IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

Data Anonymization Technique for Privacy Preservation Using MapReduce Framework

Nagajothi .S[1], Raj Kumar .N[2]

II M.E.-CSE, SVS College of Engineering, Coimbatore [1]

Assistant Professor (CSE) M.E (PhD), SVS College of Engineering, Coimbatore[2]

Abstract: Data anonymization is widely adopted for data privacy preservation in non interactive data publishing and sharing scenarios. It refers to hiding identity and/or sensitive data for owners of data records. Sharing the private data record in its most specific state poses a threat to individual privacy. This privacy of an individual can be effectively preserved while certain aggregate information is exposed to data users for diverse analysis and mining. This is mainly to investigate the scalability problem of large-scale data anonymization. Data sets are generalized in a top-down manner until k-anonymity is violated in order to expose the maximum utility. This Top-Down Specialization is efficient for high scalability and privacy concerns. High scalable two-phase top-down approach to anonymize large-scale data using map reduce is proposed.

Keywords: Anonymization, Generalization, Top-Down Specialization, MapReduce algorithm, K-anonymity, Big Data.

1. INTRODUCTION

Cloud computing is regarded as an ingenious combination of a series of technologies, establishing a novel business model by offering IT services and using economies of scale. Participants in the business chain of cloud computing can benefit from this novel model. Many companies or organizations have been migrating or building their business into cloud due to security and privacy concerns[1][2]. Personal data like electronic health records and financial transaction records are usually deemed extremely sensitive although these data can offer significant human benefits if they are analyzed and mined by organizationssuch as disease research centres. For instance, Microsoft HealthVault, an online cloud health service, aggregates data from users and shares the data with research institutes. 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[1]. Data sets have become so large thatanonymizing such data sets is becoming a considerable challenge for traditional anonymization algorithms to investigate the scalability problem of large-scale data anonymization. Large-scale data processing frameworks like MapReduce have been integrated with cloud to provide powerful computation capability for applications.

In our research, we leverage MapReduce, awidely adopted parallel data processing framework, to address the scalability problem of the top-down specialization (TDS) approach for large-scale data anonymization. The TDS approach offers a good tradeoff betweendata utility and data consistency is widely applied for data anonymization. Most TDS algorithms are centralized which results in their inadequacy in handling largescaledata sets. Although some distributed algorithms have been proposed[20][22].They mainly focus on secure anonymization of data sets from multiple parties, rather than the scalability aspect. As the MapReduce computation paradigm is relatively simple it is still a challenge to design proper MapReduce jobs for TDS.

In this paper, we propose a highly scalable two-phaseTDS 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 the second one, the intermediate results are integrated into one, and further anonymized to achieve consistent k-anonymous[23] data sets.

A group of Map Reduce jobs is deliberately designed and coordinated to perform specializations on data sets collaboratively. Existing technical approaches for preserving the privacyof data sets stored in cloud mainly include encryption and anonymization. Current privacy-preserving techniques like generalization can withstand most privacy attacks on one single data set, while preserving privacy for multipledata sets is still a challenging problem. Thus, for preserving privacy of multiple data sets, it is promising to anonymize all data sets first and then encrypt them before storing or sharing them in cloud. Usually, the volume of intermediate data sets is huge. Hence, we argue that encrypting all intermediate data sets will lead to high overhead and low efficiency when they are frequently accessed or processed. As such, we propose to encrypt part of intermediate data sets rather than all for reducing privacy-preserving cost[2]. A tree structure is modelled fromgeneration relationships of intermediate data sets to analyse privacy propagation of data sets. Based on such a constraint, we modelthe problem of saving privacy-preserving cost as a constrained optimization problem. This problem is then divided into

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

series ofsub-problems by decomposing privacy leakage constraints. The remainder of this paper is organized as follows: The next section reviews related work, and analyzes the scalability problem in existing TDS algorithms. In Section 3, we briefly present two-phase TDS approach. Section 4formulates Top-Down Specialization and elaborates algorithmic details of MapReduce jobs.Section 5 is implementation of this approach. We empirically evaluate our experimental results in Section 6. Finally, we conclude this paper and discuss future work in Section 7.

