$BSGI$: An Effective Algorithm towards Stronger l-Diversity

Yang Ye[1], Qiao Deng[2], Chi Wang[3], Dapeng Lv[3], Yu Liu[3], and Jianhua Feng[3]

[1] Institute for Theoretical Computer Science, Tsinghua University
Beijing, 100084, China
yey05@mails.tsinghua.edu.cn
[2] Department of Mathematical Science, Tsinghua University
Beijing, 100084, China
dengxinqiao@163.com
[3] Department of Computer Science, Tsinghua University
Beijing, 100084, China
{wangchi05,lvdp05,liuyu-05}@mails.tsinghua.edu.cn
fengjh@tsinghua.edu.cn

Abstract. To reduce the risk of privacy disclosure during personal data publishing, the approach of anonymization is widely employed. On this topic, current studies mainly focus on two directions: (1)developing privacy preserving models which satisfy certain constraints, such as k-anonymity, l-diversity, etc.; (2)designing algorithms for certain privacy preserving model to achieve better privacy protection as well as less information loss. This paper generally belongs to the second class. We introduce an effective algorithm "$BSGI$" for the widely accepted privacy preserving model: l-diversity. In the meantime, we propose a novel interpretation of l-diversity: Unique Distinct l-diversity, which can be properly achieved by $BSGI$. We substantiate it's a stronger l-diversity model than other interpretations. Related to the algorithm, we conduct the first research on the optimal assignment of parameter l according to certain dataset. Extensive experimental evaluation shows that Unique Distinct l-diversity provides much better protection than conventional l-diversity models, and $BSGI$ greatly outperforms the state of the art in terms of both efficiency and data quality.

Keywords: Privacy preservation, BSGI, k-anonymity, l-diversity, Unique-Distinct l-diversity.

1 Introduction

With the development of internet, more and more data on individuals are being collected and published for scientific and business uses. To reduce the risk of privacy disclosure during such publishing, the approach of anonymization is widely used. Removing the attributes that explicitly identify an individual, (e.g., name, social security number) from the released data table is necessary but insufficient, because a set of Quasi-identifying (QI) attributes (e.g., date of birth, zip code, gender) can be linked with public available datasets to reveal personal identity. To counter such "link attack", *P. Samaritan* and *L. Sweeney* proposed the model of k-anonymity[1,2,3,4]. K-anonymity requires each

S.S. Bhowmick, J. Küng, and R. Wagner (Eds.): DEXA 2008, LNCS 5181, pp. 19–32, 2008.

tuple in the published table to be indistinguishable from at least $k - 1$ other tuples on QI values. Tuples with the same *QI* values form an *equivalence class*. Thereby *k*-anonymity reduces the *identity disclosure* risk to no more than $1/k$.

However, since k-anonymity does not take into account the sensitive attribute (SA), namely, the attribute containing privacy information(e.g., disease, salary), it may be vulnerable to *sensitive attribute disclosure*[5]. [5] presents two kinds of possible attacks that *k*-anonymity cannot prevent: *homogenous attack* and *background knowledge attack*, then proposes a new model: *l*-diversity to counter such attacks. *l*-diversity ensures each equivalence class contains at least *l* "*well-represented*" SA values, thereby reduces the risk of sensitive attribute disclosure to no more than 1/*l*.

Current algorithms for l-diversity are generally derived from algorithms for k - anonymity. As proved in [5], any algorithm for *k*-anonymity, like *hierarchy-base* algorithm *Incognito*[13] and *partition-based* algorithm *Mondrian*[14], can be transformed easily to algorithm for l-diversity, just by changing the condition in each checking phase from k-anonymity to l-diversity. However, since k-anonymity algorithms do not take into account the distribution of SA values at all, which is the essence of l-diversity, the derived l-diversity algorithms may generate great and unnecessary information loss. In fact, our experiments in Section 6 reveal that *Incognito* for l-diversity is almost impractical for low efficiency and data quality while *Mondrian* for l-diversity drops behind our algorithm largely in both terms.

In [8], a new model, "$Anatomy$" was proposed for privacy preserving. Although $Anatomy$ fails to prevent identity disclosure because of no generalization on *QI* attributes, its ideas inspire us to propose an algorithm specially designed for l-diversity: $BSGI$. Since the implementation of l-diversity largely relies on the distribution of SA values, an intuitive but most effective inspiration is to firstly "bucketize" the tuples according to their SA values, then recursively "select" l tuples from l distinct buckets and "group" them into an equivalence class. As for the residual tuples, "incorporate" each of them into a proper equivalence class. The resulted table will satisfy l-diversity perfectly.

