Heuristics based Sensitive Pattern Hiding on Hadoop MapReduce Framework

A DISSERTATION

Submitted in partial fulfillment of the

requirement for the award of the degree of

MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

NISHTHA BEHAL
(14535032)

DEPARTMENT OFCOMPUTER SCIENCEAND ENGINEERING

INDIANINSTITUTEOFTECHNOLOGY, ROORKEE

ROORKEE-247667 (INDIA)

May, 2016

CANDIDATE'S DECLARATION

I hereby declare that the work, which is being presented in the dissertation entitled "**Heuristics based Sensitive Pattern Hiding on Hadoop MapReduce Framework**" towards the partial fulfillment of the requirement for the award of the degree of **Master of Technology in Computer Science and Engineering** submitted in the Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee, Uttarakhand (India) is an authentic record of my own work carried out during the period from July 2015 to May 2016, under the guidance of **Dr. Durga Toshniwal, Associate Professor,** Department of Computer Science and Engineering, IIT Roorkee.

The matter presented in this dissertation has not been submitted by me for the award of any other degree of this or any other Institute.

Date:
Place: Roorkee (**Nishtha Behal**)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

Date:
Place: Roorkee

(**Dr. Durga Toshniwal)**
Associate Professor
Department of Computer Science and Engineering
IIT Roorkee

ACKNOWLEDGEMENT

I would like to express my sincere gratitude towards my guide Dr. Durga Toshniwal, Associate Professor, Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, for her valuable suggestions, reviews and her extended support even in non-working hours. Her integrity, commitment and professional attitude has inspired me a lot.

I, hereby, express my deepest gratitude and heartiest thanks to Prof. Manoj Mishra, Head of the Department, Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, for providing me the opportunity, guidance and necessary facilities to work. I am also thankful to all the staff members of the department of computer science for their support. I sincerely thank my family and all my friends for directly or indirectly helping me and giving me moral support that motivated me during the course of the work.

(Nishtha Behal)

ABSTRACT

In recent times, data mining has gained immense application because of the ability with which it can extract previously unknown and potentially useful information from raw data. Frequent Pattern Mining is a subfield of data mining in which patterns that occur frequently in the data are extracted from the data. In case of collaborative frequent pattern mining, mining may lead to the extraction of patterns that are sensitive. The revelation of such sensitive patterns is undesirable for the data owner. Privacy preservation in data mining is the area under which techniques that allow the sensitive information present in the data to be hidden from the data mining process are designed and analysed. In order to hide the sensitive information, modifications are performed on the data and this decreases the quality of the data and hence mining results obtained from such data may not be accurate. Thus, there is a trade-off between the privacy and the utility of the data. For preserving the sensitive patterns from the frequent pattern mining process various sensitive pattern hiding techniques exist. All these techniques cause side effects to the data by decreasing its quality and also are an overhead to the frequent pattern mining process. In this work the focus is to decrease the side effect caused to the data while maintaining a low running time. Existing sensitive pattern hiding techniques can be broadly categorized as heuristics based, border based and exact approaches. Heuristics based approaches are fast but they cause maximum side effect. Here we have proposed two heuristics based sensitive pattern hiding algorithms which allow fast hiding of sensitive patterns on Hadoop MapReduce framework while reducing the side effect.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Introduction

Data mining is the process of extracting potentially useful and previously unknown information from raw data. The advancement in technology with newer technologies like cloud computing, distributed processing etc. coming up as well as with storage devices becoming cheaper, it has become possible to store as well as analyse humungous amount of data. Also parallelization of mining algorithms has reduced the time needed for this previously time-intensive task. Due to these advancements several new use cases of collaborative data mining in different domains like marketing, weather forecasting etc. have evolved. Collaborative mining is when two or three parties collectively mine their data in order to gain better insights from the data.

The problem here is that in most cases when collaborative data mining is performed privacy is lost. The owner of the data may wish to hide some sensitive information from the other collaborators while reaping the benefits of mining as well. So, though collaborative data mining may allow better planning, intelligent decision making and more efficient business strategies but privacy is one major issue to be handled here. And here is where Privacy Preserving Data Mining (PPDM) techniques come in to handle this issue. PPDM is the branch of data mining which aims at devising techniques that allow data containing sensitive information to be used for mining while keeping the sensitive information hidden. [1] This sensitive information may be sensitive attributes visible in the raw data itself or sensitive knowledge that can be extracted from the data as a result of mining. Though various PPDM techniques exist today but there is always a trade-off between privacy and data utility in every technique. For preserving the sensitive information present in the data it is necessary to make some transformations to the data so as to hide the sensitive information. This process of transforming the original dataset into a new dataset from which none of the sensitive information can be extracted is known as sanitization. This transformation always causes some loss in the utility of the data thus affecting the data mining results obtained from mining such data.

Based on our study of PPDM techniques, we have categorized them along two main lines-

- Generic PPDM techniques

  Generic approaches are the approaches that introduce privacy preservation into the data in such a way that the transformed data can be used for mining without worrying about the release of sensitive data or information. These techniques are generally used for data hiding purposes such as removal of sensitive attributes from the data. Examples include generalization, randomization, sampling etc.

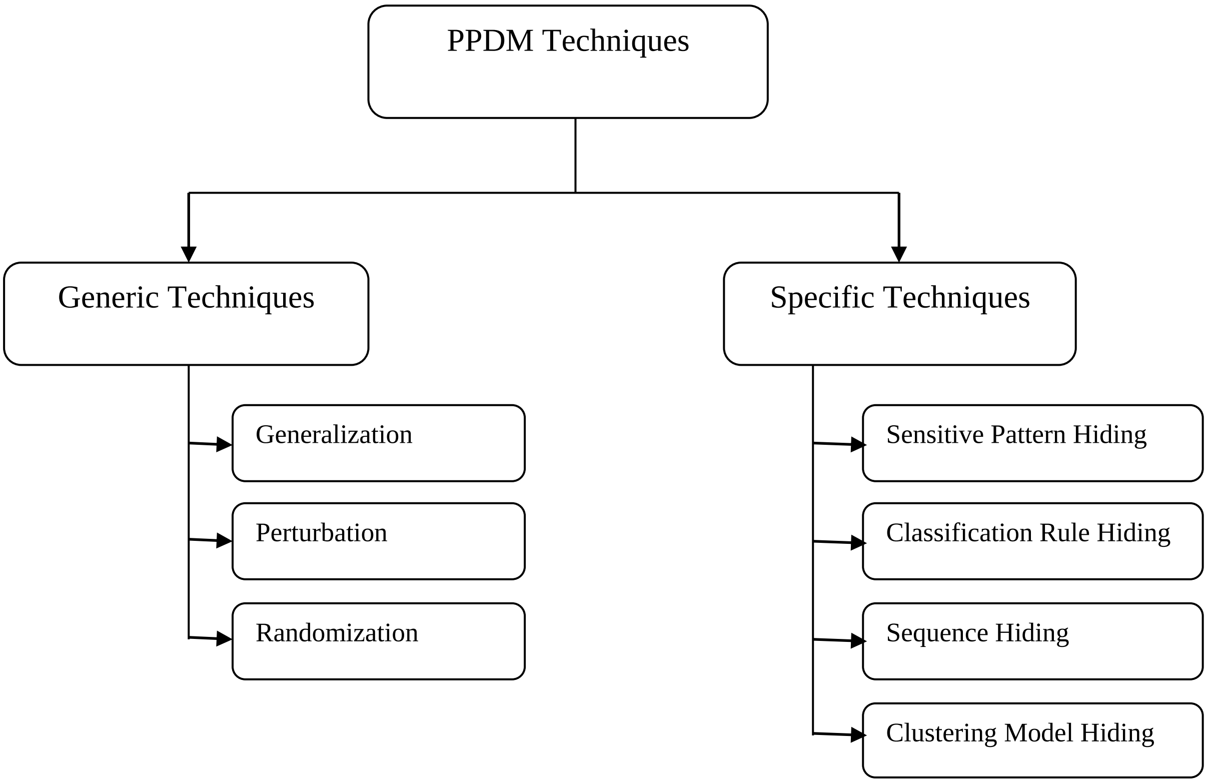


Figure 1.1: *Classification of PPDM Techniques*

- Specific PPDM techniques.

  The specific techniques are the techniques that cater to the problem of privacy in a particular data mining task. Privacy preservation in this case is done to hide sensitive information. For instance, sensitive pattern hiding techniques are used to preserve sensitive patterns present in the data. They are aimed at preserving information in case of frequent pattern mining. Performing sensitive pattern hiding transformations on a dataset does not ensure preservation of sensitive information present in the data when cluster analysis is performed on the same dataset. The specific techniques can be further categorized on the basis of the data mining task they are applicable to.

Each of these techniques whether specific or generic decreases the quality of the data and also adds to the time needed for the data mining process.

The need today is to devise PPDM techniques that ensure:

- A balance between privacy and accuracy i.e. less harm to the data's utility.
- Require less time to transform the data.
- Ability to harness the benefits and resources offered by new technologies like cloud computing, parallel processing etc.
- Scalability to be applicable to huge datasets.

1.2 Privacy Preservation in Frequent Pattern Mining

In this work, the focus is on Frequent Pattern Mining (FPM) which is a subfield of data mining in which the patterns that frequently occur in the data are extracted from the data. During collaborative mining it may happen that some patterns which may be sensitive to the owner's business may get revealed. Therefore privacy preservation techniques are needed to preserve these sensitive patterns. Based on the classification provided in the previous section it is evident that here the focus in on specific PPDM techniques for sensitive pattern hiding.

1.2.1 Sensitive Pattern Hiding

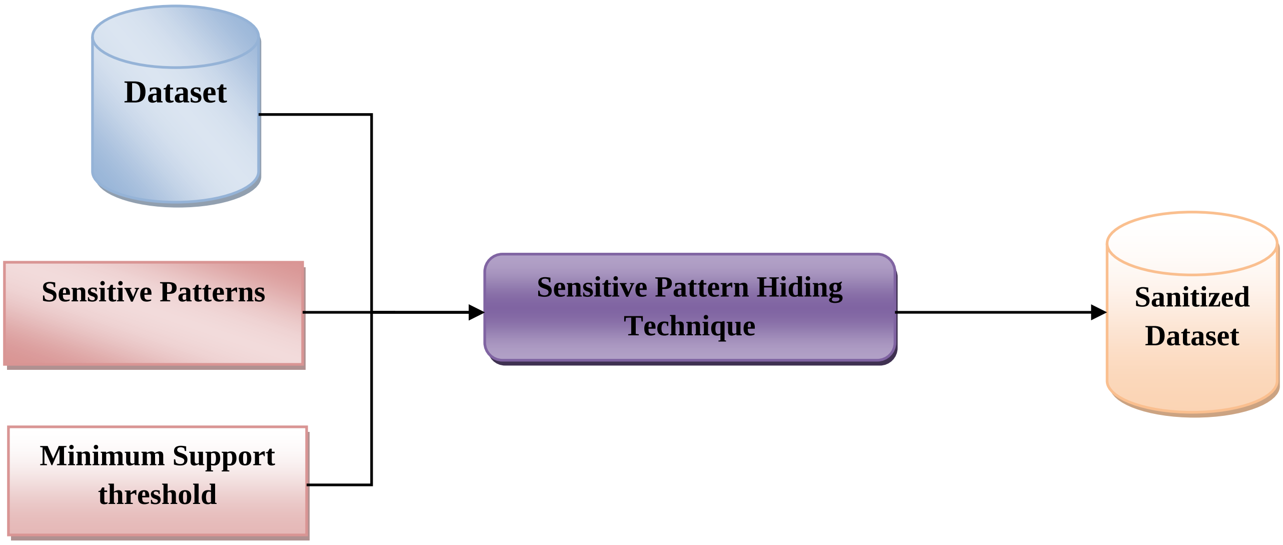


Figure 1.2: *Sensitive Pattern Hiding Process*

Sensitive Pattern Hiding (SPH) is the process of performing transformations on the data to hide the sensitive patterns i.e. prevent sensitive knowledge from getting extracted when FPM is performed on the data. Every SPH algorithm takes as input the list of sensitive itemsets and the minimum support threshold value in addition to the dataset. A sensitive itemset may or may not be a sensitive pattern based on the support with which it is present in the dataset. Basic idea behind each SPH algorithm is to first determine the sensitive patterns and then make these sensitive patterns infrequent in the dataset. For this purpose the dataset is modified. Two types of transformations can be performed on the data in order to do the same. A pattern may be made infrequent either by removal of data or by modifying the existing data. Removing data causes lesser side effects as compared to modification of information as it does not add any false information to the data. Minimum support threshold is used to determine the sensitive patterns and also the number of transactions to be sanitized in order to make them infrequent in the data.

A SPH algorithm may be absolute i.e. may be able to hide each and every sensitive itemset completely from the dataset or may hide them with a certain probability. Choosing one of the types for hiding the sensitive patterns depends on whether the application involved requires all the sensitive patterns to be hidden completely or not. We will discuss the existing sensitive pattern hiding techniques in detail in the next chapter.

