Anonymizing Transaction Databases for Publication

Yabo Xu, Ke Wang
Simon Fraser University
BC, Canada

Ada Wai-Chee Fu
The Chinese University of Hong Kong
Shatin, Hong Kong

Philip. S. Yu
University of Illinois at Chicago
IL 60607, USA

{yxu, wangk}@cs.sfu.ca adafu@cse.cuhk.edu.hk psyu@cs.uic.edu

ABSTRACT

This paper considers the problem of publishing "transaction data" for research purposes. Each transaction is an arbitrary set of items chosen from a large universe. Detailed transaction data provides an electronic image of one's life. This has two implications. One, transaction data are excellent candidates for data mining research. Two, use of transaction data would raise serious concerns over individual privacy. Therefore, before transaction data is released for data mining, it must be made anonymous so that data subjects cannot be re-identified. The challenge is that transaction data has no structure and can be extremely high dimensional. Traditional anonymization methods lose too much information on such data. To date, there has been no satisfactory privacy notion and solution proposed for anonymizing transaction data. This paper proposes one way to address this issue.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining; K.4.1 [**Public Policy Issues**]: Privacy

General Terms

Algorithms, Theory, Performance, Experimentation

Keywords

Anonymity, transaction database, privacy, data publishing

1. INTRODUCTION

1.1 Motivations

In this paper, a transaction is an arbitrary set of items chosen from a large universe. Examples of transactions are web search queries, purchase records, click streams, emails. Transaction data are generated in a wide variety of activities including querying and browsing web services, online/offline shopping and product reviews. This has made transactions rich sources for data mining [15], including association rule mining [8], user behavior prediction [14], recommender systems (http://www.amazon.com/), information retrieval [18] and personalized web search [19]. Detailed transaction data provides an electronic image of one's life, possibly containing sensitive information. Therefore, before data can be released for data mining, it must be made anonymous so that data subjects cannot be re-identified. We first consider two examples for re-identification on transaction data.

KDD'08, August 24–27, 2008, Las Vegas, Nevada, USA.

Example 1 AOL recently released a database of query logs to the public for research purposes [1]. However, by examining query terms, the searcher No. 4417749 was traced back to Thelma Arnold, a 62-year-old widow who lives in Lilburn. Even if a query does not contain address or name, a searcher may still be re-identified from combinations of query terms that are unique enough about the searcher. According to [15], this scandal leads to not only the disclosure of private information for AOL users, but also damages to data publishers' enthusiasm on offering anonymized transaction data for research purposes.■

Example 2 A web-based retailer released online shopping data to a marketing company for customer behavior analysis. Albert, who works in the marketing company, learnt that his colleague Jane purchased a Printer, a Frame and a Camera from this website some days ago. Albert matched these items against all transaction records and surprisingly found only 3 transactions matched, out of which 2 also contains AdultToy. Albert then concluded, with 67% confidence, that Jane bought AdultToy. Although privacy policies may be in place, nothing will stop such attacks on an individual. ■

In these examples, the data publisher (i.e., the retailer) publishes a collection of person-specific transactions for research purposes. Each transaction contains an arbitrary set of *items* chosen from a universe *U*. An item can be either *public* (i.e., Printer, Frame and Camera) or *private* (i. e., AdultToy). One data recipient, the attacker (i.e., Albert), seeks to re-identify the subject of some transactions. As *prior knowledge*, the attacker knows that a target person (i.e., Jane) has a transaction in the published data and that the transaction contains certain public items (e.g., Printer, Frame and Camera). The re-identification is successful if very few transactions contain these items.

The publisher's goal is publishing the *data*, not data mining results. The publisher has no interest or ability in data mining and the data recipient wants to receive the data and have the complete control over how to mine the data. This scenario is different from publishing data mining results so that no sensitive information is revealed such as in [5][6][7]. Also, the published data should be "semantically interpretable". For example, the recipient may want to visually examine each transaction; therefore, publishing encrypted data, or randomized data [9], or synthetic data [12] data does not serve our purposes. Our scenario requires publishing sensitive information, but hiding the identity of data subjects. This is different from hiding sensitive information in the above works.

Privacy models such as *k-anonymity* [4] and *l-diversity* [11] exist to prevent re-identification attacks on relational data. The key is to form "equivalence classes" on a *quasi-identifier* (QID), e.g., {Sex, Zip, BirthDate}, so that the records in the same equivalence class are not distinguishable. If applied to transaction data, the

QID would contain one attribute for each public item in the universe U. Typically, U for transaction data is very large, say 10,000 items in Example 2, and each transaction contains a small fraction of the items in U, say 1% or less. For such high dimensional QID, forming equivalence classes means suppressing mostly all items. A similar observation was made in the previous study [21]. But that study did not provide a solution to the high dimensional problem.

A few works [2][3] have attempted to anonymize query logs, but considered only re-identifications via one or two query terms. The work on anonymizing social networks [16] is loosely related to ours. In [16], the authors model prior knowledge as small subgraphs with a few nodes, in a similar way that we model attackers' prior knowledge as subsets of public items. In [17], authors demonstrated that the anonymity of Netflix subscribers can be compromised by as little prior knowledge as no more than 8 movie ratings and dates. These works however did not propose solutions to the problems identified.

1.2 Contributions

Given a large universe U, it is unlikely that an attacker has prior knowledge on the status of all public items in U. Instead, the attacker is constrained by the "effort" required to acquire prior knowledge on each item (e.g., search the Yellow Books, hire a spy). In this paper, we measure the "power" of the attacker by the maximum number p of public items that can be obtained as prior knowledge in a single attack, and measure the level of protection *relative* to that power of attackers. Given the ultimate goal of publishing data, this "relative protection" of privacy, which is not security, makes sense; "absolute protection" means no data publishing at all – a safe but totally useless solution.

Contribution 1 Our first contribution is a novel privacy notion for transaction data. We say that a database D has (h,k,p)-*coherence* if, for every such combination β of no more than p public items, either no transaction contains β, or the set of transactions containing β, called β-*cohort*, contains at least k transactions and no more than h percent of these transactions contains a common private item. In other words, (h,k,p)-coherence ensures that, for an attacker with the power p, the probability of linking an individual to a transaction is limited to 1/k and the probability of linking an individual to a private item is limited to h.

