Journal of Theoretical and Applied Information Technology
20th April 2015. Vol.74 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SCALABLE MULTIDIMENSIONAL ANONYMIZATION ALGORITHM OVER BIG DATA USING MAP REDUCE ON PUBLIC CLOUD

[1]AMALRAJ IRUDAYASAMY, [2] DR. AROCKIAM L,

[1]Research Scholar, Periyar University, Salem, Tamil Nadu, India

[2]Associate Professor in Computer Science, St. Joseph's College, Trichy, TamilNadu, India

E-mail: [1]amalprisci@gmail.com, [2]larokiam@yahoo.co.in

ABSTRACT

It appears that everybody observes with special attention, the occurrence of big data and its practice. There is no disbelief that the big data uprising has instigated. Though the practices of big data propose favorable business paybacks, there are substantial privacy implications. Multidimensional generalization anonymization scheme is an actual method for data privacy preservation. Top-Down Specialization (TDS) and Bottom-Up generalization (BUG) are two methods to attain multidimensional anonymization. However, prevailing methodologies for multidimensional generalization anonymization scheme disconcerts parallelization proficiency, thereby missing scalability while managing big data on cloud. TDS and BUG suffer from poor performance for certain value of k-anonymity parameter if they are utilized individually. In this paper, we recommend a hybrid method that combines TDS and BUG together for competent multidimensional anonymization over big data. Additionally, Map reduce based algorithms for two components (TDS and BUG) to increase high scalability cloud are designed. Experiment estimations determine that the hybrid method expressively progresses the scalability and proficiency of multidimensional generalization anonymization system over prevailing methods.

Keywords: *Big data; cloud computing; data anonymization; privacy preservation; Map reduce*

1. INTRODUCTION

Cloud computing and big data, together exposes many types of public and personal information. From an enterprises outlook, it discloses the enterprise to great hazard. Though a company cannot be alleged responsible for personal data out in the public, could run afoul of legitimate and governing issues. Making materials worse, persons are disorganized between the private data and the ways to handle it. However, it is significant to recognize that rising privacy concerns about the usage of big data are not restricted to these predictable cloud providers. One of the most serious privacy traits is, basically, the quality or exactness of the data. Another eminence issue is the technique that Internet explores terms or expressions can be misunderstood, when this type of data are collected [1, 2].

The first step in real use of big data is to become highly proficient in acquiring and handling cloud services, which are considered a requirement for big data to be cost effective. There must be well-defined accountabilities for both the cloud supplier and consumers about specific data privacy controls that are obligatory. There must also be constant observing and examinations of cloud services along with any applicable metrics that specify levels of data integrity, confidentiality and availability. The next approach to enable improved use of big data is to instrument joined storage. Joined storage is more effective and will reduce the probability of faults that impact data quality or accurateness [3]. A serious characteristic of converged storage that shares to data eminence and correctness is data de-duplication, while it has cost efficiency welfares as well. Additional preeminent practice is to correctly cleanse data, as it helps elude a number of the aforementioned privacy issues.

Data anonymization, extensively studied and widely adopted [4], is an effective way for data privacy preservation. Data anonymization refers to hiding identity and/or sensitive data for owners of data records. Then, the privacy of an individual can be effectively preserved while certain aggregate

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

information is exposed to data users for diverse analysis and mining. Multidimensional anonymization scheme is widely adopted to anonymize data sets for privacy preservation, producing a better arrangement between data utility and distortion. There are two ways to accomplish multidimensional anonymization, i.e., Top-Down Specialization (TDS) and Bottom-Up Generalization (BUG). So far, a series of approaches have been proposed for TDS or BUG [5, 6, 7, 8]. However, data sets in big data applications on cloud have become so large, that it is a big challenge for existing multidimensional anonymization algorithms to anonymize such data sets in a scalable fashion, due to their lack of parallelization capability. When TDS and BUG methods are implemented separately, it offers poor performance for certain value of *k*-anonimity parameter [9]. Specifically, TDS is preferred when k is large while BUG is favorable when *k* is small.

Map-reduce [10], a large-scale data processing framework, have been integrated with cloud to provide powerful computation capability for applications, e.g. Amazon Elastic map-reduce (EMR) service [11]. Map-reduce technique is used on cloud to address the scalability problem in the proposed approach. As the map-reduce computation paradigm is relatively simple, it is still a challenge to design proper map-reduce jobs for TDS and BUG.

In this paper, a highly scalable fusion method is proposed which combines TDS and BUG composed for multidimensional anonymization over big data. The approach automatically determines the component that must be used to conduct the anonymization when a data set is given, by comparing the user- specified *k*-anonymity parameter with a threshold derived from the data set. Both components TDS and BUG are developed based on map-reduce to advance high scalability. Having designed map-reduce based TDS [12], only the map-reduce algorithmic design of BUG is presented herein. Experimental appraisal reveals that the hybrid method expressively advances the scalability and effectiveness of multidimensional data anonymization over prevailing methods. The main contributions of this paper are divided into three segments.

Firstly, a mix approach to improve the scalability and efficiency of multidimensional data anonymization via automatically choosing TDS or BUG are suggested. Secondly, a group of innovative map-reduce jobs are designed for BUG

to concretely conduct the computation in a highly scalable fashion. Lastly, experimental evaluations demonstrate that the hybrid approach improves the scalability and efficiency of multidimensional anonymization scheme over existing approaches.

