International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 02 Issue: 05 | Aug-2015 www.irjet.net p-ISSN: 2395-0072

An Innovative Two-Step Top-Down Activity for Knowledge Anonymization Utilizing MapReduce on Cloud

Karthik Singh. Manchikalapati[1], Srinivasulu. Yaddala[2]

[1]*M. Tech Student, Department of Computer Science, Audisankara College of Engineering (Autonomous), Andhra Pradesh, India.*

[2]*Assistant Professor, Department of Computer Science, Audisankara College of Engineering (Autonomous), Andhra Pradesh, India.*

Abstract— ***At present, the scale of data in many cloud applications increases tremendously in accordance with the Big Data trend, thereby making it a challenge for commonly used software tools to capture, manage, and process such large-scale data within a tolerable elapsed time. As a result, it is a challenge for existing anonymization approaches to achieve privacy preservation on privacy-sensitive large-scale data sets due to their insufficiency of scalability. In this paper, we propose a scalable two-phase top-down specialization (TDS) approach to anonymized large-scale data sets using the MapReduce framework on cloud. Cloud provider where the MapReduce code is run on uploaded data. Secure MapReduce to provide confidentiality and privacy assurances for sensitive data. In both phases of our approach, we deliberately design a group of innovative MapReduce jobs to concretely accomplish the specialization computation in a highly scalable way. Experimental evaluation results demonstrate that with our approach, the scalability and efficiency of TDS can be significantly improved over existing approaches.***

Key Words—*Data Anonymization, Top-Down specialization, MapReduce, Cloud, Privacy preservation*

I.INTRODUCTION

Data sharing become day today activity for individuals, organizations and agencies. Most of the organizations are moving towards cloud to reduce the cost. Cloud systems provides massive computation power and storage capacity that enable cloud users to deploy applications without infrastructure investment.

Privacy is one of the most concerned issues in cloud Computing. Personal data like financial transaction records and electronic health records are extremely sensitive although that can be analyzed and mined by research organization. Data privacy issues need to be addressed before data sets are shared on cloud for analysis purpose. Data anonymization refers to as hiding sensitive data for owners of data records.

Cloud computing, a disruptive trend at present, poses a significant impact on current IT industry and research communities [1]. Cloud computing provides massive computation power and storage capacity via utilizing a large number of commodity computers together, enabling users to deploy applications cost-effectively without heavy infrastructure investment. Cloud users can reduce huge upfront investment of IT infrastructure, and concentrate on their own core business.

Data anonymization has been extensively studied and widely adopted for data privacy preservation in non-interactive data publishing and sharing scenarios [2]. 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 information is exposed to data users for diverse analysis and mining. A variety of anonymization algorithms with different anonymization operations have been proposed [3], [4]. However, the scale of data sets that need anonymizing in some cloud applications increases tremendously in accordance with the cloud computing and Big Data trends [1]. Large-scale data processing frameworks like MapReduce have been integrated with cloud to provide powerful computation capability for applications. So, it is promising to adopt such frameworks to address the scalability problem of anonymizing large-scale data for privacy preservation. In our research, we leverage MapReduce, a widely adopted parallel data processing framework, to address the scalability problem of the top-down specialization (TDS) approach [3] for large-scale data anonymization. The TDS approach, offering a good tradeoff between data utility and data consistency, is widely applied for data anonymization

[3], [6]. Most TDS algorithms are centralized, resulting in their inadequacy in handling large-scale data sets. Although some distributed algorithms have been proposed [6], they mainly focus on secure anonymization of data sets from multiple parties, rather than the scalability aspect.

