Knowl Inf Syst (2013) 37:105–128
DOI 10.1007/s10115-012-0573-y

REGULAR PAPER

TAPAS: Trustworthy privacy-aware participatory sensing

Leyla Kazemi · Cyrus Shahabi

Received: 24 January 2012 / Revised: 1 September 2012 / Accepted: 6 October 2012 /
Published online: 25 October 2012

Abstract With the advent of mobile technology, a new class of applications, called participatory sensing (PS), is emerging, with which the ubiquity of mobile devices is exploited to collect data at scale. However, privacy and trust are the two significant barriers to the success of any PS system. First, the participants may not want to associate themselves with the collected data. Second, the validity of the contributed data is not verified, since the intention of the participants is not always clear. In this paper, we formally define the problem of privacy and trust in PS systems and examine its challenges. We propose a trustworthy privacy-aware framework for PS systems dubbed TAPAS, which enables the participation of the users without compromising their privacy while improving the trustworthiness of the collected data. Our experimental evaluations verify the applicability of our proposed approaches and demonstrate their efficiency.

Keywords Participatory sensing · Privacy · Trust · Location privacy

1 Introduction

Many studies suggest significant future growth in the number of mobile phone users, the phones hardware and software features, and the broadband bandwidth. Therefore, an active area of research is to fully utilize this new platform for various tasks, among which the most promising is *participatory sensing (PS)*. The goal is to exploit the mobile users by leveraging their sensor-equipped mobile devices to collect and share data. While many PS systems are *unsolicited*, where users can participate by randomly contributing data (e.g., Youtube, Flickr), other PS systems are *campaign*-based and require a *coordinated* effort of

L. Kazemi (✉) · C. Shahabi
Information Laboratory, Computer Science Department, University of Southern California,
Los Angeles, CA 90089-0781, USA
e-mail: lkazemi@usc.edu

C. Shahabi
e-mail: shahabi@usc.edu

participants to collect a particular set of data. Some real-world examples of PS campaigns include [8,22,31,30]. For example, the Mobile Millennium project [30] by UC Berkeley is a state-of-the-art system that uses GPS-enabled mobile phones to collect en route traffic information and upload it to a server in real time. The server processes the contributed traffic data, estimates future traffic flows, and sends traffic suggestions and predictions back to the mobile users. Similar projects were implemented earlier by CalTel [22] and Nericell [31] which used mobile sensors/smart phones mounted on vehicles to collect information about traffic, WiFi access points on the route and road condition. In CycleSense [8], bikers record biking routes during their daily commute in the Los Angeles area, along with information about air quality, hazards, traffic conditions, accidents, etc. Bikers trust the server and upload all their data to the server. The server publishes the data on a public website with the mobile phone number removed.

However, two major impediments may hinder PSs practicality and success in real-world: privacy and trust. First, the users of a PS system may not want to associate themselves with the data that are collected and transmitted. Consider a scenario where a PS campaign is interested in collecting pictures and videos of the anti-government demonstrations from various locations of a city (termed data collection points or *DC-points*) through a coordinated effort of the participants. Therefore, each participant u should query the server for the locations of closeby DC-points. However, u may not be willing to reveal his identity due to safety reasons. One may argue that a simple anonymization of the participant's identity through a trusted server can resolve the problem. However, due to the strong correlation between people and their movements [16], a malicious server can identify u by associating his location information to u. We refer to this process of association of the query to the query location as a *location-based* attack. Second, the data contributed by participants cannot always be trusted, because the motivation of the participants for data contribution is not always clear. Thus, a malicious participant might intentionally collect wrong data. For example, in the same scenario, undercover agents may also upload pictures and videos painting a totally different image of what is occurring. Some skeptics of PS go as far as calling it a garbage-in-garbage-out system due to the issues of trust. Finally, the interplay of privacy and trust makes the problem even more challenging as once we can successfully hide participants identities; evaluating reputation of the participants becomes even harder.

Recently, few studies have focused on addressing the privacy challenges [20,21,24,37] in PS. However, their assumption is that the data generated by the participants are always correct. In [24], a privacy-preserving technique is proposed, which assigns to every participant a set of closeby DC-points while protecting participants from location-based attacks. Moreover, some studies propose to address the trust issues [9,15] in PS by incorporating a trusted software/hardware module in the participants' mobile devices. While this protects the sensed data from malicious software manipulation before sending it to the server, it does not protect the data from participants who either intentionally (i.e., malicious users) or unintentionally (e.g., due to malfunctioning sensors) collect incorrect data. Note that since participants are the data generators, finding a way to verify the data are both critical and challenging. The only way to ensure the correctness of data is that a trusted party collects the data, which is meaningless in the context of private PS because we do not know who contributed the data.

In this paper, we propose a Trustworthy privacy-Aware framework for PArticipatory Sensing campaigns, *TAPAS*. Our key idea here is to increase the chance of validity of the collected data by having multiple participants (termed *replicators*) collect data at each data collection point redundantly. Here, the intuitive assumption, based on the idea of the wisdom of crowds

[39], is that the majority of participants generate correct data. Thus, the data with the majority vote are verified as correct. This indicates that instead of each DC-point be assigned to a particular participant, it is assigned to k participants, where k is a campaign-defined parameter, which determines the level of trust for the collected data. Consequently, the higher the value of k, the more chance that the collected data are correct.

The main question we try to address in this paper is how to pick the k replicators. Intuitively, participants who are closer to a DC-point are better candidates to collect data at the particular DC-point. Thus, for a given participant u, the goal is to assign to him all those DC-points[1], which have u as their k nearest replicators. Thus, our problem is to find the reverse k nearest replicator set for *all* participants, while preserving the participants' location privacy. We refer to this problem as *private all reverse k nearest replicator (private aRkNR or PaRkNR)* problem. We propose TAPAS, a class of three approaches to solve this problem.In all the three approaches, in order to preserve the privacy, participants follow the PiRi approach [24] to blur their location in a cloaked area among $m - 1$ other participants and send cloaked regions to the server. Since the server does not have the exact participants' locations, only an approximate result can be retrieved. While our goal is to minimize the uncertainty in our proposed approaches, we show that the approximation only results in more replicators collecting data, thus increasing the chance of data validity.

The main contributions of this paper include (1) formalizing the interplay of trust and privacy as the private *aRknR* problem; (2) proposing three solutions for this problem, namely *LPT*, *BAL*, and *HBAL*; (3) conducting extensive experiments and comparing results. We verify the applicability of the proposed approaches by confirming no missing hits in all the three approaches. We also show that with HBAL approach, every participant is assigned to on average 25 % extra DC-points (false hits).

The remainder of this paper is organized as follows. Section 2 reviews the related work. In Sect. 3, we discuss some background studies, formally define our problem, and discuss our system model. Thereafter, in Sect. 4, we explain our TAPAS framework. Section 5 presents the experimental results. Finally, in Sect. 6, we conclude and discuss the future directions of this study.

2 Related work

In this section, we review two groups of related studies. The first group focuses on privacy-related problems, whereas the focus of the second group is on the issue of trust.

2.1 Related work in privacy

Privacy-preserving techniques for location privacy have been widely studied in the context of location-based services. One category of techniques [12,26,43] focuses on evaluating the query in a transformed space, where both the data and query are encrypted, and their spatial relationship is preserved to answer the location-based query. Another group of techniques in protecting user's privacy is based on the Private Information Retrieval (PIR) protocol, which allows a client to secretly request a record stored at an untrusted server without revealing the retrieved record to the server. However, the major drawback of most PIR-based approaches for location privacy is that they rely on hardware-based techniques, which typically

[1] The process of DC-point assignment to a particular participant is equivalent to returning the locations of a set of DC-points to the participant so that the participant can go to those locations and collect the corresponding data (e.g., pictures).

utilize a secure coprocessor at the LBS [19,27]. Another group of well-known techniques in preserving users' privacy is the spatial cloaking technique [3,6,11,13,32,23], where the user's location is blurred in a cloaked area, while satisfying the user's privacy requirements. An example of spatial cloaking is the spatial *m*-anonymity (SMA) [40], where the location of the user is cloaked among $m - 1$ other users. Note that anonymity was first discussed in relational databases, where sensitive data (e.g., census, medical) should not be linked to people's identities [1,2]. Later, *m*-anonymity was defined in [35,40]. *m*-Anonymity is also a well-known technique in privacy-preserving data publishing [10], where a data publisher releases the data collected from the data owners to a data miner who will then conduct data mining on the published data, while the privacy of the data owners should be preserved. While any of the privacy-preserving techniques can be utilized to protect the users' privacy, in this paper without loss of generality, we use cloaking techniques due to the following reasons: (1) accuracy and (2) popularity in different environments (i.e., centralized, distributed, peer to peer).

