Distrib Parallel Databases (2015) 33:253–276
DOI 10.1007/s10619-014-7148-8

A privacy-enhancing model for location-based personalized recommendations

Jin Huang · Jianzhong Qi · Yabo Xu · Jian Chen

Published online: 9 April 2014
© Springer Science+Business Media New York 2014

Abstract To receive personalized recommendation, users of a location-based service (e.g., a Location-Based Social Network, LBSN) have to provide personal information and preferences to the location-based service. However, detailed personal information could be used to identify the users, and hence compromise user privacy. In this paper, we consider an untrusted third party recommendation service used by the location-based service that may attempt to identify the sender of a recommendation query from the query log or may publish the query log. To protect user identity, anonymization must be done "online" before a query reaches the recommendation service. This is different from the usual "offline" scenario where a trusted recommendation service will receive all unanonymized queries and the focus is to anonymize the collected query log. We propose the notion of online anonymity to formalize this online requirement. The challenge for providing online anonymity is dealing with unknown and dynamic

J. Huang
School of Computer, South China Normal University, Guangzhou, China

J. Huang
Shenzhen Engineering Laboratory for Mobile Internet Application
Middleware Technology, Shenzhen, China
e-mail: jinhuang@scnu.edu.cn

J. Qi
Department of Computing and Information Systems, University of Melbourne,
Parkville, VIC, Australia
e-mail: jianzhong.qi@unimelb.edu.au

Y. Xu (✉)
School of Software, Sun Yat-Sen University, Guangzhou, China
e-mail: xuyabo@mail.sysu.edu.cn

J. Chen
School of Software Engineering, South China University of Technology, Guangzhou, China
e-mail: ellachen@scut.edu.cn

location-based service users who can get online and offline at any time. We define this problem, discuss its implications and differences from the problems in the literature, and propose a solution. Our experimental study shows that it is feasible to achieve personalized recommendation while preserve user privacy.

Keywords Location-based services · Privacy · Personalization · Online anonymity

1 Introduction

The annual personalization surveys conducted by Choicestream[1] have consistently shown that consumers were interested in personalized recommendation service and contents. On one hand, personalized recommendation service offers user tailored services according to their personal preference, and hence, is far effective to meet users' need; on the other hand, personalized recommendation service entails gathering considerable amounts of personal information from its users, and hence, raises much of privacy concern [13].

Often, data collected by recommendation services, such as query logs, are excellent candidates for various data mining applications. However, the use of such data raises privacy concerns if the data is not made anonymous enough. An example is the release of AOL query logs (New York Times, Aug 9, 2006), where the searcher No. 4417749 was traced back to Ms Thelma Arnold. For another example, Google were ordered by a federal judge to turn over YouTube user log to Viacom in a lawsuit (New York Times, July 4, 2008). If the user log is not made anonymous enough, video viewing habits of tens of millions of YouTube users may be under the risk of exposure.

In recent years, the prevalence of smart phones has brought about unprecedented use of *location-based services* such as *Location-Based Social Networks (LBSN)* like Foursquare and Google Latitude. Various personalized recommendations are provided by location-based service such as coupon recommendation and targeted advertisements. Privacy issues become centered in these recommendation services. To better understand how a user may be identified in a personalized recommendation service, let us consider a toy but concrete example. Suppose that a user Albert has just arrived at a shopping mall. Wishing to get better personalized results, he may switch on the location service of his mobile phone and submit (or the LBSN that Albert has logged in may generate) a *query* q to a third-party coupon recommendation service, which provides specialized search results on coupon information. That means each query leaves a trace $\langle d, q, t\rangle$ on the query log of the recommendation service, where d is the user's *personal location information* (e.g., positions, movements, trajectories, Zipcode, etc.) and t refers to the query time. We assume that the query log does not contain explicit identity information of the query issuers. As a secondary use, the query log is published to an advertising company for data mining research, or to the public as in the AOL case. In the following discussion, the *attacker* refers to a party that has access to the query log and seeks to re-identify the (sensitive) queries of Albert, called the *target*. Usually, the attacker has some sort of relationship with Albert, e.g., col-

[1] http://www.choicestream.com/news/.

leagues, neighbors, friends, enemies, etc. In combination with Tweets, check-ins or other information, the *attacker* might be possible to build up a more complex picture about the *target*. Consider the following two ways of re-identification.

- *Re-identification through personal information* Suppose that the attacker knows that Albert has used the recommendation service. As one of thousands of location sharing service subscribers, Albert likes to share his current location with his family, friends, or even publish it online. Then the attacker could narrow down the queries issued by Albert by matching this knowledge against the personal information d in the query log. As reported by Pan et al. [22], when being combined with profile information, such as the user's office number, location traces can yield the identity of the user easily.
- *Re-identification through approximate query time* If the attacker also acquires an approximate query time such as "Albert used FourSquare two days ago", say from Albert's public "Check-in" history, the attacker can further narrow down the candidate queries by excluding the entries in the query log that are not within this time interval.

Another good example is Google's Ad Preferences. To show users Ads more related to their interests, Google stores an advertising cookie locally in user browser, which contains a list of interests and demographic information based on the websites the user visited. The cookie containing advertisement preferences information is sent out whenever the user visits other Google partners, which then provide "relevant" advertisements (according to Google AdWords).

The scenarios above involve collecting users' personal interests and demographic data locally, and passing them to a third party service for targeted advertising. To achieve better service quality, more personal information may need to be collected and sent. However, this may also bring privacy concerns as the combination of these personal information may reveal users' offline identity. We assume that a *personalized query* $\langle d, q\rangle$ has two parts d and q. The *query* part q contains query terms (i.e., free text) on which the user wants to get results. These are sensitive information and the user does not want to be identified as the sender of the query. The *personal information* d can contains personal location information of the user (e.g., positions, movements, trajectories, Zipcode, etc.), demographic data of the user (e.g., age, gender, race, etc.) and/or other preference information of the user, and is used to tailor the search results to the user's taste. Note that this scenario is distinguished from the personalization based on information on the recommendation service side such as query histories, which is beyond users' control. Right now to address the seemingly contradictory concern between service providers and users, the industry practice is to provide an "*opt-out*" button so that users can choose to share personal information in exchange for better service or not. The research question is hence whether we can find a better solution that *enables advertising/ recommendation while still protects users from being identified*?

The flow of a re-identification attack described above is illustrated in Fig. 1. A LBSN user u submits a personalized query $\langle d = \text{“}Redmond'', q = \text{“}wonder\ herb''\rangle$ at time $t = \text{“8:06pm}''$ to a recommendation service. Over time, the recommendation service collects a query log containing all entries $\langle d_i, q_i, t_i\rangle$. At a later time, the query log is used or published for data mining research. One recipient of the query log

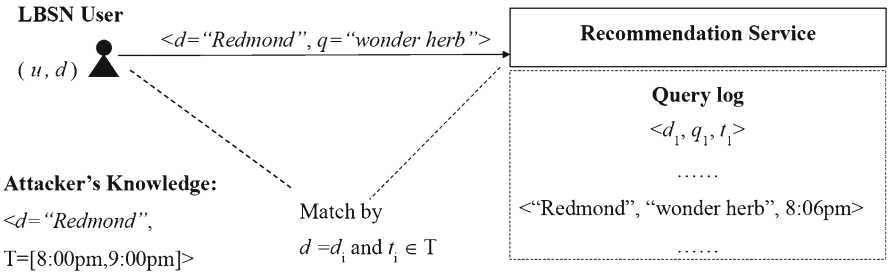


Fig. 1 Attacks in personalized recommendation services

(possibly the recommendation service itself), the attacker, has the *prior knowledge* $\langle d = \text{“}Redmond'', T = [8{:}00\text{pm}, 9{:}00\text{pm}]\rangle$ about some target user, i.e., the attacker knows that the target user with the personal information d has issued some queries at an approximated time range T. The attacker's goal is to identify the target user's queries from the query log. The anonymity of the target user is compromised if only a small number of entries $\langle d_i, q_i, t_i\rangle$ in the log match the prior knowledge $\langle d, T\rangle$.