For instance, for the disease information table: Table 1, to satisfy 2-diversity, firstly, tuples are bucketized according to the "Disease" attribute and three buckets are formed: $B_1 = \{t_1, t_4\}$, $B_2 = \{t_3, t_5\}$ and $B_3 = \{t_2, t_6, t_7\}$. Here t_i denotes the i^{th} tuple in the table. Secondly, t_1 and t_2 are selected from B_1 and B_3 and grouped. An group(equivalence class) is formed as shown in Table 2.

Continuously, t_3 and t_4, t_5 and t_6 are selected and grouped (Table 3). Finally, the residual tuple t_7 is incorporated into Group 2, the final published table is created (Table 4).

Detailed discussions about the implementation of the four steps form the mainbody of this paper, together with two natural by-products: the optimal assignment of the parameter l and the stronger l-diversity model: Unique Distinct l-diversity.

The idea of Unique Distinct l-diversity comes from the property of the transformed tables achieved by $BSGI$: without considering the incorporated tuples, each equivalence class contains exactly l distinct SA values, we call such model "Unique Distinct l-diversity" and will further discuss it in this paper.

The rest of this paper is organized as follows. Section 2 gives the basic notations and definitions, including the Unique Distinct l-diversity model. Section 3 and 4 provide the

Table 1. The Original Table

NO.	Name	Gender	Postcode	Age	Disease
1	Alice	F	10075	50	Cancer
2	Bob	M	10075	50	Obesity
3	Carl	M	10076	30	Flu
4	Diana	F	10075	40	Cancer
5	Ella	F	10077	20	Flu
6	Fiona	F	10077	25	Obesity
7	Gavin	M	10076	25	Obesity

Table 2. The First Equivalence Class

Group id.	Gender	Postcode	Age	Disease
1	*	10075	50	Cancer
1	*	10075	50	Obesity

Table 3. The Table after $Bucktizing$, $Sel-ecting$ and $Grouping$

Group id	Gender	Postcode	Age	Disease
1	*	10075	50	Cancer
1	*	10075	50	Obesity
2	*	1007*	30-40	Flu
2	*	1007*	30-40	Cancer
3	F	10077	20-25	Flu
3	F	10077	20-25	Obesity

Table 4. The Final Published Table

Group id	Gender	Postcode	Age	Disease
1	*	10075	50	Cancer
1	*	10075	50	Obesity
2	*	1007*	25-40	Flu
2	*	1007*	25-40	Cancer
2	*	1007*	25-40	Obesity
3	F	10077	20-25	Flu
3	F	10077	20-25	Obesity

essential ideas of $BSGI$ algorithm, together with the discussion about l's assignment. Section 5 formally presents the $BSGI$ algorithm with further discussions. Section 6 provides the experimental evaluations. Section 7 introduces related work and Section 8 concludes this paper with discussions about future work.

2 Preliminary

2.1 Basic Notations

Let $T = \{t_1, t_2, \cdots, t_n\}$ be the table that need to be anonymized. Here t_i,$i = 1, 2, \cdots,$ n represents the i^{th} tuple of the table. Each tuple contains a set of Quasi-identifying attribute $\{A_1, A_2, \ldots, A_N\}$. Each tuple contains one sensitive attribute S(we will discuss the *single-tuple-multi-SA* case in Section 5).We use $t[A]$ to denote the value of t's attribute A. Let $T^* = \{t_1^*, t_2^*, \cdots, t_n^*\}$ be the anonymized table, where t_i^* is the i^{th} tuple after anonymization. Also $T^* = e_1 \bigcup e_2 \bigcup \cdots \bigcup e_m$, where e_i is the i^{th} equivalence class. Let E be the set of equivalence classes. By overriding, we also use $e_i[A_j]$, etc. And $e_i[S]$ denotes the multi-set of e_i's SA values.

2.2 The Information Loss Metric

In fact, our *BSGI* algorithm does not rely on a certain information loss metric. Any metric that captures the quality of generalization[12,15,18] can be adopt by the algorithm. In our experiment, we use the metric proposed by [12], denoted as IL metric.

IL metric defines the information loss for categorical and numerical attributes separately. The information loss of a tuple is defined by summing up the loss of all attributes(multiplied by different weights). The total information loss of the whole table is defined by summing up the loss of all tuples.

2.3 *l*-diversity and Unique Distinct *l*-diversity

Definition 1. (*The l-diversity Principle*) *An anonymized table is said to satisfy* l*-diversity principle if for each of its equivalence class* e, $e[S]$ *contains at least* l *"well-represented" values[5].*

According to [5,6], the so called "well-represented" has several interpretations:

1. *Distinct l-diversity*. This interpretation just requires that for each equivalence class e_i, there are at least *l* distinct values in $e_i[S]$.

