Online Anonymity Protection in Computer-Mediated Communication

Sara Motahari*, Sotirios G. Ziavras, Quentin Jones

In any situation where a set of personal attributes are revealed, there is a chance that revealed data can be linked back to its owner. Examples of such situations are publishing user profile micro-data or information about social ties, sharing profile information on social networking sites, or revealing personal information in computer-mediated communication. Measuring user anonymity is the first step to ensuring that the identity of the owner of revealed information cannot be inferred. Most current measures of anonymity ignore important factors such as the probabilistic nature of identity inference, the inferrer's outside knowledge, and the correlation between user attributes. Furthermore, in the social computing domain variations in personal information and various levels of information exchange among users make the problem more complicated. We present an information-entropy-based realistic estimation of the user anonymity level to deal with these issues in social computing in an effort to help predict \identity inference risks. We then address implementation issues of online protection by proposing complexity reduction methods that take advantage of basic information entropy properties. Our analysis and delay estimation based on experimental data show that our methods are viable, effective and efficient in facilitating privacy in social computing and synchronous computer-mediated communications.

***Index Terms*— data security, delay estimation, inference, information theory, privacy.**

I. Introduction

SOCIAL COMPUTING applications connect users to each other and support interpersonal communication (e.g. Instant Messaging), social navigation [1] (e.g. Facebook), and data sharing (e.g., flicker.com). The widespread use of ubiquitous and social computing poses privacy threats on many aspects of personal information, such as identity, location, profile information, social relations, etc. However, studies and polls suggest that identity is the most sensitive piece of users' information [2] and anonymity preservation is a key aspect of privacy protection and application design [3, 4]. Anonymity is defined as *"not having identifying characteristics such as a name or description of physical appearance…"* [5].

There are multiple situations where personal information is partially shared, but the information owner's anonymity must be protected. For example, there are various scenarios where organizations need to share or publish their user profile micro-data for legal, business or research purposes. To surmount the identification risk, attributes such as Name and Social Security Number are generally removed or replaced by false values. Nevertheless, previous research has shown that this type of anonymization alone may not be sufficient for identity protection [6]. For example, according to one study [7], approximately 87% of the population of the United States can be uniquely identified by their Gender, Date of Birth, and 5-digit Zip-code. Therefore, an individual's gender, date of birth and zip-code could be an *identity-leaking* set of attributes. Previous research on anonymity protection has mostly focused on this type of identity inference in micro-data and data mining. Recently, researchers have noticed the problem of identity inference in social network data [8]. In the above situations, it is usually assumed that the combinations of attributes that lead to identity inference are known and the focus is on anonymization solutions include suppression, randomization, or generalization of certain attribute values or inserting noise into the dataset.

Such efforts have resulted in valuable solutions for anonymity protection [6, 9, 10].These solutions usually have minor problems, such as ignoring the probabilistic nature of identity inference (usually resulted from the inferrer's uncertain outside information) and failing to identify identity-leaking attributes. However, the pervasive use of social computing applications has made this problem more complicated for the following reasons.

- There is no single dataset shared with all potential inferrers, but users of social computing applications share different information with different potential inferrers;
- User attributes such as location and friends may be dynamic and change;
- Users' anonymity preferences may be dynamic and change based on context such as time and location;
- The socially contextualized nature of information in such applications highly enables inferrers to use their background knowledge (outside information) to make inferences;
- In synchronous computer-mediated communications, users may progressively share their information piece by piece and the possible risk of identity inference as a result of revealing a new attribute has to be detected online.

Currently, system implementation and research on privacy in mobile and social applications are mostly limited to supporting users' privacy setting through direct access control systems [11-14]. Currently, such systems do not envision the linkability of personal information as explained above and do not effectively measure or protect the users' anonymity.

A social inference risk prediction framework based on the information entropy of a specific user attribute was proposed in [15, 16]. The contributions of this paper are as follows. In

This work was the Ross Memorial Scholarship, Phonetel Fellowship, National Science Foundation Grant NSF IIS DST 0534520 and CNS 0454081.

S. Motahari, S.G. Ziavras, and Quentin Jones are with the New Jersey Institute of Technology, Newark, NJ, 07102 USA (e-mail: sg262@njit.edu; ziavras@adm.njit.edu; qjones@njit.edu).

this paper, we expand the risk prediction framework to estimate the anonymity of a user based on information entropy. The information entropy is calculated by taking the inferrer's background knowledge into account. Such estimation can be used in any situation where personal attributes are shared. In the next step, to move towards an effective implementation of a protection system in *synchronous communication*, we will first present a brute-force algorithm and approximate its computational complexity. We then present a modified algorithm using basic properties of information entropy that can reduce the complexity. Analysis of delay and complexity based on our experimental data suggests that the proposed algorithm can be used to handle many users at the same time. We do not aim to propose an optimal algorithm, but our modified algorithm does not compromise privacy to reduce complexity, and addresses many gaps in anonymity protection research which we discuss below.

II. The Challenge of Measuring Anonymity

Anonymity has been discussed in the realm of data mining, social networks and computer networks with several attempts to quantify the degree of user anonymity. For example, Reiter and Rubin [17] define the degree of anonymity as 1- p, where p is the probability assigned to a particular user by a potential attacker. This does not give information on how distinguishable the user is from the other users. To measure the degree of anonymity of a user within a dataset, Sweeny proposed the notion of *k*-anonymity [10, 18]. In a *k*-anonymized dataset, each user record is indistinguishable from at least $k-1$ other records with respect to certain identity-leaking attributes. This work gained popularity and was later expanded by many researchers [9]. For example, L-diversity [19] was suggested to protect both identity and attribute inferences in databases. L-diversity adds the constraint that each group of *k*-anonymized users has L different values for a predefined set of L sensitive attributes. *k*-anonymity techniques can be broadly classified into generalization techniques, generalization with tuple suppression techniques, and data swapping and randomization techniques.

Recently researchers have tried to address some major problems of these methods, including: 1) *k*-anonymity solutions do not specify how to identify the identity-leaking attributes and assume that the owner of the information can identify them 2) a *k*-anonymized dataset is anonymized based on a fixed pre-determined *k* which may not be the proper value for all owners and all possible situations. For example, Lodha and Thomas tried to approximate the probability that a set of attributes is shared among less than *k* individuals for an arbitrary *k* [6]. However, they make unrealistic assumptions in their approach, such as assuming that an attribute takes its different possible values with almost the same probability or assuming that user attributes are not correlated. Unfortunately, although such assumptions simplify the task of anonymity estimation to a great extent, they are often invalid in practice. For example, different values of an attribute are not equally likely to appear. Also, users' attributes are highly correlated (e.g. age, gender, and even ethnicity are actually correlated with medical conditions, occupation, education, position, income and physical characteristics; home country is correlated with religion; religion is correlated with interests and activities, etc.) Therefore, the probability of a combination of a number of attributes cannot necessarily be obtained from the probabilities of individual attributes.

Machine learning as the next potential solution does not seem to be a reliable option for this estimation either [20]. This is for the same as above reasons and because user attributes are normally categorical variables that may be revealed in chunks. To estimate the degree of anonymity after revealing a set of attributes, the tool has to be able to capture joint probabilities of all possible values for all possible combinations of profile attributes (mostly categorical) and detect the outliers that may not even appear in the teaching set.