2. RELATEDWORK AND PROBLEM ANALYSIS

2.1 RELATED WORK

In this work[1],Two-Phase Top-Down Specialization (TPTDS) approach to conduct the computation required in TDS in a highly scalable and efficient fashion. MapReduce is a programming model for processing large data sets with a parallel and distributed algorithm on a cluster. A MapReduce program is composed of a Map() procedure that performs filtering and sorting (such as sorting students by first name into queues, one queue for each name) and a Reduce() procedure that performs a summary operation (such as counting the number of students in each queue, yielding name frequencies). The MapReduce System is orchestrates by marshalling the distributed servers while running the various tasks in parallel and managing all communications and data transfers between the various parts of the system. It providesredundancy and fault tolerance for overall management of the whole process.

In the work[2], Many anonymization techniques likegeneralization have been proposed to preserve privacy but these methods alone fail to solve the problem of preserving privacy for multiple data sets. The main approachintegrates anonymization with encryption to achieve privacy preserving of multiple data sets. Moreover they consider the economical aspect of privacy preserving adhering to the pay-as-you-go feature of cloud computing.

In the work[3], Distribute anonymization and centralizedAnonymizationuses with healthcare data has become avital requirement in healthcare system management. However, inappropriate sharing and usage of healthcaredata could threaten patients privacy. It generalize their information and privacy requirements to the problems of centralized anonymization and distributed anonymization and identify the major challenges that make traditional data anonymization methods not applicable. Furthermore it proposes a new privacy model called LKC-privacy to overcome the challenges and present two anonymization algorithms to achieve LKC-privacy in both the centralized and the distributed scenarios. Experiments on real-life data demonstrate that the anonymization algorithms can effectively retain the essential information in anonymous data for data analysis and is scalable for anonymizing large datasets. Handling of the large scale data sets are very difficult. The distribute anonymization and centralized anonymization provides the privacy on cloud by Handling of the large scale data sets are very difficult.

2.2 PROBLEM ANALYSIS

Two- Phase Top- Down Specialization (TPTDS) approach to conduct the computation required in TDS in a highly scalable and efficient fashion. The two phases of the approach are based on the two levels of parallelization provisioned by MapReduce on cloud. Basically, MapReduce on cloud has two levels of parallelization, i.e., job level and task level. Job level parallelization means that multiple MapReduce jobs can be executed simultaneously to make full use of cloud infrastructure resources. Combining with cloud, MapReduce becomes more powerful and elastic as cloud can offer infrastructure resources on demand.For instance: Amazon Elastic MapReduce service. Task level parallelization refers to that multiple mapper/reducer tasks in a MapReduce job are executed simultaneously over data splits. To achieve high scalability and parallelizing multiple jobs on data partitions in the first phase 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 anonymize entire data sets. All intermediate anonymization levels are merged into one in the second phase. There exist domain values and that satisfy one of the three conditions is identical to is moregeneral than is more specific than. To ensure that the merged intermediate anonymization level never violates privacy requirements.

The major contributions of our research are threefold. First, we formally demonstrate the possibility of ensuring privacy leakage requirements without encrypting all intermediate data sets when encryption is incorporated with anonymization to preserve privacy. Second, we design a practical heuristic algorithm to identify which data sets need to be encrypted for preserving privacy while the rest of them do not. Third, experiment results demonstrate that our approach can significantly reduce privacy-preserving cost over existing approaches, which is quite beneficial for the cloud users who utilize cloud services in a pay-as-you-go fashion. This paper is a significantly improved version of [1][2]. Based on[1], we mathematically prove that our approach can ensure privacy-preserving requirements. Further, the heuristic algorithm is redesigned by considering more factors. We extend experiments over real data sets. Our approach is also extended to a graph structure.