2. *Entropy l-diversity*. The entropy of equivalence class *e* is defined as follows:

$$Entropy(e) = - \sum_{\textit{each distinct } s \in e[S]} P(e,s) \log(P(e,s))$$

   Here $P(e,s)$ denotes the proportion that value *s* takes in $e[S]$. Entropy *l*-diversity requires for each equivalence class e_i, *Entropy*$(e_i) \geq \log l$.

3. *Recursive (c,l)-diversity*. Let d be the number of distinct SA values in $e[S]$. r_i, $1 \leq r \leq d$, be the number of the i^{th} most frequent SA value in $e[S]$. Recursive (*c*,*l*)-diversity requires $r_1 < c(r_l + r_{l+1} + \cdots + r_d)$.

Here we propose our interpretation of l-diversity:

Definition 2. (*Unique Distinct l-diversity*) *An anonymized table is said to satisfy Unique Distinct* l*-diversity if for each of its equivalence class* e, $e[S]$ *contains exactly* l *distinct* SA *values.*

Observation 1. *If equivalence class* e *satisfies Unique Distinct l-diversity, then it also satisfies Distinct* l*-diversity, Entropy* l*-diversity and Recursive* (c,l)*-diversity for all constant* $c > 1$.

The proof is simple, we need only to check the demand of the three models one by one. According to this observation, Unique Distinct *l*-diversity is a stronger model. □

Observation 2. *Unique Distinct l-diversity prevents "probability inference attack"*[1] *better than other three models.*

This is also apparent, in Unique Distinct *l*-diversity, the SA attributes are uniformly distributed. Therefore, when the attacker locates some individual in a certain equivalence class e, without further background knowledge[5], he cannot disclose the individual's SA value with probability higher than 1/*l*. However, in the other three models,

[1] Or "skewness attack"[6], the privacy disclosure because of non-uniform distribution of SA values within a group.

there may be cases when one SA value appears many more times than other SA value in $e[S]$. Then the attacker could guess the individual has such SA value with high probability. □

The foregoing observations substantiate the advantages of Unique Distinct l-diversity. We shall prove its feasibility in Section 4.

3 The Implementation of the $Selecting$ Step

In $BSGI$, the tuples are first bucketized according to their SA values. Let B_i denote the i^{th} greatest bucket and $B = \{B_1, B_2, \ldots, B_m\}$ denote the set of buckets. We have: $n_i = |B_i|,\ n_1 \geq n_2 \geq \cdots \geq n_m,\ \Sigma_{i=1}^{m} n_i = n$. Since different n_i's may vary greatly, we shall use the following "$Max - l$" method to ensure the formed "l-tuple groups" are as many as possible: in each iteration of selecting, one tuple is removed from each of the l largest buckets to form a new group. Note that after one iteration, the size of some buckets will be changed. So in the beginning of every iteration, the buckets are sorted according to their sizes, as shown in Figure 2.

Theorem 1. *The* Max-l *method creates as many groups as possible.*

Proof. We prove by induction on $m = |B|$ and $n = |T|$.
Basis. $m = n = l$. This is the basis because when $m < l$ or $n < l$, no group can be created. In this case, there is exactly one tuple in each bucket, apparently, the *Max-l* method creates as many groups as possible.

Induction. When $m > l,\ n > l$. Assume the way W creates maximal number of groups, which equals k. We denote $G_i = \{i_1, i_2, \ldots, i_l\}$ $(i_1 < i_2 < \cdots < i_l)$ to be the i^{th} group created by W and G_i contains one tuple from each of $B_{i_1}, B_{i_2}, \ldots, B_{i_l}$. From W, a new way $W^{'}$ can be constructed that satisfies: (1)$W^{'}$ creates k groups; (2)The first group created by $W^{'}$ is $G_i^{'} = \{1, 2, \ldots, l\}$. The construction takes two operations: *swap* and *alter*.

1. **swap.** $((i,a),(j,b))$ $(1 \leq i,\ j \leq k,\ 1 \leq a,\ b \leq m,\ a \in G_i,\ a \notin G_j,\ b \in G_j,\ b \notin G_i)$ means to exchange a in G_i with b in G_j. For example, $G_1 = \{1,2\},\ G_2 = \{3,4\}$, $swap((1,1),(2,3))$ leads to $G_1 = \{2,3\},\ G_2 = \{1,4\}$. Since $a \notin G_j,\ b \notin G_i$, the grouping way after this operation is always valid.
2. **alter.** (a,b) $(1 \leq a < b \leq m)$ means to replace each a in every G_i with b and replace each b with a. For the above example, $alter(2,3)$ leads to $G_1 = \{1,3\},\ G_2 = \{2,4\}$. The grouping way is valid after this operation if and only if a's total appearing times is no more than b's.