Starting from end users' point of view, a key assumption in this work is that the recommendation service is untrusted due to the collection of potentially identifying personal information d. Specifically, we adapt the semi-honest model [9] for the recommendation service: the recommendation service will follow the specified computation, but may seek to use the collected query log to identify the query of a target user. This assumption ensures a stronger privacy guarantee on the query log so that it can be used by the recommendation service or published to third parties without causing end users' privacy concerns.

Our approach Under the assumption of the untrusted recommendation service, there is a privacy threat because the recommendation service owns both the personal information d and the query q. A detailed d may link a unique user or a small number of users to q. To break the link between the two, we introduce an untrusted third party called the *user pool*, which also follows the semi-honest model. Instead of sending $\langle d, q\rangle$ to the recommendation service directly, the user u first anonymizes d through the user pool and then sends $\langle d', q\rangle$ to the recommendation service, where d' is some generalization of d. The goal is to ensure that the generalized personal information d' cannot be linked to the user. We assume that all communications between a user and the recommendation service/user pool are anonymous [27]. As a result, the user pool possesses the raw personal information d, but has no knowledge about the query q that u may send. The recommendation service possess the query q and the generalized personal information d', but cannot identify u from d' because d' has been generalized. Our contribution is summarized below.

Contribution I We introduce the notion of online anonymity to ensure that each query entry $\langle d', q, t\rangle$ in the query log cannot be linked to its sender. Specifically, $\langle d', q, t\rangle$ has (k, w)-online anonymity if at least k distinct users have issued a query using the generalized personal information d' and within the w proximity of the query time t. Therefore, if the attacker's knowledge T about query time is not more accurate than w, all of these users are possible candidates for the sender of $\langle d', q, t\rangle$. We will show that online anonymity provides defense against the recommendation service.

Contribution II We propose an algorithm that achieves online anonymity through the user pool. A significant challenge comes from the assumption of untrusted recommendation service and user pool, and dealing with the dynamic sets of online users. Specifically, to provide online anonymity, the user pool must track the online users who issued queries during a certain time interval and anonymize their personal information d in an online fashion. This tracking also entails some interaction between the user pool and location-based service users. We propose a protocol for this interaction to guarantee that the additional information collected by the user pool cannot be used to compromise user anonymity.

Contribution III Although we focus on anonymizing the personal information d that is separately provided for the personalization purpose, in the same spirit, our approach can be extended to deal with personally identifying information that may be contained in the query q. In this sense, our work is also applicable to general web services where there is a need to anonymize the query, with or without personalization. We will discuss this extension in Sect. 3.

Contribution IV We conduct a simulation study using real datasets. The results show that it is feasible to achieve personalization while preserve user privacy.

The paper is structured as follows. Section 2 reviews related works. Section 3 introduces our framework. Sections 4 and 5 consider how to achieve online anonymity under our framework. Section 6 presents our simulation results. Section 7 concludes the paper.

2 Related work

In *privacy preserving data publishing* [7,21,25], a trusted party, called publisher, collects all data (i.e., query logs) first and anonymizes the data for publishing. This scenario is not applicable to our problem setting where personal information is held by individual location-based service users and there is no trusted data publisher. To put our scenario into the context of data publishing, the query entries in the query log are data records, but they must be generalized *before* they are submitted to the recommendation service. This distributed setting of data is similar to that of Zhong et al.'s study [32]. However, Zhong et al. consider a pre-defined set of users and cannot deal with the location-based service scenario where there is no pre-defined set of users because users get online and offline arbitrarily.

In *anonymous communication*, systems such as Mix-Net [27], Crowds [23], and Tor [6] aim to provide a communication channel for users to interact with the recommendation service anonymously. Similarly, *privacy-preserving data collection* [31] addresses respondents' anonymity in a data collection process. All these studies do not address the re-identification of data subjects from the *content of data* transmitted. In contrast, our work assumes communication anonymity as the infrastructure and focuses on re-identification attacks arising from examining the content of data.

Another body of work makes use of an *alias*, including *anonymous user accounts* [8], *digital pseudonyms* [14], and *anonymous web browsing*.[2] In the scenario

[2] http://www.anonymizer.com.

of personalized recommendation service, personal information is required for personalization and it is such information that links the queries to their senders. This threat exists independently of communication channels, pseudonyms, and user accounts used.

Our approach shares some similarity with studies on *privacy preserved location-based service (LBS)* [20,22,24]. For example, Chow et al. [4] propose to achieve privacy preserved LBS by sending queries where users' real locations are replaced by some fake locations. The main problem of this idea is that fake locations may lead to untrusted query results, which will affect the quality of the recommendation service significantly. Another popular technique [3,11] to enforce location privacy is to blur the users' exact location to a generalized area before it is sent to the LBS provider. This way the location data sensitivity may be reduced. Although these studies and ours share similar motivations, they are very different in terms of the goal and the problem settings. In the existing studies, privacy threat comes from the disclosure of detailed user location to a LBS provider. To anonymize location information, existing studies assume a trusted location anonymizer that acts as an intermediate tier between the users and the LBS, and all users are required to continuously report their locations to the anonymizer [22]. We do not require the continuous location reporting and allow users to get online/offline arbitrarily. Moreover, we do not require the trusted location anonymizer, which provides more flexibility. Meanwhile, existing studies focus on designing the spatial and temporal cloaking algorithms for preserving location privacy, and at the same time still keep the query efficiency and accuracy [19]. Our aim is not any particular anonymization algorithm, but achieving a new flexible framework for solving the online/dynamic privacy problem in untrusted recommendation services.

This article is an extension of our earlier short paper [30]. In the previous short paper, we introduced the notion of online anonymity to ensure that each query entry $\langle d', q, t\rangle$ in the query log cannot be linked to its sender. In this article, we extend the previous paper by adding definitions, algorithms and experiments. We propose a framework to achieve online anonymity. Firstly, we introduce the notion of $OA\text{-}groups$ to describe the set of online users who will leave query entries in the query log that are similar both in generalized personal information and query time. Secondly, we propose an algorithm that achieves online anonymity through the user pool. Here, the challenges are that the recommendation service and the user pool are both untrusted, while the location-based service users get online and offline dynamically. To overcome the challenges, we use the user pool to anonymize location-based service users' personal information d in an online fashion. Since the user pool will track the online users to provide anonymization, we further propose a protocol for this tracking to guarantee that the additional information collected by the user pool cannot be used to compromise user anonymity. We also perform a simulation study using real datasets. The results show that our framework can achieve personalization while preserving user privacy.

3 The framework

This section describes our personalization framework, the assumptions and privacy notion used in this paper.

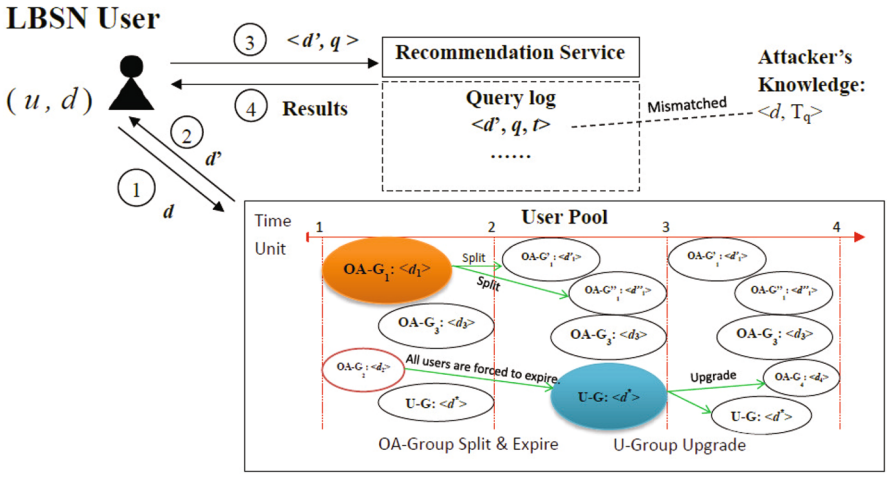


Fig. 2 The framework

3.1 Infrastructures

We consider a timeline labeled by a sequence of *time units* denoted by 1, 2, 3, ... A time unit could be a second, a minute, an hour, or a fraction of such units. A time interval or *window* is a sequence of consecutive time units. The window size refers to the number of time units in the window. Fig. 2 depicts the basic components and flow of information in our framework.

