To Do or Not To Do: The Dilemma of Disclosing Anonymized Data

Laks V.S. Lakshmanan, Raymond T. Ng, Ganesh Ramesh
University of British Columbia
{laks,rng,ramesh}@cs.ubc.ca

ABSTRACT

Decision makers of companies often face the dilemma of whether to release data for knowledge discovery, vis a vis the risk of disclosing proprietary or sensitive information. While there are various "sanitization" methods, in this paper we focus on anonymization, given its widespread use in practice. We give due diligence to the question of "just how safe the anonymized data is", in terms of protecting the true identities of the data objects. We consider both the scenarios when the hacker has no information, and more realistically, when the hacker may have partial information about items in the domain. We conduct our analyses in the context of frequent set mining. We propose to capture the prior knowledge of the hacker by means of a *belief function*, where an educated guess of the frequency of each item is assumed. For various classes of belief functions, which correspond to different degrees of prior knowledge, we derive formulas for computing the expected number of "cracks". While obtaining the exact values for the more general situations is computationally hard, we propose a heuristic called the *O-estimate*. It is easy to compute, and is shown to be accurate empirically with real benchmark datasets. Finally, based on the O-estimates, we propose a recipe for the decision makers to resolve their dilemma.

1. INTRODUCTION

Privacy-preserving data mining has attracted a lot of attention in recent years. The primary objective is to strike a balance between two opposing forces: the urge to mine data to gain further insights and knowledge, versus the responsibility to protect the privacy and identities of individuals (e.g., patients, travelers). As well, these two forces manifest themselves in similar ways in business situations. Decision makers of companies often face the dilemma of whether to release data for knowledge discovery, given the risk inherent in disclosing proprietary/sensitive information (e.g., identities of the better-selling products) to the public, particularly to potential competitors. Consider the following scenarios.

> **Mining as a service**: A company, with insufficient expertise in data mining, wants to hire a data mining service provider to mine its data. While there is legal protection (e.g., non-disclosure agreements), the company still legitimately worries about its data being leaked out somehow, and such leakages are often hard to detect.
>
> **Mining for the common good**: A company may want to participate in a consortium, which involves sharing of data. The motivation of pooling data together is to gain in scale and in diversity (e.g., geographical variations, demographic differences). The dilemma is that partners of the consortium may already be, or one day become, competitors, or work with competitors.

SIGMOD 2005 June 14-16, 2005, Baltimore, Maryland, USA.
Copyright 2005 ACM 1-59593-060-4/05/06 $5.00.

One common approach to handling these situations is to release *transformed* data; and amongst the well-known transformation techniques, *anonymization* is arguably the most common. That is, if the objects in the domain are originally identified by their social security number, product number, etc., these objects are now identified by a generated number, typically as simple as a positive integer. Compared with other transformation techniques, anonymization is simple to carry out, as mapping objects back and forth is easy. Another advantage of anonymization is that it does not perturb data characteristics. For both the above scenarios, changing the data characteristics may affect the outcome too much that it defeats the original purpose of releasing the data. Note that there are alternatives and more sophisticated techniques (e.g. k-anonymization); and we are *not* recommending that anonymization be used to replace those more sophisticated approaches. Our study here is simply based on the observation that anonymization is already widely used in practice. One prime example is clinical trial studies for new drugs in the medical and pharmaceutical domain. Even though the US Food and Drug Administration guidelines are well-known to be strict, anonymization (or de-identification) is still considered adequate in the clinical trial circles for protecting the privacy of the patients participating in the studies.

The immediate question, however, is: *Just how safe is the anonymized data?* where "safe" is interpreted as protection of the identities of the objects. Hereafter, we use the term data "owner" to refer to the party that owns the original data, and the term data "hacker" to refer to the party that tries to illegally reveal the true identities of anonymized objects. We use the term "cracks" to refer to the objects whose identities are revealed by the hacker. An answer to the above question of safety is complicated by an often overlooked issue: *How much partial information does the hacker have?* The assumption that there is no partial information out there may be unrealistic in this internet era. Furthermore, as illustrated in the two scenarios above, a hacker may be from a competitor or a rival company. The hacker may use his/her knowledge from his/her own data or data from similar sources, to infer and gain knowledge about the anonymized data. It has been recognized that incorporating partial knowledge in analyzing security is an open problem. For instance, Yang and Li consider prior knowledge captured in functional dependencies in secure XML publishing [27]. Thus, the safety question becomes: *Just how safe is the anonymized data in the presence of partial information?*

One source of partial information is when the attacker has access to similar data. In this paper, we develop a framework to answer the safety question by investigating various abstractions of similarity. Using frequent set mining as an illustrative example, we model the attacker's partial information in the form of a *belief function* which represents the attacker's educated guess about the frequency of each item from such similar data. For each of these abstractions, we explore how to determine the percentage of cracks when the attacker has access to the anonymized database and partial information. More specifically:

1. We first analyze two extremes of similarity (Section 3): When the attacker has access to no data and when the attacker has access to "almost" identical data. This is captured in the form of *ignorant* and *compliant point-valued* belief functions respectively. For both cases, we present formulas for determining the exact percentage of cracks and observe that even in the presence of identical data, the distribution itself may provide some degree of protection.

2. While the extreme cases may be highly unlikely to occur in practice, their analysis, nevertheless, is useful in practical scenarios. We model the third abstraction of similarity where the partial information is captured in a *compliant interval(-valued)* belief function (Section 4), which specifies a range of frequency values for each item. Each range is assumed to contain the true frequency (hence compliant). This abstraction models the case when the attacker has information about the true frequency of every item but is not sure about its exact value (hence a range). While this abstraction of similarity is more general than the extremes, it is still conservative as the attacker may not have compliant information about *all* the items in the domain. The fourth abstraction of similarity captures this most general notion of the attacker's partial information: α-compliant belief functions where the belief function only guesses the right ranges for a fraction α of items ($0 \leq \alpha \leq 1$).

3. Not surprisingly, determining the exact percentage of cracks for interval belief functions turns out to be a hard problem, and even known estimation algorithms have too high a complexity to be practical. For specific structures it is possible to obtain exact formulas for the expected number of cracks and for one such special case of interval belief functions whose structure resemble a chain, we derive exact formulas for computing the percentage of cracks. To overcome the complexity of estimating the percentage of cracks for a general interval belief function, we develop a heuristic, which we call the *O-estimate*(Section 5). We show that the *O-estiamte* is practically accurate for the various similarity abstractions, by evaluating its accuracy through a systematic set of experiments on benchmark data(Section 7).

4. Consequently, using these tools and similarity abstractions, we propose a heuristic recipe, using which a data owner can determine the risk of releasing anonymized data (Section 6). We show the effectiveness of this recipe (Section 7.3) by studying the effect of *compliancy* on the percentage of items that are cracked, which is used by the data owner to resolve the dilemma of releasing the anonymized data. We also demonstrate how a data owner can simluate the attacker's prior knowledge by building belief functions from samples of the original dataset (Section 7.4).

5. Finally, in Section 8, we make observations about how the proposed analyses and O-estimates can be generalized from frequent set mining to other data mining tasks.

1.1 Related Work

A majority of work in *disclosure limitation* like [17, 11, 9] focus on applying statistical disclosure limitation methods for categorical and microarray datasets. All these methods use techniques like *cell suppression*, *data swapping*, *rounding*, *sampling* and generation of *synthetic data* as a means of achieving statistical disclosure control. While limiting disclosure, they may perturb the characteristics of the original dataset. In any event, this paper does not necessarily advocate anonymization as the best method to limit disclosure. Rather, based on the observation that anonymization is one of the most common approaches, our work gives due diligence to the analysis of the risk of releasing anonymized data. Furthermore, our analysis addresses the often overlooked issue of considering whatever partial information that the hacker may possess.

Privacy has been studied in the context of association rule mining [26, 10, 15]. In [10], Evfimievski et al. propose a framework for mining association rules in which the data items in transactions are randomized to preserve privacy of individual transactions. They analyze the nature of privacy breaches caused by using the association rules discovered from this database and propose randomization operators to limit such breaches. The problem of hiding association rules by transforming the input database is studied by Verykios et al. in [26]. The authors are interested in modifying the input database such that a given set of associations is hidden in the transformed database. Techniques like removing items from transactions, adding new items to transactions are used. While the problem of transforming data shares some commonalities with our work, they modify the frequency of items in the original data. Furthermore, their studies do not deal with the possible presence of prior knowledge possessed by the hacker.

In [4], Agrawal and Srikant propose an approach for privacy preserving classification that is based on mining on perturbed data, with the perturbed distribution closely matching the real distribution. Furthermore, Agrawal and Aggarwal in [5] discuss an expectation maximization algorithm which provides robust estimates of the original distribution based on perturbation and provides some interesting results on the relative effectiveness of different perturbing distributions in terms of privacy. In [12], Iyengar uses the approaches of suppression and generalizations to satisfy privacy constraints. The tradeoff between privacy and information loss in the specific context of data usage is considered, and the search for the optimal tradeoff is considered as an optimization problem for which a genetic algorithm framework is used to search for a solution. In [2], Aggarwal and Yu use an approach based on *condensation groups* to model indistinguishability of data records and use it to create anonymized data which has similar characteristics to the original multidimensional dataset and apply it to the classification problem. Finally, the k-anonymity model studied in [22, 23, 3, 19] uses domain generalization hierarchies in order to transform and replace each record value with a corresponding generalized value. While this model is similar in spirit to our notion of anonymization, it perturbs the data (in the simplest case, making more than one item indistinguishable from each other), thus making reconstruction of patterns difficult. In sum, all these studies focus on perturbing the data so that the results of data mining the perturbed data remain similar to the original data. Our work here is very different in that it gives an analysis of the risk of releasing anonymized data.