Most of the SMA techniques assume a *centralized* architecture [3,11,32,23], which utilizes a trusted third party known as *location anonymizer*. The anonymizer is responsible for first cloaking user's location in an area, while satisfying the user's privacy requirements, and then contacting the location-based server. The server computes the result based on the cloaked region rather than the user's exact location. Thus, the result might contain false hits. The centralized approach has two drawbacks. First, the centralized approach does not scale because the users should repeatedly report their location to the anonymizer. Second, by storing all the users' locations, the anonymizer becomes a single point for attacks. To address these shortcomings, recent techniques [13] focus on distributed environments, where the users employ some complex data structures to anonymize their location among themselves via fixed infrastructures (e.g., base stations). However, because of high update cost, these approaches are not designed for the cases where users frequently move or join/leave the system. Therefore, alternative approaches have been proposed [6] for unstructured peer-to-peer networks where users cloak their location in a region by communicating with their neighboring peers without requiring a shared data structure.

Despite all the studies about privacy in the context of LBS, only a few work [7,20,21,24,25,34,37] have studied privacy in PS.In [37], the concept of participatory privacy regulation is introduced, which allows the participants to decide the limits of disclosure. Moreover, in [21,20,34], different approaches are proposed, which focus on preserving privacy in a PS campaign during the data contribution, rather than the coordination phase. That is, these approaches deal with how participants upload the collected data to the server without revealing their identity, whereas our focus is on how to privately assign a set of data collection points to each participant. The combination of private data assignment and private data contribution forms an end-to-end privacy-aware framework for the PS systems. Another related study is discussed in [7], in which a privacy-preserving framework for an unsolicited PS is proposed. However, the focus is on unsolicited PS, where participants collect data in an opportunistic manner without the need to coordinate with the server. This indicates that under certain conditions, some data will never be collected, whereas other data might redundantly be collected. This is different from our focus, in which the server should direct the data collection phase to meet certain goals (e.g., assuring the proper assignment of DC-points). The closest work to our problem is [24], where a privacy-preserving technique, namely PiRi, is proposed, which assigns to every participant a set of closeby DC-points without revealing the participants identity. However, their assumption is that the data generated by the participants are always correct. In this work, we employ the PiRi approach to protect the privacy of the participants.

2.2 Related work in trust

Our second category of related work focuses on the issue of trust. We review three sets of related work in this category. The first group studies trust in PS. Existing work in this area propose approaches which incorporate a trusted software/hardware into the mobile device. The role of the trusted module is to sign the data sensed by the mobile sensor. The goal is to avoid any malicious software to manipulate the sensed data before sending it to the server [9,15,17,29,36]. While this achieves trust at some level, it has two drawbacks. One is that it might not be practical for all participants to use such platform. The more important drawback is that these approaches detect whether malicious software modifies the sensed data, but they do not consider the analog attack where users are malicious and collect non-accurate data (e.g., a user can put a sensor in a refrigerator while reporting the temperature); thus, they are not orthogonal to our problem.

Another group of studies focuses on query integrity in data outsourcing [28,38,41,42]. In data outsourcing, a publisher owns a dataset, but it outsources the data to a service provider, which answers the queries asked by the users. However, the service provider is not trusted, and therefore might not correctly answer the query issuer. The goal here is to guarantee that the query answer is both complete (i.e., no missing data) and correct (i.e., no wrong data). These studies differ from our problem because in PS systems, it is not the query result but the data which might not be correct. That is, since the participants are not trusted, the server cannot guarantee that the collected data are valid.

The third group studies reputation systems in P2P networks [18,33]. In this group of work, members of on-line communities with no prior knowledge of each other use the feedback from their peers to assess the trustworthiness of the peers in the community. This is related to our work in a sense that people are more interested in using data from peers with higher reputation. While privacy is usually not a concern in these systems, an additional characteristic of PS that makes our problem more challenging is the *spatial* aspect of the framework. That is, while our goal is to collect data from trustworthy participants, we are also more interested in nearby participants for efficient data collection. Thus, anonymity is hard to achieve, when both location and reputation should be incorporated into the participant's query.

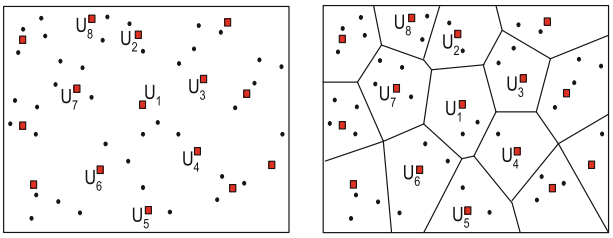
3 Preliminaries

3.1 Background

As already discussed, a privacy-preserving approach in PS systems (PiRi) is proposed in [24], which focuses on private assignment of DC-points to the participants. In the following, we first define the participant assignment problem. Thereafter, we provide a brief background on the PiRi approach.

Definition 1 (*Participant Assignment*) Given a campaign $C(P, U) \in R^2$, with P as the set of DC-points, and U as the set of participants, the *Participant Assignment (PA)* problem is to assign to each participant $u \in U$ any DC-point $p \in P$, such that p is closer to u than to any other participant in U.

The definition of the PA problem states that every DC-point should be assigned to only one participant (i.e., its closest participant). Note that to incorporate trust, multiple participants need to be assigned. In order to solve the PA problem, a straightforward solution is that each



(a) $E(U)$ **(b)** Voronoi Diagram of U

Fig. 1 Illustrating the assignment of DC-points to the participants

participant sends his location to the server. The server then assigns to each participant the set of DC-points close to him by computing the *Voronoi diagram* of the participants. Figure 1 depicts such scenario. The formal definition of the Voronoi diagram is as follows.

Definition 2 (*Voronoi Diagram*) Given an environment $E(U) \in R^2$, with U as the set of participants, the *Voronoi diagram* of U is a partitioning of E into a set of cells, where each cell V_u belongs to a participant u, and any point $p \in E$ in the cell V_u is closer to u than to any other participants in the environment. Here, the closeness between two points is defined in terms of Euclidean distance.

Once the server computes the Voronoi diagram of the participants, it forwards to each participant u, all the DC-points lying inside the corresponding cell V_u. However, for privacy concerns, a participant may not be willing to reveal his identity to the server. Even if the participant hides his identity from the server, a participant can still be identified by his location [16]. Thus, the goal of the PiRi approach is to assign to each participant those DC-points which are closer to that participant than to any other participant, without compromising participants' privacy. As stated in [24], a baseline solution using the existing privacy-preserving techniques is that participants communicate among their peers to compute their Voronoi cells. This enables the participants not to reveal their location to the server. Thereafter, every participant performs a privacy-aware range query to retrieve all the DC-points inside his Voronoi cell. The participant asks such query by first communicating with his neighboring peers via multi-hop routing to find at least $m - 1$ other peers (see the peer-to-peer SMA technique in [6]). It then blurs his location in a cloaked area among $m - 1$ other peers. Thereafter, it sends the cloaked area, along with the range query to the server. This indicates that *every* participant sends a cloaked region instead of his exact location to the server to query for all the closeby DC-points.

However, certain properties of PS campaigns prevent a direct adaptation of these techniques to PS campaigns. These properties leak enough information to the server with which the server can easily identify each participant by linking his query to the query location. One main characteristics of a PS campaign, referred to as *all-inclusivity* property, is that in order to collect data through a coordinated effort, *all* the participants query the PS server for closeby DC-points. This is in contrast to location-based services (LBS) which serve millions of users from which any arbitrary number of users might ask query at a given time and location. This property reveals extra information to the server which introduces a major privacy leak to the system. To illustrate, consider an extreme case where the server knows the locations of all the participants. Thus, on one hand the server receives a set of query regions, and on the other hand, the server has a set of users' locations. Trivially, each query region overlaps with one and only one participant (due to the PS properties). Therefore, the server

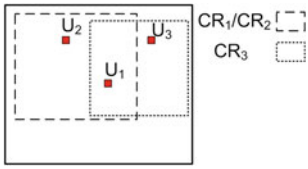


Fig. 2 Illustrating an example of all-inclusivity leak

can associate the query to its participant by solving a simple matching problem between these two sets of points-regions. In more general cases, the more information the server has about the participants' locations, the more accurate the matches.

Figure 2 illustrates such scenario, where users $U_{1...3}$ participate in the campaign. The figure shows that U_1 cloaks himself with U_2. The dotted rectangle CR_1 depicts the cloaked region of U_1. Similarly, U_2 forms a cloaked region with U_1. Consequently, both U_1 and U_2 form identical cloaked regions. The figure also depicts that U_3 cloaks himself with U_1. Accordingly, the server can easily identify U_3 by relating it to the cloaked region CR_3, since U_3 appears only once (i.e., CR_3) in all the three submitted cloaked regions to the server. This indicates the more participants submit queries to the server, the more information server has to infer the participants' identities.

