Strengthening Data Privacy During Propagation

Wei Chang and Jie Wu
Department of CIS, Temple University, PA
{wei.chang, jiewu}@temple.edu

ABSTRACT

More and more smartphone applications need microdata, but publishing a microdata table may leak respondents' privacy. Conventional research on Privacy Preserving Data Publishing (PPDP) focuses on providing an identical privacy protection. Consider that, instead of being trapped in a small coterie, information usually propagates from friend to friend. In this paper, we study the PPDP problem on a mobile social network: along a table's propagation path, how can we create a series of tables with increasing privacy protection levels? The tradeoff between the created tables' overall utility and their individual privacy requirements are not trivial, especially for distributed systems: any inappropriate sanitization operation under a lower privacy-level may cause dramatic utility loss on subsequent tables. In order to solve the problem, we propose an approximation algorithm by previewing the future privacy requirements. Simulation results show that our scheme can successfully increase the overall utility, and meet the strengthening privacy protection requirements.

Keywords

Data utility, distributed system, increasing privacy levels, mobile social networks, privacy-preserving data publishing.

1. INTRODUCTION

Learning others' social features can significantly improve the performance of many mobile social network-related tasks, such as data routing and location recommendation. In these tasks, a participant needs access to a large volume of personal information in order to spot the features. A dataset, which consists of the information at the level of individual respondents, is known as a microdata dataset. In order to protect the privacy of each individual respondent, data holders must carefully sanitize (also known as anonymize) the dataset before publishing it. In the past decade, many privacy standards have been proposed, such as k-anonymity [4], l-diversity [3], and t-closeness [2].

SPME'14, September 11, 2014, Maui, Hawaii, USA.
ACM 978-1-4503-3075-6/14/09.
http://dx.doi.org/10.1145/2646584.2646589/.

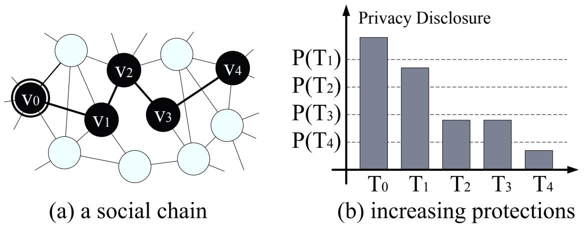

Figure 1: An example. The dash lines represent the enhanced privacy requirements for each table and the bars stand for the tables' real privacy values.

Unlike the conventional centralized database system, where data requesters *directly* interact with data owners, information on a mobile social network is disseminated from user to user via *multi-hop relays*. Due to the well-known limitations of centralized systems, such as system bottlenecks, in this paper, we study the problem of multi-hop relay-based PPDP, where a microdata table is gradually propagated from its original owner to distant people. However, the recipients present different *trust-levels* regarding to the original data owner. Therefore, after each time of relay, one should further provide more privacy protections on the data. Considering the size of transmitted package and limited energy of mobile phones, we does not allow the source node to create all tables for different receivers. For example, in Fig. 1, along a social contacting path with length 4, each participant (black node) will eventually get certain information about v_0's table. Assuming each participant can create a data table based on his obtained information, we need v_4's table to satisfy the highest privacy-level, T_3 satisfying the second highest privacy-level, and so on. This propagation scheme creates a unique problem: 'for a group of friends, how can they create a series of tables with maximal overall data utility, and assure that the tables' levels of privacy are increasingly protected at the same time?' To our best knowledge, this problem has never been proposed or solved.

Take Fig. 2 as an example. Suppose that v_0, v_1, and v_2 are the participants, l-diversity is the privacy requirement, and the total target propagation distance is equal to 2. After the first hop of propagation, the resulting table T_1 should satisfy 2-diversity, and after the second hop, T_2 should satisfy 3-diversity. Fig. 2(a) shows the original dataset on v_0. Figs. 2(b) and (c) give the results by directly using anonymizing operations on Fig. 2(a). We can see that T_1 and T_2's sanitized values are different. However, during multi-hop relays, a participant can only observe the table passed from the previous one: if v_0 gives T_1 to v_1, v_2 can

Name	Age	Zipcode	Disease
Ashley	23	19024	Hepatitis
Brooke	23	19024	Brochitis
Charish	28	19024	Flu
Dave	28	19122	Cancer
Ellen	29	19122	Hepatitis
Frank	24	19122	Brochitis