In this paper, we propose a highly scalable two-phase TDS approach for data anonymization based on MapReduce on cloud. To make full use of the parallel capability of MapReduce on cloud, specializations required in an anonymization process are split into two phases. In the first one, original data sets are partitioned into a group of smaller data sets, and these data sets are anonymized in parallel, producing intermediate results. In second one, the intermediate results are integrated into one, and further anonymized to achieve consistent k-anonymous data sets. We leverage MapReduce to accomplish the concrete computation in both phases. A group of MapReduce jobs is deliberately designed and coordinated to perform specializations on data sets collaboratively. We evaluate our approach by conducting experiments on real- world data sets. Experimental results demonstrate that with our approach, the scalability and efficiency of TDS can be improved significantly over existing approaches. The major contributions of our research are threefold. First, we creatively apply MapReduce on cloud to TDS for data anonymization and deliberately design a group of innovative MapReduce jobs to concretely accomplish the specializations in a highly scalable fashion. Second, we propose a two-phase TDS approach to gain high scalability via allowing specializations to be conducted on multiple data partitions in parallel during the first phase. Third, experimental results show that our approach can significantly improve the scalability and efficiency of TDS for data anonymization over existing approaches.

II. LITERATURE REVIEW

A. Related Work

Recently, data privacy preservation has been extensively investigated [2]. Addressed the scalability problem of anonymization algorithms via introducing scalable decision trees and sampling techniques. Iwuchukwu and Naughton [5] proposed an R-tree index-based approach by building a spatial index over data sets, achieving high efficiency. However, the above approaches aim at multidimensional generalization, thereby failing to work in the TDS approach. Fung et al. [3], [6] proposed the TDS approach that produces anonymous data sets without the data exploration problem [2]. A data structure Taxonomy Indexed Partitions (TIPS) is exploited to improve the efficiency of TDS. But the approach is centralized, leading to its inadequacy in handling large-scale data sets.

Several distributed algorithms are proposed to preserve privacy of multiple data sets retained by multiple parties. Mohammed et al. [6] proposed distributed algorithms to anonymized vertically partitioned data from different data sources without disclosing privacy information from one party to another. Mohammed et al. [6] Proposed distributed algorithms to anonymize horizontally partitioned data sets retained by multiple holders. However, the above distributed algorithms mainly aim at securely integrating and anonym zing multiple data sources. Our research mainly focuses on the scalability issue of TDS anonymization, and is, therefore, orthogonal and complementary to them. As to MapReduce-relevant privacy protection, Roy et al. investigated the data privacy problem caused by MapReduce and presented a system named Arafat incorporating mandatory access control with differential privacy.

Further, MapReduce to automatically partition a computing job in terms of data security levels, protecting data privacy in hybrid cloud. Our research exploits MapReduce itself to anonymized large-scale data sets before data are further processed by other MapReduce jobs, arriving at privacy preservation.

B. Problem Analysis

We analyze the scalability problem of existing TDS approaches when handling large-scale data sets on cloud. The centralized TDS approaches in [3], [6] exploits the data structure TIPS to improve the scalability and efficiency by indexing anonymous data records and retaining statistical information in TIPS. The data structure speeds up the specialization process because indexing structure avoids frequently scanning entire data sets and storing statistical results circumvents recompilation overheads. On the other hand, the amount of metadata retained to maintain the statistical information and linkage information of record partitions is relatively large compared with data sets themselves, thereby consuming considerable memory. Moreover, the overheads incurred by maintaining the linkage structure and updating the statistic information will be huge when date sets become large. Hence, centralized approaches probably suffer from low efficiency and scalability when handling large-scale data sets. There is an assumption that all data processed should fit in memory for the centralized approaches [3]. Unfortunately, this assumption often fails to hold in most data-intensive cloud applications nowadays. In cloud environments, computation is provisioned in the form of virtual machines (VMs). Usually, cloud compute services offer several flavors of VMs. As a result, the centralized approaches are difficult in handling large-scale data sets well on cloud using just one single VM even if the VM has the highest computation and storage capability. A distributed TDS approach [6] is proposed to address the distributed anonymization problem which mainly concerns privacy protection against other parties, rather than scalability issues. Further, the approach only employs information gain, rather than its combination with privacy loss, as the

search metric when determining the best specializations. As pointed out in [3], a TDS algorithm without considering privacy loss probably chooses a specialization that leads to a quick violation of anonymity requirements. Hence, the distributed algorithm fails to produce anonymous data sets exposing the same data utility as centralized ones.

