ISSN:0975-9646

Supporting Privacy Protection In Personalized Web Searching and Browsing

Manali Wadnerkar

Department of Computer Engineering, University of Mumbai
Bharati Vidyapeeth College of Engineering, Sector-7, Belpada, Navi Mumbai, India

Dr. D.R.Ingle

Department of Computer Engineering, University of Mumbai
Bharati Vidyapeeth College of Engineering, Sector-7, Belpada, Navi Mumbai, India

***Abstract*— Personalized web search (PWS) has illustrated its effectiveness by improving the quality of search services on the Internet. But, evidence shows that users' hesitation to disclose their private information during search has become a major barrier for PWS. Privacy protection in PWS applications can be adopted that model user preferences as hierarchical user profiles by studying a PWS framework called UPS that adaptively generalizes profiles by queries while keeping in mind user-specified privacy requirements. Program slicing technique is used for decomposition of a program by analysing that particular program data and control flow.**

***Keywords*— Personalized web search, privacy, dynamic slicing, WWW, user profile.**

I. INTRODUCTION

It has become increasingly difficult for users to find information on the World Wide Web that satisfies their individual needs since information resources on the WWW continue to grow. Under these circumstances, Web search engines help users find useful information on the WWW. However, when the same query is submitted by different users, most search engines return the same results regardless of who submits the query. In general, each user has different information needs for his/her query. For example, for the query "Java," some users may be interested in documents dealing with the programming language, "Java," while other users may want documents related to "coffee." Therefore, Web search results should adapt to users with different information needs. The web search engine has become an important doorway for people for finding useful and necessary information. Nonetheless a user might come across failure when these search engines return unrelated results that do not meet their requirements. This happens due to enormous data, users' background and knowledge and ambiguity of texts. Personalized Web Search (PWS) is a search technique which aims at providing more efficient results, according to the users' needs. This requires user information to figure out the actual intention behind the requested query.

There are two solutions to PWS, click-log-based methods and profile-based-methods. The former is bias to clicked URLs or pages in the particular user's history and can work only on repeated queries. In contrast to this, the latter improves the search experience with user-interest models [1]. These user interest models are generated from users' profiles. PWS has illustrated more effectiveness in improving the quality of web data search. For this, implicit user data has to be collected which can be collected from query history [2], browsing history, bookmarks [1], and click-through data [3]. This raises privacy issues due to the lack of protection of user's private data. This may raise panic among the users and can also smother the publisher's enthusiasm for offering such services. As more and more topics are being discussed on the web and our vocabulary remains relatively stable, it is increasingly difficult to let the search engine know what we want. Coping with ambiguous queries has long been an important part in the research of Information Retrieval, but still remains to be a challenging task. Personalized search has recently got significant attention to address this challenge in the web search community, based on the premise that a user's general preference may help the search engine disambiguate the true intention of a query. However, studies have shown that users are reluctant to provide any explicit input on their personal preference. In this paper, study shows how a search engine can learn a user's preference automatically based on her past click history and how it can use the user preference to personalize search results.

There are two solutions to PWS, click-log-based methods and profile-based-methods. The former is bias to clicked URLs or pages in the particular user's history and can work only on repeated queries. In contrast to this, the latter improves the search experience with user-interest models [1]. These user interest models are generated from users' profiles. PWS has illustrated more effectiveness in improving the quality of web data search. For this, implicit user data has to be collected which can be collected from query history [2], browsing history, bookmarks [1], and click-through data [3]. This raises privacy issues due to the lack of protection of user's private data. This may raise panic among the users and can also smother the publisher's enthusiasm for offering such services. For protecting user privacy in profile-based PWS, developers have to consider two contradicting effects while performing the search process. They have to make an attempt to improve the search quality with the personalization utility and on the other hand they need to hide the privacy contents existing in the user profile for keeping the privacy risk under control

[1]. People are willing to compromise their private data if this will help in an easy access to required information and an efficient search quality. A significant amount of gain can be obtained by personalizing users' information at the cost of a small information, a generalized profile. . Hence, without compromising the search quality if the web user privacy can be protected. The previous works showing privacy preservation are not optimal. There are following concerns with the existing methods which can be explained as below:

- The existing methods do not perform customization of privacy requirements. This makes some user privacy to be insufficiently protected and some over-protected. The sensitive topics are detected using the absolute metric called surprisal [1]. The topics which are sensitive and the user wants to hide them may not be well protected. This increases the risk of losing a sensitive data.
- The existing profile-based PWS are unable to support runtime profiling. A user, when searches the web engine, his profile is generalized only once. This strategy has certain drawbacks since it uses one profile for all the queries. A better approach can be to make an online decision for whether to personalize the query and at runtime what to expose in a user profile.
- Iterative user interactions are needed when creating personalized search outcome. Predictive metrics unlike, average rank, are required to measure the search quality.

