Data Privacy by Top Down Specialization using MapReduce Framework

Santhosh Kumar K
Dept. of Computer Science
HKBK College of Engineering
Bangalore, India.

***Abstract*—For the current research, data analysis and data mining we require the private data to be shared which brings privacy concerns. The privacy preservation of users data is very important since some of the countries have passed the privacy laws which tells that the sensitive information should not be exposed. Even after removing explicit identifying information such as Name and SSN, it is still possible to link released records back to their identities by matching some combination of nonidentifying attributes such as Sex, Zip, Birthdate. A useful approach to prevent such linking attacks, called k-anonymization is used to preserve the privacy by generalization. At present, the trend is Big Data, normal algorithms cannot handle the very large data. Scalability problems arise since current anonymization algorithms cannot handle the large datasets. In this paper we propose top down specialization algorithm which is done in two phase for data anonymization using mapreduce framework. Our algorithm will effectively perform anonymization and handles the scalability**.

Keywords*—K-Anonymity; Data Anonymization; top down specialization; Data Privacy; Cloud

I. Introduction

Cloud computing is an evolving paradigm withtremendousmomentum, but its unique aspects has concerns about security and privacy challenges. Cloud computing has generated significantinterest in both academia and industry, butit's still an evolving paradigm. Cloud computing provides massive computation power and storage capacity by combining the commodity computers and it can be accessed through internet. Cloud computing reduces the investment for IT infrastructure and it can be used based on pay as you go basis. Even though cloud computing provides all the facilities; the customers are hesitant to use cloud services due to privacy and security concerns [1].

Privacy is one of the most concerned issues in cloud computing. Electronic health records and financial transaction records usually contains sensitive information but these data can offer significant human benefits if they are analyzed and mined by various research centers. For instance, Microsoft HealthVault, an online cloud health service, aggregates data from users and shares data with various research organizations. The data can be easily exposed by using traditional privacy protection on cloud. This can bring considerable economic loss or severe social reputation impairment to data owners. Hence, data privacy issues need to be addressed urgently before data sets are analyzed or shared on cloud.

Data anonymization has been extensively studied and widely adopted for data privacy and preservation in non interactive data publishing and sharing scenarios [11]. Data anonymization refers to hiding identity and/or sensitive data of owner's data records. Then, the privacy of an individual can be effectively preserved while certain aggregate information is exposed to data users for various analysis and mining. A variety of anonymization algorithms have been proposed [12], [13], [14], [15]. However, the scale of datasets that need anonymization in some cloud increases tremendously in accordance with the cloud computing and Big Data trend [1], [16]. Since the data sets size is very large it is difficult for the traditional anonymization algorithms to handle large datasets. The researchers have started investigation on the scalability problem of the large scale data anonymization [17], [18].

Large scale data processing framework like MapReduce [19] has been integrated with cloud to provide higher and powerful computation capability for applications. So it will be useful for us to use such framework for addressing scalability problem in large scale data anonymization. So we use MapReduce to address the scalability problem of Top Down Specialization (TDS) approach [12] for large scale data anonymization. TDS is one of the widely used approach for data anonymization [12], [20], [21], [22]. TDS algorithms are centralized so it cannot handle large data sets. There are few distributed algorithms proposed [20], [22] but they handle the anonymization of third party data sets and they do not focus on scalability. Even though MapReduce is simple to implement, it is difficult to fit TDS in MapReduce framework.

In this paper, we propose a scalable two phase TDS approach for data anonymization using MapReduce on cloud. Here we use highly optimized and highly efficient ARX anonymization tool libraries for k anonymity and Top Down Specialization [34]. In this we split the anonymization process into 2 phases. In the first phase, original data sets are partitioned into group of smaller data sets, and these data sets are anonymized in parallel, to get intermediate results. In the second phase, the intermediate results are integrated into one, and further anonymized to get consistent k-anonymous [23] data sets. We use MapReduce in both the phases. Experimental results show that in our approach efficiency and scalability of TDS is improved over existing approaches.

II. Related Work

A.Related Work

Recently, data privacy preservation has beenextensivelyinvestigated [11]. We briefly review related work below.LeFevre et al. [17] addressed the scalability problem ofanonymization algorithms via introducing scalable decisiontrees and sampling techniques. Iwuchukwu and Naughton[18] proposed an R-tree index-based approach by building aspatialindex over data sets, achieving high efficiency.However, the above approaches aim at multidimensionalgeneralization [15], thereby failing to work in the TDSapproach. Fung et al. [12], [20], [21] proposed the TDSapproach that produces anonymous data sets without thedata exploration problem [11]. A data structure TaxonomyIndexed PartitionS (TIPS) is exploited to improve theefficiency of TDS. But the approach is centralized, leadingto its inadequacy in handling large-scale data sets.