While privacy in data mining has been an important topic for many years, privacy for social network (social ties) data is a relatively new area of interest. A few researchers have suggested graph-based metrics to measure the degree of anonymity [8] or algorithms to test a social network, e.g. by de-anonymizing it [21]. Very little has been written on preserving anonymity within social network data. Campan and Truta [22] suggested an algorithm to maintain *k*-anonymity in social network data.

The first step in finding a set of identity-leaking attributes or social connections is to estimate an inferrer's outside information (background knowledge) as identity-leaking attributes consist of attributes that can be linked to the outside world. Although the need to model background knowledge has been recognized as an issue in database confidentiality [23], previous research on anonymity protection usually fails to address this important issue. Therefore, identifying such attributes remains an unsolved problem.

The final noteworthy problem of all mentioned solutions is that the notion of *k*-anonymity implies that *k* individuals (who share the revealed information) are completely indistinguishable from each other. This means that all *k* individuals are equally likely to be the true information owner. We will show in the next section that this may not be true due to various reasons, including nondeterministic background knowledge of the inferrer. Therefore, different probabilities should be assigned to different individuals to convey this probabilistic nature of identity inference. Denning and Morgenstern [24-26] were the first to use information entropy to predict the risk of such probabilistic inferences in multilevel databases. Given two data items x and y, let $H(y)$ denote the entropy of y and $H_x(y)$ denote the conditional entropy of y given x. They defined the reduction in the uncertainty of determining the value of y given x as $Infer(x \rightarrow y) = (H(y)-H_x(y))/H(y)$. The value of $Infer$ $(x \rightarrow y)$ is between 0 and 1, representing how likely it is to derive y given x. They did not show the process of determining this value as they did not dwell into the calculation of conditional entropies. We are only aware of the proposed use of information entropy in the context of *connection anonymity*; Serjantov and Danezis [27], Diaz et al. [28], and Toth et al. [29] suggested information theoretic measures to estimate the degree of anonymity of a message transmitter node in a network that uses mixing and delaying in the routing of messages. While [27] and [28] try to measure the average

anonymity of the nodes in the network, the work in [29] measures the worst case anonymity in a local network. Unlike the earlier approaches, their approach does not ignore the issue of the attacker's background knowledge, but they make abstract and limited assumptions about it that may not result in a realistic estimation of the probability distributions for nodes. More importantly, their approach measures the degree of anonymity for fixed nodes (such as desktops) and not necessarily their users.

In Section III, we will present a framework to model background knowledge and then dynamically calculate the information entropy based on already revealed attributes and background knowledge.

III. Information Theoretic Estimation of Anonymity

In this section, we employ information theory to estimate the users' level of anonymity. Before that, we briefly explain why we need to model an inferrer's background knowledge and then summarize an early user experiment in the domain of synchronous Computer-Mediated Communication (CMC) as its scenarios will be used as examples to elaborate on the calculations.

A. Background Knowledge Modeling

A reliable estimation of the anonymity level and the runtime identification of identity-leaking user attributes depend on effectively modeling the background knowledge as well as the development of an efficient algorithmic process to determine identity-leaking sets. The purpose of modeling the background knowledge in this context is to identify 1) what attributes, if revealed, can help the inferrer reduce the identity entropy of a user and how they change conditional probabilities, and 2) what attributes, even if not revealed, can help the inferrer reduce the identity entropy of a user and how they change conditional probabilities. As Jajodia and Meadows [30] say, "we have no way of controlling what data is learned outside of the database, and our abilities to predict it will be limited. Thus, even the best model can give us only an approximate idea of how safe a database is from illegal inferences". Accurate estimation of this knowledge may seem too difficult or expensive. However, we specified the following methods resulting in different levels of accuracy in estimating background knowledge and compared them in a previous study [16].

1. The simplest method is to assume that the inferrer can link what we have in the existing application database to the outside world, thus being able to estimate the number of matching users and their probabilities based on the existing database. The weakness of this method is that some of the attributes in the database are not usually known to the inferrer while some parts of the inferrer's background knowledge may not exist in the database.
2. The second method is to hypothesize about the inferrer's likely background knowledge taking the context of the application into consideration.
3. The third method is to utilize the results of relevant user studies designed to capture the users' background knowledge. The advantage of this method is a reliable modeling of background knowledge.
4. The last method may be an extension of the latter two methods with application usage data that allow for continuous monitoring of an inferrer's knowledge.

We investigated the comparative value and practicality of the second and third methods through two user studies. The results suggested that method 2 was almost as accurate as method 3 in the realm of CMC and proximity-based applications. This means considering the context and community of application users enables us to effectively model the background knowledge. However this may not be the case in all applications and user studies may be needed. Such studies can be merged with initial studies of the application, such as usability studies, so the estimation can be obtained with a low cost.

The framework explained in this section can estimate the level of anonymity in any situation where personal attributes are shared, especially in social computing. However, the computational complexity of calculating parameters such as V and $P_c(i)$ might raise concerns over the practicality of building an identity inference protection system for synchronous communications. In the next section we propose a brute-force algorithm and will see that its complexity calls for a faster algorithm, which will be proposed in Section VI.

B. Laboratory Experiment: Online Communication between Unknown Chat Partners

This experiment was originally designed to achieve various goals including 1) Investigate an inferrer's background knowledge in CMC for calculating the information entropy; 2) Test the ability of information entropy, calculated as explained in section III, to predict the inference risk; and 3) Explore the risk of social inferences in CMC.

Our subjects participated in a study consisting of three phases: 1) online personal profile entry; 2) an experiment involving subjects chatting with an unknown online partner; followed by 3) a post chat survey about the subject's ability to guess their chat partner's identity. Five hundred and thirty students entered a personal profile, 304 participated in the chat session of which 292 subjects completed all three study components. A detailed presentation of this study can be found in [16]. However, we mention some key results here:

- The only measure found to strongly predict the identity inference was information entropy of a user's identity as calculated in the next subsection.
- Different users desire different levels of anonymity.
- The background knowledge of a specific community in this context (attributes that can be linked to users' identities or be obtained from outside) was summarized as follows.
  1. Profile information that is visually observable, such as gender, approximate weight, height and age, ethnicity, attended classes, smoker/non-smoker, and on-campus jobs and activities.
  2. Profile information that is accessible through phone and address directories, or the organization's (community's) directories and website, such as phone number, address, email address, advisor/ boss, group membership, courses and on-campus jobs.
  3. Profile information that could be guessed based on the partner's chat style and be linked to the outside world

even without being revealed. They included gender with a probability of 10.4% and ethnicity with a probability of 5.2%, if not revealed.

Based on this study, we categorize the profile attributes that are included in the inferrer's background (linkable attributes) knowledge as follows.

Definition 1- *Linkable general attributes* if revealed can be linked to the outside world by the inferrer using his sources of background knowledge. For example, our experiment suggested that gender and on-campus jobs are linkable attributes in CMC, but favorite books or actors are not.

Definition 2- *Linkable probabilistic attributes* are attributes that could probably be obtained or guessed (even if not revealed) and then be linked to the outside world. They included gender and ethnicity in our experiment as they could be guessed from the chat style.

Definition 3- *Linkable identifying attributes* are attributes that uniquely specify people in most cases regardless of their value and regardless of values of other attributes of the user, such as the social security number, driving license number, cell phone number, first and last names, and often street home address.

This categorization is done based on the results of our user experiments. In other applications, attributes that can be linked to the outside worlds, may fall under different categories than above. In this paper, when we mention 'linkable attributes', we imply all of the above categories.