In the literature of k-anonymity problem, there are two main models. One model is global recoding while the other is local recoding . Here, we assume that each attribute has a corresponding conceptual generalization hierarchy or taxonomy tree. A lowerlevel domain in the hierarchy provides more details than a higher level domain. For example, Zip Code 14248 is a lower level domain and Zip Code 142∗∗ is a higher level domain.
We assume such hierarchies for numerical attributes too. In particular, we have a hierarchical structure defined with {value, interval, ∗}, where value is the raw numerical data,

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

interval is the range of the raw data and ∗ is a symbol representing any values. Generalization replaces lower level domain values with higher level domain values. For example, Age 27, 28 in the lower level can be replaced by the interval (27-28) in the higher level.

3. CENTRALIZED AND DISTRIBUTED ANONYMIZATION

The centralized anonymization method can be viewed as "integrate-then generalize" approach where the central government health agency first integrates the data from different hospitals then performs generalization. In this approach the distributed anonymization problem has two major challenges in addition to high dimensionality. First, the data utility of the anonymous integrated data should be as good as the data quality produced by the centralized anonymization algorithm. Second, in the process of anonymization, the algorithm should not reveal more specific information than the final anonymous integrated table.
In this, we propose a top-down specialization algorithm to achieve L KC-privacy. The general idea is to anonymize a table by a sequence of specializations starting from the topmost general state in which each attribute has the topmost value of its taxonomy tree. We assume that a taxonomy tree is specified for each categorical attribute in Q ID. A leaf node represents a domain value and a parent node represents a less specific value.

For a numerical attribute in Q ID, a taxonomy tree can be grown at runtime, where each node represents an interval, and each non-leaf node has two child nodes representing some optimal binary split of the parent interval. A specialization, written v → child(v), where child(v) denotes the set of child values of v, replaces the parent value v with the child value that generalizes the domain value in a record. A specialization is valid if the specialization results in a table satisfying the anonymity requirement after the specialization. A specialization is performed only if it is valid. The specialization process can be viewed as pushing the "cut" of each taxonomy tree downwards. A cut of the taxonomy tree for an attribute Di, denoted by Cuti, contains exactly one value on each root-to-leaf path. Our specialization starts from the topmost cut and pushes down the cut iteratively by specializing some value in the current cut until violating the anonymity requirement. In other words, the specialization process pushes the cut downwards until no valid specialization is possible. Each specialization tends to increase data utility and decrease privacy because records are more distinguishable by specific values. We define two utility measures depending on the information requirement to evaluate the "goodness" of a specialization. Here, we assume that BTS only receives one version of the sanitized data for a given dataset anonymized by using one of the following Score functions.

Case 1: Score for Classification Analysis. For the requirement of classification analysis, we use information gain, denoted by InfoGain(v), to measure the goodness of a specialization. Our selection criterion, Score(v), is to favor the specialization v → child(v) that has the maximum InfoGain(v):
Score(v) = InfoGain(v).
InfoGain(v): Let T[x] denote the set of records in T generalized to the value x. Let f req(T[x], cls) denote the number of records in T[x] having the class cls Note that |T[v]| = c |T[c]|, where c ∈ child(v). We have
InfoGain(v) = E(T[v]) − c |T[c] / |T[v]| |E(T[c]),
where E(T[x]) is the entropy of T[x]
E(T[x]) = −clsfreq(T[x], cls)
freq(T[x], cls) × log2
|T[x] || T[x]|

Intuitively, I(T[x]) measures the mix of classes for the records in T[x], and InfoGain(v) is the reduction of the mix by specializing v into c ∈ child(v). For a numerical attribute, the specialization of an interval refers to the optimal binary split that maximizes information gain on the Class attribute.