Example 3 (The running example) Suppose that a health care provider published the database D in Figure 1 for research on life styles and illnesses. "Activities" refers to the activities a person engages in (e.g., drinking, smoking) and are public. "Medical History" refers to the person's major illness and is private. Each person can have an arbitrary number of activities and illness chosen from a universe U. Let k = 2, p = 2, h = 80%. D violates (h,k,p)-coherence. *ab*-cohort (we use *ab* for {*a*,*b*}) has only one transaction T_2, so an "attacker" acquiring the prior knowledge *ab* on a target individual can uniquely identify T_2 as the transaction of the individual. *bf*-cohort has two transactions T_2 and T_3, both containing "Hepatitis". So an "attacker" acquiring the prior knowledge *bf* can infer "Hepatitis" with 100% probability. ■

TID	Activities	Medical History
T_1	a, c, d, f, g	Diabetes
T_2	a, b, c, f	Hepatitis
T_3	b, d, f, x	Hepatitis
T_4	b, c, g, y, z	HIV
T_5	a, c, f, g	HIV

Figure 1. D, k = 2, p = 2, h = 80%

Let us explain why (h,k,p)-coherence better preserves information than the QID-based k-anonymity. Consider a set of public items β with |β|≤p. β-cohort requires its transactions to contain all the items in β, but not necessarily contain items not in β. In contrast, QID contains one attribute for each public item in the universe U, and an equivalence class on QID must agree on *all the items in U*. Since p is typically much smaller than |QID|, more items are suppressed to form an equivalence class on QID than to form a β-cohort. For example, to form the equivalence classes EC_1={T_1,T_2,T_5} and EC_2={T_3,T_4} on the QID containing the 9 activities in Example 3, EC_1 requires suppressing *b*, *d* and *g* and EC_2 requires suppressing all activities except *a* and *b*. In the end, only the item *a* remains.

Another interesting property of (h,k,p)-coherence is that it allows prediction of private items for the *researcher*, though not for the attacker. Suppose that the health care provider in Example 3 is interested in predicting illnesses, D contains the useful pattern *bf*→ Hepatitis with 100% probability. However, for an attacker with the power p=1, and for k = 2 and h = 80%, D is (h,k,p)-coherent. In particular, *bf* is not a threat because its size exceeds the power p=1 of the attacker. This is interesting because accurate patterns β→e usually involve a long antecedent β [20], which sets up a high bar to acquire such β as prior knowledge. Our privacy notion exploits the difference between an attacker and a genuine researcher: the former must acquire prior knowledge, but the latter does not.

Contribution 2 Our second contribution is an algorithm for achieving (h,k,p)-coherence while preserving as much information as possible. We measure information loss by the amount of items suppressed. We show that an optimal solution is NP-hard and focus on finding a local optimal solution. The challenge is eliminating all damaging prior knowledge from the database, i.e., all subsets β of public items, |β|≤p, that violate (h,k,p)-coherence. For example, with p=4 and 1000 public items, the number of such subsets β can be as large as 1000^4 and enumerating all is not an option. We propose an efficient algorithm to eliminate such prior knowledge from the database.

The rest of the paper is organized as follows. Section 2 defines the notion of (h,k,p)-coherence and the problem of achieving (h,k,p)-coherence. Section 3 presents our solution. Section 4 presents experimental results. Section 5 concludes the paper.

2. PROBLEM STATEMENTS

Let U={e_1,…,e_m} be the universe of items. An item is either *public* or *private* (but not both). Public items correspond to potentially *identifying information* on which prior knowledge could be acquired by an attacker. Private items correspond to *sensitive information* to be protected. An *itemset* is a set of items from U. A *public itemset* is an itemset containing only public

items. $D=\{T_1,\ldots, T_n\}$ denotes a database of transactions. Each *transaction* T_i is a set of items from U and corresponds to an individual. If an individual has several transactions, we merge all his transactions into a single transaction.

Private items typically refer to financial information, health information, sexual orientation, religion and political beliefs. Public items refer to any items that are potentially public, therefore, all non-private items. In specialized applications such as heath care, financial sectors and insurance industry, well defined guidelines for public/private items often exist. Public/private items may also be specified by data subjects during data collection. For our discussion purpose, we assume that public/private items have been specified.

To launch an attack on a target individual, the attacker must know that the individual has a transaction in D. Also, the attacker has the prior knowledge that the transaction contains some public items β. Let |β| denote the number of items in β. We describe such an attack by $\beta\rightarrow e$, for some private item e that the attacker intends to infer. β-*cohort* refers to the set of transactions that contain β as a subset. Sup(β), the *support* of β, denotes the number of transactions in β-cohort. The probability that a transaction contains e, given that it contains β, is

$$P(\beta\rightarrow e)=Sup(\beta\cup\{e\})/Sup(\beta).$$

We define $P_{breach}(\beta)$, called the *breach probability* of β, to be the maximum $P(\beta\rightarrow e)$ for *any* private item e. Since $P(\beta\rightarrow e)\geq P(\beta\rightarrow\alpha)$ for any set α containing e, we consider only a single private item e in an attack $\beta\rightarrow e$.

In Example 2, the attacker has the prior knowledge β={Printer, Frame, Camera} and finds that, out of the 3 transactions that contain β, 2 also contains e =AdultToy. Sup(β)=3, Sup(β∪{e})=2, and $P(\beta\rightarrow e)$=2/3. So the attacker infers that Jane bought AdultToy with the probability $P(\beta\rightarrow e)$=2/3=67%.

2.1 Coherence

Our goal is to bound $P_{breach}(\beta)$ for all possible public itemsets β that can be acquired by the attacker as prior knowledge on a target individual. As the size |β| increases, so does the attacker's effort required to acquire the prior knowledge β. Suppose that the "power" of the attacker is measured by the maximum size |β| of such prior knowledge β. An attacker with the power p can potentially acquire any public itemset β as prior knowledge on a target individual, where $|\beta|\leq p$. If either $Sup(\beta)<k$ or $P_{breach}(\beta)>h$, by focusing on the transactions in β-cohort, the attacker is able to link a target individual to a transaction with more than $1/k$ probability, or to a private item with more than h probability. To prevent such linking, we define the following privacy notion.