(a) T0: Patient Data (original)

	Age	Zipcode	Disease
1	23	19024	Hepatitis
2	23	19024	Brochitis
3	28	19***	Flu
4	28	19***	Cancer
5	2*	19122	Hepatitis
6	2*	19122	Brochitis

(b) T1: 2-diversity table

	Age	Zipcode	Disease
1	2*	19024	Hepatitis
2	2*	19024	Brochitis
3	2*	19024	Flu
4	2*	19122	Cancer
5	2*	19122	Hepatitis
6	2*	19122	Brochitis

(c) T2: 3-diversity table

	Age	Zipcode	Disease
1	2*	19***	Hepatitis
2	2*	19***	Brochitis
3	2*	19***	Flu
4	2*	19***	Cancer
5	2*	19***	Hepatitis
6	2*	19***	Brochitis

(d) T3: 3-diversity table (based on T1)

Figure 2: Generated tables based on requirements with different privacy levels.

only obtain Fig. 2(c), instead of T_2. Consider that the tables, which satisfy (l+1)-diversity, must satisfy l-diversity. v_0 can either send T_3 (option I) or T_2 (option II) to v_1. For option I, v_1 only needs to forward T_3 to v_2 without any changes, while for option II, v_1 should further sanitize T_2 and send the result T_3 to v_2. We cannot simply claim which option preserves more data utility. For instance, if we define that any suppression operation costs 1 unit of utility, the option I loses a total of 12 units of utility (as 6 units of utility are lost for T_1), while the option II costs 16 units (as 4 utility units are lost for T_2 and 12 for T_3). However, if the age attribute is more important than the zip code, assuming that the suppression on age costs 2.5 units of utility and suppression on zip code costs 1, the utility loss of the second option becomes 28, while that of the first one is 30.

In order to solve the problem, we first propose two greedy schemes for sanitizing tables with a single privacy-requirement: a bottom-up algorithm and a top-down algorithm. The bottom-up algorithm always merges similar tuples into one equivalent class until all classes meet the corresponding privacy requirement. The top-down approach considers how to split a table into the maximum number of sub-tables, such that each small table has the required privacy protection. By comparing the results of both algorithms, we propose an approximation algorithm. Consider that if a table satisfies a higher privacy requirement, then it also meets the lower one. At each participant, the algorithm does not simply maximize the data utility at its required privacy-level; instead, it takes future, more stringent privacy requirements into consideration. Simulation results show that more data utilities are preserved by adopting our scheme.

2. MODEL AND PROBLEM

On a mobile social network, there is a social contacting chain consisting of $l+1$ participants (also called nodes), assuming $v_0, v_1, \ldots, v_l$. Let $T = \{t_1, t_2, \ldots, t_m\}$ be a microdata table of m tuples and n attributes $A = \{A_i\}$. In practice, a microdata table may contain several sensitive attributes. Since we can consider the combination of sensitive attributes to be a data point in a hyper-dimension, we simply assume that attributes $\{A_1, A_2, \ldots, A_{n-1}\}$ are the quasi-identifiers (non-sensitive), and A_n is the sensitive attribute.

In a table T, tuples with the same quasi-identifiers' values form an equivalent class $\langle t \rangle$. For example, in Fig. 2(b), $\langle t_3 \rangle$ means the equivalent class of the third tuple, and therefore, $\langle t_3 \rangle$=$\langle t_4 \rangle$= {(28, 19***, Flu); (28, 19***, Cancer)}. Let σ be the selection operation from a table, and $\sigma_i(T_j)$ stands for the ith tuple of table T_j. For instance, in Fig. 2(c), $\sigma_4(T_2)$= {(2*, 19122, Cancer)}. We use π as the projection operation on a table. $\pi_i(T)$ gives the ith column of table T. For instance, in Fig. 2(a), $\pi_3(T_0)$={Hepatitis; Brochitis; Flu; Cancer; Hepatitis; Brochitis }. Any value in table T_k, attribute (column) A_j, tuple (row) t_i is represented as $\sigma_i(\pi_j(T_k))$. In Fig. 2(b), $\sigma_3(\pi(T_1))$ ={Flu}, and $\sigma_{\langle 3 \rangle}(\pi(T_1))$ ={Flu, Cancer}. Since we assume that the nth attribute is the sensitive one, $\pi_n(T)$ indicates the sensitive values in T. For the ease of description, let $\pi(T)$ be short for $\pi_n(T)$. So, $\pi(T_0) = \pi_n(T_0)$.