Recommendation service and location-based service users We consider a *recommendation service* and a collection of *location-based service users (e.g., LSBN users)*. For simplicity we refer the user as LBSN users. A user initiates a query q to the recommendation service, attached with his personal information d. Unlike database queries, a recommendation query q consists of several query terms, which are unstructured, unedited, and lack of pre-defined semantics. On receiving $\langle d, q\rangle$, the recommendation service returns personalized services (or results) to the user. Over time, the recommendation service collects and maintains a query log that contains all entries $\langle d, q, t\rangle$ ordered by the query time t.

Starting from end users' point of view, a key assumption in this work is that the recommendation service is untrusted due to the collection of potentially identifying personal information d. Specifically, we adapt a widely used model, the semi-honest model [9,16,17], for the recommendation service: the recommendation service will follow the specified computation, but may also seek to use the collected query log to identify the query of a target user. This assumption ensures a stronger privacy guarantee on the query log so that it can be used by the recommendation service or published to third parties without causing end users' privacy concerns.

We also assume that there is an anonymous communication channel [27] between each user and the recommendation service (e.g., the location-based service anonymizes the user identity). This means that queries and results will be transmitted between users and the recommendation service as expected, but the recommendation service

has no way to know the user behind a query and has no trace of different queries from the same user by observing where a query comes from. More details on anonymous communication channel implementations [6,23,27] can be found in a survey [5] by Danezis and Diaz.

User pool The function of *user pool* is to pull all LBSN users together and determine for each LBSN user the disclosure of personal information d in order to access personalized recommendation services anonymously. We assume the semi-honest model for the user pool and an anonymous communication channel between each LBSN user and the user pool. By choosing the semi-honest model for the user pool, we cover the case that the user pool may also seek to identify the sender of a query or even collude with the recommendation service to do so. See more discussions on attackers below.

In implementation, the user pool can be hosted by either some third party as a public service, or by recommendation services to offer their users anonymity so as to gain a competitive advantage over competitors. We envisage that the adoption process of the user pool might be similar to that of OpenID,[3] a shared identity service that allows Internet users to log on different recommendation services using a single digital identity. It initially arose from open source community as an open and free service, but later gained its popularity among large sites with large organizations such as Google and Facebook as providers.

Figure 2 illustrates the data flow among the user, user pool and the recommendation service.

1. Prior to sending a query q to the recommendation service, a LBSN user must first register at the user pool his personal information d.
2. The user pool determines the guarding group for u. There are two cases. If there exists some user group that can give the appropriate level of anonymization, then u gets registered to the group and the user pool returns a unique UID and d', where d' denotes the generalized personal information based on the group. If no such user group can be found, u will not be registered to any group. The user pool returns a unique UID and d^*, where d^* denotes a predefined most generalized personal information. Here, the user groups for anonymization are called the *guarding groups* and will be detail in Sect. 4. The generalized personal information d' (or d^*) contains less, but semantically consistent information based on d, e.g., if $d =$ "$Metrotown''$, the d' (or d^*) may contain "Burnaby".
3. Suppose u gets the generalized personal information d'. Subsequently the LBSN user submits the generalized personalized query $\langle d', q\rangle$ to the recommendation service.
4. Upon receiving $\langle d', q\rangle$, the recommendation service returns a query result to the user. It also adds an entry $\langle d', q, t\rangle$ to the query log.

To minimize the "*information loss*" on d (i.e., the difference between d and the returned version d', which will be detailed in Sect. 5.3), the guarding groups will be adjusted dynamically to provide online anonymity over time, since the LBSN user get online and offline arbitrarily. The dynamically adjustment is also illustrated in Fig. 2: (i) If there are too many users registering to a group and the group size reaches a

[3] http://www.openid.net.

pre-defined threshold, we split it into smaller groups to achieve better personalization level. (ii) If the users registered to a group get offline and the group size becomes too small, we force the remaining users to expire and remove the group. More details of the dynamically adjustment will be discussed in Sect. 5.

Attacker An *attacker* is a party that seeks to identify a query sent by a *target user* u from the query log. To do so, the attacker has the following information:

- *Query log*. We assume that the recommendation service has published the query log (for research purpose) and the attacker is one of the recipients.
- *Personal information d of u*. The attacker has obtained the personal information d of u as public knowledge.
- *Approximate time interval T_q during which u has sent a query q*. The attacker knows that u sent a query q within a time interval T_q (but does not know the content of q). Often, T_q is an interval containing the actual query time because the attacker knows an approximate query time, but not the exact query time. The size of the interval T_q indicates the "power" of the attacker.

Tuple $\langle d, T_q \rangle$ is called *prior knowledge* of the attacker about u. Note that our semi-honest model for the recommendation service and user pool covers the case that these parities may be the attacker if they obtain the above information. For example, the user pool could be one of the recipients of the published query log. In fact, the recommendation service and the user pool may even collude to identify a user's query. Note that, besides the prior knowledge $\langle d, T_q \rangle$, the user pool also has additional information collected from the interaction with LBSN users. We will have more discussion in Sect. 5.2.

The temporal accuracy of the attacker's knowledge T_q about query time can be modeled as follows.

Definition 1 (*w-oblivious*) Consider the attacker with prior knowledge $\langle d, T_q \rangle$ on a query q, we say that the attacker is w-oblivious if $[t - w, t + w] \subseteq T_q$, where t is the actual query time of a query q.

In other words, a w-oblivious attacker has at least $\pm w$ error around the actual query time. The smaller w is, the more precise the attacker's knowledge T_q is about the query time, therefore, the more powerful the attacker is. Our goal is to provide users anonymity against the w-oblivious attacker for a given w.

3.2 Online anonymity

Given the query log and the prior knowledge $\langle d, T_q \rangle$ on a target user u, the attacker tries to narrow down the candidate entries $\langle d', q, t \rangle$ (in the query log) originating from u. Such entries must match $\langle d, T_q \rangle$, i.e., $d \in d'$ and $t \in T_q$, which means d' is a generalization of d and T_q contains the query time t. In general, not all matched entries originated from u because other users have similar generalized personal information d' and have issued a query within T_q. The more matched queries were sent by different users, the less certain the attacker is about whether a matched query was actually sent by u.

Definition 2 (*k-identification*) We say that the target user is k-identified if the queries matching $\langle d, T_q \rangle$ in the query log were sent by less than k distinct users.

To prevent k-identification, we require that, for each query $\langle d', q, t \rangle$ in the query log, there are at least k "similar" queries sent by distinct users: these queries share the same generalized personal information d' and were sent within time proximity from t that is not distinguishable by the attacker. This motivates the following privacy notion.

Definition 3 (*(k, w)-online-anonymity*) A query $\langle d', q, t \rangle$ in the query log is said to have (k, w)-online-anonymity if there are at least k queries $\langle d'_i, q_i, t_i \rangle$ in the query log such that each $\langle d'_i, q_i, t_i \rangle$ was sent by a distinct user, $d'_i = d'$, and $|t - t_i| \leq w$. The query log is said to have (k, w)-online-anonymity if all queries in the log have (k, w)-online-anonymity.

Theorem 1 *If a query log has (k, w)-online-anonymity, for any query in the log, the sender of the query is not k-identified by any w-oblivious attacker.*

Proof Consider a query $\langle d', q, t \rangle$ in the query log with (k, w)-online-anonymity. Suppose that the attacker has the prior knowledge $\langle d, T_q \rangle$ about u and q. Since the attacker is w-oblivious (Definition 1), we have $[t - w, t + w] \subseteq T_q$. (k, w)-online-anonymity of $\langle d', q, t \rangle$ implies that at least k queries were sent within the interval $[t - w, t + w]$ by distinct users and those users share d' (Definition 3). All these queries match $\langle d, T_q \rangle$ because $[t - w, t + w] \subseteq T_q$. Since these queries were sent by k distinct users, from Definition 2, the sender of $\langle d', q, t \rangle$ is not k-identified. □

3.3 Discussion

3.3.1 Queries containing personal information

So far, we consider only attacks based on the personal information d provided for personalization. Sometimes, the query q itself may contain personally identifying information. We can extend the notion of online anonymity to cover personally identifying information contained in the query q.

