task needs. Finer granularity leads to better quality control, but incurs higher system overhead in computing R-PSD, choosing the geocast region, and contacting mobile servers. A sub-PSD is constructed, in the same way as constructing a PSD, for each group of mobile servers with the same reputation level. The only difference between a sub-PSD and a PSD is the number of data points. We label sub-PSDs based on its corresponding reputation level. For example, sub-PSD for mobile servers with reputation level i is named as the i-th sub-PSD. Fig. 3 shows an example of R-PSD with two reputation levels.

The privacy budget allocated to all sub-PSDs is set to be ϵ. Since sub-PSDs are calculated based on disjoint subsets of mobile servers, the privacy budget for the combination of sub-PSDs is the maximum of privacy budget consumed in each sub-PSD according to Theorem 2, which is equal to ϵ. The resulting cells in each sub-PSD are different from each other when reputation level is not uniformly distributed among mobile servers.

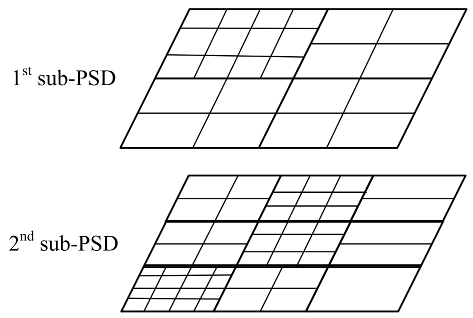


FIGURE 3. Example of R-PSD with two reputation levels.

V. TASK ALLOCATION

In this section, we solve the second challenging issue as explained in Section III. To allocate a task among mobile servers, the CCP queries the R-PSD and computes a geocast region. All mobile servers in the region are notified of the task. The goal of the CCP when determining the geocast region is to achieve high acceptance rate of task allocation while meeting the quality-of-service requirement of the task and reducing the system overhead.

A. ACCEPTANCE RATE CHARACTERIZATION

The mobile cloud in our scenario consists of proximate mobile computing entities instead of distant cloud-based resources. Considering the cost incurred by traveling or communication, a mobile server is more likely to accept a nearby task than a distant one [34]. We model the probability of accepting a task for a mobile server as a function of the distance between the mobile server and the task. For simplicity, we use a linear model to characterize the relationship between individual acceptance rate and mobile server-task distance. Let p_a denote the probability for a mobile server to accept a notified task and d denote the distance between the mobile server and the task. Then we have

$$p_a = p_a^0 - \beta d, \quad 0 \leq d \leq d_{\max} \tag{5}$$

where p_a^0 denotes the acceptance rate of a server within the same level-2 cell as the task, β is a positive parameter, and $d_{\max}$ is a threshold over which the acceptance rate of a server is zero. Obviously, in this model, as the distance increases, the acceptance rate decreases linearly. Note that other models depending on different applications could be used in our framework as well.

Now we propose an analytical model that enables the CCP to estimate the expected acceptance rate of task allocation AR_k, i.e., the probability that at least k mobile servers accept the task in a given geocast region. It is determined by the acceptance rate of each individual mobile server and the number of mobile servers in the geocast region.

First, assume we have m independent mobile servers in a single level-2 cell of the PSD (hence each server has the same p_a), and the task requires at least k mobile servers to perform the task. The overall acceptance rate AR_k is calculated as follows:

$$AR_k = 1 - \sum_{i=0}^{k-1} \binom{m}{i} (p_a)^i (1 - p_a)^{m-i}. \qquad (6)$$

Next, if a geocast region contains multiple level-2 cells, the overall acceptance rate can be calculated iteratively: We first start with an empty set and initialize AR_k to 0. Then a new cell is added to the set, and the value of AR_k is updated. The process continues until we have added all cells in the geocast region. Let $AR^{\text{old}}(j)$ denote the probability that exact j mobile servers accept the received task in the original region, $AR^{\text{new}}(j)$ denote the probability that exact j mobile servers accept the received task in the new region, and $AR^{+}(j)$ denote the probability that exact j mobile servers accept the received task in the added region. We have the following update equation:

$$AR^{\text{new}}(j) = \sum_{n=0}^{j} \left(AR^{\text{old}}(n) \times AR^{+}(j-n) \right). \qquad (7)$$

Then the probability that at least k mobile servers accept a task in the geocast region is calculated as:

$$AR_k = 1 - \sum_{j=0}^{k-1} AR^{\text{new}}(j). \qquad (8)$$

Note that each task can have a minimum requirement on its acceptance rate. When constructing the geocast region, the CCP needs to ensure that the region can satisfy this threshold $\overline{AR_k}$.