Along the social chain, v_0 is the original data owner, and based on trust relations, user v_i ($i \in [1, l]$) receives an anonymized table T_i from his previous user, creates a modified version T_{i+1}, and sends it to the next user. Since trust fades along a propagation path, instead of having l tables with the same privacy-preserving strength, our system builds a series of tables with *increasing* privacy protections. Therefore, T_{i+1} is the sanitization result of T_i. For benefiting the whole society on a social chain, the data owner wants to provide the maximal *overall* utility of these tables.

In practice, different attributes may have different levels of importance; there is a set of utility weights $W = \{w_1, w_2, \ldots, w_{n-1}\}$ associated with quasi-identifiers ($w_i \in [0, 1], \sum_i w_i = 1$). The larger the weight is, the more important the attribute is. Utility weight is a *source-based* concept: along a table's propagation path, all participants use the same W since the tables have the same source. Let $p(\cdot)$ denote the distribution of a given set of values, then $p\left(\pi(T)\right)$ represents the sensitive values' distribution over the whole table T, and $p\left(\sigma_{\langle t \rangle}(\pi(T))\right)$ indicates that distribution within an equivalent class $\langle t \rangle$. For a given table T, its privacy disclosure $P(T)$ is measured as the maximum amount of privacy disclosure from its equivalent classes:

$$P(T) = \max_{\forall \langle t \rangle \subseteq T} JS\left(p(\pi(T)), p(\sigma_{\langle t \rangle}(\pi(T)))\right)$$

where $JS(\cdot, \cdot)$ is the Jensen Shannon divergence function.

Let $\sigma_i(\pi_j(T))$ and $\sigma_i(\pi_j(T'))$ be the same data point appearing in two tables. Due to the sanitization operations, their values may be different. Let $d\left(\sigma_i(\pi_j(T)), \sigma_i(\pi_j(T'))\right)$ be the sematic distance between them. The utility of an attribute mainly comes from its diversity. If we use sanitization operations to merge all tuples of table T into one equivalent class, then the resulting table $\widetilde{T}$ reaches maximum privacy but minimum utility. Let $U(T)$ stand for the data utility of any table T, then we have:

$$U(T) = \sum_{i=1}^{n-1} \left[w_i \sum_{j=1}^{m} d\left(\sigma_j\left(\pi_i(T)\right), \sigma_j(\pi_i(\widetilde{T}))\right) \right]$$

Problem Formulation: for a given table T_0, along a social path with length l, we want to create l tables; each table T_i is the sanitization result of T_{i-1} ($i \in [1, l]$), and these tables satisfy the following requirements:

$$\text{Maximize} \quad \sum_{i=1}^{l} U(T_i)$$

$$\text{Subject to} \quad P(T_i) < (l - i + 1)\delta, \text{ for } \forall i \in [1, l]$$

REMARK 1. *Given a microdata table T_0 and a pair of source-selected parameters δ, l, there is at least one series of*

Algorithm 1: BUA

```
Data: Privacy threshold $\delta$ and original table $T$
Result: The grouping index set $IDX$
/*Phase I: Create sensitive-value dominating groups*/;
Create basic group set $\{g\}$;
for Each basic group $g_i \in \{g\}$ do
    for Each type of sensitive value $a_i \in A_n$ do
        Compute distance $d$ from $g_i$ to tuples with $a_i$;
Assign dominating type $a_i$ to each group;
Each un-grouped tuple joins a group according to $d$, $a_i$;
/*Phase II: Group combinations*/;
while $\exists g_i \in \{g\}$, whose $P(g_i) > \delta$ do
    for Each group $g_j \in \{g\}$, and $g_j \neq g_i$ do
        Compute the distance $d(g_i, g_j)$;
    Sort $d(g_i, g_j)$ in ascending orders;
    for Each value in $d(g_i, g_j)$ do
        if $P(\{g_i, g_j\}) < P(g_i)$ then
            merge $g_i$ and $g_j$ into one equivalent class;
Create grouping index $IDX$ based on $\{g\}$;
Call Algorithm. 2, $IDX \leftarrow PLSA(\delta, T, IDX)$;
```

tables $T_1 = T_2 = \ldots = T_l = \pi(T_0)$ that satisfied the increasing privacy requirements $P(T_i) < (l - i + 1)\delta$ for $\forall i \in [1, l]$.