III. PROPOSED SYSTEM

The issue is how to handle the data in such a way that the privacy of individuals can be preserve. Various proposals have been designed for privacy preserving.

A. Top-Down Specialization

Generally, TDS is an iterative process starting from the topmost domain values in the taxonomy trees of attributes. Each round of iteration consists of three main steps, namely, finding the best specialization, performing specialization and updating values of the search metric for the next round [3]. Such a process is repeated until k-anonymity is violated, to expose the maximum data utility. The goodness of a specialization is measured by a search metric. We adopt the information gain per privacy loss (IGPL), a tradeoff metric that considers both the privacy and information requirements, as the search metric in our approach. A specialization with the highest IGPL value is regarded as the best one and selected in each round.

IGPL (spec) =IG (spec)/ (PL (SPEC) +1).

Advantages Of Proposed System

- ❖ Accomplish the specializations in a highly scalable fashion.
- ❖ Gain high scalability.
- ❖ Significantly improve the scalability and efficiency of TDS for data anonymization over existing approaches.
- ❖ The overall performance of the providing privacy is high.
- ❖ Its ability to handles the large amount of data sets.
- ❖ The anonymization is effective to provide the privacy on data sets.
- ❖ Here we using the scheduling strategies to handle the high amount of datasets.

B. Two-Phase Top-Down Specialization (TPTDS)

Three components of the TPTDS approach, namely, data partition, anonymization level merging, and data specialization.

A TPTDS approach in TDS is a highly scalable and efficient approach. The two phases of our approach are based on the two levels of parallelization provisioned by Map Reduce on cloud. Basically, Map Reduce on cloud has two levels of parallelization

- Job level
- Task level

Job level parallelization deals multiple MapReduce jobs that can be executed simultaneously to make full use of cloud infrastructure resources. Combined with cloud, MapReduce becomes more powerful and elastic as cloud can offer infrastructure resources on demand, for example, Amazon Elastic MapReduce service. Task level parallelization refers to that multiple mapper/reducer tasks in a MapReduce job are executed simultaneously over data splits. By parallelizing multiple jobs on data partitions in the first phase to achieve high scalability, but the resultant anonymization levels are not identical. To obtain finally consistent anonymous data sets, the second phase is necessary to integrate the intermediate results and further anonymized entire data sets.

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

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

Output: Anonymous data set D*.

1. Partition D into D_i, $1 \le I \le p$.
2. Execute MRTDS (D_i, kI, AL0) → AL′I, $1 \le I \le p$ In parallel as multiple MapReduce jobs.
3. Merge all intermediate anonymization levels into one, merge (AL′1, AL′2, and AL′p) → ALI.
4. Execute MRTDS (D, k, ALI) → AL*to achieve k-anonymity.
5. Specialize D according to AL*, Output D*.

C. Data Partition

The data partition is performed on the cloud. Here it collects the large no of data sets. It are split the large as D into small data sets as D_i, $1 \le i \le p$. Then provides the random number rand, where $1 \le rand \le p$ for each data sets. Partitioning is the process of determining which reducer instance will receive which intermediate keys and values. Each mapper must determine for all of its output (key, value) pairs which reducer will receive them. It is necessary that for any key, regardless of which mapper instance generated it, the destination partition is the same. If the key is generated in two separate (key, value) pairs, they must both be reduced together. It is also important for performance reasons that the mappers be able to partition data independently they should never need to exchange information with one another to determine the partition for a particular key.