The security problem in statistical databases deals with protecting the database from returning information about an individual or answering a sequence of queries from which individual information can be deduced, where the statistical databases allow only queries that retrieve statistical information (like sum, average, median) of certain subsets of records. The various approaches used by the security control methods are categorized in the survey by Adam and Wortmann [1]. An evaluation of three data perturbation methods to protect the confidentiality of numerical attributes is presented in [18].

2. ANONYMIZATION, BELIEF FUNCTIONS

2.1 Anonymization

The original domain is a non-empty universe of items $\mathcal{I}$. For the sequel, we assume $|\mathcal{I}| = n$. A database $\mathcal{D}$ is a sequence of transactions $\langle T_1, \ldots, T_m \rangle$. Each transaction is a non-empty subset of $\mathcal{I}$. As in [6], the frequency of an item $x \in \mathcal{I}$ is the fraction of transactions in $\mathcal{D}$ that contain x.

Let $\mathcal{J}$ be an anonymized domain of items such that $|\mathcal{J}| = |\mathcal{I}|$ and $\mathcal{J} \cap \mathcal{I} = \emptyset$. An *anonymization mapping* is a bijection from $\mathcal{I}$ to $\mathcal{J}$. Transactions are anonymized by replacing each item in the transaction with its anonymized item. Databases are anonymized by anonymizing each transaction. Note that the anonymization mapping is applied uniformly across all the transactions in the database. Hence, if 1 is anonymized to $1'$,this happens in every transaction in the database. Figure 1 shows a simple example, referred to hereafter as the BigMart example. For the rest of the paper, the primed item, $x' \in \mathcal{J}$, will be used to denote the anonymized item corresponding to $x \in \mathcal{I}$.

Big Mart's Database

TID	Transaction
1	{1,2,3}
2	{1,2,3,4}
3	{4,6}
4	{3,4,5,6}
5	{5,6}
6	{6}
7	{1,2}
8	{1,3,4}
9	{1,3,5}
10	{2,4,6}

I = {1,2,3,4,5,6}

After Anonymization

TID	Transaction
1	{1',2',3'}
2	{1',2',3',4'}
3	{4',6'}
4	{3',4',5',6'}
5	{5',6'}
6	{6'}
7	{1',2'}
8	{1',3',4'}
9	{1',3',5'}
10	{2',4',6'}

J = {1',2',3',4',5',6'}

Figure 1: Example Anonymized Database

2.2 Belief Functions

In this paper, we assume that the hacker knows the domain $\mathcal{I}$ and captures this prior knowledge about the domain in a *belief function*, β. A belief function maps each item x in $\mathcal{I}$ to an interval $[l, r]$, modeling the belief that the frequency of x in the database is in the range $[l, r]$ where $0 \le l \le r \le 1$.

Two special belief functions are considered which represent extremes in the hacker's prior belief. When a hacker has *no* knowledge of the frequency of any item in $\mathcal{I}$, each item in $\mathcal{I}$ maps to $[0, 1]$. In this case, the hacker is *ignorant* of the frequency of any item in $\mathcal{I}$ and the belief function is called an *ignorant* belief function.

The other extreme is when the hacker has *exact* knowledge of the frequency of every item in $\mathcal{I}$, with each interval $[l, r]$ essentially becoming a point value in the interval $[0, 1]$. This belief function is called a *point-valued* belief function. A belief function is called an *interval* belief function at least one item's belief interval is a true range, i.e., $l < r$.

A belief function is *compliant*, if for every item $x \in \mathcal{I}$, the range $[l, r]$ contains the true frequency of x. A belief function is α-compliant if only a fraction α $(0 \le \alpha \le 1)$ of items satisfy the requirement of true frequency containment. For simplicity, whenever $\alpha = 1$, we simply refer the belief function as compliant.

Figure 2 shows four belief functions f, g, h and k over Big Mart's domain: f is a compliant point-valued belief function; g is an ignorant (and compliant) belief function; h is a compliant interval belief function; and k is 0.5-compliant, as it guesses wrong on the first three items.

Belief Functions f, g, h, k:

Compliant Point−Valued	*Ignorant*	*Compliant Interval*	*0.5−Compliant Interval*
f(1) = 0.5	g(1) = [0,1]	h(1) = [0,1]	k(1) = [0.1,0.4]
f(2) = 0.4	g(2) = [0,1]	h(2) = [0.4,0.5]	k(2) = 0.5
f(3) = 0.5	g(3) = [0,1]	h(3) = 0.5	k(3) = [0.1,0.3]
f(4) = 0.5	g(4) = [0,1]	h(4) = [0.4,0.6]	k(4) = [0.4,0.6]
f(5) = 0.3	g(5) = [0,1]	h(5) = [0.1,0.4]	k(5) = [0.1,0.4]
f(6) = 0.5	g(6) = [0,1]	h(6) = 0.5	k(6) = 0.5

Figure 2: Examples of Belief Functions

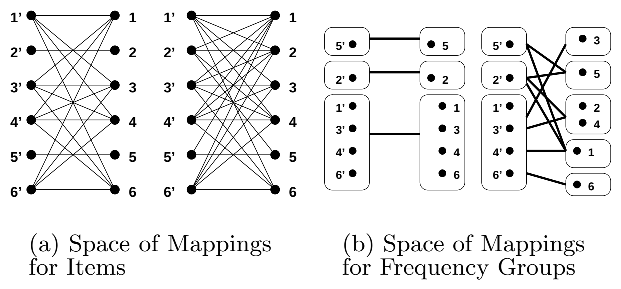
2.3 Inference and Consistency Assumptions

A hacker uses a *crack mapping*, $C : \mathcal{J} \to \mathcal{I}$, to identify anonymized items. We assume the hacker only uses 1-1 crack mappings, i.e., (s)he assigns exactly one item to each anonymized item. An item $x \in \mathcal{I}$ is said to be *cracked* by C whenever the mapping correctly maps the anonymized item x' to x. The question here is which crack mappings are used by the hacker. We assume that the hacker uses his/her belief function to derive crack mappings. Specifically, let x' be an anonymized item and let $F(x')$ be its observed frequency in the database. Then, x' is mapped by the hacker to only those items in $\mathcal{I}$ whose belief interval contains $F(x')$. Such mappings are called *consistent mappings*, as they are consistent with the prior knowledge a hacker has about $\mathcal{I}$. Henceforth, mapping always means a *consistent* mapping.

Consider for example the belief function h shown in Figure 2. By analyzing the anonymized data, the hacker will surely find out the frequencies of $1'$, $2'$, $3'$, $4'$, $5'$ and $6'$ are respectively 0.5, 0.4, 0.5, 0.5, 0.3 and 0.5 (i.e., specified precisely by the compliant, point-valued belief function f). What can be the true identity of $1'$ then? To be consistent with the belief function h, $1'$ can be mapped to 1, 2, 3, 4 and 6; $h(5) = [0.1, 0.4]$ is the only range not containing 0.5. Similarly, the observed frequency of $2'$ is 0.4, and $2'$ can be mapped to 1, 2, 4 and 5.

Given a belief function and an anonymized database, the space of all consistent crack mappings can be represented by a bipartite graph $G = (\mathcal{J} \cup \mathcal{I}, E)$, where each mapping is a perfect matching in G. The edge (x', y) in G denotes the fact that a mapping used by a hacker can map the anonymized item x' to the item $y \in \mathcal{I}$. Thus, for the ignorant belief function, we have a complete bipartite graph. Note that, if a belief function is compliant on item x, then the edge (x', x) is present in the bipartite graph.

Figure 3(a) shows the bipartite graphs corresponding to the belief functions f and h in Figure 2. For the latter, as discussed in the previous paragraph, 1' is connected to 1, 2, 3, 4 and 6; 2' is connected to 1, 2, 4 and 5; and so on.



(a) Space of Mappings for Items (b) Space of Mappings for Frequency Groups

Figure 3: Space of Consistent Mappings for f, h

Figure 3(b) shows an alternative presentation of the mappings of Figure 3(a) by grouping the items in the anonymized domain and the original domain and viewing mappings in terms of these groups. The anonymized items can be grouped based on their observed frequencies. On the other hand, the items in the original domain can be grouped based on the anonymized items that can map to them. Specifically, items x and y in $\mathcal{I}$ belong to the same group if $\{w' | (w', x) \in G\} = \{w' | (w', y) \in G\}$, where G denotes the bipartite graph. Consider the group mapping corresponding to the belief function h (of figure 2). Even though items 2 and 4 have different belief intervals, they belong to the same group as the same set of anonymized items can map to these two items.

Note that given a belief function and a corresponding bipartite graph, there may not exist a perfect matching. As a simple example, let the original domain be {1, 2} and the corresponding anonymized domain be {1', 2'}. The bipartite graph may map both 1' and 2' to 2, with no anonymized item mapped to 1. Throughout our analysis, situations like this are dealt with using α-compliancy (Section 5.3). Until then, we consider only compliant belief functions.