Several distributed algorithms are proposed to preserveprivacy of multiple data sets retained by multiple parties.Jiang and Clifton [24] and Mohammed [22] proposeddistributed algorithms to anonymize vertically partitioneddata from different data sources without disclosing privacyinformation from one party to another. Jurczyk and Xiong[25] and Mohammed et al. [20] proposed distributedalgorithms to anonymize horizontally partitioned data setsretained by multiple holders. However, the above distributedalgorithms mainly aim at securely integrating andanonymizing multiple data sources. Our research mainlyfocuses on the scalability issue of TDS anonymization, andis, therefore, orthogonal and complementary to them.

As to MapReduce-relevant privacy protection, Roy et al.[26] investigated the data privacy problem caused byMapReduce and presented a system named Airavat incorporatingmandatory access control with differentialprivacy. Further, Zhang [27] leveraged MapReduceto automatically partition a computing job in terms of datasecurity levels, protecting data privacy in hybrid cloud. Ourresearch exploits MapRedue itself to anonymize large-scaledata sets before data are further processed by otherMapReduce jobs, arriving at privacy preservation.

III. PRELIMINERY

A.Top-Down Speciaization

Generally, TDS is an iterative process starting from thetopmost domain values in the taxonomy trees of attributes.Each round of iteration consists of three main steps, namely,finding the best specialization, performing specializationand updating values of the search metric for the next round[12]. Such a process is repeated until k-anonymity isviolated, to expose the maximum data utility. The goodnessof a specialization is measured by a search metric. We adoptthe information gain per privacy loss (IGPL), a tradeoffmetric that considers both the privacy and informationrequirements, as the search metric in our approach. Aspecialization with the highest IGPL value is regarded asthe best one and selected in each round. We briefly describehow to calculate the value of IGPL subsequently to makereaders understand our approach well.

Given a specialization$spec: p \rightarrow Child(p)$, the IGPL ofthe specialization is calculated by

$$IGPL(spec) = IG(spec)/(PG(spec) + 1) \quad (1)$$

The term$IG(spec)$is the information gain after performing$spec$, $PL(spec)$is the privacy loss. $IG(spec)$and$PL(spec)$can be computed via statistical informationderived from data sets.Let$R_x$denote the set of originalrecords containing attribute values that can be generalizedto$x$. $|R_x|$is the number of data records inR_x. Let $I(R_x)$bethe entropy ofR_x. Then, $IG(spec)$ is calculated by

$$IG(spec) = I(R_p) - \sum_{c \in Child\,(p)} \left(\frac{|R_c|}{|R_p|}\right) I) \quad (2)$$

Let $|(R_x, sv)|$denote the number of the data records withsensitive valuesvin R_x. $I(R_x)$is computed by

$$I(R_x) = -\sum_{sv \in SV} \left(\frac{|(R_x,\ sv)|}{|(R_x|}\right). log_2\left(\frac{|(R_x,\ sv)|}{|(R_x|}\right) \quad (3)$$

The anonymity of a data set is defined by the minimumgroup size out of all QI-groups, denoted as A, i.e., A= $\min_{qid \in QID}\{|QIG(qid)|\}$ where $|QIG(qid)|$ is the size of QIG(qid). Let $A_p(spec)$ be that after performingspec. Privacy loss by spec is calculated by

$$PL(spec) = A_p(spec) - A_c(spec) \quad (4)$$

IV. TWO PHASE TOP DOWN SPECIALIZATION (TPTDS)

A. Sketch of Two Phase Top Down Specialization

We propose a TPTDS approach to conduct the computationrequired in TDS in a highly scalable and efficient fashion.The two phases of our approach are based on the two levelsof parallelization provisioned by MapReduce on cloud. Combined with cloud, MapReduce becomesmore powerful and elastic as cloud can offer infrastructureresources on demand, for example, Amazon ElasticMapReduce service [29]. To achievehigh scalability, we are parallelizing multiple jobs on datapartitions in the first phase, but the resultant anonymizationlevels are not identical. To obtain finally consistentanonymous data sets, the second phase is necessary tointegrate the intermediate results and further anonymized entire data sets. Details are formulated as follows.