One of the objectives of the PiRi approach is to prevent the all-inclusivity leak by minimizing the number of queries submitted to the server. PiRi is based on the observation that the range queries sent to the server have significant overlaps. Therefore, instead of each participant asking a separate query, only a group of representative participants ask queries from the server on behalf of all the participants and share their results with those who have not posed any query. Note that the only entity that has knowledge of the location of DC-points is the server. Therefore, somehow the points to be collected must be acquired from the server after all. The question is how to pick the representatives. To do this, a distributed voting mechanism is devised where every participant votes for one participant. The representatives are then selected based on the majority of votes among their *local peers* (i.e., set of close-by participants whose query regions are contained in that of the representative participant) to query the server on behalf of the rest of the participants. Consequently, once the representative participants receive their query results from the server, they share the results with their local peers.

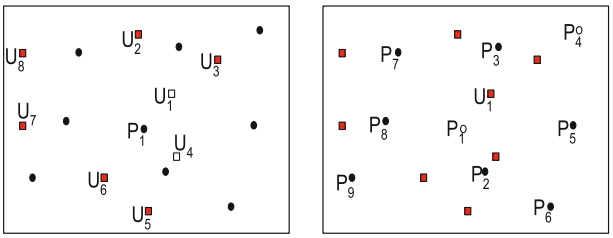
3.2 Formal problem definition

Unlike the PiRi [24] approach where each DC-point is assigned to one participant, to ensure trust, we need multiple participants assigned to each DC-point. Hence, in order to extend PiRi to incorporate a trust level of k, every DC-point should be assigned to at least k participants. Thus, we need to introduce the concepts of replica and replicator.

Definition 3 (*Replica*) Given a set of participants U and a set of DC-points P, we refer to a *replica* r_i^j of $P_i \in P$ as a copy of data collected at point P_i by participant U_j. We also refer to U_j as a *replicator* of P_i.

Intuitively, in order to collect multiple replicas for a DC-point, we are more interested in the replicators with closer Euclidean distance[2] to the corresponding DC-point. Hence, we define the notions of *kN-replicator set* and *RkN-replicator set*.

[2] Other distance metrics, such as network distance, can be incorporated as well.



(a) kN-replicator set for DC-point P_1 **(b)** RkN-replicator set for participant U_1

Fig. 3 Illustrating examples of kN-replicator and RkN-replicator sets in $C(P, U)$ with $k = 2$

Definition 4 (*kN-replicator set*) Given a campaign $C(P, U) \in R^2$, with P as the set of DC-points, and U as the set of participants, let $P_i \in P$. We refer to *k-nearest (kN) replicator set* $R_i \subset U$ of point P_i as a set of k replicators of P_i (i.e., $|R_i| = k$), of which every replicator in R_i corresponds to one of the k closest participants of P_i.

Figure 3 illustrates an example of a campaign, where participants are represented with squares and DC-points are shown with circles. In Fig. 3a, the 2N-replicator set for P_1 (i.e., $k = 2$) is depicted, where the elements of R_1 are shown with hollow squares ($R_1 = \{U_1, U_4\}$).

Definition 5 (*RkN-replicator set*) Given a participant U_i, we refer to *reverse k-nearest (RkN) replicator set* $R_i^{-1} \subset P$ of participant U_i as a set of all DC-points to which U_i is one of their k closest replicators.

Figure 3b depicts the R2N-replicator set (i.e., $k = 2$) for U_1, where the elements of R_1^{-1} are shown with hollow circles ($R_1^{-1} = \{P_1, P_4\}$).

Definition 6 (*aRknR problem*) Given a campaign $C(P, U) \in R^2$ and a trust value k, with P as the set of DC-points, and U as the set of participants, the problem is to find the *RkN*-replicator set of every participant. We refer to this problem as *all reverse k-nearest replicator (aRknR)* problem.

The aRknR problem can be restated as a special case of the bichromatic reverse k-nearest neighbor problem, in which the reverse k-nearest neighbor of *all* participants should be retrieved. Therefore, since bRkNN should be solved for every participant, the problem is analogous to solving kNN for every DC-point, which is a less complex problem. Therefore, a straightforward solution is that each participant sends his location to the server. The server then computes the k closest participants of every DC-point (i.e., *kN*-replicator set), inverts the result and sends to every participant his bRkNN result (i.e., *RkN*-replicator set). Note that the queries are issued from the participants. Therefore, from the view of the participant, the bRkNN problem should be solved. However, due to the nature of the aRknR problem , since all participants query the server, the server can solve the aRknR by solving kNN for every DC-point. Consider the example of Fig. 3 for $k = 2$. The solution to aRknR problem is shown in Table 1, which shows the *R2N*-replicator set of every participant.

However, in many scenarios, the server is not trusted, and therefore, a participant may not be willing to reveal his identity to the server. Even if the participant hides his identity from the server (i.e., only reveals his location), due to the strong correlation between people and their movements, a participant can still be identified by his location. Thus, we formally define our privacy attack.

Table 1 *R2N*-replicator set of set U

Participant	*R2N*-replicator set
U_1	$\{P_1, P_4\}$
U_2	$\{P_3, P_7\}$
U_3	$\{P_3, P_4, P_5\}$
U_4	$\{P_1, P_2, P_5, P_6\}$
U_5	$\{P_2, P_6\}$
U_6	$\{P_8, P_9\}$
U_7	$\{P_8, P_9\}$
U_8	$\{P_7\}$

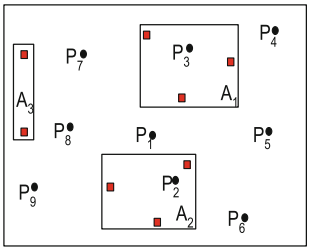


Fig. 4 Illustrating an example of PaRknR problem

Definition 7 (*Location-based attack*) A *location-based attack* is to identify a query issuer by associating the query to the query location (i.e., location from which the query is issued).

To defend against the location-based attack, we cannot reduce aRknR to multiple kNNs in server. Hence, we need to solve the aRknR problem privately to ensure both trust and privacy.

Definition 8 (*Problem definition*) The *private all reverse k-nearest replicator (PaRknR)* problem is a variation of the aRknR problem (Definition 6), in which the goal is to protect participants' identity from location-based attacks.

In order to solve the PaRknR problem, the server should compute the k closest replicators for every DC-point, for which it needs the participants locations. However, participants cannot share their locations with the untrustworthy server. Thus, instead of sending their exact location, participants follow the PiRi approach (Sect. 3.1) to blur their location in a cloaked area among $m-1$ other participants, from which a subset of them (i.e., by utilizing the voting mechanism) are selected as representatives to send cloaked regions to the server. Note that we cannot directly apply the PiRi approach to solve the PaRknR problem, since in the PiRi approach, we only assign one participant to each DC-point. The direct adaptation of PiRi requires the order-k Voronoi cell computation, which is computationally expensive. Thus, we only utilize PiRi for preserving privacy. In the following, we formally define the notion of m-anonymity [23].

Definition 9 (*m-ASR*) Given a set of participants U, we refer to the set $S \in U$ as an *m-anonymized set* where $|S| = m$, and for any participant $U_i \in S$, U_i's location is blurred in a *cloaked area* that encloses the set S. We refer to this area as *m-anonymized spatial region (m-ASR or ASR)*.

Figure 4 illustrates a set of ASRs, each containing a set of participants. For example, participants U_1, U_2, U_3 form an ASR A_1 with $m = 3$, whereas U_7 and U_8 form an ASR A_3 with $m = 2$.

With these definitions, we reformulate our problem as follows.

Definition 10 (*Problem definition*) Given a set of participants U, and a set of data collection points P, let A be the set of ASRs sent to the server, where every participant U_i is cloaked in at least one ASR of the set A. The problem is to find the *RkN*-replicator set of all participants.

Figure 4 depicts an example of PaRknR problem, in which participants of Fig. 3 form groups of different sizes (i.e., m). Thereafter, each group sends the corresponding ASR to the server. The goal is to find the *RkN*-replicator set of every participant.

3.3 System model

In this section, we first describe our privacy threat model and then discuss our system architecture which consists of two entities, participants and the PS server.