1.2.2 Challenges

The biggest challenge in case of SPH is to maintain the utility of the data. While hiding the sensitive patterns it may happen that some of the non-sensitive frequent patterns become infrequent. This may lead to inaccurate mining results. Also modification of the existing information may result in the generation of some patterns that did not exist in the original dataset. So the main challenge in SPH is to maintain a balance between the privacy and the utility of the data. Ideally, the non-sensitive information should be preserved and no false patterns should become frequent but in reality preserving all the non-sensitive information is not always possible. So the challenge is to preserve maximum information in the dataset while hiding the sensitive patterns.

Another challenge is in terms of the extra cost added to the mining process in terms of time. The SPH process in a way adds to the cost of the mining process and is definitely an overhead. Reducing the time cost of the SPH process is hence also one challenge though the bigger challenge is to maintain the quality of the data to the maximum level while hiding the sensitive information.

1.3 Motivation

The realization of the potential benefits gained through collaborative FPM is in itself the greatest motivation towards SPH techniques. The loss of sensitive information is a threat to the data's privacy and this may lead to the data owner not engaging in collaborative mining and thus as a result the potentially valuable information hidden in the data stays hidden.

For instance, consider a scenario where a wholesaler wishes to mine the data collected by recording the transactions occurring at the retailers who sell his/her stuff collectively in order to determine the demand patterns accurately. Now consider that there is one retailer who witnesses very high sales of the wholesalers' products and also the wholesalers' products enjoy a monopoly at the retailer's store. If this information is revealed to the wholesaler then the retailer's dependence on the wholesaler's products will become visible to the wholesaler. The wholesaler may then take advantage of this dependence by increasing the prices of the involved products. In such a situation the retailer may incur an unseen loss on sharing his/her transaction data. So some way is needed by which the data shared for mining does not reveal the sensitive information. Retailer also needs to make sure that the non-sensitive information still stays intact in the dataset because it is necessary that the results of the mining process are accurate so as to be useful. Hiding should be performed in such a way that all the important non-sensitive information should be preserved and no new artificial non existing information should be generated in the process. The trade-off between privacy and accuracy should be minimal.

In order to hide the sensitive patterns, transformations are needed on either the dataset before the mining process or hiding should be embedded in the mining process itself. No matter whichever way is chosen an increase in the cost of the mining process is encountered. So, new SPH techniques are needed which can harness the power of new technologies like cloud computing, parallel processing etc. to be more cost efficient in terms of running time.

1.4 Problem Statement

The problem statement for the present work can be stated as follows:

"To improve the sensitive pattern hiding process using better heuristics and also reduce the running time cost by using parallel programming techniques."

The problem can be broken into three sub problems:

- Hide all the sensitive itemsets,

- Ensure that maximal amount of non-sensitive information is preserved in the data and,
- No non-existing artificial patterns are generated.

1.5 Specific Research Contribution

We have proposed two new heuristics based SPH algorithms viz. Parallel Maximum Support Item Removal (PMSIR) and Parallel Maximum Support Item Removal from transactions with Maximum Degree (PMSIRMD) for absolute hiding of sensitive itemsets while reducing the side effects caused to the dataset by the heuristic algorithms. The second algorithm is a further improvement of the first algorithm.

Both the algorithms in addition to using heuristics group together the frequent sensitive itemsets or sensitive patterns together on the basis of common items and thus more than one pattern can be sanitized at the same time. The degree of a transaction is termed as the number of sensitive patterns that are contained in the transaction. In the second algorithm the choice of which transaction to be sanitized at each point is based on degree. Greater the degree of a transaction better is the gain as multiple transactions can be sanitized using a single transaction. This reduces the side effect caused to the data and leads to better data quality. In addition to reducing the side effect our algorithms perform much better than the existing SPH algorithms in terms of running time as our algorithms are implemented on Hadoop MapReduce framework. We have discussed the algorithms in detail in the further chapters.

Our algorithms guarantee that each of the sensitive itemsets will be hidden with 100% certainty and thus are useful for privacy critical applications where it is required that all sensitive information should be hidden completely from the data. For example data related to military operations may contain some sensitive information that needs to be hidden at any cost. In such a situation an absolute SPH approach is the only solution.

1.6 Organization of the Report

The organization of the rest of the report is as follows. Chapter II describes the existing SPH approaches and also describes the required theoretical background. Chapter III describes the proposed algorithms Parallel Maximum Support Item Removal (PMSIR) and Parallel Maximum Support Item Removal from transactions with Maximum Degree (PMSIRMD). Chapter IV describes the performance of the algorithms with the help of experimental results obtained on executing the algorithms on real as well as synthetic datasets. With Chapter V we conclude the report by discussing the future work.

Before describing the past work done under the area of SPH we begin by providing some background knowledge regarding the process in order to better understand the approaches.

2.1 Frequent Pattern Mining

Before moving to SPH let us first understand the FPM process which is extraction of knowledge from data in the form of patterns that occur frequently in it. In order to have a good understanding we begin by introducing the involved terminology.

Frequent pattern mining is generally done on market basket data where each record is a transaction describing the items bought together in a single purchase or basket. A set of many such transactions forms a dataset. Let *I* be a set of literals $\{i_1, i_2, i_{3,\ldots} i_n\}$, denoting the items present in the dataset. In a given a dataset *D*, each transaction *T* is a set of items such that $T \subseteq I$. Each transaction is identified with the help of unique number denoted as *TID*.

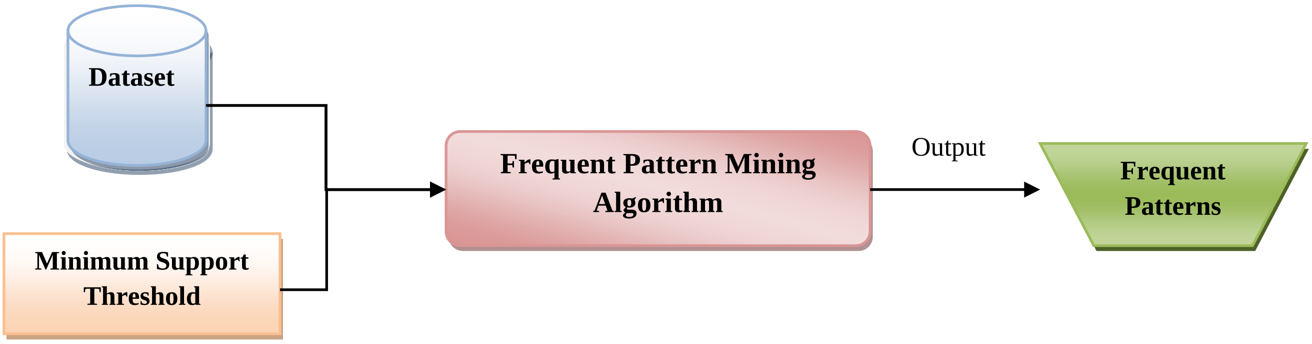


Figure 2.1: *Frequent pattern mining process*

A set of items is known as an itemset. For any itemset *X* such that $X \subseteq I$, support count of *X* in *D* is the number of transactions in *D* that contain *X*,

$$\text{Support of } X \text{ in } D, \sigma(X)_D = |\{T, \text{ such that } T \in D \text{ and } X \subseteq T\}| \tag{2.1}$$

A pattern *P* is a set of items such that $P \subseteq I$. Let the minimum support threshold be denoted by *MST*. *MST* is the criterion which is used to determine whether a pattern is frequent or not. Any pattern *P* such that $P \subseteq I$ is the frequent in a dataset D if and only if the support count for *P* is greater than *MST*. A frequent pattern may contain one or more than one items.

Various algorithms exist that can be used to mine frequent patterns but almost all of them are variants of two basic algorithms, apriori and fp-growth.

2.2 Sensitive Pattern Hiding

For a sensitive pattern to be hidden it is necessary that the support count of the sensitive pattern goes below *MST*. So the basic aim of each SPH approach is to make the sensitive patterns infrequent. This may be done by removing certain items from transactions or by adding noise to the transactions. This process of transforming the original data into data in which privacy is preserved is known as sanitization. All the SPH approaches are based on four general steps which we call the general sanitization algorithm [2].

Given a dataset *D*, minimum support threshold *MST* and the set of sensitive itemsets *I*, the general sanitization algorithm has the following four steps:

1. Based on the itemsets in *I* first identify the set of sensitive patterns, *R* and also identify the set of sensitive transactions from the dataset. Any transaction that contains any sensitive pattern is a sensitive transaction.
2. For decreasing the support of a pattern we can either remove the complete pattern *P* or remove any item or group of items that are contained in *P* from the sensitive transactions that contain *P*. In most algorithms a single item contained in the pattern is chosen as the candidate for removal. This candidate item is known as victim. In this step for each pattern *P* in *R*, identify the candidate item that will be removed from the sensitive transactions containing *P*.
3. Identify the number of sensitive transactions that need to be transformed for each pattern *P* in *R* to make *P* infrequent in the dataset with respect to *MST*, the minimum support threshold. Let us denote this number by *N*.
4. Post calculation of *N*, for each pattern *P* in *R,* choose the *N* transactions to be sanitized from the set of sensitive transactions containing *P*.

Steps 1 and 3 are same for almost all the algorithms, what makes them different is the way step 2 and 4 are performed i.e. on what basis the candidate items are chosen and what basis the sensitive transactions to be transformed are chosen. This selection can be done in different ways. Based on the way this choice is done, SPH approaches can be classified. There are broadly three types of SPH approaches as shown in figure 2.2. These are heuristics based, border based and exact approaches.

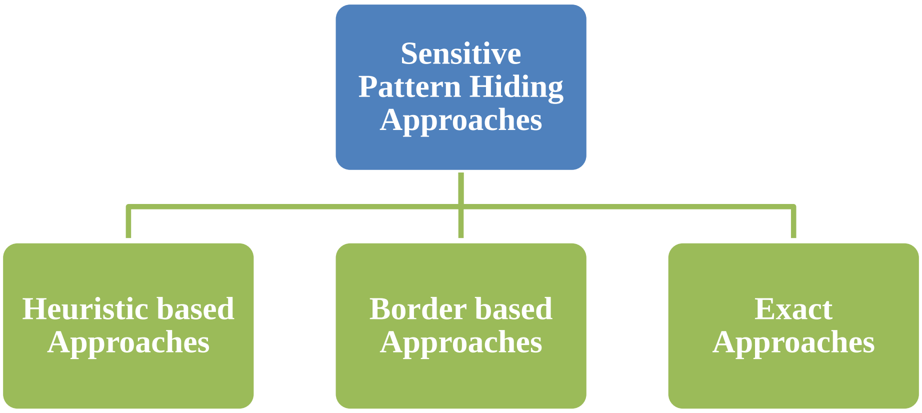

Sensitive Pattern Hiding Approaches
Heuristic based Approaches
Border based Approaches
Exact Approaches

Figure 2.2: *Classification of SPH approaches*

The problem of hiding the sensitive itemsets while retaining the maximal level of non-sensitive data in the dataset is an NP hard problem. And so use of heuristics for sensitive pattern hiding is one possible option. Heuristic based approaches make use of some heuristics to choose the sensitive transactions to be sanitized and the victim item to be removed. As heuristics are involved these approaches are fast. But these approaches cause a loss of non-sensitive information and many at times affect the utility of the data very badly. These approaches are preferred because of their simplicity and speed but they cause many side effects to the data.

By taking up concepts from the border theory and applying it to SPH a new set of SPH approaches have evolved. Border based approaches work by modifying the border of the dataset in the lattice of frequent and infrequent patterns. The main idea is to move all the sensitive patterns to the negative border while retaining the non-sensitive frequent patterns in the positive border. Border based approaches are computationally complex in comparison to heuristic approaches but in most cases are better than heuristic approaches at maintaining the data's utility.

Exact approaches work by formulating the sanitization problem into constraint satisfaction problem and use mathematical concepts to provide the solution. Exact approaches provide highly optimal solutions but these are rarely used because of the high computational complexity.

2.2.1 Evaluation Metrics

The performance of a SPH approach depends on how well it hides the sensitive patterns while preserving the non-sensitive frequent patterns. To quantify the performance of sensitive pattern hiding approaches the following three metrics can be used.

- ***Hiding Failure***: It is defined as the percentage of sensitive patterns that can be mined from the sanitized dataset. It is an indication of how well does the approach hide the sensitive patterns. Ideally it should be 0%.
- ***Misses Cost***: It is defined as the percentage of non-sensitive frequent patterns that were present in the original dataset but cannot be mined from the sanitized dataset. It is an indication of the quality of the data post sanitization; to what extent does the sanitization process degrade the utility of data. Ideally it should be 0%.
- ***Artificial Patterns***: It may happen that when frequent pattern mining is performed on the sanitized dataset then some new patterns are mined that did not exist in the original dataset. Such patterns are called artificial patterns and are unwanted in the sense that they decrease the accuracy of the mining results. They represent information that actually does not exist. Ideally no artificial patterns should be generated.

