TABLE 1
Symbols and Descriptions

Symbol	Description
$\lvert\mathcal{T}\rvert$	The count of nodes of the tree $\mathcal{T}$
$t \in \mathcal{T}/N \subset \mathcal{T}$	t is a node (N is a node set) in the tree $\mathcal{T}$
$subtr(t, \mathcal{T})$	The subtree rooted on t within the tree $\mathcal{T}$
$rsbtr(N, \mathcal{T})$	The rooted subtree of $\mathcal{T}$ by removing the set N
$trie(N)$	The topic-path prefix tree built with the set N
$root(\mathcal{T})$	The root of the tree $\mathcal{T}$
$par(t, \mathcal{T})$	The parent of t in the tree $\mathcal{T}$
$lca(N, \mathcal{T})$	The least common ancestor of the set N in $\mathcal{T}$
$C(t, \mathcal{T})$	The children of t within the tree $\mathcal{T}$

found that personalization may have different effects on different queries. Queries with smaller *click-entropies*, namely distinct queries, are expected to benefit more from personalization, while those with larger values (ambiguous ones) are not. Moreover, the latter may even cause privacy disclosure. Therefore, the need for personalization becomes questionable for such queries. Teevan et al. [26] collect a set of features of the query to classify queries by their click-entropy. While these works are motivative in questioning whether to personalize or not to, they assume the availability of massive user query logs (on the server side) and user feedback. In our UPS framework, we differentiate distinct queries from ambiguous ones based on a client-side solution using the predictive query utility metric.

This paper is an extension to our preliminary study reported in [27]. In the previous work, we have proposed the prototype of UPS, together with a greedy algorithm GreedyDP (named as GreedyUtility in [27]) to support online profiling based on predictive metrics of personalization utility and privacy risk. In this paper, we extend and detail the implementation of UPS. We extend the metric of personalization utility to capture our three new observations. We also refine the evaluation model of privacy risk to support user-customized sensitivities. Moreover, we propose a new profile generalization algorithm called GreedyIL. Based on three heuristics newly added in the extention, the efficiency and stability of the new algorithm outperforms the old one significantly.

3 Preliminaries and Problem Definition

In this section, we first introduce the structure of user profile in UPS. Then, we define the customized privacy requirements on a user profile. Finally, we present the attack model and formulate the problem of privacy-preserving profile generalization. For ease of presentation, Table 1 summarizes all the symbols used in this paper.

3.1 User Profile

Consistent with many previous works in personalized web services, each user profile in UPS adopts a hierarchical structure. Moreover, our profile is constructed based on the availability of a public accessible taxonomy, denoted as $\mathcal{R}$, which satisfies the following assumption.

Assumption 1. *The repository $\mathcal{R}$ is a huge topic hierarchy covering the entire topic domain of human knowledge. That is, given any human recognizable topic t, a corresponding node (also referred to as t) can be found in $\mathcal{R}$, with the subtree $subtr(t, \mathcal{R})$ as the taxonomy accompanying t.*

The repository is regarded as publicly available and can be used by anyone as the background knowledge. Such repositories do exist in the literature, for example, the ODP [1], [14], [3], [15], Wikipedia [16], [17], WordNet [22], and so on. In addition, each topic $t \in \mathcal{R}$ is associated with a *repository support*, denoted by $sup_{\mathcal{R}}(t)$, which quantifies how often the respective topic is touched in human knowledge. If we consider each topic to be the result of a random walk from its parent topic in $\mathcal{R}$, we have the following recursive equation:

$$sup_{\mathcal{R}}(t) = \sum_{t' \in C(t,\mathcal{R})} sup_{\mathcal{R}}(t'). \quad (1)$$

Equation (1) can be used to calculate the repository support of all topics in $\mathcal{R}$, relying on the following assumption that the support values of all leaf topics in $\mathcal{R}$ are available.

Assumption 2. *Given a taxonomy repository $\mathcal{R}$, the* repository support *is provided by $\mathcal{R}$ itself for each leaf topic.*

In fact, Assumption 2 can be relaxed if the support values are not available. In such case, it is still possible to "simulate" these *repository supports* with the topological structure of $\mathcal{R}$. That is, $sup_{\mathcal{R}}(t)$ can be calculated as the count of leaves in $subtr(t, \mathcal{R})$.