Suppose that the query q can be divided into two parts. The *private sub-query* q^s refers to the set of private terms in q that the user wants to submit as it is and does not want to be identified as the sender. Such terms typically refer to financial information, health information, religion and political beliefs. The *public sub-query* q^p refers to the set of public terms in q that may potentially identify the user and the user is willing to modify. For example, for a query $q = \{stripper\ club, Redmond\ WA\}$, q^s could be "stripper club" and q^p could be "Redmond WA". We assume that public/private terms can be specified by the user.

With the above partition $\langle q^p, q^s \rangle$ of q, we can treat the public sub-query q^p as an extension of the personal information d and generalize both d and q^p using our method. To do so, we need to extend online anonymity as follows.

Definition 4 (*Extended (k, w)-online-anonymity*) A query $\langle d', q^{p'}, q^s, t \rangle$ in the query log is said to have (k, w)-online-anonymity if there are at least k queries $\langle d'_i, q_i^{p'}, q_i^s, t_i \rangle$

in the query log such that each $\langle d'_i, q_i^{p'}, q_i^s, t_i \rangle$ was sent by a distinct user, $d'_i = d'$, $q_i^{p'} = q^{p'}$ and $|t - t_i| \leq w$. The query log is said to have (k, w)-online-anonymity if all queries in the log have (k, w)-online-anonymity.

Unlike d, q^p may be unstructured (i.e., free text). Though much is known for anonymizing structured data in the literature [12,15,18,25], anonymization of unstructured data was not examined until recently [26,29]. However, the issue of how to anonymize d and q^p is orthogonal to our approach in that it is entirely local to the user pool and any generalization algorithm can be plugged into our approach. Our focus is on the challenge of providing online anonymity in the open and dynamic web setting without assuming a trusted recommendation service. Another implication of Definition 4 is that our approach is applicable to general recommendation services, with or without personalization. In the absence of personalization, d and d' are empty, and anonymization focuses on the personally identifying information in the query q.

3.3.2 Departure from well-defined semantics

Several types of attacks have been previously considered in the context of privacy preserving data publishing for relational data. One such attack is *homogeneity attack* [18] where most records in the same group share the same sensitive attribute. In this case, the attacker can infer the value with a high probability despite of the group size. In our setting, the query q corresponds to the sensitive attribute in a relational table. The study on the AOL query log [1] shows that 97 % of all queries are issued 3 times or fewer. This is more so if only queries within a certain time interval are considered. Thus, the homegeneity observed on dense relational data is less frequently observed on web queries that are in the long tail and extremely sparse. Nevertheless, given the power law distribution of web queries, in some cases it is still possible that users in a group may issue the same query that falls into the 3 % of the highly frequent queries. This privacy risk indeed exists but may be less severe for users when they are found to share a common interest with millions of other users. Overall, homogeneity attacks pose a great challenge in the open and dynamic context where queries are submitted separately and are unable to be anonymized in a centralized manner. Meanwhile, traditional methods can only deal with the static/centralized setting [18]. Solving the problem in an open and dynamic setting is beyond the scope of this study. As such, we will leave it as future work.

In *background attack* [18], knowing that an individual is a male, the attacker could exclude all records with $Disease =$"$Breast\ cancer$". This works on relational data because $Disease$ is the property of the record subject. However, the same reasoning does not hold for a recommendation query containing "$Breast\ cancer$" because a man can issue such queries simply out of interest in this topic. The difference in our scenario is that recommendation queries are unstructured and open to a wide range of semantics.

Another attack in the literature is *intersection attack* in the context of incremental data publishing [2]. Suppose that two anonymized publications both contain the record of some individual. The attacker can narrow down the candidate records for the

individual based on the assumption that the sensitive attribute of an individual *never changes*, which is not valid in our LBSN setting.

For the simplicity of explanation, we consider the online anonymity as given in Definition 3 in the rest of the paper.

4 Achieving online anonymity

To achieve online anonymity of the query log, we will generalize the raw personal information d (e.g., $d = \text{“}Metrotown''$) to a higher level representation (such as $d' = \text{“}Burnaby''$) that is less precise but semantically consistent with the original information. If a categorical attribute is involved, we may generalize it using a taxonomy of values. This generalization process will cause information loss and affect recommendation quality. Note that when there is a minimum requirement on user privacy, there may be a bound on the quality of recommendation service. To work towards this bound, we only generalize the raw personal information to the extent that meets the minimum user privacy requirement, so that we retain user personal data as precise as possible to achieve quality recommendation service. In this section we focus on how to achieve online anonymity using the “online anonymity groups”. We will detail our generalization algorithm in Sect. 5.

4.1 Online anonymity groups

The idea of anonymization is to group LBSN users by generalized personal information and query time so that their query entries in the query log provide (k, w)-online-anonymity for each other. We formalize this notion of user groups. Let W be a time window and d' be some generalized personal information.

Definition 5 (*OA-groups*) A LBSN user is online w.r.t. W and d' if he has sent at least one query to the recommendation service during W using d'. An OA-group (Online Anonymity Group) w.r.t. W and d', denoted by $G(W) : \langle d' \rangle$, consists of at least k online users w.r.t. W and d'. $|G(W) : \langle d' \rangle|$ is the number of online users in $G(W) : \langle d' \rangle$.

Intuitively, an OA-group is a set of online users who will leave query entries in the query log that are similar both in generalized personal information and query time.

Theorem 2 *Consider a user u in $G(W) : \langle d' \rangle$. Any query sent by u during W using d' has (k, w)-online-anonymity, where w is the size of W.*

Proof Let $\langle d', q, t \rangle$ be a query sent by the user u during W. From Definition 5, an OA-group $G(W) : \langle d' \rangle$ contains at least k users, and every user u_i in the group has sent some query q_i using d' at some time $t_i \in W$, therefore, has left an entry $\langle d', q_i, t_i \rangle$ in the log. Since $t \in W$ and $t_i \in W$ for all $\langle d', q_i, t_i \rangle$, $|t - t_i| \leq w$. From Definition 3, $\langle d', q, t \rangle$ has (k, w)-online-anonymity. □

Combining Theorems 1 and 2 together, by setting the size of W to w, OA-groups ensure that no sender of a query will be k-identified by a w-oblivious attacker. Here, w serves both the window size for W and the measure of the attacker's temporal accuracy.

The notion of OA-groups provides (k, w)-online-anonymity for one window W. At the next window, a new user may become online and an online user may become offline (if he does not issue a query). This will affect online anonymity and the generalization level. Our goal is to provide (k, w)-online-anonymity throughout the entire timeline and to provide the best personalization level. To achieve this goal, we must reflect the users who are "currently" online and update OA-groups as the window slides forward. In Sect. 5, we will consider this update.

4.2 User registration

Let us consider how a new user joins our system. For each user, we assume that a user *agent* running on the user's site will handle the details of interaction with the user pool such as registration, sending and receiving messages. Subsequently we simply say that the user sends or receives a message without mentioning the user agent. The user agent also ensures that each user registers to the user pool only once in each window. This is important since each registration is considered for a *distinct* user by the user pool.

Consider any moment in the current window W_i. Suppose that a new user u attempts to gain online anonymity for subsequent queries. Before interacting with the recommendation service, the new user u should first register with the user pool, by sending a request with his personal information d to the user pool. The user pool then determines the guarding group for u. There are two cases.

Registered users (OA-groups) If there exists some matching OA-group $G(W_i) : \langle d' \rangle$ where $d \in d'$, the one that minimizes the "information loss" on d is chosen as the guarding group for u (See Sect. 5.3). In this case, u gets registered and is returned a unique pseudo UID and the generalized personal information d'. UID is used by the user pool for keeping track of the users. In the rest of the window W_i, u can use d' to sent a query to the recommendation service. Since a registered user is guarded by an OA-group, from Theorem 2, all queries sent by a registered user within W_i have (k, w)-online-anonymity.

Unregistered users (U-group) If no OA-group matched, u gets unregistered and is guarded by the special *U-group* denoted by $G(W_i) : \langle d^* \rangle$, where d^* denotes the most generalized personal information. u is still returned a unique UID. The "unregistered" status indicates that within W_i the user pool is not able to provide the required online anonymity. This case occurs if there are less than k online users during W_i with similar personal information as the new user. An unregistered user can either interact with the recommendation service using d^* or wait to register at the next time window. The purpose of creating the U-group is to pull together all unregistered users so that they may be upgraded into OA-groups in W_{i+1} if at least k users in this group become online.