B. SERVICE QUALITY CHARACTERIZATION

Besides the acceptance rate of task allocation, the CCP also needs to ensure the quality of results. Intuitively, the quality of results depends on the reputation of participated servers. However, it is hard to learn which servers will participate in the task allocation process, since the CCP can only disseminate a task to a region and wait for mobile servers to reply. Suppose a task needs k mobile servers to perform, the CCP will allocate the task to the first k mobile servers who reply. The above factor makes it difficult to directly control the quality of results.

Our proposed R-PSD provides noisy counts of mobile servers with different reputation levels in a region, which enables the CCP to determine the cells in which more trustworthy mobile servers are located. Therefore the problem of selecting reliable mobile servers can be transformed into the problem of determining a geocast region that contains enough trustworthy mobile servers. Note that in our system, multiple mobile servers can perform the same task, and their results need to be aggregated together to get the final result. Suppose we have servers with l different reputation levels, and the number of servers for each reputation level is $w_i, i = 1, 2, \ldots, l$. The quality of results can be captured by a function $\rho(\mathbf{w})$ where $\mathbf{w} = [w_1, w_2, \ldots, w_l]$ denotes the reputation distributions. Intuitively, $\rho(\cdot)$ is a positive-valued function, and its value increases monotonically when the portion of servers with high reputation level increases. The function $\rho(\cdot)$ quantifies different aspects of the result under different application scenarios. In application that monitors the value of certain metrics, such as epidemic monitoring, $\rho(\cdot)$ may represent the accuracy of estimation. On the other hand, in applications that requires decision making, such as forecasting an epidemic outbreak, $\rho(\cdot)$ may denote the detection and false alarm probabilities. Here, we assume that the minimum service quality indicated by the task is represented as a threshold $\overline{\rho}$, which must be satisfied by the CCP.

C. GEOCAST REGION CONSTRUCTION WITH PSD

Given a task, the geocast region should be constructed to balance three goals: (1) acceptance rate of task allocation should be close to 100%, (2) the number of notified mobile servers should be small, and (3) the aggregate result of participating mobile servers should be trustworthy. However, it is not hard to see that optimizing these three design goals simultaneously may be impossible in practice. For instance, to get a high acceptance rate of task allocation and a small number of notified mobile servers, the CCP has to select mobile servers whose individual acceptance rate is high (i.e., nearby mobile servers) despite of their trust levels, which makes the service quality suboptimal. On the other hand, if we want to improve the service quality, the CCP should notify mobile servers with high reputation levels, which increases the difficulty of getting enough mobile servers. In this case, either the acceptance rate is reduced, or the number of notified mobile servers is increased.

For ease of presentation, we first present in this section an algorithm of geocast region construction which only considers the first two goals. The algorithm takes task t and the PSD of mobile servers as input, and outputs the geocast region where task t is allocated to. The basic idea of our algorithm is to first initialize the geocast region with the level-2 cell that covers task t, and determines the acceptance rate AR of this cell by (6). At each step, the cell that produces the largest increase of acceptance rate is added. Considering the relationship between acceptance rate and distance, we first choose cells that are closer to task t. The geocast region stops expanding either when AR exceeds the threshold $\overline{AR_k}$, or when the size of the geocast region is larger than $2 \times d_{\max}$ since AR cannot increase any further beyond this range. Algorithm 1 gives the detailed steps of the proposed greedy algorithm.

In summary, Algorithm 1 expands the geocast region by adding the cell with the maximum marginal AR at each step, which depends on the cell position and mobile server distributions in that cell based on (6).

D. GEOCAST REGION CONSTRUCTION WITH R-PSD

Now we present the greedy algorithm that considers all three goals, and determine a geocast region based on the R-PSD. Note that in the R-PSD, each sub-PSD partitions the space in their own ways, and thus cells in different sub-PSDs

Algorithm 1 Greedy Algorithm With PSD

```
Input: Task $t$, $d_{\max}$, $\overline{AR}_k$, $k$
Output: Geocast region $\Omega$
 1: Initialize $\Omega = \emptyset$, $AR_k = 0$;
 2: Let $\mathcal{U}$ denote the square of length $2 \times d_{\max}$ centered at
    the task location;
 3: Let $AR(\cdot)$ denote the overall acceptance rate $AR_k$ of a
    region;
 4: $\mathcal{Q} \leftarrow$ {the level-2 cell that covers task $t$};
 5: repeat
 6:   if $\mathcal{Q} = \emptyset$ then
 7:     return $\Omega$
 8:   else
 9:     $c^* \leftarrow \text{argmax}_{c \in \mathcal{Q}} AR\,(GR \cup c)$;
10:     $\mathcal{Q} \leftarrow \mathcal{Q} \setminus \{c^*\}$;
11:     $\Omega \leftarrow \Omega \cup \{c^*\}$;
12:     $AR_k \leftarrow AR(\Omega)$;
13:     $\mathcal{S} \leftarrow (\{\text{neighbors of } c^*\} \setminus \Omega) \cap \mathcal{U}$;
14:     $\mathcal{Q} \leftarrow \mathcal{Q} \cup \mathcal{S}$;
15:   end if
16: until $AR_k \geq \overline{AR}_k$
17: return $\Omega$;
```