Based on the taxonomy repository, we define a probability model for the topic domain of the human knowledge. In the model, the repository $\mathcal{R}$ can be viewed as a hierarchical partitioning of the *universe* (represented by the *root* topic) and every topic $t \in \mathcal{R}$ stands for a random event. The conditional probability $Pr(t \mid s)$ (s is an ancestor of t) is defined as the proportion of *repository support*:

$$Pr(t \mid s) = \frac{sup_{\mathcal{R}}(t)}{sup_{\mathcal{R}}(s)}, \quad t \in subtr(s, \mathcal{R}). \quad (2)$$

Thus, $Pr(t)$ can be further defined as

$$Pr(t) = Pr(t \mid root(\mathcal{R})), \quad (3)$$

where $root(\mathcal{R})$ is the *root* topic which has probability 1. Now, we present the formal definition of user profile.

Definition 1 (User Profile/$\mathcal{H}$). *A user profile $\mathcal{H}$, as a hierarchical representation of user interests, is a* rooted subtree *of $\mathcal{R}$. The notion* rooted subtree *is given in Definition 2.*

Definition 2 (Rooted Subtree). *Given two trees $\mathcal{S}$ and $\mathcal{T}$, $\mathcal{S}$ is a* rooted subtree *of $\mathcal{T}$ if $\mathcal{S}$ can be generated from $\mathcal{T}$ by removing a node set $X \subset \mathcal{T}$ (together with subtrees) from $\mathcal{T}$, i.e., $\mathcal{S} = rsbtr(X, \mathcal{T})$.*

A diagram of a sample user profile is illustrated in Fig. 2a, which is constructed based on the sample taxonomy repository in Fig. 2b. We can observe that the owner of this profile is mainly interested in *Computer Science* and *Music*, because the major portion of this profile is made up of fragments from taxonomies of these two topics in the sample repository. Some other taxonomies also serve in comprising the profile, for example, *Sports* and *Adults*.

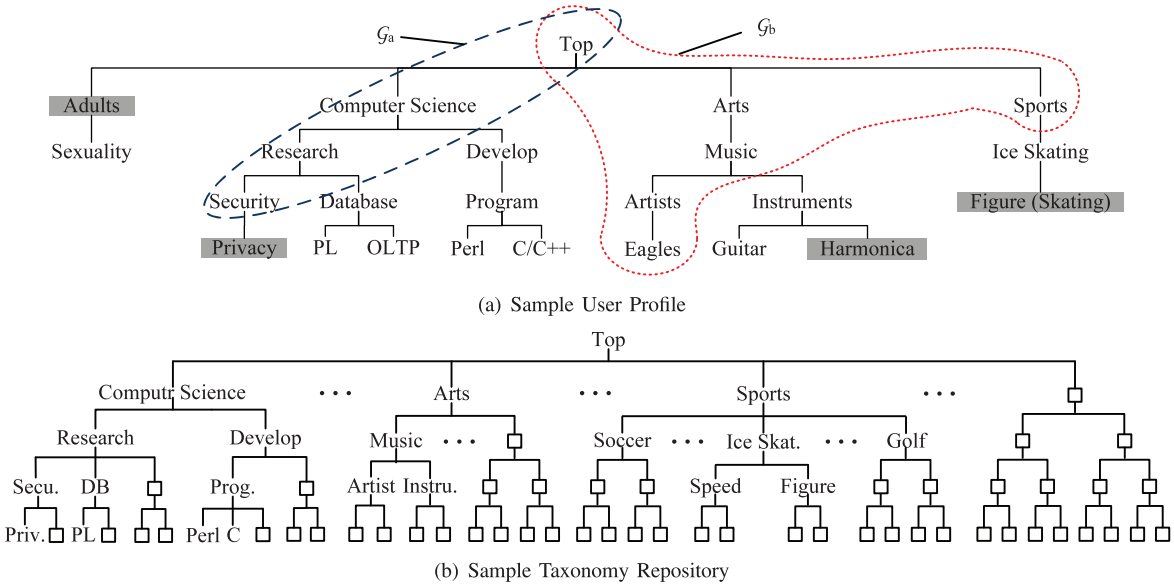

Fig. 2. Taxonomy-based user profile.

Although a user profile $\mathcal{H}$ inherits from $\mathcal{R}$ a subset of topic nodes and their links, it does not duplicate the repository supports. Instead, each topic $t \in \mathcal{H}$ is labeled with a *user support*, denoted by $sup_{\mathcal{H}}(t)$, which describes the user's preference on the respective topic t. Similar to its repository counterpart, the user support can be recursively aggregated from those specified on the leaf topics:

$$sup_{\mathcal{H}}(t) = \sum_{t' \in C(t,\mathcal{H})} sup_{\mathcal{H}}(t'). \quad (4)$$

The user support is different from the repository support as the former describes the user's preference on t, while the latter indicates the importance of t in the entire human knowledge.