Figure 2.3 describes the metrics diagrammatically. An ideal SPH approach would be one that has 0% hiding failure, 0% misses cost and does not introduce any artificial patterns in the sanitized dataset. But in reality such an approach cannot exist because whenever a pattern is sanitized some information loss is bound to happen.

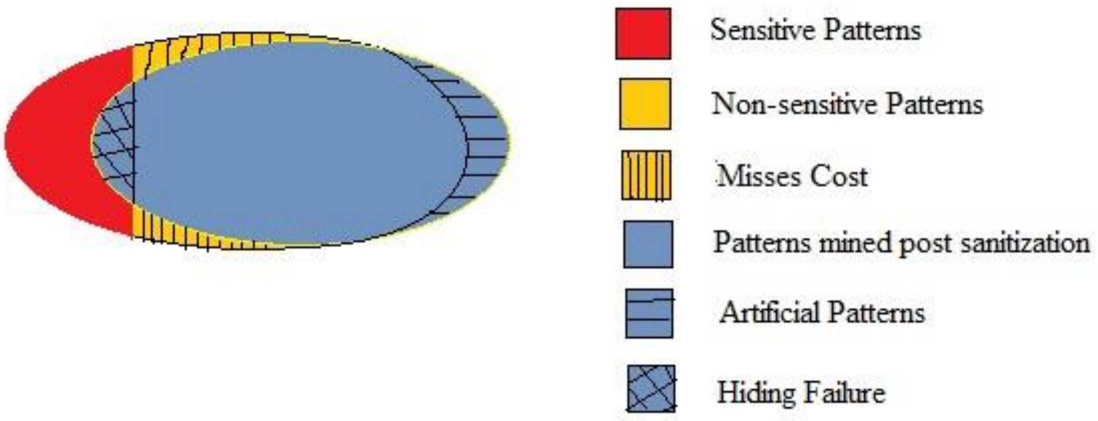


Figure 2.3: *Side effects of the sensitive pattern hiding process*

2.3 Hadoop MapReduce Framework

Hadoop MapReduce Framework or Hadoop in short allows reliable and fault tolerant distributed processing of humungous amounts of data at fast speed across a clusters of computers using the MapReduce programming model [3]. Figure 2.4 describes the main components of Hadoop MapReduce Framework version 2 or Hadoop 2 which are Hadoop Distributed File System (HDFS), YARN, MapReduce Framework and other processing modules like HBase, Hive, Zookeeper, etc.

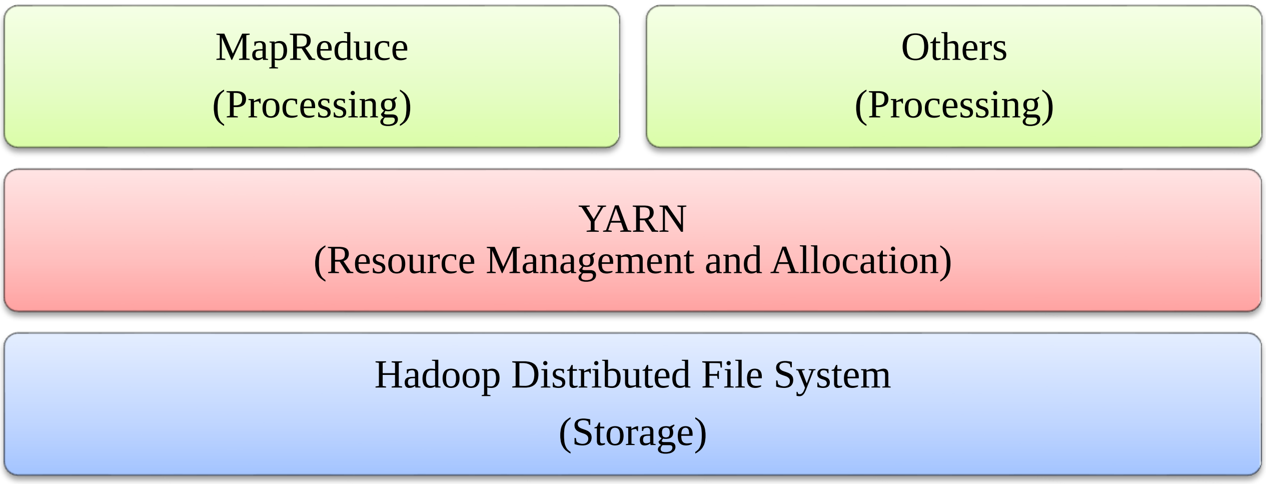

MapReduce
(Processing)
Others
(Processing)
YARN
(Resource Management and Allocation)
Hadoop Distributed File System
(Storage)

Figure 2.4: *Components of Hadoop 2.0*

HDFS provides the solution to cheap and reliable storage of huge amounts of data by being a distributed file system that allows the usage of commodity hardware connected through a network for reliably storing data. By replicating the data across storage devices HDFS allows reliable storage as well as quick access time.

Figure 2.5 shows the flow of data in a basic MapReduce job.

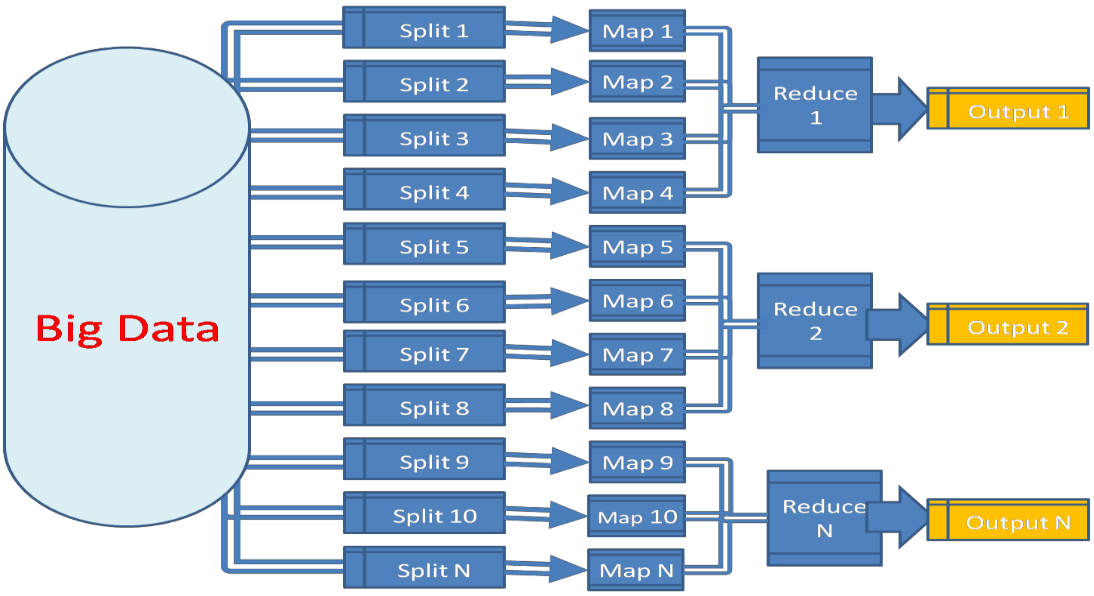


Figure 2.5: *Flow diagram for MapReduce [3]*

MapReduce is a parallel programming model and associated implementation that allows the programmer to process data in parallel in a distributed environment. As shown in figure 2.5 first step is the splitting up of data into chunks. This is handled by the framework. After this the data is processed first by Map and then by Reduce in order to generate the final output. The programmer needs to define only two functions; Map and Reduce and the rest is handled by the framework. In Hadoop both resource management as well as data processing was handled by the MapReduce framework but with Hadoop 2 these functions are separated. With other data processing modules also coming up a separate resource manager and allocator was introduced which is called YARN. YARN is an acronym for Yet Another Resource Allocator. With Hadoop 2 YARN became the managing tool for connecting data and the data processing tools.

The Hadoop model takes care of the communication overhead as well as system failures on its own thus making it easy for the programmer to perform parallel processing.

2.4 Literature Review

In this section the existing sensitive pattern hiding approaches are discussed. [4] is one of the first works that investigates the problem of disclosure of sensitive rules and gives a heuristic based solution that uses concepts from graph theory. It also proves that finding an optimal sanitization of the original dataset is NP-hard.

As the problem is NP-hard, so many heuristic approaches have been proposed for sensitive pattern hiding [2, 4, 5, 6, 7]. In [2] Oliveira has proposed an approach that works for Boolean association rules. Instead of adding noise by randomly removing or adding an item to the transaction, the items present in the sensitive patterns are removed from the transactions. One of heuristics proposed in [2] is to select the transaction with minimum length and remove the item with maximum support value.

The author in [5] has given a solution to the problem in which instead of switching a 0 to 1 i.e. removing or adding any item the value is made unknown. This is just a way of ignoring the presence of the item instead of changing its count. With some modifications made to the values of support and confidence by making changes to the data the sensitive information is hidden and the non-sensitive frequent patterns are preserved.

In [6] a heuristic algorithm that clusters the association rules on the basis of items in the consequent of a rule is proposed. By grouping together the sensitive patterns the algorithm manages to hide more than one pattern at the same time. Only drawback of the approach is that it needs the sensitive rules to have a single item in the consequent.

In [7] the sensitive patterns are hidden by reducing the confidence of the corresponding rules instead of reducing the support of the sensitive item. Advantage is that the data quality is preserved whereas the disadvantage is that the algorithm fails to hide all the sensitive patterns.

In [8] a heuristic sensitive pattern hiding approach is proposed for risk management for retail supply chain management. The algorithm at each step chooses the victim item such that the side effect on the non-sensitive frequent patterns is minimal.

Border based approaches to the hiding of sensitive patterns are proposed in [9, 10, 11]. The main idea behind all the border based approaches is that the impact of the changes in the dataset can be minimized by just considering the impact made on the positive border of frequent patterns. The aim of the algorithm is to exercise minimal impact on the expected positive border i.e. the set of non-sensitive frequent patterns. The expected negative border now contains all the sensitive frequent patterns in addition to the infrequent patterns.

Exact approaches [12] provide the optimal solution but these approaches are computationally very expensive. In [12] with the use of integer programming the author proposes an exact algorithm for sensitive pattern hiding that aims to reduce the distance between the original and the sanitized dataset so as to ensure data quality. The approach also makes use of the border theory by considering the problem to be a border revision problem.

2.5 Research Gaps

Based on the study of the existing SPH approaches following three research gaps are identified:

- Heuristic approaches are fast but cause more side effects as compared to the other approaches. Most of the existing heuristic approaches cause a huge decrease to the utility of the data but are still a good choice because of their simplicity and speed. New better heuristics are needed that cause lesser side effects to the data.
- Both the border based and exact approaches are very complex as compared to the heuristic approaches. In case of exact approaches the complexity is so high that the

benefits gained in terms of data utility may seem of no use. Hence here the focus is on heuristics based SPH approaches.

- Most of the existing SPH algorithms are sequential in nature. Not much work has been done in designing parallel PPDM techniques. As the privacy preservation process can be seen as an overhead to the mining process a reduction in the time cost of the process is desirable. Harnessing the power of newer technologies like cloud computing and parallel processing can lead to very efficient and fast SPH approaches.

3.1 Proposed Framework

The focus here is phase 3. Phase 1 and 2 generate as output the inputs required for phase 3.

3.1.1 Generation of the dataset

A Synthetic dataset generator is used for generating the dataset containing market basket data [13]. The dataset generator takes as input the number of transactions, the number of itemsets and the average length of each transaction and produces the dataset in a text file. By varying the value of parameters- average transaction length, number of transactions, number of items different datasets are generated which serve as inputs for Phase 2 and Phase 3. We have preprocessed the generated dataset to convert it to a format that is suitable for Hadoop MapReduce framework. In addition to the synthetic datasets we have also used a real dataset for analyzing the performance of the proposed SPH algorithms.

3.1.2 Frequent Pattern Mining

To test the SPH approaches frequent patterns present in the dataset are needed. For this purpose in Phase 2 FPM is performed on the dataset. For this purpose a FPM algorithms is needed. There are broadly two types of FPM algorithms viz. candidate based and candidate less. Apriori algorithm is the most popular candidate based FPM algorithm [13]. Many parallel variants of apriori algorithm exist [14, 15, 16, 17]. Here the one in [16] is used to generate the frequent patterns. The algorithm is implemented on Hadoop MapReduce framework. The algorithm is used to generate the frequent patterns for different values of *MST*. In order to determine the suitable value for *MST* the FPM algorithm is executed for different *MST* values. In contrast to the sequential apriori algorithm which requires many scans of the dataset this parallel version of the algorithms works in two phases and requires far lesser number of scans of the dataset. It takes only two scans of the dataset.