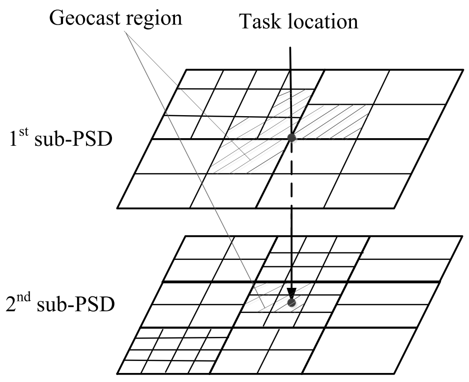


FIGURE 4. Illustration of a geocast region with R-PSD.

may overlap. The geocast region in this case is a combination of cells in all sub-PSDs for the R-PSD. An example of a geocast region is illustrated in Fig. 4.

The input to the algorithm is task t, R-PSD with l sub-PSDs, and parameters $d_{\max}$, $\overline{AR}_k$, $\overline{\rho}$, and k. The variable w_i represents the noisy count of servers included in the geocast region GR that belongs to the i-th sub-PSD, $i = 1, 2, \ldots, l$. In addition to the constraint $AR_k \geq \overline{AR}_k$ considered in Algorithm 1, we add a new constraint $\rho(\{w_i\}_1^l) < \overline{\rho}$, which guarantees the service quality of the chosen mobile servers. The geocast region GR is first initialized to an empty set and then expanded iteratively. In each iteration, a new cell that both best improves AR_k and ensures $\rho(\{w_i\}_1^l) < \overline{\rho}$ is selected and added to GR. The geocast region stops expanding when no new cells within distance of $d_{\max}$ can be added or until AR_k exceeds $\overline{AR}_k$. The algorithm is a greedy approach that always chooses the cell with the highest acceptance rate while guaranteeing service quality at each iteration.

VI. GEOCAST COMMUNICATION PROCESS

To solve the third challenge as explained in Section III, task dissemination in the selected geocast region can be implemented with the infrastructure of the CSP. The CSP either directly sends a message to each mobile server in the region, or notifies the message to several mobile servers and let the servers relay the message hop-by-hop. The communication cost for the former approach is proportional to the average number of notified mobile servers, which may be high when a large number of mobile servers should be notified. Hence it is suitable only when servers are sparsely distributed. On the other hand, the hop-by-hop approach reduces the overhead for the CSP. An efficient way to deliver packets in a geographic region is using geographic routing, i.e., geocast. Geocast has the advantage of lower overhead and faster response to dynamics over ad hoc routing protocols when a message needs to be broadcasted to a geographic area. Geocast in our protocol is slightly different from that in previous papers [26], [35] by adding a new dimension, the reputation level. Only mobile servers that satisfy both the reputation and the location requirements of the geocast region will be notified. This is possible since the CSP already has access to locations and reputation scores of mobile servers when calculating R-PSD. In this paper, we use the average number of notified mobile servers to measure the system overhead which includes the geocast overhead.

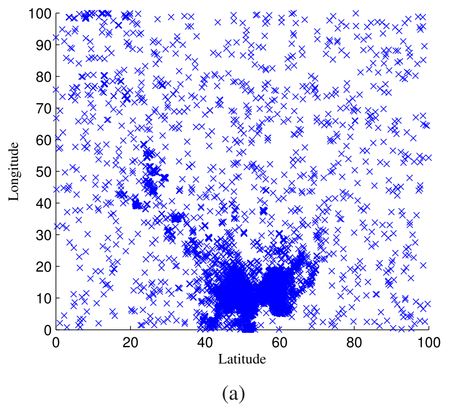
VII. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our proposed framework using real-world datasets.

A. EXPERIMENTAL SETUP

We use two real-world datasets: Gowalla [36] and CrowdFlower [37]. The Gowalla dataset is used to simulate the spatial distribution of mobile servers in our experiments, which contains a total of 6, 442, 890 check-ins on a location-based social networking website from Feb. 2009 to Oct. 2010. We use the check-in history of Gowalla users as the task allocation history of mobile servers. We consider Gowalla users as the mobile servers. We assume all check-ins of a Gowalla user, except the latest one, are tasks that have been completed by him/her, and the latest check-in location is treated as his/her current location. Due to data sparsity, there are no data points in some area of the dataset. Hence we overlay the dataset with a set of uniformly distributed mobile servers. The resulting mobile server distribution is shown in Fig. 5a, where each cross in the figure represents the current location of a mobile server.