Case 2: Score for General Data Analysis. Sometimes, the data is shared without a specific task. In this case of general data analysis, we use discernibility cost to measure the data distortion in the anonymous data table. The discernibility cost charges a penalty to each record for being indistinguishable from other records. For each record in an equivalence group qid, the penalty is |T[qid]|. Thus, the penalty on a group is |T[qid]|. To minimize the discernibility cost, we choose the specialization v → child(v) that maximizes the value of |T[qidv]|2 Score(v) = qidv over all qidv containing v.

4. TOP-DOWN SPECIALIZATION

In Top-Down Specialization all the attribute values are initialized to the root value of the hierarchy tree. The specialization is carried out iteratively over the attribute values until the k -anonymity is violated. The specialization is performed by replacing parent attribute value by its child value in Taxonomy Tree.

ALGORITHM: 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<i <p.
2. Execute MRTDS(D_i , k^I , AL^0)→AL^0_I , 1 <i < p in parallel as multiple MapReduce jobs.
3. Merge all intermediate anonymization levels into one, merge (AL'_I ,AL'_2,…,AL'_p)→AL^I.
4. Execute MRTDS (D,k,AL^I)→AL* to achieve k-anonymity.
5. Specialize D according to AL*, Output D*.

4.1 OPTIMIZED BALANCING SCHEDULING

The OBS called optimized balancing scheduling. Scheduling map tasks to improve data locality is crucial to the performance of MapReduce. Many works have been

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

devoted to increasing data locality for better efficiency. However, to the best of our knowledge, fundamental limits of MapReduce computing clusters with data locality, including the capacity region and theoretical bounds on the delay performance have not been studied. In this paper it addresses these problems from a stochastic network perspective.

4.2 DATA PARTITION

The data partition is performed on the cloud. Here it collects the large no of data sets. It are split the large into small data sets. Then provides the random no 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 cat 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. It is necessary that for any key, regardless of which mapper instance generated it, the destination partition is the same. If the key cat 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: 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)).
Once partitioned data sets D_i, $1 \le i \le p$, are obtained, we run MRTDS (D_i,k_I ,AL^0) on these data sets in parallel to derive intermediate anonymization levels AL^*_I ,$1 \le i \le p$.

4.3 MERGING

All intermediate anonymization levels are merged into onein the second phase. The merging of anonymization levels is completed by merging cuts. Cut_a in AL'_aandCut_b in AL'_bbe two cuts of an attribute. There exist domain values q_a€Cut_a and q_b€ Cut_b that satisfy one of the three conditions: q_a is identical to q_b, q_a is more general than q_b, or q_a is more specific than q_b. To ensure that the merged intermediate anonymization level AL_I never violates privacy requirements, the more general one is selected as the merged one, for example, q_a will be selected if $q_a$is more general than or identical to q_b. For the case of multiple anonymization levels, we can merge them in the same way iteratively. The following lemma ensures that AL_I still complies privacy requirements.

Lemma 1. If intermediate anonymization levels AL'_i, $1 \le i \le p$, satisfy k^Ianonymity, the merged intermediate anonymization level AL^I will satisfies k'-anonymity, where
$AL^I \leftarrow merge((AL'_1,AL'_{2,\ldots,}AL'_p))$, $k^J \ge k^I$.

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

4.4 BIG DATA ANALYTICS SPECIALIZATION

Big Data Analytics specialization will prepare students to address real-life problems along each of those dimensions. For instance, it is not uncommon for digital archives to store terabytes and even petabyte of data in hundreds of data repositories supporting thousands of applications. Maintaining such data repositories requires knowledge in ultra-large scale distributed systems, virtualization technologies, cloud computing, unstructured and semi-structured data management, optimization methods based on data replication and data migration, as well as in advanced data protection techniques. The exponential growth of the amount of data calls for competence in advanced dynamic data processing techniques includes scalable data processing methods and technologies, data stream management and large-scale process monitoring, modeling and mining. In order to comprehensively analyze such volumes of information from disparate and various disciplines, information professionals will need to master advanced data integration techniques and business intelligence tools, crowd sourcing technologies, large-scale information fusion, data intensive computation and semantic data management. After gets the intermediate result those results are merged into one. Again applies the anonymization on the merged data is called specialization.