The construction is like this: for variable i from 1 to l, assume the i^{th} element in G_1 is b. If $i = b$, we do nothing. Otherwise, b must be greater than i. We check for other $k-1$ groups $G_2, \ldots, G_k$. There are two possible cases:

1. There is a group G_j such that $i \in G_j$ and $b \notin G_j$. In this case, we perform $swap((1,b),(j,i))$ to obtain a new grouping way. Since $i \notin G_1, b \notin G_j$, it is still a valid grouping way.

2. Every group that contains i also contains b. Therefore, the total number of i's is no more than that of b's. In this case, we perform $alter(i, b)$, the grouping way is still valid after this operation.

Note operation on i ensures the i^{th} element in G_1 to be i and does not change the first $i-1$ elements. So when the whole process finishes, we obtain a valid grouping way $W^{'}$ with $G_1^{'} = \{1, 2, \ldots, l\}$. Removing tuples responding to the elements in $G_1^{'}$, we obtain a new instance of the problem with $m' \le m, n' = n - l < n$. Due to induction hypopiesis, we know our algorithm generates as many groups as possible for the new instance. In the meantime, the best solution to the new instance contains at lest $k-1$ groups, because $G_2^{'}, G_3^{'}, \ldots, G_k^{'}$ is such a grouping way. So for the original instance, our algorithm generates at least k groups. That is the maximal number as assumed. The proof is completed. □

During selecting, in order to reduce information loss and avoid exhaustively searching the solution space, the following greedy method is adopted: in each iteration of selecting, a random tuple t_1 is selected from B_1 and it forms the original equivalence class(group) e. For variable i from 2 to l, from B_i, a tuple t_i that minimize $IL(e \bigcup t_i)$ is selected and merged into e, as shown in Figure 2.

4 The Property of Residual Tuples after *Selecting* and *Grouping*

In this section, well shall investigate the property of residual tuples after selecting and grouping steps.

Theorem 2. *When the selecting and grouping steps terminate, there will be no residual tuples if and only if the buckets formed after the "bucketizing" step satisfy the following properties (we call it l-Property):*
(1) $\frac{n_i}{n} \le \frac{1}{l}$, $i = 1, 2, \ldots, m$*(Use the same notation:* n_i, m, n, *as in Section* 3*)*
(2) $n = kl$ *for some integer* k

Proof. First notice that $\frac{n_i}{n} \le \frac{1}{l}$ is equivalent with $n_1 \le k$, because n_1 is the largest among all n_i's.

(*If*) We prove by induction on $m = |B|$ and $n = |T|$.
Basis. $m = n = l$, this is the basis because m cannot be smaller than l. Now there's one tuple in each bucket. Obviously the algorithm leaves none.

Induction. $m > l$ or $n > l$. Resembling the proof of Theorem 1, we assume that when the first group is created by our algorithm, the remaining buckets and tuples form a new instance of the problem with parameter (m', n'). We shall prove this new instance also has l-Property.

Apparently $m' \le m, n' = n - l = (k-1)l$. To prove $\frac{n_1'}{n'} \le \frac{1}{l}$. We discuss two cases for different values of n_1.

1. $n_1 = k$. Assume that $n_1 = n_2 = \cdots = n_j = k,\ n_{j+1} < k$. We have:

$$n = kl = \Sigma_{i=1}^{m} n_i = \Sigma_{i=1}^{j} n_i + \Sigma_{i=j+1}^{m} n_i \ge kj$$

So $l \geq j$. This means the number of the buckets with k tuples does not exceed l. According to our algorithm, after the first group is removed, the bucket with most tuples has size $k-1$ because all the buckets previously has size k contribute one to that group. That is $n'_1 = k - 1 = \frac{n'}{l}$, or $\frac{n'_1}{n'} \leq \frac{1}{l}$.

2. $n_1 \leq k - 1$. This case is simple because $n'_1 \leq n_1 \leq k - 1$, so $\frac{n'_1}{n'} \leq \frac{1}{l}$.

In both cases, we obtain that the new instance has l-Property. With the very same idea as used in the proof of Theorem 1, the outcome of the remaining execution of the algorithm equals to what we obtain by running the algorithm individually for the new instance. Due to induction hypopiesis, we know our algorithm will leave no non-empty buckets. So for the original instance, the conclusion also holds. The proof of if-part is completed.