In the first phase, an original data set$D$is partitionedinto smaller ones.Let $D_i, 1 \le i \le p$, denote the data setspartitioned from$D$, where p is the number of partitions, and $D = \sum_{i=1}^{p} D_i\,, D_i, D_i \cap D_j = \emptyset, 1 \le i < j \le p$.

Then, we run a subroutine over each of the partitioneddata sets in parallel to make full use of the job levelparallelization of MapReduce. The subroutine is a MapReduceversion of centralized TDS (MRTDS) which concretelyconducts the computation required in TPTDS. MRTDSanonymizes data partitions to generate intermediate anonymizationlevels. An intermediate anonymization levelmeans that further specialization can be performed withoutviolating k-anonymity. MRTDS only leverages the task levelparallelization of MapReduce.Formally, let function$MRTDS(D, k, AL) \rightarrow AL^1$ represent a MRTDS routine thatanonymizes data setDto satisfy k-anonymity from anonymizationlevelALto AL^1. Thus, a series of functions$MRTDS(D_i, k^I, AL^0) \rightarrow AL_i^1, 1 \le i \le p$, are executed simultaneouslyin the first phase. The termk^Idenotes theintermediate anonymity parameter, usually given by applicationdomain experts. Note thatk^Ishould satisfy $k^I \ge k$toensure privacy preservation.$AL^0$is the initial anonymizationlevel, i.e.,$AL^0 = \langle\{Top_1\},\{Top_2\},\ldots,\{Top_m\}\rangle$, where $Top_j \in Dom_j, 1 \le j \le m$, is the topmost domain value inTT_i. $AL_i^1$is the resultant intermediate anonymization level.

In the second phase, all intermediate anonymizationlevels are merged into one. The merged anonymizationlevel is denoted asAL^I. The merging process is formallyrepresented as function$merge(\langle AL_1', AL_2^1, \ldots, AL_p^1\rangle) \rightarrow AL'$. Then,the whole data setD is further anonymized based onAL^I, achieving k-anonymity finally, i.e.,$MRTDS(D, k, AL^I) \rightarrow AL^*$, where AL^* denotes the final anonymization level. Ultimately,D is concretely anonymized according toAL^*. Above all, Algorithm 1 depicts thesketch of the two-phase TDS approach.

Algorithm 1.SKETCH OF TWO-PHASE TDS (TPTDS).

Input: Data set D, anonymity parameters k, k^I and the number of partitions p.

Output: Anonymous data set D^*

1. Partition D into $D_i, 1 \leq i \leq p$.
2. Execute $MRTDS(D_i, k^I, AL^0) \rightarrow AL', 1 \leq i \leq p$ in parallel as multiple MapReduce jobs.
3. Merge all intermediate anonymization levels into one, $merge(AL_1', AL_2', \ldots, AL_p') \rightarrow AL'$.
4. Execute $MRTDS(D, k, AL^I) \rightarrow AL^*$ to achieve k-anonymity.
5. Specialize D according to AL^*, Output D^*.

The basic idea of TPTDS is to gain high scalability bymaking a tradeoff between scalability and data utility. Weexpect that slight decrease of data utility can lead to highscalability. The influence ofk^I and $p$on the data utility isanalyzed as follows. The data utility produced via TPTDS isroughly determined by$SP^I \cup SP_2$. Greater$p$means that the specializations in $SP^I$are selected according to IGPL valuesfrom smaller data sets, resulting in exposing less datautility. However, greater$p$also implies smallerSP^IbutlargerSP_2, which means more data utility can be producedbecause specializations in$SP_2$are selected according anentire data set. Largerk^I indicates larger SP_2, generatingmore data utility.

B. Data Partition

When D is partitioned into$D_i, 1 \leq i \leq p$, it is required that thedistribution of data records inD_i is similar to D. Adata recordhere can be treated as a point in an m-dimension space,where m is the number of attributes. Thus, the intermediateanonymization levels derived from$D_i, 1 \leq i \leq p$, can be moresimilar so that we can get a better merged anonymizationlevel. Random sampling technique is adopted to partition$D$, which can satisfy the above requirement.Specifically, arandom number$rand, 1 \leq rand \leq p$, is generated for eachdata record. A record is assigned to the partition D_{rand} .Algorithm 2 shows the MapReduce program of datapartition. Note that the number of Reducers should be equalto$p$, so that each Reducer handles one value of$rand$, exactlyproducing$p$resultant files. Each file contains a randomsample of$D$.

Algorithm 2. DATA PARTITION MAP & REDUCE

Input: Data record $(ID_r,\ r), r \in D$, partition parameter p.

Output: $D_i, 1 \leq i \leq p$

Map: Generate a random number $rand$, where $1 \leq rand \leq p$; emit $(rand, r)$.