THEOREM 1. *The target problem: "given table T_0, propagation length l, and a threshold δ, maximize $\sum_{i=1}^{l} U(T_i)$ subject to $P(T_i) < (l - i + 1)\delta$ for $\forall i \in [1, l]$" is NP-hard.*

Due to paper limitations, we only give the basic idea of the proof. Consider the simplest case of our problem, where $l = 1$, and all sensitive values follow a uniform distribution. Since the amount of privacy disclosure is measured by Jensen-Shannon divergence, each type of sensitive value must appear at least once within each equivalent class. Assume that there are a total of L types of sensitive values. Under this special case, our problem becomes a typical L-diversity problem, which is NP-hard [5].

Maximizing the overall utility of l tables is more challenging than a single one. Simply maximizing a table's utility subject to a current privacy-level is not a good option. For example, in Fig. 2, tuples 3 and 4 are closer to each other, and it is optimal to put them together for a 2-diversity requirement. Once it is done, their values become the same except for the sensitive attribute column. However, distances from the modified tuples to their second closest tuples are increased (e.g. the distance between tuples 2 and 3 in T_1), which may cause extra utility losses for creating tables with a higher privacy requirement. Clearly, when sequentially creating multiple tables with increasing privacy protection, tuples grouping orders affect the overall utilities.

3. SOLUTION DETAILS

3.1 Bottom-up Approach (BUA)

There are two objections to our problem: (i) maximizing tables' overall remaining utility, and (ii) bounding each table's privacy disclosure amount. Since privacy and utility are not comparable with each other, we cannot consider these two objections at the same time. The Bottom-up Approach (BUA) gives more consideration on the amount of

Algorithm 2: Privacy Level-preserved Split Algorithm

```
Data: $\delta$, $T$, and grouping index $IDX$
Result: The grouping index set $IDX$
Setup: operating table set $S \leftarrow IDX$ and $IDX \leftarrow \phi$;
for $S \neq \phi$ do
    Fetch table $T'$ from $S$, $S \leftarrow S \setminus \{T'\}$;
    for Each type of sensitive value $a_i \in A_n$ in $T'$ do
        Assign tuples with $a_i$ to groups $S_{a_i}$ and $S'_{a_i}$;
    Create $T_1$, $T_2$ by optimally combine $\{S_{a_i}\}$, $\{S'_{a_i}\}$;
    if $P(T_1) < \delta$ and $P(T_2) < \delta$ then
        Update the operation set $S \leftarrow S \bigcup \{T_1, T_2\}$;
    else
        Update the index set $IDX \leftarrow IDX \bigcup \{T'\}$;
```

privacy disclosure. The basic idea of BUA (Algorithm. 1) is as follows: if each equivalent class dominates in one type of sensitive value, the merging of different classes may make their sensitive values' distribution closer to that of the whole table. By merging different groups together, BUA can create tables with different privacy disclosure amount.

BUA consists of two phases. Since each type of sensitive values must occur once in each equivalent class, in phase I, BUA first creates several basic groups, where each sensitive value appears once within each group. For the remaining tuples, we compute their quasi-identifiers' distances to each group. BUA proportionally assigns each basic group with a dominating sensitive value type, according to the occurrence frequency of sensitive values. Based on tuple's sensitive values and their distance to groups, each tuple joins a group. In phase II, BUA gradually merges groups in order to satisfy a given privacy requirement, and the merging process will not terminate until all equivalent classes reach the required privacy-level. Algorithm 2 is an auxiliary algorithm, which splits equivalent classes into smaller groups, while keeping their privacy-level unchanged. The algorithm is based on the fact that, for a given set of tuples, whose sensitivity values' occurrence times follow a certain distribution, if we partition the set into several subsets such that the sensitive values' occurrence frequencies in each subset are unchanged, then the amount of privacy disclosure of each subset is the same as the original set's privacy disclosure.