Our assumption is that the server trusts the majority of the participants (i.e., the data generated by majority of the participants is correct). Moreover, participants trust other peers to share their location information. However, they do not trust the PS server. Moreover, the server (or any adversary), if needed, can obtain the locations of all participants [14]. The reason is that participants often issue their queries from the same locations (office, home), which can be identified through physical observation, triangulation, etc. In general, since it is difficult to model the exact knowledge available to the server, this is a necessary assumption to guarantee that the privacy-preserving technique is secure under the most pessimistic scenario. That is, even though the participants' locations might be known to the server, it should not pose a threat (i.e., location-based attack) to the system if the system can successfully disassociate the queries from their locations. Moreover, we perform the assignment process for a given snapshot of participants' locations. That is, we assume that the participants do not move during the assignment process. This assignment process is a one-time operation that can be performed offline. This is intuitive, since participants usually plan their paths from their residential location (e.g., home, office) before starting their movement. Moreover, participants are the current active users of the system willing to participate in the process. The server (or any adversary) has the location information of all the DC-points, and the value of the campaign-defined k. The adversary is also aware of the anonymization technique which is used by the participants. However, each participant determines his own privacy level (i.e., m), which is only available to himself. That is, revealing his privacy level may result in privacy leak [14]. In order to guarantee the pseudonymity of participants' location information, each query is assigned with a unique pseudonymous identity, which is totally unrelated to the participants' personal identity. Finally, we make no guarantees if the sensed data contain any sensitive information (e.g., a photo containing someone's home).

Our PS server which contains the list of DC-points is equipped with a privacy-aware query processor, which processes the queries issued by the participants. Each participant can determine his privacy level, by specifying m, which determines the m-anonymity. Each participant is equipped with two wireless network interface cards. One is dedicated to the communication with the PS server through a base station or wireless modem. The other one is dedicated to the P2P communication among the peers through a wireless LAN, e.g., Bluetooth or IEEE 802.11. Also, each participant is equipped with a positioning device, e.g., GPS, which can determine its current location.

4 TAPAS

In order to solve the PaRknR problem, all of our approaches follow a filter and refinement technique, where the filtering step is performed at the server side, and the refinement step is performed at the participant-side. In the filtering step, since the server receives a set of ASRs, the idea is to prune a subset of DC-points that cannot be in the *RkN*-replicator set of the participants of a given ASR. Thereafter, the filtered results are sent to the participants, where the goal is to exploit some local information to refine the retrieved result. The challenge here is twofold. First, the server receives a set of ASRs instead of the exact locations of participants. Thus, it can only compute the *RkN*-replicator set for the ASRs rather than the participants. Second, in order to refine the results, a global knowledge of participants' locations is required. However, a participant only has limited knowledge about his local peers. This results in approximate answer. In the following, we propose three solutions to this problem. Our first approach is based on a limited pruning technique to reduce the uncertainty. Our second approach improves the first approach by enforcing a realistic assumption that results in a better pruning. Finally, our third approach applies some heuristics to achieve more accurate approximation.

4.1 Limited pruning technique (LPT)

In this approach, participants first communicate locally among themselves and blur their locations among $m-1$ other participants [24]. Thereafter, by employing the voting mechanism in [24], a set of representative participants are selected to send their ASRs to the server[3]. The server receives a set of ASRs, of which every ASR is associated with a different anonymity level m. This value m is dependent on the privacy requirement of the participants inside the corresponding ASR. Due to the unavailability of the participants' locations, the server computes the *kN*-replicator set of every DC-point with the given ASRs during the filtering step. Clearly, the server needs to explore at least k closest ASRs as a lower bound to assure that the k closest participants reside in the result set. The reason is that even though an ASR contains m participants, the server does not know the value m for each ASR. Considering the worst case when for all ASRs, we have $m = 1$, *kN*-replicator set of a DC-point is located in at least k closest ASRs. Once the set of candidate ASRs, which include the exact query answer for each DC-point, are retrieved (termed *kN*-ASR), the server inverts the result to find the *RkN*-ASR set for every ASR. These are all the DC-points which potentially have the participants in the given ASR as part of their *kN*-replicator set. Since we used a lower bound to find the *kN*-replicator set of every DC-point, the *RkN*-ASR set of every ASR contains the exact answer and possibly a set of false hits. This means that there is no DC-point in the *RkN*-replicator set of a participant which is not in the *RkN*-ASR set of its corresponding ASR. Section 4.1.1 explains the filtering step in more detail.

Once the result of the filtering step is sent to every representative participant, in order to prune the false hits, the representative refines the result by verifying the *kN*-replicator set of every DC-point in the result set. In order to verify the correctness of the result, the representative should have the location of all participants. However, the representative participant only has the location of his local peers and therefore prunes a subset of the false hits. Finally, the representative sends the corresponding partially refined result to each of his local peers. This indicates that every participant is assigned to a set of DC-points so that for each DC-point,

[3] Note that for $k = 1$, participants also compute their Voronoi cells during their local communication, and send their range queries along with their ASRs to the server. Thus, the assignment is performed once the server simply returns the range query result of every representative participant.

PkNNP Algorithm (set of ASRs A, DC-point p, k)
01. for each ASR $A_j \in A$
02. let $maxdist_p^{A_j}$ = distance from p to farthest point on A_j;
03. let A_{sorted} = list of all ASRs in A sorted by their $maxdist_p$ in an increased order;
04. let $c = 1$;
05. let $Res = \{\}$;
06. while ($c \leq k$)
07. let $A_{sorted}[c]$ = the ASR with cth smallest $maxdist_p$;
08. let Res_c = all the ASRs in radius $maxdist_p^{A_{sorted}[c]}$ from p;
09. add Res_c to Res;
10. $c = c + 1$;
11. return Res;

Fig. 5 Public *kNN* query over private data algorithm

at least k replicas will be collected, thus satisfying the k level of trust for the campaign. We explain the refinement step with more detail in Sect. 4.1.2.

4.1.1 Filtering step

The filtering step starts by computing the *kN*-ASR set of every DC-point, which can be interpreted as a public *kNN* query over private data. The reason is that from the server point of view, the query point (i.e., DC-point) is public, whereas the data points (i.e., participants) are private and are represented with a set of ASRs. To solve this problem for $k = 1$, we can use the proposed approach in [4]. We extend this approach for our filtering case where $k > 1$.

The pseudo-code of this algorithm (i.e., $PkNNP$ Algorithm) is shown in Fig. 5. We explain the details of this algorithm with the example of Fig. 4 with $k = 2$. The $PkNNP$ algorithm for a given DC-point p works in an incremental fashion by first finding the closest neighbor, and incrementally expanding the search until the k closest neighbors are found (lines 6–10). In order to compute the nearest participant to p, as an upper bound, we assume that the location of the participant is in the farthest corner of the given ASR from p. In other words, the distance between a DC-point p and an ASR A_j is the distance from p to the farthest corner of A_j denoted by $maxdist_p^{A_j}$ (line 2). Figure 6 depicts A_2 as the closest ASR, and the distance between p and A_2 is represented by a dashed line. Note that finding the closest ASR to p with the distance of $maxdist_p^{A_{\min}}$ does not guarantee that the closest participant is found. The reason is that $maxdist_p^{A_{\min}}$ is only an upper bound, and not every participant is located in the farthest corner. To address this, once the closest ASR is found, a range query with a radius $maxdist_p^{A_{\min}}$ should be computed to retrieve all the possible results. In the example of Fig. 6, a range query with radius $maxdist_p^{A_2}$ is performed, which returns A_1. This indicates that the nearest participant is located in either A_1 or A_2. The next step is to repeat the iteration for $k = 2$. In this step, we find the second closest ASR (i.e., A_1) and perform a range query with the radius $maxdist_p^{A_1}$. This will add A_3 to the result set. At this point, the algorithm terminates and returns A_1, A_2, and A_3 as the result set. Table 2 depicts the *2N*-ASR set for every DC-point.

After the computation of *kN*-ASR set for every DC-point, the server inverts the result to find the *RkN*-ASR set for every ASR. Finally, each *RkN*-ASR set of an ASR is sent to its corresponding owner. Table 3 shows the result of this step.