We extract the reputation scores of mobile servers based on a study carried out in [38], which asks participants to report traffic events in Dublin. They created and assigned approximately 4000 tasks and calculated reputation scores of participants based on the ground truth of the tasks and historical performance in CrowdFlower. We randomly assign the reputation scores to Gowalla users so that we would get a dataset which contains both task performance history and



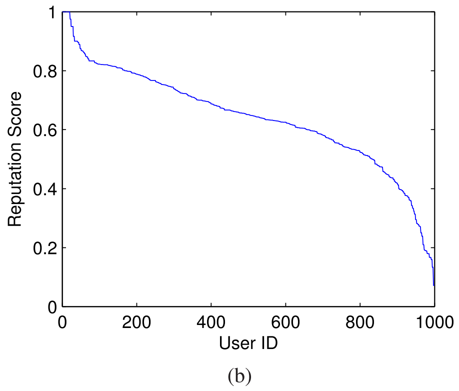


FIGURE 5. (a) Current locations of mobile servers; (b) Reputation distribution of mobile servers.

reputation scores of servers. The reputation distribution is given in Fig. 5b.

As we presented in Section V, the service quality of the task can be captured by function $\rho(\cdot)$. In our experiment, we use the results in [39] to estimate the service quality of the task when k mobile servers with potentially different reputation levels perform the same task. In their paper, the error rate of completing a task, denoted as ER, is the metric to quantify the service quality. Suppose we use the majority voting to aggregate the results of mobile servers for a task. It is proved in [39] that the error rate ER, the required number of servers k, and the collective quality Q satisfy the following inequality:

$$\frac{kQ^2}{4} \leq \ln \frac{1}{ER}. \tag{9}$$

The collective quality Q is calculated in the following way. Define X as a random variable to describe the event that a mobile server submits a correct answer. We have $\Pr(X = \text{True}) = p_r$ and $\Pr(X = \text{False}) = 1 - p_r$, where p_r is the reputation score of a mobile server in our scenario. If the reputation scores of mobile servers are independent and identically distributed, we have

$$Q = \mathbb{E}\big[(2p_r - 1)^2\big], \tag{10}$$

where the expectation is take with respect to the distribution of reputation scores. Therefore, given an error rate requirement ER for a task and the number of required mobile servers k, we can deduce a corresponding requirement on the reputation score distribution in the geocast region.

In our experiments, we suppose that mobile servers are divided into two groups whose reputation scores fall into [0, 0.5] and (0.5, 1], respectively. The number of servers in each group is w_1 and w_2, respectively. For a given geocast region, the collective quality depends on the ratio of the number of mobile servers in each group, i.e., w_1/w_2. If the reputation score in each reputation level follows a uniform distribution, the collective quality Q can be calculated from (10) as

$$\begin{aligned} Q &= \mathbf{E}\big[(2p_r - 1)^2\big] \\ &= \int_0^{0.5} \frac{w_1}{w_1 + w_2}(2x-1)^2 \frac{1}{0.5}\,\mathrm{d}x \\ &\quad + \int_{0.5}^{1} \frac{w_2}{w_1 + w_2}(2x-1)^2 \frac{1}{0.5}\,\mathrm{d}x \\ &= \frac{w_1}{(w_1 + w_2) \times 3}. \end{aligned} \tag{11}$$

Given a requirement on ER and the number of mobile servers k, we can deduce a lower bound for Q and further calculate a requirement on w_1 and w_2. When constructing the geocast region, the CCP needs to ensure that the region can satisfy this requirement.

We randomly generate 1,000 tasks which are uniformly distributed in an area, and use our algorithms to calculate GR regions for each task. We also implement a baseline algorithm that is privacy-oblivious. The baseline algorithm has access to exact locations of all servers and always adds the nearest server to a set until the acceptance rate of the set surpasses the acceptance threshold $\overline{AR_k}$.

B. EXPERIMENTAL RESULTS

1) EVALUATION OF SYSTEM OVERHEAD FOR ACHIEVING PRIVACY

We first evaluate the system overhead (i.e. the average number of notified mobile servers) incurred by our privacy-preserving method. Fig. 6 presents the system overhead for our private algorithm (the greedy algorithm based on PSD) and the baseline algorithm when varying privacy budget ϵ.

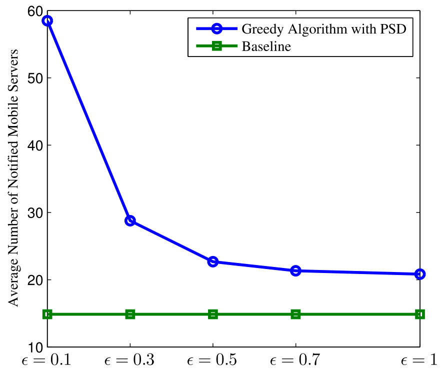


FIGURE 6. Effect of privacy budget ϵ when $\overline{AR_k} = 0.9$.