The remainder sections of this paper are ordered as follows. The next section reviews related work, and analyzes the problems in existing multidimensional anonymization approaches. In section III, the preliminary concepts for this approach are discussed. Section IV elaborates algorithmic details of map-reduce jobs for BUG, and Section V formulates the hybrid approach. Empirically evaluation of the proposed is discussed in section VI. Finally, conclusion and the future work is summarized in section VII.

2. RELATED WORK AND PROBLEM ANALYSIS

2.1 Related Work

Privacy preservation on data has been extensively investigated and fruitful progress has been made by research communities [4]. It is briefly reviewed the related work as follows. A bulk of privacy models and anonymization approaches have been put forth to preserve the privacy sensitive information in data sets. *k*-anonymity [9] and *l*- diversity [13] are two basic and widely-adopted privacy models to measure the degree of privacy-sensitive information disclosure against record linkage attacks and attribute linkage attacks, respectively. Other privacy models like t-closeness [14] and m-invariance [15] are also proposed for various privacy attack scenarios. Numerous anonymizing processes are leveraged to anonymize data sets, comprising generalization [5, 16, 17], anatomization [18], slicing [19], disassociation [20], etc. In this paper, generalization technique is used which replaces some domain values with a parent value in the taxonomy tree. Roughly, there are few generalization schemes [4], namely, full domain [21], sub-tree [5], multidimensional, siblings [16] and cell generalization [17].

This research herein concentrates on the multidimensional generalization scheme. Unlike sub-tree or cell schemes, multidimensional scheme can produce consistent anonymous data that can be directly used by existing data mining and data

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

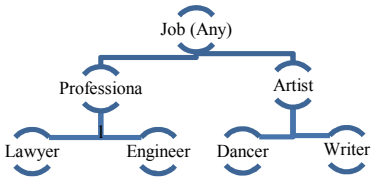
analysis tools. This scheme offers a good trade-off between data value and data stability. Thus, this scheme has been extensively explored. Top-Down Specialization (TDS) [5, 6, 7, 12] and Bottom-Up Generalization (BUG) [8] are two ways to accomplish the multidimensional scheme. Most exiting algorithms exploit indexing data structure to assist the process of anonymization. Specifically, TIPS (Taxonomy Indexed Partition S) for TDS and TEA (Taxonomy encoded Anonymity) index for BUG. Although indexing data structures can speed up the process of data anonymization, these approaches often fail to work in parallel or distributed environments like cloud systems because the indexing structures are centralized. Mohammed et al. [6] proposed a TDS approach which however, mainly concerns privacy protection against other parities rather than scalability issues. Still, this approach only employs information gains as the search metric, resulting lower data utility than centralized ones. Previous work [12] leverages map-reduce to accomplish the intensive computation required in big data anonymization via TDS. But TDS probably performs slower than BUG when *k*-anonymity parameter is small. Scalability and efficiency of anonymization algorithms for privacy preservation has drawn attention of researchers. R-tree indexing, scalable decision trees and sampling techniques are introduced to achieve high scalability and efficiency [22, 23]. However, the proposed approaches aim at multidimensional scheme, thereby failing to work for sub-treegeneralization.

Map-reduce have been widely adopted in various data processing applications to boost scalability and efficiency [24, 25, 26]. Following this line, map-reduce is used to advance scalability and efficiency in this research on big data anonymization in cloud.

2.2 Problem Analysis

In this section, analysis is made on the problem of utilizing Top-Down Specialization (TDS) or Bottom-Up Generalization (BUG) alone for multidimensional generalization, and the scalability problem of existing BUG approaches. At present, existing TDS and BUG approaches are developed individually for multidimensional generalization scheme. Both of them lack the awareness of the user-specified *k*-anonymity parameter. In fact, the values of the *k*-anonymity parameter can impact their performance.

Intuitively, if parameter *k* is large, TDS is more suitable while BUG will probably get bad performance. The case is reversed when *k* is small. A simple example is described below to demonstrate the above intuition. Assume that a data set has 10 records with the attribute Job that needs generalization for privacy preservation. The taxonomy tree of this attribute and attribute value of records are depicted in Fig. 1.



Any Job	# of Recs.
Lawyer	2
Engineer	2
Dancers	3
Writer	3

Figure 1: Taxonomy trees for Job

At one extreme, let *k* be set as 2. In this case, TDS has to specialize several domain values to achieve 2-anonymity while BUG will do nothing as the data set is already 2- anonymity. At other extreme, let *k* be set as 5. In such a case, TDS will do nothing to achieve 5-anonymity while BUG has to conduct generalization. It can be seen from the example that selecting TDS or BUG according to *k* significantly impacts the performance of the multidimensional anonymization scheme. As such, it is promising to develop a hybrid approach that encompasses TDS and BUG as two components so that the high scalability and efficiency can be gained regardless of valuing of the *k*-anonymity parameter. The key to the hybrid approach is to design a user-friendly method to automatically determine the component that should be chosen. Although a domain expert is able to determine the approach to be preferred to conduct anonymization manually according to the value of parameter *k*. Ordinary users in the cloud probably fail to do this due to their lack of background knowledge.

As to BUG, the existing approach [8] exploits indexing data structure to promote efficiency, thereby falling short of high scalability and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

parallelization in cloud environments. Thus, it is worthwhile investigating the methodology to develop BUG algorithm with map-reduce in order to improve the scalability and efficiency. A promising way is to conduct generalization operations in parallel. As map-reduce only provides primitive programming model to develop map-reduce jobs to conduct the computation required by data anonymization is still critical in the whole design and needs intensive research. Above all, the problem is to address the design together with scalable and efficient BUG algorithm, based on map-reduce and to automatically select a component for the proposed hybrid approach according to parameter *k*.