C. Level of Anonymity

Information [31], as used in information theory for telecommunications, is a measure of the decrease of uncertainty in a signal value at the receiver site. Here we use the fact that the more uncertain or random an event (outcome) is, the higher the *entropy* it possesses. If an event is very likely or very unlikely to happen, it will not be highly random and will have low entropy. Therefore, entropy is influenced by the probability of possible outcomes. It also depends on the number of possible events, because more possible outcomes make the result more uncertain. In our context, the probability of an event is the probability that a user's identity takes a specific value. As the inferrer collects more information, the number of users that match her/his collected information decreases, resulting in fewer possible values for the identity and lower information entropy.

To explain this in more detail, we refer to a real story from the above experiment; Bob, a university student, uses the chat software and engages in an online communication with Alice, a student from the same university. At the start of communication, Bob does not know anything about his chat partner. He is not told the name of the chat partner or anything else about her, so all potential users are equally likely to be his partner (the user probability is uniformly distributed). Thus, the information entropy has its highest possible value. After Alice starts chatting, her language and chat style may help Bob determine (guess) her gender and home country. At this point, users of the same gender and nationality are more likely to be his chat partner. Thus, the probability for Bob to guess his chat partner is no longer uniformly distributed over the users and the entropy decreases. After a while, Alice reveals that she is a Hispanic female and also plays for the university's women's soccer team. Bob, who has prior knowledge of this soccer team, knows that it has only one Hispanic member. This allows Bob to then infer Alice's identity. In summary, identity inferences in social applications happen when newly collected information reduces an inferrer's uncertainty about a user's identity to a level that she/he could deduce the user's identity. Collected information includes not only the information provided to users by the system, but also other information available outside of the application database or background knowledge.

We denote all the statistically significant information available to the inferrer, including background knowledge, by Q; Q includes the inferrer's background knowledge as well as answers to queries (or revealed information). Before the inferrer knows Q, a user's identity (Φ) maintains its maximum entropy. The maximum entropy of Φ, $H_{\max}$, is calculated by:

$$H_{\max} = -\sum_{1}^{N} \mathrm{P}.\log_2 \mathrm{P} \qquad \textbf{(1)}$$

where P=1/N and N is the maximum number of potential users related to the application.

Definition 4- We define the *level of anonymity* of user ***A*** as her/his conditional identity entropy, which is calculated by

$$L_{anon}(A) = H(\Phi \mid Q) = -\sum_{i=1}^{V} P_c(i).\log_2 P_c(i) \quad \textbf{(2)}$$

where Φ is the user's identity, $H(\Phi|Q)$ is the conditional entropy of Φ given Q, as defined in information theory, V is the number of possible values for attribute Φ, and $P_c(i)$ is the probability that the i^{th} possible identity is thought to be true by the inferrer. $P_c(i)$ is the posterior probability of each value given Q. Since here only linkable attributes can affect the information entropy, Q consists of linkable attributes that are already revealed and probabilistic attributes that are not revealed.

We illustrate the entropy model through the study example mentioned above; Alice is engaged in an on-line chat with Bob. In this case, Φ is Alice's identity at name or face granularity. At first her identity entropy is at its maximum level. After a while her chat style may enable Bob to guess her gender and home country. At this stage, Q comprises guesses on gender and home country which change the probability distribution of values as: $P_c(i)=$

$$\begin{cases} \dfrac{\alpha_2\alpha_1}{\mathrm{X3}} + \dfrac{\alpha_2(1-\alpha_1)}{\mathrm{X1}} + \dfrac{(1-\alpha_2)\alpha_1}{\mathrm{X2}} + \dfrac{(1-\alpha_2)(1-\alpha_1)}{\mathrm{V}}, \\ \text{(for users of the same gender and the same country)} \\ \alpha_2.(1-\alpha_1)\ /\mathrm{X1} + (1-\alpha_2).(1-\alpha_1)/\mathrm{V}, \text{ for users of only the same gender} \\ (1-\alpha_2).\alpha_1/\mathrm{X2} + (1-\alpha_2).(1-\alpha_1)/\mathrm{V}, \text{ for users of only the same country} \\ (1-\alpha_2).(1-\alpha_1)/\mathrm{V}, \text{ for the rest of the users} \end{cases}$$

where V is the number of possible users of the applications, X1 is the number of users of the same gender (females), and X2 is the number of users of the same ethnicity (Hispanics), X3 is the number of users of the same gender and ethnicity, α_1 is the probability of correctly guessing Alice's gender, α_2 is the probability of correctly guessing her home country [16] (α_k is the probability of guessing the k^{th} linkable probabilistic attribute correctly).

Alice then reveals she is Hispanic. At this stage, Q comprises the revealed information (ethnicity= Hispanic) and background knowledge. Since ethnicity was found to be part of her partner's background knowledge (a linkable profile

item), background knowledge includes users that are Hispanic. V is the number of users that satisfy Q, which is the number of Hispanic users:

$$P_c(\mathrm{i})=\begin{cases}\alpha_2/\mathrm{X1}+(1-\alpha_1)/\mathrm{V}, & \text{for users of the same gender that satisfy } Q\\ (1-\alpha_1)/\mathrm{V}, & \text{for other users that satisfy } Q\end{cases}$$

where V is the number of Hispanics and X1 is the number of female Hispanics.

When Alice reveals she is a female too, the probability is uniformly distributed over all Hispanic females. After she reveals her team membership, V is the number of users that satisfy [gender= female, ethnicity= Hispanic, and group membership= soccer team]. At this point, V=1, $P_c(1)=1$, and entropy is at its minimum level.

Definition 5- A *Matching set of users* based on a set of attribute values at each moment are the users who share the same values for the attributes at that moment. Let's consider the above example. At the very beginning, Alice's matching users based on her revealed attributes include all users, and at the end, her matching users are female Hispanic soccer players. Therefore, the number of ***A***'s matching users based on revealed attributes is V-1, excluding ***A***.

Let's assume in general that the inferrer's probabilistic attributes include k attributes $q_l,\ldots,q_k$ that have not been revealed yet and can be known independently with probabilities $\alpha_l,\ldots,\alpha_k$, respectively. If the profile of user i matches the attributes $q_l,\ldots,q_l$, then $P_c(i)$ is obtained from the equation: $P_c(i)=$

$$\prod_{z=l+1}^{k}(1-\alpha_z)\left\{\sum_{\Gamma_j\subset\{q_1\ldots q_l\}}\frac{\left(\prod_{q_h\in\Gamma_j}\alpha_h\right)\left(\prod_{q_r\sim\in\Gamma_j}1-\alpha_r\right)}{X(\Gamma_j)}\right\} \quad \textbf{(3)}$$

where Γ_j is any subset of $\{q_1\ldots q_l\}$ including null and $X(\Gamma_j)$ is the number of matching users only based on Γ_j.

In the special case that $P_c(i)$ equals 1/ V for all i, user ***A*** is completely indistinguishable from V-1 other users (the assumption made in the notion of k-anonymity). Therefore,

$$H(\Phi|Q)=-\Sigma(1/\mathrm{V}).\log_2(1/\mathrm{V})=\log_2\mathrm{V} \quad \textbf{(4)}$$

In this case, the entropy is only a function of V. Since ***A*** is indistinguishable from V-1 users, V is ***A***'s degree of anonymity as defined in the notion of k-anonymity. To avoid confusion, we always call V a user's *degree of obscurity*.