Definition 1 (Coherence) Let β be a public itemset with $|\beta|\leq p$ and Sup(β)>0. β is called a *mole* wrt (h,k,p) if either $Sup(\beta)<k$ or $P_{breach}(\beta)>h$; otherwise, β is called a *non-mole* wrt (h,k,p). We say that D is (h, k, p)-*coherent* if D contains no moles wrt (h,k,p). ■

Intuitively, a mole is a piece of prior knowledge that could be used to link a target individual to a transaction with more than $1/k$ probability, or to a private item with more than h probability. If a database is (h,k,p)-coherent, such linking is not possible.

Example 4 In Example 1, suppose that Ms Thelma Arnold searches her own name and "Diabetes", her query log will be T_i={Thelma, Arnold, Diabetes}. If β={Thelma, Arnold} is unique in D, Sup(β)=1 and β is a mole wrt (k = 2, p = 2, h = 80%). In Example 3, where k = 2, p = 2, h = 80%, *ab* is a mole because Sup(*ab*)=1. *bf* is a mole because $P_{breach}(bf)$=100%. *a* is a non-mole because Sup(*a*)=3 and $P_{breach}(a)$=1/3. ■

2.2 Item Suppression/Information Loss

If D does not have coherence, we will modify D to satisfy coherence before publishing it. We believe that the modification operation should satisfy the following requirements: (R1) Private items are essential for research and should remain intact. (R2) A published transaction should not contain external items not in the original transaction. (R3) The support of any itemset in the published data should be the same as in the original data. Many data mining problems rely on the support of itemsets [8][13][20]. For example, association rule mining [8] is aimed to find all patterns $\alpha\rightarrow\beta$, where α and β are itemsets, such that $Sup(\alpha\cup\beta)/Sup(\alpha)$ is above some threshold value. Such patterns are good candidates for prediction and classification [20]. If $Sup(\alpha\cup\beta)$ or $Sup(\alpha)$ on the modified data is different from those obtained on the original data, even a small difference could lead to a significant difference in $Sup(\alpha\cup\beta)/Sup(\alpha)$, thus an arbitrary prediction or classification.

Item Suppression To meet the above requirements, we consider *suppression* of public items to achieve coherence. By suppressing an item from D, we simply delete the item from *all* transactions that contain the item. For example, after suppressing the items *a* and *f* from the transactions {*a, b,* HIV}, {*a, d, f*, HIV} and {*b*, *d*, Diabetes}, the transactions become {*b*, HIV}, {*d*, HIV} and {*b, d*, diabetes}.

Observation 1 By suppressing an item, (1) every itemset containing the item are eliminated from the database, and (2) every itemset that remains in the database has the same support as in the original database. Therefore, item suppression satisfies R2 and R3. In contrast, suppressing an item from some but not all transactions that contain the item will violate R3 because the support statistic may be altered. We do not consider such partial suppression.

Suppression of each item leads to some loss of information. The best way to model information loss is considering the specific purpose of data. However, this approach is not applicable if the purpose of the data is not known at the time of publication. Publication on the web and publication as required by law, such as "public records", are such examples because there is no specific data recipient. Another downside of specialized information metrics is that a different release must be published for each purpose, leaving the attacker with multiple releases to launch more powerful attacks.

Information Loss Our approach is to let the data publisher assign a certain information loss to the suppression of an item e, denoted IL(e), based on some perceived importance of the item. In particular, IL(e)=1 charges one unit of information loss for the item e suppressed, and IL(e)=Sup(e) charges one unit of information loss for each occurrence of the item e suppressed. The latter penalizes more the suppression of an item e that occurs in more transactions. Suppose that D is transformed to D' by suppressing zero or more public item. $IL(D, D')=\sum IL(e)$ denotes

the total information loss in the transformation, where $\sum$ is over all the items *e* suppressed.

2.3 Problem

Here is the problem we want to study.

Definition 2 *D*' is called a (*h*,*k*,*p*)-*cohesion* of *D* if *D* is transformed to a (*h*,*k*,*p*)-coherent *D'* by suppressing some public items. *D'* is called an *optimal (h,k,p)-cohesion* of *D* if *D'* is a (*h*,*k*,*p*)-cohesion of *D* and for any other (*h*,*k*,*p*)-cohesion *D"* of *D*, IL(*D*, *D"*)≥ IL(*D*, *D'*). The optimal cohesion problem is to find an optimal (*h*,*k*,*p*)-cohesion of *D*.■

Theorem 1 *D* has no (*h*,*k*,*p*)-cohesion if and only if the empty itemset is a mole wrt (*h*,*k*,*p*).

Proof: The empty itemset is a mole if some private item is contained in more than *h* percent of all transactions. If the empty itemset is a mole, it cannot be eliminated by item suppression and *D* has no (*h*,*k*,*p*)-cohesion. If *D* has no *(h*,*k*,*p*)-cohesion, the empty itemset must be a mole, otherwise suppressing all public items would give a (*h*,*k*,*p*)-cohesion. ■

Theorem 2 For *k*=2, *p*=2, and IL(*e*)=1, the optimal cohesion problem is NP-hard.

Proof: The following vertex cover problem is NP-hard[1]: A *vertex cover* for an undirected graph *G* = (*V*,*E*) is a subset *S* of its vertices such that each edge has at least one endpoint in *S*. To map an instance of the vertex cover problem to an instance of optimal cohesion problem, let the item universe *U* contain all the vertexes in *V* and let the database *D* contain a transaction {*a*,*b*} for each edge <*a*,*b*> in *E*. Let all items in *U* be public items. For *k*=2, *p*=2 and any *h*, every transaction {*a*,*b*} in *D* is a mole because {*a*,*b*} has support 1. So for IL(*e*)=1, *S* is a vertex cover for *G* if and only if *D'* is an optimal (*h*,*k*,*p*)-cohesion of *D*, where *D'* is *D* after suppressing the items in *S*. ■

Since the optimal cohesion problem is inherently hard, we consider a heuristic solution to the problem. From now on, we assume that *D* has a (*h*,*k*,*p*)-cohesion.