ALGORITHM: DATA SPECIALIZATION MAP & REDUCE.

Input: Data record (ID_r, r), r € D. ; Anonymizationlevel AL*.
Output: Anonymous record (r*, count).
Map: Construct anonymous record r* = p_1, (p_2,….,p_m,sv), p_i, $1 \le i \le m$, 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).

4.5 ANONYMIZATION

Anonymization of data can mitigate privacy and security concerns and comply with legal requirements. Anonymization is not invulnerable countermeasures that compromise Current anonymization techniques can expose protected information in released datasets. After gets the individual data sets it applies the anonymization. The anonymization means hide or remove the sensitive field in data sets. Then it gets the intermediate result for the small data sets. The intermediate results are used for the

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

specialization process. Data anonymization algorithm that converts clear text data into a nonhuman readable and irreversible form including but not limited to pre-image resistant hashes and encryption techniques in which the decryption key has been discarded.

4.5.1 ANONYMIZATION ALGORITHM

DA(D,I,k,m)
1. scan D and create count-tree
2. initializeCout
3. for each node v in preorder count-tree tranversal do
4. if the item of v has been generalized in Cout then
5. backtrack
6. if v is a leaf node and v.count<k then
7. J:=itemset corresponding to v
8. find generalization of items in J that make J k-anonymous
9. merge generalization rules with Cout
10. backtrack to longest prefix of path J,wherein noitem has been generalized in Cout
11. ReturnCout
12. for i :=1 to Cout do
13. initialize count=0
14. scan each transactions in Cout
15. Seperate each item in a transaction and store it in p
16. Increment count
17. for j:=1 to count do
18. for all g belongs Cout do
19. compare each item of p with that of Cout
20. if all items of i equal to cout
21. Increment the r
22. Ifka equal to r then backtrack to i
23. else if r greater than ka then get the index positionof the similar transactions
24. make them NULL until ka equal to r
25. else update the transactions in database

5. IMPLEMENTATION

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 the 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. 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.

As an example, table 1 is to be anonymized with Anonymization Level (AL) set to 2 and the set of Quasi Identifiers as QI = {AGE, SEX, ZIP, PHONE}. The quasi-identifiers are identified by the organization according to their rules and regulations.

Table 1: Anonymization level

Name	Age	Sex	Zip	Phone	Disease
Ali	20	M	190014	9419	Bronchitis
Bale	30	M	190001	9592	Lung Cancer
Calvin	40	M	192231	9823	STI
Doris	50	F	190001	8988	Skin Allergy
Elle	75	F	190002	8088	Skin Allergy

The NAME attribute here is "Sensitive", so we would like to “suppress” this attribute before anonymizing the above table. After suppression the table 2 will look like as below

Table 2: AL after suppression

AGE	SEX	ZIP	PHONE	DISEASE
20	M	190014	9419	Bronchitis
30	M	190001	9592	LungCancer
40	M	192231	9823	STI

Anonymizing data through Top-Down Specialization each attribute value will be initialized to the root of Taxonomy Tree in table 3 will look like as below

Table 3: Root of taxonomy tree

AGE	SEX	ZIP	PHONE	DISEASE
[0 - 100]	ANY	******	****	Bronchitis
[0- 100]	ANY	******	****	Lung Cancer
[0 - 100]	ANY	******	****	STI
[0 - 100]	ANY	******	****	Skin Allergy
[0 - 100]	ANY	******	****	Skin Allergy

The data in the above table is highly privacy preserved, but the data utility is very low. The data is highly anonymized. We make a note here that Data Anonymization is not only the single goal that we are trying to achieve through Anonymization. We also make sure that data utility is high enough to make the information useful for mining.
The Top-Down Specialization Algorithm will iteratively specialize the attribute values till the k Anonymization is violated.The given table after anonymizing it for k=2 in table 4 will look like