Definition 6- User ***A***'s *desired degree of obscurity* is **U** if he/she wishes to be indistinguishable from **U**-1 other users.

A user is at the risk of identity inference if her/his *level of anonymity* is less than a certain *threshold*. To take a user's privacy preferences into consideration, this anonymity threshold can be obtained by using the desired degree of obscurity and replacing V by **U** in Equation **(2)**:

Definition 7- *Anonymity Threshold*=$\log_2\mathbf{U}$.

Definition 8- A set of attributes in ***A***'s profile is called an *identity-leaking set* if revealing the set brings ***A***'s level of anonymity down to a value less than his anonymity threshold.

A reliable estimation of the level of anonymity and detecting the identity-leaking attributes depend on effectively modeling the background knowledge and efficient computational complexity to determine identity-leaking sets.

IV. Brute-Force Algorithm for Online Estimation of The Anonymity

Usually, when profile exchanges happen during synchronous CMC, identity leaking sets should be detected online so that users can be warned before sending a new piece of information. For dynamic user profiles consisting of attributes allowed to change, prior anonymity estimations cannot be safely assumed as valid. Thus, relevant estimations have to be computed dynamically on-demand. Here we first propose a brute-force algorithm and then estimate its computational complexity.

A. Brute-Force Algorithm (Algorithm I)

Let's assume that user ***A*** is engaged in a communication session with user ***B*** and reveals some of his profile items. For simplicity, let's also assume that all the user profiles are stored in a multi-dimensional database where the first dimension is the user ID and the other dimensions represent the user attributes (i.e., profile items). The anonymity *thresholds* for all the users have been calculated based on their desired degree of obscurity and are stored in another dimension of this database. We denote the set of ***A***'s already revealed profile attributes with ***S***. Before ***A*** reveals anything, ***S*** is null. The steps in Algorithm I for each newly revealed profile attribute are:

1. Every time ***A*** decides to reveal a new attribute, q_j, which is a linkable profile item, search the entire database of user profiles and find ***A***'s set of *matching users* based on $\boldsymbol{S}\cup\{q_j\}$.
2. Let V be equal to the number of so obtained matching users. Derive $P_c(i)$ from Equation **(3)**.
3. Calculate this user's anonymity level by applying Equation **(2)**.
4. If the level of anonymity is equal to or less than this user's anonymity threshold, $\boldsymbol{S}\cup\{q_j\}$ is an identity-leaking set. Otherwise, reveal q_j and set $\boldsymbol{S}=\boldsymbol{S}\cup\{q_j\}$.

B. Computational Complexity of Algorithm I

The most computationally expensive step in Algorithm I is to search for the set of matching users to obtain V and $P_c(i)$. In step 1, $\{q_1,\ldots,q_{j-1}\}$, which includes the already revealed linkable attributes of ***A*** along with the to-be-revealed item q_j, are compared with the same attributes stored for all the system users. This results in j comparisons for each known user. Assuming that there are at most n linkable profile attributes (including general, probabilistic, and identifying attributes) and N is the total number of users, in the worst case n*(N-1) comparisons and V<N summations may be needed. Thus, the worst case computational complexity is O(n*N), which grows linearly with both n and N. This complexity may be an issue for a large community of users, thus we summarize a few properties of information entropy that can be used to further reduce the complexity by modifying Algorithm I.

V. Properties of Information Entropy Used To Reduce the Computational Complexity

We take advantage in Section VI of the following entropy properties in order to derive a faster algorithm.

a) Information entropy here (i.e., the level of user anonymity) is an increasing function of V (e.g., user ***A***'s degree of obscurity) at each stage. Let's assume that **Y** and **Z** are two subsets of users, where **Y** is a subset of **Z**. If ***A***'s

anonymity level is higher than ***A***'s anonymity threshold among the users in **Y**, his anonymity level will also be higher than its threshold among the users in **Z**:
($\boldsymbol{A} \in \mathbf{Y}$, $\mathbf{Y} \subset \mathbf{Z}$, $\sum_{i \in Y} P_1(i)$ >threshold)=> $\sum_{i \in Z} P_1(i)$ >threshold

b) Although probabilistic attributes in the inferrer's background knowledge can slightly deviate $P_c(i)$ from a uniform distribution, a sufficiently large V still results in a level of anonymity being higher than its threshold. We call this value of V the *sufficiency threshold*, *T*. The value of the sufficiency threshold that guarantees a high enough level of anonymity for these V users is determined by the smallest possible value of $P_c(i)$ and the maximum desired degree of obscurity. It can be derived from the following equation.

c) $\log_2(\mathbf{U}_{max}) = \sum_{i=1}^{T} \left(-\left(\frac{\prod_{l=1}^{k}(1-\alpha_l)}{T}\right) . \log_2 \left(\frac{\prod_{l=1}^{k}(1-\alpha_l)}{T}\right)\right)$. The following definition is pertinent.

Sufficiency threshold: $T = \prod_{l=1}^{k} (1-\alpha_l) * \mathbf{U}_{max}^{\,1/\prod(1-\alpha_l)}$

d) The maximum level of anonymity for a given degree of obscurity V is $\log_2 V$. If V is less than the desired degree of obscurity **U**, even the maximum level of anonymity $\log_2 V$ is less than the threshold $\log_2 \mathbf{U}$. Therefore, for V<**U** the level of anonymity is always below its threshold regardless of the probability P_c.

e) This last characteristic relates to sets. Every time ***A*** reveals a new attribute q_j, since ***S*** is a subset of $\boldsymbol{S} \cup \{q_j\}$, the set of ***A***'s matching users based on $\boldsymbol{S} \cup \{q_j\}$ is a subset of ***A***'s matching users based on ***S***.

VI. IMPROVED ALGORITHM

We do not aim to propose here an optimal algorithm in terms of space savings and time efficiency primarily because our main intention is to demonstrate the viability and effectiveness of our proposed privacy-protection scheme. Many existing solutions can be used to store and index the involved very sparse arrays. We show here how considering the basic properties mentioned above can reduce the computational complexity to produce a viable system without compromising on user privacy.

A. Algorithm II

Let's assume that a user, say ***A,*** engages in on-line communication with another user, say ***B***. Assume that ***A***'s desired degree of obscurity, anonymity threshold, and sufficiency threshold are already pre-calculated and stored. Algorithm II works with m '*lists of values*' and one '*m-dimensional array* **E**, where m is the number of general and probabilistic attributes (excluding linkable identifying attributes, which means m<n). Each dimension of the m-dimensional array **E** represents one attribute and the number of elements in the k^{th} dimension equals the number of distinct values of the k^{th} attribute, including null. The value of each element represents the number of *matching users* that have the same set of m attribute values denoted by the indices of this element in array **E**. For example, for m=3 element $E_{4,2,6}$ holds the number of users whose first attribute has its fourth possible value, the second attribute has its second possible value, and the third attribute has its sixth possible value. Therefore, the total number of users who match based only on the fourth value of the first attribute is $\sum_j \sum_i E_{4,i,j}$. The summation involves all the values for each unrevealed (i.e., don't care) attribute in **E**. **E** is calculated using the database of user profiles. This database for the set of application users is assumed to be known to the application server as users who sign up to social computing applications usually fill out a profile. $\mathbf{E}_{k_1,\ldots,k_m}$ represents the number of matching users based on the set of values $\{k_1,\ldots,k_m\}$. The matching users are found by searching the database of user profiles. To simplify the presentation, we assume that the database contains information about all the users that have ever used the application. We also assume that for any revision of the profile attributes carried out by the application provider, a model is first developed to map old attributes to new attributes.

