Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 45 (2015) 133 – 142
Procedia Computer Science

International Conference on Advanced Computing Technologies and Applications (ICACTA-2015)

Frequent Item set Mining using INC_MINE in Massive Online Analysis Frame work

Prof.Dr.P.K.Srimani[a], Mrs. Malini M. Patil[b]*

[a]*Former Chairman and Director, R & D, Bangalore University, Karnataka, India*
[b]*Assistant Professor , Dept of ISE , J.S.S. Academy of Technical Education, Bangalore-560060, Karnataka, India*
Research Scholar, Bharthiar University, Coimbatore, Tamilnadu

Abstract

Frequent Pattern Mining is one of the major data mining techniques, which is exhaustively studied in the past decade. The technological advancements have resulted in huge data generation, having increased rate of data distribution. The generated data is called as a 'data stream'. Data streams can be mined only by using sophisticated techniques. The paper aims at carrying out frequent pattern mining on data streams. Stream mining has great challenges due to high memory usage and computational costs. Massive online analysis frame work is a software environment used to perform frequent pattern mining using INC_MINE algorithm. The algorithm uses the method of closed frequent mining. The data sets used in the analysis are Electricity data set and Airline data set. The authors also generated their own data set, OUR-GENERATOR for the purpose of analysis and the results are found interesting. In the experiments five samples of instance sizes (10000, 15000, 25000, 35000, 50000) are used with varying minimum support and window sizes for determining frequent closed itemsets and semi frequent closed itemsets respectively. The present work establishes that association rule mining could be performed even in the case of data stream mining by INC_MINE algorithm by generating closed frequent itemsets which is first of its kind in the literature.

Peer-review under responsibility of scientific committee of International Conference on Advanced Computing Technologies and Applications (ICACTA-2015).

Keywords: Data Streams; Data mining; Association Rule Mining; Frequent itemsets; Closed frequent itemsets; Rules;

* Corresponding author. Tel.:919343401796 ;
E-mail address: patilmalini31@gmail.com

Peer-review under responsibility of scientific committee of International Conference on Advanced Computing Technologies and Applications (ICACTA-2015).
doi:10.1016/j.procs.2015.03.105

1. Introduction

Data Mining (DM) has attained a lot of scope in the past two decades. Due to rapid advancements in IT industries and many real time systems have collected huge amount of data. Hence it has attracted researchers across the globe. Such data is referred to as a large data set or very large data base (VLDB). Few examples of real time applications cover wide areas like super market basket analysis, fraud detection etc. This has lead to the evolution of key basic data mining concepts and techniques (Classification, Clustering and Association Rule Mining) for discovering the hidden interesting data patterns in large data sets. DM is defined as the process of knowledge discovery from large databases (KDD) and data warehouses.

In the present era of technological growth, the increased rate of data generation is mainly due to different real time systems. To quote a few: mobile and sensor applications, network traffic monitoring systems, log record management, internet packet streams, web click streams, email records and twitter data etc. The data generated from these systems can be referred to as a data stream or streaming data. Data stream[1,2] is defined as an ordered sequence of items that arrive in a timely order. They are certainly different from traditional databases. Mining a stream data is referred to as Data Stream Mining (DSM). Data streams are huge size, fast changing nature and need of fast response and limited storage as only summary can be stored; random access of data is not possible. They are continuous, unbounded, having a very high speeds and changing data distribution with time and is explained in[1,2] exhaustively. The state of art in data stream mining is explained in[3]. Other important features of DSM are: their huge size and fast changing nature, random data access is not allowed, need fast response and only summary can be stored which needs limited storage and any time predictions can be made. Some of the key challenges of data streams are: multiple scans are not allowed in DSM. The mining methods of DSM should be able to handle the change in data distribution, should be faster than the speed of data stream as they arrive in high speeds. Issues related to data storage and CPU speed are also challenging. From the literature it is found that DSM also emphasis on the use of core data mining techniques related to classification, clustering, association rule mining regression modelling.

1.1 Types of Data Streams : Data streams can be classified into two types[4]. Authors have explained about different types of data streams. viz., static streams(SS) and evolving streams(ES). Static streams(offline streams) which arrive at regular bulk intervals. In a certain period of time most of the reports are generated in the case of Web logs and are considered as good examples. Yet another best example would be queries on data warehouses. The situation in the case of ES, the updated data arrives one by one. The best examples are: frequency estimation of internet packet streams, stock market data, and sensor data which require online processing. The most important feature of ES is that there should not be a mismatch between the processing and the rapid data arrival speeds. Further, it should be recalled that bulk data processing is not possible in ES as on the case of SS.

