International Journal On Engineering Technology and Sciences – IJETS™
ISSN(P): 2349-3968, ISSN (O): 2349-3976
Volume III, Issue VII, July- 2016

PRIVACY PRESERVED FREQUENT ITEMSET MINING FOR SERVICE RECOMMENDATIONS UNDER CLOUDS

Ms. Leelavathi. G. MCA, Research Scholar,
Ms. Mohana Priya. C. MSc (CT), MPhil, Asst. Professor,
Department of Computer Applications,
Tirupur Kumaran College For Women, Tirupur, Tamilnadu, India

Abstract

Resources and services are shared through the Internet under the cloud computing model. User requirement based mechanism is adapted to provide the services and resources. Big data applications are building to manage huge amount of data values. MapReduce techniques are applied to divide and process the data and tasks. Hadoop is a widely-adopted distributed computing platform using the MapReduce parallel processing paradigm. Service details and user access information are maintained under the cloud for service discovery process. Service access information are maintained for the future references to assess the performance levels. Service recommendation tools are employed to discover appropriate services for the users. Big data applications are constructed to manage customer, services and access information for the service recommendation process. Scalability and inefficiency are the main problems identified under the traditional service discovery process. The service recommender systems use the same rating and ranking model for the discovery process.

Frequent itemset mining methods are employed to discover the frequently repeated items. Privacy preserved mining models are adapted to protect the sensitive attributes. The KASR method is mainly focused to perform personalized service recommendation process with user preferences. Frequent itemsets are used to discover the service selection process. Semantic query based service discovery process is supported by the KASR scheme. KASR is implemented on Hadoop to improve the scalability and efficiency on big data. KASR significantly improves the accuracy and scalability of service recommender systems.

1. INTRODUCTION

Big data analysis can discover trends of various social aspects and preferences of individual everyday behaviors. This provides a new opportunity to explore fundamental questions about the complex world [6]. For example, to build an efficient investment strategy, Preis et al. [5] analyzed the massive behavioral data sets related to finance and yielded a profit of even 326 percent higher than that of a random investment strategy. Choi and Varian [10] presented estimate sketches to forecast economic indicators, such as social unemployment, automobile sale, and even destinations for personal travelling [11]. Currently, it is important to provide efficient methods and tools for big data analysis. We give an application example of big data analysis. Mapreduce has emerged as a popular and easy-to-use programming model for large-scale data analytics in data centers. It is an important application for numerous organizations to process explosive amounts of data, perform massive computation, and extract critical knowledge out of big data for business intelligence [12]. The efficiency of MapReduce performance and scalability can directly affect our society's ability to mine knowledge out of raw data. Energy consumption accounts for a large portion of the operating cost of data centers in analyzing such big data. While business and scientific applications are increasingly relying on the MapReduce model, the energy efficiency of MapReduceis also critical for data centers' energy conservation.

Hadoop [9] is an open-source implementation of MapReduce, currently maintained by the Apache Foundation, and supported by leading IT companies such as Facebook and Yahoo!. It implements the

MapReduce model by distributing user inputs as data splits across a large number of compute nodes. Hadoop uses a master program to command many TaskTrackers and schedule map tasks and reduce tasks to the TaskTrackers, A Hadoop program processes data through two main functions. Accordingly, the analytic functions are performed in two phases: mapping and reducing. In the mapping phase, the input dataset of a program is divided into many data splits. Each split is organized as many records of key and value (< >) pairs. One MapTask is launched per data split to convert the original records into intermediate data in the form of many ordered < k',v' > pairs. These ordered records are stored as a MOF (Map Output File) split. A MOF is organized into many data partitions, one per ReduceTask. In the reducing phase, each ReduceTask applies the reduce function to its own share of data partitions.

Recommender systems developed as an independent research area in the mid-1990s when recommendation problems started focusing on rating models. According to the definition of recommender system in [12], recommender system can be defined as system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful services in a large space of possible options. Current recommendation methods usually can be classified into three main categories: content-based, collaborative, and hybrid recommendation approaches. Content-based approaches recommend services similar to those the user preferred in the past. Collaborative filtering (CF) approaches recommend services to the user that users with similar tastes preferred in the past. Hybrid approaches combine content-based and CF methods in several different ways. CF algorithm is a classic personalized recommendation algorithm, which is widely used in many commercial recommender systems [3]. In CF based systems, users receive recommendations based on people who have similar tastes and preferences, which can be further classified into item-based CF and user-based CF. In item-based systems, the predicted rating depends on the ratings of other similar items by the same user. While in user-based systems, the prediction of the rating of an item for a user depends upon the ratings of the same item rated by similar users. And in this work, we will take advantage of a user-based CF algorithm to deal with our problem.