3. GREEDY ALGORITHM

For a given loss metric IL(*e*), we want to suppress some public items from *D* such that the resulting database is (*h*,*k*,*p*)-coherent and $\sum$IL(*e*) over all suppressed items e is minimized. To prune the search space, the meaningful first step is to suppress all public items that must be suppressed. A public item must be suppressed if the item on its own is a mole, in which case this mole cannot be eliminated unless the item is suppressed. This observation is stated below.

Observation 2 If a public item is a (size-1) mole, the item will not occur in any (*h*,*k*,*p*)-cohesion of *D*, thus, can be suppressed in a preprocessing step. In Figure 1, each of *x*, *y*, *z* is a size-1 mole. Figure 2 shows the database after suppressing them. For tracking changes, we cross out a suppressed item instead of deleting it. In the following discussion, we assume that all size-1 moles have been suppressed from *D*. The remaining task is to eliminate all moles of size in [2,*p*] from *D*.

[1] http://en.wikipedia.org/wiki/Vertex_cover_problem

TID	Activities	Medical History
T_1	a, c, d, f, g	Diabetes
T_2	a, b, c, f	Hepatitis
T_3	b, d, f, ~~x~~	Hepatitis
T_4	b, c, g, ~~y~~, ~~z~~	HIV
T_5	a, c, f, g	HIV

Figure 2. *D* after the preprocessing step

There are two cases for a mole β wrt (*h*,*k*,*p*): either Sup(β)<*k* or $P_{breach}(\beta)>h$. For a mole β with Sup(β)<*k*, every superset β' of β with |β'|≤*p* is also a mole because Sup(β')≤Sup(β)<*k*; for a mole β with $P_{breach}(\beta)>h$ and Sup(β)≥*k*, a superset β' of β may or may not be a mole because $P_{breach}(\beta')\geq P_{breach}(\beta)$ does not always hold. The first case implies that the number of moles may grow fast and considering all moles is not practical. We consider a smaller set of "minimal moles" defined below.

Definition 3 A mole is *minimal* if every proper subset is a non-mole.■

Example 5 In Figure 1, all of *x, y, z* are size-1 minimal moles because the empty itemset is not a mole. All of *ab, ad, bd, bf, bg, cd, dg* are size-2 minimal moles. For example, *ab* is a minimal mole because it is a mole but the subsets *a* and *b* are non-moles. ■

Observation 3 *D* is (*h, k, p*)-coherent if and only if *D* contains no minimal mole. This observation follows because each mole contains a minimal mole, so if we can eliminate all minimal moles, we also eliminate all moles.

Our strategy is greedily eliminating minimal moles. Let MM(*e*) denote the number of minimal moles containing the public item *e*. By suppressing the item *e*, we eliminate MM(*e*) minimal moles at the cost of IL(*e*) information loss. To eliminate all minimal moles and minimize information loss, we greedily suppress the public item *e* that maximizes MM(*e*)/IL(*e*). The following algorithm is based on this heuristics.

Suppression Algorithm Figure 3 outlines our item suppression algorithm for achieving coherence. Line 1 suppresses all size-1 moles in the preprocessing step (Observation 2). Subsequently, in each iteration Line 3 suppresses a remaining public item *e* having the maximum MM(*e*)/IL(*e*). There are two key steps. The first key step is checking if there are minimal moles in *D* on Line 2. The second key step is identifying the item *e* to suppress on Line 3 since MM(*e*) is dynamically changing. In the rest of this section, we propose an efficient algorithm for these steps. The general idea is first finding all minimal moles from *D* and then maintaining minimal moles and MM(*e*) in each iteration.

1. suppress all size-1 moles from *D* (Observation 2);
2. **while** there are minimal moles in *D* **do**
3. suppress the public item *e* with the maximum MM(*e*)/IL(*e*) from D;

Figure 3. Outline of greedy algorithm

3.1 Identifying Minimal Moles

Let us consider how to find all minimal moles. We assume that public items are ordered according to some pre-determined order.

Imagine that all public itemsets are organized into the lattice with subsets below supersets. To find all minimal moles, we start from size-1 non-moles at the bottom and walk up the lattice if the current node is a non-mole (Definition 1) and contains no mole. We stop walking up when the current node becomes a mole *for the first time*, at which point the current node is a minimal mole because every subset is a non-mole. The non-moles that contain no mole have potential to be extended into a minimal mole. This type of non-moles is defined below.

Definition 4 A non-mole is said to be *extendible* if it contains no mole. ■

Essentially, the above walking up of the lattice corresponds to constructing extendible non-moles in the growing size until it reaches minimal moles. Let M_i denote the set of all minimal moles of size i and let F_i denote the set of all extendible non-moles of size i. Let $\beta = \langle e_1, \ldots, e_{i-1}, e_i, e_{i+1}\rangle$ be a minimal mole in M_{i+1}. From Definition 3, no i-subset of β is in M_i, and both $\langle e_1, \ldots, e_{i-1}, e_i\rangle$ and $\langle e_1, \ldots, e_{i-1}, e_{i+1}\rangle$ are in F_i. Similarly, from Definition 4, for an extendible non-mole $\beta=\langle e_1, \ldots, e_{i-1}, e_i, e_{i+1}\rangle$ in F_{i+1}, no i-subset of β is in M_i, and both $\langle e_1, \ldots, e_{i-1}, e_i\rangle$ and $\langle e_1, \ldots, e_{i-1}, e_{i+1}\rangle$ are in F_i. This gives rise to the following construction.

Observation 4 Every minimal mole in M_{i+1} and every extendible non-mole in F_{i+1} has the form $\beta=\langle e_1, \ldots, e_{i-1}, e_i, e_{i+1}\rangle$, such that $\langle e_1, \ldots, e_{i-1}, e_i\rangle$ and $\langle e_1, \ldots, e_{i-1}, e_{i+1}\rangle$ are in F_i, no i–subset of β is in M_i, and e_i precedes e_{i+1}.

