LITERATURE REVIEW: Parallel Anonymization using MapReduce on Apache Spark^{TM}

Macarious Abadeer
School of Computer Science
Carleton University
Ottawa, Canada K1S 5B6
macariousabadeer@cmail.carleton.ca

September 29, 2019

1 Introduction

Since the introduction of multi-core processors in 2004 by Intel®, parallel computing evolved to exploit the advantages of multiple processing units that became the norm for personal computers. This evolution was also expanded and accelerated by the advancements in Cloud Computing that supported running compute-intensive applications over a network of clusters. Parallel computing enabled the development of solutions to different real world applications that were hindered by scalability limitations such as big data analytics, machine learning and artificial intelligence. One of the problems that parallel computing provided scaleable solutions for is data anonymization - especially for big data.

In today's abundance of big data ranging from retail and banking transactions, health care, social media interactions and sensor data, a need was created for measures that protects people's most private and sensitive data. One of the most popular theories that were developed in this area was k-anonymity developed by Samarati and Sweeney in 1998 [9]. Sweeney argued that an individual in a dataset can be identified when it is linked with other public datasets even if the original dataset did not contain identifying information such as name, date of birth and social insurance number. He was able to show that when linking voter registration cards and health care data, individuals can be identified with 87% accuracy. Those potentially identifying attributes are called Quasi-Identifiers (QID). k-anonymity states that a dataset is called k-anonymous when for a given record, there exists at least k$-$1 records in the same dataset with the same QID values. Further modifications to k-anonymity were made to overcome its shortcomings such as introducing ℓ-diversity [6] and t-closeness [5]. ℓ-diversity ensures that sensitive attributes, such as diagnosis in a health care dataset, need to have diverse values so that an adversary who knows the values of a given QIDs cannot deduce their diagnosis. t-closeness ensures that the distribution of these diverse values is close to their distribution in the original dataset.

While these theories contributed immensely to the practices of data anonymization, k-anonymity remains NP-complete [10]. Further research used these models as a baseline in order to develop scalable parallel algorithms that can handle big data. In the next section I will go over the different ideas that were proposed to optimize and scale k-anonymity.

2 Literature Review

There are three different masking types that are used in order to satisfy k-anonymity: interval, taxonomy tree and suppression [1]. Suppression requires certain outlier tuples to be removed in order to satisfy k-anonymity [9]. Intervals are taxonomy trees are generalization techniques applied to numerical and categorical attributes respectively [9]. For example two records with birth year of 1971 and 1973 can be generalized to 1970-1975. For a taxonomy tree, a categorical attribute such as education level, a post-graduate node in a taxonomy tree can have PhD, Masters and Post Graduate Diploma as its child leaves so that records with these values can be generalized to the parent node. The majority of research papers on anonymization in the context of big data involved taxonomy trees thus this is where I focus my literature survey.

One of the techniques that researchers attempted to optimize was Bottom-Up Generalization (BUG) which involves traversing the taxonomy tree of attribute hierarchies from the bottom (most specific) upwards (most general) [4]. Wang suggested that the taxonomy tree would be provided by the data supplier or the data recipient [4]. As the tree is traversed, two metrics are calculated to ensure a high quality generalization: information loss per anonymity gain. An indexed approach to bottom-up generalization was proposed by Hoang [3] where the taxonomy tree was generated automatically at runtime. Hoang's indexed approach could also handle numerical as well as categorical attributes. Indexed BUG starts with collecting statistical information about the dataset as well as partition it to be used in the generalization step which was further broken down to four steps: calculate the best generalization score based on the minimum information loss, calculate k-anonymity for every partition, generate an indexed generalization map which maps every value to its generalized value, and the last step creates the anonymized dataset using this map. Hoang's experiments showed that the generalization time did not increase with the dataset size due to the use of indexed generalization map however performance was impacted by the distinct values count for each QID.