3.1.3 Sensitive Pattern Hiding

Our proposed work starts in Phase 3. Both the previous phases are prerequisites to it and it is here that the proposed work begins. In this phase the sanitization of the dataset will be performed to hide the sensitive itemsets. The basic idea would be to first determine the sensitive patterns and then decrease the support of the sensitive patterns in order to make them infrequent while achieving minimal harmful impact on the non-sensitive frequent patterns. The amount of work needed to perform sanitization will depend on two major factors-

- Number of sensitive patterns to be hidden and
- Minimum support threshold at which the patterns should be hidden

Larger the number of sensitive patterns more is the work required to hide them. This leads to a decline in the utility of the data. So the focus of this work is to design SPH algorithms that minimize the side effects caused to the data while hiding the sensitive patterns. We have proposed two heuristics based SPH approaches which are described in detail in the next Section.

3.2 Proposed Sensitive Pattern Hiding Approaches

As discussed in chapter 2 most of the existing SPH approaches are heuristics based. These approaches are fast but have a lot of side effects on the dataset. Here the focus is on devising new heuristics for SPH that can reduce the side effects caused to the dataset. The SPH problem can be described as follows.

We have a dataset *D* and a set *S* of sensitive itemsets that should not be mined from *D* at *MST* α. The problem is to transform *D* into D' such that

a. All the itemsets in S are infrequent in D' and
b. Maximal amount of non-sensitive information is retained in D' post sanitization.

It should also be ensured that no new non-existing patterns are generated from the sanitized dataset. And this should be achieved in the lowest cost possible.

As discussed in chapter 2 there are broadly two types of sanitization algorithms. They are-

- Algorithms that only remove information from the dataset and
- Algorithms that modify the existing information by adding noise.

The second type suffers from the problem of artificial patterns. Artificial patterns give false knowledge which did not exist in the original data and hence can be misguiding.

In this work we have proposed two heuristic approaches Parallel Maximum Support Item Removal (PMSIR) and Parallel Maximum Support Item Removal from transactions with maximum Degree (PMSIRMD) based on the Hadoop MapReduce framework. Our proposed algorithms PMSIR and PMSIRMD only remove information and don’t modify it and hence don’t suffer from artificial patterns.

The two algorithms are described in detail in the next section.

3.2.1 Approach 1: Parallel Maximum Support Item Removal (PMSIR)

The Parallel Maximum Support Item Removal (PMSIR) algorithm is a heuristics based SPH algorithm that aims to reduce the side effects caused to the dataset by grouping together the sensitive patterns on the basis of common items. The maximum support item present in a sensitive pattern is chosen as the victim for removal in this algorithm.

By grouping together the transactions on the basis of victim items PMSIR does reduce the side effects cause to the dataset but the algorithm selects the sensitive transactions randomly until the sensitive patterns are not sanitized. Applying a suitable heuristic to make this selection will further improve the SPH process and so the next proposed algorithm further extends PMSIR by heuristically choosing the sensitive transactions for sanitization. The extended algorithm is described in the next Section.

3.2.2 Approach 2: Parallel Maximum Support Item Removal from Transactions with Maximum Degree (PMSIRMD)

This algorithm is very similar to PMSIR and the only difference is in the way sensitive transactions are chosen for sanitization. At every step transactions with maximum degree are chosen for sanitization. Degree of a transaction is defined as the number of sensitive patterns that a transaction contains.

The experiments conducted by us were aimed along two main directions. The first was to quantify the scalability and running time of our algorithms and the second one was to check the effectiveness of our pattern hiding approaches. All the experiments were conducted on a Linux virtual machine running the stand-alone version of Hadoop 2.7.1. The datasets used for this purpose are generated using the synthetic dataset generator [1]. We have also used a real dataset containing accident information to analyze our algorithms. In the subsequent sections we describe the results obtained on running the algorithms on synthetic as well as real dataset in varying conditions. For comparison we have also executed the experiments on existing SPH approaches.

In order to test the algorithms the set of sensitive patterns is required as input. Sensitivity of information is a semantic attribute and is completely dependent on the application and the data owner. In the real world situations the determination of which itemsets are sensitive would be done by experts and analysts after thorough investigation. Here as we are using synthetic datasets and cannot apply any application based constraints so we have chosen the set of sensitive patterns randomly.

4.1 Dataset Description

Two types of datasets are used here- Synthetic Datasets (SDs) and Accidents Dataset (AD). Table 4.1 describes the SDs.

Number of Transactions	Number of items	Average Transaction Length
100000	1000	30
200000	1000	30
300000	1000	30
400000	1000	30
500000	1000	30
600000	1000	30
700000	1000	30
800000	1000	30
900000	1000	30
1000000	1000	30
1000000	2000	30
1000000	3000	30
1000000	4000	30
1000000	5000	30

Table 4.1: *Description of the SDs using the parameters used for generation*

As shown in Table 4.1 by changing the number of items and transactions a multitude of datasets are generated. As the Hadoop MapReduce framework is specially designed for distributed processing of humungous datasets so here we have used datasets with millions of transactions. In order to experiment with different dataset sizes we have varied the number of transactions from 100000 to 1000000 and the number of items from 1000 to 5000.

The accidents dataset is a bench mark dataset for FPM. We have used the dataset to analyse the performance of the proposed algorithms.

4.2 Experiments on SDs

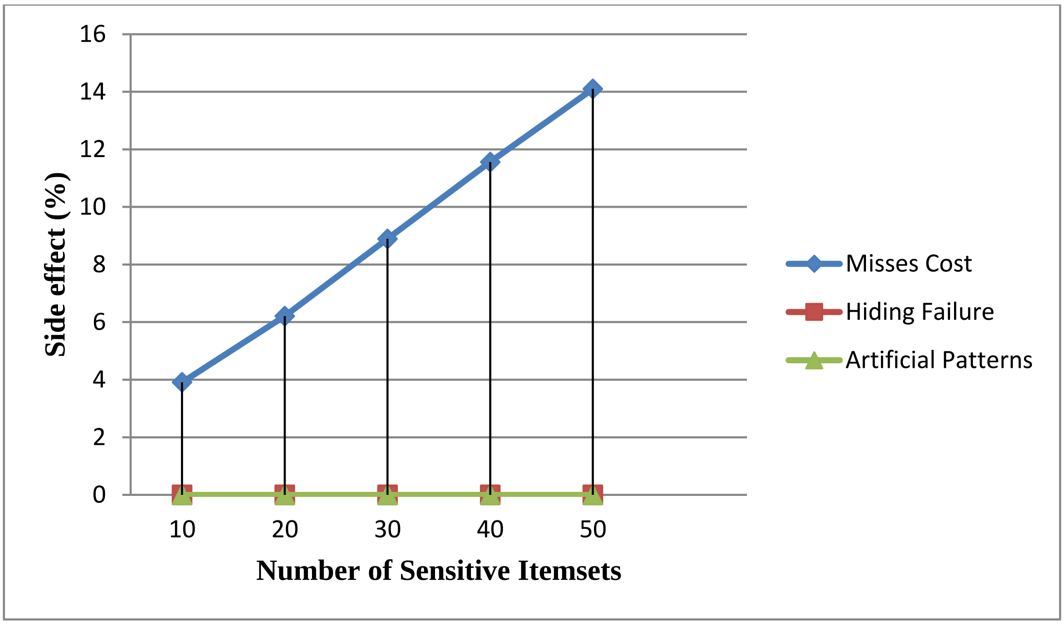
We have compared the performance of the proposed algorithms with two existing heuristics based SPH approaches.

The effectiveness of a sanitizing algorithm depends on three factors. First is the algorithm's ability to hide all the sensitive itemsets. Secondly how many non-sensitive frequent patterns does it hide as a side-effect. And thirdly does the sanitized dataset generate any artifacts. The dataset used for comparison has 1 million transactions and MST is 1%.

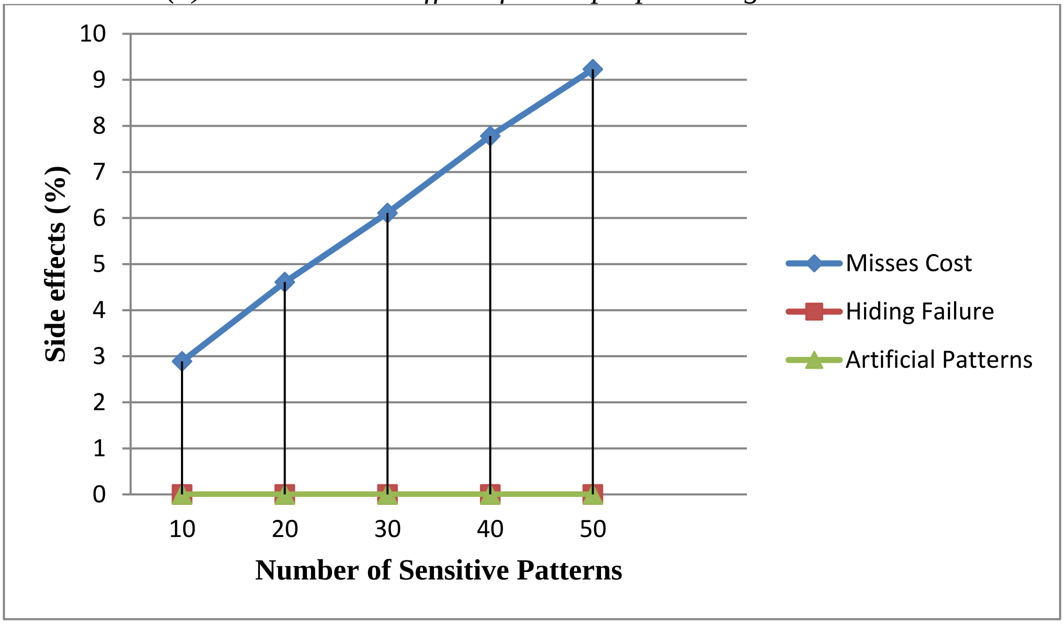
Table 4.2 shows the results obtained on analysing the proposed algorithms. Figure 4.1 displays the results obtained in the form of graphs.

Number of Sensitive Itemsets	Misses Cost (%)			
	Proposed PMSIR	Heuristics-based 1	Proposed PMSIRMD	Heuristics-based 2
10	3.91	7.78	2.89	6.29
20	6.21	10.89	4.61	9.23
30	8.89	14.87	6.11	12.99
40	11.56	17.78	7.78	14.87
50	14.10	22.12	9.23	19.89

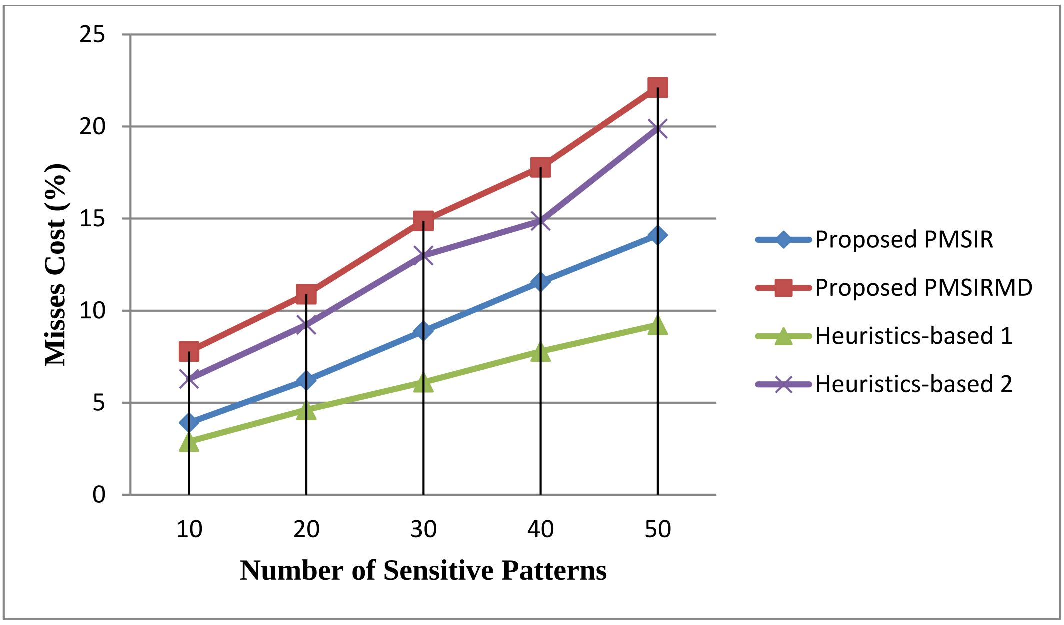
Table 4.2: *Misses cost for the proposed and existing algorithms. The dataset used contains 1000000 transactions and the MST is 1%.*



(a) Shows the side effects for the proposed algorithm PMSIR



(b) Shows the side-effects for proposed algorithm PMSIRMD



(c) Shows the misses cost comparison for the proposed and the existing algorithms