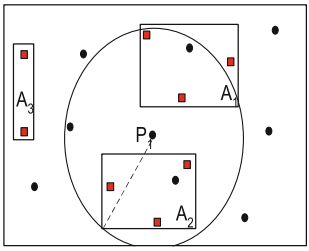


Fig. 6 Illustrating the first iteration in *PkNNP* algorithm

Table 2 *2N*-ASR set for set P in LPT

DC-point	*kN*-ASR
P_1	$\{A_1,A_2,A_3\}$
P_2	$\{A_1,A_2,A_3\}$
P_3	$\{A_1,A_2,A_3\}$
P_4	$\{A_1,A_2,A_3\}$
P_5	$\{A_1,A_2\}$
P_6	$\{A_1,A_2\}$
P_7	$\{A_1,A_2,A_3\}$
P_8	$\{A_1,A_2,A_3\}$
P_9	$\{A_1,A_2,A_3\}$

Table 3 *R2N*-ASR set for set P in LPT

ASR	$RkN-ASR$
A_1	$\{P_1,\ldots,P_9\}$
A_2	$\{P_1,\ldots,P_9\}$
A_3	$\{P_1,P_2,P_3,P_4,P_7,P_8,P_9\}$

4.1.2 Refinement step

Once the *RkN*-ASR set of a given ASR is sent to its corresponding representative participant, the representative performs the refinement step before sending the result to each of the local peers (i.e., participants inside the ASR). The goal of the refinement is to prune the extra DC-points from the *RkN*-replicator set of each of the peers. To do so, the representative validates the *kN*-replicator set of every DC-point in the result. However, since the representative only has the location of his local peers, the *kN*-replicator set of the DC-points can only be verified with respect to the participants inside the given ASR. Table 4 depicts the *kN*-replicator set of every DC-point with respect to each of the ASRs. For example, the *kN*-replicator set of P_1 with respect to A_1 (denoted by $kN\text{-replicator}_{P_1}^{A_1}$) includes U_1 and U_2. The reason is that among the participants inside A_1 (i.e., U_1,U_2, and U_3), U_1 and U_2 are closer to P_1. Therefore, the refinement step eliminates P_1 from the *RkN*-replicator set of U_3.

After the validation step, the algorithm inverts the result, and sends the corresponding *RkN*-replicator set to every participant in the ASR. Table 5 shows the final result sent to every participant. Every participant receives the exact result, shown in bold, along with a set of false positives. This means that all DC-points meet the *kN*-replica's requirement (i.e., LPT successfully finds k closest participants for every DC-point to collect k replicas). However,

Table 4 *2N*-replicator^A for set P in LPT

DC-point	*2N*-r^{A_1}	*2N*-r^{A_2}	*2N*-r^{A_3}
P_1	$\{U_1,U_2\}$	$\{U_4,U_6\}$	$\{U_7,U_8\}$
P_2	$\{U_1,U_3\}$	$\{U_4,U_5\}$	$\{U_7,U_8\}$
P_3	$\{U_2,U_3\}$	$\{U_4,U_6\}$	$\{U_7,U_8\}$
P_4	$\{U_1,U_3\}$	$\{U_4,U_6\}$	$\{U_7,U_8\}$
P_5	$\{U_1,U_3\}$	$\{U_4,U_5\}$	N/A
P_6	$\{U_1,U_3\}$	$\{U_4,U_5\}$	N/A
P_7	$\{U_1,U_2\}$	$\{U_4,U_6\}$	$\{U_7,U_8\}$
P_8	$\{U_1,U_2\}$	$\{U_4,U_6\}$	$\{U_7,U_8\}$
P_9	$\{U_1,U_2\}$	$\{U_5,U_6\}$	$\{U_7,U_8\}$

Table 5 *R2N*-replicator set for set U in LPT

Participant	*R2N*-replicator	*WC* (%)
U_1	$\{\mathbf{P_1},P_2,\mathbf{P_4},P_5,P_6,P_7,P_8,P_9\}$	75
U_2	$\{P_1,\mathbf{P_3},\mathbf{P_7},P_8,P_9\}$	60
U_3	$\{P_2,\mathbf{P_3},\mathbf{P_4},\mathbf{P_5},P_6\}$	40
U_4	$\{\mathbf{P_1},\mathbf{P_2},P_3,P_4,\mathbf{P_5},\mathbf{P_6},P_7,P_8\}$	50
U_5	$\{\mathbf{P_2},P_5,\mathbf{P_6},P_9\}$	50
U_6	$\{P_1,P_3,P_4,P_7,\mathbf{P_8},\mathbf{P_9}\}$	67
U_7	$\{P_1,P_2,P_3,P_4,P_7,\mathbf{P_8},\mathbf{P_9}\}$	71
U_8	$\{P_1,P_2,P_3,P_4,\mathbf{P_7},P_8,P_9\}$	86

due to privacy concerns, every participant is assigned to more DC-points than expected. Below, we define the notion of *wasteful collection (WC)*.

Definition 11 (*Wasteful collection*) Given a participant U_i, we refer to the percentage of extra DC-point assignments to U_i as the *wasteful collection* of U_i denoted by WC_i.

$$WC_i = \frac{|false\ positives_i|}{|true\ positives_i| + |false\ positives_i|} \times 100 \tag{1}$$

The term of wasteful collection is defined per individual participant. We compute the average of the wasteful collections for all participants, denoted by *WC*, as the overall wasteful collection of the system. It is evident that larger values of *WC* result in more replicas per DC-point. In Table 5, the wasteful collection for every participant is calculated. For example, the wasteful collection for U_1 is calculated as follows: $WC_1 = (6/(6+2)) * 100 = 75\,\%$. Therefore, the average *WC* is 62 %, which means that on average every participant is assigned to 62 % extra DC-points. Our goal is to improve the technique by minimizing this extra assignment.

4.1.3 LPT completeness

The following theorem proves the completeness of LPT.

Theorem 1 *The LPT approach is complete (i.e., no missing data).*

Proof 1 In order to prove the completeness, we should prove that the *PkNNP* algorithm retrieves all the ASRs which contain the k closest participants to a given DC-point p.

We prove this by contradiction. Assume the kth closest participant is outside the radius $maxdist_p^{A_{sorted}[k]}$. This means that all ASRs in the given radius contain at most $k-1$ participants. However, there are at least k ASRs in the given radius. Moreover, every ASR contains at least one participant. Therefore, at least k participants exist in the radius $maxdist_p^{A_{sorted}[k]}$ from p, which contradicts our prior assumption. □

4.2 Bounded anonymity level (BAL)

Our LPT approach does not make any assumption about the anonymity level m of any ASR. This means m can have any value dependent on the privacy requirement of the participants in the given ASR. Therefore, in order to guarantee that the k closest participants for every DC-point are retrieved, the server needs to explore at least k closest ASRs considering the worst case where $m = 1$. However, due to privacy concerns, cloaking usually occurs among more than one person. In this case, if the value of m becomes available to the server, the server can find the k closest participants by exploring less number of ASRs. Consequently, the number of extra assignments to every participant (i.e., wasteful collection) would drop. However, knowing the anonymity level of a given ASR results in privacy leak [14]. Instead, the server can enforce a constraint, where the anonymity level of any ASR should stay beyond a certain threshold. In other words, the server defines a system value, denoted by $m_{\min}$, where the anonymity level of any ASR should be larger than $m_{\min}$. However, this only works if the ASRs do not overlap, in which every ASR contains at least $m_{\min}$ distinct users. In the case of an overlap, a participant might be in more than one ASR. Thus, every ASR should have at least $m_{\min}$ participants who voted for the given ASR, namely *voting participants*, to ensure enough number of distinct participants (i.e., $m_{\min}$) in every ASR. The reason is that every participant votes for only one ASR [24]. For example, if a participant is inside two overlapping ASRs, he has only voted for one of the two and therefore will be counted toward only that ASR. Thus, this constraint can be enforced when every representative agrees upon it (i.e., its ASR contains at least $m_{\min}$ voting participants).

Our second approach, referred to as *bounded anonymity level (BAL)*, is based on this assumption. Enforcing the minimum anonymity level constraint has few advantages. First, the server is still unaware of the anonymity level of any ASR. Second, for $m_{\min} > 1$, less number of ASRs might get explored, and therefore, less false hits in the result. Third, LPT is a special case of BAL, where $m_{\min} = 1$. In the following, we explain the details of BAL approach.