As ϵ increases, which means mobile servers are less sensitive to their privacy breach, the PSD provides more accurate data for geocast, and the geocast overhead decreases as well. Additionally, we can observe that compared with the baseline, our private algorithm does not significantly increase the system overhead, especially when the privacy budget ϵ is larger than 0.3. This shows the ability for our algorithm to choose nearby mobile servers for a task.

2) EFFECT OF VARYING ACCEPTANCE RATE THRESHOLD ON SYSTEM OVERHEAD

We evaluate the performance of Algorithm 1 and Algorithm 2 by varying the threshold of the expected acceptance rate $\overline{AR}_k$. We use the privacy budget $\epsilon = 0.5$ for both PSD and R-PSD. Fig. 7 illustrates the impact of increasing $\overline{AR}_k$. As one would expect, for a higher $\overline{AR}_k$, a larger GR region should be selected, and thus the overhead will increase.

Algorithm 2 Greedy Algorithm With R-PSD

```
Input: Task $t$, R-PSD with $l$ sub-PSDs, $d_{\max}$, $\overline{AR}_k$, $\overline{\rho}$, $k$
Output: Geocast region $\Omega$
1:  Initialize $\Omega = \emptyset$, $AR_k = 0$, $w_i = 0$, $i = 1, 2, \ldots, l$;
2:  Let $\mathcal{U}$ denote the square of length $2 \times d_{\max}$ centered at the task location;
3:  Let $AR(\cdot)$ denote the overall acceptance rate $AR_k$ of a region;
4:  Let $\rho(w_1, w_2, \ldots, w_l)$ denote the service quality given $w_1, w_2, \ldots, w_l$;
5:  for $i = 1, 2, \ldots, l$ do
6:      $\mathcal{Q}_i \leftarrow$ {level-2 cell in $i$-th sub-PSD that covers task $t$};
7:  end for
8:  repeat
9:      if $\mathcal{Q}_i = \emptyset, \forall i$ then
10:         return $\Omega$
11:     else
12:         for $i = 1, 2, \ldots, l$ do
13:             $c_i^* \leftarrow \mathrm{argmax}_{c_i \in \mathcal{Q}_i} AR(\Omega \cup c_i)$;
14:         end for
15:         Sort $c_i^*, i = 1, 2, \ldots, l$ in decreasing order of $AR(\Omega \cup \{c_i^*\})$;
16:         Compute $\rho(w_1, w_2, \ldots, w_t)$ for $\Omega \cup \{c_i^*\}$ in the previously computed order until we meet a $c_j^*$ that satisfies $\rho(w_1, w_2, \ldots, w_t) \geq \overline{\rho}$;
17:         $AR_k \leftarrow AR(\Omega \cup \{c_j^*\})$;
18:         $\mathcal{Q}_j \leftarrow \mathcal{Q}_j \setminus \{c_j^*\}$;
19:         $\Omega \leftarrow \Omega \cup \{c_j^*\}$;
20:         for $i = 1, 2, \ldots, l$ do
21:             $\mathcal{S}_i = (\{\text{neighbors of } c_i^*\} \setminus \Omega) \cap \mathcal{U}$;
22:             $\mathcal{Q}_i = \mathcal{Q}_i \cup \mathcal{S}_i$;
23:         end for
24:     end if
25: until $(AR_k \geq \overline{AR}_k) \wedge (\rho(w_1, w_2, \ldots, w_l) \geq \overline{\rho})$
26: return $\Omega$
```

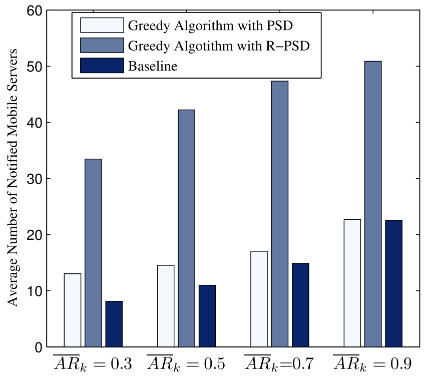


FIGURE 7. Comparison on varying acceptance rate threshold $\overline{AR}_k$ with $\epsilon = 0.5$.