A good personalization algorithm relies on rich user profiles and web corpus. However, as the web corpus is on the server, re-ranking on the client side is bandwidth intensive because it requires a large number of search results transmitted to the client before re-ranking. Alternatively, if the amount of information transmitted is limited through filtering on the server side, it pins high hope on the existence of desired information among filtered results, which is not always the case. Therefore, most of personalized search services online like Google Personalized Search and Yahoo! My Web adopt the second approach to tailor results on the server by analysing collected personal information, e.g. personal interests, and search histories.

A key factor for today's popular search engines is that they provide a user-friendly interface. The topics which are displayed on the web page related to a particular query are in the form of list of keywords entered by the user in the search bar, ranked according to their relevance with the original query. Ranking has become a central research problem for informational retrieval and Web data search, as it directly influences the relevance of the search results, the quality of a search system and users' search experience. Given a query, the deployed ranking function measures the relevance of each document to the query, sorts all the relevant documents and presents a list of top-ranked ones to the user. Despite of the simple interaction which proved to be successful, a list of keywords is not a good descriptor of the required information by the users. Users can not always formulate an efficient query to these search engines. One reason for this is the ambiguous data which is entered by the user. Often, users try different queries till get satisfied with the appropriate results. If users are familiar with the specific terminologies required, effective formulation can be achieved. But this may not be the case always. Users may have a little knowledge about what they are searching or even worse they do not what they are searching at all. An example explained in [2], a tourist is searching for summer rentals ad in Chile may not know that most of such ads appearing on the web are for apartments in Vina del Mar which is a popular beach in Chile. But local users are well aware of such facts. Hence, the idea is to use these expert queries for helping the non-expert users. So, to overcome this problem some search engines help the users to specify alternative queries related to the original query in their search process. Nonetheless, this approach has privacy issues on exposing personal information to a public server. It usually requires users to grant the server full access to their personal and behaviour information on the Internet. Without the user's permission, gleaning such information would violate an individual's privacy. In practice, however, privacy is not absolute. There exist already many examples where people give up some privacy to gain economic benefit. One example is frequent shopper card in grocery stores. Consumers trade the benefit of extra saving in the grocery stores versus the creation of a detailed profile of their shopping behaviour. As another example, consider a basketball fan. He may not be comfortable broadcasting a weekly work-out schedule, but might not mind revealing an interest on basketball if a search engine can help identify "Rockets" as an NBA team instead of anything related to space exploration. Thus, people may compromise some personal information if this yields them some gain in service quality or profitability.

Paper is organized as section two deals with related works, section three deals with system architecture, section four deals with User Customizable Privacy Preserving Search (UPS), section five deals with slicing algorithm, section six deals with Generalization Algorithms, section seven deals with experimental results and section eight conclude with the results.

II. RELATED WORKS

In this section, the related works are overviewed. Focus is on the literature of profile-based personalization and privacy protection in PWS system.

A. Profile-based personalization

There have been several prior attempts to personalize Web search. One approach to personalization is to have users describe their general interests. For example, Google Personal asks users to build a profile of themselves by selecting categories of interests. This profile can then be used to personalize search results by mapping Web pages to the same categories. Many commercial information filtering systems use this approach, and it has been explored before to personalize Web search results. Personal profiles have also been used in the context of the Web search to create a personalized version of PageRank [10] for setting the query-independent priors on Web pages. A similar technique for mapping user queries to categories based on

the user's search history. Information about the user's intent can also be collected at query time by means of techniques such as relevance feedback or query refinement. The basic idea of these works is to tailor the search results by referring to, often implicitly, a user profile that reveals an individual information goal. In the remainder of this section, previous solutions to PWS can be reviewed the on two aspects, namely the representation of profiles, and the measure of the effectiveness of personalization.

Most recent works build profiles in hierarchical structures due to their stronger descriptive ability, better scalability, and higher access efficiency. The majority of the hierarchical representations are constructed with existing weighted topic hierarchy/graph, such as ODP, Wikipedia and so on. Another work in [5] builds the hierarchical profile automatically via term-frequency analysis on the user data. In the proposed UPS framework, focus is not on the implementation of the user profiles.

Actually, this framework can potentially adopt any hierarchical representation based on a taxonomy of knowledge. As for the performance measures of PWS in the literature, Normalized Discounted Cumulative Gain (nDCG) is a common measure of the effectiveness of an information retrieval system. It is based on a human-graded relevance scale of item-positions in the result list, and is, therefore, known for its high cost in explicit feedback collection. To reduce the human involvement in performance measuring, researchers also propose other metrics of personalized web search that rely on clicking decisions, including Average Precision, Rank Scoring and Average Rank [3]. Average Precision metric, proposed by Dou et al. [1], to measure the effectiveness of the personalization in UPS. Meanwhile, our work is distinguished from previous studies as it also proposes two predictive metrics, namely personalization utility and privacy risk, on a profile instance without requesting for user feedback.