1. find M_1 and F_1 in one scan of D;
2. **while** $i<l$ and F_i is not empty **do**
3. generate the candidate set C_{i+1} for M_{i+1} and F_{i+1} from F_i based on Observation 4;
4. scan D to count Sup(β) and P_{breach}(β) for all β in C_{i+1};
5. **forall** β in C_{i+1} **do**
6. **if** Sup(β) < k or P_{breach}(β)>h
7. **then** add β to M_{i+1} **else** add β to F_{i+1};
8. i++;
9. output all M_i;

Figure 4. Identifying minimal moles

Figure 4 shows the construction of M_{i+1} and F_{i+1}. Line 1 finds M_1 and F_1 in one scan of D. At level $i\geq 1$, Line 3 generates all candidates $\langle e_1, \ldots, e_{i-1}, e_i, e_{i+1}\rangle$, C_{i+1}, for M_{i+1} and F_{i+1} based on Observation 4. At Line 4, Sup(β) and P_{breach}(β) of all candidates β in C_{i+1} are computed in one scan of D. At Line 6-7, candidates are added to M_{i+1} or F_{i+1} based on Sup(β) and P_{breach}(β) following Definition 3 and Definition 4.

The above computation shares some similarity with Apriori [8] for finding frequent itemsets, where an itemset is frequent if its support is above some threshold. The key to Apriori is that every proper subset of a frequent itemset is a frequent itemset. However, a minimal mole does not have this property because a mole involves conditions on both breach probability and support. Instead, every proper subset of a minimal mole and an extendible non-mole is an extendible non-mole. Therefore, we have to construct M_{i+1} and F_{i+1} in parallel. Observe that F_i is not larger than the set of frequent itemsets wrt the minimum support k because each non-mole is a frequent itemset wrt k. For this reason, finding minimal moles is not more expensive than finding frequent itemsets.

3.2 Eliminating Minimal Moles

Let M^* denote the set of all minimal moles of size $2\leq i\leq p$ found in Section 3.1 (size-1 moles have been eliminated from D). In the next step, we eliminate all moles in M^* from D. This is done by iteratively suppressing the remaining public item e with the maximum MM(e)/IL(e). The key is computing MM(e) and IL(e). For a remaining public item e' with $e'\neq e$, by suppressing e, IL(e')=Sup(e') is not affected and all minimal moles that contain both e' and e are eliminated (Observation 1). Therefore, MM(e') should be decreased by the number of minimal moles that contain both e' and e. We introduce the following MOLE-tree to organize minimal moles and update MM(e').

Definition 5 (*MOLE*-tree) The *MOLE-tree* for M^* contains the root labeled "null". Each root-to-leaf path represents a minimal mole in M^*. Each node (except for the root) has three fields: *label* - the item at this node; *mole-num* - the number of minimal moles that pass this node; *node-link* – the link pointing to the next node with the same label. The *Score* table contains three fields for each remaining public item e: MM(e), IL(e), *head-of-link*(e) that points to the first node on the node-link for e. ■

The key property of the MOLE-tree is that, for each public item e in the Score table, we can find all minimal moles containing e by following the node-link for e, starting from head-of-link(e).

Example 6 Figure 5 shows the MOLE-tree for M^*={*db, da, dg, dc, ba, bg, bf*}, where items are arranged in the descending order of MM(e). Let IL(e)=Sup(e). The node <b:3> means that 3 minimal moles pass the node, i.e., *ba, bg, bf*. The entry <b:4,3> in the Score table means that MM(b)=4, IL(b)=3. The 4 minimal moles containing b are found by following the link in the entry: *db, ba, bg, bf* . If b is suppressed, we can find and delete these minimal moles by following this link.■

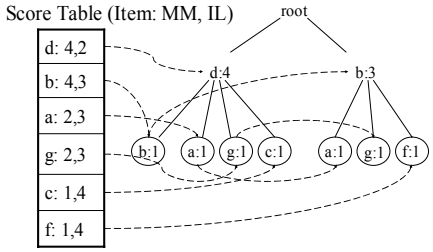


Figure 5. MOLE-tree, k=2, p=2, h=80%

Figure 6 describes the overall algorithm for eliminating all minimal moles from D. Line 1 deletes all size-1 moles. Line 2 finds all size-2 or larger minimal moles M^*. Line 3 builds the MOLE-tree. Line 5-9 iteratively selects the public item e with the maximum MM(e)/IL(e) for suppression. The main step is deleting e from the MOLE-tree (Line 7-9), i.e., deleting all minimal moles containing e. To delete all minimal moles containing e, for each *node* on the node-link of e, Line 8 deletes the entire subtree at

node and Line 9 updates the ancestors of *node*. Finally, Line 10 suppresses all items in *SuppItem* from *D*. Let us explain Step 8 and 9 in details.

Step 8: This step deletes all the minimal moles in the subtree at *node*. For each node *w* in the subtree at *node*, if *w* has the label *e'*, MM(*e'*) is decremented by mole_num(*w*), to account for the elimination of all minimal moles passing *w*. If MM(*e'*) becomes 0, delete the entry for *e'* from the Score table.

Step 9: This step updates the mole_num for all ancestors of *node*. For an ancestor node *w* of *node*, if *w* has the label *e'*, mole_num(*w*) and MM(*e'*) are decremented by mole_num(*node*), to account for the elimination of all minimal moles passing *node*. If mole_num(*w*) becomes 0, delete the node *w*. If MM(*e'*) becomes 0, delete the entry for *e'* from the Score table.

```
1.  let D be the database with size-1 moles removed;
2.  find minimal moles M* from D (Figure 4);
3.  build the MOLE-tree for M*;
4.  initialize SuppItem to the empty set;
5.  while Score table is not empty do
6.      add the item e with the maximum MM(e)/IL(e) to SuppItem;
7.      forall each node on the node-link for e do
8.          delete all minimal moles that pass node;
9.          update the mole-num at node's ancestors;
10. suppress all items in SuppItem from D;
```

Figure 6. Overall greedy algorithm

Example 7 Consider the MOLE-tree in Figure 5. Since the item *d* has the maximum MM/IL, we first suppress *d* by deleting all minimal moles passing the (only) node for *d* (Line 8). To do this, we can traverse the subtree at the node for *d* and decrease MM for *b*, *a*, *g*, and *c* by 1, and decreases MM for *d* by 4. Since MM(*d*) and MM(*c*) become 0, the entries for *d* and *c* are deleted from the Score table. The new MOLE-tree and Score table are shown in Figure 7. Next, the item *b* has the maximum MM/IL and is suppressed. As a result, all remaining moles are deleted and now the Score table becomes empty. ■

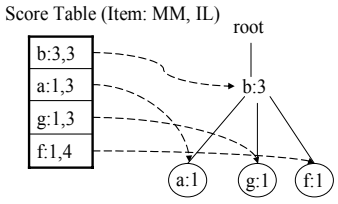


Figure 7. MOLE-tree after deleting *d*

Cost Analysis The overall work consists of two parts. The first part involves finding minimal moles (Figure 4). As discussed in Section 3.1, this part is not more expensive than finding frequent itemsets. The second part involves inserting and deleting minimal moles using the MOLE-tree (Figure 6). Each minimal mole is inserted into and deleted from the MOLE-tree exactly once, thus, the work is proportional to the number of minimal moles |*M**|. The benefit of dealing with minimal moles is three-fold: speed up the work for finding minimal moles, reduce the space for storing the MOLE-tree, and reduce the work for eliminating moles.