3.2 Top-down Approach (TDA)

Top-down Approach (TDA) considers how to partition a table into multiple sub-tables under a single-level privacy requirement. The basic idea of TDA comes from paper [1]. Our TDA is based on two insights. (1) For a given set of data $\{S\}$, the distribution of which follows $D(\{S\})$, if we partition $\{S\}$ into two subsets ($\{S_1\}$, $\{S_2\}$) and keep each value's occurrence frequency unchanged in the subsets, then we have $D(\{S_1\}) \approxeq D(\{S\})$ and $D(\{S_2\}) \approxeq D(\{S\})$. This feature maximizes the protection of privacy but hurts data utility. (2) If we randomly partition a table into several sub-tables, the larger a sub-table is, the more likely it is that the distribution of a sensitive attribute in the sub-table is close to the distribution of the attribute in the original table.

Based on these two insights, we propose TDA, as shown in Algorithm 3. TDA consists of two phases. The first phase gradually partitions the original data table into several groups. This phase tries to maximally preserve data

Algorithm 3: TDA

```
Data: Privacy threshold $\delta$ and original table $T$
Result: The grouping index set $IDX$
Set up the operation table set by $S \leftarrow \{T\}$;
Set up output equivalent class index set by $IDX \leftarrow \phi$;
while $S \neq \phi$ do
    Fetch a table $T'$ from $S$, $S \leftarrow S \setminus \{T'\}$, and create
    clusters $T_1$ and $T_2$ s.t. $T_1 \bigcap T_2 = \phi$, $T_1 \bigcup T_2 = T'$;
    if $P(T_1) < \delta$ and $P(T_2) < \delta$ then
        Update the operation set $S \leftarrow S \bigcup \{T_1, T_2\}$;
    else
        Update the index set $IDX \leftarrow IDX \bigcup \{T'\}$;
Call Algorithm. 2, $IDX \leftarrow PLSA(\delta, T, IDX)$;
```

utility and to satisfy the needs of privacy. Here, one group corresponds to an equivalent class; once a pair of tuples has been merged into one equivalent class at one participant, it is hard for the following participants to take them apart, since they cannot discriminate the tuples from the same equivalent class. Based on this consideration, during TDA Phase I, we strictly use binary partitioning: TDA first partitions the whole tuples into two groups, then splits each group into another two sub-groups, and so on. For any sub-group, the partitioning process stops when the further partitioning on it will break the privacy requirement. Due to the fact that the smaller an equivalent class is, the more data utility is preserved, the second phase of TDA tries to further reduce the size of each sub-table T, and in the meanwhile, keeps their privacy-preserving degree unchanged by using Algorithm 2.

3.3 Forward Looking Approach (FLA)

In order to achieve the optimization of the overall utility of tables, we propose another algorithm called Forward Looking-based table sanitization Approach (FLA). For reducing the impacts of locality, instead of only considering the current privacy requirement, FLA takes the next privacy-level into account. Suppose that we have a table T_0 and two privacy thresholds $(l-i+1)\delta$ and $(l-i)\delta$, where $1 < i \le l$. From T_0, FLA directly creates two tables T_1 and T_2 with maximal utilities, where $P(T_1) < (l-i)\delta < (l-i+1)\delta$ and $P(T_2) < (l-i+1)\delta$. Based on T_2, FLA further builds up another table T_3 such that $P(T_3) < (l-i)\delta$, as shown by Fig. 3(b). Next, FLA compares the utilities between $2 \times U(T_1)$ and $U(T_2) + U(T_3)$. If the former is greater, then FLA will publish T_1 under the privacy requirement as $(l-i+1)\delta$; otherwise, it will publish T_2.