B. Privacy Protection in PWS System

Typical works in the literature of protecting user identifications try to solve the privacy problem on different levels, including the pseudo-identity, the group identity, no identity, and no personal information. Solution to the first level is proved to fragile. The third and fourth levels are impractical due to high cost in communication and cryptography. Therefore, the existing efforts focus on the second level. Both [8] and [9] provide online anonymity on user profiles by generating a group profile of k users. Using this approach, the linkage between the query and a single user is broken. The useless user profile (UUP) protocol is proposed to shuffle queries among a group of users who issue them. As a result any entity cannot profile a certain individual. These works assume the existence of a trustworthy third-party anonymizer, which is not readily available over the Internet at large.

A more important property that distinguishes our work from [10] is that we provide personalized privacy protection in PWS. A person can specify the degree of privacy protection for her/his sensitive values by specifying "guarding nodes" in the taxonomy of the sensitive attribute. Motivate by this, we allow users to customize privacy needs in their hierarchical user profiles. Aside from the above works, a couple of recent studies have raised an interesting question that concerns the privacy protection in PWS. The previous works have 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. While these works are motivated 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.

C. Slicing

Two popular Anonymization techniques are generalization and bucketization. Generalization, replaces a value with a "less-specific but semantically consistent" value. The main problems with generalization are:

- It fails on high-dimensional data due to the curse of dimensionality.
- It causes too much information loss due to the uniform-distribution assumption.

Bucketization first partitions tuples in the table into buckets and then separates the quasi identifiers with the sensitive attribute by randomly permuting the sensitive attribute values in each bucket. The anonymized data consist of a set of buckets with permuted sensitive attribute values. In particular, bucketization has been used for anonymizing high-dimensional data. However, their approach assumes a clear separation between Qis and SAs. In addition, because the exact values of all QIs are released, membership information is disclosed. The key idea of slicing is to preserve correlations between highly correlated attributes and to break correlations between uncorrelated attributes thus achieving both better utility and better privacy. Third, existing data analysis (e.g. query answering) methods can be easily used on the sliced data.

III. SYSTEM ARCHITECTURE

A. Online profile

The proposed idea also suggests that the queries issued are recommended that are related to the input query and also search for different issues. This redirects the search process to related information of interest to the users searching previously and also keeping track of the related queries issued by other users. The key component for privacy protection is an online profiler implemented as a search proxy that runs on client side. This proxy maintains both the complete user profile in a hierarchical structure with semantics, and the user-specified privacy requirements i.e. sensitive nodes. It works in two phases, namely the offline phase and the online phase. In the offline phase, hierarchical profile is constructed and then customized with the user-specified privacy requirements [1]. The online phase can be conducted as follows:

- When query is generated the proxy generates a runtime user profile. This process is guided by considering two conflicting metrics, personalization utility and privacy risk.
- Then, the query and the generalized profile are sent together to the server.
- These results are then personalized with the profile and delivered back to the query proxy.
- Finally, the proxy sends back the results to the client.

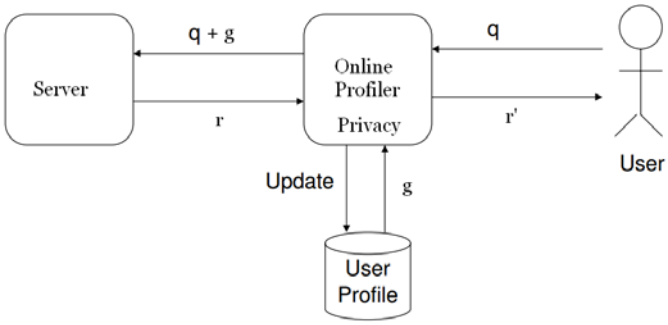


Fig. 1 Block diagram of UPS.

UPS differs from the conventional PWS since it provides runtime profiling which optimizes personalization utility, which performs customization on the sensitive data defined by the users, and does not require iterative user interaction.

Again, for efficient browsing, it is required to find the ranks of the related queries and cluster them. Queries along with the text of their clicked URLs extracted from the web log are clustered. This is done on the basis of two notions:

- Similarity of the query. The similarity of the query to the input query.
- Support of the query. This is a measure of how relevant is the query in the cluster. It is measured with the support of the query as the fraction of the documents returned by the query that captured the attention of users (clicked documents). It is estimated from the query log as well.