2. RELATED WORK

Discovering frequent graph patterns in a graph database offers valuable information in a variety of applications. If the graph dataset contains sensitive data of individuals such as mobile phone-call graphs and web-click graphs, releasing discovered frequent patterns may present a threat to the privacy of individuals. Differential privacy has recently emerged as the de facto standard for private data analysis due to its provable privacy guarantee [1]. In this paper we propose the first differentially private algorithm for mining frequent graph patterns.

Frequent sequential pattern mining is a central task in many fields such as biology and inane. Release of these patterns is raising increasing concerns on individual privacy. In this paper, we study the sequential pattern mining problem under the differential privacy framework which provides formal and provable guarantees of privacy [2]. Due to the nature of the differential privacy mechanism which perturbs the frequency results with noise, and the high dimensionality of the pattern space, this mining problem is particularly challenging. In this work, we propose a novel two-phase algorithm for mining both prefixes and substring patterns. In the first phase, our approach takes advantage of the statistical proper- ties of the data to construct a model-based prefix tree which is used to mine prefixes and a candidate set of substring patterns. The frequency of the substring patterns is further refined in the successive phase where we employ a novel transformation of the original data to reduce the perturbation noise. Extensive experiment results using real datasets showed that our approach is effective for mining both substring and prefix patterns in comparison to the state-of-the- art solutions.

With the wide deployment of smart card automated fare collection (SCAFC) systems, public transit agencies have been benefiting from huge volume of transit data, a kind of sequential data, collected every day. Yet, improper publishing and use of transit data could jeopardize passengers' privacy. In this paper, we present our solution to transit data publication under the rigorous differential privacy model for the Société de transport de Montréal (STM). We propose an efficient data-dependent yet differentially private transit data sanitization approach based on a hybrid-granularity prefix tree structure [13]. Moreover, as a post-processing step, we make use of the inherent consistency constraints of a prefix tree to conduct constrained inferences, which lead to better utility. Our solution not only applies to general sequential data, but also can be seamlessly extended to trajectory data. To our best knowledge, this is the first paper to introduce a practical solution for publishing large volume of sequential data under differential privacy. We examine data utility in terms of two popular data analysis tasks conducted at the STM, namely count queries and frequent sequential pattern mining. Extensive experiments on real-life STM datasets confirm that our approach maintains high utility and is scalable to large datasets.

The discovery of frequent itemsets can serve valuable economic and research purposes. Releasing discovered frequent itemsets, however, presents privacy challenges. In this paper, we study the problem of how to perform frequent item- set mining on transaction databases while satisfying differential privacy. We propose an approach, called Priv-Basis, which leverages a novel notion called basis sets [4]. A basis set has the property that any itemset with frequency higher than θ is a subset of some basis. We introduce algorithms for privately constructing a basis set and then using it to find the most frequent itemsets. Experiments show that our approach greatly outperforms the current state of the art.

3. BIG DATA AND MAP REDUCE CONCEPTS

BIG DATA

Big Data is becoming one of the most talked about technology trends nowadays. The real challenge with the big organization is to get maximum out of the data already available and predict what kind of data to collect in the future. How to take the existing data and make it meaningful that it provides us accurate insight in the past data is one of the key discussion points in many of the executive meetings in organizations. With the explosion of the data the challenge has gone to the next level and now a Big Data is becoming the reality in many organizations.

Big Data is exactly like a Rubik's cube – even though the goal of every organization and expert is same to get maximum out of the data, the route and the starting point are different for each organization and expert. As organizations are evaluating and architecting big data solutions they are also learning the ways and opportunities which are related to Big Data. There is not a single solution to big data as well there is not a single vendor which can claim to know all about Big Data. Honestly, Big Data is too big a concept and there are many players – different architectures, different vendors and different technology.

MAP REDUCE

MapReduce was designed by Google as a programming model for processing large data sets with a parallel, distributed algorithm on a cluster. Though, MapReduce was originally Google proprietary technology, it has been quite a generalized term in the recent time. MapReduce comprises a Map() and Reduce() procedures. Procedure Map() performance filtering and sorting operation on data where as procedure Reduce() performs a summary operation of the data. This model is based on modified concepts of the map and reduces functions commonly available in functional programing. The library where procedure Map() and Reduce() belongs is written in many different languages. The most popular free implementation of MapReduce is Apache Hadoop which we will explore tomorrow.The MapReduce Framework usually contains distributed servers and it runs various tasks in parallel to each other. There are

various components which manages the communications between various nodes of the data and provides the high availability and fault tolerance. Programs written in MapReduce functional styles are automatically parallelized and executed on commodity machines. The MapReduce Framework takes care of the details of partitioning the data and executing the processes on distributed server on run time. During this process if there is any disaster the framework provides high availability and other available modes take care of the responsibility of the failed node. As you can clearly see more this entire MapReduce Frameworks provides much more than just Map() and Reduce() procedures; it provides scalability and fault tolerance as well. A typical implementation of the MapReduce Framework processes many petabytes of data and thousands of the processing machines.A typical MapReduce Framework contains petabytes of the data and thousands of the nodes. Here is the basic explanation of the MapReduce Procedures which uses this massive commodity of the servers.