For each OA-group, the user pool maintains the following information:

- $G(W_i) : \langle d' \rangle$ denotes the set of registered users who were *online* users in the group, in the form of pairs $\langle d, \text{UID} \rangle$, where d is the original personal information and UID is the pseudo identifier assigned to the user.
- $G^+(W_i) : \langle d' \rangle$ denotes the set of all *registered users* associated with the group, also in the form of pairs $\langle d, \text{UID} \rangle$.

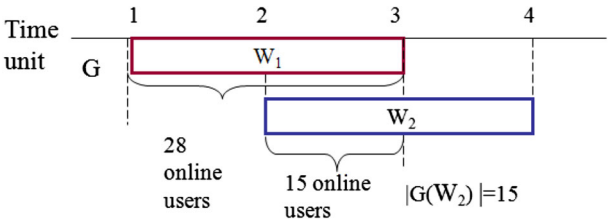


Fig. 3 Sliding the time window with $w = 2$, $p = 50$ %

Note that $G(W_i) : \langle d' \rangle \subseteq G^+(W_i) : \langle d' \rangle$. $G^+(W_i) : \langle d' \rangle - G(W_i) : \langle d' \rangle$ contains all registered users who have not issued a query during W_i. $G(W_i) : \langle d^* \rangle$ and $G^+(W_i) : \langle d^* \rangle$ denote corresponding users for the U-group.

5 Sliding the window

At the end of the current window, online anonymity and UID for all users expire because the online status is on the per window basis. To provide online anonymity over time, OA-groups and U-group needs to be updated from the current window to the next for two reasons: as new users get online, an OA-group may grow big enough to be split into smaller groups for better personalization, and the U-group may have enough online users to form OA-groups; as online users get offline, an OA-group may become too small and actions must be taken to protect remaining users in the OA-group. Our discussion focuses on OA-groups; there is a similar discussion for the U-group.

5.1 Window setting

Assume that OA-groups have been formed in the current window W_i. We want to form OA-groups in the next window W_{i+1}. In the most obvious window setting, W_{i+1} starts exactly after W_i ends. So $W_i \cap W_{i+1} = \emptyset$. This means that, at the start of W_{i+1}, no user is considered online in W_{i+1} because no users have sent any query at the start of W_{i+1}. Now, if a new user wants to register in W_{i+1}, he will find no OA-group for W_{i+1}, therefore, will go to the U-group. This is undesirable because no new user will get personalization.

We solve this problem by considering only window setting with some non-empty overlap $W_i \cap W_{i+1}$. Now, if we create the OA-groups for W_{i+1} at the end of W_i, such groups can jumpstart using the users that were online during the overlap $W_i \cap W_{i+1}$ because such users are considered online in both W_i and W_{i+1}. This idea is illustrated in Fig. 3 where the window size w is 2. The two consecutive windows W_1 and W_2 overlap by 50 %. There are 28 users who sent a query during W_1 (time units 1–3) and 15 of them sent a query in the overlap (time units 2–3). Therefore, at the end of W_1 (time unit 3), the 15 users in $W_1 \cap W_2$ are considered online in W_2 and can be used to form the OA-groups for W_2.

Another question is how much the overlap $W_i \cap W_{i+1}$ should be. Let $p = \frac{|W_i \cap W_{i+1}|}{w}$, called the *window overlap ratio*. With the group update taking place at the end of each window, there is one group update for every $(1 - p)w$ time units. The choice of p is

determined by the following tradeoff among efficiency, anonymity, and personalization: a larger p inherits more online users from the previous window and initializes larger groups in the next window, which means more anonymity and more personalization; on the other hand, a larger p means a more frequent update of groups, thus, more communications between users and the user pool, and more computation at the user pool.

5.2 Updating online users for W_{i+1}

At the end of W_i, two things happen. First, online anonymity and UID for all registered users automatically expire; if a user wishes to have online anonymity in the next window W_{i+1}, the user must get "extension" from the user pool, which confirms that the user is guarded by some OA-group in W_{i+1}. Second, the user pool will form the OA-groups $G(W_{i+1}) : \langle d' \rangle$ for the next window W_{i+1}, by inheriting the users who were online during the overlap $W_i \cap W_{i+1}$. For this purpose, all users who were online during $W_i \cap W_{i+1}$ should update the user pool at the end of W_i with this online status. Note that such users are not necessarily contained in $G(W_i) : \langle d' \rangle$ because they may become online only after $G(W_i) : \langle d' \rangle$ was formed. Such users are contained in $G^+(W_i) : \langle d' \rangle$.

The challenge is how to update this online status without leaking new information that may compromise user anonymity. First, let us consider a straightforward but unsafe update.

In a straightforward update, each user u with UID sends the update message $\langle \text{UID} \rangle$ to the user pool, indicating that he was online during $W_i \cap W_{i+1}$. With the UID, the user pool can know some user with personal information d has issued a query during $W_i \cap W_{i+1}$. Now if fewer than k users in $G^+(W_i) : \langle d' \rangle$ were online during $W_i \cap W_{i+1}$, the query log will contain fewer than k queries that match the prior knowledge $\langle d, W_i \cap W_{i+1} \rangle$; therefore, u is k-identified by the user pool.

The problem with the above solution is that, before confirming that there are at least k users online during $W_i \cap W_{i+1}$, a user u has announced that he is one of the online users during $W_i \cap W_{i+}$. To prevent this case, we propose a *two-phase messaging protocol* for a user to update his online status. In the first phase, every user in $G^+(W_i) : \langle d' \rangle$ who was online during $W_i \cap W_{i+1}$ sends the message $\langle d' \rangle$ (instead of $\langle \text{UID} \rangle$) to the user pool by the end of W_i. (The user agent will ensure that only such users send this message and each sends the message only once.) The purpose of this phase is to count the number of users online during $W_i \cap W_{i+1}$ without identifying their UID. Since there are at least k online users in $G^+(W_i) : \langle d' \rangle$ for an OA-group, all having the same d', the user pool cannot narrow down the sender of a message to fewer than k candidates. Let $N(i, d')$ denote the number of messages $\langle d' \rangle$ received for the group $G^+(W_i) : \langle d' \rangle$.

In the second phase, for every group $G^+(W_i) : \langle d' \rangle$ with $N(i, d') \geq k$, all message senders in the first phase send the second message $\langle \text{UID} \rangle$ to the user pool to identify their UID. Let $G(W_{i+1}) : \langle d' \rangle$ be the set of all users who send the second messages. Note that $|G(W_{i+1}) : \langle d' \rangle| = N(i, d') \geq k$. Therefore, $G(W_{i+1}) : \langle d' \rangle$ is an OA-group and members $G(W_{i+1}) : \langle d' \rangle$ are protected from being k-identified. For all

groups $G^+(W_i) : \langle d' \rangle$ with $N(i, d') < k$, $G(W_{i+1}) : \langle d' \rangle$ is formed because there are no enough online users.

We summarize the above protocol as follows.

Counting Phase

- If a user in $G^+(W_i) : \langle d' \rangle$ has sent at least a query during $W_i \cap W_{i+1}$, the user sends the message $M_1 = \langle d' \rangle$ to the user pool by the end of W_i.
- After collecting all messages M_1, if $N(i, d') \geq k$, the user pool informs all senders of the M_1 messages to identify their UID, by broadcasting a request to all members in $G^+(W_i) : \langle d' \rangle$.

Identification Phase

- On receiving the request from the user pool, all and only senders of M_1 send the second message $M_2 = \langle \text{UID} \rangle$ to the user pool. This can be ensured by the user agent.
- After collecting all messages M_2, $G(W_{i+1}) : \langle d' \rangle$ and $G^+(W_{i+1}) : \langle d' \rangle$ are formed by the set of UIDs contained in M_2. In addition, the user pool splits the group $G(W_{i+1}) : \langle d' \rangle$ into smaller groups (if necessary), by calling the algorithm SPLIT($G(W_{i+1}) : \langle d' \rangle$). This algorithm is run locally by the user pool. We will discuss this algorithm in Sect. 5.3.
- For each new $G(W_{i+1}) : \langle d' \rangle$, the user pool notifies all users in $G(W_{i+1}) : \langle d' \rangle$ of the extension of UID and d' to the next window W_{i+1}. For all users not receiving an extension notification, their UID expires at the end of W_i.