The quality of service can be improved when the location of the users are closer [4]. So, if the users share more data with each other the services provided by the web will be accurate. The studies show that the user is biased when it comes to searching information on the web. It can be trusts-biased or quality-biased [3]. This shows that clicks should be interpreted relative to the order of abstracts and presentation. Some attempts are made to use implicit feedback [4]. The reading time is indicative of interest while reading new stories. The reading time as well as number of times the user scrolls page can predict the relevance in browsing web. But it is generally considered that reading time varies between subjects and tasks, which makes it difficult to interpret. This difficulty can be resolved by the concept of eye-tracking. A general user approaches the results from top to bottom. It appears that users scan the viewable results before heading to scrolling. It gives evidences about users' decision making and indicates that users' clicking decisions are influenced by relevant results.

B. Session time-out

An experiment can be conducted where the users are observed with their clicked URL and session lengths and then can be re-enacted. For further help, clicks can be observed and assessment of the user's objectives can be done to label each session. Each query and clicked URL are assigned with ID number. A strength of this approach is that data is recorded without having an intervention and additionally we can observe large amount of users. There is a chance that the observer is biased to the user's goals but preliminary results show that the results achieved are reliable. The utility of adopting a hierarchical model for the grouping of user queries will allow us to more easily model what type of task the user may be doing when querying.

C. Attack Model

The user profile should be protected from adversaries which try to hamper the privacy and sensitive nodes defined by the user by a typical attack, namely eavesdropping. As shown in the Fig. 2 the eavesdropper intercepts successfully the communication happening between the server and the user by a measure, such as man-in-the-middle attack, invading the server. Accordingly, whenever the user issues any query q, the entire copy of q along with the runtime profile of the user will be seized the attacker.

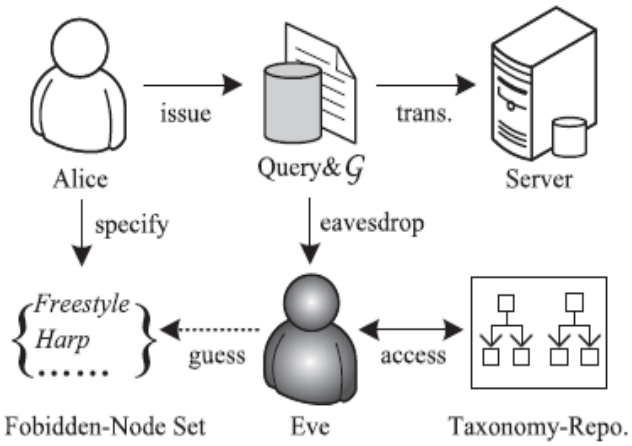


Fig. 2 Attack Model.

The attacker will then try to recover the hidden segments defined as private by the user. Now, the adversary is considered to satisfy the following assumptions:

Knowledge Bound. The background knowledge of the adversary is limited to the entire information available on the web. Both the original user profile and the privacy are defined within this information.

Session Bound. Previously captured information is not available for tracing the same victim. The eavesdropping will be started and ended within a single session.

These assumptions are strong but are reasonable in practice. This is considered since majority of attacks across the web happen by some automatic programs that sends advertisements (spam) to a wide range of users. An approach can be made to keep this privacy risk under control.

D. Generalizing user profile

This technique can be considered during the offline phase processing without involving any of the user's queries. But however it is impractical to perform this in offline phase because:

- This output from the offline phase may contain many topics that are completely irrelevant to the particular query. This can be solve if profile is generalized in the online phase.
- It avoids unnecessary privacy disclosure to the adversaries and also avoids noisy topics which are irrelevant to the query.
- It is very important to monitor the personalization factor during generalizing. But overgeneralization may cause ambiguity.

There are four phases in [1] which are used in generalization of the user profile. They can be explained in the following manner:

- *Offline profile construction*: This is the first step of the offline processing wherein the original user profile is built in a topic hierarchy which reveals the interest of the user.
- *Offline privacy requirement customization*: This phase requests the user to specify sensitive nodes which the user considers to remain hidden from the world. When any query q is issued, this customized user profile goes through the online phases.
- *Online query topic mapping*: There are two purposes for a query q, namely to compute a profile, seed, so that all the topics will be relevant to the query q. and to obtain the preference values i.e. values which are preferred to be present in the relevant topics of the query.
- *Online profile generalization*: This process generalizes the seed profile which relies on the privacy requirements of the user.

E. Online decision

When the profile is sent to the server the decision is made online. Here the user can decide whether the profile should be personalized or not. This depends on the number of distinct or similar queries. The similar queries are clustered together so that whenever the search is made the similar queries can be found in one place. This improves and enhances the search quality of the search engine.