(*Only-if*) It is easy to verify that n must be multiple of l to guarantee that all the tuples can be grouped. So there exists some integer k such that $n = kl$

Since there's no residual tuples, for the requirement of l-diversity, each group contains at most one tuple from the first bucket. The mapping from the tuples in B_1 to the groups is $one - to - one$, but not necessarily $onto$. Therefore, we have $n_1 \leq k = \frac{n}{l}$, or $\frac{n_1}{n} \leq \frac{1}{l}$.The proof of only-if part is completed. □

When the buckets satisfy the first condition while do not satisfy the second condition of l-Property, we have following conclusion:

Corollary 1. *If the buckets satisfy following Property:* $\frac{n_i}{n} \leq \frac{1}{l}$*, then after the selecting and grouping steps, each non-empty bucket has only one tuple.*

Proof. Assume $n = kl + r, 0 \leq r < l$, hypothetically change our algorithm like this: first subtract one tuple from each of $B_1, B_2, \ldots B_r$, then operate the "$Max - l$" selecting method in Section 3. The new instance satisfies l-Property and k groups will be formed. Therefore the best solution creates no less than k groups. In the meantime it creates no more than k groups because $n = kl + r$.

Now we already know there are k iterations of "selecting and grouping" in total[2], denote them to be $I_1, I_2 \ldots I_k$. Assume one bucket(denoted B_{bad}) contains at least 2 tuples after I_k. Note before I_k, there are at most $l - 1$ buckets with size at least 2, otherwise there will be at least l non-empty buckets after I_k. So a tuple from B_{bad} is selected during I_k and $|B_{bad}| \geq 3$ before I_k. Similarly, before I_{k-1}, there are at most $l - 1$ buckets with size at most 3. So a tuple from B_{bad} is selected during I_{k-1} and $|B_{bad}| \geq 4$ before I_{k-1}. Recursively, we obtain $|B_{bad}| \geq k + 2$ before I_1, this contradicts the condition. The proof is completed. □

The above result is of great merits. On one side, the number of residual tuples is limited and bounded by l, our algorithm will not suffer from large number of residual tuples. Thus the feasibility of Unique Distinct l-diversity can be assured. As proved in Section 2, Unique Distinct l-diversity is a stronger l-diversity model which provides better privacy preservation. The experiment in Section 6 will also substantiate this. In sum, we have:

[2] Similar theorem is proved in [8], however, we find that proof ungrounded because it assumes the number of iteration equals k, without proof.

Corollary 2. *Unique Distinct l-diversity can be exactly achieved if the original table satisfy both l-Property (1) and (2). If the table just satisfy l-Property (1), Unique Distinct l-diversity can be achieved with less than l residual tuples.*

On the other side, we can choose a proper l according to the distribution of SA values. Consider, assigning a large number to l provides better privacy preservation but greater information loss, while a small number leads to less data distortion but higher privacy disclosure risk. Current studies ignore to investigate the optimal assignment of l to balance such trade-off. However, from previous discussion we can reach the following conclusion:

Corollary 3. *The optimal assignment to parameter l in l-diversity is max$\{2, \lfloor \frac{n}{n_1} \rfloor\}$.*

If $\lfloor \frac{n}{n_1} \rfloor = 1$, this reflects the most frequent SA value takes a proportion more than 50%. This is a greatly "skew" distribution and the privacy disclosure risk cannot be reduced to below $1/2$.

As for the residual tuples, the simplest way is to $suppress$ them. Here we perform $incorporating$: for each of them, find a proper equivalence class to incorporate it. The so called "proper" has two requirements: (1)The chosen equivalence class had better not contain the new SA value, thus it will satisfy Unique Distinct ($l+1$)-diversity after incorporation. (2)The incorporation leads to minimal information loss. The detailed implementation is in Figure 3.

5 The BSGI Algorithm

5.1 The Algorithm

Summing up the previous discussions, we formally present the *BSGI* algorithm in this section.

The "$Select$" procedure in Figure 2 implements the "*Max-l*" selecting method in Section 3 and the "$Incorporate$" procedure implements the incorporating method in Section 4. Say, if there exists some equivalence class e that $t[S] \notin e[S]$, t is incorporated into one of such classes that minimize the information loss. Otherwise, for each e, $t[S] \in e[S]$, the choosing of e to incorporate t is only based on minimal information loss.

5.2 Further Discussion about the Algorithm

In this section, we shall discuss some special cases with regard to $BSGI$.

1. The *single-Individual-Multi-Class* Case
Note our algorithm can be categorized into "local-recoding"[13] that the created equivalence classes may overlap each other. Thus one individual may be associated with more than one equivalence classes. For instance, in Table 4, the individual $George$ can be associated with both Group 2 and Group 3. With regard to its influence on privacy disclosure risk, we shall prove:

Input: Original table T
Output: Anonymized table T^* which satisfies l-diversity
Data: $E = \emptyset$, E is the set of equivalence classes
begin
  /* *The bucketizing step* */
  Bucketize tuples of T according to their SA values;
  $B = \{B_i\}$ /* B is the set of buckets */
  /* *The selecting and grouping steps* */
  while $|B| \geq l$ **do**
    $E = E \bigcup$ ***Select***$()$;
  /* *The incorporating step* */
  foreach *residual tuple* t **do**
    Incorporate(t);
  return T^*;
end