For each window update, each registered and unregistered user sends at most two messages M_1 and M_2, and receives at most two messages. The size of each message is negligible.

Expired Users The expiration of UID means that the user is eliminated from the user pool. Subsequently, if the user wants online anonymity, he must register with the user pool as a new user (Sect. 4.2).

There are two types of expired users. A *natural-expired* user refers to an expired user who was not online during $W_i \cap W_{i+1}$. A *forced-expired* user refers to an expired user who was online during $W_i \cap W_{i+1}$. A forced-expired user may have the tendency to register in the next window W_{i+1}. Such users belong to some $G(W_{i+1}) : \langle d' \rangle$ that is never formed because $|G(W_{i+1}) : \langle d' \rangle| < k$. A question is: would such users be under the risk of being k-identified by the user pool if the user pool can identify them by observing who are registering in W_{i+1}. The answer is no. Since $|G(W_{i+1}) : \langle d' \rangle| < k$, the user pool received only the first message $M_1 = \langle d' \rangle$, where d' is shared by all the members in $G^+(W_i) : \langle d' \rangle$. Since $|G^+(W_i) : \langle d' \rangle| \geq k$, even if the user pool can observe some user registering at W_{i+1}, this user could be any member from $G^+(W_i) : \langle d' \rangle$. Therefore, the user pool cannot narrow down to fewer than k candidates.

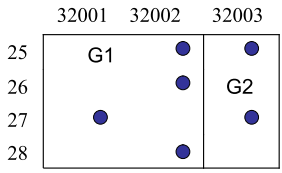
5.3 Updating OA-groups

At Step 2 of Identification Phase, SPLIT splits $G(W_{i+1}) : \langle d' \rangle$ into smaller groups $G(W_{i+1}) : \langle d'_1 \rangle$ and $G(W_{i+1}) : \langle d'_2 \rangle$, where d'_1 and d'_2 are two specializations of d'. Note that this operation is entirely local to the user pool and is orthogonal to the rest of

Fig. 4 A snapshot of online users in the user pool

User	<Age, Zipcode>
$u1$	<25, 32002>
$u2$	<25, 32003>
$u3$	<26, 32002>
$u4$	<27, 32001 >
$u5$	<27, 32003 >
$u6$	<28, 32002 >

(a) Online users



(b) Spatial Representation

our approach; any existing algorithm can be employed. For completeness, we describe one splitting algorithm.

Assume that every user has personal information d following the fixed template $\langle A_1, \ldots, A_n \rangle$, where A_i is an attribute on personal information. Let $dom(A_i)$ be the domain for attribute A_i. For ease of discussion, we assume that all attributes have a totally-ordered domain. The personal information d can be mapped to a point in the n-dimensional space $dom(A_1) \times \cdots \times dom(A_n)$. Figure 4 shows an example of 6 online users with 2-dimensional representation.

Each OA-group $G : \langle d' \rangle$ is represented by an n-dimensional rectangle: for each dimension A_i, there is an interval associated with the rectangle. Recall that each registered user is guarded by its OA-group. Continue with the example in Fig. 4. The user pool has two OA-groups with $k = 2 : G1 : \langle [25, 28], [32001 - 32002] \rangle$ where $|G_1| = 4$, and $G_2 : \langle [25, 28], 32003 \rangle$ where $|G_2| = 2$.

Recall that for any user joining in the user pool with personal information d, the user is guarded by the OA-group $G : \langle d' \rangle$ that results in the minimum information loss from d to d', denoted by IL(d'). There are several measures of information loss in the literature. The choice is orthogonal to our approach. For concreteness, we adopt the information loss measure used by Xiao and Tao [28].

Information loss The *information loss (IL)* in generalizing personal information d to d' is defined by

$$IL(d') = \frac{S(d') - 1}{S(D)}, \tag{1}$$

where S(x) returns the number of distinct points covered by the region x and d represents the whole space $dom(A_1) \times \cdots \times dom(A_n)$. In other words, IL($d'$) is the fraction of points covered by d' in the whole space. The more general d' is, the larger IL(d') is. The information loss for a group $G : \langle d' \rangle$ is defined by

$$IL(G) = |G| \times IL(d'), \tag{2}$$

where $|G|$ is the number of online users in G. Consider Fig. 4. S($d1'$ $= \langle [25, 28], [32001 - 32002] \rangle) = 4 \times 2 = 8$, S($D = \langle [25, 28], [32001 - 32003] \rangle$) $= 4 \times 3 = 12$. The information loss for any user in G_1 is IL(d_1') $= \frac{S(d_1')-1}{S(D)} = \frac{7}{12}$. The information loss for G_1 is IL(G_1) $= |G_1| \times$IL(d_1') $= 4 \times \frac{7}{12} = \frac{7}{3}$.

```
Algorithm SPLIT(G)
1.  if (|G|<2k) return G;
2.  bestSplit = <null, null,   >; //<dim, splitVal, infoLoss>
3.  for every dimension dim
4.          splitVal   find_median(G, dim);
5.          infoLoss = compute_IL(G, splitVal);
6.          if ( infoLoss <  bestSplit.infoLoss )
7.                bestSplit = <dim, splitVal, infoLoss>;
8.  endfor;
9.  G1    {<d, UID>  G | d.dim   bestSplit.splitVal  };
10. G2    {<d, UID>  G | d.dim > bestSplit.splitVal };
11. return SPLIT(G1) and SPLIT(G2)
```

Fig. 5 The group splitting algorithm

Splitting Algorithm We consider splitting a group G into two groups G_1 and G_2. A splitting is *valid* only if $|G_1| \geq k$ and $|G_2| \geq k$. Figure 5 shows the splitting algorithm adopted from [15]. The algorithm searches for the best dimension that produces the least information loss. For each dimension dim, instead of finding the best split criterion, it considers the (more efficient) splitting at the median that evenly distributes the users in G between G_1 and G_2 (Line 4). If there exists a valid splitting, the splitting at the median must be valid [15]. The information loss of the splitting then is computed by IL(G_1)+IL(G_2) (Line 5) and compared with the best information loss so far (Line 6). Finally, G is split by the dimension that produces the least information loss (Line 9-10). The splitting process is recursively applied to G_1 and G_2 until none of them has $size \geq 2k$.

According to [15], the time complexity of SPLIT(G) is $O(n \log n)$, where $n = |G|$ is the number of online users in G. Note that SPLIT(G) is run locally by the user pool.

6 Experiment

Our goal is to maximize personalization within the boundary set by the privacy requirement, i.e., online anonymity. To verify how much of this goal has been achieved, we conducted a simulation of online web environments. To the best of our knowledge no existing study has the same purpose or proposed a solution that is comparable. Therefore, we did not implement any competitor. The simulation system was implemented in C++ and run on a PC with 2.4 GHz CPU and 512 MB main memory.

6.1 Simulation setup

The online environment was simulated as follows.

User Base We used the Adults database from UC Irvine Machine Learning Repository[4] as our LBSN user population base. This data contains person-specific records from the US Census, which makes it an excellent sample of real US demographics.

[4] http://archive.ics.uci.edu/ml/.

As it was used by Iyengar [12], we selected eight regular attributes, i.e., age, work class, education, marital status, occupation, race, gender and native country, as the personal information d, and removed the records with missing values. The resulting database contains 30,162 records. As suggested by Iyengar [12], categorical attributes are totally-ordered by the preorder traversal of their taxonomies so that the splitting algorithm in Sect. 5.3 can be applied to both numeric and categorical attributes.

Online Users $(\tau, \lambda, \mu, \sigma)$ We denote each time unit by τ. For each time unit, we generated a number of new online users following the *Poisson distribution* with the arrival rate λ. Each new user was randomly picked from the user base and is associated with an online duration (in the number of τ) for the length of staying online. The online duration is randomly generated by the *Normal distribution* $N(\mu, \sigma)$ with the mean μ and the variance σ.