3. ANALYSIS OF THE TWO EXTREMES

To decide whether to release the anonymized data, the data owner needs to assess the risk of disclosure. Given the bipartite graph corresponding to the belief function, there are many possible crack mappings, giving rise to different numbers of cracks. Throughout this paper, we assume that each consistent crack mapping is equally likely. (In practice, the hacker may use additional prior knowledge to favor one crack mapping over another. We consider this beyond the scope of this work.) Hence, the risk of disclosure can be reasonably measured by the expected percentage or number of cracks. In this section, we provide formulas for computing the expected number of cracks in the presence of prior knowledge modeled by the ignorant belief function and the compliant point-valued belief functions. In the next section, the analysis will be generalized to interval belief functions.

3.1 Ignorant Belief Function

Let us suppose that the hacker is ignorant of the frequencies of items in $\mathcal{I}$. Hence, a hacker can map an anonymized item to *any* item in $\mathcal{I}$ and the space of mappings is a complete bipartite graph. The expected number of cracks in this case is the expected number of cracks in any mapping in this complete bipartite graph. The following lemma shows that the expected number of cracks in a mapping for a complete bipartite graph is 1.

LEMMA 1. *Let* $G = (\mathcal{J} \bigcup \mathcal{I}, E)$ *be a complete bipartite graph. Let* X *be a random variable that represents the number of cracks in a mapping obtained from* G. *Then the expected number of cracks is* $E(X) = 1$.

Proof Sketch: Let $X_{i'}$ be a random variable associated with any $i' \in \mathcal{J}$ which is 1 when i' maps to the correct item $i \in \mathcal{I}$ and 0 otherwise. The probability that anonymized item i' maps to i is $\frac{1}{n}$, where $|\mathcal{I}| = |\mathcal{J}| = n$. The expected value $E(X_{i'})$ is thus $\frac{1}{n}$. The random variable $X = \sum_{i' \in \mathcal{J}} X_{i'}$. Thus, $E(X) = E(\sum_{i' \in \mathcal{J}} X_{i'}) = \sum_{i \in \mathcal{J}} E(X_{i'}) = n \times \frac{1}{n} = 1$. $\square$

The implication of the lemma is that, as $|\mathcal{I}| = n$ gets bigger, the expected fraction of cracks is only $\frac{1}{n}$. Thus, when the hacker has absolutely no prior knowledge of the domain, anonymization is indeed a good option for the data owner, especially when the domain is large.

The above lemma can be generalized to a subset of items of interest, i.e., $\mathcal{I}_1 \subset \mathcal{I}$. For example, the data owner may only be concerned with the identities of the frequent items, or the items with the highest profit margin. Let us assume that $|\mathcal{I}_1| = n_1$ and the corresponding set of anonymized items is $\mathcal{J}_1 \subset \mathcal{J}$. Let the random variables $X_{i'}$ and X denote the same as before. The probability of an anonymized item $i' \in \mathcal{J}$ getting mapped correctly to its original item is still $\frac{1}{n}$. However, we are now interested only in $\mathcal{I}_1$. Therefore, X is given by $\sum_{i' \in \mathcal{J}_1} X_{i'}$, rather than the sum of all the $X_{i'}$. This argument gives rise to the following lemma.

LEMMA 2. *Let* $G = (\mathcal{J} \bigcup \mathcal{I}, E)$ *be a complete bipartite graph. Let* $\mathcal{I}_1 \subset \mathcal{I}$ *be a set of items of interest in the original domain and let* $|\mathcal{I}_1| = n_1$. *Let* X *be a random variable that represents the number of cracks of items in* $\mathcal{I}_1$ *in a mapping obtained from* G. *Then the expected number of interested items being cracked is* $E(X) = \frac{n_1}{n}$. $\square$

3.2 Compliant Point-Valued Belief Function

The assumption that the hacker is completely ignorant may be unrealistic. The data owner may in fact be completely "paranoid", and may conduct a disclosure risk analysis assuming that the hacker correctly guesses the frequency of each item. Note that this extreme case may be unrealistically conservative. Nonetheless, the analysis may lead to interesting observations.

This extreme is modeled by the compliant, point-valued belief function, i.e., the observed set of frequencies of items in the anonymized database and the belief frequencies in the prior knowledge, match exactly. The space of mappings in this case, gets partitioned into frequency groups, as shown in Figure 3(b). The three groups $\{1', 3', 4', 6'\}$, $\{2'\}$ and $\{5'\}$ correspond to the frequencies 0.5, 0.4 and 0.3 respectively. When the group size is 1, the hacker comes up with the cracks directly (e.g., $2'$ mapped to 2, and $5'$ mapped to 5). However, with the group consisting of the four elements $1', 3', 4', 6'$, any permutation of the original items $1, 3, 4, 6$ is a mapping. This is because the prior knowledge does not enable further discrimination between the items of the same group. Hence, the mappings within a group are similar to the case of ignorant belief functions. Mappings within each group are also independent of the mappings within other groups, thus permitting group-wise analysis for expected number of cracks. The result can be summed up across all the groups to obtain the expected number of cracks in the final mapping. We thus have the following result that gives the expected number of cracks for the compliant point-valued belief function.

LEMMA 3. *Let g be the number of distinct observed frequencies of the anonymized items. Let $G = (\mathcal{J} \bigcup \mathcal{I}, E)$ be a bipartite graph modeling the space of mappings for the compliant point-valued belief function. Let X be a random variable denoting the number of cracks in a mapping obtained from G. Then the expected number of cracks is $E(X) = g$.*

Proof Sketch: If there are g distinct observed frequencies, then the compliant point-valued belief frequencies match these g observed frequencies. Hence, the bipartite graph contain g components. Within each component is a complete bipartite subgraph. Thus, by Lemma 1, the expected number of cracks is 1 per component. Hence $E(X) = g$. $\square$

The above lemma says that even if the hacker has complete knowledge of the frequencies, the expected number of cracks g does not necessarily equal n. In fact, if there are a lot of items with equal frequencies, g can be much less than n. In a sense, the items in each group provide camouflage to each other so that their true identities are better protected.

Suppose that the data owner is concerned about the identities of only a subset $\mathcal{I}_1 \subset \mathcal{I}$. Then the results obtained thus far can be used to compute the expected number of cracks as follows. Lemma 2 can be used on each frequency group independently to give the expected number of cracks.

LEMMA 4. *Let g be the number of distinct observed frequencies of the anonymized items. Let $n_1, \ldots, n_g$ be the size of each of the frequency groups. Let $c_1, \ldots, c_g$ be the number of items the data owner is interested in, for each of the frequency groups. Let $G = (\mathcal{J} \cup \mathcal{I}, E)$ be the bipartite graph representing the space of all mappings and let X be a random variable representing the number of cracks of the items of interest in a mapping obtained from G. Then, the expected number of interested items being cracked is $E(X) = \sum_{i=1}^{g} \frac{c_i}{n_i}$.* $\square$

4. COMPLIANT INTERVAL BELIEF FUNCTIONS

The compliant point-valued belief function analyzed above represents the worst case from the data owner's perspective. In this section, we analyze more realistic situations when the hacker's belief function is a compliant interval function. Depending on prior knowledge, a hacker may associate arbitrary intervals with various items in the domain. For example, based on prior experience, the hacker may believe that item x has a frequency in the range $[0.4, 0.6]$, item y has a frequency in $[0.7, 0.8]$ and having no knowledge about the frequency of z, the hacker may associate the belief interval $[0, 1]$ with z. The belief function h in figure 2 is an example of an interval belief function.

4.1 A Direct Method

A direct method to compute the expected number of cracks is as follows. The size of the space of mappings is precisely the number of perfect matchings in the bipartite graph G, which is given by the *permanent* of the adjacency matrix A_G of G. The number of ways in which k out of the n items in $\mathcal{I}$ are cracked is computed as follows. We can choose the k cracks from n in $\binom{n}{k}$ ways. Once the k items are picked, remove these items and their primed counterparts from G. Also remove the edges corresponding to cracks from the other $n - k$ items and compute the number of perfect matchings in the remaining graph.

The exact expression for the expected value in terms of permanents is computed as follows. Let $\mathcal{I}^k$ denote the set of all subsets of size k of $\mathcal{I}$ and let $\Pi(A_g)$ denote the permanent of the matrix A_G. The probability of obtaining k cracks is given by:

$$P(X = k) = \frac{\sum_{S \in \mathcal{I}^k} \Pi(A_{G(S)})}{\Pi(A_G)}$$

where, for a given $S \subset \mathcal{I}^k$, $G(S)$ is the graph obtained by removing the edges (y', y) for all $y \in \mathcal{I}$ and the nodes x and x' and their incident edges for each $x \in S$ from G. The expected number of cracks is thus:

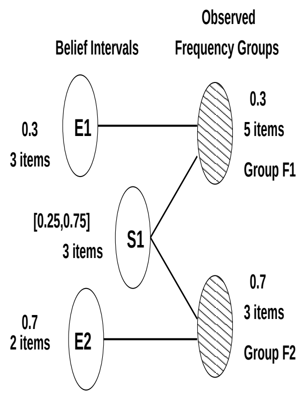
$$E(X) = \sum_{k=0}^{n} k * P(X = k) = \sum_{k=0}^{n} k * \sum_{S \in \mathcal{I}^k} \frac{\Pi(A_{G(S)})}{\Pi(A_G)}.$$

While the above formula is exact, the problem is that computing the permanent is known to be very difficult – more precisely, a #P-complete problem [25]. Various approximations have been developed for computing permanents [14, 21]. The state of the art is the polynomial-time randomized approximation scheme presented in [13]. However, the running time is of the order of $O(n^{22})$! Thus, for arbitrary interval belief functions and realistic domain sizes, it is not feasible to use the direct approach to compute the expected number of cracks. However, we explore special cases for which exact or approximate formulae are possible.