Figure 4.1: *Effect of the increase in the number of sensitive patterns on the side effects caused to the dataset; Number of transactions is 1 million and minimum support threshold is fixed as 1%.*

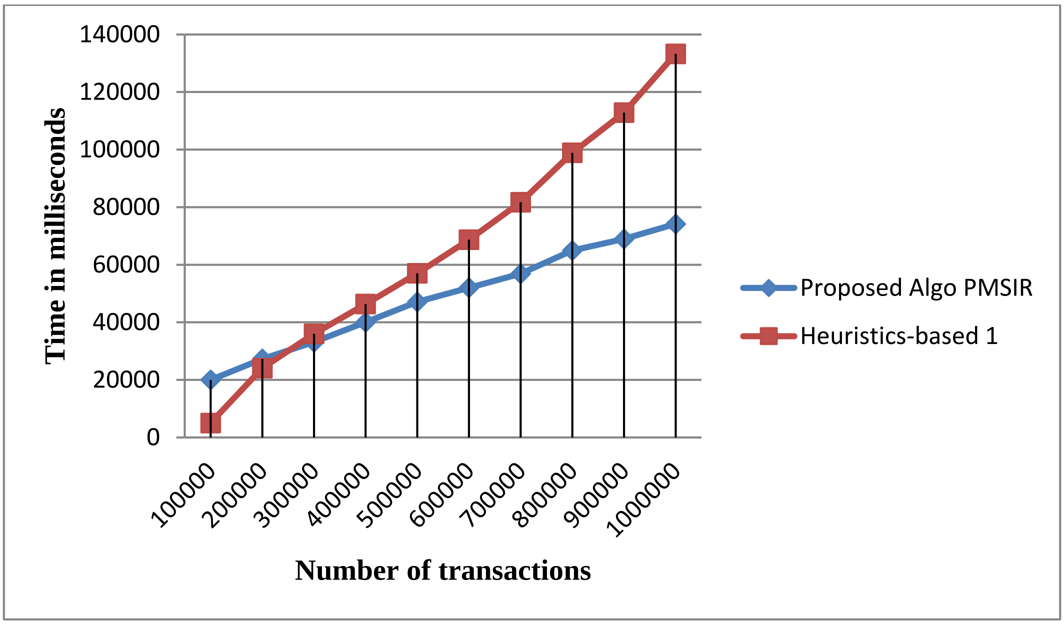
Figure 4.1 shows the cost in terms of side effects for both PMSIR and PMSIRMD.

For quantifying the scalability of our algorithms we tested their performance for increasing loads. We executed them on datasets of varying sizes to investigate how well the algorithms scaled as the number of transactions increased. We also executed them for different values of minimum support threshold. And we also tested their performance with increasing number of sensitive patterns. The first set of experiments was conducted for analysing the performance with increasing number of transactions. Table 4.3 displays the results. Figure 4.2 shows a comparison of PMSIR and PMSIRMD with heuristics-based 1 and heuristics-based 2. The number of sensitive itemsets here is 10 and minimum support threshold is 1%.

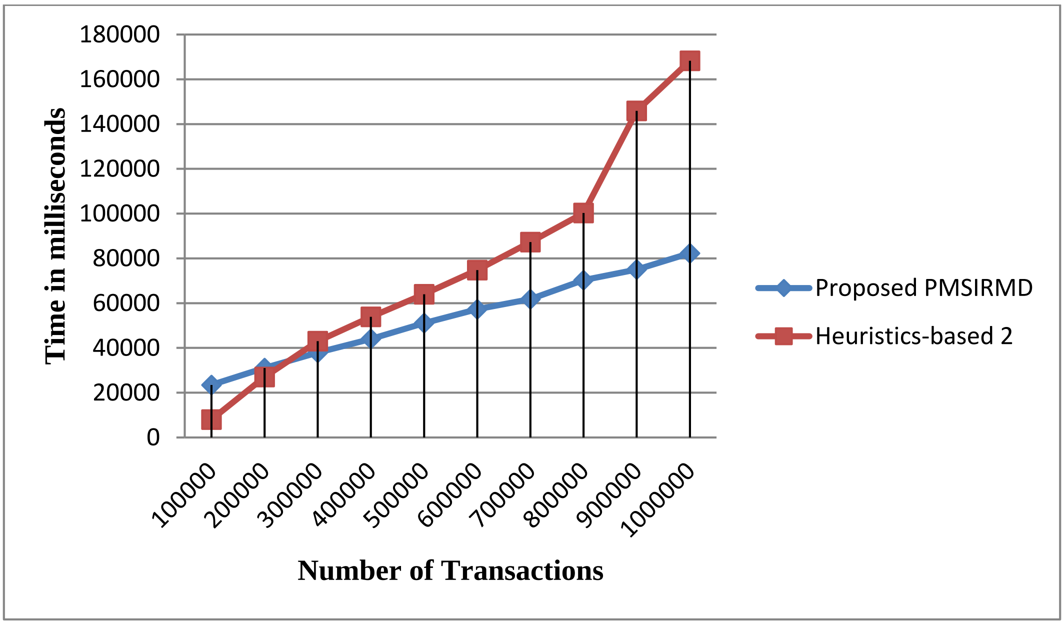
Number of Transactions	Running Time in Milliseconds			
	Proposed Algorithm PMSIR	Heuristics-based 1	Proposed Algorithm PMSIRMD	Heuristics-based 2
100000	19979	4988	23456	8002
200000	27265	24072	31023	27010
300000	33134	35984	37890	43002
400000	40039	46335	44007	53893

500000	47099	57012	51099	63980
600000	52003	68676	57291	74787
700000	56867	81737	61767	87267
800000	64933	98883	70273	100289
900000	68932	112829	75029	145871
1000000	74152	133266	82289	168239

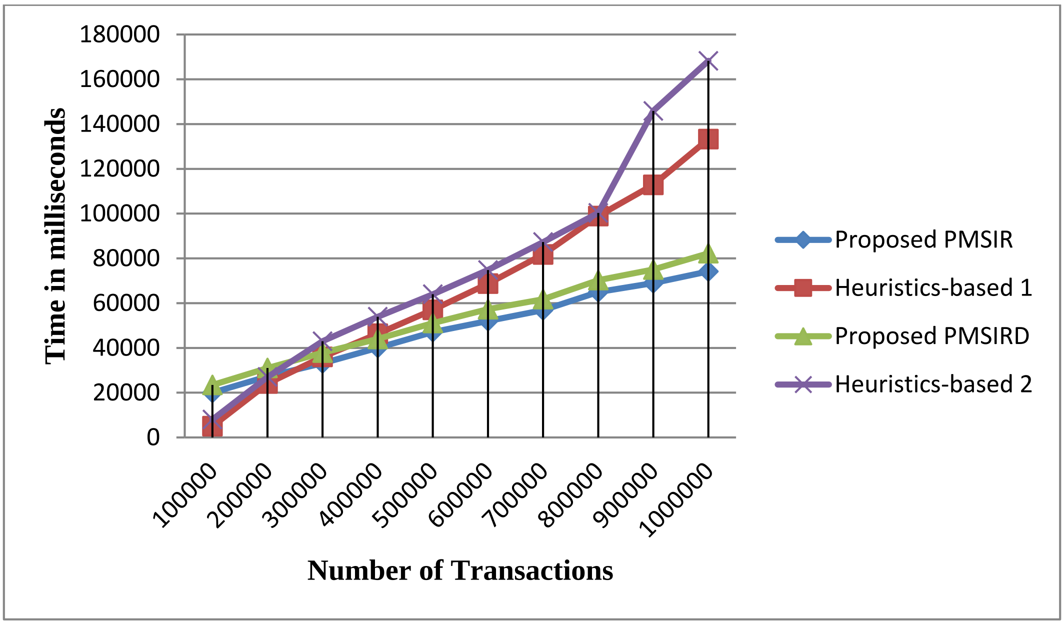
Table 4.3: *Running time obtained for the proposed algorithms and existing algorithms for increasing dataset size*



(a) Shows the running time comparison of PMSIR and Heuristics-based 1



(b) Shows the time complexity of PMSIRMD and Heuristics-based 2



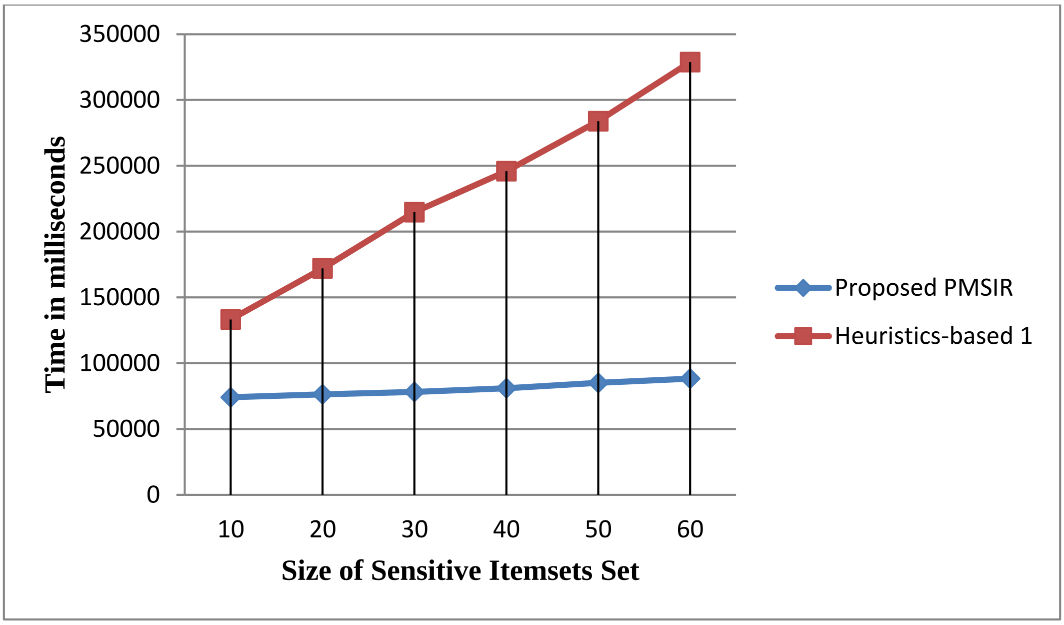
(c) Shows how the four algorithms perform under varying dataset size

Figure 4.2: *Effect of dataset size on the running time of the algorithms; Minimum Support Threshold is fixed as 1% ; Number of sensitive patterns is 10 and the set is fixed.*

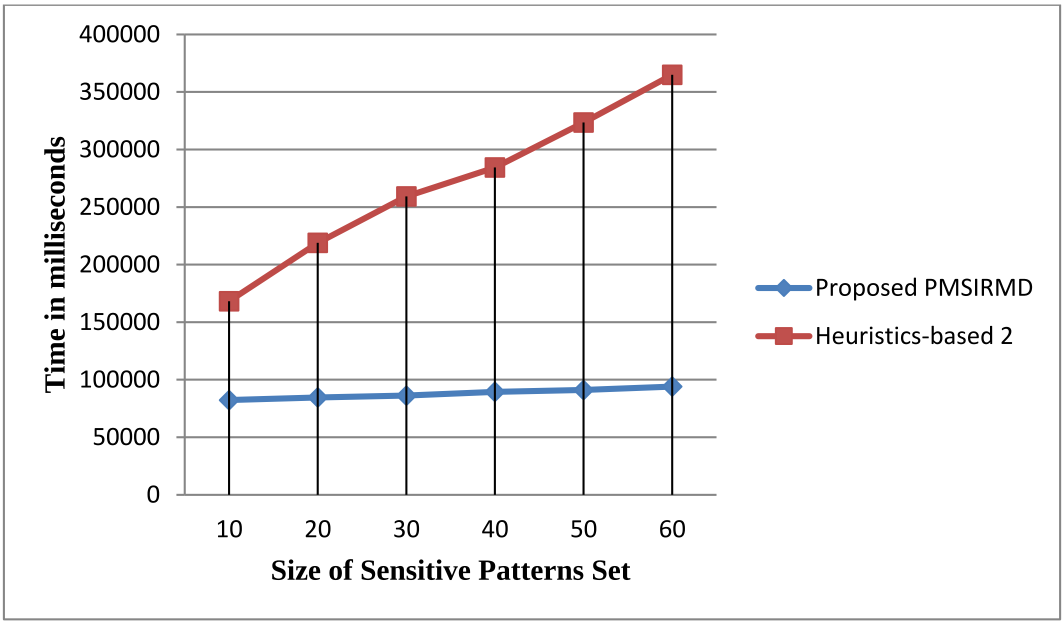
Now we analyse the performance of the algorithms when the number of sensitive itemsets increases. Table 4.4 displays the running time for varying number of sensitive itemsets. Figure 4.3 shows the variation in running time of all the four algorithms on increasing the number of sensitive itemsets. The size of the dataset used here is 1 million transactions and MST is 1%.