MAP() PROCEDURES:

There is always a master node in this infrastructure which takes an input. Right after taking input master node divides it into smaller sub-inputs or sub-problems. These sub-problems are distributed to worker nodes. A worker node later processes them and does necessary analysis. Once the worker node completes the process with this sub-problem it returns it back to master node.

REDUCE() PROCEDURES

All the worker nodes return the answer to the sub-problem assigned to them to master node. The master node collects the answer and once again aggregate that in the form of the answer to the original big problem which was assigned master node. The MapReduce Framework does the above Map () and Reduce () procedure in the parallel and independent to each other. All the Map() procedures can run parallel to each other and once each worker node had completed their task they can send it back to master code to compile it with a single answer. This particular procedure can be very effective when it is implemented on a very large amount of data (Big Data).

HDFS

Hadoop Distributed File System (HDFS) is a virtual file system. There is a big difference between any other file system and Hadoop. When we move a file on HDFS, it is automatically split into many small pieces. These small chunks of the file are replicated and stored on other servers (usually 3) for the fault tolerance or high availability.

- highly fault-tolerant and is designed to be deployed on low-cost hardware.
- provides high throughput access to application data and is suitable for applications that have large data sets.
- relaxes a few POSIX requirements to enable streaming access to file system data.

4. KEYWORD AWARE SERVICE DISCOVERY USING MAPREDUCE

In recent years, the amount of data in our world has been increasing explosively, and analyzing large data sets—so-called “Big Data”— becomes a key basis of competition underpinning new waves of productivity growth, innovation, and consumer surplus. Then, what is “Big Data”?, Big Data refers to datasets whose size is beyond the ability of current technology, method and theory to capture, man-age, and process the data within a tolerable elapsed time. Today, Big Data management stands out as a challenge for IT companies. The solution to such a challenge is shifting increasingly from providing hardware to provisioning more manageable software solutions [2]. Big Data also brings new opportunities and critical challenges to industry and academia. Similar to most big data applications, the big data tendency also poses heavy impacts on service recommender systems. With the growing number of alternative services, effectively recommending services that users preferred have become an important research issue. Service recommender systems have been shown as valuable tools to help users deal with services overload and provide appropriate recommendations to them. Examples of such practical applications include CDs,

books, web pages and various other products now use recommender systems [7]. Over the last decade, there has been much research done both in industry and academia on developing new approaches for service recommender systems [8].

With the success of the Web 2.0, more and more companies capture large-scale information about their customers, providers, and operations. The rapid growth of the number of customers, services and other online information yields service recommender systems in "Big Data" environment, which poses critical challenges for service recommender systems. Moreover, in most existing service recommender systems, such as hotel reservation systems and restaurant guides, the ratings of services and the service recommendation lists presented to users are the same. They have not considered users' different preferences, without meeting users' personalized requirements. Following is an example in hotel reservation system illustrating such a case.

So the problem is how to provide Alice and Tom a personalized recommendation list respectively to recommend the most appropriate hotels to them in "Big Data" environment efficiently. There are two reviews of W Hong Kong hotel snapshotted from a well-known travel review site. User u_a prefers modern feel and delicious food, while another user u_b is concerned more about good location, great service, breakfast, shopping and convenient transportation. We know that the reviews of the users commented for the hotels are related with both their preferences and the quality of the hotels. So the review of u_b to is useful to both Alice and Tom, while the review of u_a is useful to Tom only, since the review of u_a is irrelevant to the preferences of Alice. Thus it could be found that the information extracted from the reviews of previous users can be used to help recommend appropriate services to new users. Motivated by these observations, in this paper, we ad-dress these challenges through the following contributions: (1) A keyword-aware service recommendation method, named KASR, is proposed in this paper, which is based on a user-based Collaborative Filtering algorithm. (2) In KASR, keywords extracted from reviews of previous users are used to indicate their preferences. Moreover, we implement it on a distributed computing platform, Hadoop, which uses MapReduce as its computing framework.