3.2 Customized Privacy Requirements

Customized privacy requirements can be specified with a number of *sensitive-nodes* (topics) in the user profile, whose disclosure (to the server) introduces privacy risk to the user.

Definition 3 (SENSITIVE NODES/S). *Given a user profile $\mathcal{H}$, the* sensitive nodes *are a set of user specified sensitive topics $S \subset \mathcal{H}$, whose subtrees are* nonoverlapping, *i.e.,* $\forall s_1, s_2 \in S (s_1 \neq s_2), s_2 \notin subtr(s_1, \mathcal{H})$.

In the sample profile shown in Fig. 2a, the sensitive nodes $S = \{Adults, Privacy, Harmonica, Figure\ (Skating)\}$ are shaded in gray color in $\mathcal{H}$.

It must be noted that user's privacy concern differs from one sensitive topic to another. In the above example, the user may hesitate to share her personal interests (e.g., *Harmonica*, *Figure Skating*) only to avoid various advertisements. Thus, the user might still tolerate the exposure of such interests to trade for better personalization utility. However, the user may never allow another interest in topic *Adults* to be disclosed. To address the difference in privacy concerns, we allow the user to specify a *sensitivity* for each node $s \in S$.

Definition 4 (SENSITIVITY/$sen(s)$). *Given a* sensitive-node *s, its sensitivity, i.e., $sen(s)$, is a positive value that quantifies the severity of the privacy leakage caused by disclosing s.*

As the sensitivity values explicitly indicate the user's privacy concerns, the most straightforward privacy-preserving method is to remove subtrees rooted at all *sensitive-nodes* whose sensitivity values are greater than a threshold. Such method is referred to as *forbidding*. However, forbidding is far from enough against a more sophisticated adversary. To clearly illustrate the limitation of forbidding, we first introduce the attack model which we aim at resisting.

3.3 Attack Model

Our work aims at providing protection against a typical model of privacy attack, namely *eavesdropping*. As shown in Fig. 3, to corrupt Alice's privacy, the eavesdropper Eve successfully intercepts the communication between Alice and the PWS-server via some measures, such as *man-in-the-middle* attack, invading the server, and so on. Consequently, whenever Alice issues a query q, the entire copy of q together with a runtime profile $\mathcal{G}$ will be captured by Eve. Based on $\mathcal{G}$, Eve will attempt to touch the *sensitive nodes* of

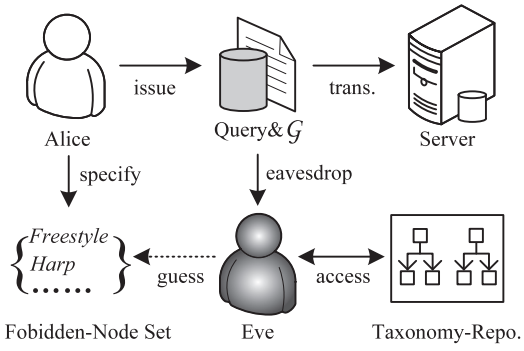

Fig. 3. Attack model of personalized web search.

Alice by recovering the segments hidden from the original $\mathcal{H}$ and computing a confidence for each recovered topic, relying on the background knowledge in the publicly available taxonomy repository $\mathcal{R}$.

Note that in our attack model, Eve is regarded as an adversary satisfying the following assumptions:

Knowledge bounded. The background knowledge of the adversary is limited to the taxonomy repository $\mathcal{R}$. Both the profile $\mathcal{H}$ and privacy are defined based on $\mathcal{R}$.

Session bounded. None of previously captured information is available for tracing the same victim in a long duration. In other words, the eavesdropping will be started and ended within a single query session.

The above assumptions seem strong, but are reasonable in practice. This is due to the fact that the majority of privacy attacks on the web are undertaken by some automatic programs for sending targeted (spam) advertisements to a large amount of PWS-users. These programs rarely act as a real person that collects prolific information of a specific victim for a long time as the latter is much more costly.

If we consider the sensitivity of each sensitive topic as the cost of recovering it, the *privacy risk* can be defined as the total (probabilistic) sensitivity of the sensitive nodes, which the adversary can probably recover from $\mathcal{G}$. For fairness among different users, we can normalize the privacy risk with $\sum_{s \in S} sen(s)$, which stands for the total wealth of the user. Our approach to privacy protection of personalized web search has to keep this privacy risk under control.