A *list of values* for a given attribute (see Fig. 1) is a one-dimensional array of size identical to the number of this attribute's possible values; each element returns the indices of **E**'s nonzero elements corresponding to the respective attribute value. For example, for 'female' as the value of the 'gender' attribute, the 'list of values' element contains a pointer to a one-dimensional array hosting the indices of all elements in **E** that fall under female (their other m-1 attributes can take any value) and contain a nonzero value. Obviously, the size of the latter one-dimensional array is always less than the number of users. If the k^{th} attribute has J_k possible values, the k^{th} list of values will need at most $\log_2(J_k)$ bits for addressing. Long attribute values (e.g., names) can be distinguished based on their first few characters. In practice, the m-dimensional array **E** should be built to its full extend only if each attribute can have many values. For example, in order to drastically reduce the storage space taken by the aforementioned pointers, two distinct (m-1)-dimensional arrays may be created for gender, one each for 'female' and 'male', respectively. Also, value combinations of different attributes that rarely occur could be combined into a "single hybrid attribute" in the space of E. We will talk more about the required storage space in the subsequent subsection. However, since the main purpose of this paper is to demonstrate the viability of the proposed identity-protection process, the topic of reducing further the required storage space is not treated here. It will be the target of future work.

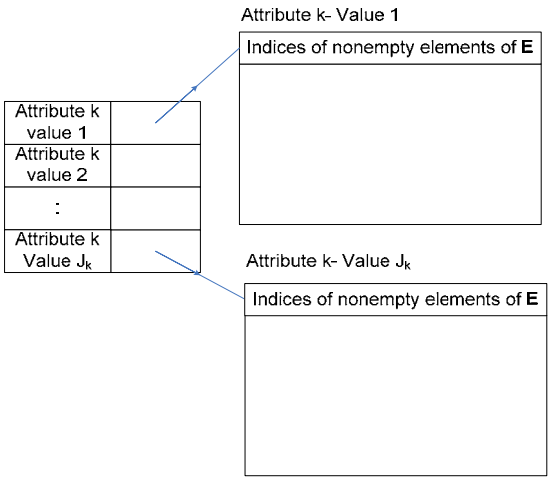

Fig. 1. List of values for attribute k

The algorithm takes the following steps and its flow chart is depicted in Fig. 2. ***S*** is an empty set and **G** is an empty one-dimensional array at the start of each communication session.

1. Every time ***A*** decides to reveal an attribute, q_j, **if** it is not a linkable attribute, then reveal q_j. **Else**,
2. **If** it is an *identifying attribute*, $\boldsymbol{S} \cup \{q_j\}$ is an identity-leaking set so warn user ***A***. **Else,**
3. **If G** is empty, find the array of indices of the nonzero elements of **E** that relate to the value of q_j from the corresponding 'list of values'. Set **G** equal to this array.
4. **Else** eliminate the indices of **G** that do not correspond to the values of $\boldsymbol{S} \cup \{q_j\}$.
5. **If** the length of **G** is larger than the sufficiency threshold, reveal q_j and set $\boldsymbol{S}=\boldsymbol{S} \cup \{q_j\}$.
6. **ElseIf** the length of **G** is less than ***A***'s desired degree of obscurity, $\boldsymbol{S} \cup \{q_j\}$ is an identity-leaking set. Warn the user, and if ***S*** is empty set **G** to empty.
7. **Else** read the values of all elements of **E** whose indices are stored in array **G**.
8. Let V equal the sum of all so obtained values. Then, derive $P_c(i)$ from Equation **(3)** and calculate the user's anonymity level using Equation **(2)**.
9. If the level of anonymity is equal to or less than its threshold, $\boldsymbol{S} \cup \{q_j\}$ is an identity-leaking set. If ***S*** is empty, set **G** to empty.
10. Else, reveal q_j and set $\boldsymbol{S}=\boldsymbol{S} \cup \{q_j\}$.

Steps 2 and 6 in the above algorithm take advantage of entropy property (c) to decide that $\boldsymbol{S} \cup \{q_j\}$ is an identity-leaking set. Step 4 takes advantage of property (d) to determine that the set of new indices (corresponding to $\boldsymbol{S} \cup \{q_j\}$) is a subset of old indices (corresponding to ***S***). In step 5, since the value of each nonzero element of **E** is equal to or higher than one, the number of users who match q_j is equal to or more than the size of this array. According to property (b), this revelation is safe. Finally, $P_c(i)$ can be easily calculated in step 8 as it is known what probabilistic value an element refers to. Another advantage of this algorithm relates to situations where rich or completely filled out profiles are not available for all community members and calculations are done based on the available subset of them. In this case, since the profile owners form a subset of a bigger community, based on property (a) a previously safe revelation remains safe. Only the false positive rate may increase, which may result in false warnings.

B. Computational Complexity of Algorithm II

When ***A*** is about to reveal a first attribute to ***B***, the first step is to search the 'list of values' array for this attribute to locate the to-be-revealed value. The k^{th} list of values has J_k entries, where J_k is the number of possible values of the k^{th} attribute. The worst case complexity of this step with binary search is $O(\log(J_k))$ for a fixed number of characters in the value. Since array **G** for each new revelation is a subset of array **G** for the previous revelation, this search is only performed for the first revelation. A decision can be immediately made if the size of **G** is more than the sufficiency threshold T or less than the desired degree of obscurity **U**. In the worst case, the size of **G** is less than T, but more than **U**. In this case, all corresponding nonzero elements of **E** are read using array **G** with indirect addressing and are added to calculate V. Then, their probabilities have to be added to calculate $P_c(i)$. Since the number of these nonzero elements is less than T, the worst case complexity of this step is $O(T)$. In summary, the worst case complexity of processing ***A***'s first revelation to ***B*** is $O(\log(J_k))$ and after that it becomes $O(T)$. This means that the computational complexity does not necessarily increase with the total number of users and most of the time its order is that of a rather small number T. For a large community of users, this maximum complexity is substantially less than the maximum complexity of Algorithm I.