Table 4: Anonymized dataset

AGE	SEX	ZIP	PHONE	DISEASE
[0 - 50]	M	1900**	9***	Bronchitis
[26-50]	M	190001	9***	LungCancer
[26-50]	M	19*****	9***	STI
[26-50]	F	190001	8***	SkinAllergy
[51-100]	F	19000*	8***	SkinAllergy

DOI 10.17148/IJARCCE.2016.55247

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

6. EXPERIMENTAL RESULTS

Three groups of experiments in this section evaluate the effectiveness and efficiency. In the first one, TPTDS with CentTDSsays the perspectives of scalability and efficiency. In the other two, the trade-off between scalability and data utility via adjusting configurations. The execution time and ILoss are affected by three factors, namely the size of a data set (S), the number of data partitions (p), and the intermediate anonymity parameter (k^I). How the three factors influence the execution time and ILoss of TPTDS is observed in the following experiments in Fig 1.

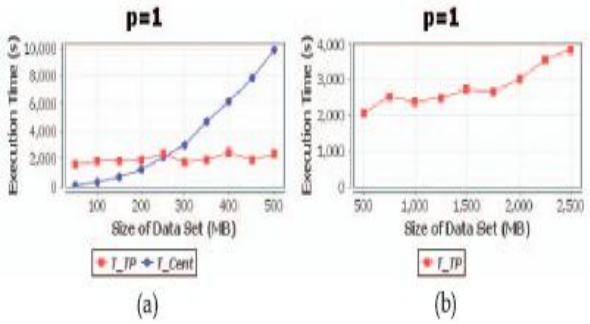

Fig 1: Change of execution time w.r.t. data size: TPTDS versus CentTDS.

In the first group, it measure the change of execution time TCent and TTP with respect to S when p = 1. The size S varies from 50 MB to 2.5 GB. The 2.5 GB data set contains nearly 2:5 * 107 data records. The data sets are big enough to evaluate the effectiveness of approach in terms of data volume or the number of data records. $IL_{Cent} = IL_{TP}$ because TPTDS is equivalent to MRTDS when p = 1.The results of the first group of experiments are depicted in Fig 6.9 shows the change of TTP and TCent with respect to the data size ranging from 50 to 500 MB. From Fig 2 can see that both TTP and TCent go up when data size increases although some slight fluctuations exist.

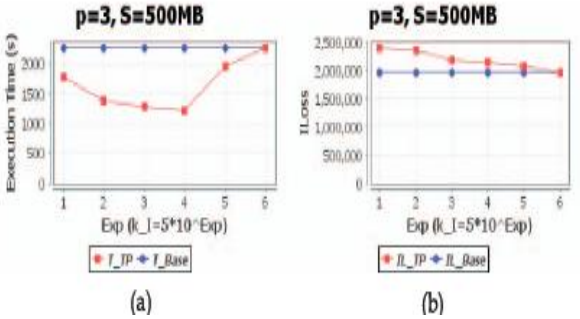

Fig 2: Change of execution time and ILoss w.r.t. intermediate anonymity parameter.

The fluctuations are mainly caused by the content of data sets. TCent surges from tens of seconds to nearly 10,000 seconds, while TTP increase slightly. The dramatic increase of TCent illustrates that the overheads incurred by maintaining linkage structure and updating statistic information rise considerably when data size increases. Before the point S = 250 MB, TTP is greater than TCent. But after the point TTP is greater than TCent, and the difference between TCent and TTP becomes larger and larger with the size of data sets increasing. The trend of TTP and TCent indicates in Fig 3 that TPTDS becomes more efficient compared with CentTDS for largescale data sets.