Measuring Personalization Various metrics have been used to evaluate the recommendation quality of recommendation algorithms (see [10] for a survey). For example, the *Root of the Mean Square Error (RMSE)* is a popular method for evaluating a recommendation algorithm, which measures the difference between the predicted preferences and the true preferences over items. Variants of this metric include the *Mean Square Error (MSE)*, *Mean Average Error (MAE)*, and *Normalized Mean Average Error (NMAE)*, etc. Another popular metric is the *recall*, which measures how many true preferences are recommended by a recommendation algorithm. While these metrics are intuitive, they rely on manually prepared ground truth data, which is subjective and data dependent. To avoid this subjectiveness, and since we are not measuring a particular recommendation algorithm, we use a new metric *AvgIL*. This metric measures the information loss caused by generalizing personal information, which is more objective. It can reflect the effect of our framework on the quality of recommendation because once a recommendation algorithm is chosen, its recommendation quality is mostly determined by how precisely the input data reflects the true feature of a user (e.g., user age).

Let IL(G) be defined by Eq. 2 for a group G of online users. For any window W_i, the average information loss for all online users is computed by AvgIL(W_i) $= \frac{\sum IL(G)}{\sum |G|}$, where $\sum$ is over all OA-groups G and the special U-group in the window W_i. Suppose that we apply our approach to windows $W_1, \ldots, W_N$. The personalization level is measured by averaging AvgIL(W_i) over all windows $W_1, \ldots, W_N$, i.e.,

$$AvgIL = \frac{\sum_{i=1,\ldots,N} AvgIL(W_i)}{N}. \tag{3}$$

A smaller AvgIL means a higher personalization level.

Parameters Several parameters affect personalization and privacy. These parameters fall into two categories.

- *User online parameters* include the user arrival rate λ and the (average) online duration μ. These two parameters, controlled by a user simulator, determine the user online characteristics. The larger the values of these parameters, the more online users in each window and the easier it is to achieve online anonymity and personalization.

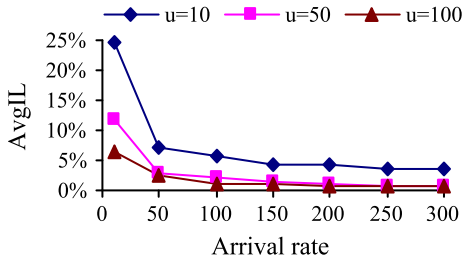


Fig. 6 Information Loss versus λ and μ

– *User pool parameters* include the anonymity threshold k, the window size W (in the number of τ) and the window overlap ratio p. A larger k provides more anonymity. A smaller w deals with an attacker with more accurate knowledge on query time, but leads to more information loss because there are fewer users in a smaller window. A larger p inherits more online users from the previous window, thus, reduces information loss. A smaller w and a larger p lead to a more frequent update of groups. See Sect. 5.1 for more discussion.

Unless otherwise specified, the following default settings are used: $\lambda = 50$, $\mu = 50$ (with the variance σ fixed at 10), $k = 30$, $w = 50$ and $p = 50$ %. For example, if τ corresponds to a second, these default settings represent the window size of 50 s, one update in every 25 s, the user arrival rate determined by the Poisson distribution with the mean of 50 users per second ($\lambda = 50$), the online duration determined by the Normal distribution with the mean of 50 s ($\mu = 50$). Note that the reported results are independent of the interpretation of τ. Below, we report the simulation results by varying one category of parameters at the default setting while fixing the other. Every experiment is run over 100 time windows.

6.2 User online characteristics

Varying λ and μ This experiment studies how user online characteristics λ and μ affect personalization and privacy, while fixing the user pool parameters k, w and p at the default value. The user arrival rate λ is varied from 10 to 300. This covers from small recommendation services having tens of new online users every time unit to popular recommendation services serving hundreds of queries per time unit. The mean of online duration μ is set at 10, 50 and 100 (with the variance σ fixed at 10), covering users who have a brief stay and users who have a longer stay.

The personalization level, represented by AvgIL, is summarized in Fig. 6. Not surprisingly, the personalization level benefits from having a larger user arrival rate λ. With more online users entering the system in each window, OA-groups have more refined regions for the generalized personal information d'. Interestingly, when λ is slightly increased from 10 to 50, for $\mu = 50$ and $\mu = 100$, AvgIL quickly drops to the highly satisfactory level of 5 %, and for $\mu = 10$, AvgIL drops to below 10 %. This clearly indicates that most information for personalization has been preserved for a wide range of user arrival rate.

Table 1 The percentage of unregistered/force-expired users

λ	10	50	100	150	200	250
Unregistered	3.0 %	0.9 %	0.5 %	0.5 %	0.2 %	0.1 %
Forced-expired	0.1 %	0.1 %	0.1 %	0.1 %	0.2 %	0.1 %

Figure 6 also shows that the longer the users stay online, i.e., a larger mean μ of online duration, the less information is lost. In fact, a longer online duration has a similar effect of increasing the number of online users within each window, thus, forming OA-groups with more refined regions for generalized personal information. We note that information loss is only reduced slightly by the change from $\mu = 50$ to $\mu = 100$. This is because the regions for OA-groups at $\mu = 50$ are already small (i.e., specialized).

These experiments suggest that it is not necessary to have a very large user arrival rate λ and a very large online duration μ in order to achieve a good personalization. Rather, it is more important to have some minimum user arrival rate and minimum online duration. In fact, these minimum requirements can be fulfilled by most popular recommendation services.

We also studied the frequency of unregistered users and forced-expired users (Sect. 5.2). Unregistered users will use the most generalized personal information d^* and do not get personalization. Forced-expired users are forced to expire, thus, become unregistered, due to elimination of their groups in the next window. We want to minimize the number of such users. Table 1 shows the percentage of unregistered users and forced-expired users among all users in a window. The default value for μ is 50 and the default value for k is 30. The percentage of forced-expired users remains at a very low level, indicating that most of OA-groups "survive" when the window slides. The percentage of unregistered users has a descending trend when λ increases. In this experiment, the default setting for k is 30. These percentages will be further reduced if k is set to a smaller value such as 5–10, which is sufficient for anonymity in many situations.

6.3 User pool characteristics

The parameters k, w and p define the characteristics of the user pool and jointly determine the tradeoff between personalization and privacy. This experiment studies how these parameters affect personalization and privacy, while fixing the user online parameters λ and μ at the default value.

Varying k The parameter k sets the minimum OA-group size. Figure 7 shows information loss vs. k. A larger k increases information loss because a larger OA-group requires a more generalized personal information d'. Typically, a small k in the range of 5–10 is sufficient to provide a reasonable anonymity level. Therefore, practically information loss is no more than 4 %.

Varying w and p The parameters w and p determine the window setting of the user pool. Note that w serves two measures: the window size, and the temporal accuracy of the attacker's knowledge about query time (i.e., the w-oblivious attacker, Definition 1). As shown in Fig. 8, a smaller w comes with more information loss. This is because a

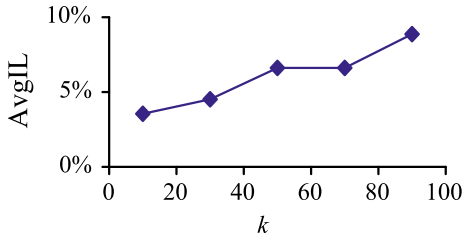


Fig. 7 Information Loss versus k

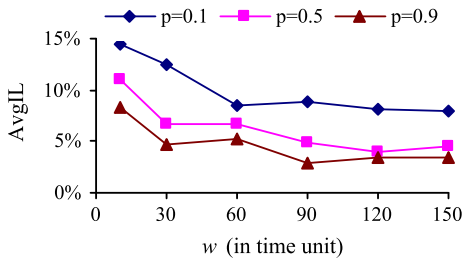


Fig. 8 Information Loss versus Window Settings

smaller window contains fewer online users to form OA-groups, which translates into more generalization of personal information. Also, a smaller w means a more frequent update of OA-groups, thus, more communication between LBSN users and the user pool. There seems to have a balance point at $w = 90$ where AvgIL drops to a low and stable level and further increasing w has little effect on AvgIL.

The window overlap ratio p does not affect user privacy. It however has an important implication on the personalization level. As we discussed in Sect. 5.1, the next window relies on online users in the overlap with the previous window to jumpstart OA-groups. With a fixed w, a larger p means more online users in the overlap to jumpstart the next window, which leads to more OA-groups and less information loss thereby. This is verified in Fig. 8. On the other hand, a larger p leads to a more frequent update of OA-groups. For example, with $w = 60$, $p = 0.1$ means one update every $(1 - p)w = 54$ time units and $p = 0.5$ means one update every $(1 - p)w = 30$ time units.