3.4 Generalizing User Profile

Now, we exemplify the inadequacy of *forbidding* operation. In the sample profile in Fig. 2a, *Figure* is specified as a sensitive node. Thus, $rsbtr(S, \mathcal{H})$ only releases its parent *Ice Skating*. Unfortunately, an adversary can recover the subtree of *Ice Skating* relying on the repository shown in Fig. 2b, where *Figure* is a main branch of *Ice Skating* besides *Speed*. If the probability of touching both branches is equal, the adversary can have 50 percent confidence on *Figure*. This may lead to high privacy risk if $sen(Figure)$ is high. A safer solution would remove node *Ice Skating* in such case for privacy protection. In contrast, it might be unnecessary to remove sensitive nodes with low sensitivity. Therefore, simply forbidding the sensitive topics does not protect the user's privacy needs precisely.

To address the problem with forbidding, we propose a technique, which detects and removes a set of nodes X from $\mathcal{H}$, such that the privacy risk introduced by exposing $\mathcal{G} = rsbtr(X, \mathcal{H})$ is always under control. Set X is typically different from S. For clarity of description, we assume that all the subtrees of $\mathcal{H}$ rooted at the nodes in X do not overlap each other. This process is called *generalization*, and the output $\mathcal{G}$ is a *generalized* profile.

The generalization technique can seemingly be conducted during offline processing without involving user queries. However, it is impractical to perform offline generalization due to two reasons:

1. The output from offline generalization may contain many topic branches, which are irrelevant to a query. A more flexible solution requires *online generalization*, which depends on the queries. Online generalization not only avoids unnecessary privacy disclosure, but also removes noisy topics that are irrelevant to the current query.

   For example, given a query $q_a =$ "K-Anonymity," which is a privacy protection technique used in data publishing, a desirable result of online generalization might be $\mathcal{G}_a$, surrounded by the dashed ellipse in Fig. 2a. For comparison, if the query is $q_b =$ "Eagles," the generalized profile would better become $\mathcal{G}_b$ contained in the dotted curve, which includes two possible intentions (one being a rock band and the other being an American football team Philadelphia Eagles). The node sets to be removed are $X_a = \{Adults, Privacy, Database, Develop, Arts, Sports\}$, and $X_b = \{Adults, Computer\ Science, Instrument, Ice\ Skating\}$, respectively.
2. It is important to monitor the personalization utility during the generalization. Using the running example, profiles $\mathcal{G}_a$ and $\mathcal{G}_b$ might be generalized to smaller rooted subtrees. However, overgeneralization may cause ambiguity in the personalization, and eventually lead to poor search results. Monitoring the utility would be possible only if we perform the generalization at runtime.

We now define the problem of privacy-preserving generalization in UPS as follows, based on two notions named *utility* and *risk*. The former measures the *personalization utility* of the generalized profile, while the latter measures the *privacy risk* of exposing the profile.

Problem 1 (δ-RISK PROFILE GENERALIZATION/δ-RPG). *Given a user profile $\mathcal{H}$ with* sensitive-nodes *S being specified, a query q, metric of privacy $risk(q, \mathcal{G})$, metric of utility $util(q, \mathcal{G})$, and a user specified threshold δ, the δ-risk profile generalization is to find an optimal instance of $\mathcal{G}$ (denoted as $\mathcal{G}*$), which satisfies*

$$\mathcal{G}* = \underset{\mathcal{G}}{\operatorname{argmax}}(util(q, \mathcal{G})), \quad risk(q, \mathcal{G}) < \delta. \tag{5}$$

In the above definition, δ represents the user's tolerance to the privacy risk (expense rate) of exposing the profile. Note that metric $risk(q, \mathcal{G})$ and $util(q, \mathcal{G})$ only depend on the instance of $\mathcal{G}$ and the query q as they are implemented to predict the privacy risk and personalization utility of $\mathcal{G}$ on q, without any user feedback. Details of these metrics will be presented in Section 5.1.

4 UPS PROCEDURES

In this section, we present the procedures carried out for each user during two different execution phases, namely the *offline* and *online* phases. Generally, the offline phase constructs the original user profile and then performs *privacy requirement customization* according to user-specified topic sensitivity. The subsequent online phase finds the Optimal δ-Risk Generalization solution in the search space determined by the customized user profile.

As mentioned in the previous section, the online generalization procedure is guided by the global risk and utility metrics. The computation of these metrics relies on two intermediate data structures, namely a *cost layer* and a *preference layer* defined on the user profile. The cost layer