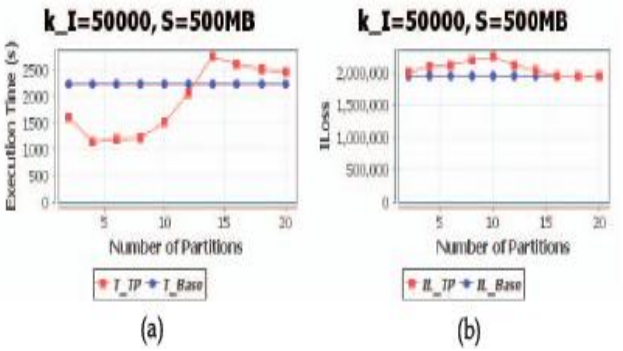

Fig 3: Change in execution time and ILoss w.r.t. number of partitions

In our experiments, CentTDS fails due to insufficientmemory when the size of data set is greater than 500 MB. Hence, CentTDS suffers from scalability problem for largescale data sets. To further evaluate the scalability andefficiency of TPTDS, we run TPTDS over data sets withlarger sizes. Fig. 6.9 shows the change of TTP with respect to the data size ranging from 500 MB to 2.5 GB. It can be seen from Fig 6.9 that TTP grows linearly and stably with respect to the size of data sets. Based on the tendency of TTP, we maintain that TPTDS is capable of scaling over large-scale data sets efficiently.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have investigated the scalability problem of large-scale data anonymization but Top-Down Specialization approach using MapReduce on cloud. Datasets are partitioned and anonymized in parallel and intermediate results are merged and anonymized to produce consistent k-anonymous data sets.

In cloud environment, the privacy preservation for data analysis, sharing and mining is a challenging due to larger volume of data sets. It is scalable privacy preservation aware analysis ad scheduling and optimized balanced scheduling strategies toward overall scalable privacy preservation.

REFERENCES

[1] Xuyun Zhang, Laurence T. Yang, Senior Member, IEEE, Chang Liu, and Jinjun Chen, Member, IEEE, "A Scalable Two-Phase Top-Down Specialization Approach for Data Anonymization Using MapReduce on Cloud", IEEE transactions on parallel and distributed systems, vol. 25, no. 2, february 2014.

[2] X. Zhang, C. Liu, S. Nepal, S. Pandey, and J. Chen, "A Privacy Leakage 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.

[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] Roy, S.T.V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel, "Airavat: Security and Privacy for Mapreduce," Proc. Seventh USENIX Conf. Networked Systems Design and Implementation (NSDI '10), pp. 297-312, 2010.
[5] 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.
[6] S. Chaudhuri, "What Next?: A Half-Dozen Data Management Research Goals for Big Data and the Cloud," Proc. 31st Symp. Principles of Database Systems (PODS '12), pp. 1-4, 2012.
[7] 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.
[8] L. Wang, J. Zhan, W. Shi, and Y. Liang, "In Cloud, Can Scientific Communities Benefit from the Economies of Scale?," IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 2, pp.296-303, Feb. 2012.
[9] 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, pp. 24-31, Nov. 2010.
[10] D. Zissis and D. Lekkas, "Addressing Cloud Computing Security Issues," Future Generation Computer Systems, vol. 28, no. 3, pp. 583- 592, 2011.
[11] L. Hsiao-Ying and W.G. Tzeng, "A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding," IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 6, pp. 995-1003, 2012.
[12] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data," Proc. IEEE INFOCOM, pp. 829-837, 2011.
[13] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, "Gupt: Privacy Preserving Data Analysis Made Easy," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '12), pp. 349- 360, 2012.
[14] B.C.M. Fung, K. Wang, R. Chen, and P.S. Yu, "Privacy-Preserving Data Publishing: A Survey of Recent Devel- opments," ACM Computing Surveys, vol. 42, no. 4, pp. 1-53, 2010.
[15] 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.
[16] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Incognito: Efficient Full-Domain K-Anonymity," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '05), pp. 49-60, 2005.
[17] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional K-Anonymity," Proc. 22nd Int'l Conf. Data Eng. (ICDE '06), 2006.

DOI 10.17148/IJARCCE.2016.55247