Fig. 1. The *BSGI* Algorithm

Data: B =the set of buckets; $e = \emptyset$, the equivalence class to be created
begin
  Sort buckets in B according to their size;
  $B = \{B_1, B_2, \cdots, B_m\}$ where B_i is the i^{th} greatest bucket in B;
  Randomly remove one tuple t_1 from B_1;
  $e = \{t_1\}$;
  for $i \leftarrow 2$ **to** l **do**
    Remove one tuple t_i from B_i that minimize $IL(e \bigcup t_i)$;
    $e = e \bigcup t_i$;
  return e;
end

Fig. 2. The *Select* Procedure

Data: E =the set of equivalence classes; t =the tuple to be incorporated
begin
  $E^{'} = \{e | e \in E$ and $t[S] \notin e[S]\}$;
  if $|E^{'}| \neq 0$ **then**
    Find e in $E^{'}$ that minimize $IL(e \bigcup t)$;
  else
    Find e in E that minimize $IL(e \bigcup t)$;
  $e = e \bigcup t$;
end

Fig. 3. The *Incorporate* Procedure

Theorem 3. *The case of single-individual-multi-class does not increase sensitive attribute disclosure risk to more than 1/l.*

Proof. Assume one individual I, with SA value $I[s]$, can be associated with equivalence classes $e_{i_1}, e_{i_2}, \ldots, e_{i_j}$. According to probability's Bayes Model, the risk of sensitive attribute disclosure is

$$\sum_{k=1}^{j} Pr(I \in e_{i_k}) \cdot Pr(privacy\ disclosure|I \in e_{i_k})$$

Consider

$$\forall k,\ Pr(privacy\ disclosure|I \in e_{i_k}) \leq 1/l$$

and

$$\sum_{k=1}^{j} Pr(I \in e_{i_k}) = 1$$

We have, the total risk of sensitive attribute disclosure:

$$Pr(privacy\ disclosure) \leq 1/l$$ □

2. The *Single-Individual-Multi-Tuple* Case
Traditionally, we assume one single individual corresponds to a single tuple in the table. However, there are cases where one single individual corresponds to multiple tuples. (e.g., one person's multiple disease records for different diseases). In this case, if multiple tuples of a same individual is grouped together, the proportion of tuples containing the individual's SA values within that group will be larger than $1/l$, thus leads to higher privacy disclosure risk.

To counter such case, we need only to add a "*check*" procedure during the selecting step. If a candidate tuple belongs to a already-selected individual, that tuple will not be selected.

3. The *Single-Tuple-Multi-SA* Case
Traditionally, we deal with the case where a single tuple contains only one sensitive attribute. For the *single-tuple-multi-SA* case, an intuitive thinking is to consider the SA value as one multi-dimensional vector. However, this may lead to privacy disclosure. Consider the case of two sensitive attributes: ($Disease$, $Salary$). The values (flu, \$10000), ($cancer$, \$10000), ($obesity$, \$10000) do not equal to each other. But if tuples with these SA values are grouped, the disclosure risk for attribute $Salary$ is 100%.

To counter such case, in the *selecting* step, the new tuple should be unequal to each of the already-selected tuples on all sensitive attributes. However, this is quite a preliminary approach, it's performance deserves extensive study.

6 Experiments

In this section, we conducted several experiments using the real world database $Adult$, from the UCI Machine Learning Repository[20] to verify the performance of $BSGI$ in both efficiency and data quality by comparing with full-domain generalization algorithm "$Incognito$" and multi-dimensional partition algorithm "$Mondrian$" respectively.

6.1 Experimental Data and Setup

Adults database is comprised of data from the US Census. There are 6 numerical attributes and 8 categorical attributes in Adult. It leaves 30162 records after removing the records with missing value. In our experiments, we retain only eight attributes. {*Age, Final-Weight, Education, Hours per Week, Martial Status, Race, Gender*} are considered as Quasi-identifying attributes. The former four attributes are treated as numeric attributes while the latter three are treated as categorical attributes. *WorkClass* is the sensitive attribute. According to Corollary 1, the upper bound of l is determined to be 7 because the most frequent SA value *"Prof-specialty"* takes a proportion greater than $1/8$ while less than $1/7$.

We modify LeFevre's *Incognito*[13] and *Mondrian*[14] into the l-diversity versions. These two algorithms and our $BSGI$ are all built in Eclipse 3.1.2, JDK 5.0, and executed on a dual-processor Intel Pentium D 2.8 GHz machine with 1 GB physical memory. The operating system is Microsoft Windows Server 2003.