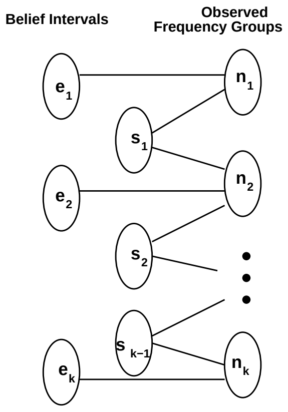
4.2 Chain Belief Functions

Consider the example shown in figure 4(a). The example gives a scenario where the domain has 8 items. There are two distinct frequency groups in the database, F_1 and F_2, corresponding to the frequencies 0.3 and 0.7. F_1 contains 5 items while F_2 contains 3 items. The hacker's belief function is that 3 items have the frequency 0.3, 2 have frequency 0.7 and the frequency of the remaining 3 items lie in the interval $[0.25, 0.75]$, and is assumed to be fully compliant. The space of crack functions is precisely the group structure which is shown in figure 4(a). The question here is: what is the expected number of cracks in such a case?

Let us consider an item in group E_1. The probability of this item getting mapped to its corresponding anonymized item in F_1 is $\frac{1}{5}$ as this item can only map to items in F_1. For an item in group E_2, this probability is $\frac{1}{3}$. The most complex case corresponds to items in the overlapped group S_1. An item in this group can be mapped to items in either F_1 or F_2. Suppose the corresponding anonymized item x' belongs to F_1. Then, the probability that x gets mapped to x' is the product of the probability that x gets mapped to group F_1 and the probability that within this group, x gets mapped to x'. This is given by $\frac{2}{3} \times \frac{1}{5}$. A similar argument holds for



(a) Chain of size 2



(b) Chain of size k

Figure 4: Chains - Size 2 and k

items in S_1 whose corresponding anonymized item is in F_2 and the probability for such items is given by $\frac{1}{3} \times \frac{1}{3}$. Let S_1^1 be the set of items in S_1 whose corresponding anonymized item belongs to F_1 and let S_1^2 be the remaining set of items that map to group F_2. The expected number of cracks is thus given by:

$$E(X) = \sum_{x \in E_1} \frac{1}{5} + \sum_{x \in E_2} \frac{1}{3} + \sum_{x \in S_1^1} \frac{2}{3} \times \frac{1}{5} + \sum_{x \in S_1^2} \frac{1}{3} \times \frac{1}{3} = \frac{74}{45}.$$

Hence, for this example, we can expect $\frac{74}{45}$ or 1.644 cracks on average.

Let us formalize the situation depicted in the above example. Consider the set of frequency groups based on the observed frequencies of the anonymized items arranged in the increasing order of frequency. Let there be k groups with frequencies $f_1 < f_2 < \cdots < f_{k-1} < f_k$. Let n_i, $1 \le i \le k$ be the number of items in the i^{th} group. The original items can be partitioned into belief groups based on their belief intervals. Two items x and y in $\mathcal{I}$ belong to the same belief group if they can map to the same set of frequency groups (based on their belief intervals). The interval belief function is said to form a *chain* whenever every belief group maps to either exactly one frequency group or two successive frequency groups. A belief group is called *exclusive* if it can map to exactly one frequency group and is called *shared* otherwise. Since there are k frequency groups of anonymized items, the chain is said to be of length k. A chain of length k is shown in figure 4(b).

LEMMA 5. *Let there be two distinct observed frequencies in the anonymized database and let n_1 and n_2 be the size of the frequency groups. Let the interval belief function be a chain of length 2 where the shared belief group is of size s_1 and the exclusive groups are of sizes e_1 and e_2 respectively, such that $e_1 + e_2 + s_1 = n_1 + n_2$. Let X denote a random variable that gives the number of cracks in a mapping that is consistent with the interval belief function. Then, the expected number of cracks is:*

$$E(X) = \frac{e_1}{n_1} + \frac{e_2}{n_2} + (n_1 - e_1) \times \frac{(n_1 - e_1)}{s_1} \times \frac{1}{n_1}$$
$$+(n_2 - e_2) \times \frac{(n_2 - e_2)}{s_1} \times \frac{1}{n_2}.$$

Proof Sketch: The probability that an item in the exclusive group is mapped correctly to its anonymized item is given by $\frac{1}{n_1}$ or $\frac{1}{n_2}$ depending on the two exclusive groups. An item in the shared group correctly maps either to frequency group 1 or frequency group 2. The probability of an item correctly mapping to its group is given by $\frac{(n_1 - e_1)}{s_1}$ or $\frac{(n_2 - e_2)}{s_1}$ depending on the frequency group. Once it maps to its group correctly, the probability of mapping it correctly to an anonymized item is $\frac{1}{n_1}$ or $\frac{1}{n_2}$. The number of items in the shared group that map to the frequency group 1 (resply. 2) is $n_1 - e_1$ (resply. $n_2 - e_2$). This explains the last two terms of the summation. The first two terms simply give the expected number of cracks for the two exclusive groups. □

We now generalize the above argument to derive the expected number of cracks for a chain of length k. For a chain of length k, there are k frequency groups in the anonymized database. There are k exclusive belief groups and $k-1$ shared belief groups. Let them be as shown in figure 4(b). We further assume that frequency group i is the frequency group containing n_i items, for $1 \le i \le k$. For an item in an exclusive group containing e_i items, the probability of the item mapping to its correct anonymized item is given by $\frac{1}{n_i}$. Now, consider the shared group of s_i items. An item in this group can correctly map to an item in frequency group i or frequency group $i+1$. The number of items that map correctly to frequency group i (to $i+1$) is given by $\sum_{j=1}^{i}(n_j - e_j - s_{j-1})$ (resply. $\sum_{j=1}^{i}(s_j + e_j - n_j)$). Using the same argument as for chains of length 2, the probability that an item in a shared group maps correctly to its anonymized item is given by the product of the probability of mapping it correctly to that group and the probability that it maps correctly within that group. We thus have the following formula for a chain of length k.

LEMMA 6. *Let there be k distinct observed frequencies in the anonymized database and let $n_1, \ldots, n_k$ be the size of the frequency groups. Let the interval belief function be a chain of length k where the shared belief groups are of sizes $s_1, \ldots, s_{k-1}$ respectively and the exclusive groups are of sizes $e_1, \ldots, e_k$. Let X denote a random variable that gives the number of cracks in a mapping that is consistent with the interval belief function. The expected number of cracks is:*

$$E(X) = \sum_{j=1}^{k} \frac{e_j}{n_j} + \sum_{i=1}^{k-1} \frac{[\sum_{j=1}^{i}(n_j - e_j - s_{j-1})]^2}{s_i n_i}$$
$$+\sum_{i=1}^{k-1} \frac{[\sum_{j=1}^{i}(s_j + e_j - n_j)]^2}{s_i n_{i+1}} \quad \square$$

5. O-ESTIMATES FOR GENERAL INTERVAL BELIEF FUNCTIONS

So far we have derived an exact formula for computing the expected number of cracks in a chain of arbitrary length. Unfortunately, a general belief function modeling prior knowledge need not always form a chain. Thus, in this section, we deviate from exactness and turn to approximation. We propose a heuristic algorithm that applies to any general interval function.

5.1 The O-estimate Heuristic

Recall from the proof sketches of the previous lemmas that we repeatedly apply a well-known result from statistics. Let X and Y be two random variables. Then it is the case that $E(aX + bY) = aE(X) + bE(Y)$. This identity does not require that X and Y be independent. Thus, regardless of whether the bipartite graph forms a chain or not, we can still analyze

```
Algorithm O-estimate
Input:  $\beta$ - interval belief function over $\mathcal{I}$
        $\mathcal{D}$ - Anonymized Database over $\mathcal{J}$
1.  Compute frequency of single items in $\mathcal{J}$ from $\mathcal{D}$.
2.  Compute the frequency groups $g_1, \ldots, g_k$.
    Let $n_1, \ldots, n_k$ be their sizes.
    Let $f_1 < f_2 \cdots < f_k$ be their frequencies.
3.  Initialize $O_{est} = 0$
4.  For each $x \in \mathcal{I}$
    a.  Compute $n_x$, the number of anonymized
        items that can map to x.
    b.  $O_{est} = O_{est} + \frac{1}{n_x}$
5.  return $O_{est}$;
```

Figure 5: The O-estimate Heuristic

the expected number of cracks by examining each item in an isolated fashion.

Let β be an interval belief function and let G be the bipartite graph representing the space of all mappings. For each $x \in \mathcal{I}$, let O_x denote the outdegree of the node x in G. The outdegree of x basically denotes the number of anonymized items that can be mapped to it. Note that the probability that the anonymized item x' correctly maps to x is given by $\frac{1}{O_x}$ (under the compliancy assumption, this edge is always guaranteed to exist in the space of mappings). Thus, the expected number of cracks can be estimated by summing this probability across all the items in $\mathcal{I}$. We call this heuristic the **O-estimate** (denoted $OE(\beta, \mathcal{D})$) and it is defined as $\sum_{x \in \mathcal{I}} \frac{1}{O_x}$ where O_x is the outdegree of the node x in G.