3. PRELIMINARY

3.1 Multidimensional Generalization Scheme

Each generalization or suppression operation hides some details in *QID*. For a categorical attribute, a specific value can be replaced with a general value according to a given taxonomy.

In Figure 1, the parent node Professional is more general than the child nodes Engineer and Lawyer. The root node, ANY Job, represents the most general value in Job. For a numerical attribute, exact values can be replaced with an interval that covers exact values. If taxonomy of intervals is given, the situation is similar to categorical attributes. More often, however, no pre-determined taxonomy is given for a numerical attribute [27]. Different classes of anonymization operations have different implications on privacy protection, data utility, and search space. But they all result in a less precise but consistent representation of original data. A generalization replaces some values with a parent value in the taxonomy of an attribute. The reverse operation of generalization is called specialization. One such generalization scheme is Multidimensional generalization [28,29].

Let D_i be the domain of an attribute A_i. A single- dimensional generalization, such as full-domain generalization and subtree generalization, is defined by a function $f_i : D_{Ai} \rightarrow D'$ for each attribute A_i in *QID*. In contrast, a multidimensional generalization is defined by a single function $fi : D_{A1} \times \cdots \times D_{An} \rightarrow D'$, which is used to generalize $qid = <v_1, \ldots, v_n>$ to $qid' = <u_1, \ldots, u_n>$ where for every v_i, either $v_i = u_i$ or v_i is a child node of u_i in the taxonomy of Ai.. This scheme flexibly allows two *qid* groups, even having the same value on some v_i and u_i, to be independently generalized into different parent groups. For example <*Engineer, Male*> can be generalized to <*Engineer, ANY_Sex*> while <*Engineer, Female*> can be generalized to <*Professional, Female*>. The generalized table contains both *Engineer* and *Professional*. This scheme produces less distortion than the other schemes generalization schemes because it needs to generalize only the *qid* groups that violate the specified threshold. Note, in this multidimensional scheme, all records in a *qid* are generalized to the same *qid'*, but cell generalization does not have such constraint.

Two operations can be employed to accomplish this scheme, i.e., generalization for BUG and specialization for TDS, respectively. A generalization operation is to replace a domain value with its parent in a taxonomy tree while a specialization operation is to replace a domain value with its all child values. Formally, a generalization is represented as
gen: child(q) → *q* while a specialization is represented as *spe*
is the is a dom
consists of all
concept of anonymization level [12] is utilized to capture the degree of anonymization. Specifically, anonymization
level, denoted as *AL*, is a vector of domain values sets, i.e.,
$AL = < D_{A1} \times \cdots \times D_{An} >$ which is used to generalize *qid* to *qid'*.

To guide the selection of the best operations in the anonymization process, the goodness of a candidate generalization or specialization is measured by a search metric. The information/privacy is used as the search metric for this approach, i.e., the Information Gain per Privacy Loss (IGPL) for TDS and the Information Loss per Privacy Gain (ILPG) for BUG, respectively [4].

It is briefly describe the way to calculate the value of ILPG subsequently. Given a generalization *gen: child(q)* → *q* ,the ILPG of the generalization is calculated by:

$$ILPG(gen) = IL(gen) / (PG(gen) + 1) \quad (1)$$

The term *IL(gen)* is the information loss after performing *gen*, and *PG(gen)* is the privacy gain. Both of them are computed via statistical information derived from data sets. Let R_x denote the set of original records containing attribute

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

values that can be generalized to *x*. $|Rx|$ is the number of data records in R_x. Let $I(R_x)$ be the entropy of R_x . Then, *IL(gen)* is given by

$$IL(gen) = \sum c \in Child(q) \left(\frac{|Rc|}{|Rq|} \right) I(Rc) - I(Rq) \quad (2)$$

Let $A_P(gen)$ denote the anonymity after performing *gen*, while $A_c(gen)$ be that before performing *gen*. Then, the privacy gain from *gen* is calculated by

$$PG(gen) = A_P(gen) - A_C(gen) \quad (3)$$

3.2 Map Reduce Basics

Map-reduce is a scalable and fault-tolerant data processing framework that is capable of processing huge size of data in parallel with many low-end product computers [10]. In general, a map-reduce job comprises of two basic functions, *Map* and *Reduce*, defined over a data structure named key, value pair *(key, value)*. Specifically, the *Map* function can be formalized as *Map:* $(k_1, v_1) \rightarrow (k_2, v_2)$, i.e., *Map* takes a pair (k_1, v_1) as input and then outputs another intermediate key-value pair (k_2, v_2). These intermediate pairs are consumed by the *Reduce* function as input. Syntactically, the *Reduce* function can be signified as *Reduce:* $(k_2, list(v_2)) \rightarrow (k_3, v_3)$, i.e., *Reduce* takes intermediate k_2 and all its equivalent values $list(v_2)$ as input and outputs another pair (k_3, v_3). Usually, (k_3, v_3) list is the results which map-reduce users attempt to obtain. An instance running *Map* function is called *Mapper*, and that running *Reduce* function is called *Reducer*, respectively. Between *Map* phase and *Reduce* phase exists a *Shuffle* phase, during which the intermediate key-value pairs are sorted according to keys.