Reduce: For each rand, emit $(null, list(r))$.

Once partitioned data sets$D_i, 1 \leq i \leq p$, are obtained, werun$MRTDS(D_i, k_I, AL^0)$on these data sets in parallel toderive intermediate anonymization levels$AL_i^*, 1 \leq i \leq p$.

C. Anonymization Level Merging

All intermediate anonymization levels are merged into onein the second phase. The merging of anonymization levels iscompleted by merging cuts. Specifically, letCut_a in AL_a' and Cut_b in AL_b'be two cuts of an attribute. There existdomain values$q_a \in Cut_a$and $q_b \in Cut_b$. that satisfy one of the three conditions $q_a$is identical toq_b, $q_a$is more generalthanq_b, or q_a is more specific than q_b. To ensure that themerged intermediate anonymization levelAL^Ineverviolates privacy requirements, the more general one isselected as the merged one, for example,q_b will be selected. $q_a$will be selected if $q_a$is more general than or identical toq_b. For the case ofmultiple anonymization levels, we can merge them in thesame way iteratively. The following lemma ensures thatAL^Istillcomplies privacy requirements.

Lemma 1. $If\ intermediate\ anonymization\ levels\ AL_i', 1 \leq i \leq p, satisfy\ k^I - anonymity, where$

$$AL^I \leftarrow merge(\langle AL_1', AL_2', \ldots, AL_p'\rangle), k' \geq k^I$$

Ourapproach can ensure the degree of data privacy preservation,as TPTDS produces k-anonymous data sets finally.Lemma 1 ensures that the first phase produces consistentanonymous data sets that satisfy higher degree of privacypreservation than users' specification. Then, MRTDS canfurther anonymize the entire data sets to produce final k-anonymousdata sets in the second phase.

D. Data Specialization

An original data set$D$ is is concretely specialized foranonymization in a one-pass MapReduce job. After obtainingthe merged intermediate anonymization levelAL^I, we run $MRTDS(D, k, AL^I)$on the entire data set$D$, and get thefinal anonymization levelAL^*. Then, the data set$D$isanonymized by replacing original attribute values in D withthe responding domain values inAL^*.

Details of Map and Reduce functions of the dataspecialization MapReduce job are described in Algorithm3. The Map function emits anonymous records and itscount. The Reduce function simply aggregates these anonymousrecords and counts their number. An anonymousrecord and its count represent a QI-group. The QI-groupsconstitute the final anonymous data sets.

Algorithm 3. DATA SPECIALIZATION MAP & REDUCE

Input: Data record $(ID_r, r), r \in D$; Anonymization level AL^*.

Output: Anonymous Record $(r^*, count)$.

Map: Construct anonymous record $r^* = p_1, \langle p_2, \ldots, p_m, sv\rangle, p_i, 1 \leq i \leq m$, is the parent of a specialization in current. $AL$and is also an ancestor of v_i in r; emit $(r^*, count)$.

Reduce: For each $r^*, sum \leftarrow \sum count$; emit (r^*, sum).

V. MAP REDUCE VERSION OF CENTRALIZED TDS

We elaborate the MRTDS in this section. MRTDS plays acore role in the two-phase TDS approach, as it is invoked inboth phases to concretely conduct computation. Basically, apractical MapReduce program consists ofMap and $Reduce$functions, and a$Driver$that coordinates the macro executionof jobs.

A.MRTDS Driver

Usually, a single MapReduce job is inadequate toaccomplisha complex task in many applications. Thus, a group ofMapReduce jobs are orchestrated in a driver program toachieve such an objective. MRTDS consists of$MRTDS\ Driver$and two types of jobs, i.e., $IGPL\ Initialization$and $IGPL\ Update$. The driver arranges the execution of jobs.

Algorithm 4 frames$MRTDS\ Driver$where a data set isanonymized by TDS. It is the algorithmic design of function$MRTDS(D, k, AL) \rightarrow AL'$ Notethat we leverage anonymization level to manage the processof anonymization. Step 1 initializes the values of informationgain and privacy loss for all specializations, which canbe done by the job$IGPL\ Initialization$.

Algorithm 4. MRTDS DRIVER

Input: Data set D anonymization level AL and k-anonymity parameter k.

Output: Anonymization level AL'.