Figure 5 outlines a procedure to compute the O-estimate, given a belief function and a database. Step 1 takes $|\mathcal{D}|$ time to compute the frequency of anonymized items by making a single database pass. Let n be the number of items in $\mathcal{I}$ (resply. $\mathcal{J}$). There can be $k = n$ frequency groups as each item can have a distinct frequency. Step 2 thus takes $O(n \log n)$ time. Step 4 then counts for each item x, the number of anonymized items that can be mapped to it. This step takes $O(n^2)$ time when implemented naively. But by using frequency groups and prefix sums of the counts of these groups, this reduces to $O(n \log n)$. Hence the total running time of an efficient implementation is $O(|\mathcal{D}| + n \log n)$.

5.2 Properties of the O-estimates

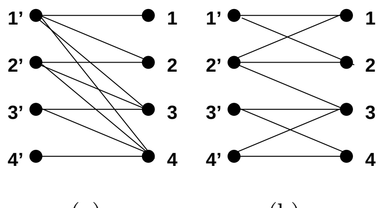


Figure 6: Inexactness of the O-estimates

It is worthwhile to try to understand why the O-estimate is not exact. Consider the example in Figure 6(a). The outdegrees of $1, 2, 3$ and 4 are $1, 2, 3$ and 4 respectively. Thus, the O-estimate of the expected number of cracks is $1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = \frac{25}{12}$. However, consider node 1. Only $1'$ can map to it. A perfect matching can thus map only $1'$ to 1. This leads to the other edges on 1 to be removed from the graph. As a consequence, $2'$ is the only node that can map to 2 and this cascades to the scenario when $1'$, $2'$, $3'$ and $4'$ map to 1,2,3 and 4 respectively. Hence, the number of cracks is 4. The same argument can be given by starting with node $4'$ (which is also of degree 1).

```
Algorithm Propagate
Input:  G - A fully compliant bipartite graph over $\mathcal{I} \cup \mathcal{J}$
1.  while there is a node with degree 1 in G
    Let $x$ be this node (the same holds for $x'$).
        a. Remove the nodes $x$ and $x'$ from G
        and their incident edges.
        b. For each $y$ such that the edge $(x', y)$ is just
        removed, decrement $O_y$ by 1.
```

Figure 7: Reducing the Outdegrees by Propagation

This example shows that when the outdegree of a node is 1, propagation may take place that essentially reduces the outdegrees of other items. Figure 7 outlines the procedure to handle this propagation. The complexity of the algorithm is $O(ve)$, where v is the number of nodes in the graph and e is the number of edges in the graph. This is because, in the worst case (as in example 6(a)), the propagation may go on for v steps until each vertex gets mapped to a single item. But in practice, the fixed point is often reached in a few iterations. This propagation procedure should be applied after step 4(a) in Figure 5. Hereafter, whenever we refer to outdegrees, we assume that this algorithm has been applied.

The discussion so far focuses on a special case when an (anonymized) item has an outdegree 1. The question is whether a similar phenomenon occurs when the outdegree is 2 or higher. Consider the example in Figure 6(b). This differs from the example in Figure 6(a) in that no item can be surely cracked. However, it is clear that a perfect matching would map $\{1', 2'\}$ to $\{1, 2\}$, and $\{3', 4'\}$ to $\{3, 4\}$. Thus, the edge $(2', 3)$ is irrelevant. Yet, the O-estimate continues to count the edge towards the outdegree of item 3.

Furthermore, let us review the situation for a chain belief function. Consider again the chain of length 2 in Figure 4(a). By Lemma 5, the answer is $\frac{74}{45}$. However, by the O-estimate, the answer is $\frac{197}{120} = 1.6417$. The reason for this inexactness can be explained by considering the items in group S_1. An item in group S_1 can map either to F_1 or F_2 but not both. Let x and y be items in S_1 such that x' is in F_1 and y' is in F_2. The O-estimate assigns a uniform probability of $\frac{1}{8}$ to both x and y as they can map to items in either group. However, the likelihood of x and y being assigned to their correct groups are not uniform. The O-estimate fails to capture this non-uniformity.

We generalize the discussion of this example to a general chain of length k. Consider the chain in figure 4(b). For each $x \in E_i$, the outdegree O_x is given by $\frac{1}{n_i}$. For each $x \in S_i$, the outdegree O_x is given by $\frac{1}{n_i + n_{i+1}}$. Hence, the **O-estimate** of a belief function β which models a chain of length k is given by

$$OE(\beta) = \sum_{j=1}^{k} \frac{e_j}{n_j} + \sum_{j=1}^{k-1} \frac{s_j}{n_j + n_{j+1}}$$

The question is how the O-estimate compares with the exact formula for chains. The difference (Δ) is given by:

$$\Delta = \sum_{i=1}^{k-1} \frac{[\sum_{j=1}^{i} (n_j - e_j - s_{j-1})]^2}{s_i n_i} + \sum_{i=1}^{k-1} \frac{[\sum_{j=1}^{i} (s_j + e_j - n_j)]^2}{s_i n_{i+1}} - \sum_{j=1}^{k-1} \frac{s_j}{n_j + n_{j+1}}$$

While analyzing this more formally may be too involved, we illustrate the magnitude of Δ with some examples. Consider a chain of size 3 with the following values: $n_1 = 20$,

$n_2 = 30$ and $n_3 = 20$; these values are chosen arbitrarily. The table below shows the size of Δ relative to the exact value, for varying values of e_1,e_2,e_3,s_1 and s_2. We see that the percentage difference is small, showing that for chains, O-estimates are reasonably accurate. In Section 7, the accuracy of O-estimates is evaluated with real datasets.

e_1	e_2	e_3	s_1	s_2	Percentage error (%)
10	10	10	20	20	1.54
5	10	10	25	20	4.8
5	10	5	25	25	8.3
5	6	5	27	27	5.76
10	20	10	15	15	7.23

So far, we have discussed the shortcomings of O-estimates. Below we show one nice property of O-estimates, namely, *monotonicity*. As the belief interval for an item x gets wider, the number of anonymized items that can map to it increases. Thus, the expected number of cracks drops. We formalize this notion below.

Definition 7. *Let β_1 and β_2 be two interval belief functions on $\mathcal{I}$. Then, $\beta_1 \preceq \beta_2$ whenever $\forall x \in \mathcal{I}$, $\beta_1(x) \subseteq \beta_2(x)$. We say that an interval $[l_1, r_1] \subseteq [l_2, r_2]$ whenever $l_1 \geq l_2$ and $r_1 \leq r_2$.*

As the uncertainty in belief (width of intervals) grows, we would expect it to be more difficult on the average to crack individual items. This is captured by the following lemma, which states that the O-estimate is monotonically non-decreasing with respect to the relation $\preceq$. We will see in the next section how this property can be made use of.

Lemma 8 (Monotonicity of OE). *$\mathcal{D}$ is an anonymized database. Let β_1 and β_2 be two compliant interval belief functions such that $\beta_1 \preceq \beta_2$. Then, $\mathsf{OE}(\beta_1, \mathcal{D}) \geq \mathsf{OE}(\beta_2, \mathcal{D})$.*

Proof Sketch: Let x be any item in $\mathcal{I}$ and let $O_1(x)$ and $O_2(x)$ be the outdegrees of x in the space of compliant mappings for β_1 and β_2 respectively. Then $O_1(x) \leq O_2(x)$. This implies that $\sum_{x\in\mathcal{I}} \frac{1}{O_1(x)} \geq \sum_{x\in\mathcal{I}} \frac{1}{O_2(x)} \Rightarrow \mathsf{OE}(\beta_1, \mathcal{D}) \geq \mathsf{OE}(\beta_2, \mathcal{D})$. Hence the lemma follows. □

5.3 α-Compliant Belief Functions

So far, we have derived various formulas for computing the expected number of cracks with one assumption – full compliancy. That is, we assume that for each item, the hacker's belief interval contains the true frequency of the item. This is possible if the hacker has very good knowledge about the items, or if the intervals are wide or conservative. However, in general, there is no reason to believe that full compliancy is always possible, or even likely. Thus, below we examine α-compliant belief functions. However, they are tricky to deal with because the hacker would have no idea which ones of his/her guessed intervals are not correct. (Had (s)he known, (s)he would have changed them in the first place!) We ignore this issue until the next section. Below we proceed with the analysis of computing the expected number of cracks, given that *somehow we know which items are guessed wrong.*

Let β be a belief function on $\mathcal{I}$ such that for a subset of items $\mathcal{I}_C \subset \mathcal{I}$, the function satisfies the compliancy assumption and for $\mathcal{I} - \mathcal{I}_C$ the function is non-compliant. Thus, α is defined as the ratio of the size of $\mathcal{I}_C$ to that of $\mathcal{I}$. To approximate the expected number of cracks, the O-estimate heuristic can easily be applied. However, for the items $x \in (\mathcal{I} - \mathcal{I}_C)$, the consistency assumption guarantees that these items will not be cracked. Thus, it is sufficient to simply sum over those items $x \in \mathcal{I}_C$. Hence, the O-estimate for β is defined as $\mathsf{OE}(\beta, \mathcal{D}) = \sum_{x\in\mathcal{I}_C} \frac{1}{O_x}$.