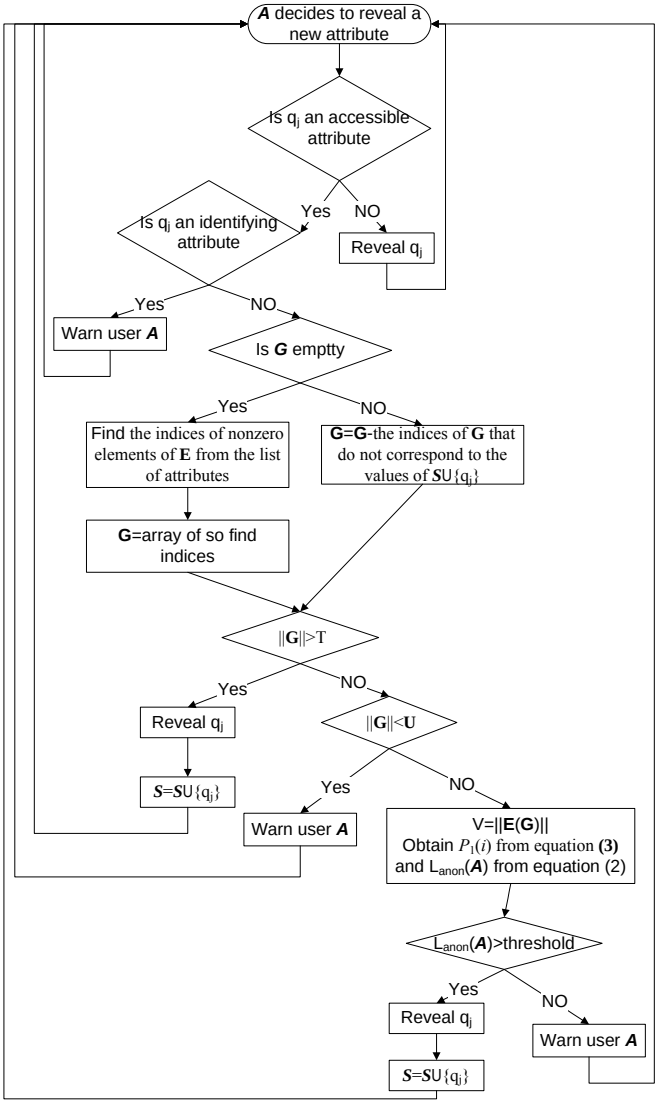

Fig. 2. Flow chart of Algorithm II.

This algorithm, however, takes more space on the disk than Algorithm I. The lists of values for the m attributes have a total of $\sum_{k=1}^{m} J_k$ rows and the size of the m-dimensional array **E** is $\prod_{k=1}^{m} J_k$ without array compaction. In a social profile with 10 *general* and *probabilistic linkable* profile attributes, and an average of 20 distinct values for each attribute, the length of the profile list will be 200 entries. The size of **E** will be 20^{10} Bytes or 10.24 TeraBytes, assuming that each attribute value uses a single byte (some attributes, like name, may actually require several bytes as explained before, while others, like sex, may require a single bit). An array of this size can be stored in the disks of a rather small PC cluster. However,

firstly, only up to T elements of **E** are read each time, using indirect addressing. Secondly, at most N elements of **E** have nonzero values, which means **E** is a very sparse array. Sparse arrays can be compressed to substantially reduce the storage space. The topic of advanced compression schemes is not treated in this paper since our main objective is to demonstrate the viability of our automated privacy-protection technique that centers on information entropy.

C. *Updating the Arrays*

If a user just starts or ends a communication session or decides to make certain attributes of his profile public or private, arrays do not need to be updated. If a new user joins the application and fills out a profile, then his profile is added to the database of profiles. If he fills out all m accessible profiles, then m elements in **E** are increased by one. The indices of these m elements are known, which makes the update very fast. If an element was previously zero, it has to be added to the list of nonzero elements.

If a user changes the value of an attribute, the values of two elements in **E** have to be updated; the element that corresponds to the old values of his m attributes decreases by one, and the element that corresponds to the new values of his m attributes increases by one. Since the indices of both elements are known, this update is very fast. If the former element changes to zero, it has to be removed from the array of nonzero elements through the list of attributes and if the latter element was previously zero, it has to be added to the list of nonzero elements. These updates are not time-sensitive and can be performed in the background.

Although adding or removing attributes for all users by the application provider does not usually happen in practice, these tasks can be handled by modifying the arrays. Removing an attribute is equivalent to projecting the m-dimensional array **E** to m-1 dimensions by adding the values of all the elements along the m^{th} dimension. Adding an attribute is equivalent to adding a dimension to **E** and requires N comparisons associated with the new attribute.

VII. Delay Analysis Of Simultaneous Communication For Many Users

The above complexity analysis deals with the worst case assuming one *member of* a pair of inter-personal communication partners during their session. Here, we try to estimate the average delay of making a decision about the safety of revealing an attribute just after a user decides to reveal its value. We assume a large community of registered application users, many of whom could be communicating at the same time. We explore this issue for a community similar to that of our downtown campus where we conducted our laboratory experiment mentioned in Section II.

Similar to common models for costumer services, such as call centers, networks, telecommunications, and server queueing, we assume that users arrive at the system according to a Poisson distribution with mean λ and spend an exponentially distributed chat-time with mean $1/\mu$ in the system. Since the users cannot be blocked or dropped, they form an M/M/∞ queuing system [32] in which the number of users in the system follows the Poisson distribution with the mean $N_s=\lambda/\mu(1-\lambda/\mu)$ [32] (all of the parameters used in this paper are listed in Appendix).

As we discussed above, the computational complexity of the first revelation is $O(\log(J_k))$. This means that the worst case processing time is $c_1^* \log(J_k)+c_2T$, where c_1 and c_2 are small constant time measures. c_1 is the maximum time needed to compare one attribute value against another and c_2 is the time needed for reading an element from the array, adding it to another number, and multiplying it by a number. This processing time is less than $c^*(\log(J_k)+T)$, where $c=\max\{c_1,c_2\}$. We assume the maximum time of $c^*(\log(J_k)+T)$ for the worst case delay analysis. The probability that x users reveal their first attribute during the same millisecond interval can be approximated with the Poisson distribution of mean λ for the worst case where, any user who starts a session, reveals at least one linkable attribute right after joining the system.

The worst case computational complexity of all other revelations is $O(T)$. We again assume the worst case where processing the safety of revealing each attribute always takes a processing time of cT, where $c=\max\{c_1,c_2\}$.

The probability that x revelations have to be processed during the same millisecond interval, p(x), is the probability that at least x users are currently present and communicating in the system and x users among them decide to reveal a linkable attribute. We consider the worst case here too where all profile attributes are linkable and all x users need to be processed simultaneously. If the probability that a user present in the system reveals an attribute during any given time unit interval is α, the probability p(x) is obtained as follows.

$$p(x)=\sum_{i=x}^{N}\binom{i}{x}\alpha^x(1-\alpha)^{i-x}\frac{N_s^i}{i!}e^{-N_s}=\frac{e^{-N_s}.(N_s\alpha)^x}{x!}\sum_{i=x}^{N}\frac{(N_s(1-\alpha))^i}{i!}$$

$\approx$(for a large number of users)

$$\frac{e^{-\alpha N_s}.(\alpha N_s)^x e^{N_s(1-\alpha)}}{x!}=\frac{(\alpha N_s)^x}{x!}e^{-\alpha N_s}$$

Therefore, the number of revelations that need to be processed in the same millisecond follows the Poisson distribution with mean $\alpha N_s=\alpha\lambda/\mu(1-\lambda/\mu)$.

Assuming that the first revelation is made independent of further revelations, the total number of simultaneous revelations that need to be processed follows the Poisson distribution with mean $\alpha N_s+\lambda$. Consequently, assuming one server, the revelations to be processed form an M/G/1 queuing system [33]. The average waiting time in such a system is obtained from Equation **(5)**[33]: Ave(*waiting-time*)=

$$\frac{(\alpha N_s+\lambda)*[\text{Ave}(\textit{processing-time})^2+\text{VAR}(\textit{processing-time})]}{2[1-(\alpha N_s+\lambda)\,\text{Ave}(\textit{processing-time})]} \quad \textbf{(5)}$$

On average, $\lambda/(\alpha N_s+\lambda)$ of the revelations are the first revelation and take $c^*(\log(J_k)+T)$ milliseconds, while the rest take cT milliseconds. Therefore, the average and variance of processing time are obtained as follows.