1. Initialize the values of search metric IGPL, i.e., for each specialization $spec \in \cup_{j=1}^{m} Cut_j$. The IGPL value of $spec$ is computed by $IGPL\ Initialization$.
2. **While** $\exists\ spec \in \cup_{j=1}^{m} Cut_j$ is valid

   2.1. Find the best specialization from $AL_i, spec_{Best}$

   2.2. Update AL_i to AL_{i+1}

   2.3. Update information gain of the new specializations in AL_{i+1}, and privacy loss for each specialization via job $IGPL\ Update$.

   end while

$$AL' \leftarrow AL$$

Step 2 is iterative. First, the best specialization is selected from valid specializations in current anonymization level as described in Step 2.1. A specialization$spec$is a valid oneif it satisfies two conditions. One is that its parent value isnot a leaf, and the other is that the anonymity$A_c(spec) > k$, i.e., the data set is still k-anonymous if$spec$is performed.Then, the current anonymization level is modified viaperforming the best specialization in Step 2.2, i.e., removingthe old specialization and inserting new ones that arederived from the old one. In Step 2.3, information gain ofthe newly added specializations and privacy loss of allspecializations need to be recomputed, which are accomplishedby job$IGPL\ Update$. The iteration continues until allspecializations become invalid, achieving the maximumdata utility.

MRTDS produces the same anonymous data as thecentralized TDS in [12], because they follow the same steps.MTRDS mainly differs from centralized TDS on calculatingIGPL values. However, calculating IGPL values dominatesthe scalability of TDS approaches, as it requires TDSalgorithms to count the statistical information of data setsiteratively.

B.IGPL Initialization Job

The Map andReducefunctions of the job$IGPL\ Initialization$are described in Algorithms 5 and 6,respectively. The maintask of$IGPL\ Initialization$is toinitialize information gainand privacy loss of all specializations in the initialanonymization levelAL. According to (2) and (3), thestatistical information$|R_p|, |R_p, sv|, |R_c|$ and $|R_c, sv|$isrequired for each specialization to calculate informationgain.

Algorithm 5. IGPL INITIALIZATION MAP

Input: Data record $(ID_r, r), r \in D$; anonymization level AL.

Output: Intermediate key-value pair $(key, count)$.

1. For each attribute value $v_i$in r, find its specialization in current AL: $spec$. Let p be the parent in $spec$ and c be the p'schild value that is also an ancestor of v_i in TT_i.
2. For each v_i, emit $(\langle p, c, sv \rangle, count)$.
3. Construct quasi-identifier $qid^* = \langle p_1, p_2, \ldots, p_m \rangle$, where $p_i, 1 \leq i \leq m$, is the parent of a specialization in current AL. Emit $(\langle qid^*, \$, \# \rangle, count)$.
4. For each $i \in [1, m]$, replace p_i in qid^* with its child c_i is also ancestor of v_i. Let the resultant quasi-identifier be qid. Emit $(\langle qid^*, \$, \# \rangle, count)$.

Algorithm 5 describes the Map function. The input isdata sets that consist of a number of records. $ID_r$is thesequence number of the record$r$. Steps 1 and 2 are tocompute$|R_p|, |R_p, sv|, |R_c|$and $|R_c, sv|$. Step 1 gets thepotential specialization for the attribute values in$r$. ThenStep 2 emits key-value pairs containing the information ofspecialization, sensitive value, and the count information ofthis record. According to the above information, wecompute information gain for a potential specialization inthe corresponding$Reduce$ function. Step 3 aims at computingthe current anonymity$A_p(spec)$, while Step 4 is tocompute anonymity$A_c(spec)$after potential specializations.The symbol "#" is used to identify whether a key is emittedto compute information gain or anonymity loss, while thesymbol "$" is employed to differentiate the cases whether akey is for computing$A_p(spec)$or $A_c(spec)$.

Algorithm 6 specifies the$Reduce$function. The first step isto accumulate the values for each input key. If a key is forcomputing information gain, then the corresponding statisticalinformation is updated in Step 2.1.$I(R_p), I(R_c)$, and $IG(spec)$are calculated if all the count information they needhas been computed in Steps 2.2 and 2.3 in terms of (2) and

(3).A salient MapReduce feature that intermediate key-valuepairs are sorted in the shuffle phase makes the computationof$IG(spec)$sequential with respect to the order of specializationsarriving at the same reducer. Hence, the reducer justneeds to keep statistical information for one specialization ata time, which makes the reduce algorithm highly scalable.

Algorithm 6. IGPL INITIALIZATION REDUCE.

Input: Intermediate key-value pair $key, list(count)$.

Output: Information gain $spec, IG(spec)$ and anonymity $(spec, A_c(spec))$, $(spec, A_p(spec))$ for all specialization.