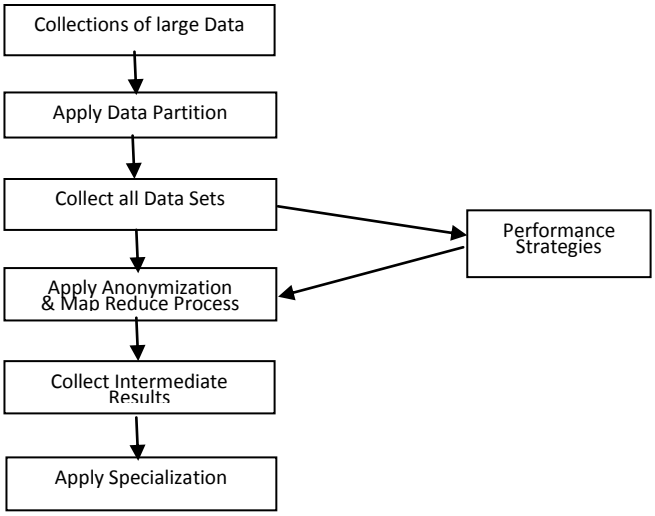
ALGORITHM 2. DATA PARTITION MAP &REDUCE.

Input: Data record (ID_r, r), r € D, partition parameter p.

Output: D_I, $1 \le i \le p$.

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

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



Data flow diagram for our approach

D. Anonymization Level Merging

Data anonymization is the process of masking sensitive data while preserving its format and data type. The masked data can be realistic or a random sequence of data. Output of anonymization can be deterministic, that is, the same value every time. Nature of output is dependent on the techniques used for anonymization. Anonymized data still look real in test environments and yields same results. In this process we remove or modify the identifying variables in micro data datasets .Identifying variables describes characteristics of person i.e. observable or registered. Direct identifiers, which are variables such as names, addresses, identity card numbers, social security numbers. They permit direct identification of an individual respondent, are not needed for statistical or research purposes. Thus they should be removed from the published dataset. Indirect identifiers are those characteristics that may be shared by several respondents, after combining these it may be possible to the re-identification of one of them. For example, the combination of attributes such as district of residence, age, sex, and occupation would be identifying if only one individual of that particular sex, age and occupation lived in that particular district. These variables are needed for statistical mining purposes, and should thus not be removed from the published data. Anonym zing the data will consist of recognizing which variables are potential identifiers and modifying the level of precision of these variables to reduce the risk of identification to an acceptable level. The key challenge is to maximize the security while minimizing the resulting data loss. Anonymization data can be placed on cloud without

Worrying about others will capture it .Afterwards it can be mapped to original data in secure and trusted area. Following are the anonymization techniques which will help us to provide security to data over cloud: k-anonymity.

K-Anonymity

Publishing data about individuals without revealing sensitive information about them is an important problem. In recent years, a new definition of privacy called k-anonymity has gained popularity. The goal is to make each record indistinguishable from a defined number (k) other records, if attempts are made to identify the record.

K-anonymity guarantees that each sensitive attribute is hidden in the scale of k groups. This means that the probability of recognizing the individual does not exceed 1/k. The level of privacy depends on the size of k. The statistical characteristics of the data are retained as much as possible; however, k-anonymity is not only applicable to sensitive data. An attacker could mount a consistency attack or background- knowledge attack to confirm a link between sensitive data and personal data. This would constitute a breach of privacy. The extensive study resolved some shortcomings of k-anonymity model as listed below.

1) It can't resist a kind of attack, which is assuming that the attacker has background knowledge to rule out some possible values in a sensitive attribute for the targeted victim. That is, k-anonymity does not guarantee privacy against attackers using background knowledge. It is also susceptible to homogeneity attack. An attacker can discover the values of sensitive attributes when there is little diversity in those sensitive attributes. Thus some stronger definitions of privacy are generated.

2) It protects identification information. However, it does not protect sensitive relationships in a data set.

3) Although the existing k-anonymity property protects against identity disclosure, it fails to protect against attribute disclosure.

4) It is suitable only for categorical sensitive attributes. However, if we apply them directly to numerical sensitive attributes (e.g., salary) may result in undesirable information leakage.