4. EMPIRICAL STUDIES

This section evaluates the data distortion of the proposed anonymization. The data distortion is defined by the percentage of item occurrence suppressed, *S/N*. *S* is the occurrence of the items suppressed, i.e., ∑IL(*e*) over all suppressed items *e* with IL(*e*)=Sup(*e*). *N* is the occurrence of all items in the original database. We consider the following approaches:

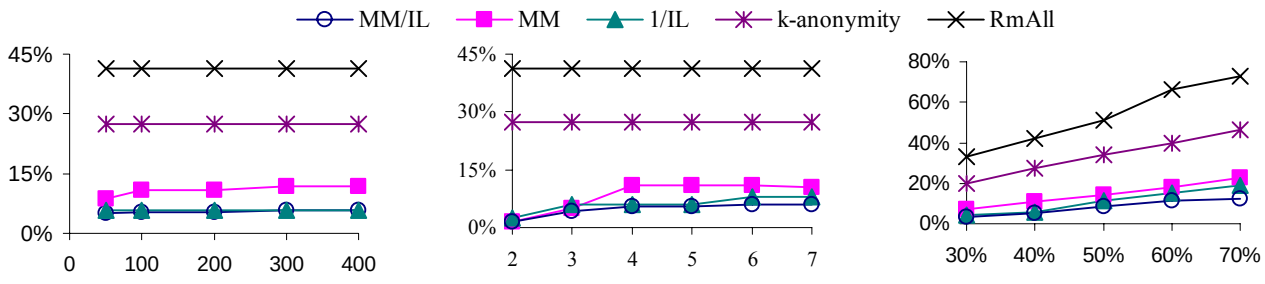
- "RmAll" – suppress all public items. Thus, *S/N* is the percentage of the occurrence of public items over the occurrence of all items.
- "*k*-anonymity" – *k*-anonymization on the QID containing all public items. We used the global recoding *TDS* in [10] because it preserves the support of remaining itemsets (see Observation 1 and the follow-up discussion). Since *l*-diversity is also based on the QID, we consider only *k*-anonymization.
- "MM/IL" – greedily suppress the public item with the maximum MM/IL. This is the method presented in Section 3. We also consider two variants: "MM" – greedily suppress the public item with the maximum MM, and "1/IL" – greedily suppress the public item with the minimum IL.

We considered three vastly different datasets: Connect, Retail, and T40I10D100K, all being publicly available from FIMI Repository[2]. As a preprocessing step, we removed the items with support less than 0.1%|*D*|, where |*D*| is the number of transactions. Such items usually are pruned due to insufficient statistical significance by frequent itemset mining [8] and classification tasks [20]. Table 1 provides a brief description of these datasets after preprocessing. Connect is a real dataset containing game state information. Retail is a real dataset supplied by anonymous Belgian retail supermarket store. T40I10D100K is a synthetic dataset. Retail and T40I10D100K have a very large item universe *U*, thus, extremely high dimensionality if represented by a relational table. In this aspect, Retail is more similar to the AOL query log data set in Example 1. Connect is denser but still considered high dimensional in relational data. Due to such a diverse data characteristics, we do not expect that all data sets respond equally well in terms of data distortion.

Table 1. Dataset statistics after preprocessing

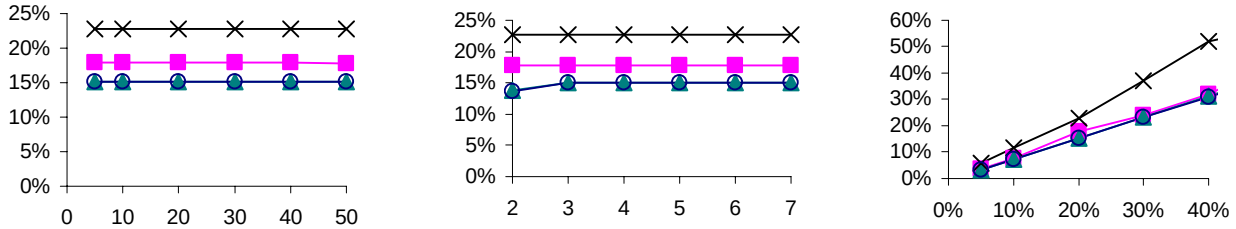
Dataset	# Trans-actions |*D*|	Average Length	# Items |*U*|	Data size (K bytes)
Connect	67,557	43	129	2,836
Retail	88,162	7.5	2117	2,930
T40I10D100K	100,000	39.4	862	15,796

[2] http://fimi.cs.helsinki.fi/data/



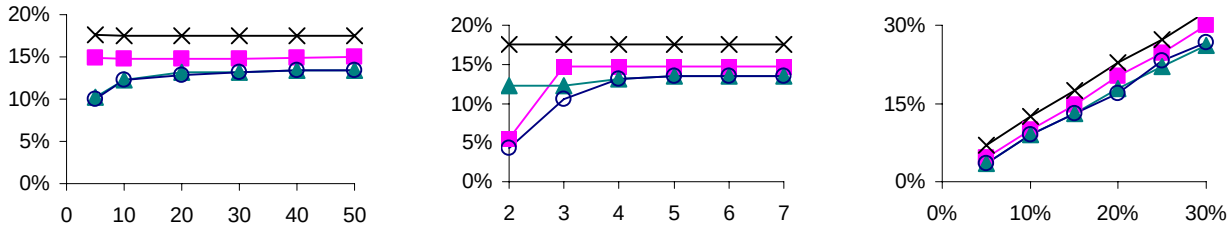
(a) Distortion vs *k* (b) Distortion vs *p* (c) Distortion vs δ