The profile-based personalization contributes a little and even reduces the quality of search when there is a large amount of distinct queries. This may expose the profile to the server and will risk the privacy of the user. There is a solution to this problem. The decision to personalize the users' profile or not can be made in the online phase. The idea behind this phase it very simple, if a query issued is a distinct query during generalization the complete runtime profiling will then be aborted. Then the query will be sent to the server without any user profile. This enhances the stability of the quality of the search and also avoids the unnecessary exposure of the users' profile.

IV. User Customizable Privacy Preserving Search (UPS)

Personalized web search is a promising way to improve search quality by customizing search results for people with individual information goals. However, users are uncomfortable with exposing private preference information to search engines. On the other hand, privacy is not absolute, and often can be compromised if there is a gain in service or profitability to the user. Thus, a balance must be struck between search quality and privacy protection. A significant improvement on search quality can be achieved by only sharing some higher-level user profile information, which is potentially less sensitive than detailed personal information. Yet, there was a requirement of balance between the privacy protection and search quality because the information displayed by the web was not particularly relevant.

These problems can be overcome by giving a dynamic or runtime approach to building the user profile. This opts for efficient browsing and searching with data protection. For this a framework can be implemented called UPS. Personalized web search (PWS) has demonstrated its effectiveness in improving the quality of various search services on the Internet but evidences show that users' reluctance to disclose their private information during search has become a major barrier for the wide proliferation of PWS. UPS that can adaptively generalize profiles by queries while respecting user specified privacy requirements. The runtime generalization aims at striking a balance between two predictive metrics that evaluate the utility of personalization and the privacy risk of exposing the generalized profile. For this, there are two greedy algorithms, namely GreedyDP and GreedyIL, for runtime generalization. Also, provision is made for an online prediction mechanism for deciding whether personalizing a query is beneficial.

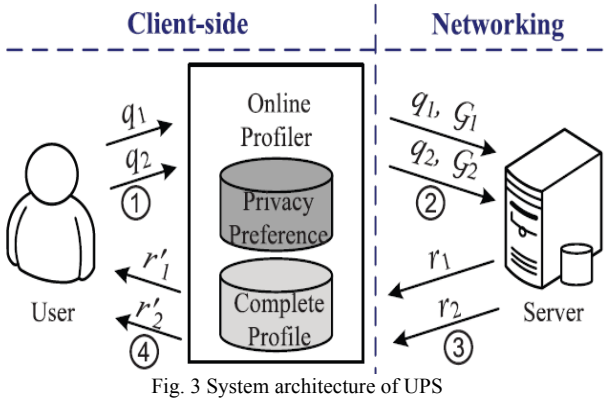


Fig. 3 System architecture of UPS

These algorithms work inside a web browser search plug-in to provide contextual search on the client-side, which would greatly benefit people's daily search. Also, balance can be achieved if web search are personalized by considering only exposing those information related to a specific query which can be done using these algorithms.

There are two metrics that are used to solve the generalization problem. They can be illustrated as:

- *Metric of utility:* The purpose of the utility metric is to predict the search quality (in revealing the user's intention) of the query q on a generalized profile G. The reason for not measuring the search quality directly is because search quality depends largely on the implementation of PWS search engine, which is hard to predict. In addition, it is too expensive to solicit user feedback on search results. Given a hierarchical profile G and a query q, we can intuitively expect more discriminating power when more specific topics are observed, the distribution is more concentrated on a few topics, and the topics are more similar to each other.
- *Metric of Privacy:* The privacy risk when exposing G is defined as the total sensitivity contained in it, given in normalized form. In the worst case, the original profile is exposed, and the risk of exposing all sensitive nodes reaches its maximum, namely 1. However, if a sensitive node is pruned and its ancestor nodes are retained during the generalization, the risk of exposing the ancestors should be evaluated. This can be done using the cost layer computed during Offline-2. However, in some cases, the cost of a non-leaf node might even be greater than the total risk aggregated from its children.

UPS consists of non-trusty search engine server and a number of clients. Each client (user) accessing the search service trusts no one but himself/ herself. The key component for privacy protection is an online profiler implemented as a search proxy running on the client machine itself. The proxy maintains both the complete user profile, in a hierarchy of nodes with semantics, and the user-specified (customized) privacy requirements represented as a set of sensitive-nodes. The framework works in two phases, namely the offline and online phase, for each user.

V. Slicing Algorithm

Many algorithms like bucketization, generalization have tried to preserve privacy however they exhibit attribute disclosure. So to overcome this problem an algorithm called slicing is used. This algorithm consists of three phases: attribute partitioning, column generalization, and tuple partitioning.

A. Attribute Partitioning

This algorithm partitions attributes so that highly correlated attributes are in the same column. This is good for both utility and privacy. In terms of data utility, grouping highly correlated attributes preserves the correlations among those attributes. In terms of privacy, the association of uncorrelated attributes presents higher identification risks than the association of highly correlated attributes because the associations of uncorrelated attribute values is much less frequent and thus more identifiable.