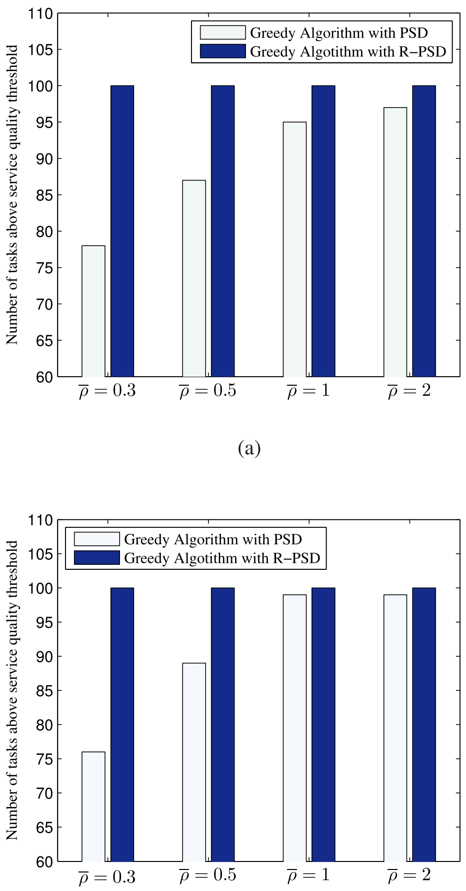


FIGURE 8. Effect of reputation threshold $\overline{\rho}$ on the number of tasks that reaches the threshold of service quality when (a) $\overline{AR}_k = 0.7$, $\epsilon = 0.5$; (b) $\overline{AR}_k = 0.9$, $\epsilon = 0.5$.

3) ACCEPTANCE RATE AND RELIABILITY

Fig. 8 compares Algorithm 1 and Algorithm 2 with different ratios of mobile servers with different reputation levels. We can see that as the threshold $\overline{\rho}$ decreases, i.e., more mobile servers with good reputation are needed, the advantage of Algorithm 2 over Algorithm 2 becomes larger. Note that we use the reputation distribution in Fig. 5b for our experiment, where a large portion of mobile servers have high reputation scores. However, in practice, the resource of highly reliable servers are usually scarce, and thus the advantage of Algorithm 2 will become more prominent.

VIII. RELATED WORK

Mobile cloud computing (MCC) extends the concept of cloud computing into the mobile domain. There are generally two types of mobile clouds in MCC: infrastructure-based and ad hoc [6]. The infrastructure-based mobile cloud consists of stationary computing resources and provides services to the mobile users via the Internet. Research in this direction generally focuses on reducing the cost and complexity of using cloud resources. Chun et al. [40] propose a CloneCloud that clones the mobile platform into the cloud VM and enables the remote server to execute computation-extensive part of a job. In [41], Zhang et al. partition a job into multiple small tasks which are least dependent on each other. These small tasks are either executed locally or remotely based on the resource intensity.

Alternatively, in the ad hoc mobile cloud, a collection of mobile devices performs as cloud resources and provides access to local or Internet-based cloud services to other mobile users. In this paper, we focus on the second case. There are only a few papers along this line [7]–[10]. The work in [10] exploits the resource of nearby mobile devices to perform intense computation jobs and designs a scheme to alleviate frequent disconnections of mobile servers. On the other hand, Huerta-Canepa and Lee in [9] use an ad hoc cluster of nearby mobile devices to enhance computing capabilities of a mobile device with minimum network latency and traffic. Their system continues monitoring traces of mobile servers and establishes peer-to-peer connection among them. The work in [7] provides a preliminary framework that pays nearby smartphones to run resource-intensive tasks.

As emphasized in [7], security and privacy of mobile servers is a critical concern in ad hoc mobile cloud. Moreover, it is also challenging to ensure the quality of the service provided by these dynamic mobile servers. There is an inherent conflict between quality of service (i.e., utility) and privacy in task allocation, which complicates the problem. None of previous works focus on the privacy issue of mobile servers. In this paper, we propose a framework that provides solutions to the above challenges, where both location privacy and service quality are considered.

Several solutions to privacy issues in mobile applications have been proposed. For example, aggregation is a common approach to hide individual sensitive information when only statistics of users are required. In [12], a user can learn traffic condition based on reports of other mobile users, and thus the location information of all the users are learned by the service provider. To protect location privacy, individual sensitive information is hidden through aggregation, and only statistics of users are provided. However, this approach only calculates statistics and thus cannot be used to select mobile servers in an ad hoc mobile cloud. Another approach is used in location-based services, where the true locations are obfuscated in location-based queries, and the service provider returns results based on the obfuscated query. In [13], Duckham and Kulik provide a mechanism to calculate location that balances location privacy and the need for high quality location-based service. With the mechanism, the location-based service provider only learns minimum information to provide service of required quality. In [14], Shokri et al. propose a game-theoretic framework that provides an optimal location-privacy protection mechanism and consider the adversarial knowledge during mechanism design. In our scenario, however, the private information is no longer part of a location-based query, but the result of a location-based query regarding the task. Some papers [15], [16] consider queries on private locations in an outsourced database, but they only protect private data from an intermediate service provider while assuming a trust relationship between the data owner and the querying entity. This is not true in our scenario because mobile servers and the CCP may not share an inherent trust relationship. In our scenario, however, we want to protect location privacy from the mobile cloud computing provider (CCP), who needs locations to perform task allocation. In [42], Andrés *et al.* define geo-distinguishability to protect exact locations while exposing approximate information. However, in our scenario, approximate location information would undermine the performance of task allocation. The approaches in [43] and [44] encrypt sensitive information to protect them against eavesdroppers, while our approach protects sensitive information against data processors. Thus, they are complementary to our paper.