5) It does not take into account personal anonymity requirements and a k-anonymity table may lose considerable information from the micro data which is a valuable source of information for the allocation of public funds, medical research, and trend analysis.

E. Data Specialization

An original data set D is concretely specialized for anonymization in a one-pass MapReduce job. After obtaining the merged intermediate anonymization level ALI, we run MRTDS(D,k,AL) on the entire data set D, and get the final anonymization level AL*. Then, the data set D is anonymized by replacing original attribute values in D with the responding domain values in AL*. Details of Map and Reduce functions of the data specialization MapReduce job are described in Algorithm 3. The Map function emits

anonymous records and its count. The Reduce function simply aggregates these anonymous records and counts their number. An anonymous record and its count represent a QI-group. The QI-groups constitute the final anonymous data sets.

ALGORITHM 3.DATA SPECIALIZATION MAP & REDUCE.

Input: Data record (ID_r, r), r € D.; Anonymization `level AL*.
Output: Anonymous record (r*, count).
Map: Construct anonymous record r* =p_1, ‹p_2,...,p_m,sv›, p_i, 1 ≤ i ≤ 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 ←∑count; emit (r*, sum).

IV. MAPREDUCE VERSION OF CENTRALIZED TDS

MRTDS plays a core role in the two-phase TDS approach, as it is invoked in both phases to concretely conduct computation. Basically, a practical MapReduce program consists of Map and Reduce functions, and a driver that coordinates the macro execution of jobs.

A.MRTDS Driver

Usually, a single MapReduce job is inadequate to accomplish a complex task in many applications. Thus, a group of MapReduce jobs are orchestrated in a driver program to achieve 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. MRTDS produces the same anonymous data as the centralized TDS in [3], because they follow the same steps. MTRDS mainly differs from centralized TDS on calculating IGPL values. However, calculating IGPL values dominates the scalability of TDS approaches, as it requires TDS algorithms to count the statistical information of data sets iteratively. MRTDS exploits MapReduce on cloud to make the computation of IGPL parallel and scalable. We present IGPL Initialization and IGPL Update subsequently.

B. IGPL Initialization Job

The Map and Reduce functions of the job IGPL Initialization are described in Algorithms. The main task of IGPL Initialization is to initialize information gain and privacy loss of all specializations in the initial anonymization level AL. The statistical information is required for each specialization to calculate information gain. The number of records in each current QI-group needs computing, so does the number of records in each QI-group after potential specializations.

Algorithm IGPL INITIALIZATION MAP, describes the Map function. The input is data sets that consist of a number of records. ID_r is the sequence number of the record r. gets the potential specialization for the attribute values in r. Then emits key-value pairs containing the information of specialization, sensitive value, and the count information of this record. According to the above information, we compute information gain for a potential specialization in the corresponding Reduce function. Computing the current anonymity A_P (spec), while next Step is to compute anonymity A_c (spec) after potential specializations. The symbol "#" is used to identify whether a key is emitted to compute information gain or anonymity loss, while the symbol "$" is employed to differentiate the cases whether a key is for computing A_P (spec) or A_c (spec).

Algorithm IGPL INITIALIZATION REDUCE Specifies the Reduce function. The first step is to accumulate the values for each input key. If a key is for computing information gain, then the corresponding statistical information is updated. A salient MapReduce feature that intermediate key-value pairs are sorted in the shuffle phase makes the computation of IG(spec) sequential with respect to the order of specializations arriving at the same reducer. Hence, the reducer just needs to keep statistical information for one specialization at a time, which makes the reduce algorithm highly scalable.

To compute the anonymity of data sets before and after a specialization, finds the smallest number of records out of all current QI-groups, and finds all the smallest number of records out of all potential QI-groups for each specialization. Next step emits the results of anonymity. Note that there may be more than one key-value pair (spec, a (spec)) for one specialization in output files if more than one reducer is set. But we can find the smallest anonymity value in the driver program. Then, the privacy loss PL (spec) is computed. Finally, IGPL (spec) for each specialization.