Intuitively, when belief functions become more and more non-compliant, the expected number of cracks should decrease, as it is impossible to crack the non-compliant items by a consistent mapping. This is captured by the following lemma which states that the O-estimate decreases as the degree of non-compliancy increases.

Definition 9. *Let β_1 and β_2 be two interval belief functions on $\mathcal{I}$. Let β_1 be compliant on the set of items $\mathcal{I}_C^1 \subset \mathcal{I}$, and β_2 be compliant on $\mathcal{I}_C^2 \subset \mathcal{I}$. We say that $\beta_2 \preceq_C \beta_1$ whenever: (i) $\mathcal{I}_C^2 \subseteq \mathcal{I}_C^1$; and (ii) $\forall x \in \mathcal{I}_C^2$, $\beta_1(x) \subseteq \beta_2(x)$.*

The above definition imposes a partial order on α-compliant belief functions, based on the subset of compliant items. As this subset becomes smaller and smaller, the expected number of cracks becomes smaller as the guessed intervals of the compliant items do not shrink. The proof of this lemma is similar to that of the earlier monotonicity lemma.

Lemma 10 (Monotonicity of OE for α-compliancy). *Let $\mathcal{D}$ be an anonymized database. Let β_1 and β_2 be two interval belief functions such that $\beta_2 \preceq_C \beta_1$. Then, $\mathsf{OE}(\beta_2, \mathcal{D}) \leq \mathsf{OE}(\beta_1, \mathcal{D})$.* □

6. A RECIPE FOR RISK ASSESSMENT

Thus far, we have studied the various cases involving full or partial compliancy involving point-valued or interval belief functions. These are various ways to capture information that a hacker may possess. Now we are ready to tackle the original dilemma facing the data owner, of whether to release the anonymized data: *Just how safe is the anonymized data in the presence of partial information?*

Let us begin with the absolute worst case: the compliant, point-valued belief function. We feel that for most applications, this worst case is too conservative, as it is unrealistic to expect that the hacker to know each frequency precisely. The expected percentage of cracks is typically unrealistically high. Thus, the owner is ill advised to make decisions based on this value, unless (s)he is paranoid.

6.1 Determining the Width of the Intervals

It makes sense to relax the worst case in two ways. First, the compliant point-valued belief function can be extended to a compliant interval belief function. That is, if f_x denotes the true frequency of item x, then the interval guessed by the compliant interval belief function is set to $[f_x - \delta, f_x + \delta]$. From the data owner's perspective, this belief function corresponds to the situation when the hacker is accurate in guessing a right "ball-park" frequency range for every item.

The question here is what is an appropriate value of δ to use. As a heuristic, we propose using the median frequency gap for every item. That is, the frequency gaps between two successive frequency groups in the data are computed, and the median gap between the groups is used.

While more details of the benchmark datasets we use will be given in the next section, the table in Figure 9 shows various statistics of these datasets. The columns of the table give the number of items in the domain, the number of transactions, the number of distinct frequency groups, the number of singleton frequency groups, and the mean, median, minimum and maximum frequency gap between successive groups. Two observations can be made.

- These datasets are chosen to represent various characteristics. For instance, the 130 items of the CONNECT dataset form 125 distinct frequency groups, 122 of which consists of a single item. In contrast, the 7120 items of the PUMSB items cluster into 651 frequency groups. Nevertheless, there are still a large number of singleton frequency groups, confirming that for real datasets, the compliant point-valued belief function gives too high an estimate on the percentage of cracks.
- For all the datasets, the median frequency gap is much closer to the minimum than to the maximum. In contrast, the average frequency gap is much larger than the median. Thus, by choosing the median value as the width δ of the intervals (hereafter denoted as δ_{med}), the data owner errs on the conservative side. This is because according to the monotonicity result in Lemma 8, the wider the interval, the smaller is the expected number of cracks. Thus, as compared with using the median, using the average may under-estimate the percentage of cracks. In section 7.4, we give additional details about using other alternatives, such as the sampled median and the sampled average, as the width of the intervals.

Figure 8 shows a recipe we suggest to a data owner to assess the risk of releasing anonymized data. Steps (1) to (7) follow what we have discussed so far. Notice that the recipe requires an input percentage τ, called the *degree of tolerance*, which gives a fraction of the items $\mathcal{I}$ that the data owner can tolerate being cracked. If the expected number of cracks based on the compliant point-value belief function is already within the tolerance, then it is an easy decision to release the anonymized data. In the most likely case, however, this estimate is too high. The recipe then suggests computing the O-estimate based on the compliant interval belief function with the width δ_{med} (i.e., steps (3) to (7)). Note that we use a uniform width of $2 \times \delta_{med}$ for all the intervals. This does not restrict a hacker from using a non-uniform width as (s)he might have varying levels of knowledge about different items. But even if this is true, there is little reason to believe that the data owner has access to the hacker's belief function. Thus, in the recipe, it is a reasonable simplification to use a uniform width.

It is possible that the O-estimate based on the compliant interval belief function is still higher than the owner's tolerance. After all, it is unlikely that the belief function satisfies the compliancy assumption for all the items in $\mathcal{I}$, particularly if the domain is large. The hacker's guesses may be correct for some items but incorrect for the others. Thus, we resort to α-compliant belief functions.

6.2 Determining the Degrees of Compliancy

While in the previous section we give a formula for computing the O-estimate for an α-compliant β, applying the formula is tricky in practice. The difficulties are that (i) it is not clear which values of α could be used, and (ii) even if an α value is established, which specific subset of compliant items $\mathcal{I}_C$ could apply.

Instead of picking specific α values to use, our approach is to examine the risk over a range of α values. Specifically, we use the data owner's specified tolerance τ to determine the largest α value for which the O-estimate of expected cracks fall within the tolerance. Given the monotonicity behavior stated in Lemma 10, we can use a binary search to find

```
Algorithm Assess-Risk
Inputs: $\tau$ - degree of tolerance; $\mathcal{D}$ - Anonymized Database
1. Compute $g$ based on Lemma 3.
2. If $g \leq (\tau \times |\mathcal{I}|)$, disclose $\mathcal{D}$ and stop.
3. Compute the frequency groups from $\mathcal{D}$, and the median gap
   $M$ between frequency groups
4. Set width $\delta_{med}$ to be $M$.
5. Set up $\beta(x) = [f_x - \delta_{med}, f_x + \delta_{med}]$ for $x \in \mathcal{I}$, where
   $f_x$ denotes the frequency of $x \in \mathcal{I}$.
6. Compute the O-estimate $OE(\beta, \mathcal{D})$ according to Figure 5.
7. If $OE(\beta, \mathcal{D}) \leq (\tau \times |\mathcal{I}|)$, disclose $\mathcal{D}$ and stop.
8. Set $\alpha$ to be 1.
9. Perform a binary search on $\alpha$ to determine the largest
   $\alpha$ so that the corresponding $\alpha$-compliant belief function
   $\beta$ satisfies the condition $OE(\beta, \mathcal{D}) \leq (\tau \times |\mathcal{I}|)$.
10. Return the value of $\alpha$.
```

Figure 8: A Suggested Recipe for Risk Assessment

this value α_{max}. Essentially, it says that in order for the hacker not to crack more than the fraction τ of items that the owner can tolerate, the hacker must not correctly guess the frequency intervals of more than $\alpha_{max} \times |\mathcal{I}|$ items. It is up to the owner to decide whether α_{max} is high enough for comfort. For example, if $\alpha_{max} = 0.8$, then the data owner may decide to disclose the data because the owner considers it highly unlikely that the hacker can guess correctly the frequency intervals, within δ_{med}, of 80% of the items, particularly when the domain is large. On the other hand, if $\alpha_{max} = 0.2$, the data owner may decide to withhold the data because 20% correct guesses may be too small for comfort. Later in Figure 13, a heuristic is proposed to evaluate whether α_{max} is high enough.

Steps (8) and (9) in Figure 8 outline the binary search. There is, however, one important detail. The binary search is based on the monotonicity stated in Lemma 10, which requires a partial ordering on the subset of compliant items $\mathcal{I}_C$. This brings us back to the general issue of how to determine membership in $\mathcal{I}_C$. Our approach is to take a random selection of items and average over a few runs. For example, let us say that when $\alpha = 1$, step (7) gives too high an expected number of cracks. Thus, the algorithm enters into the first iteration of the binary search, with $\alpha = 0.5$. Say the algorithm averages over 5 runs. Thus, there are five subsets of compliant items $\mathcal{I}_C^1, \ldots, \mathcal{I}_C^5$, each having 50% of items randomly picked to be non-compliant. These subsets give rise to five 0.5-compliant belief functions $\beta_1, \ldots, \beta_5$. Thus, the average value of $OE(\beta_1, \mathcal{D}), \ldots, OE(\beta_5, \mathcal{D})$ is used in the condition check in Step (9).

Let us suppose that the above average value is still beyond the owner's tolerance, and the binary search continues with $\alpha = 0.25$. In that case, half of the compliant items in each of $\mathcal{I}_C^1, \ldots, \mathcal{I}_C^5$ are randomly picked to be non-compliant. The search then continues as discussed before. Anchoring step (9) in this manner over multiple $\mathcal{I}_C$'s satisfies the requirement of Lemma 10.