Among all the simulations conducted, the maximum number of online users processed by the user pool per window is over 44,000. Even that, the average runtime spent on group updating per window is less than 1 s. Analytically, the computational complexity is mainly determined by the group splitting algorithm, SPLIT, which is $O(n \log n)$, where n is number of online users in the group being split. Note that this algorithm is run locally by the user pool. If this approach is deployed on the web, there will be some message transmissions between LBSN users and the user pool prior to running SPLIT. As discussed in Sect. 5.2, this cost is minor because the number and the size of messages are very small.

6.4 Discussion

Until now, we considered the attacker who passively waits for the query log to be published. A more aggressive attacker could actively create the query log to compromise

the privacy of a target user u. One such case is that the attacker "controls" some number of LBSN users to send "special queries" to the recommendation service. Later after the query log is published, the attacker can remove such special queries; therefore, defeat the notion of online anonymity. While such attacks are not impossible, there are significant challenges in terms of the effort required. First, a controlled user must issue the special query in exactly the same time window as the target user u in order to be effective. If the time window is large, there are likely many online users in the window and the attacker's strategy will not be effective. If the time window is small, the attacker must know pretty accurately when u will issue a query in advance, which can be a challenge to the attacker because often the target user u itself may not know an accurate query time in advance due to various uncertainties.

7 Conclusion

This paper was motivated by two emerging trends: LBSN users want personalized services and LBSN users want privacy. One challenge is that personal information must be made anonymous under the assumption that the participating parties, including the recommendation service, are not completely trusted, due to systematic collection of personal information in addition to queries. Another challenge is the online and dynamic nature of LBSN users. We proposed the notion of online anonymity to protect LBSN users and we proposed an approach to maintain online anonymity through time. Our approach makes use of a third party called the user pool and we do not require the user pool to be trusted. The simulation study on real US demographics showed promising results: it is feasible to achieve personalization for reasonable privacy settings.

Acknowledgments Dr. Jin Huang is supported by the National Natural Science Foundation of China (Grant No. 61370229), the National Key Technology R&D Program of China (Grant No. 2013BAH72B01), and the Science-Technology Project of DEGP (Grant No.2012KJCX0037). A/Prof. Yabo Xu is supported by the National Natural Science Foundation of China (Grant No. 61100003). Prof. Jian Chen is supported by the National Natural Science Foundation of China (Grant No. 61272065), the Natural Science Foundation of Guangdong Province, China (Grant No. S2012010009311), and the Fundamental Research Funds for the Central Universities, SCUT (Grant No. 2012ZZ0088).

References

1. Adar, E.: User 4xxxxx9: Anonymizing query logs. In: Proceedings of the 16th International Conference on World Wide Web (WWW) (2007)
2. Byun, J.W., Li, T., Bertino, E., Li, N., Sohn, Y.: Privacy-preserving incremental data dissemination. J. Comput. Secur. **17**(1), 43–68 (2009)
3. Chow, C.Y., Mokbel, M.F.: Trajectory privacy in location-based services and data publication. SIGKDD Explor. Newsl. **13**(1), 19–29 (2011)
4. Chow, C.Y., Mokbel, M.F., Liu, X.: Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. Geoinformatica **15**(2), 351–380 (2011)
5. Danezis, G., Diaz, C.: A survey of anonymous communication channels. Tech. Rep. MSR-TR-2008-35, Microsoft Research (2008)
6. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In: Proceedings of the 13th Conference on USENIX Security Symposium (USENIX Security), pp. 21–21 (2004)

7. Fung, B.C.M., Wang, K., Chen, R., Yu, P.S.: Privacy-preserving data publishing: a survey of recent developments. ACM Comput. Surv. **42**(4), 14:1–14:53 (2010)
8. Gabber, E., Gibbons, P.B., Matias, Y., Mayer, A.J.: How to make personalized web browising simple, secure, and anonymous. In: Proceedings of the First International Conference on Financial Cryptography (FC), pp. 17–32 (1997)
9. Goldreich, O.: Foundations of Cryptography: Basic Tools, vol. 1. Cambridge University Press, Cambridge (2001)
10. Gunawardana, A., Shani, G.: A survey of accuracy evaluation metrics of recommendation tasks. J. Mach. Learn. Res. **10**, 2935–2962 (2009)
11. Hu, H., Xu, J., On, S.T., Du, J., Ng, J.K.Y.: Privacy-aware location data publishing. ACM Trans. Database Syst. **35**(3), 18:1–18:42 (2010)
12. Iyengar, V.S.: Transforming data to satisfy privacy constraints. In: Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data Mining (KDD), pp. 279–288 (2002)
13. Kobsa, A.: Privacy-enhanced personalization. Commun. ACM **50**(8), 24–33 (2007)
14. Kobsa, A., Schreck, J.: Privacy through pseudonymity in user-adaptive systems. ACM Trans. Internet Technol. **3**(2), 149–183 (2003)
15. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: Proceedings of the 22nd International Conference on Data Engineering (ICDE), pp. 25- (2006)
16. Li, D., Lv, Q., Xia, H., Shang, L., Lu, T., Gu, N.: Pistis: a privacy-preserving content recommender system for online social communities. In: Proceedings of the IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT), pp. 79–86 (2011)
17. Liu, J., Xiong, L., Luo, J.: A privacy framework: Indistinguishable privacy. In: Proceedings of the Joint EDBT/ICDT 2013 Workshops (EDBT), pp. 131–136 (2013)
18. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: L-Diversity: privacy beyond k-anonymity. ACM Trans. Knowl. Discov. Data **1**(1), 3 (2007)
19. Mascetti, S., Freni, D., Bettini, C., Wang, X.S., Jajodia, S.: Privacy in geo-social networks: proximity notification with untrusted service providers and curious buddies. VLDB J. **20**(4), 541–566 (2011)
20. Mokbel, M.F., Chow, C.Y., Aref, W.G.: The new casper: query processing for location services without compromising privacy. In: Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB), pp. 763–774 (2006)
21. Navarro-Arribas, G., Torra, V., Erola, A., Castellí-Roca, J.: User k-anonymity for privacy preserving data mining of query logs. Inf. Process. Manage. **48**(3), 476–487 (2012)
22. Pan, X., Xu, J., Meng, X.: Protecting location privacy against location-dependent attacks in mobile services. IEEE Trans. Knowl. Data Eng. **24**(8), 1506–1519 (2012)
23. Reiter, M.K., Rubin, A.D.: Crowds: anonymity for web transactions. ACM Trans. Inf. Syst. Secur. **1**(1), 66–92 (1998)
24. Stenneth, L., Yu, P.S.: Mobile systems privacy: 'mobipriv' a robust system for snapshot or continuous querying location based mobile systems. Trans. Data Priv. **5**(1), 333–376 (2012)
25. Sweeney, L.: k-Anonymity a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl. Based Syst. **10**(5), 557–570 (2002)
26. Terrovitis, M., Mamoulis, N., Kalnis, P.: Privacy-preserving anonymization of set-valued data. Proc. VLDB Endow. **1**(1), 115–125 (2008)
27. von Ahn, L., Bortz, A., Hopper, N.J.: k-anonymous message transmission. In: Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS), pp. 122–130 (2003)
28. Xiao, X., Tao, Y.: Personalized privacy preservation. In: Proceedings of the ACM international Conference on Management of Data (SIGMOD), pp. 229–240 (2006)
29. Xu, Y., Wang, K., Fu, A.W.C., Yu, P.S.: Anonymizing transaction databases for publication. In: Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (KDD), pp. 767–775 (2008)
30. Xu, Y., Wang, K., Yang, G., Fu, A.W.: Online anonymity for personalized web services. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM), pp. 1497–1500 (2009)
31. Xue, M., Papadimitriou, P., Raïssi, C., Kalnis, P., Pung, H.K.: Distributed privacy preserving data collection. In: Proceedings of the 16th International Conference on Database Systems for Advanced Applications (DASFAA), pp. 93–107 (2011)
32. Zhong, S., Yang, Z., Wright, R.N.: Privacy-enhancing k-anonymization of customer data. In: Proceedings of the 24th ACM Symposium on Principles of Database Systems (PODS), pp. 139–147 (2005)