Number of Sensitive Itemsets	Running Time in Milliseconds			
	Proposed PMSIR	Heuristics-based 1	Proposed PMSIRMD	Heuristics-based 2
10	74152	133266	82289	168239
20	76394	172030	84582	218924
30	78204	214829	86202	259238
40	81020	245879	89459	284391
50	85102	283910	91022	323489
60	88320	328724	94025	364910

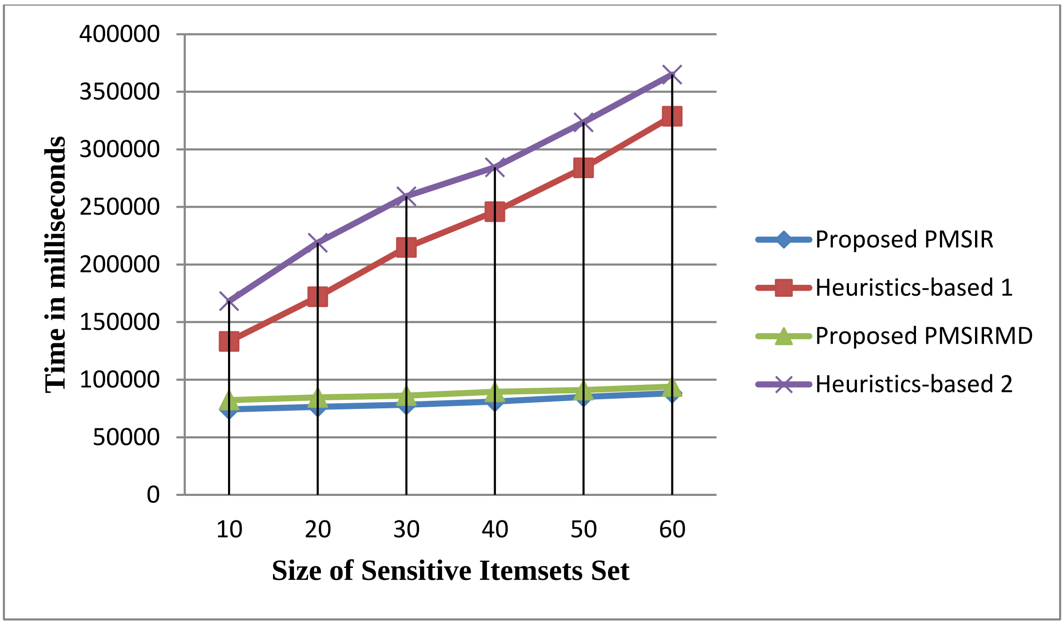
Table 4.4: *Running time for varying number of sensitive itemsets*



(a) Shows the running time comparison between Proposed PMSIR and Heuristics-based 1;



(b) Shows the running time comparison of Proposed PMSIRMD and Heuristics-based 2



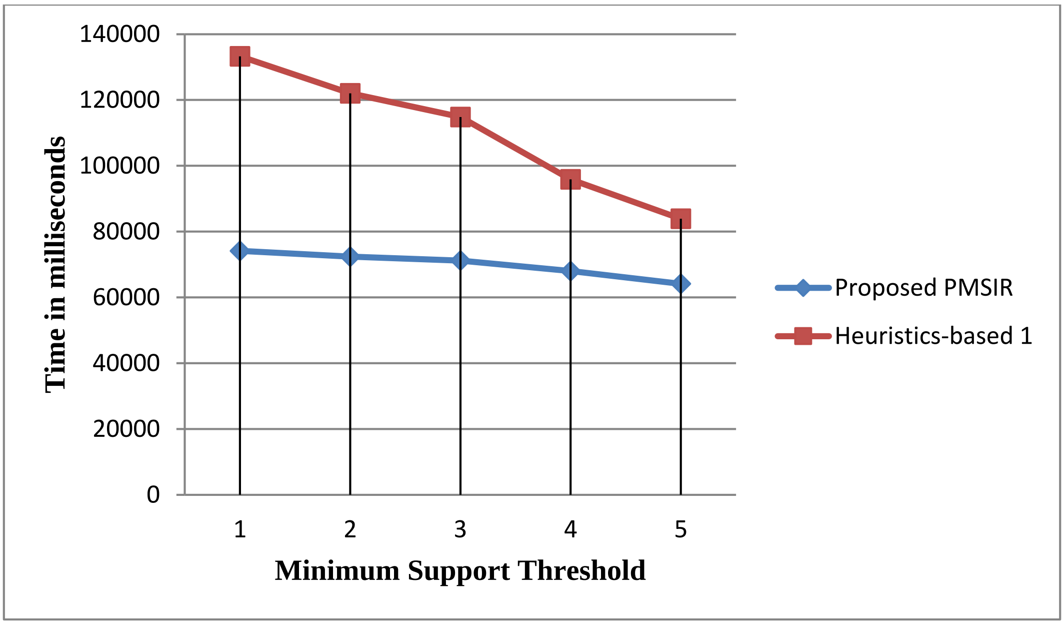
(c) Shows how the four algorithms perform under varying size of sensitive itemsets set

Figure 4.3: *Effect of increase in the number of sensitive patterns on the time complexity of the algorithm; Size of dataset is 1 million transactions; Minimum support threshold is 1 %.*

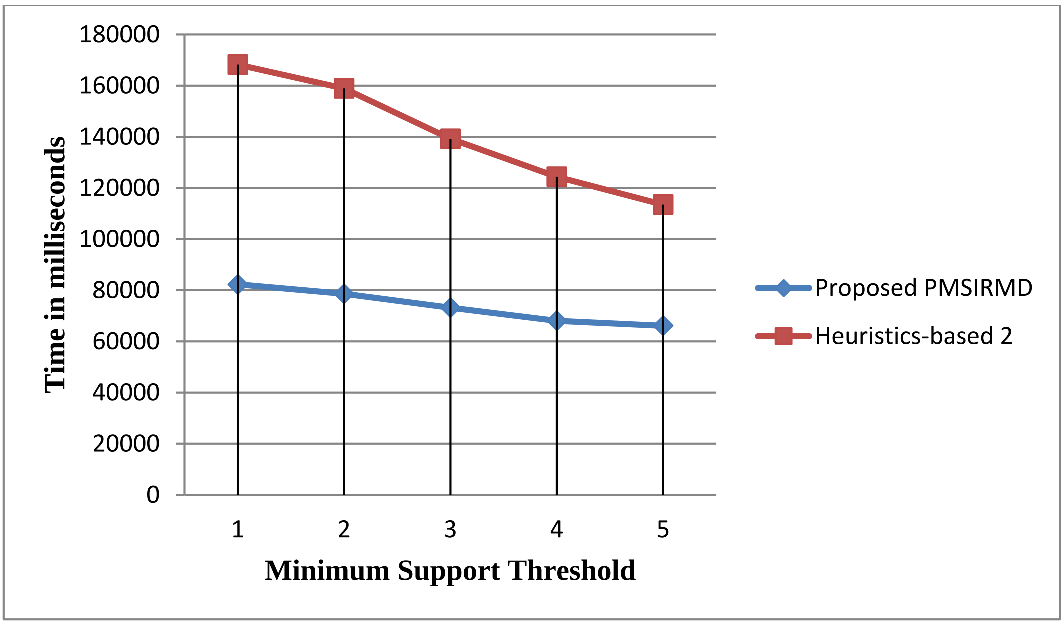
Next we observe the performance of algorithms under changing minimum support threshold values. Table 4.5 displays the running time needed under varying MST values. And figure 4.4 displays the results in form of graphs. The MST is varied from 1% to 5%. The SD used here has 1000000 transactions and the number of sensitive itemsets is 10.

Minimum Support threshold	Running Time in Milliseconds			
	Proposed PMSIR	Heuristics-based 1	Proposed PMSIRMD	Heuristics-based 2
1	74132	133266	82289	168239
2	72394	122030	78593	158924
3	71204	114829	73202	139238
4	68020	95879	68059	124391
5	64132	83910	66132	113489

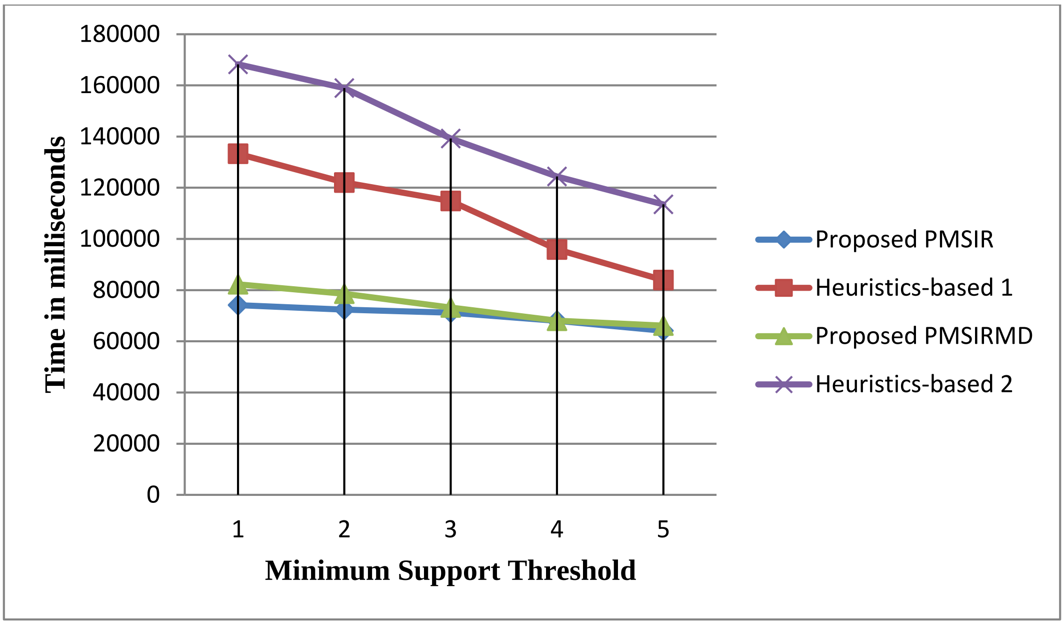
Table 4.5: *Running time under varying MST values.*



(a) Shows the running time of Proposed PMSIR and Heuristics-based 1 under varying MST values.



(b) Shows the running time complexity of Proposed PMSIRMD and Heuristics-based 2 under varying MST values



(c) Shows how the four algorithms perform under varying MST values.

Figure 1.4 *Effect of minimum support threshold on the time complexity of the algorithm; Number of transactions is 1 million; Number of sensitive patterns is 10.*

Figure 4.4 shows the effect of variation in minimum support threshold on the running time of the algorithms. The size of the dataset used here is 1 million transactions and size of sensitive itemsets set is 10.

4.3 Experiments on Real Dataset

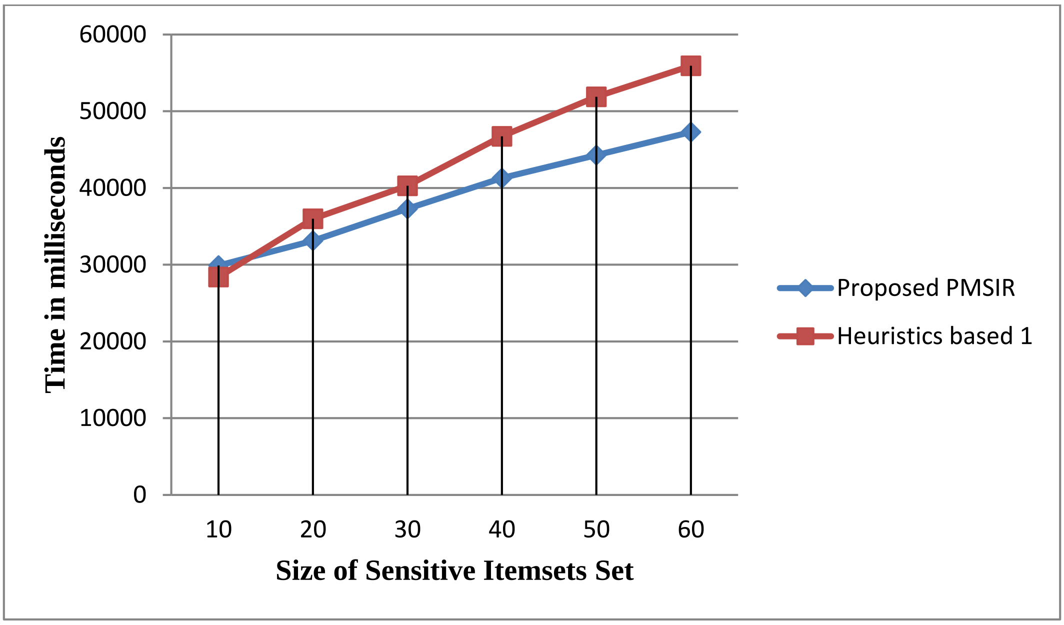
Accidents dataset is a benchmark dataset for frequent pattern mining. We have conducted various experiments on the dataset to analyze the performance of PMSIR and PMSIRMD.