7. EXPERIMENTAL EVALUATION

7.1 Experimental Setup

In this section, we evaluate empirically the accuracy of the O-estimates and the effectiveness of the recipe. We used real datasets that we obtained from the UCI repository (http : //kdd.ics.uci.edu) and the FIMI repository (http : //fimi.cs.helsinki.fi/fimi03/). Figure 9 shows the characteristics of the datasets. The domain varies from 75 to 16470 items, and the number of transactions varies from

Dataset	# items	# Trans.	# Gps.	Size 1 Gps.
CONNECT	130	67557	125	122
PUMSB	2113	49046	650	421
ACCIDENTS	469	340184	310	286
RETAIL	16470	88163	582	218
MUSHROOM	120	8124	90	77
CHESS	75	3196	73	71

Dataset	Mean	Median	Min.	Max.
CONNECT	0.0081	0.0029	0.000015	0.0519
PUMSB	0.00154	0.000041	0.00002	0.0536
ACCIDENTS	0.00324	0.000176	0.000029	0.04966
RETAIL	0.00099	0.0000113	0.0000113	0.30102
MUSHROOM	0.01124	0.00394	0.00049	0.1477
CHESS	0.01389	0.00657	0.000313	0.0494

Figure 9: Frequency Statistics for Various Benchmarks

3196 to 340184. The number of frequency groups and the number of singleton frequency groups are given. In many cases, the latter number is high in relation to the total number of items, thus confirming that the compliant, point-valued belief function usually gives too high an estimate on the percentage of cracks in practice. However, the RETAIL dataset is very different. The ratio of the number of transactions to the domain size is 1-2 orders of magnitude smaller than those for the other datasets. We label this dataset as "sparse". The table also shows the average, median, minimum and maximum gap between frequency groups. Even though we obtained the experimental results for all these datasets, we present results only for a subset of the datasets due to space limitations.

All procedures were implemented in C++. The code to simulate the expected number of cracks for a given interval belief function needs to generate many perfect matchings from the bipartite graph that represents the space of all consistent mappings. This generation of samples is complicated by the fact that we need matchings that are perfect, consistent, and as much as possible, random. To do so, the generation procedure initially starts with a perfect matching where every edge is of the form (i', i) i.e, every item is cracked. Then, for a fixed number of iterations $(100, 000)$, a random permutation P of $\mathcal{I}$ is generated. For each $i \in \mathcal{I}$, the edge (i, x) in the matching is swapped with the edge $(P(i), y)$ in the matching if the resulting edges (i, y) and $(P(i), x)$ are still consistent. This gives a *seed matching*. Then the procedure applies another 10,000 iterations as before to generate the first sample. The iterations continue and for every 10,000 iterations, a new sample is generated. In this way, the procedure generates 250 samples. At this point, a new seed matching is re-generated *from scratch*, and another 250 samples are generated. For the results reported below, we used 5,000 samples. The experiments were run on a dual Athlon MP1800+ processor (1533 MHz), with 256KB cache, 1GB of main memory (266MHz DDR RAM).

7.2 Accuracy of the O-Estimates

Figure 10 compares the O-estimates with the estimates from simulation on the four benchmarks with full compliancy (i.e., Step (6) of Figure 8). In all cases, the O-estimates are very accurate as compared with the average simulated estimates. Recall that the average simulated estimates is obtained from 5 runs. We also recorded the standard deviation on the simulated estimates. In all cases, the differences between the O-estimates and the average simulated estimates are well within one standard deviation. The accuracy shown by the O-estimates also confirms that the recipe's choice of

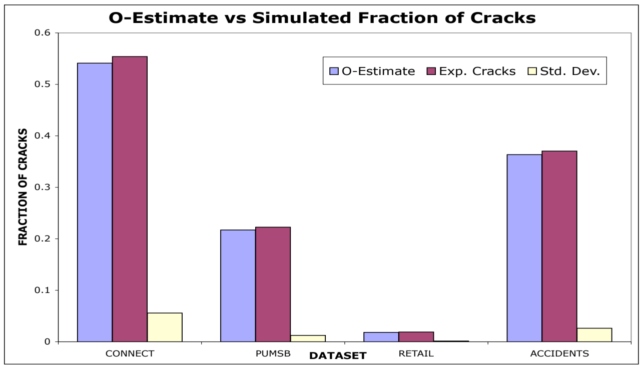


Figure 10: O-estimates vs Average Simulated Estimates

using δ_{med} as the width of the intervals works out well. Incidentally, even for the RETAIL dataset, it takes only a few seconds to compute the O-estimate.

7.3 The Effectiveness of the Recipe

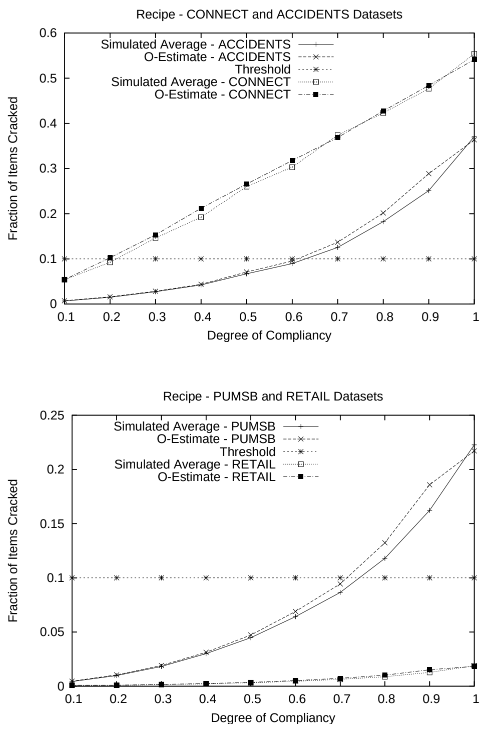


Figure 11: Varying the Degree of Compliancy

Figure 11 illustrates how the proposed recipe in Figure 8 works for real datasets. The x-axis shows the values of α and the y-axis shows the O-estimates expressed in fractions of the domain size. Let say that the data owner has a tolerance level of $\tau = 0.1$, as shown by the horizontal line. First, for the RETAIL dataset, it is a clear decision to release the anonymized data. In fact, even if the hacker correctly guesses all the frequency intervals, the expected fraction of cracks is still below 0.02.

The other datasets are different from RETAIL. For the PUMSB dataset, a tolerance level of 0.1 corresponds to a compliancy $\alpha_{max} \approx 0.7$. In other words, the hacker needs to have correctly guessed the intervals for 70% of items. This percentage is probably high enough to make the data owner feel secured in releasing the data. The situation is similar for the ACCIDENTS dataset.

The situation for CONNECT is very different. The tolerance level of 0.1 corresponds to $\alpha_{max} \approx 0.2$. Thus, the comfort level of the data owner is only about 20% of the items, which corresponds to 26 items to be exact. The data owner may want to think twice before releasing the data.

The shapes of the curves in Figure 11 are interesting. For RETAIL and CONNECT, the curves are approximately linear. Notice that CONNECT has a small number of items but a relatively large number of transactions, whereas the situation is rather contrary for RETAIL. Yet, their curves look similar. In contrast, the curves for PUMSB and ACCIDENTS are super-linear. Thus, how the O-estimates vary with the degree of compliancy does not appear to be determined directly by domain size or transaction size.

Finally, Figure 11 shows again the comparison between the O-estimates and the average simulated estimates. They remain very close for all degrees of compliancy, confirming the accuracy of the O-estimates.

7.4 Degrees of Compliancy from Similar Data

As shown above, the data owner needs to wrestle with the decision whether α_{max} is high enough, particularly if the hacker may have gained partial information based on similar data. The question is how to define "similar" data. Below, we experiment with the idea that the data owner *simulates similarity by sampling.* That is, the larger the sample $\mathcal{D}' \subset \mathcal{D}$, we expect $\mathcal{D}'$ to be more similar to $\mathcal{D}$.

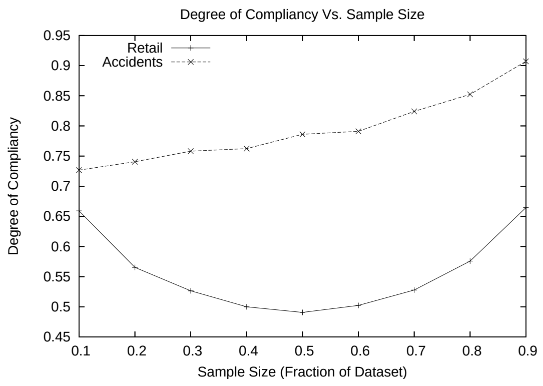


Figure 12: Degrees of Compliancy from Similar Data

For the ACCIDENTS and RETAIL datasets, Figure 12 shows the variation in the degree of compliancy of a belief function created from samples of varying percentages. Rather surprisingly, the degrees of compliancy can be high even for small samples. For instance, for a sample of 10%, the α value is above 0.7 for ACCIDENTS. This information can be very significant for the data owner. For example, for the ACCIDENTS dataset, recall from our earlier discussion based on Figure 11 that for a tolerance of $\tau = 0.1$, the corresponding $\alpha_{max} = 0.65$. Seeing that even a 10% sample, corresponding to a somewhat similar data set, can easily give an α value higher than 0.7, the owner may decide not to disclose the data after all.