Figure 8. Distortion on Connect



(a) Distortion vs *k* (b) Distortion vs *p* (c) Distortion vs δ

Figure 9. Distortion on Retail



(a) Distortion vs *k* (b) Distortion vs *p* (c) Distortion vs δ

Figure 10. Distortion on T40I10D100K

4.1 Public Items/Private items

First we explain our strategy of specifying public and private items. We introduce a parameter δ to determine the percentage of public items. For each data set, we randomly select δ|*U*| items from *U* as public items. The set of private items consists of all non-public items *e*, where *e* occurs in some transaction as the non-public item of highest support. This choice is conservative because it generates more moles with a high breach probability, thus forcing elimination of more moles and increasing distortion. All item selections are random and the distortion reported is the average of five random selections. Below, we study the effect of δ, *h, k, p* on distortion and runtime for achieving (*h,k,p*)-coherence.

4.2 Connect

Figure 8(a-c) shows the distortion for Connect with *p, k* and δ being varied one at a time. The default setting is δ=40%, *k*=100, *p*=5. *h* has little impact and is set *h*=40%. The key findings are summarized as follows. "RmAll" has the highest distortion of 41.94%. The distortion of "*k*-anonymity" is significantly higher than "MM", "MM/IL" and "1/IL", suggesting that the traditional k-anonymization is not suitable. "MM" has a higher distortion than "MM/IL" and "1/IL". In fact, "MM" tends to suppress high frequency items to eliminate more moles at each step, but this also leads to a larger distortion. "1/IL" and "MM/IL" have a small distortion because they minimize information loss at each step. "MM/IL" sometimes suppresses high frequency items if doing so eliminates many moles, which is exactly the design goal of this selection criterion. *k* and *p* do not have a major impact on Connect because the number of moles only increases slightly. As δ increases, "RmAll" and "*k*-anonymity" distort the data more severely than other methods.

4.3 Retail

Figure 9 (a-c) shows distortion of Retail vs *k*, *p*, and δ. The default setting is δ=20%, *k*=20, *p*=4 and *h*=40%. "*k*-anonymity" did not finish within a set time limit due to a large item universe. The distortion of "MM/IL" and "1/IL" is about 8% lower than "RmAll" and the gap increases with a larger δ. Compared to Connect, this gap is smaller because this dataset is sparser and

more public items were suppressed to eliminate moles even for small k and p. With AOL query logs having similar data characteristics, we anticipate that anonymizing AOL query logs will lead to a similar data distortion.

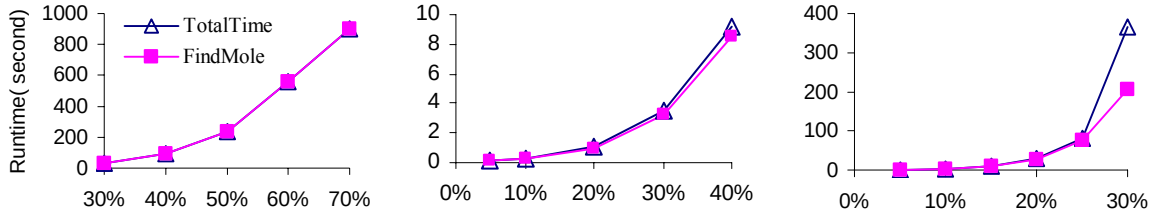
4.4 T40I10D100K

Figure 10 (a-c) shows distortion of T40I10D100K vs k, p and δ. The default setting is δ=15%, k=30, p=4 and h=40%. The trend is similar to that for Retail except that distortion is smaller. For small k and p, say k=5 and p=2, this dataset is dense enough so that most itemsets of size $\leq p$ have support $\geq k$, thus, enjoys a low distortion as seen on Connect. As k and p increase, a large portion of itemsets turns into moles, leading to a high distortion as seen on Retail.

4.5 Efficiency

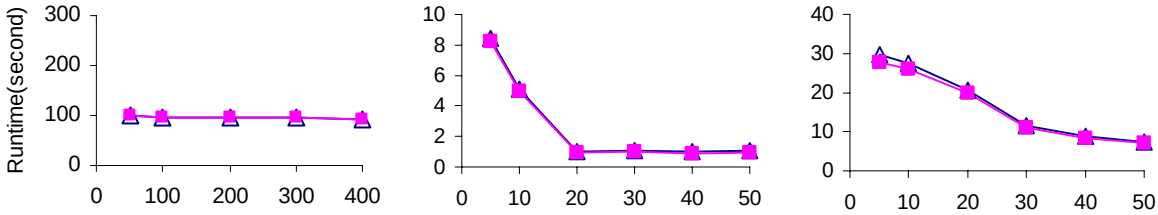
Our second goal is to evaluate the efficiency of the proposed anonymization algorithm. All programs were coded in C++ and run on a PC with 2GHz CPU, 512M memory and Windows XP. "k-anonymity" did not finish on Retail and T40I10D100K within a set time limit. Since all of "MM/IL", "MM" and "1/IL" have a similar runtime, we report only the efficiency of "MM/IL". The runtime vs h was omitted as the number of moles having high breach probability is very small. Figure 11-13 show the runtime vs δ, k, p, with the default settings used earlier. TotalTime represents the total runtime for "MM/IL" and FindMole represents the runtime for finding moles. So the time for eliminating moles is TotalTime-FindMole.

Runtime vs δ (Figure 11). For a larger δ, there are more public items and the runtime increases quickly. Recall that the search of minimal moles proceeds bottom-up in the lattice of itemsets and the set of minimal moles forms a cut of the lattice (Section 3.1). For the dense Connect, the number of moles is small, but the search of minimal moles goes into the higher part of the lattice and dominates the runtime. For the sparse Retail, there are many moles, but most are not searched because the cut for minimal moles is close to the bottom. As a result, the number of moles generated is small and little time is spent on eliminating minimal moles. For T40I10D100K, being less dense than Connect and being denser than Retail leads to many minimal moles in the middle part of the lattice. Therefore, both search and elimination of minimal moles take time.