Ave(*processing-time*)=

$[\lambda/(\alpha N_s+\lambda)][cT+c^*\log(J_k)]+[\alpha N_s/(\alpha N_s+\lambda)](cT)$. **(6)**

VAR(*processing-time*)=

$$\int_{-\infty}^{+\infty}\tau^2([\alpha N_s/(\alpha N_s+\lambda)]\Delta(\tau-cT) + [\lambda/(\alpha N_s+\lambda)]\Delta(\tau-cT-c^*\log(J_k)))\,d_\tau=$$

$$\frac{\lambda\,(cT+c^*\log(J_k))^2+\alpha N_s\,(cT)^2}{\alpha N_s+\lambda} \quad \textbf{(7)}$$

The average waiting time is obtained by substituting the average and variance of the processing time from Equations **(6)** and **(7)** in Equation **(5)**. The total delay which includes queuing delay and processing time equals:

Ave(total-delay)=Ave(*waiting-time*)+Ave(*processing-time*) **(8)**

We used the data from our laboratory experiment to simulate how this delay changes by the number of users in the system N_s which is equal to $\lambda/\mu(1-\lambda/\mu)$ and the average duration of a communication session $(1/\mu)$. Based on the experimental data, the maximum desired degree of obscurity was 5 and the probability of guessing gender and ethnicity were respectively 0.104 and 0.052 as mentioned in Section II. Hence, the sufficiency threshold was equal to 5.6. The probability of revealing an attribute T in any millisecond was $3.8*10^{-7}$. We assumed the average value of J_k to be equal to 20 which fits our data and other rich profiles. Fig. 3 shows the average queuing delay that users experience due to the presence of other users versus the average number N_s of users in the system who are communicating simultaneously and the average duration of communication session. Fig. 4 shows the average of total delay for processing the safety of their intended revelation versus the average number of users in the system and the session duration. The average queueing delay (waiting time) in the figure is shown in seconds and the average duration of the communication session is shown in minutes. The variable c is expressed in microseconds.

In a single network, the variable c is on the order of microseconds if we do not consider the need for remote array accesses. Since λ equals $\mu N_s/(1+N_s)$, when the number of simultaneous communications (N_s) and session duration (μ) are low, first revelations represent a high percentage of the overall revelations. Therefore, the average processing time and, consequently, the average total delay is higher. When N_s is sufficiently high, the total delay increases with an increasing N_s. However, as seen in the figures, the total delay in all cases is on the order of microseconds for up to a million users. This delay should not be noticeable by human users, which means that revelations involving many users can be processed in a very time-efficient manner.

VIII. Conclusion

Partial disclosures of user attributes happen in various scenarios, such as publishing micro-data for legal, business or research purposes, and revealing profile attributes in social networking sites, social matching systems and computer-mediated communication. Anonymity is a major concern in such scenarios as revealed data might be linked back to owners. We highlighted the drawbacks of previous anonymity measurement methods and, in particular, specified why more realistic anonymity estimations are needed in the realm of social computing. We then proposed a framework for estimating the level of anonymity when user attributes are partially shared.

In synchronous online communications, an anonymity protection system needs to be implemented to protect users. Such systems should be able to estimate the risk in acceptable time. We used basic properties of information entropy and proposed an algorithm for online estimation of identity risk. Analysis of its computational complexity and delay based on experimental data shows that a large number of users can be effectively processed and protected simultaneously with unnoticeable delays. We analyzed the worst case computational complexity for a single server as this work lays out the foundation for instantaneous protection and complexity reduction. Using distributed servers which will be the focus of future work speeds ups the calculations and reduces the delay even further.

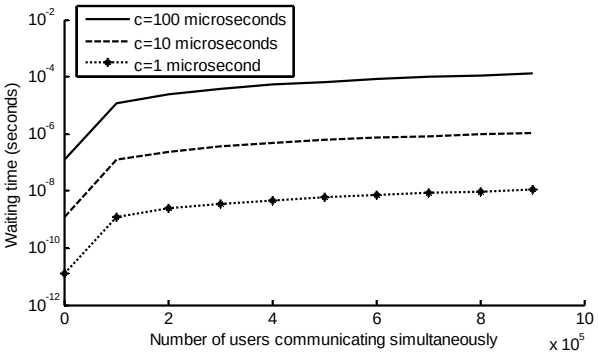

Fig. 3. The average waiting (queueing) time experienced by a user for processing the safety of a revelation in simultaneous communications.

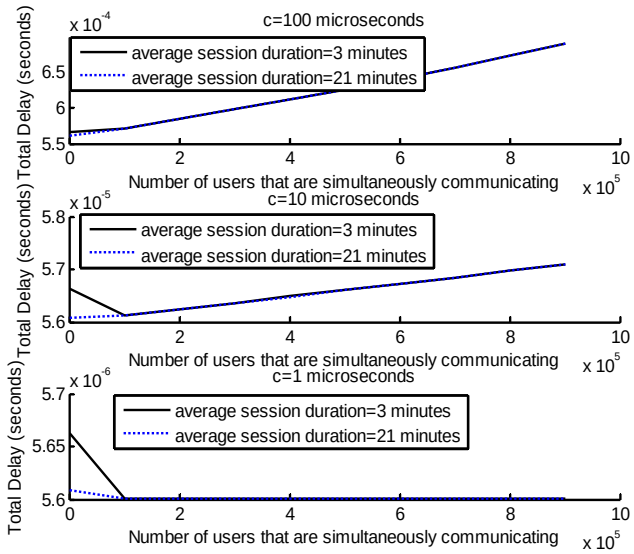

Fig. 4. The average total delay experienced by a user for processing the safety of an application when many users are communicating simultaneously.

Our proposed anonymity protection system was not implemented and tested for a large number of users. Large scale implementations in future work may reveal further complications. Furthermore, we did not aim to find an optimal solution to this problem, but to find an algorithm that reduces the complexity substantially, as compared to the brute-force algorithm. Finally, our estimation of processing time and delay was based on our experimental data. We considered one server for the worst case and assumed rich and dynamic user profiles that suffice for today's common social applications and micro-datasets. Nevertheless, in an application with a very large number of users, many linkable attributes and highly dynamic profiles, distributed servers or even more efficient algorithms may be needed.

While new ways of data disclosure, especially in ubiquitous and social computing applications, are emerging

and growing rapidly, they currently miss a realistic estimation of a user's anonymity level and an efficient system for protecting anonymity against identity inferences. Implementation of such systems would greatly improve user privacy and anonymity protection, and would be a major step forward towards successful deployment of next generation ubiquitous social computing systems.

REFERENCES

[1] S. Whittaker, Jones, Q., Nardi, B., Creech, M., Terveen, L., Isaacs, E., and Hainsworth, J. , "ContactMap: Using Social Networks to Access and Organize Communication," in *The ACM's Conference on Computer Supported Cooperative Work, Video paper.*, 2002.

[2] S. Motahari, C. Manikopoulos, R. Hiltz, and Q. Jones, "Seven Privacy Worries in Ubiquitous Social Computing," in *ACM International Conference Proceeding Series; Proceedings of the 3rd Symposium on Usable Privacy and Security* 2007, pp. 171-172.

[3] E. Heinrich, "Electronic Repositories of Marked Student Work and their Contributions to Formative Evaluation," *Educational Technology & Society,*, vol. 7, pp. 82-96, 2004.