6.2 Efficiency

The running time of $Incognito$ is not in Figure 4 because such exhaustive algorithm takes nearly exponential time in the worst case. In our experiment, its execution time is more than half an hour, exceed the other two by several orders of magnitude. We execute both $BSGI$ and $Mondrian$ three times, and calculate the average. Figure 4 reports the average time of both algorithms. As is shown, the running time of $Mondrian$ decreases from about $90s$ to $75s$, because when l increases, the recursive depth of the algorithm reduces. However, as is shown, $BSGI$ performs much better than $Mondrian$ and almost does not increase with l. In fact, it is easy to conclude the time complexity of $BSGI$ is $O(n^2)$, highly efficient and independent of l.

6.3 Data Quality

Figure 6 depicts the widely adopted metric: Discernibility Metric cost(DM)[16] of the three algorithms and Table 5 shows the average group size resulted from them. These two metrics are mutually related, because without suppression, DM is defined as

$$DM = \sum_{\textit{each equivalence class } e} (|e|)^2$$

In both metrics, the cost of $Incognito$ exceeds the other two by orders of magnitude. Since $Incognito$ always maps all the QI values within the same level of its generalization hierarchy into a same generalized value, as a result, it tends to over-generalize the original table. In fact, over-generalization is the fatal shortcoming of the class of full-domain generalization algorithms. Secondly, $BSGI$ does a much better job than $Mondrian$. Actually, $BSGI$ always achieves the best result with regard to these two

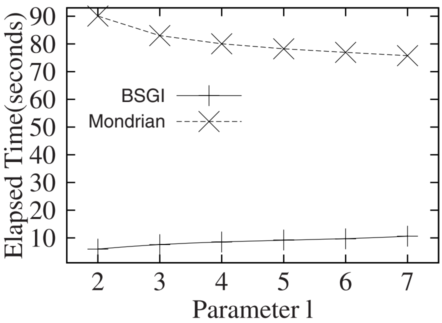

Fig. 4. Elapsed Time

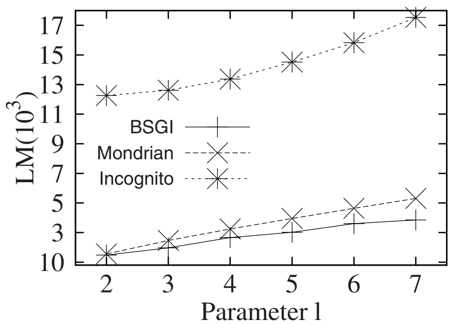

Fig. 5. Information Loss Metric

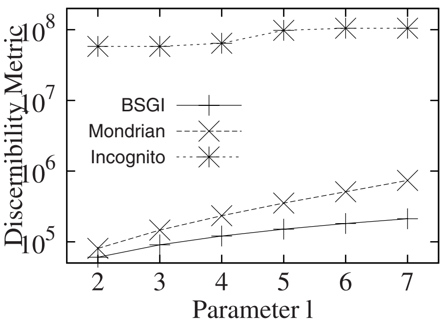

Fig. 6. Discernibility Metric

Table 5. Average Group Size

	Average Group Size		
l	BSGI	Mondrian	Incognito
2	2.00	2.47	471
3	3.00	4.32	471
4	4.00	6.71	628
5	5.00	9.81	942
6	6.00	13.79	1005
7	7.00	18.73	1005

metrics, because it implements the Unique Distinct l-diversity model and every equivalence class is of the minimal size l. We can learn that there are almost exactly l tuples in each equivalent class generated by $BSGI$.

Besides DM and average group size metrics, we adopt the IL metric in Section 2.2, which gives more information about how much the tuples are generalized. Figure 5 demonstrates the IL as a function of l. Again, $Incognito$ causes more loss by orders of magnitude. $BSGI$ is the best but the advantage seems not so significant in comparison with $Mondrian$. When $l = 7$, the IL of $BSGI$ is 70% of $Mondrian$'s. This can be explained by the implementation of $selecting$ step: the new selected tuple that minimize IL is not from the whole table, but from an appointed bucket. As proved in Section 3, such selecting method ensures the maximum number of created groups, however may be unable to achieve minimal information loss. This cost is worthwhile, because Unique Distinct l-diversity largely enhances privacy preservation.

In sum, the excessively long execution time and high information loss render *Incognito* almost impractical. $BSGI$ achieves the optimal DM and $AverageSize$ metric. With regard to the IL metric, $BSGI$ still outperforms $Mondrian$ apparently. The $BSGI$ is an highly efficient algorithm with low information loss. In the meantime, it achieves the stronger Unique Distinct l-diversity model, which preserves privacy excellently.