FLA also improves the results from methodology. The advantage of BUA is that, by clustering tuples into groups dominating on certain sensitive values, one can create tables with different privacy-levels. However, BUA's main problem is caused by the inappropriate merging at the lower privacy requirements, which results in lots of utility loss at the conditions with higher privacy requirements. The advantage of TDA is that, during the top-down partitioning process, the intermediate partitioning results always preserve more privacy than their children. Since TDA adopts a binary-style of splitting, the partitioning of equivalent classes under a lower privacy-level will never hinder the construction of equivalent classes under a higher privacy-level. However, the phase II of TDA lacks flexibility: any partition on a sub-table will not

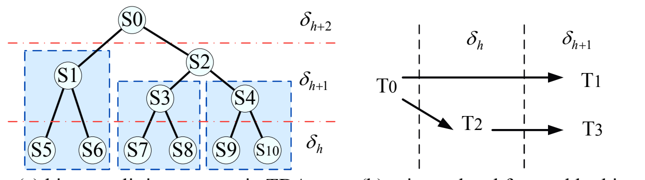

Figure 3: The idea of FLA. $\delta_{h+2} < \delta_{h+1} < \delta_h$.

change the sensitive values' distributions, which potentially cause utility loss.

FLA combines both TDA and BUA together. When creating a table T_i with the privacy-level δ_h, FLA first adopts TDA Phase I to create server master equivalent classes, such that each master class meets the current privacy level. For example, in Fig. 3 (a), data sets S_5 to S_{10} represent these classes. Next, within each master class (i.e. S_1, S_3, and S_4 in Fig. 3 (a)), FLA compares the utility loss by using BUA or TDA Phase II. The final equivalent classes come from the partitions with maximal utility.

4. SECURITY ANALYSIS

We consider an honest but curious attacking model: attackers always follow the rule for providing correct information to the next participant on a social chain, but they want to learn more information than are supposed to. In practice, the information propagation paths form a graph, instead of a chain; the information that an attack is able to obtain only comes from his direct neighbors. A system is privacy-preserving if, for any participant, the overall information that he could learn from all of his neighbors is no greater than the information that he directly obtains from the neighbor, who has the closest distance to the source node v_0. A scheme is called anti-colluding if the total information gain from the tables of the colluding users is not greater than cahoot's maximum information gain. Let $\|u - v\|$ be the social distance between u and v, which is counted as the number of hops along the shortest path.

THEOREM 2. *If all participants adopt the same δ and attribute weights $\{W\}$, for any pair of participants v_i and v_j with $\|v_i - v_o\| \le \|v_j - v_o\|$, then T_j must be a sanitization result of T_i by using FLA.*

PROOF. In FLA, along the information propagation path, each participant takes the output from his previous participant as its input during sanitization operations. Since sanitization is irreversible, if $\|v_i - v_o\| < \|v_j - v_o\|$, then T_j must be a sanitization result of T_i. Moreover, each step in FLA is deterministic: for the same input table T and privacy requirement δ, the output of FLA is always the same (Note that, even if two tuples are identical, FLA uses the row numbers to provide a unique operation order to them.) Now, consider two shortest information propagation paths from v_o, assuming $\{v_1, v_2, \ldots, v_l\}$ and $\{v'_1, v'_2, \ldots, v'_l\}$. Since all participants use the same δ and the original data owner v_0 gives the same table to the first receiver on each chain, the outputs of v_1 and v'_1 are the same. By induction, we know that whenever $\|v_i - v_o\| = \|v'_i - v_o\|$, the sanitization results on v_i and v'_i will be the same. Therefore, the result of FLA only relates with the shortest hop numbers from its executer to v_0. $\square$

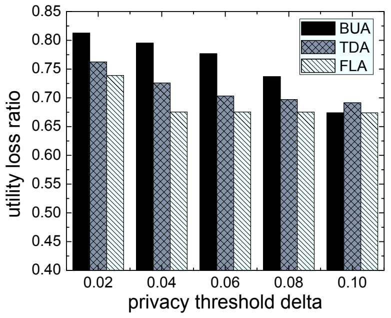

Figure 4: Each table's utility

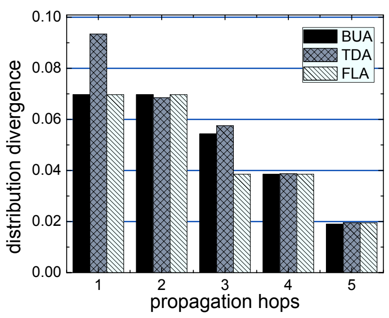

Figure 5: Each table's privacy

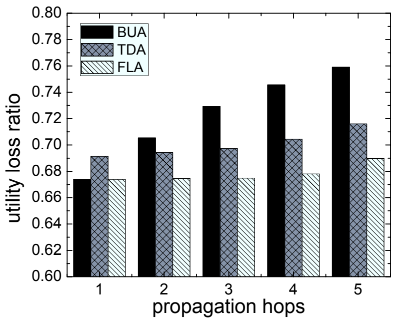

Figure 6: Overall tables' utility

THEOREM 3. *For a given initial table T_0 at v_0, FLA is naturally against colluding attacks, when all participants adopt the same δ and attribute weights $\{W\}$.*