In [7], Clifton argues that releasing a small random sample poses no threat to the data owner as little information can be revealed. In the context of compliancy, this does not appear to be true for every dataset.

```
Procedure Similarity-by-Sampling
1. For a given range of sample sizes $p$
    a. Get a sample $\mathcal{D}_p$ of the database $\mathcal{D}$.
    b. Determine the frequency $\hat{f}_x$ for every $x \in \mathcal{I}$ in $\mathcal{D}_p$.
    c. Determine the sampled median frequency gap $\delta'_{med}$
    of the frequency groups in $D_p$.
    d. Determine the degree of compliancy $\alpha$ by checking
    for each $x \in \mathcal{I}$, whether $f_x \in [\hat{f}_x - \delta'_{med}, \hat{f}_x + \delta'_{med}]$.
    e. Repeat (a) to (d) for 10 samples and get the average $\alpha_p$.
```

Figure 13: Data Similarity by Sampling

As shown in Figure 12, for the RETAIL dataset, there is a gradual drop in compliancy as the sample size increases until the size reaches 50% of the original size. This is counter-intuitive on first sight. But there is an interesting subtlety here. For a normal dataset, like ACCIDENTS, as the sample size increases, the sampled median gap between frequency groups increases. Thus, the width of the intervals increases, making compliancy easier to satisfy. Hence, there is a gradual increase in compliancy as the sample size increases. In contrast, the RETAIL dataset is abnormally sparse. In particular, for a 10% sample, there are only about 8800 transactions over a domain of about 16,000 items. Thus, the items tend to cluster together as their frequencies are under-determined. As the sample size increases, some of the items in one frequency group, start to separate into more frequency groups. Thus, the sampled median gap between frequency groups drops, narrowing the intervals of the belief function. Hence, the compliancy drops accordingly. This phenomenon persists until the number of frequency groups stabilizes; from that point on, the normal trend kicks in. Incidentally, if instead of the sampled median, the sampled average gap is used as the width of the intervals in the belief function, the degree of compliancy is at 0.99 (not shown in the figure), uniformly across all sample sizes. This again confirms that using the average can be misleading.

We provide a simple procedure shown in Figure 13 that implements the idea of simulating similarity by sampling. It uses multiple samples of varying sizes to generate the kind of curves shown in Figure 12 for a given dataset. This curve can then be used in conjunction with the recipe in Figure 8. Specifically, the recipe returns the α_{max} value. The data owner can then use the curve to ascertain whether α_{max} is high enough based on the corresponding sample size value.

8. DISCUSSION

8.1 Beyond Frequent Sets

In this paper, we address the question of how safe anonymized data is in the presence of partial information. So far, we use a belief function to represent the partial information on the frequencies of items. This is the first level of our analysis. In the second level of our analysis, we use the belief function and the anonymized data to set up an appropriate bipartite graph which represents the space of all consistent crack mappings. It is important to note that while the first level of partial information representation is specific to frequent sets, the second level of analyzing the bipartite graph is completely general. That is to say, much of the results presented here can carry over to other situations, as long as the bipartite graph is set up by some means. Specifically, Lemmas 1 to 4 are already expressed in terms of the underlying bipartite graph. At present, Lemmas 5, 6, 8 and 10 are expressed in terms of the belief functions, mainly for ease of understanding. However, they can re-stated from the perspective of the underlying bipartite graph. As an example, Lemma 8 can be restated as follows:

Let G_1 and G_2 be bipartite graphs with the vertex set $\mathcal{I} \cup \mathcal{J}$ and edge sets E_1 and E_2 respectively, modeling the space of crack functions β_1 and β_2. Then, if G_1 is a subgraph of G_2, then the O-estimate for G_2 is smaller than that of G_1.

Let us consider an example. Suppose the task at hand is classification, and the data owner needs to decide whether to release an anonymized relation with attributes: age, ethnicity and car-model. Let say that the real domain consists of people identified by their names (e.g. {Bob,Mary,... }), and the anonymized domain identified by an integer {1', 2', ... }. Suppose that the hacker has partial information about certain individuals. Recall from Figure 12 that even a small sample of 10% can reveal a lot of true information. In any case, regardless of how this piece of partial information comes about, if the hacker somehow knows that John is Chinese owning a Toyota, then edges can be set up between (x', John) for all anonymized items x' with ethnicity being Chinese and car-model being Toyota. Similarly, if the hacker somehow knows that Mary's age is between 30 and 35, the appropriate edges can be set up in the bipartite graph to connect Mary to all anonymized items x' in the same age group. And if the hacker has no knowledge of Bob, Bob is connected to every anonymized item in the graph. Once the graph is set up, we can re-apply all the lemmas above to estimate the expected number of cracks, which may help the data owner to decide whether it is safe to release the anonymized relation.

8.2 Summary and Ongoing Work

A classic dilemma that an organization faces is if the data is not released, they cannot take advantage of the opportunities offered by data mining. On the other hand, if they do, they run the risk of disclosing sensitive data. To mitigate the latter, they may sanitize their data or more generally apply some data transformations. In this paper, we address the question of how safe the anonymized data is with respect to protecting the true identities of the data items. The novelty of our work is to incorporate the possible existence of partial information possessed by the hacker. We propose various classes of belief functions, representing the capturing of various degrees of partial information. We derive exact or approximate formulas for computing the expected number of cracks. In particular, we propose the O-estimate heuristic which is applicable to any bipartite graph/belief function, and easy to compute. We evaluate the accuracy of O-estimates first by comparing with chain interval belief functions, and later by experimenting with real datasets. Last but not least, we provide a recipe to help the data owner to decide whether releasing the anonymized data is safe enough. And as pointed out above, our analytic framework based on bipartite graph extends beyond frequent sets to other data mining and analysis tasks where evaluating the risk of disclosure is important.

In ongoing work, we extend belief functions defined over the domain of items to those defined over the powerset. That is, while this paper focuses on the identities of individual items, the next step is to concentrate on the identities of sets of items. Consider the situation in 6(b) again. While there is no information to distinguish between 1' and 2', the itemset {1', 2'} indisputably maps correctly to the itemset {1, 2}. This is particularly interesting for frequent set mining.

9. REFERENCES

[1] Adam N.R. and Wortmann J.C. Security-Control Methods for Statistical Databases: A Comparative Study. *ACM Computing Surveys*, 21(4), 1989.

[2] Aggarwal C.C and Yu P.S. A Condensation Approach to Privacy Preserving Data Mining. *EDBT*, 2004.

[3] Aggarwal.G et. al. Anonymizing Tables. *ICDT Conference*, 2005.

[4] Agrawal R. and Srikant R. Privacy Preserving Data Mining. *ACM SIGMOD Conference*, 2000.

[5] Agrawal D. and Aggarwal C.C. On the Design and Quantification of Privacy Preserving Data Mining Algorithms. *ACM PODS Conference*, 2001.

[6] Agrawal R., Imielinksi T., Swami A. Mining Association Rules between Sets of Items in Large Databases. *ACM SIGMOD Conference*, 1993.

[7] Clifton C. Using Sample Size to Limit Exposure to Data Mining. *Journal of Computer Security*, 8(4), 2000.

[8] Dinur I., Nissim K. Revealing Information while Preserving Privacy. *ACM PODS Conference*, 2003.

[9] Domingo-Ferrer J. et. al. Information-Theoretic Disclosure Risk Measures in Statistical Disclosure Control of Tabular Data. *IEEE SSDBM Conference*, 2002.

[10] Evfimievski A. et. al. Privacy Preserving Mining of Association Rules. *Information Systems*, 29(4) 2004.

[11] Fienberg S.E. et. al. Disclosure Limitation using perturbation and related methods for Categorical Data. *Journal of Office Statistics*, 14, 1998.

[12] Iyengar V. S. Transforming Data to Satisfy Privacy Constraints. *ACM KDD Conference*, 2002.

[13] Jerrum M. et. al. A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries. *ACM STOC Conference*, 2001.

[14] M. Jerrum and U. Vazirani. A mildly exponential approximation algorithm for the permanent. *Algorithmica* 16, 1996.

[15] Kantarcioglu M., Clifton C. Privacy-Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data. *IEEE TKDE*, 16(9), 2004.

[16] Y. Lindell and B. Pinkas. Privacy Preserving Data Mining. *Advances in Cryptology LNCS 1880*, Aug 2000.

[17] Moore Jr. R. A. Controlled Data-Swapping Techniques for Masking Public Use Microdata Sets. *Statistical Research Division Report Series, RR 96-04, US Bureau of Census, Washington D.C.*, 1996.

[18] Muralidhar K. and Sarathy R. Security of Random Data Perturbation Methods. *ACM TODS*, 24(4), 1999.

[19] Meyerson A. and Williams R. On the Complexity of Optimal k-Anonymity *ACM PODS Conference*, 2004.

[20] B. Pinkas. Cryptographic Techniques for Privacy Preserving Data Mining. *SIGKDD Explorations*, 2003.

[21] Rasmussen L.E. Approximating the permanent: A simple approach. *Random Structures and Algorithms*, 5, 1994.

[22] Samarati P. and Sweeney L. Protecting Privacy when Disclosing Information: k-anonymity and its Enforcement through Generalization and Suppression. *IEEE Symposium on Research in Security and Privacy*,1998.

[23] L.Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5), 2002.

[24] Traub J.F. et. al. The Statistical security of a Statistical Database. *ACM TODS*, 9(4), 1984.

[25] Valiant L.G. The Complexity of Computing the Permanent. *Theoretical Computer Science*, 8, 1979.

[26] Verykios V. et. al. Association Rule Hiding. *IEEE TKDE*, 16(4) 2004.

[27] Yang X. and Li C. Secure XML Publishing without Information Leakage in the Presence of Data Inference. *VLDB Conference*, 2004.