4. BOTTOM- UP GENERALIZATION USING MAP REDUCE

Bottom-Up Generalization is elaborated using map-reduce (MRBUG) in this section. Basically, a practical map-reduce program encompasses *Map* and *Reduce* functions, and a *Driver* that coordinates the macro execution of map-reduce jobs. Thus, the Bottom-Up Generalization map-reduce *Driver* is described in section IV. A. Section IV.B and IV.C presents the *ILPG* calculation and *data generalization* map-reduce in detail.

4.1 Bottom- Up Generalization Map reduce driver

Basically, Bottom-Up Generalization (BUG) approach of anonymization is an iterative process starting from the lowest anonymization level. The lowest anonymization level contains the internal domain nodes in the lowest level of taxonomy trees. Each round of iteration includes four major steps, namely, checking the current data set whether satisfies the anonymity requirement, calculating the ILPG, finding the best generalization and generalizing the data set according to the selected best generalization candidate. Calculating the ILPG and generalizing the data set involve accessing a large number of data records, thereby dominating the scalability and efficiency of bottom-up generalization. An existing approach [8] utilizes indexing data structure and retaining statistic information to improve the efficiency. But the approach suffers from poor scalability and efficiency in a big data scenario. Still, the approach fails to be adapted into map-reduce since Map-reduce does not support indexing data structure. As such, we propose to develop innovative map-reduce jobs for the ILPG computation. As the notion of anonymization level is introduced to describe anonymization status of a data set, it is unnecessary to generalize the data set concretely in each round in regards to efficiency. Instead, we abstractly generalize the data set over the current anonymization level. After the final anonymization level is obtained, we anonymize the data set in a one-pass map-reduce job. Algorithm 1 presents the Bottom-Up Generalization map-reduce (BUGMR) driver.

Algorithm 1 is described in detail as follows. Firstly, ILPG values of all generalizations are initialized (line 1). Line 2 checks whether the current anonymized data set satisfies the *k*-anonymity requirement. Line 3 finds the best generalization gen_{best} with the highest ILPG value and Line 4 generalizes this generalization by labeling it as *INACTIVE*. That a generalization is labeled as *INACTIVE* means the generalization will not be considered any more in following rounds, abstractly fulfilling anonymization on the data set. Let *SGSet (gen)* denote the set containing generalization *gen* and it's all siblings in the domain taxonomy tree. When the generalizations in *SGSet (gen)* are all labeled as *INACTIVE*, a new higher level generalization should be inserted to replace these inactive ones (lines 5, 6 and 7). Note that this is a remarkable difference from TDS. Line 9 updates the privacy gain of each active

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

generalization as the performing gen_{best} probably changes the anonymity of the data set. Also, information loss computation is required if a new generalization has been inserted. As the last step, line 11 concretely anonymizes the data set according the final anonymization level. Lines 1 and 9 require *ILPG* calculation that involves accessing to the original data set and computing statistic information over the data set. Line 11 also needs processing the whole data set. Map-reduce technology is applied to conduct the intensive computation in these situations. Specifically, we design a couple of innovative map-reduce jobs: job *ILPG* Calculation for accomplishing the computation required in lines 1 and 9, and job *Data Generalization* for achieving the final concrete anonymization in line 11. The *ILPG* related map-reduce job is elaborated in Section IV.B, and job *Data Generalization* is in IV.C, respectively.

Algorithm 1. **BUGMR Driver**

Input: data set D, anonymization level AL_0, anonymity parameter k.
Output: final anonymous data set D^*.
1: Initialize the values of search metric ILPG for each generalization
$gen \in U_{i=1}^{n} D_i$ with respect to AL_0 via job *ILPG Calculation*;
2: while $\exists_{gen}$, $A_c(gen) < k$
3: Find the best generalization gen_{best} out of all the active ones;
4: Label gen_{best} as *INACTIVE* to perform gen_{best} on the current
anonymization level;
5: if $\forall\, gen \in SGSet(gen_{best})$ is labeled as *INACTIVE*;
6: Insert a new generalization gen_{new}: $child(q) \rightarrow q$ where
$Child(q) = \{ q_i \mid gen': Child(q_i) \rightarrow q_i , gen' \in SGSet(gen_{Best})\}$;
7: Remove all generalizations in $SGSet(gen_{best})$
8: end if
9: $AL_{i+1} \leftarrow AL_i$;Update ILPG values for all active generalization
candidates in AL_{i+1} via *ILPG Calculation*;
10: end while
11: Generalize D to D* in terms of AL_i via job Data Generalization;

4.2 ILPG Calculation Job

The *ILPG Calculation job* is responsible to ILPG initialization in line 1 of Algorithm 1 and ILPG update in line 9. The computation required in ILPG initialization is quite similar to that of ILPG update. The *Map* function of the *ILPG Calculation* is depicted in Algorithm 2, while the Reduce function is presented in Algorithm 3. In Algorithm 2 and 3, the symbol '#' is used to identify whether a key is emitted to compute information gain or anonymity loss, and '$' is to differentiate the cases whether a key is for computing A_P *(Spec) or* A_C *(Spec)*.