1. For each key, $sum \leftarrow \sum count$.
2. For each key, if $key.sv \neq \#$, update statistical counts:
   2.1. $|(R_c, sv)| \leftarrow sum, |R_c| \leftarrow sum + |R_c|$,
   $|(R_p, sv)| \leftarrow sum + |(R_p, sv)|, |R_p| \leftarrow sum + |R_p|$.
   2.2. If all sensitive values for child c have arrived, compute $I(R_c)$ according to 3.
3. For each key, if $key.sv = \#$, update anonymity.
   3.1. If $key.c = \$$and $sum < A_p(spec)$, update current anonymity: $A_p(spec) \leftarrow sum$.
   3.2. If $key.c \neq \$$and $sum < A_c(spec)$, update potential anonymity of $spec$: $A_c(spec) \leftarrow sum$.
4. Emit $(spec, A_p(spec))$ and emit $(spec, A_c(spec))$.

To compute the anonymity of data sets before and after aspecialization, Step 3.1 finds the smallest number of recordsout of all current QI-groups, and Step 3.2 finds all thesmallest number of records out of all potential QI-groupsfor each specialization. Step 4 emits the results ofanonymity. Note that there may be more than one keyvaluepair$(spec, A_p(spec))$for one specialization in outputfiles if more than one reducer is set. But we can find thesmallest anonymity value in the driver program. Then interms of (4), the privacy loss$PL(spec)$is computed. Finally,$IGPL(spec)$for each specialization is obtained by (1).

C. IGPL Update Job

The IGPL Updatejob dominates the scalability and efficiencyof MRTDS, since it is executed iteratively as described inAlgorithm 4. So far, iterative MapReduce jobs have not beenwell supported by standard MapReduce framework likeHadoop [30]. Accordingly, Hadoop variations like Haloop[31] and Twister [32] have been proposed recently tosupport efficient iterative MapReduce computation. Ourapproach is based on the standard MapReduce frameworkto facilitate the discussion herein.

The $IGPL\ Update$jobis quite similar to$IGPL\ Initialization$, except that it requires less computation and consumes lessnetwork bandwidth. Thus, the former is more efficient thanthe latter. Algorithm 7 describes theMapfunction of$IGPL\ Update$. The $Reduce$ function is same as IGPL Initialization, already described in Algorithm 3.

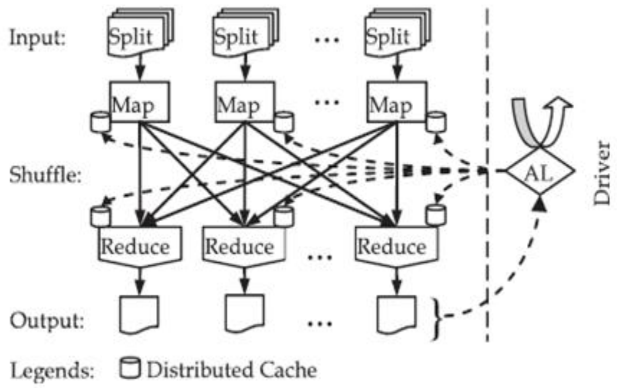

Fig. 1. Execution framework overview of MRTDS

Algorithm 7. IGPL UPDATE MAP

Input: Data Record $(ID_r, r), r \in D$; Anonymization Level AL.

Output: Intermediate key-value pair $(key, count)$.

1. Let $attr$ be the attribute of the last best specialization. The value of this attribute in $r$is v. Find its specialization in AL: $spec$. Let p be the parent in $spec$, and c be p's child that is also an ancestor of v; Emit $(\langle p, c, sv \rangle, count)$.
2. Construct quasi identifier $qid^* = \langle p_1, p_2, \ldots, p_m \rangle, P_i, 1 \leq i \leq m$, is the parent of a specialization in current AL and is also an ancestor of v_i in r.
3. For each $i \in [1, m]$, replace p_i in qid^* with its child c_i is also the ancestor of v_i in r.

After a specialization$spec$is selected as the bestcandidate, it is required to compute the information gainfor the new specializations derived from$spec$. So, Step 1 inAlgorithm 7 only emits the key-value pairs for the newspecializations, rather than all in Algorithm 5. Note that it isunnecessary to compute the information gain of otherspecializations because conducting the selected specializationnever affects the information gain of others. Comparedwith$IGPL\ Initialization$, only a part of data is processed andless network bandwidth is consumed.