5. SERVICE RECOMMENDATION USING PRIVACY PRESERVED FREQUENT ITEMSET MINING TECHNIQUE

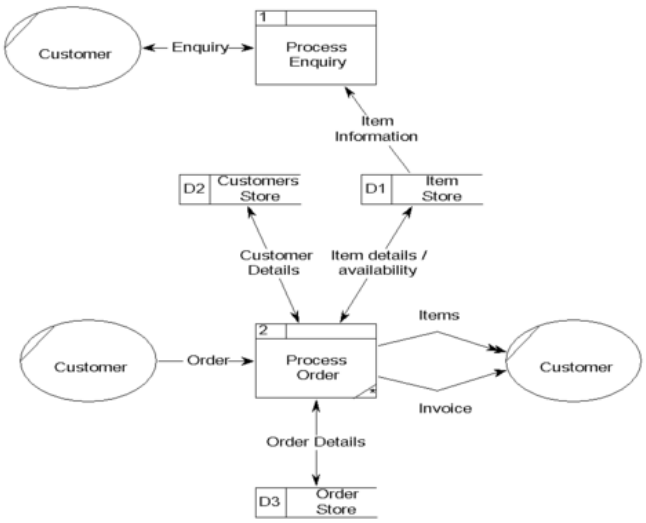

Fig. No: 5.1. PRIVACY PRESERVED SERVICE RECOMMENDATIONS UNDER CLOUDS

5.1. Hadoop-cygwin Configuration

In this configuration module Hadoop should be configured with Cygwin in windows environment. Cygwin is a large collection of GNU and Open Source tools which provide functionality similar to a Linux distribution on Windows. It is needed to run the scripts supplied with Hadoop. Once the installation is finished, datanode, secondary node and jobtracker node will give as a command in Cygwin and it will execute in Hadoop.

5.2. Transaction Module (dataset)

Every user transaction items should be saved in a database. In earlier days we are using a traditional approach for online transaction. It could not be applied

in present days due to huge population growth. So we go for mining techniques.

5.3. Frequent Itemstes Mining

Once the user purchase the product the transaction id, order no should be saved in a dynamic database. The same product should be purchased by different user. Here, we predict frequent item sets in a database should be predicted. Hence, frequent item sets mining should be used.

5.4. Actual Database

Now frequent item sets are saved in a actual database. Hence datasets should be predicted.

6. CONCLUSION

The wide spread use of online reviews as a way of conveying views and comments has provided a unique opportunity to understand the general public's sentiments and derive business intelligence. We investigate the problem of designing a differentially private FIM algorithm. We propose our private FP-growth (PFP-growth) algorithm, which consists of a pre- processing phase and a mining phase. Formal privacy analysis and the results of extensive experiments on real datasets show that our PFP-growth algorithm is time-efficient and can achieve both good utility and good privacy.

REFERENCES

[1] Entong Shen and Ting Yu, "Mining Frequent Graph Patterns with Differential Privacy", North Carolina State University, March 2013.

[2] Luca Bonomi and Li Xiong D , "A Two-Phase Algorithm for Mining Sequential Patterns with Differential Privacy", ACM, 2013

[3] D. Logothetis, C. Olston, B. Reed and K. Yocum, "Stateful bulk processing for incremental analytics," in Proc. 1st ACM Symp. Cloud Comput., 2010, pp. 51–62.

[4] Ninghui Li, Wahbeh Qardaji, Dong Su, Jianneng Cao, "PrivBasis: Frequent Itemset Mining with Differential Privacy", Proceedings of the VLDB Endowment, Vol. 5, No. 11, 2012

[5] T. Preis, H. S. Moat, and E. H. Stanley, "Quantifying trading behavior in financial markets using Google trends," Sci. Rep., vol. 3, p. 1684, 2013.

[6] P. Mika and G. Tummarello, "Web semantics in the clouds," IEEE Intell. Syst., vol. 23, no. 5, pp. 82–87, Sep./Oct. 2008.

[7] S. Ewen, K. Tzoumas, M. Kaufmann and V. Markl, "Spinning fast iterative data flows," in Proc. VLDB Endowment, 2012, vol. 5, no. 11, pp. 1268–1279.

[8] Y. Zhang, S. Chen, Q. Wang, and G. Yu, "i2mapreduce: Incremental mapreduce for mining evolving big data," CoRR, vol. abs/ 1501.04854, 2015.

[9] Apache Software Foundation, Apache Hadoop Project. [Online]. Available: http://hadoop.apache.org/.

[10] H. Choi and H. Varian, "Predicting the present with Google trends," Econ. Rec., vol. 88, no. s1, pp. 2–9, 2012.

[11] Xiaochun Yun, Guangjun Wu, Guangyan Zhang and Shupeng Wang, "FastRAQ: A Fast Approach to Range-Aggregate Queries in Big Data Environments", IEEE Transactions On Cloud Computing, Vol. 3, No. 2, April/June 2015

[12] Weikuan Yu, Yandong Wang and Cong Xu, "Virtual Shuffling for Efficient Data Movement in MapReduce", IEEE Transactions On Computers, Vol. 64, No. 2, February 2015

[13] Rui Chen, Benjamin C. M. Fung, Bipin C. Desai and Neriah M. Sossou, "Differentially Private Transit Data Publication: A Case Study on the Montreal Transportation System", ACM, 2012