IX. CONCLUSION

In this paper, we have investigated the privacy issues in the ad hoc mobile cloud computing, and have proposed a framework that protects the location privacy of mobile servers when allocating mobile cloud computing tasks. Considering the dynamic and diverse nature of mobile servers, we have designed a new data structure R-PSD and developed an efficient search strategy that finds appropriate R-PSD partitions to ensure high service quality. We have conducted extensive experiments based on real-world datasets to demonstrate the effectiveness of our proposed framework.

REFERENCES

[1] J. Schiller and A. Voisard, *Location-Based Services*. Amsterdam, The Netherlands: Elsevier, 2004.

[2] M. Spreitzer and M. Theimer, "Providing location information in a ubiquitous computing environment," in *Mobile Computing*. New York, NY, USA: Springer-Verlag, 1996, pp. 397–423.

[3] T. Choudhury *et al.*, "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Comput.*, vol. 7, no. 2, pp. 32–41, Apr./Jun. 2008.

[4] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell, "BikeNet: A mobile sensing system for cyclist experience mapping," *ACM Trans. Sensor Netw.*, vol. 6, no. 1, p. 6, Dec. 2009.

[5] F. Alt, A. S. Shirazi, A. Schmidt, U. Kramer, and Z. Nawaz, "Location-based crowdsourcing: Extending crowdsourcing to the real world," in *Proc. 6th Nordic Conf. Human-Comput. Interact., Extending Boundaries*, 2010, pp. 13–22.

[6] A. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 393–413, Feb. 2014.

[7] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 337–368, Feb. 2014.

[8] N. Fernando, S. W. Loke, and W. Rahayu, "Dynamic mobile cloud computing: Ad hoc and opportunistic job sharing," in *Proc. 4th IEEE Int. Conf. Utility Cloud Comput. (UCC)*, Dec. 2011, pp. 281–286.

[9] G. Huerta-Canepa and D. Lee, "A virtual cloud computing provider for mobile devices," in *Proc. 1st ACM Workshop Mobile Cloud Comput. Services, Social Netw. Beyond*, 2010, p. 6.

[10] E. E. Marinelli, "Hyrax: Cloud computing on mobile devices using MapReduce," M.S. thesis, School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, 2009.

[11] I. Krontiris, F. C. Freiling, and T. Dimitriou, "Location privacy in urban sensing networks: Research challenges and directions [Security and Privacy in Emerging Wireless Networks]," *IEEE Wireless Commun.*, vol. 17, no. 5, pp. 30–35, Oct. 2010.

[12] J. W. S. Brown, O. Ohrimenko, and R. Tamassia, "Haze: Privacy-preserving real-time traffic statistics," in *Proc. 21st ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2013, pp. 540–543.

[13] M. Duckham and L. Kulik, "A formal model of obfuscation and negotiation for location privacy," in *Pervasive Computing*. New York, NY, USA: Springer-Verlag, 2005, pp. 152–170.

[14] R. Shokri, G. Theodorakopoulos, C. Troncoso, J.-P. Hubaux, and J.-Y. L. Boudec, "Protecting location privacy: Optimal strategy against localization attacks," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2012, pp. 617–627.

[15] M. L. Yiu, G. Ghinita, C. S. Jensen, and P. Kalnis, "Enabling search services on outsourced private spatial data," *VLDB J.*, vol. 19, no. 3, pp. 363–384, Jun. 2010.

[16] B. Yao, F. Li, and X. Xiao, "Secure nearest neighbor revisited," in *Proc. IEEE 29th Int. Conf. Data Eng. (ICDE)*, Apr. 2013, pp. 733–744.

[17] H. To, G. Ghinita, and C. Shahabi, "A framework for protecting worker location privacy in spatial crowdsourcing," *Proc. VLDB Endowment*, vol. 7, no. 10, pp. 919–930, Jun. 2014.

[18] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu, "Differentially private spatial decompositions," in *Proc. IEEE 28th Int. Conf. Data Eng. (ICDE)*, Apr. 2012, pp. 20–31.

[19] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Advances in Cryptology—EUROCRYPT*. New York, NY, USA: Springer-Verlag, 2006, pp. 486–503.

[20] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography*. New York, NY, USA: Springer-Verlag, 2006, pp. 265–284.