Weneed to compute$A_c(spec)$for all specializations inAL, described in Step 2 and 3 of Algorithm 7. Yet$A_p(spec)$canbe directly obtained from the statistical information kept bythe last best specialization. Note that if the specializationrelated to$p_i$in Step 3 is not valid, no resultant quasi-identifierwill be created.

D. Implementation and Optimization

To elaborate how data sets are processed in MRTDS, theexecution framework based on standard MapReduce isdepicted in Fig. 1. The solid arrow lines represent the dataflows in the canonical MapReduce framework. From Fig. 1,we can see that the iteration of MapReduce jobs iscontrolled by anonymization levelALin $Driver$. The dataflows for handling iterations are denoted by dashed arrowlines.$AL$is dispatched from $Driver$ to all workers including $Mappers$ and $Reducers$ via the distributed cache mechanism. The value of AL is modified in $Driver$ according to the output of the $IGPL\ Intialization$ and $IGPL\ Update$ jobs. As the amount of such data is extremely small compared with datasets that will be anonymized, they can be efficiently transmitted between $Driver$ and workers.

We adopt Hadoop [30], an open-source implementation of MapReduce, to implement MRTDS. Since most of Map and $Reduce$ functions need to access current anonymization level AL. we use the distributed cache mechanism to pass the content of AL to each $Mapper$ or $Reducer$ node as shown in Fig. 1.

VI. CONCLUSIONS AND FUTUREWORKS

In this paper, we have investigated the scalability problem of large scale data anonymization by TDS, and proposed a highly scalable two-phase TDS approach using MapReduce on cloud. Here we use highly efficient and highly optimized ARX anonymization tools libraries for k anonymity and Top Down Specialization. Data sets are partitioned and anonymized in parallel in the first phase, producing intermediate results. Then, the intermediate results are merged and further anonymized to produce consistent k-anonymous data sets in the second phase. We have integrated anonymization algorithms to fit into MapReduce framework to achieve scalability. Experimental results show that the scalability and efficiency of TDS are improved significantly over existing approaches.

In cloud environment, the privacy preservation for data analysis. Sharing and mining is a challenging research issue due to increasingly larger volumes of data sets, thereby requiring intensive investigation. We will investigate the adoption of our approach with $l-diversity$, $t-closeness$ for data anonymization.

REFERENCES

[1] S. Chaudhuri, "What Next?: A Half-Dozen Data ManagementResearch Goals for Big Data and the Cloud," Proc. 31st Symp.Principles of Database Systems (PODS '12), pp. 1-4, 2012.

[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A.Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M.Zaharia, "A View of Cloud Computing," Comm. ACM, vol. 53,no. 4, pp. 50-58, 2010.

[3] L. Wang, J. Zhan, W. Shi, and Y. Liang, "In Cloud, Can ScientificCommunities Benefit from the Economies of Scale?," IEEE Trans.Parallel and Distributed Systems, vol. 23, no. 2, pp.296-303, Feb.2012.

[4] H. Takabi, J.B.D. Joshi, and G. Ahn, "Security and PrivacyChallenges in Cloud Computing Environments," IEEE Securityand Privacy, vol. 8, no. 6, pp. 24-31, Nov. 2010.

[5] D. Zissis and D. Lekkas, "Addressing Cloud Computing SecurityIssues," Future Generation Computer Systems, vol. 28, no. 3, pp. 83-592, 2011.

[6] X. Zhang, C. Liu, S. Nepal, S. Pandey, and J. Chen, "A PrivacyLeakage Upper-Bound Constraint Based Approach for Cost-Effective Privacy Preserving of Intermediate Data Sets in Cloud,"IEEE Trans. Parallel and Distributed Systems, to be published, 2012.

[7] L. Hsiao-Ying and W.G. Tzeng, "A Secure Erasure Code-BasedCloud Storage System with Secure Data Forwarding," IEEE Trans.Parallel and Distributed Systems, vol. 23, no. 6, pp. 995-1003, 2012.

[8] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-PreservingMulti-Keyword Ranked Search over Encrypted Cloud Data," Proc.IEEE INFOCOM, pp. 829-837, 2011.

[9] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, "Gupt:Privacy Preserving Data Analysis Made Easy," Proc. ACMSIGMOD Int'l Conf. Management of Data (SIGMOD '12), pp. 349-360, 2012.

[10] Microsoft HealthVault, http://www.microsoft.com/health/ww/products/Pages/healthvault.aspx, 2013.

[11] B.C.M. Fung, K. Wang, R. Chen, and P.S. Yu, "Privacy-PreservingData Publishing: A Survey of Recent Devel- opments," ACMComputing Surveys, vol. 42, no. 4, pp. 1-53, 2010.