For experiments we have chosen the set of sensitive itemsets randomly from the set of frequent patterns. In order to derive the frequent patterns we have used a parallel fast implementation of apriori algorithm [16].

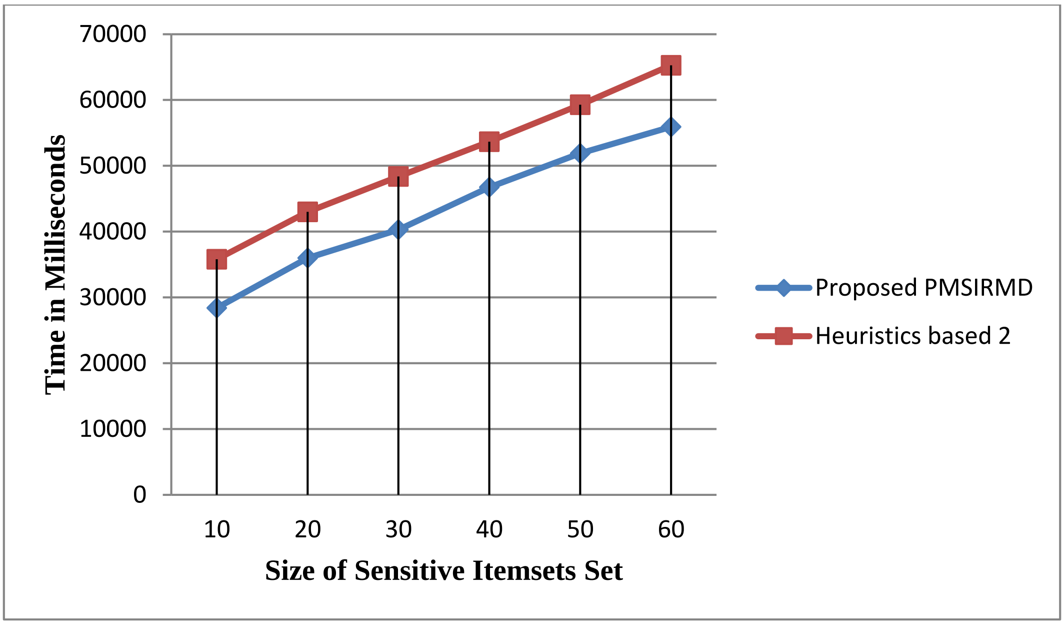
Similar to the analysis done in case of SDs here also we have tested the algorithms under different loads. Figure 4.5 depicts the performance of the algorithms in case of increasing number of sensitive itemsets. The MST here is 1%. Figure 4.5 (c) depicts this. The results are more or less in correspondence to the observations made on the SDs. Table 4.6 gives the running time observed for different number of sensitive itemsets.

Number of Sensitive Itemsets	Running Time in Milliseconds			
	Proposed PMSIR	Heuristics-based 1	Proposed PMSIRMD	Heuristics-based 2
10	29891	28392	32810	35792
20	33134	35984	37890	43002
30	37291	40292	40392	48379
40	41291	46739	43291	53672
50	44291	51892	46281	59272
60	47291	55922	48372	65278

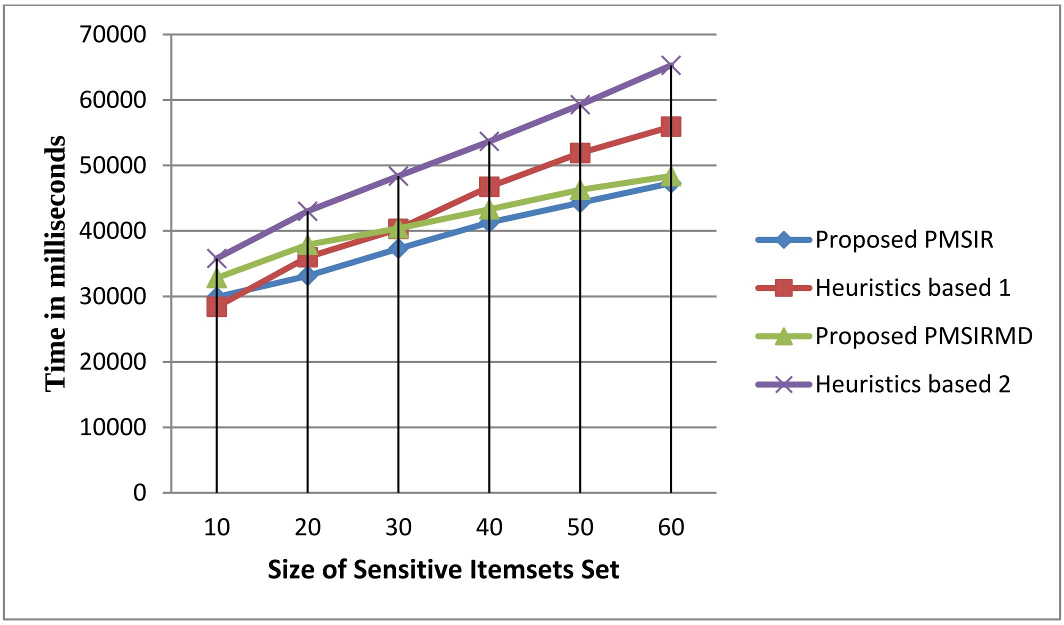
Table 4.6: *Running time for the proposed and existing algorithms under varying number of sensitive itemsets*



(a) Shows the running time of Proposed PMSIR and Heuristics based 1



(b) Shows the time complexity of Proposed PMSIRMD and Heuristics based 2



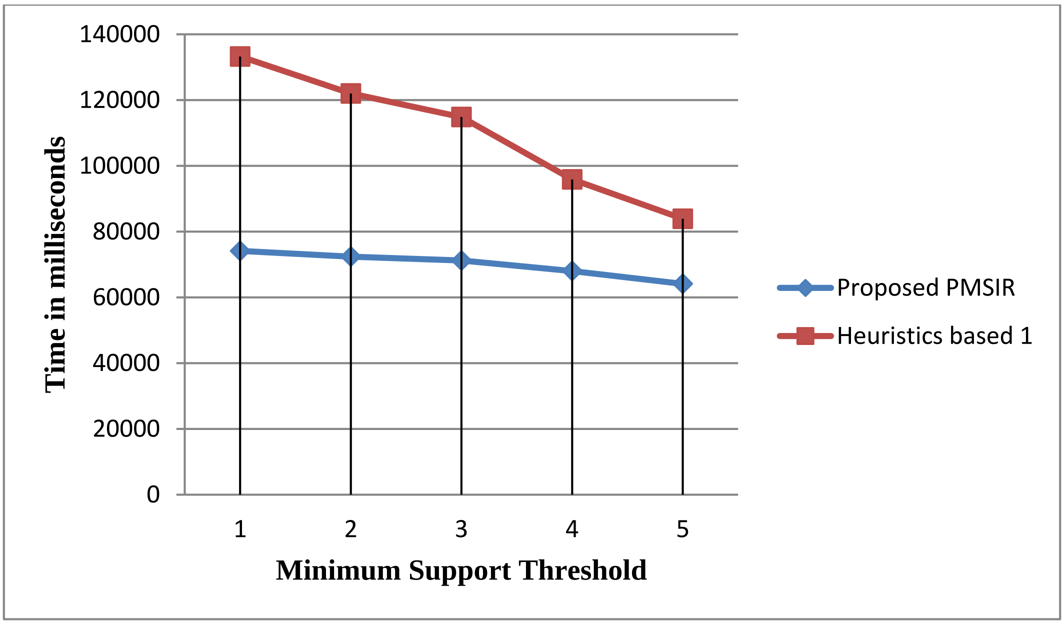
(c) Shows how the four algorithms perform under varying size of sensitive itemsets

Figure 4.5: *Effect of increase in the number of sensitive patterns on the time complexity of the algorithm; Minimum support threshold is fixed as 1%;*

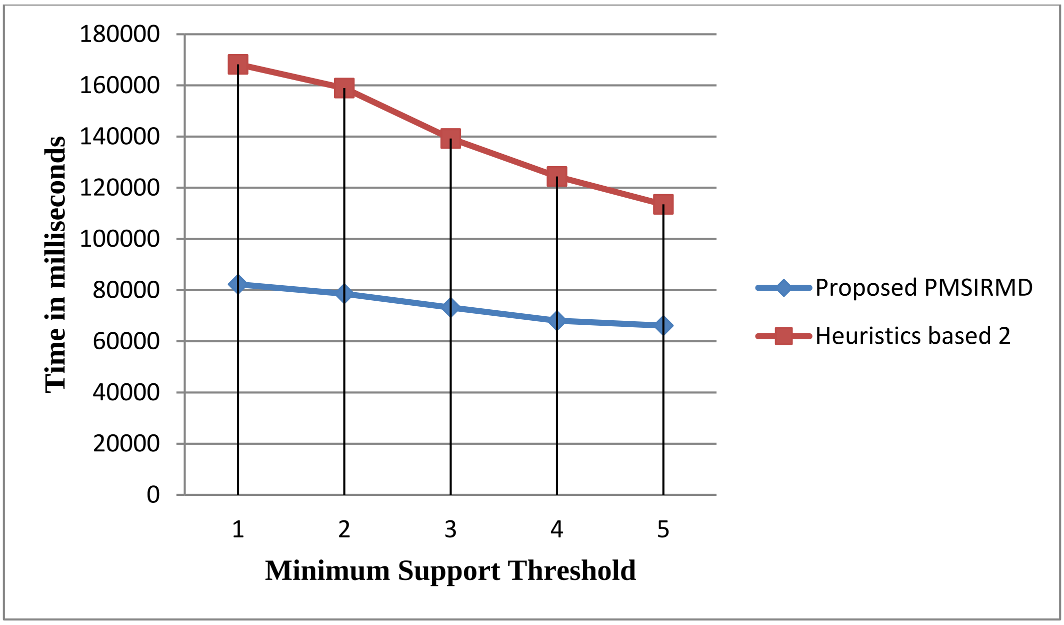
Table 4.7 gives the results obtained on varying the minimum support threshold. Figure 4.6 shows the results obtained when the value of support is varied from 1% to 5%.

Minimum Support Threshold (%)	Running Time in Milliseconds			
	Proposed PMSIR	Heuristics based 1	Proposed PMSIRMD	Heuristics based 2
1	74132	133266	82289	168239
2	72394	122030	78593	158924
3	71204	114829	73202	139238
4	68020	95879	68059	124391
5	64132	83910	66132	113489

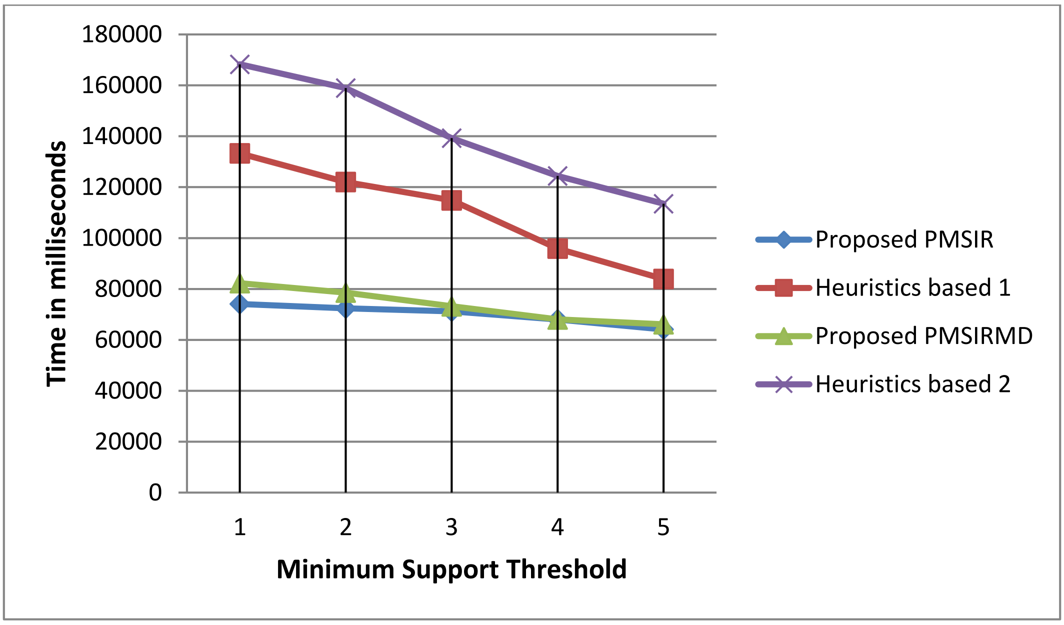
Table 4.7: *Running time required for AD when MST is varied*