Algorithm 2 is detailed as follows. Let *NGSet* denote the set of newly inserted generalizations. For ILPG initialization, *NGSet* is the set of all the initial generalizations with respect to AL_0 , while for ILPG updates, it is set by $\{gen_{new}\}$ if a new generalization is inserted in the iteration of Algorithm 1. Line 1 of Algorithm 2 transforms an original record into its anonymized form according to the current anonymization level, for the sake of being counted. To compute $|R_p|$, $|(R_p , sv)|$, $|R_c|$ and $|(R_c , sv)|$ in (2) for information loss calculation, line 2 emits the key-value pair to the *Reduce* function for information loss computation if this pair is a new generalization candidate. Note that the information loss of a generalization will not be affected when we perform other generalizations or insert a new generalization, while privacy gain will probably be impacted as the anonymity of the data set will change.

Algorithm 2. **ILPG Calculation Map.**

Input: Data record (ID_r, r), $r \in D$ anonymization level *AL, NGSet.*
Output: Intermediate key-value pair (*key,count).*
1: For each attribute value v_i in r ,find its generalization in current
AL: gen_i . Let p_i be the parent in gen_i, and c_i be v_i itself or p_i ,
child that is also v_i' ancestor;
2: If $gen_i \in$, emit ($< p_i , c_i , sv >$, *count)* ;
3: Construct quasi-identifier $qid = < q_1 , q_{2,\ldots,} q_m >$ where q_i

$$= \begin{cases} pi, if\ geni\ is\ INACTIVE \\ ci, \quad o.w \end{cases}, 1 \le i \le m;$$ Emit (<*qid* , *$,#> , count);*

4: For each $i \in [1, m]$, replace q_i in *qid* with its parent p_i if $q_i = c_i$, producing the resultant quasi-identifier *qid* * ; emit (<*qid* , p_i ,#> , *count);*

Line 3 of Algorithm 2 aims at figuring out the *anonymity* of the data set before performing a generalization, i.e., A_c *(gen)*, while line 4 emits key-value pairs to obtain the anonymity after performing a generalization, i.e., A_p *(gen)*. As A_c *(gen)* is unique globally, we just emit the current quasi-identifier *qid* for statistics. As to A_p *(gen)*, potential anonymous quasi-identifiers for *qid* will be emitted for computing A_p *(gen)* for different

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

active generalization candidates. After obtaining A_c *(gen)* and A_p *(gen)*, we can update privacy gain for each generalization in terms of (3).

The *Reduce* function described in Algorithm 3 mainly aggregates the statistical information to calculate information loss and privacy gain. Lines 1 to 5 calculate information loss in terms of (2). Due to that the key-value pairs are sorted by map-reduce before being fed to *Reducer* workers, the *Reduce* function can compute information loss for generalizations in sequence, without requiring large amount of memory to retaining statistical information. Therefore, the *Reduce* function is highly scalable for calculating information loss. The essential of computing anonymity of a data set is to find out the minimum QI-group size. Lines 6 to 10 aim at calculating privacy gain. The *Reducer* workers find out the locally minimum QI-group size before and after performing a generalization in parallel. Then, we can obtain the globally one in the driver program through comparing the outputs of *Reducer* workers. As such, the *ILPG Calculation* Reduce function is highly scalable for both information loss and privacy gain computation. After obtaining information loss and privacy gain, ILPG values can be calculate according to (1).

Algorithm 3: ILPG Calculation Reduce

Input: Intermediate key-value pair (*key , list(count)).*
Output: Information gain (*gen, IL(gen))* and anonymity (*gen*, A_c *(gen)* , (*gen* , A_p *(gen)*) for generalizations.
1: For each *key, sum* ← ∑ *count;*
2: For each *key*, if *key.sv* ≠ **#** , update statistical counts:
3: $|(R_c, sv)|$ ← *sum,* $|R_c|$ ← *sum* +$|R_c|$, $|(R_c, sv)|$ ← *sum* +
$|(R_c, sv)|$, $|R_p|$ ← *sum* +$|R_p|$;
4: If all sensitive values for child *c* have arrived, compute $I(R_c)$;
5: If all children *c* of parent *p* have arrived, compute $I(R_p)$ and
IL(gen) ; emit (*gen, IL(gen))*;
6: For each *key*, if *key.sv* = **#**, update anonymity:
7: If *key.c* = **$** and *sum* < A_p *(gen)*, update current anonymity:
A_p *(gen)* ← *sum;*
8: If *key*.c ≠ *$;*
9: If *sum* < *Ac (gen)*, update potential anonymity of *gen* :
A_c *(gen)* ← *sum;*
10: Emit (*gen* , A_p *(gen)* and emit (*gen* , A_c *(gen)* .

4.3 Data Generalization

The original data set is concretely generalized for data anonymization by a one-pass map-reduce job, i.e., *Data Generalization*. Details of *Map* and *Reduce* functions of the data specialization Map-reduce job are described in Algorithm 4.

Algorithm 4 : Data Generalization Map & Reduce.

Input: Data record (ID_r , r), $r \in D$; final anonymization level AL^*.
Output: Anonymous record r^*, *count)*..
Map:
1: For each attribute value v_i in r, find its generalization in current
AL: gen_i . Let p_i be the parent in gen_i and c_i be v_i itself or p_i'
child that is also v_i' ancestor;
2: Construct quasi-identifier $r^* = < q_1 , q_{2,\ldots\ldots,} q_m >$ where q_i

$= \begin{cases} pi, if\ geni\ is\ INACTIVE \\ ci, \quad o.w \end{cases}$, $1 \le i \le m$; Emit (r^*, *count);*
Reduce: For each r^*; emit *sum* ← ∑ *count;* emit (r^*, *sum);*

The *Map* function emits anonymous records and its count according to the current anonymization level. The *Reduce* function simply aggregates these anonymous records and counts their number. An anonymous record and its count represent a *QI-group*, and the *QI-groups* constitute the final anonymous data sets.