[12] B.C.M. Fung, K. Wang, and P.S. Yu, "Anonymizing ClassificationData for Privacy Preservation," IEEE Trans. Knowledge and DataEng., vol. 19, no. 5, pp. 711-725, May 2007.

[13] X. Xiao and Y. Tao, "Anatomy: Simple and Effective PrivacyPreservation," Proc. 32nd Int'l Conf. Very Large Data Bases (VLDB'06), pp. 139-150, 2006.

[14] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Incognito:Efficient Full-Domain K-Anonymity," Proc. ACM SIGMOD Int'lConf. Management of Data (SIGMOD '05), pp. 49-60, 2005.

[15] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "MondrianMultidimensional K-Anonymity," Proc. 22nd Int'l Conf. DataEng. (ICDE '06), 2006.

[16] V. Borkar, M.J. Carey, and C. Li, "Inside 'Big Data Management':Ogres, Onions, or Parfaits?," Proc. 15th Int'l Conf. ExtendingDatabase Technology (EDBT '12), pp. 3-14, 2012.

[17] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Workload-AwareAnonymization Techniques for Large-Scale Data Sets," ACMTrans. Database Systems, vol. 33, no. 3, pp. 1-47, 2008.

[18] T. Iwuchukwu and J.F. Naughton, "K-Anonymization as SpatialIndexing: Toward Scalable and Incremental Anonymization,"Proc. 33rd Int'l Conf. Very Large Data Bases (VLDB '07), pp. 746-757,2007.

[19] J. Dean and S. Ghemawat, "Mapreduce: Simplified Data Processingon Large Clusters," Comm. ACM, vol. 51, no. 1, pp. 107-113,2008.

[20] N. Mohammed, B. Fung, P.C.K. Hung, and C.K. Lee, Centralizedand Distributed Anonymization for High-Dimensional HealthcareData," ACM Trans. Knowledge Discovery from Data, vol. 4, no. 4,Article 18, 2010.

[21] B. Fung, K. Wang, L. Wang, and P.C.K. Hung, "Privacy-Preserving Data Publishing for Cluster Analysis," Data andKnowledge Eng., vol. 68, no. 6, pp. 552-575, 2009.

[22] N. Mohammed, B.C. Fung, and M. Debbabi, "Anonymity MeetsGame Theory: Secure Data Integration with Malicious Participants,"VLDB J., vol. 20, no. 4, pp. 567-588, 2011.

[23] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," Int'lJ. Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 10, no. 5,pp. 557-570, 2002.

[24] W. Jiang and C. Clifton, "A Secure Distributed Framework forAchieving k-Anonymity," VLDB J., vol. 15, no. 4, pp. 316-333,2006.

[25] P. Jurczyk and L. Xiong, "Distributed Anonymization: AchievingPrivacy for Both Data Subjects and Data Providers," Proc. 23rdAnn. IFIP WG 11.3 Working Conf. Data and Applications SecurityXXIII (DBSec '09), pp. 191-207, 2009.

[26] I. Roy, S.T.V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel,"Airavat: Security and Privacy for Mapreduce," Proc. SeventhUSENIX Conf. Networked Systems Design and Implementation (NSDI'10), pp. 297-312, 2010.

[27] K. Zhang, X. Zhou, Y. Chen, X. Wang, and Y. Ruan, "Sedic:Privacy-Aware Data Intensive Computing on Hybrid Clouds,"Proc. 18th ACM Conf. Computer and Comm. Security (CCS '11),pp. 515-526, 2011.

[28] X. Xiao and Y. Tao, "Personalized Privacy Preservation," Proc.ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '06),pp. 229-240, 2006.

[29] Amazon Web Services, "Amazon Elastic Mapreduce," http://aws.amazon.com/elasticmapreduce/, 2013.

[30] Apache, "Hadoop,"http://hadoop.apache.org, 2013.

[31] Y. Bu, B. Howe, M. Balazinska, and M.D. Ernst, "The HaloopApproach to Large-Scale Iterative Data Analysis," VLDB J., vol. 21,no. 2, pp. 169-190, 2012.

[32] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu,and G. Fox, "Twister: A Runtime for Iterative Mapreduce," Proc.19th ACM Int'l Symp. High Performance Distributed Computing(HDPC '10), pp. 810-818, 2010.

[33] UCI Machine Learning Repository, ftp://ftp.ics.uci.edu/pub/ machine-learning-databases/,2013.

[34] ARX powerful data anonymization, http://arx.deidentifier.org/