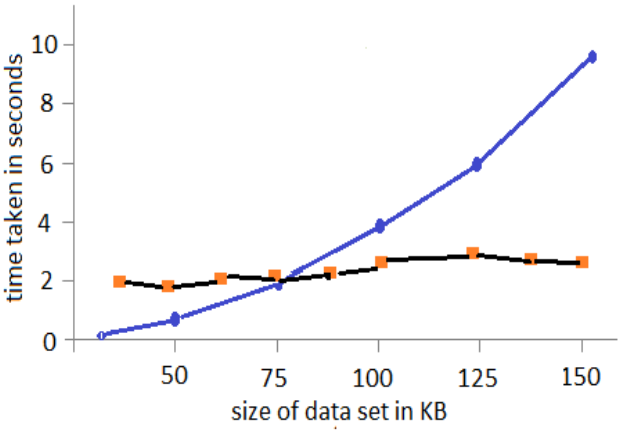
C. IGPL Update Job

The IGPL Update job dominates the scalability and efficiency of MRTDS, since it is executed iteratively as described in Algorithm MRTDS DRIVER. So far, iterative MapReduce jobs have not been well supported by standard MapReduce framework like Hadoop [7]. Accordingly, Hadoop variations like Hadoop [8] and Twister have been proposed recently to support efficient iterative MapReduce computation. Our approach is based on the standard MapReduce framework to facilitate the discussion herein. The IGPL Update job is quite similar to IGPL Initialization, except that it requires less computation and consumes less network bandwidth. Thus, the former is more efficient than the latter. Algorithm IGPL UPDATE MAP describes the Map function of IGPL Update. The Reduce function is the same as IGPL Initialization, already described in Algorithm DATA SPECIALIZATION MAP &REDUCE.

In algorithm IGPL UPDATE MAP, after a specialization spec is selected as the best candidate, it is required to compute the information gain for the new specializations derived from spec. So, Step in Algorithm IGPL UPDATE MAP only emits the key-value pairs for the new

specializations, rather than all in Algorithm IGPL INITIALIZATION MAP. Note that it is unnecessary to recompute the information gain of other specializations because conducting the selected specialization never affects the information gain of others. Compared with IGPL Initialization, only a part of data is processed and less network bandwidth is consumed. On the contrary, the anonymity values of other specializations will be influenced with high probability because splitting QI-groups according to spec changes the minimality of the smallest QI-group in last round. Therefore, we need to compute A_c (spec) for all specializations in AL. Yet A_P (spec) can be directly obtained from the Statistical information kept by the last best specialization. Note that if the specialization related to pi is not valid, no resultant quasi-identifier will be created. Since the IGPL Update job dominates the scalability and efficiency of MRTDS, we briefly analyze its complexity as follows. Let n denote all the records in a data set, m be the number of attributes, s be the number of mappers, and t be the number of reducers. As a mapper emits (m+1) key-value pairs, it takes O (1) space and O (m*n/s) time. Similarly, a

Fig.1. Execution framework overview of MRTDS



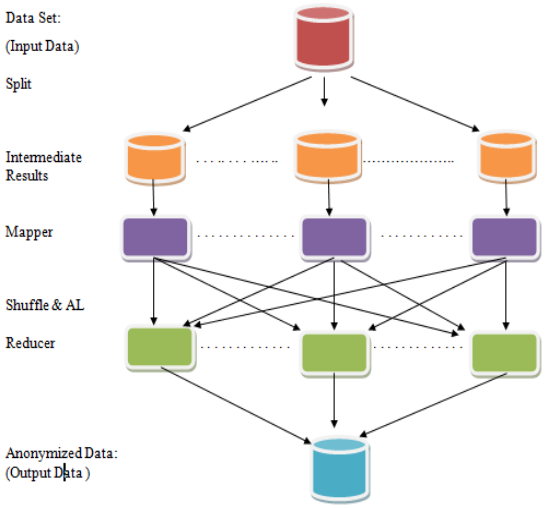
Reducer takes O (1) space and O (m*n/t) time. Note that a reducer only needs O (1) space due to the MapReduce feature that the key-value pairs are sorted in the shuffle phase. Otherwise, the reducer needs more space to accumulate statistic information for a variety of specializations.