7 Related Work

As introduced in the abstract, the work dealing with developing privacy models includes [5,6,7,8,9,10,11] and etc. [6] proposes the model of t-closeness, which requires the distribution of SA values in each equivalence class to be close to the entire table. [7] enable personal specified degree of privacy preserving. Instead of generalizing original QI values, [8] anatomize the original table into a quasi-identifier table (QIT) and a sensitive table(ST). [9] propose the model of δ-presence to the case of individual presence should be hidden. Unlike previous work on static datasets, [10,11] deal with privacy preserving for dynamic, or incremental datasets. The work on designing algorithms for privacy models includes [13,14,15,16,17] and etc. [13], [14] and [15] represent three main classes of algorithms: *hierarchy-based*, *partition-based* and *clustering-based*. In fact, our work can be categorized into *clustering-based* algorithms. There are still other related works. The information loss metric proposed by [12] is adopted by this paper. [19] investigates the large information loss that privacy preservation techniques encounter in high-dimension cases.

8 Conclusion and Future Work

In this paper, we propose a specially designed algorithm: $BSGI$ for l-diversity. Through such algorithm, a stronger l-diversity model, Unique Distinct l-diversity can be achieved with less information loss. We also investigate the optimal assignment to parameter l in the model.

For the future work, although we have dealt with the *single-tuple-multi-SA* case, further analysis on the influence of multiple sensitive attributes and designing specific algorithm are of great merits. In the meantime, it may be worthwhile to extend $BSGI$ to work on dynamic growing datasets.

Acknowledgments. This work was Supported in part by the National Natural Science Foundation of China Grant 60553001, 60573094, the National Basic Research Program of China Grant 2007CB807900, 2007CB807901, the National High Technology Development 863 Program of China under Grant No.2007AA01Z152 and 2006AA01A101, the National Grand Fundamental Research 973 Program of China under Grant No. 2006CB303103, and Basic Research Foundation of Tsinghua National Laboratory for Information Science and Technology (TNList).

References

1. Samarati, P.: Protecting respondents identities in microdata release. TKDE 13(6), 1010–1027 (2001)
2. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10(5), 571–588 (2002)
3. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information. In: PODS, p. 188 (1998)

4. Sweeney, L.: k-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness, and Knowlege-Based Systems 10(5), 557–570 (2002)
5. Machanavajjhala, A., Gehrke, J., Kifer, D.: l-diversity: Privacy beyond k-anonymity. In: ICDE, p. 24 (2006)
6. Li, N., Li, T.: t-closeness: Privacy beyond k-anonymity and l-diversity. In: ICDE, pp. 106–115 (2007)
7. Xiao, X., Tao, Y.: Personalized privacy preservation. In: SIGMOD, pp. 229–240 (2006)
8. Xiao, X., Tao, Y.: Anatomy: Simple and effective privacy preservation. In: VLDB, pp. 139–150 (2006)
9. Ercan Nergiz, M., Atzori, M., Clifton, C.W.: Hiding the Presence of Individuals from Shared Databases. In: SIGMOD, pp. 665–676 (2007)
10. Xiao, X., Tao, Y.: m-Invariance: Towards Privacy Preserving Re-publication of Dynamic Datasets. In: SIGMOD, pp. 689–700 (2007)
11. Byun, J.-W., Li, T., Bertino, E., Li, N., Sohn, Y.: Privacy-Preserving Incremental Data Dissemination. CERIAS Tech Report, Purdue University (2007-07)
12. Xu, J., Wang, W., Pei, J., Wang, X., Shi, B., Fu, A.: Utility-Based Anonymization Using Local Recoding. In: SIGKDD, pp. 785–790 (2006)
13. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Incognito: Efficient full-domain k-anonymity. In: SIGMOD, pp. 49–60 (2005)
14. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: ICDE, p. 25 (2006)
15. Byun, J.-W., Kamra, A., Bertino, E., Li, N.: Efficient k-Anonymization Using Clustering Techniques. In: Li Lee, M., Tan, K.-L., Wuwongse, V. (eds.) DASFAA 2006. LNCS, vol. 3882. Springer, Heidelberg (2006)
16. Bayardo, R., Agrawal, R.: Data privacy through optimal k-anonymization. In: ICDE, pp. 217–228 (2005)
17. Aggarwal, G., Feder, T., Kenthapadi, K., Motwani, R., Panigrahy, R., Thomas, D., Zhu, A.: Approximation algorithms for k-anonymity. In: JOPT (2005)
18. Iyengar, V.: Transforming data to satisfy privacy constraints. In: SIGKDD, pp. 279–288 (2002)
19. Aggarwal, C.C.: On k-anonymity and the curse of dimensionality. In: VLDB, pp. 901–909 (2005)
20. U.C. Irvin Machine Learning Repository, `http://archive.ics.uci.edu/ml/`