[21] C. Dwork, "Differential privacy: A survey of results," in *Theory and Applications of Models of Computation*. New York, NY, USA: Springer-Verlag, 2008, pp. 1–19.

[22] F. McSherry and I. Mironov, "Differentially private recommender systems: Building privacy into the net," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 627–636.

[23] H. Samet, *The Design and Analysis of Spatial Data Structures*. Boston, MA, USA: Addison-Wesley, 1990.

[24] W. Qardaji, W. Yang, and N. Li, "Differentially private grids for geospatial data," in *Proc. IEEE 29th Int. Conf. Data Eng. (ICDE)*, Apr. 2013, pp. 757–768.

[25] S. Abolfazli, Z. Sanaei, M. Shiraz, and A. Gani, "MOMCC: Market-oriented architecture for mobile cloud computing based on service oriented architecture," in *Proc. 1st IEEE Int. Conf. Commun. China Workshops (ICCC)*, Aug. 2012, pp. 8–13.

[26] J. C. Navas and T. Imielinski, "GeoCast—Geographic addressing and routing," in *Proc. 3rd Annu. ACM/IEEE Int. Conf. Mobile Comput. Netw.*, 1997, pp. 66–76.

[27] A. Chandra and J. Weissman, "Nebulas: Using distributed voluntary resources to build clouds," in *Proc. 1st Usenix Workshop Hot Topics Cloud Comput. (HotCloud)*, 2009, p. 2.

[28] T. Yan, M. Marzilli, R. Holmes, D. Ganesan, and M. Corner, "mCrowd: A platform for mobile crowdsourcing," in *Proc. 7th ACM Conf. Embedded Netw. Sensor Syst.*, 2009, pp. 347–348.

[29] S. Buchegger and J.-Y. L. Boudec, "Performance analysis of the CONFIDANT protocol," in *Proc. 3rd ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2002, pp. 226–236.

[30] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Trans. Sensor Netw.*, vol. 4, no. 3, p. 15, May 2008.

[31] K. L. Huang, S. S. Kanhere, and W. Hu, "Are you contributing trustworthy data? The case for a reputation system in participatory sensing," in *Proc. 13th ACM Int. Conf. Modeling, Anal., Simulation Wireless Mobile Syst.*, 2010, pp. 14–22.

[32] H. Lin, X. Zhu, Y. Fang, D. Xing, C. Zhang, and Z. Cao, "Efficient trust based information sharing schemes over distributed collaborative networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 9, pp. 279–290, Sep. 2013.

[33] R. Dingledine, N. Mathewson, and P. F. Syverson, "Tor: The second-generation onion router," in *Proc. USENIX Secur. Symp.*, 2004, pp. 303–320.

[34] L. Kazemi and C. Shahabi, "GeoCrowd: Enabling query answering with spatial crowdsourcing," in *Proc. 20th Int. Conf. Adv. Geograph. Inf. Syst.*, 2012, pp. 189–198.

[35] C. Maihofer, "A survey of geocast routing protocols," *IEEE Commun. Surveys Tuts.*, vol. 6, no. 2, pp. 32–42, Dec. 2004.

[36] *Gowalla*, accessed on Mar. 2015. [Online]. Available: https://snap.stanford.edu/data/loc-gowalla.html

[37] *Crowdflower*, accessed on Jan. 2014. [Online]. Available: http://www.crowdflower.com/

[38] I. Boutsis and V. Kalogeraki, "On task assignment for real-time reliable crowdsourcing," in *Proc. IEEE 34th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun./Jul. 2014, pp. 1–10.

[39] D. R. Karger, S. Oh, and D. Shah, "Budget-optimal task allocation for reliable crowdsourcing systems," *Oper. Res.*, vol. 62, no. 1, pp. 1–24, 2014.

[40] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "CloneCloud: Elastic execution between mobile device and cloud," in *Proc. 6th Conf. Comput. Syst.*, 2011, pp. 301–314.

[41] X. Zhang, S. Jeong, A. Kunjithapatham, and S. Gibbs, "Towards an elastic application model for augmenting computing capabilities of mobile platforms," in *Mobile Wireless Middleware, Operating Systems, and Applications*. New York, NY, USA: Springer-Verlag, 2010, pp. 161–174.

[42] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2013, pp. 901–914.

[43] X. Lin, R. Lu, X. Shen, Y. Nemoto, and N. Kato, "Sage: A strong privacy-preserving scheme against global eavesdropping for ehealth systems," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 365–378, May 2009.

[44] M. M. Fouda, Z. M. Fadlullah, N. Kato, R. Lu, and X. Shen, "A lightweight message authentication scheme for smart grid communications," *IEEE Trans. Smart Grid*, vol. 2, no. 4, pp. 675–685, Dec. 2011.