5. HYBRID APPROACH FOR MULTI-DIMENSIONAL ANONYMIZATION

5.1 Combining Top-Down Specialization and Bottom-Up Generalization

Now that the map-reduce version of Bottom-Up Generalization (MRBUG) has been developed in the last section, the two components, i.e., MRTDS [12] and MRBUG, are ready for the proposed hybrid approach of multidimensional anonymization over big data. In terms of the problem analysis in Section II.B, we need to determine which component is used to anonymize data after the anonymity parameter k is specified by a user. It is promising that the hybrid approach can automatically give out a threshold K such that if k ≥ K , MRTDS is selected, otherwise MRBUG is selected. Formally, we define this threshold as Workload Balancing Point.

Definition 1 (Workload Balancing Point) anonymity value of a data set, denoted as K, is defined as workload balancing point if it satisfies

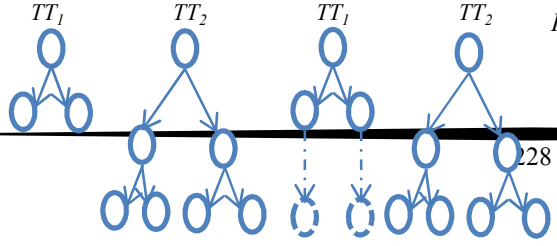
Journal of Theoretical and Applied Information Technology
20th April 2015. Vol.74 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

the condition that the amount of computation of anonymizing the data set to *K*-anonymous required by MRTDS is equal to that by MRBUG. Once the workload balancing point *K* is identified, it is easy to choose which component to be employed. If 8 > *K*, MRTDS is selected because MRBUG will incur more computation, whereas MRBUG is selected if 8 < *K*. Figuring out the exact value of *K* is difficult since *K* heavily depends on some properties of data sets, like data distribution and skewness. However, it is unnecessary to get the exact value because the performances of MRTDS and MRBUG make little difference when *k* is valued around *K*. As such, we roughly estimate the value *k* according to the size of the data set and taxonomy trees.

5.2 Workload Balancing Point Estimation

To roughly estimate the workload balancing point, we assume that the values of an attribute are evenly distributed. The basic idea is to estimate *K* according to the layering of taxonomy trees. Let *H* be the highest height among taxonomy trees. To facilitate the estimation, other taxonomy trees with height less than *H* are modified by making their height *H*. A string of dummy nodes are attached to their leaf nodes, in order to increase tree height. An example is demonstrated in Fig.2 to illustrate the above process. The tree TT_2 has the highest tree height two. As TT_1 has height less than two, dummy nodes (dashed circles in the right part of Fig.2) are added after modification. The height of modified version of TT_1 is also two. The data records in a data set *D* are logically partitioned by the domain value nodes in taxonomy trees. Let L_j denote the j^{th} layer of the taxonomy forest. The number of domain values of TT_i on L_j is denoted as N_{ij}. The data set can be partitioned into $\prod_{i=1}^{m} Nij$ QI-groups in regard to the domain values on level L_j. For instance, the data set is partitioned into 4 groups (2*2=4) on L_1 in shown in Fig.2, and 8 groups (2*4=8) on L_2. Since we assume data sets are evenly distributed, the average anonymity of data set with respect to L_j is $K_j = |D| / \prod_{i=1}^{m} Nij$ i.e., the average QI-group size.



(Original) (Modified)

Figure 2: An example of modifying the height of taxonomy trees.

To roughly identify *K*, we narrow down the interval where *K* is located. Firstly, the computation required by MRTDS and MRBUG to achieve K_j -anonymous on level L_j are estimated as follows. The performances of each round in both MRTDS and MRBUG are mainly dominated by the computation of the anonymity of data sets in the loop. Further, the computation required in one round of MRTDS and MRBUG, are roughly equal. Let Diagram C_{Unit} denote the computation required in one round of MRTDS or MRBUG. In MRTDS, performing a specialization operation will launch IGPL update and incur C_{Unit} computation. So, the total amount of computation to specialize *D* to L_j, denoted as C_j^{TDS} be estimated by

$$C_j^{TDS} = \sum_{l=1}^{j} \sum_{i=1}^{m} C_{Unit} \quad N_{il} \tag{4}$$

In MRBUG, unlike MRTDS, performing a generalization operation possibly does not trigger ILPG update. Generally, several operations can launch ILPG update and incur C_{Unit} computation. On average, we assume $1/\alpha$ operations can do this, where $0 \le \alpha \le 1$. Then, the total amount of computation to generalize *D* to L_j, denoted as C_j^{BUG} can be estimated by

$$C_j^{BUG} = \sum_{l=j}^{H-1} \sum_{i=1}^{m} C_{Unit} \cdot \alpha \cdot N_{il} \tag{5}$$

The value of *j* that make C_j^{TDS} equal to C_j^{BUG} must be located in interval [*J-1, J*], where *J* satisfies the condition: $C_j^{TDS} \ge C_j^{BUG}$ and $C_{J-1}^{TDS} \le C_{J-1}^{BUG}$. Once *J* is identified, we can estimate that the workload balancing point *K* lies within $[K_J, K_{J-1}]$. Let β be the average branch factor of the taxonomy forest. As *K* varies exponentially with respect to j, where $1 \le j \le H$, we estimate the workload balancing point at the middle position between K_J and K_{J-1} by the following formula:

$$K = K_J + |D| \cdot (1/\beta^{m(j-1)} - 1/\beta^{mj}) \tag{6}$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Once the workload balancing point is estimated, the hybrid approach can easily determine which component to be chosen, via comparing K with k-anonymity parameter.