B. Column Generalization

Although column generalization is not a required phase, it can be useful in several aspects. First, column generalization may be required for identity/membership disclosure protection. If a column value is unique in a column (i.e., the column value appears only once in the column), a tuple with this unique column value can only have one matching bucket. This is not good for privacy protection, as in the case of generalization/bucketization where each tuple can belong to only one equivalence-class/bucket. The main problem is that this unique column value can be identifying. In this case, it would be useful to apply column generalization to ensure that each column value appears with at least some frequency. Second, when column generalization is applied, to achieve the same level of privacy against attribute disclosure, bucket sizes can be smaller. While column generalization may result in information loss, smaller bucket-sizes allow better data utility.

Therefore, there is a trade-off between column generalization and tuple partitioning.

C. Tuple Partitioning

The algorithm maintains two data structures: 1) a queue of buckets Q and 2) a set of sliced buckets SB. Initially, Q contains only one bucket which includes all tuples and SB is empty. For each iteration, the algorithm removes a bucket from Q and splits the bucket into two buckets. If the sliced table after the split satisfies l-diversity, then the algorithm puts the two buckets at the end of the queue Q Otherwise, we cannot split the bucket anymore and the algorithm puts the bucket into SB. When Q becomes empty, we have computed the sliced table. The set of sliced buckets is SB.

VI. Generalization Algorithms

A. The GreedyDP Algorithm

Given the complexity of our problem, a more practical solution would be a near-optimal greedy algorithm. As preliminary, we introduce an operator $\xrightarrow{-t}$ called prune-leaf, which indicates the removal of a leaf topic t from a profile. Formally, we denote by $G_i \xrightarrow{-t} G_{i+1}$ the process of pruning leaf t from Gi to obtain Gi+1. Obviously, the optimal profile G0 can be generated with a finite-length transitive closure of prune-leaf.

The first greedy algorithm GreedyDP works in a bottom-up manner. Starting from G0, in every ith iteration, GreedyDP chooses a leaf topic t ε TGi (q) for pruning, trying to maximize the utility of the output of the current iteration, namely Gi+1. During the iterations, we also maintain a best -profile- so-far, which indicates the Gi+1 having the highest discriminating power while satisfying the ┐ -risk constraint. The iterative process terminates when the profile is generalized to a root-topic. The best-profile-so-far will be the final result (G*) of the algorithm. The main problem of GreedyDP is that it requires recomputation of all candidate profiles (together with their discriminating power and privacy risk) generated from attempts of prune-leaf on all t ε TGi(q). This causes significant memory requirements and computational cost.

B. The GreedyIL algorithm

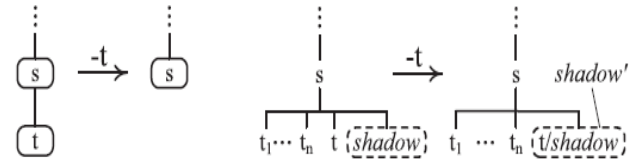
It improves the efficiency of the generalization using heuristics based on several findings. One important finding is that any prune-leaf operation reduces the discriminating power of the profile. In other words, the DP displays monotonicity by prune-leaf. Three following heuristics extends this algorithm:

- The iterative process can terminate whenever δ-risk is satisfied.
- Once a leaf topic t is pruned, only the candidate operators pruning t's sibling topics need to be updated in Q.

- IL(t)=Pr(t | q,G)(IC(t)-IC(par(tG))), case C1
  =dp(t) + dp(shadow)-dp(shadow'), case C2

where, $IC(t)=\log^{-1} Pr(t)$ (Pr is the probability of finding topic t)

& *dp* is the discriminating power.



(a) C1: t has no siblings.

(b) C2: t has siblings $t_1 \sim t_n$. Pruning t causes the preference value of t to be shifted into that of the shadow node.

Fig. 4 Two cases of prune-leaf on leaf a t

VII. EXPERIMENTAL RESULTS

In order to evaluate the effectiveness of the proposed GreedyIL algorithm, which consists of the slicing algorithm and the user profile generating algorithm, and evaluate the discovered user access patterns, experiments are conducted on real world data set and make comparisons with the previous work.