(a) Shows the running time comparison of Proposed PMSIR and Heuristics based 1



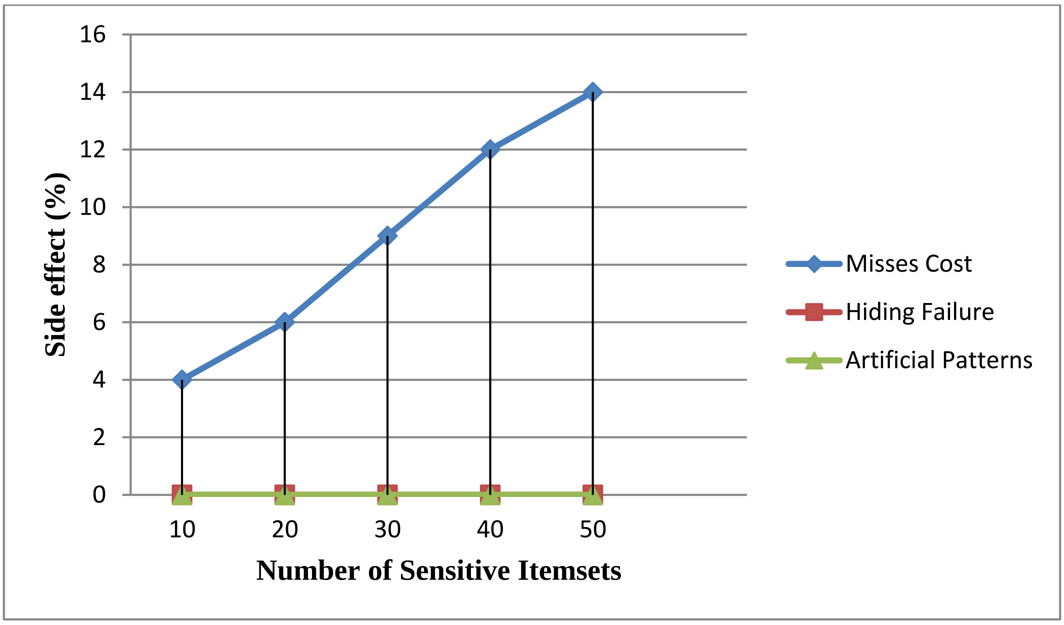
(b) Shows the running time comparison of Proposed PMSIRMD and Heuristics based 2



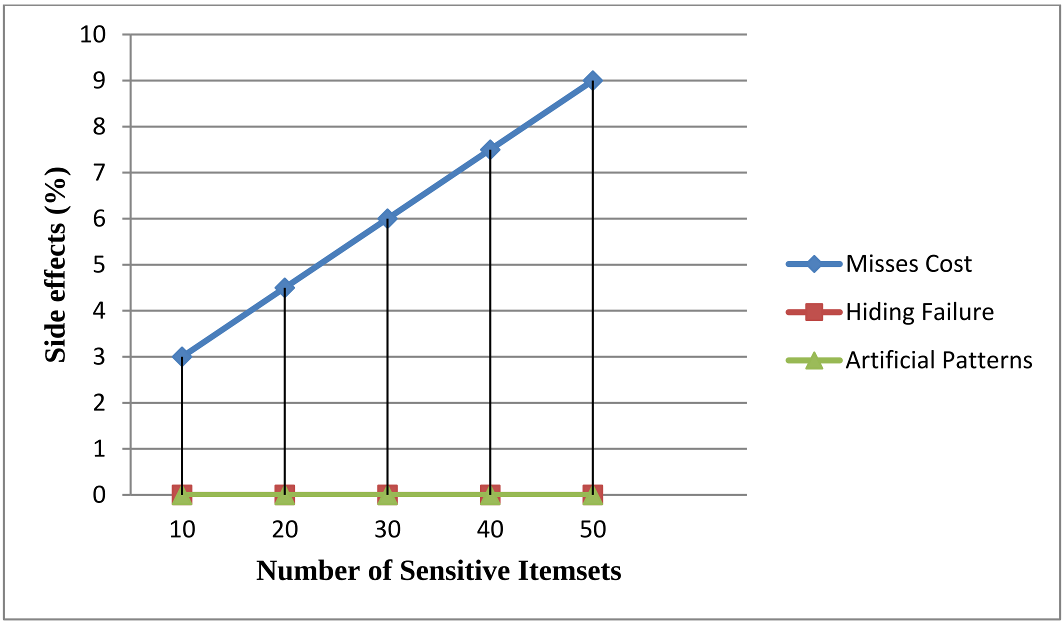
(c) Shows how the four algorithms perform under varying minimum support threshold values

Figure 2.6: *Effect of minimum support threshold on the time complexity of the algorithm; Number of sensitive patterns is 10.*

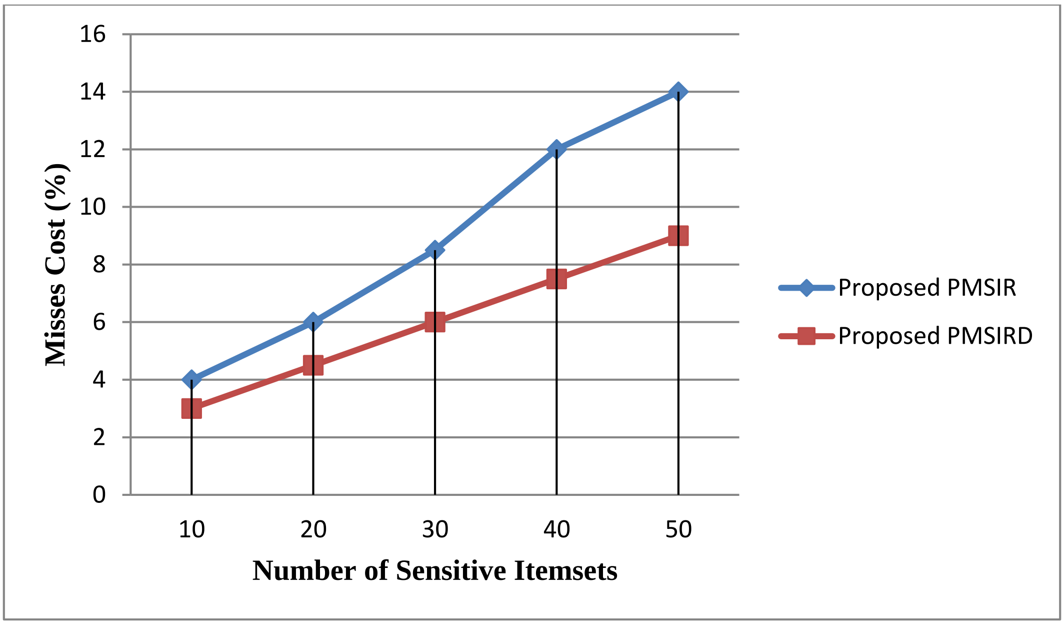
Figure 4.6 shows the behavior of the algorithms under varying support. The results are in correspondence to what was observed in case of SDs. In this case also PMSIR performs better in terms of running time.



(a) Shows the side effects caused using Proposed PMSIR



(b) Shows the side effects using Proposed PMSIRMD



(c) Shows the misses cost comparison for Proposed PMSIR and Proposed PMSIRMD

Figure 4.7: *Effect of the increase in the number of sensitive patterns on the side effects caused to the dataset; Minimum support threshold is 1%.*

Figure 4.7 describes the side effects caused by the sanitization process on the dataset in terms of misses cost, hiding failure and the number of artificial patterns. For both algorithms PMSIR and PMSIRMD hiding failure and the number of artificial patterns is zero. The only side effect caused is the masking of non sensitive frequent patterns i.e. misses cost. As evident from figure 4.7 (c) PMSIRMD performs better in this respect.

Chapter 5 Conclusion

5.1 Conclusion

The existing SPH approaches are either very complex and have huge running time costs or cause side effects to the data. Out of the three types of SPH approaches, exact approaches and border based approaches are very complex and costly where as heuristic approaches are fast but cause side effects. Here we have focused on heuristic approaches as they are fast. The aim is to use new heuristics that cause lesser side effects.

Here we have proposed two heuristic based sensitive pattern hiding algorithms PMSIR and PMSIRMD based on Hadoop MapReduce Framework that can hide sensitive patterns from data in parallel. We have evaluated our algorithms by using synthetic datasets as well as a real dataset. During our analysis it was observed that the grouping together of patterns based on victim item causes a great reduction in the ill effects caused to the dataset. Also for sufficiently large datasets our algorithms perform better than the existing pattern hiding algorithms in terms of running time cost. Proposed algorithms maintain better data quality than most of the existing SPH approaches.

Amongst the two algorithms PMSIR and PMSIRMD, PMSIR performs better than PMSIRMD in terms of time but PMSIRMD maintains better data quality by taking the degree of transactions into account. We have tested the performance of the algorithms extensively by executing them under varying parameters.

5.2 Future Work

With the use of more complex heuristics and by taking into account the effect on the non-sensitive patterns while choosing the victim item we can achieve better data quality than the currently achieved data quality. And this is what can be done in future. Integration of the border based hiding approach to our existing heuristic approach is also one potential direction that can be examined. These algorithms can be extended to come up with more efficient parallel sensitive hiding approaches in the future that can make the analysis of humungous volumes of data easier and more efficient.

References

[1] Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Saygin, Y., Theodoridis, Y.: State-of-the-Art in Privacy Preserving Data Mining. In: ACM SIGMOD Record. vol. 3, is-sue. 1, pp. 50-57,(2004).

[2] S.R.M. Oliveria and O. R. Zaiane, “Privacy preserving frequent itemset mining,” in proceedings of the IEEE international conference on Privacy, security and data mining, pp 43-54, 2002.

[3] J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” Communications of the ACM, Vol. 51, No. 1, 2008.

[4] Atallah, M., Bertino, E., Elmagarmid, A. K., Ibrahim, M., Verykios, V. S.: Disclosure Limitation of Sensitive Rules. In: IEEE Knowledge and Data Engineering Exchange. pp. 45-52, Chicago, Illinois, USA (1999).

[5] Saygin, Y.,Verykios, V.S., Clifton,C.: Using unknowns to prevent discovery of association rules. In: ACM SIGMOD Record. vol. 30, issue 4, pp 45–54, New York, USA (2000).

[6] Modi, C.N., Rao, U.P., Patel, D.: Maintaining Privacy and Data Quality in Privacy Preserving Association Rule Mining. In: proceedings of the IEEE international conference on computing, communication and networking technologies,Karur, India(2010).

[7] Jain, D., Khatri, P.,Soni, R.,Chaurasia, B.K.: Hiding Sensitive Association Rules without Altering the Support of Sensitive Item(s). In:LNCS, vol. 84, pp. 500–509, (2012).

[8] Le, H., Arch-int, S., Nguyen, H., Arch-int, N.: Association rule hiding in risk management for retail supply chain collaboration. Computers in Industry. vol. 64, pp. 776-784 (2013).

[9]Moustakides, G.V.,Verykios, V.S.:A max-min approachforhiding frequent itemsets. IEEEData and Knowledgeengineering.vol. 65, issue 1, pp.75-89(2008).

[10] Lee, G., Chen, Y., Peng, S., Lin, J.: Solving the Sensitive Itemset Hiding Problem whilst Minimizing Side Effects on a Sanitized Database. Communications in Computer and Information Science. vol. 223, pp. 104-113, Taiwan (2011).

[11] Sun, X., Yu, P.S.: A border-based approach for hiding sensitive frequent itemsets. In: IEEE International Conference on Data Mining, pp. 426-433, 2005.

[12] Gkoulalas-Divanis, A., Verykios, V.S.: An integer programming approach for frequent itemset hiding. In: ACM Conference on Information and Knowledge Management, pp. 748-757, New York, USA (2006).

[13] R. Agarwal, R. Srikant, “Fast Algorithms for Mining Association Rules,” IBM Research Center, 1994.

[14] X. Y. Yang, Z. Liu, Y. Fu, “MapReduce as a Programming Model for Association Rules Algorithm on Hadoop,” IEEE Transactions on Information Sciences and Interaction Sciences, pp 99-102, 2010.

[15] L. Li, M. Zhang, “The Strategy of Mining Association Rule Based on Cloud Computing,”

Proceeding of IEEE, International Conference on Business Computing and Global Informatization, Washington, DC, USA, pp 475- 478,2011.

[16] O. Yahya, O. Hegazy, E. Ezat, “An Efficient implementation of Apriori Algorithm based on Hadoop- MapReduce Model,” International Journal of Reviews in Computing, Vol 12, 2012.

[17] Z. Farzanyar, N. Cercone, “Efficient mining of frequent itemsets in social network data based on MapReduce framework,” ACM International Conference on Advances in Social Network Analysis, pp 1183-1188, 2013.

Publications

[1] Nishtha Behal, Durga Toshniwal, “Heuristics based Sensitive Pattern Hiding based on Hadoop MapReduce Framework”, 27th International Conference on Database and Expert Systems Applications, Portugal, September 5 - 8, 2016. *[Communicated]*