4.2.1 Filtering step

Similar to the LPT approach, the filtering step in BAL approach starts with computing the *kN*-ASR set of every DC-point. This means that an incremental approach is used by first finding the closest neighbor, and expanding the search until k closest ones are found. The difference is that here $m_{\min}$ is enforced to every ASR. Consequently, once an ASR is explored, the server knows that at the worst case $m_{\min}$ participants reside in the given ASR. Thus, the algorithm might stop at an earlier iteration. In our example of Fig. 4, considering $m_{\min} = 2$, the algorithm finds A_2 as the closest ASR to P_1. The server knows that A_2 has anonymity level of at least 2. However, the algorithm does not stop at this point, since a participant in a distance of $maxdist_{P_1}^{A_2}$ in another ASR might be closer to P_1. Thus, the algorithm performs a range query with a radius of $maxdist_{P_1}^{A_2}$ and adds any intersecting ASR to the result set (i.e., A_1). At this point, the algorithm stops, since the two closest participants cannot reside

Table 6 *2N*-ASR set for set P in BAL

DC-point	*kN*-ASR
P_1	$\{A_1,A_2\}$
P_2	$\{A_1,A_2\}$
P_3	$\{A_1\}$
P_4	$\{A_1,A_2\}$
P_5	$\{A_1,A_2\}$
P_6	$\{A_1,A_2\}$
P_7	$\{A_1,A_2,A_3\}$
P_8	$\{A_1,A_2,A_3\}$
P_9	$\{A_1,A_2,A_3\}$

Table 7 *R2N*-replicator set for set U in BAL

Participant	*R2N*-replicator	*WC* (%)
U_1	$\{\mathbf{P_1},P_2,\mathbf{P_4},P_5,P_6,P_7,P_8,P_9\}$	75
U_2	$\{P_1,\mathbf{P_3},\mathbf{P_7},P_8,P_9\}$	60
U_3	$\{P_2,\mathbf{P_3},\mathbf{P_4},\mathbf{P_5},P_6\}$	40
U_4	$\{\mathbf{P_1},\mathbf{P_2},P_4,\mathbf{P_5},\mathbf{P_6},P_7,P_8\}$	43
U_5	$\{\mathbf{P_2},P_5,\mathbf{P_6},P_9\}$	50
U_6	$\{P_1,P_4,P_7,\mathbf{P_8},\mathbf{P_9}\}$	60
U_7	$\{P_7,\mathbf{P_8},\mathbf{P_9}\}$	33
U_8	$\{\mathbf{P_7},P_8,P_9\}$	67

outside the radius $maxdist_{P_1}^{A_2}$ from P_1. Finally, the algorithm returns A_1 and A_2 as the result. Table 6 depicts the *2N*-ASR sets of all DC-points using the BAL approach. As the table shows, less number of ASRs are explored comparing to LPT approach.

Once the *kN*-ASR set of every DC-point is computed, the server inverts the result and sends the corresponding *RkN*-ASR sets to the owners for the refinement process.

4.2.2 Refinement step

The refinement process is exactly similar to the LPT approach. After receiving the *RkN*-ASR set, the representative participant validates the *kN*-replicator set of every DC-point in the result with respect to all participants in the given ASR. Thereafter, the representative inverts the result and sends the corresponding *RkN*-replicator set to every participant in the given ASR. Table 7 depicts the final result sent to every participant along with the WC percentage. As also expected, we see a slight decrease in the extra DC-point assignment to the participants. The average WC is reduced to 53 %.

4.2.3 BAL completeness

In the following, we prove the completeness of BAL.

Theorem 2 *The BAL approach is complete.*

Proof 2 The proof is similar to that of Theorem 1, and therefore is omitted. □

4.3 Heuristics-based bounded anonymity level (HBAL)

In both LPT and BAL, the refinement step is performed based on the local knowledge that each representative participant has about his local peers. Therefore, the validation of *kN*-replicator set for every DC-point is only based on the local participants in the given ASR. This results in a limited pruning capability. In order to improve this, the representative participants require some knowledge about other non-local participants. However, the server does not have the location information of other participants. Instead, it can share some information about their ASRs. Therefore, we can employ a set of heuristics to expand the pruning with the extra information sent to the representative participants. We refer to this approach as *Heuristics-based Bounded Anonymity Level (HBAL)*. We explain more details in the following sections.

4.3.1 Filtering step

The filtering step of HBAL is similar to that of the BAL approach in that the server computes the *kN*-ASR set of all DC-points. Next, the server inverts the result and sends the *RkN*-ASR set of every ASR to its corresponding representative participant. However, for every DC-point p in the *RkN*-ASR set of a given ASR, the server also sends the *kN*-ASR set of p to the corresponding ASR. This extra knowledge helps the refinement step with more pruning. Following our example of Fig. 4, once Table 6 is generated, the server not only sends $P_1, \ldots, P_9$ to A_1, it also sends their *kN*-ASR sets. This means that the server sends the *kN*-ASR set of P_1 (i.e., A_1 and A_2) to both A_1 and A_2.

4.3.2 Refinement step

Once the representative participant receives the *RkN*-ASR set, it refines the result in two phases. The first phase is similar to the refinement step of both LPT and BAL, where the *kN*-replicator sets of the DC-points are validated against all local participants in the given ASR. In the second phase, the results of the previous phase are examined against a set of ASRs of non-locals, which are contained in the *kN*-ASR set of the corresponding DC-point. For example, by validating the *2N*-replicator set of P_4 with respect to A_1, we retrieved U_1 and U_3 in the first phase. In the next phase, since $A_2 \in 2N\text{-ASR}_{P_4}$ (see Table 6), we employ some heuristics to validate the first-phase result against A_2. The question is how to employ the non-local ASRs to do the refinement. The following lemmas answer this question.

Lemma 1 *Given a DC-point p, and an ASR A_i, let kN-replicator$_p^{A_i}$ be the kN-replicator set of p with respect to elements in A_i. Also, let u $\in$kN-replicator$_p^{A_i}$. We say u belongs to kN-replicator$_p$ if the distance between p and u is smaller than the distance between p and the closest point on any of the non-local ASRs in kN-ASR set of p (i.e., dist(p,u) $<$ mindist$_p^{A_j} \forall A_{j \neq i} \in$kN-ASR$_p$).*

Based on Lemma 1, we can guarantee that a participant is in *kN*-replicator set of a DC-point, if their distance is smaller than the minimum distance to any other ASR. Following our example of validating *2N*-replicator set of P_4, we retrieved U_1 and U_3 in the first phase. In the next phase, we have to validate them against A_2. According to the above lemma, we can guarantee that U_1 and U_3 are the *2N*-replicators of P_4, since no participant in A_2 is in a closer distance to P_4.

Table 8 *R2N*-replicator for set U in HBAL

Participant	*R2N*-replicator	*WC* (%)
U_1	{**P1**,P_2,**P4**,P_5,P_6,P_7}	67
U_2	{P_1,**P3**,**P7**}	33
U_3	{**P3**,**P4**,**P5**,P_6}	25
U_4	{**P1**,**P2**,**P5**,**P6**}	0
U_5	{**P2**,P_5,**P6**,P_9}	50
U_6	{P_1,**P8**,**P9**}	33
U_7	{P_7,**P8**,**P9**}	33
U_8	{**P7**,P_8,P_9}	67

Lemma 2 *Given a DC-point p, and an ASR A_i, let kN-replicator$_p^{A_i}$ be the kN-replicator set of p with respect to elements in A_i. Also let u be the jth nearest replicator in kN-replicator$_p^{A_i}$. We say u $\notin$kN-replicator$_p$ if the distance between p and u is larger than the distance between p and the farthest point on n number of the ASRs in kN-ASR set of p, where* $(n \times m_{\min}) + (j-1) \geq k$.

This indicates that we can prune a participant from the *kN*-replicator set of a DC-point, if their distance is larger than the maximum distance to a set of ASRs, in which the total number of possible results exceeds k. In our following example of Fig. 4, by validating the *2N*-replicator set of P_8 with respect to A_1, we retrieved U_1 and U_2 in the first phase. In the second phase, we should validate them against A_2 and A_3 (see Table 6). As the figure depicts, the distance between P_8 and U_2 is larger than $maxdist_{P_8}^{A_3}$. Since A_3 contains at least two participants, the k requirement is satisfied. This indicates that U_2 cannot be any of the two closest participants to P_8. Thus, we can prune U_2 from the result set. Similarly, U_1 is also pruned from the result set. Consequently, P_8 is no longer in the *R2N*-replicator set of any of the participants in A_1. In other words, none of the participants in A_1 are assigned to collect replicas at P_8, since according to Lemma 2, they cannot be the two closest participants to P_8.

Table 8 depicts the final result for each participant in the HBAL approach. The table shows a drop in the number of false hits, and thus the percentage of *WC*. The average percentage of *WC* is reduced to 38 %.

4.3.3 HBAL completeness

The following theorem proves the completeness of HBAL.

Theorem 3 *The HBAL approach is complete.*

Proof 3 The proof is trivial and therefore is omitted.

5 Performance evaluation

We conducted several simulation-based experiments to evaluate the performance of our proposed approaches: LPT, BAL, and HBAL. Below, first we discuss our experimental methodology. Next, we present our experimental results.