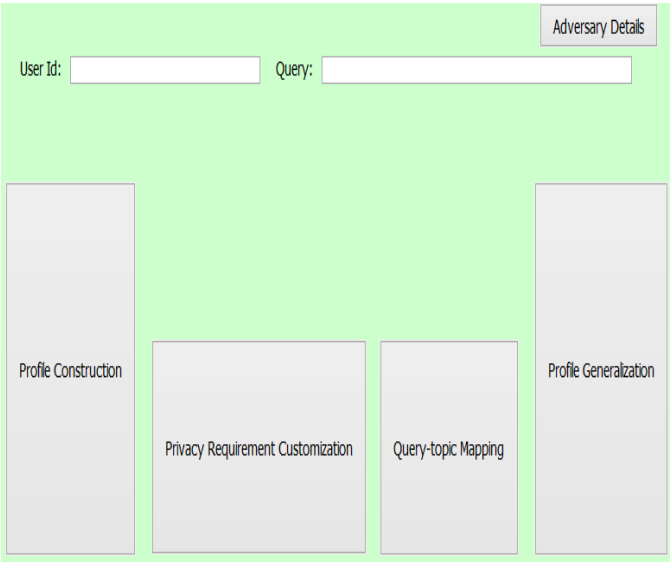


Fig. 5 Steps of the online profiler

The steps performed by the online profiler are displayed in the fig. 5. Profile is constructed firstly. This profile is the entire information of a particular user every time he/she logins. Then the privacy requirements which are given by the user at the time of registration are customized, the seed profile G_0 is created and then generalized.

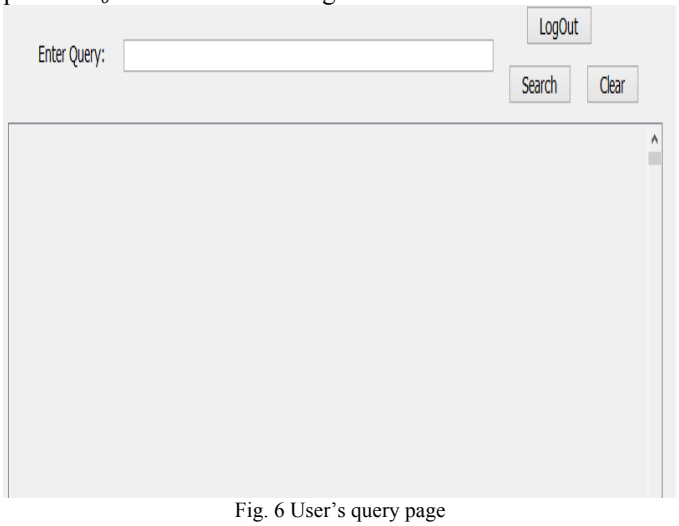


Fig. 6 User's query page

When user fires any query, instead of going to the server, the query and the generalized profile are sent to the online profiler. The data is then matched and if the user is valid and the risk is minimal the query is sent to the server. If any third party is trying to attack the user private data, he/she will be blocked by the proxy whenever the attacker makes an attack. A message will pop up about the blocking the IP address of the adversary as shown in fig. 7.

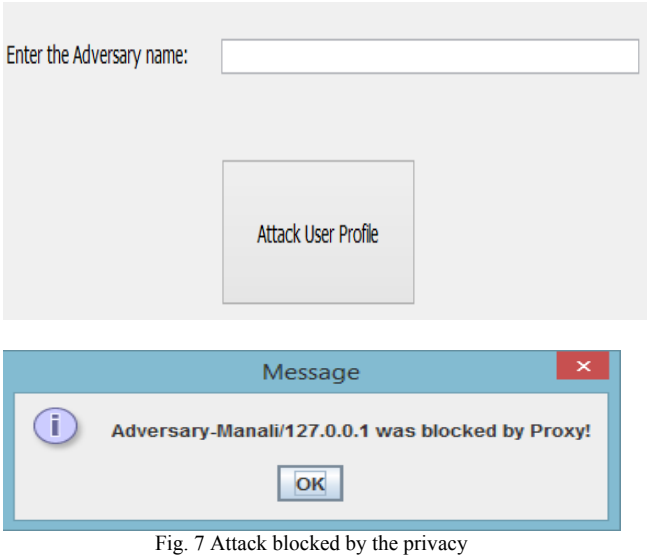


Fig. 7 Attack blocked by the privacy

The slicing algorithm helps to increase the efficiency of the search and enhances user's privacy. Because of this addition the user profile becomes difficult to understand and the attacker has a minimal chance to identify the user. Since the attacker is unable to find or identify the user profile, he is unable to access the hidden node-set of the particular user and thus the privacy of the user is now safe.