PROOF. For a given threshold δ and propagation length l, the amount of disclosed information is only related to the recipient's distance toward v_0. According to theorem 2, for a group of cahoots, the maximum information that they could obtain must come from the member with the least distance toward the source v_0. As a result, FLA is against colluding attacks. $\square$

5. EVALUATION

In the simulation, we use the German Credit data set (german.data-numeric). We use tuples' weighted Euclidean distances as their merging costs:

$$d(\sigma_i(T), \sigma_j(T)) = \sqrt{\sum_{k=1}^{n-1} w_k \times (\sigma_i(\pi_k(T)) - \sigma_j(\pi_k(T)))^2}$$

where w_i is the attribute weight. In practice, this distance could be computed as a hamming code distance or the number of modified digits by suppression. After creating an equivalent class, the values of its members' quasi-identifiers are replaced with their mean value within the class. Instead of directly computing the remaining utility of each table, we count the percentage of data utility that has been wasted after sanitization. Let $\sigma_i(T')$ be the members from the same equivalent class after a sanitization operation, and $\sigma_i(T)$ be their corresponding original data, then the utility loss UL is defined as following: $UL(\sigma_i(T), \sigma_i(T')) = 1 - \frac{d(\sigma_i(T'), \sigma_i(\widetilde{T}))}{d(\sigma_i(T), \sigma_i(\widetilde{T}))}$ where $\widetilde{T}$ stands for the resulting table by sanitizing the initial table T_0 into one equivalent class, $\widetilde{T} = \pi(T_0)$.

Fig. 4 gives the pattern of utility loss with an increase in privacy-disclosure threshold δ. From the figure, we can see that the amount of utility loss dramatically increases at the higher privacy-preserving threshold (a smaller δ). As we expected, the tables created by BUA loses the most data utility, since its tuple-merging scheme (at an earlier phase) may cause inappropriate overall clustering results. We can also find that, at the condition of lower privacy-preserving thresholds, there are no significant differences between the TDA and FLA. The main reason for this phenomenon is that the members of each class created by these two approaches are basically similar to each other at the lower thresholds.

During the table propagation, although all result tables satisfy the same set of privacy requirements, the exact privacy-preserving values are different. Fig. 5 shows the changing pattern of each tables' privacy values. The horizontal lines show the increasing privacy requirements. The y-axis represents the privacy-level of each created table along a propagation path, and we measure the privacy-level by function $P(T)$ from section II. In most cases, BUA preserves more privacy than TDA. Since FLA considers both the current and future privacy requirements, at hop 3 in Fig. 5, it directly uses the privacy requirement of the fourth hop.

Fig. 6 shows system's overall utility loss. The x-axis indicates the total number of propagated hops, and the y-axis gives the average utility loss per table. We can see that the average amount of utility loss increases with the growing length of propagations. FLA has the smallest growing speed, while BUA has the largest one. In general, FLA preserves more data utilities than the other two approaches.

6. CONCLUSION

In this paper, we propose a new research problem: in a distributed system, given a source node and an information propagation path consisting of l participants, how can they sequentially generate l tables, such that the tables' overall utility is maximized and data privacy is gradually enhanced along the propagation path. To solve the problem, we first design two local algorithms for sanitizing tables with a single privacy-requirement, and then, we combine these two algorithms together and use a privacy requirement forward looking scheme. Extensive simulation results show that our approach can successfully increase the overall data utility, and meet the enhancing privacy-preserving requirements.

7. REFERENCES

[1] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *IEEE ICDE*, 2006.

[2] N. Li, T. Li, and S. Venkatasubramanian. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *ICDE*, 2007.

[3] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2007.

[4] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2002.

[5] X. Xiao, K. Yi, and Y. Tao. The hardness and approximation algorithms for l-diversity. In *ACM EDBT*, 2010.