6. EXPERIMENT EVALUATION

6.1 Experiment Settings

To evaluate the effectiveness and efficiency of the hybrid approach, we compare it with MRTDS [12] and MRBUG. We denote the execution time of the three approaches as T_{Hyb}, T_{TDS} and T_{BUG}, respectively. In general, T_{Hyb} is similar to T_{TDS} if k is larger than K, while T_{Hyb} is similar to T_{BUG} if k is less than K. Estimating K incurs overheads, yet it is minor compared with the computation conducted in map-reduce jobs. Our experiments are conducted in a cloud environment named U-Cloud. U-Cloud is a cloud computing environment at University of Technology Sydney (UTS).

The Hadoop cluster consists of 20 VMs with type m1.medium which has 2 virtual CPUs and 4 GB Memory. Each round of experiment is repeated 20 times. The mean of the measured results is regarded as the representative. We utilize the Adult data set and its generated data sets like [12].

6.2 Experiment Process and Results

We measure the change of execution time T_{Hyb}, T_{TDS} and T_{BUG} with respect to anonymity parameter k. The size of data set is set as 1000 MB. Equally, the data set contains 1.1_107 data records, which is big enough to evaluate the effectiveness of our approach in terms of data volume or the number of data records. Parameter U is set as 0.5.

The measures are shown in the following table:

Table 1: Execution time THyb, TTDS and TBUG with respect to anonymity parameter k

Exp (K_I= 5 *10^Exp)	1	2	3	4	5	6	7	8
T_Hyb	0	500	2000	2000	1500	700	500	0
T_TDS	5000	2000	2000	2500	4000	4500	4700	5000
T_BUG	0	500	2000	2000	1500	700	500	0

The graph for the above table is given below in Fig. 3 which demonstrates the change of T_{Hyb}, T_{TDS} and T_{BUG} with respect to k ranging from 0 to 1.1 *10^7. For conciseness, k is indicated by Exp which is the exponent of the scientific notation of k, i.e., $k = 1.1*10^{Exp}$. So, Exp ranges from 0 to 7. We can see from Fig.4 that the execution time of the hybrid approach is kept under a certain level, while both MRTDS and MRBUG incur high execution time when k is small and large, respectively. Further, the right part of the curve of T_{Hyb} is near to T_{BUG}, while the right part is near to T_{TDS}. This is because the hybrid approach utilizes MRTDS and MRBUG as components to conduct concrete computation. As a conclusion, the experimental results demonstrate that the hybrid approach significantly improves the performance of multidimensional anonymization over existing approaches regardless of the k – anonymity parameter.

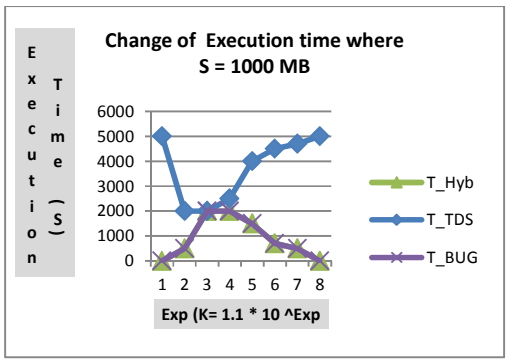


Figure 3: Change of execution time w.r.t anonymity parameter k.

7. CONCLUSION AND FUTURE WORK

In this paper, we have investigated the scalability issue of multidimensional anonymization over big data on cloud, and proposed a hybrid approach that combines Top-Down Specialization (TDS) and Bottom-Up Generalization (BUG) together. The hybrid approach automatically selects one of the two components via comparing the user specified k-anonymity parameter with workload balancing point. Both TDS and BUG have been accomplished in a highly scalable way via a series of deliberately designed map-reduce jobs. Experimental results have demonstrated that the hybrid method expressively advances the scalability and efficiency of multidimensional data anonymization associated with prevailing methods. In cloud environment, the privacy preservation for data analysis, share and mining is a challenging research issue due to increasingly larger volumes of datasets, thereby requiring intensive investigation. Based on the contributions herein, we plan to further explore the next step on scalable privacy reservation aware analysis and scheduling on large-scale datasets.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

REFERENCES

[1] S. Chaudhuri, "What next?: A half-dozen data management research goals for big data and the cloud", *Proceedings of 31st Symposium on Principles of Database Systems (PODS'12),* 2012,pp. 1-4.

[2] L. Wang, J. Zhan, W. Shi and Y. Liang, "In cloud, can scientific communities benefit from the economies of scale?", *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 2, 2012, pp. 296-303.

[3] H. Takabi, J.B.D. Joshi and G. Ahn, "Security and privacy challenges in cloud computing environments", *IEEE Security and privacy*, vol. 8, no. 6, 2010, pp. 24-31.

[4] B.C.M. Fung, K. Wang, R. Chen and P.S. Yu, "Privacy preserving data publishing: A survey of recent developments", *ACM Computer Survey*, vol. 42, no. 4, 2010, pp. 11-15.