Parallel BUG was introduced to address the limitations of traditional and indexed BUG approaches. Pandilakshmi attempted to solve the limitations of indexing structures since they are centralized and hard to parallelize or run on distributed systems such as the Cloud [7]. Pandilakshmi introduced Bi-Level BUG algorithm where MapReduce framework was used to take advantage of job-level and task-level parallelization. Job-level parallelization was achieved by using multiple MapReduce jobs and task-level parallelization was achieved by using multiple map-reducer tasks for every MapReduce job so that they are executed in parallel on every partition. Data is partitioned according to a random number generated between 1 and p where p is the number of partitions. Pandilakshmi then runs MapReduce BUG driver (MRBUG) iteratively on the partitioned datasets and calculates generalization score (least information loss with the most anonymity gain) and stops until it finds the best generalization with the highest score that satisfies k-anonymity. Pandilakshmi experiments performed on varying datasets up to 4GB showed that execution time was virtually capped at 33 minutes regardless of dataset size.

Another technique was Top-Down Specialization (TDS). TDS traverses the taxonomy tree from the top downwards where it starts with the most generalized values and specializes the value and stops when it violates k-anonymity [2]. Multiple solutions have been developed such as a scalable two-phase TDS introduced by [8] and [12]. The first phase involves partitioning the original dataset to p partitions using random sampling. A MapReduce TDS job runs in parallel on each partition. Each job specializes the data iteratively while calculating information gain and privacy loss metrics and creates an intermediate anonymized dataset. In the second phase the intermediate datasets are merged and further anonymized if necessary to satisfy k-anonymity. In [12], Zhang et al. adopted Hadoop® and took advantage of distributed cache capability to pass the intermediate anonymized dataset to each mapper/reducer node. The experiments for this solution showed the overheard in the partitioning phase of the dataset.

A hybrid approach of BUG and TDS using MapReduce was introduced by [11] where it was shown that when either TDS or BUG were used individually, they performed poorly for certain values of k. The hybrid approach applies TDS for large k values and BUG for smaller ones. The notion of Workload Balancing Point is introduced which is defined as the point where the amount of computation required for TDS is the same as BUG. Once that point is identified, the hybrid approach chooses TDS for k greater than the workload balancing point and chooses BUG when k is smaller. The workload balancing point is estimated using the height of the taxonomy tree as a reference.

A multi-dimensional sensitivity-based algorithm was developed by

References

[1] Ruan Chun Al-Zobbi Mohammed, Shahrestani Seyed. Experimenting sensitivity-based anonymization framework in apache spark. *Journal of Big Data*, 5(1):38, October 2018.

[2] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *21st International Conference on Data Engineering (ICDE'05)*, pages 205–216, April 2005.

[3] A. Hoang, M. Tran, A. Duong, and I. Echizen. An indexed bottom-up approach for publishing anonymized data. In *2012 Eighth International Conference on Computational Intelligence and Security*, pages 641–645, November 2012.

[4] Ke Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: a data mining solution to privacy protection. In *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 249–256, November 2004.

[5] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115, April 2007.

[6] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.

[7] K Pandilakshmi and G Rashitha Banu. An advanced bottom up generalization approach for big data on cloud. *Int J Comput Algor*, 3:1054–9, 2014.

[8] Zorige Priyanka, K Nagaraju, and Y Venkateswarlu. Data anonymization using map reduce on cloud based a scalable two-phase top-down specialization. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(12):3879–3883, 2014.

[9] Sweeney L. Samarati P. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and supression. Technical report, Massachusetts Institute of Technology and SRI International, 1998.

[10] U. Sopaoglu and O. Abul. A top-down k-anonymization implementation for apache spark. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 4513–4521, December 2017.

[11] X. Zhang, C. Liu, S. Nepal, C. Yang, W. Dou, and J. Chen. Combining top-down and bottom-up: Scalable sub-tree anonymization over big data using mapreduce on cloud. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pages 501–508, July 2013.

[12] X. Zhang, L. T. Yang, C. Liu, and J. Chen. A scalable two-phase top-down specialization approach for data anonymization using mapreduce on cloud. *IEEE Transactions on Parallel and Distributed Systems*, 25(2):363–373, February 2014.