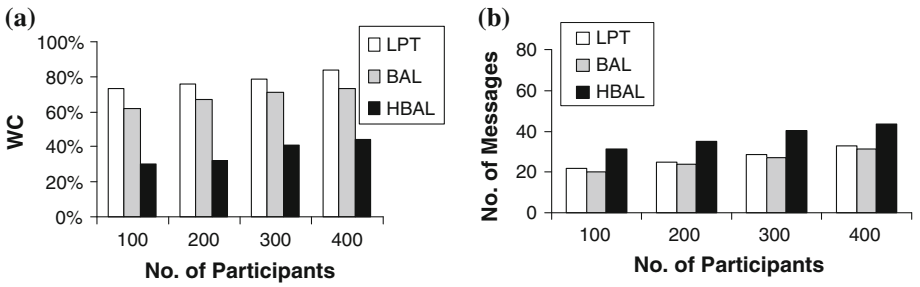


Fig. 7 Scalability

5.1 Experimental methodology

We performed three sets of experiments. With the first set of experiments, we evaluated the scalability of our proposed approaches. For the rest of the experiments, we evaluated the impact of the campaign's trust level and the participant's privacy requirement on our approaches. With these experiments, we used two performance measures: (1) *WC*, and (2) communication cost, in which the communication cost is measured in terms of the number of messages incurred by our algorithms for each representative participant[4].

We conducted our experiments with the objective of collecting a set of photos from 500 locations in part of the Los Angeles county. These DC-points were randomly selected. Moreover, our participants dataset includes random generation of 400 users' location. Since usually a limited number of users participate in a PS campaign, we set the default number of participants to 200 and vary it between 100 to 400. Moreover, we vary the trust level of the campaign between 2 to 5, with 3 as the default value (i.e., $k = 3$). We set the transmission range to 250 m [5]. At the transport layer, we set the MTU (Maximum Transmission Unit) to be 500 bytes. The degree of anonymity (m) for each participant varies between 5 to 20, with 5 as the default value. Moreover, for both BAL and HBAL, we set $m_{\min}$ to 2. We set $m_{\min}$ to a small value for two reasons: (1) privacy and (2) to ensure that all ASRs satisfy the $m_{\min}$ requirement, which was also verified by our experiments. Finally, for each of our experiments, we ran 500 cases and reported the average of the results.

5.2 Scalability

In the first set of experiments, we evaluated the scalability of our approach by varying the number of participants from 100 to 400. As Fig. 7a depicts, we see a slight increase in the *WC* percentage for all the three approaches as the number of participants grows. The reason is that larger number of participants results in denser ASRs, and therefore, more overlap between ASRs. In other words, since more ASRs are returned as the *kN*-ASR set of every DC-point, the number of false hits increases. In general, we see LPT with the highest percentage of *WC* in all cases. Moreover, BAL only has a slight improvement over LPT. The reason is that we chose a low value for $m_{\min}$ (i.e., $m_{\min} = 2$). Choosing higher values would result in more

[4] In this paper we have not defined a trust metric to measure the amount of trust we achieve by redundant data collection. Note that defining such a trust metric is non-trivial in a privacy-aware PS and is the focus of our future work.

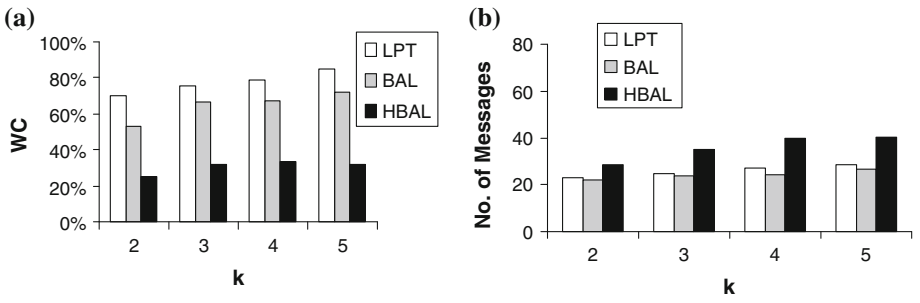


Fig. 8 Effect of trust level

pruning during the filtering step, thus improving the *WC* percentage of BAL. Finally, we see HBAL with the least percentage of *WC* (up to 2.5 times better than LPT).

Figure 7b shows the impact of varying the number of participants on the number of messages. As the figure shows, the number of messages increases in all cases. In a denser network, more communication is required among the peers to perform their queries. It is worth mentioning that the dominant communication overhead (on average 70 %) in all the three approaches is due to the P2P communication for preserving the privacy, which we used the PiRi approach[5]. Moreover, we see that with HBAL, due to the extra information sent to the representatives for pruning during the refinement, the communication cost is higher than both LPT and BAL. However, the extra cost is only 30 % higher than LPT in the worst case. Another interesting observation is that BAL has less communication cost as compared to LPT in all cases. The reason is that with the BAL approach the extra pruning is performed at the server side, resulting in less information to be sent to the representatives.

5.3 Effect of trust level

In the next set of experiments, we evaluated the performance of our approaches with respect to the campaign's trust level varied from 2 to 5. Figure 8a illustrates an increase in the *WC* percentage as k grows in most cases. The reason is that as k increases, less number of participants are pruned during the local validation phase of the refinement step. However, the increase is less significant for HBAL due to the extra pruning in the refinement step. Moreover, with an increase in k, the kN-ASR set of every DC-point becomes larger; thus increasing the communication cost in all cases (Fig. 8b). Similar to the previous experiments, HBAL acts the best in terms of improving the *WC* percentage (up to 2.8 times better than LPT), while the extra communication cost stays between 15 and 30 % of that of LPT.

5.4 Effect of privacy requirement

In our final set of experiments, we measured the performance of our approaches with respect to increasing the privacy requirement (m) from 5 to 20. As Fig. 9a shows, with an increase in m, the percentage of *WC* reduces in all cases. The reason is that for larger values of m, each ASR contains more number of participants. Consequently, kN-replicator set of a DC-point exists in a less number of ASRs, which results in more pruning during the validation phase of

[5] To the best of our knowledge, PiRi is the only existing approach for a privacy-aware assignment of DC-points to the participants. However, any other privacy-preserving technique for PS systems is also applicable.

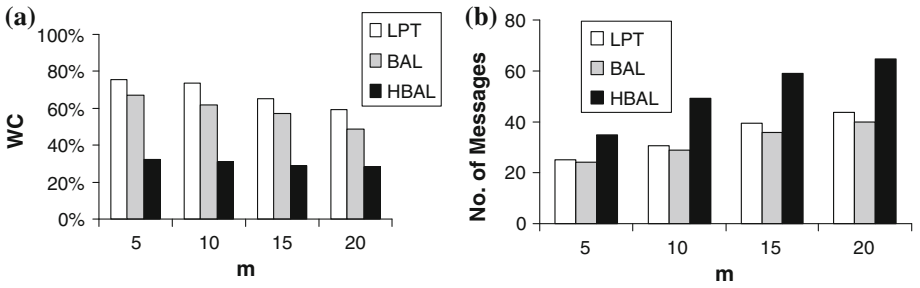


Fig. 9 Effect of privacy requirement

the refinement step. Moreover, Fig. 9b shows the effect of varying m on the communication cost. The figure illustrates that the number of messages increases with an increase in m. This is because as m grows, more communication is required among the peers of a given ASR. Similarly, we see HBAL outperforming LPT and BAL in terms of the WC percentage, while the communication overhead is only slightly higher than those of the two.

5.5 Discussion

Our main observation from our experiments is that with an average of 30 % extra communication cost, HBAL can improve the approximation by up to 2.8 times over LPT and BAL. However, we argue that this communication cost is not a burden to the participants since this is only a one-time cost associated to assigning DC-points to the participants during the planning phase. Moreover, our experiments showed that with HBAL the approximation improves by increasing the privacy requirement (e.g., WC decreases to 22 % with $m = 20$). However, as the number of participants grows, we see an increase in the percentage of WC (e.g., WC increases to 40 % with 400 participants). Our experiments also showed that the dominant growth of WC is caused by increasing the number of participants. This shows that our proposed approach performs better in campaigns which have small number of participants with higher privacy requirements.

6 Conclusion and future work

In this paper, for the first time, we formalized the notion of the interplay between trust and privacy in PS as a private all reverse k nearest replicator (PaRknR) problem. Subsequently, we proposed TAPAS, a trustworthy privacy-aware framework that included three various solutions to the PaRknR problem, namely LPT, BAL, and HBAL. In our experiments, we demonstrated the overall efficiency of our approaches in preserving the privacy and trust in PS campaigns. Our main observation from our experiments is that with an average of 30 % extra communication cost, HBAL can improve the approximation by up to 2.8 times over LPT and BAL. As future work, we aim to extend the proposed approaches to more cost-effective and energy-efficient solutions. We also plan to propose efficient and privacy-aware approaches to handle the user mobility during the assignment.

Acknowledgments This research is supported in part by Award No. 2011-IJ-CX-K054 from National Institute of Justice, Office of Justice Programs, U.S. Department of Justice, as well as by NSF grants CNS-0831505 (CyberTrust) and IIS-1115153, the USC Integrated Media Systems Center (IMSC), and unrestricted cash and

equipment gifts from Google, Microsoft and Qualcomm. The opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect those of the Department of Justice and the National Science Foundation.