[5] B.C.M. Fung, K. Wang and P.S. Yu, "Anonymizing classification data for privacy preservation", *IEEE Transactions on knowledge and Data Engineering,* vol. 19, no. 5, 2007, pp. 711-725.

[6] N. Mohammed, B. Fung, P.C.K. Hung and C.K. Lee, "Centralized and distributed anonymization for high dimensional healthcare data", *ACM Transactions on Knowledge Discovery from Data*, vol. 4, no. 4, article 18, 2010.

[7] B. Fung, K. Wang, L. Wang and P.C.K. Hung, "Privacy preserving data publishing for cluster analysis", *Data and Knowledge Engineering*, vol. 68, no. 6, 2009, pp. 552-575.

[8] W. Ke, P.S. Yu and S. Chakraborty, "Bottom-up generalization: A data mining solution to privacy protection", *Proceedings of 4th IEEE International Conference on Data Mining (ICDM'04),* 2004 pp. 249-256.

[9] L. Sweeney, "k-anonymity: A model for protecting privacy", *Knowledge-Based Systems*, vol. 10, no. 5, 2002, pp. 557-570.

[10] J. Dean and S. Ghemawat, "Map-reduce: A flexible data processing tool", *Communications of the ACM*, vol. 53, no. 1, 2010, pp. 72-77.

[11] Amazon Web Services, "Amazon Elastic Map-reduce (Amazon EMR)," *http://aws.amazon.com/elasticmap reduce/*, accessed on March 15, 2013.

[12] X. Zhang, L.T. Yang, C. Liu and J. Chen, "A scalable two phase top-down specialization approach for data anonymization using Map Reduce on cloud", *IEEE Transactions on Parallel and Distributed Systems,* Vol. 25, no. 2,2013, pp. 363-373.

[13] A. Machanavajjhala, D. Kifer, J. Gehrke and M.Venkitasubramaniam, "l-diversity: Privacy beyond kanonymity", *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, article 3, 2007.

[14] N. Li, T. Li and S. Venkatasubramanian, "Closeness: A new privacy measure for data publishing", *IEEE Transactions on Knowledge and Data Engineering,* vol. 22, no. 7, 2010, pp. 943-956.

[15] X. Xiao and Y. Tao, "m-invariance: Towards privacy preserving re-publication of dynamic datasets", *Proceedings of 2007 ACM SIGMOD International Conference on Management of Data (SIGMOD'07)*, 2007, pp. 689-700.

[16] K. LeFevre, D.J. DeWitt and R. Ramakrishnan, "Mondrian multidimensional k-anonymity", *Proceedings of 22nd International Conference on Data Engineering (ICDE '06)*, article 25, 2006.

[17] J. Xu, et al., "Utility-based anonymization for privacy preservation with less information loss", *ACM SIGKDD Explorations Newsletter*, vol. 8, no. 2, 2006, pp. 21-30.

[18] X. Xiao and Y. Tao, "Anatomy: Simple and effective privacy preservation", *Proceedings of 32nd International Conference on Very Large Data Bases (VLDB'06)*, 2006, pp. 139-150.

[19] T. Li, N. Li, J. Zhang and I. Molloy, "Slicing: A new approach for privacy preserving data publishing", *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 3, 2012, pp. 561-574.

[20] M. Terrovitis, J. Liagouris, N. Mamoulis and S. Skiadopolous, "Privacy preservation by disassociation", *Proceedings of VLDB Endowment*, vol. 5, no. 10, 2012, pp. 944-955.

[21] K. LeFevre, D.J. DeWitt and R. Ramakrishnan, "Incognito:Efficient full-domain k-anonymity", *Proceedings of 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD '05),* 2005, pp. 49-60.

[22] K. LeFevre, D.J. DeWitt and R. Ramakrishnan, "Workload aware anonymization techniques for large-scale datasets", *ACM Transactions on Database Systems,* vol. 33, no. 3, 2008, , pp. 1-47.

[23] T. Iwuchukwu and J.F. Naughton, "K-anonymization as spatial indexing: Toward scalable and incremental anonymization", *Proceedings of 33rd International Conference on Very Large Data Bases (VLDB'07),* 2007, pp. 746-757.

[24] A. Ene, S. Im and B. Moseley, "Fast Clustering Using Map Reduce", *Proceedings of 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*, 2011, pp. 681-689.

[25] I. Palit and C.K. Reddy, "Scalable and Parallel Boosting with Map Reduce", *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 10, 2012, pp. 1904-1916.

[26] R. Vernica, M.J. Carey and C. Li, "Efficient Parallel Set Similarity Joins Using Map Reduce", *Proceedings of 2010 ACM SIGMOD International Conference on Management of Data (SIGMOD'10),* 2010, pp. 495-506.

[27]

[28] X. Xiao and Y. Tao, "Personalized privacy preservation", *Proceedings of 2006 ACM SIGMOD International Conference on Management of Data (SIGMOD'06),* 2006, pp. 229-240.

[29] KVM, *http://www.linux-kvm.org/page/Main_Page*, accessed on March 15, 2013.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[30] Open Stack, *http://openstack.org/,* accessed on March 15, 2013.
[31] Apache, "*Hadoop," http://hadoop.apache.org,* accessed on March 15, 2013.
[32] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional k-anonymity", *Proceedings of 22nd IEEE International Conference on Data Engineering (ICDE)*, Atlanta, GA, 2006.
[33] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Workload-aware anonymization", *Proceedings of 12th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD),* Philadelphia, PA, August 2006.