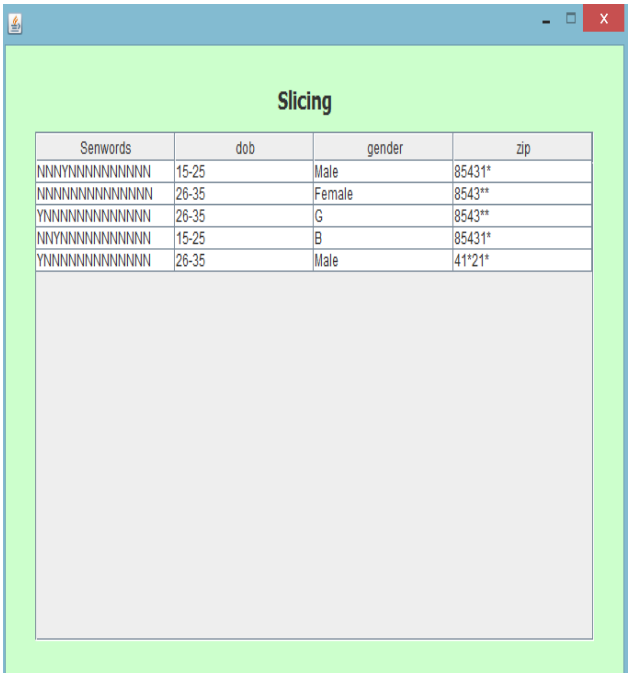


Fig. 8 Slicing on user profiles

VIII. CONCLUSION AND FUTURE WORK

It presented a client-side privacy protection framework called UPS for personalized web search. UPS could potentially be adopted by any PWS that captures user profiles in a hierarchical taxonomy. The framework allowed users to specify customized privacy requirements via the hierarchical profiles. In addition, UPS also performed online generalization on user profiles to protect the personal privacy without compromising the search quality. We proposed two greedy algorithms, namely GreedyDP and GreedyIL, for the online generalization. Personalized search is a promising way to improve search quality. However, this approach requires users to grant the server full access to personal information on Internet, which violates users' privacy. Hence, a way is suggested of achieving a balance between users' privacy and search quality. Users' data is collected, summarized, and organized their personal information into a hierarchical user profile, where general terms are ranked to higher levels than specific terms.

REFERENCES

[1] Yabo Xu, Benyu Zhang, Zheng Chen, Ke Wang, "Privacy-Enhancing Personalized Web Search," *WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.*

[2] Lidan Shou, He Bai, Ke Chen, and Gang Chen, "Supporting Privacy Protection in Personalized Web Search," *Proc. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING VOL:26 NO:2 YEAR 2014.*

[3] Ashwini Andhalkar, and Pradnya Ingawale, "Slicing: Privacy Preserving Data Publishing Technique," *International Journal of Computer & Organization Trends – Volume 5 Number 2 – February 2014.*

[4] Ricardo Baeza-Yates1, Carlos Hurtado1, and Marcelo Mendoza, "Query Recommendation using Query Logs in Search Engines," *Proc. Millennium Nucleus, Center for Web Research (P01-029-F), Mideplan, Chile.*

[5] Zhicheng Dou, Ruihua Song, JiRong Wen, "A Largescale Evaluation and Analysis of Personalized Search Strategies," *International World Wide Web Conference Committee (IW3C2), May 8–12, 2007, Banff, Alberta, Canada.*

[6] Jaime Teevan, Susan T. Dumais, Eric Horvitz, "Personalizing Search via Automated Analysis of Interests and Activities," *SIGIR '05, August 15–19, 2005, Salvador, Brazil.*

[7] Mirco Speretta, Susan Gauch, "Personalizing Search Based on User Search Histories", *Conference'04, Month 1–2, 2004, Salvador, Brazil.*

[8] *] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, Hang Li, "Context-Aware Query Suggestion by Mining Click-Through and Session Data," KDD'08, August 24–27, 2008, Las Vegas, Nevada, USA.*

[9] Kazunari Sugiyama, Kenji Hatano, Masatoshi Yoshikawa, "Adaptive Web Search Based on User Profile Constructed without Any Effort from Users," *WWW2004, May 17–22, 2004, New York, New York, USA.*

[10] Sonam Jain, Sandeep Poonia, "A New approach of program slicing: Mixed S-D (static & dynamic) slicing," *International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 5, May 2013.*

AUTHOR BIBLIOGRAPHY

Ms. Manali Wadnerkar currently pursuing M.E(CSE) in department of Computer Engineering , Bharati Vidyapeeth College of Engineering, Navi Mumbai. She has published one paper in the international journal and her areas of interest are web security, advanced databases, and web technology

Dr. D.R. Ingle is working as a Professor and Head in department of Computer Engineering, Bharati Vidyapeeth College of Engineering, Navi Mumbai. He has more than 40 papers to his credit at national and international level. He has taught various subjects as Object Oriented Analysis and design, Advanced Databases, Web Technology, Intelligent Systems, and Web Engineering etc. at graduate and post-graduate level. He has guided several projects at graduate and post-graduate level. He is a member of ACM and the life member of ISTE.