References

1. Adam NR, Worthmann JC (1989) Security-control methods for statistical databases: a comparative study. ACM Comput Surv 21(4):515–556
2. Agrawal R, Srikant R (2000) Privacy-preserving data mining. In: SIGMOD'00. ACM, Dallas, pp 439–450
3. Bamba B, Liu L, Pesti P, Wang T (2008) Supporting anonymous location queries in mobile environments with privacygrid. In: WWW'08. ACM, Beijing, pp 237–246
4. Chow C-Y, Mokbel MF, Aref WG (2009) Casper*: query processing for location services without compromising privacy. ACM TODS 34(4):24:1–24:48
5. Chow C-Y, Mokbel MF, Liu X (2006) A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In: GIS'06. ACM, Arlington, Virginia, pp 171–178
6. Chow C-Y, Mokbel MF, Liu X (2009) Spatial cloaking for anonymous location-based services in mobile peer-to-peer, environments. GeoInformatica '09 15:351–380
7. Cornelius C, Kapadia A, Kotz D, Peebles D, Shin M, Triandopoulos N (2008) Anonysense: privacy-aware people-centric sensing. In: MobiSys '08. ACM, Breckenridge, pp 211–224
8. CycleSense (2009) Center for embedded networked sensing (cens). http://urban.cens.ucla.edu/projects/
9. Dua A, Bulusu N, Feng W-C, Hu W (2009) Towards trustworthy participatory sensing. In: HotSec'09. USENIX Association, Berkeley, pp 8–8
10. Fung BCM, Wang K, Chen R, Yu PS (2010) Privacy-preserving data publishing: a survey of recent developments. ACM Comput Surv 42(4):14:1–14:53
11. Gedik B, Liu L (2008) Protecting location privacy with personalized k-anonymity: architecture and algorithms. IEEE TMC'08 7(1):1–18
12. Ghinita G, Kalnis P, Khoshgozaran A, Shahabi C, Tan K-L (2008) Private queries in location based services: anonymizers are not necessary. In: SIGMOD '08. ACM, Vancouver, pp 121–132
13. Ghinita G, Kalnis P, Skiadopoulos S (2007) Mobihide: a mobilea peer-to-peer system for anonymous location-based queries. In: SSTD'07. Springer, Boston, pp 221–238
14. Ghinita G, Zhao K, Papadias D, Kalnis P (2010) A reciprocal framework for spatial k-anonymity. Inf Syst 35:299–314
15. Gilbert P, Cox LP, Jung J, Wetherall D (2010) Toward trustworthy mobile sensing. In: HotMobile '10. ACM, Annapolis, pp 31–36
16. Gonzalez MC, Hidalgo CA, Barabasi A-L (2008) Understanding individual human mobility patterns. Nature 453(7196):779–782
17. Gummadi R, Balakrishnan H, Maniatis P, Ratnasamy S (2009) Not-a-bot: improving service availability in the face of botnet attacks. In:NSDI'09. USENIX Association, Boston, pp 307–320
18. Gupta M, Judge P, Ammar M (2003) A reputation system for peer-to-peer networks. In: NOSSDAV '03. ACM, Monterey, pp 144–152
19. Hengartner U (2007) Hiding location information from location-based services. In: MDM '07. IEEE Computer Society, pp 268–272
20. Hu L, Shahabi C (2010) Privacy assurance in mobile sensing networks: go beyond trusted servers. In: PerCom Workshops. IEEE, Mannheim, pp 613–619
21. Huang KL, Kanhere SS, Hu W (2009) Towards privacy-sensitive participatory sensing. In: PERCOM '09. IEEE, Galveston, pp 1–6
22. Hull B, Bychkovsky V, Zhang Y, Chen K, Goraczko M, Miu A, Shih E, Balakrishnan H, Madden S (2006) Cartel: a distributed mobile sensor computing system. In: SenSys '06. ACM, Boulder, pp 125–138
23. Kalnis P, Ghinita G, Mouratidis K, Papadias D (2007) Preventing location-based identity inference in anonymous spatial queries. IEEE TKDE'07 12(19):1719–1733
24. Kazemi L, Shahabi C (2011) A privacy-aware framework for participatory sensing. SIGKDD Explorations 13(1):43–51
25. Kazemi L, Shahabi C (2011) Towards preserving privacy in participatory sensing (short paper). In: PerCom'11. IEEE, Seattle
26. Khoshgozaran A, Shahabi C (2007) Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In: SSTD'07. Springer, Boston, pp 239–257
27. Khoshgozaran A, Shahabi C, Shirani-Mehr H (2011) Location privacy: going beyond k-anonymity, cloaking and anonymizers. Knowl Inf Syst 26(3):435–465

28. Ku W-S, Hu L, Shahabi C, Wang H (2009) Query integrity assurance of location-based services accessing outsourced spatial databases. In: SSTD '09. Springer, Aalborg, pp 80–97
29. Lenders V, Koukoumidis E, Zhang P, Martonosi M (2008) Location-based trust for mobile user-generated content: applications, challenges and implementations. In: HotMobile '08. ACM, Napa Valley, pp 60–64
30. Millenium (2008) Mobile millenium project. http://traffic.berkeley.edu/
31. Mohan P, Padmanabhan VN, Ramjee R (2008) Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In: SenSys'08. ACM, Raleigh, pp 323–336
32. Mokbel MF, Chow C-Y, Aref WG (2006) The new casper: query processing for location services without compromising privacy. In: VLDB'06. VLDB Endowment, Seoul, pp 763–774
33. Ooi BC, Liau CY, Tau K-L (2003) Managing trust in peer-to-peer systems using reputation-based techniques. In: WAIM'03. Springer, Berlin, pp 2–12
34. Puttaswamy KPN, Bhagwan R, Padmanabhan VN (2010) Anonygator: Privacy and integrity preserving data aggregation. In: Middleware. Springer, Bangalore, pp 85–106
35. Samarati P (2001) Protecting respondents' identities in microdata release. IEEE Trans Knowl Data Eng 13(6):1010–1027
36. Saroiu S, Wolman A (2010) I am a sensor, and i approve this message. In: HotMobile '10. ACM, Annapolis, pp 37–42
37. Shilton K, Burke J, Estrin D, Hansen M, Srivastava MB (2008) Participatory privacy in urban sensing. MODUS'08. St. Louis, Missouri, pp 1–7
38. Sion R (2005) Query execution assurance for outsourced databases. In: VLDB'05. VLDB Endowment, Trondheim, pp 601–612
39. Surowiecki J (2004) The wisdom of crowds: why the many are smarter than the few and how collective wisdom shapes business, economies, societies and nations. Knopf Doubleday Publishing Group, USA. ISBN 9780385503860
40. Sweeney L (2002) k-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl-Based Syst 10(5):557–570
41. Yang Y, Papadopoulos S, Papadias D, Kollios G (2008) Spatial outsourcing for location-based services. In: ICDE'08. IEEE, Cancun, pp 1082–1091
42. Yiu ML, Ghinita G, Jensen CS, Kalnis P (2009) Outsourcing search services on private spatial data. In: ICDE'09. IEEE, Shanghai, pp 1140–1143
43. Yiu ML, Ghinita G, Jensen CS, Kalnis P (2010) Enabling search services on outsourced private spatial data. VLDBJ'10 19(3):363–384

Author Biographies

Leyla Kazemi received a B.S. degree in Computer Engineering from Tehran PolyTechnic, Tehran, Iran, in 2005 and an M.S. degree in Computer Science from the University of Southern California, Los Angeles, in 2008. She is currently working toward her Ph.D. degree in Computer Science at the University of Southern California. Her research interests include geo-spatial databases, participatory sensing, and location privacy.

Cyrus Shahabi is a Professor of Computer Science and Electrical Engineering and the Director of the NSF's Integrated Media Systems Center (IMSC) at the University of Southern California. He was also the CTO and co-founder of a USC spin-off and an InQTel portfolio company, Geosemble Technologies, which was acquired in June 2012. He received his B.S. in Computer Engineering from Sharif University of Technology in 1989 and then his M.S. and Ph.D. Degrees in Computer Science from the University of Southern California in May 1993 and August 1996, respectively. He authored two books and more than two hundred research papers in the areas of databases, GIS and multimedia. He is currently on the editorial board of the VLDB Journal and IEEE Transactions on Knowledge and Data Engineering. Dr. Shahabi is a recipient of the ACM Distinguished Scientist award and the U.S. Presidential Early Career Awards for Scientists and Engineers (PECASE).