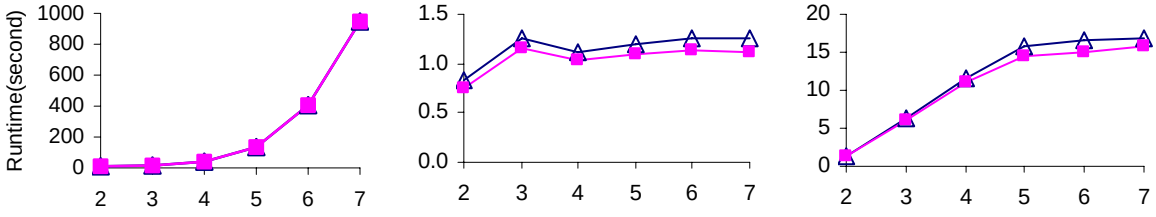
(a) Connect (b) Retail (c) T40I10D100K

Figure 11. Runtime vs δ



(a) Connect (b) Retail (c) T40I10D100K

Figure 12. Runtime vs *k*



(a) Connect (b) Retail (c) T40I10D100K

Figure 13. Runtime vs *p*

Runtime vs *k* (Figure 12). *k* has different effects on the three datasets. For the dense Connect, even when *k* is increased from 50 to 400, most itemsets have support ≥*k* and the search for minimal moles goes into many iterations, so the runtime is not reduced by a large *k*. In contrast, the runtime drops quickly on the sparse Retail. As most itemsets have a low support, the search process for minimal moles tends to stop at a lower part of the lattice with a larger *k*, which reduces the runtime quickly. The trend on T40I10D100K is similar to that on Retail.

Runtime vs *p* (Figure 13). The increase of *p* results in a boost of runtime on the dense Connect due to the increase of search space of moles. On the contrary, for Retail, the increase of *p* after *p*=3 hardly has any effect on runtime because most 3-itemsets have turned into moles and the search for minimal moles stops early. T40I10D100K follows this trend except its turning point is delayed to *p*=5.

5. CONCLUSION AND FUTURE WORK

An important research problem is anonymizing transaction databases for publication. The traditional anonymization for relational data loses too much information due to the high dimensionality of transaction data. To address this problem, we model the power of attackers by the maximum size of public itemsets that may be acquired as prior knowledge, and propose a novel privacy notion called "coherence" suitable for transactional databases. The empirical study shows that the coherence can be achieved efficiently while with a low data distortion, especially compared to the traditional privacy model such as *k*-anonymity.

This work is directly motivated by the recent privacy breaches on two well-known transactional datasets[1] [17], which have caused huge impact in public space. We believe our work represents an important step to address these privacy problems generated in real-life scenarios. Yet, we also think that there are quite a few promising directions to be explored towards a more satisfactory solution. First, the current coherence model treat every public item equally identifying, while in practice different public items may have different weights to identify a transaction. Secondly, the current model relies on the assumption that an item is either public or private. However, in certain situations, the line between public/private may be blurred, i.e. different users may treat different item as private items. We will explore all these in our future work.

6. REFERENCES

[1] M. Barbaro, T. Zeller and S. Hansell. A Face Is Exposed for AOL Searcher No. 4417749. New York Times, Aug 9, 2006.

[2] E. Adar. User 4XXXXX9: Anonymizing Query Logs. Query Log Analysis Workshop, WWW 2007.

[3] R. Kumar, J. Novak, B. Pang, and A. Tomkins. On Anonymizing Query Logs via Token-based Hashing. WWW 2007.

[4] L. Sweeney. Achieving *k*-Anonymity Privacy Protection Using Generalization and Suppression. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5), 2002.

[5] V. S. Verykios, A. K. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni. Association Rule Hiding. TKDE, 16(4):434-447, 2004.

[6] Y. Saygin, V. S. Verykios, C. Clifton. Using Unknowns to Prevent Discovery of Association Rules, Conference on Research Issues in Data Engineering, 2002.

[7] F. Bonchi, F. Giannotti and D. Pedreschi. Blocking Anonymity Threats Raised by Frequent Itemset Mining. ICDM 2005.

[8] R. Agrawal, T. Imielinski, and A. N. Swami. Mining Association Rules between Sets of Items in Large Databases. SIGMOD 1993.

[9] A. Evfimievski, R. Srikant, R. Agrawal and J. Gehrke. Privacy Preserving Association Rule Mining. SIGKDD 2002.

[10] B. Fung, K. Wang and P. Yu. Top-Down Specialization for Information and Privacy Preservation. ICDE 2005.

[11] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. *l*-Diversity: Privacy beyond *k*-Anonymity. ICDE 2006.

[12] Y. Wang, X. Wu. Approximate Inverse Frequent Itemset Mining: Privacy, Complexity, and Approximation. ICDM 2005.

[13] S. Brin, R. Motwani, and C. Silverstein. Beyond Market Basket: Generalizing Association Rules to Correlations. SIGMOD 1997.

[14] E. Adar, D. S. Weld, B. N. Bershad, S. D. Gribble. Why We Search: Visualizing and Predicting User Behavior. WWW 2007.

[15] K. Hafner. Researchers Yearn to Use AOL Logs, but They Hesitate. New York Times, August 23, 2006.

[16] L. Backstrom, C. Dwork and J. Kleinberg. Wherefore Art Thou R3579x?: Anonymized Social Networks, Hidden Patterns, and Structural Steganography. WWW 2007.

[17] A. Narayanan and V. Shmatikov. How to Break Anonymity of the Netflix Prize Dataset. *ArXiv Computer Science e-prints*, October 2006.

[18] H. Cui, J. Wen, J. Nie, and W. Ma. Probabilistic Query Expansion Using Query Logs. WWW 2002.

[19] Z. Dou, R. Song, and J. Wen. A Large-scale Evaluation and Analysis of Personalized Search Strategies. WWW 2007.

[20] B. Liu, W. Hsu, and Y. Ma. Integrating Classification and Association Rule Mining. KDD 1998.

[21] C. Aggarwal. On *k*-Anonymity and the Curse of Dimensionality. VLDB 2005