[4] A. Kobsa and J. Schreck, "Privacy through pseudonymity in user-adaptive systems," *ACM Transactions on Internet Technology (TOIT),* vol. 3, pp. 149 - 183, 2003.

[5] D. Doyle, "Trans-disciplinary Inquiry – Researching with Rather than Researching on," in *An Ethical Approach to Practitioner Research: Dealing with Issues and Dilemmas in Action Research*, A. Campbell and S. Groundwater-Smith, Eds.: Routledge 2007.

[6] S. Lodha and D. Thomas, "Probabilistic Anonymity," in *PinKDD Workshop (International Workshop on Privacy Security and Trust in KDD) with Knowledge Discovery and Data Mining*, 2007.

[7] L. Sweeney, "Uniqueness of Simple Demographics in the U.S. Population.," Technical Report LIDAPWP4, Laboratory for International Data Privacy, Carnegie Mellon University, PA 2000.

[8] L. Singh and J. Zhan, "Measuring Topological Anonymity in Social Networks," in *Proceedings of the 2007 IEEE International Conference on Granular Computing* 2007, p. 770.

[9] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu, "Approximation Algorithms for k-Anonymity1," *Journal of Privacy Technology,* vol. 20051120001, pp. 1-18, 2005.

[10] L. Sweeney, "k-anonymity: A Model for Protecting Privacy," *International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 10, pp. 557-570, 2002.

[11] M. Ackerman and L. Cranor, "Privacy Critics: UI Components to Safeguard Users' Privacy," in *SIGCHI Conference on Human Factors in Computing Systems (CHI 99)*, 1999.

[12] D. Hong, M. Yuan, and V. Y. Shen, "Dynamic Privacy Management: a Plugin Service for the Middleware in Pervasive Computing," in *ACM 7th International Conference on Human Computer Interaction with Mobile Devices & Services* 2005, pp. 1-8.

[13] M. Langheinrich, "A Privacy Awareness System for Ubiquitous Computing Environments," in *4th International Conference on Ubiquitous Computing (Ubicomp 2002)*, 2002, pp. 237-245.

[14] U. Jendricke, M. Kreutzer, and A. Zugenmaier, "Pervasive Privacy with Identity Management," in *Workshop on Security in Ubiquitous Computing - Ubicomp*, 2002.

[15] S. Motahari, S. Ziavras, M. Naaman, M. Ismail, and Q. Jones, "Social Inference Risk Modeling in Mobile and Social Applications," in *IEEE International Conference on Information Privacy, Security, Risk and Trust (PASSAT).* 2009, pp. 125-132.

[16] S. Motahari, S. Ziavras, R. Schular, and Q. Jones, "Identity Inference as a Privacy Risk in Computer-Mediated Communication.," in *IEEE Hawaii International Conference on System Sciences*, 2008, pp. 1-10.

[17] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for Web Transactions," *Communications of the ACM,* vol. 42, pp. 32-48, 1999.

[18] L. Sweeney, "Achieving k-Anonymity Privacy Protection Using Generalization And Suppression," *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems,* vol. 10, pp. 571-588, 2002.

[19] A. Machanavajjhala, J. Gehrke, and D. Kifer, "ℓ-Diversity: Privacy Beyond k-Anonymity," in *Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE 2006)*, 2006.

[20] S. Motahari, S. Ziavras, and Q. Jones, "Preventing Unwanted Social Inferences with Classification Tree Analysis," in *IEEE International Conference on Tools with Artificial Intelligence*, 2009, pp. 500-507.

[21] A. Narayanan and V. Shmatikov, "De-anonymizing Social Networks.," in *IEEE symposium on Security and Privacy*, 2009.

[22] A. Campan and T. M Truta, "Data and Structural k-Anonymity in Social Networks," in *Lecture Notes in Computer Science (Privacy, Security, and Trust in KDD)*: Springer Berlin /Heidelberg, 2009, pp. 33-54.

[23] B. Thuraisingham, "Privacy Constraint Processing in a Privacy-Enhanced Database Management System," *Data and Knowledge Engineering,* vol. 55, pp. 159-188, 2005.

[24] D. E. Denning and M. Morgenstern, "Military database technology study: AI techniques for security and reliability," 1986.

[25] M. Morgenstern, "Security and Inference in Multilevel Database and Knowledge-Based Systems," in *International Conference on Management of Data archive, Proceedings of the 1987 ACM SIGMOD international conference on Management of data* 1987, pp. 357-373.

[26] M. Morgenstern, "Controlling Logical Inference in Multilevel Database Systems," in *IEEE Symposium on Security and Privacy*, 1988, pp. 245-255.

[27] A. Serjantov and G. Danezis, "Towards an Information Theoretic Metric for Anonymity," in *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*, 2002.

[28] C. Diaz, S. Seys, J. Claessens, and B. Preneel, "Towards Measuring Anonymity," in *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*, 2002.

[29] G. Tóth, Z. Hornák, and F. Vajda, "Measuring Anonymity Revisited," in *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*, 2004, pp. 85-90.

[30] S. Jajodia and C. Meadows, *Inference Problems in Multilevel Secure Database Management Systems*. Los Alamitos, California, USA IEEE Computer Society Press, 1995.

[31] C. E. Shannon, "Prediction and entropy of printed English," *The Bell System Technical Journal,* vol. 30, pp. 50-64, 1950.

[32] D. Bertsekas and R. Gallager, *Data Networks*: Prentice Hall, 1987.

[33] A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*: Addison Wesley, 1993.

Sara Motahari received her Ph.D. degree in Electrical Engineering from the New Jersey Institute of Technology (NJIT) in 2010, Newark, the M.Sc. degree in digital communications from Chalmers University of Technology, Sweden in 2005, and the B.Sc. in Electrical Engineering from Sharif University of Technology, Iran in 2003. She also worked as researcher at the University of Toronto, Canada in 2004-2005. She received the Ross Memorial Scholarship, Hashimoto Scholarship and Phonetel Fellowship from NJIT, and Adlerbertska Hospitiefonden Scholarship from Chalmers. She is also a Google's Anita Borg Scholarship finalist.

Sotirios G. Ziavras is a Professor of ECE at NJIT, and the Director of its Computer Architecture and Parallel Processing Laboratory. He received the Diploma in EE from the National Technical University of Athens, Greece in 1984 and the D.Sc. degree in Computer Science from George Washington University in 1990. He was with the Center for Automation Research at the University of Maryland, College Park, from 1988 to 1989. He was a visiting Assistant Professor of ECE at George Mason University in Spring 1990. He joined NJIT in Fall 1990. He received an NSF Research Initiation Award in 1991, as well as many other grants from NSF, DoE, etc. He has served as an Associate Editor of the Pattern Recognition journal and serves regularly as a member of Conference Program Committees. He has authored about 150 papers.

Quentin Jones is an Associate Professor in the Department of Information Systems, New Jersey Institute of Technology (NJIT). He is the founder and director of NJIT/NSF's SmartCampus initiative exploring how mobile social computing can be used to increase community connectivity. Before coming to NJIT, he was a Research Scientist in the Human Computer Interaction Group at ATT Labs-Research, USA. He holds a Ph.D. in Information Systems from the University of Haifa, Israel, a M.P.H. (Bio Statistics) from the Department of Medicine, University of Sydney, Australia and a B.A.Hons in Psychology from University of Sydney, Australia.