V. EVALUATION

Overall Comparison

To evaluate the effectiveness and efficiency of our two-phase approach, we compare it with the centralized TDS approach proposed in [3], denoted as CentTDS. CentTDS is the state-of-the-art approach for TDS anonymization. Scalability and data utility are considered for the effectiveness. For scalability, we check whether both approaches can still work and scale over large-scale data sets. Data utility is measured by the metric I Loss, a general purpose data metric proposed. Literally, I Loss means information loss caused by data anonymization. Basically, higher I Loss indicates less data utility. How to calculate I Loss can be found in Appendix A.2, which is available in the online supplemental material. The I Loss of CentTDS and TPTDS are denoted as ILcent and ILTP, respectively. The execution time of CentTDS and TPTDS are denoted as TCent and TTP, respectively.

We roughly compare TPTDS and CentTDS as follows. TPTDS can scale over more computation nodes with the volume of data sets increasing, thereby gaining higher scalability. CentTDS will suffer from low scalability on large-scale data sets because it requires too much memory, while TPTDS can linearly scale over data sets of any size. Correspondingly, TTP is often less than TCent for large-scale data sets. But note that, TCent can be less than TTP due to extra overheads engendered by TPTDS when the scale of data sets or the MapReduce cluster is small. TPTDS is equivalent to MRTDS if parameter $p = 1$ or $k^l \geq k^{max}$, where k^{max} is the number of all records. As MRTDS produces the same anonymous data as centralized TDS, the



value of ILTP is equal to IL_{cent} when $p = 1$ or $k^l \geq k^{max}$. In other cases, ILTP is probably greater than IL_{cent}, as some specializations selected in TPTDS are not globally optimal.

Fig. 2. Change of execution time w.r.t. Data size: TPTDS versus CentTDS.

VI. CONCLUSION & FUTURE WORK

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. 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 creatively applied MapReduce on cloud to data anonymization and deliberately designed a group of innovative MapReduce jobs to concretely accomplish the specialization computation in a highly scalable way.

In cloud environment, the privacy preservation for data analysis, share and mining is a challenging research issue due to increasingly larger volumes of data sets, thereby requiring intensive investigation. Based on the contributions herein, we plan to further explore the next step on scalable privacy preservation aware analysis and scheduling on large-scale data sets.

REFERENCES

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

[2] B.C.M. Fung, K. Wang, R. Chen, and P.S. Yu, "Privacy-Preserving Data Publishing: A Survey of Recent Devil- opens," ACM Computing Surveys, vol. 42, no. 4, pp. 1-53, 2010.

[3] B.C.M. Fung, K. Wang, and P.S. Yu, "Anonymizing Classification Data For Privacy Preservation," IEEE Trans. Knowledge and Data Eng., vol. 19, No. 5, pp. 711-725, May 2007.

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

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

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

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

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

BIOGRAPHIES:

Karthik Singh. M received Bachelor's degree in Computer Science from SPE College affiliated to jntu Anantapur, India. He is currently working towards Master's degree at Audisankara College of Engineering affiliated to jntu Anantapur, India. His research interest's includes Cloud Computing, Privacy and security.

Mr. Srinivasulu Yaddala, M. Tech, MISTE Assistant Professor, Department of Computer Science, Audisankara College of Engineering Gudur. Received B. Tech in Computer Science and Engineering from V.R. SIDDARTHA College of Engineering affiliated to Acharya Nagarjuna University. Received M. Tech degree in Computer Science and Engineering from NIST affiliated to JNTU Kakinada. Having 09 years of teaching experience. Interest in areas are Cloud Computing, Formal Methods.