1.2 Data Stream processing Models: Data stream processing models are also the key features in data stream mining which are explained in[5]. They are *landmark model, damped model and sliding windows* model. The *landmark*[5] model generates frequent itemsets over the entire history of stream data from a specific time point called landmark to present. In the case of stock monitoring systems this model finds its application since people show great interest in the most recent information of data streams. In the case of *Damped*[5] model only the frequent items mines frequent items in data streams are mined where each transaction is attached with a weight which diminishes with age. Clearly the contributions from older transactions will be less weight towards the itemset frequencies. Therefore this model has its applications in case where old data have significant effect on the results of DM but the influence is temporary. Finally, it can be noted that in the case of *sliding window*[5] model, the sliding window model finds its applications in data streams and the size of the window and the applications are machine dependent.

1.3 Mathematical model for a data stream: Let D be a set of items. An itemset (or a pattern), $I = \{x_1, x_2, . .,x_k \}$, is a subset of D. An itemset consisting of n items is called a k-itemset and is written as $k_1, k_2, . .,k_n$. It is assumed that the items in an itemset are lexicographically ordered. A transaction is a tuple, (TID, Y), where TID is the ID of the transaction and Y is an itemset. A transaction data stream is a sequence of incoming transactions. From these streams an *excerpt is* taken for analysis, which is called as a *window.* Let W be the window which is either time-based or count-based according to the number of transactions that are updated each time and either a landmark window or a sliding window.

Rest of the paper is organized as follows: Section 2 focuses mainly on the related work in the area of data stream mining. Section 3 discusses about the preliminaries. Methods and Models are discussed in section 4. Section 5 is about experiments and the results. Future enhancement of the work and conclusions are briefed at the end of the paper.

2. Related Work

Last one decade has witnessed the study of mining frequent itemsets in static data bases. Some of the breakthrough algorithms in this direction are Apriori, FP-growth and have been proposed by[6]. The other findings in this direction that incremental mining of frequent itemsets in dynamic databases are proposed by[7] which proposes that all the all the frequent itemsets and their support counts derived from the original database are retained. The support counts of the frequent itemsets are recounted when transactions are added or deleted. All these methods have to rescan the original database because non-frequent itemsets can be frequent after the database is updated. Therefore, they cannot work without seeing the entire database and cannot be applied to data streams. Recently mining frequent itemsets over data streams are classified into two groups, mining frequent items and mining frequent itemsets. Most of them use the techniques including the data processing models. For mining frequent itemsets, Lossy-counting is the representative approach under the landmark model by[8]. Methods for frequent pattern mining in data streams using massive online analysis (MOA) frame work is proposed by[15]. The MOA framework is exhaustively presented by[11-13]. The other works of authors in MOA framework are[16-21]. Very sparse literature is available with regard to mining frequent itemsets in MOA frame work. The present investigation is first of its kind.

3. Preliminaries

Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction. Association Rule: It is an implication expression of the form $X \rightarrow Y$, where X and Y are itemsets. Example: {Milk, Wheat} → {Bread}.

3.1 Rule Evaluation Metrics: Support (s) and Confidence(c) are the two rule evaluation metrics. Support is fraction of transactions that contain both X and Y Confidence (c). Measures how often items in Y appear in transactions that contain X. Given a set of transactions T, the goal of association rule mining is to find all rules having support ≥ Minimum support (*min_sup)* threshold confidence ≥ Minimum Confidence(*min_conf)* threshold. Association rule mining is a two-step approach:

1. *Frequent Itemset Generation* :The objective of this step is to find all the itemsets that satisfy the minimum support(min_sup) threshold. These itemsets are called as frequent itemsets. It is computationally expensive than rule generation step.
2. *Rule Generation* : The objective of this step is to generate high confidence rules from each frequent itemset which are found in step 1. These rules are called as strong rules.

3.2 Closed Itemset: The study reveals[10] that the itemsets generated from a transaction data set is normally very large. The progress in of study in this area derived a useful method of generating a small set of itemsets from which all other frequent itemsets can be generated. It is named as closed frequent itemset. An itemset X is closed if none of its immediate supersets has exactly the same support as the itemset X. Pictorially the lattice structure is used to enumerate the list of all frequent and closed frequent itemsets. The transaction data set used for the generation of closed frequent itemsets is shown in fig.1 and is self explanatory. Let I={a,b,c,d,e} be the items. To illustrate the support count of each itemset, each node of lattice structure with the list of its corresponding transaction IDs. For Example, since the node {b,c} is associated with transaction Ids 1, 2 and 3, its support count is three. Every transaction that contains b also contain c. Support for {b} is identical to {b,c} and {b} should not be considered a closed itemset. Similarly, since c occurs in every transaction that contains both a and d, itemset {a,d} is not closed because it does not have the same support count as any of its supersets. The concept of frequent itemset and closed frequent itemset is exhaustively explained in[10,22] respectively.

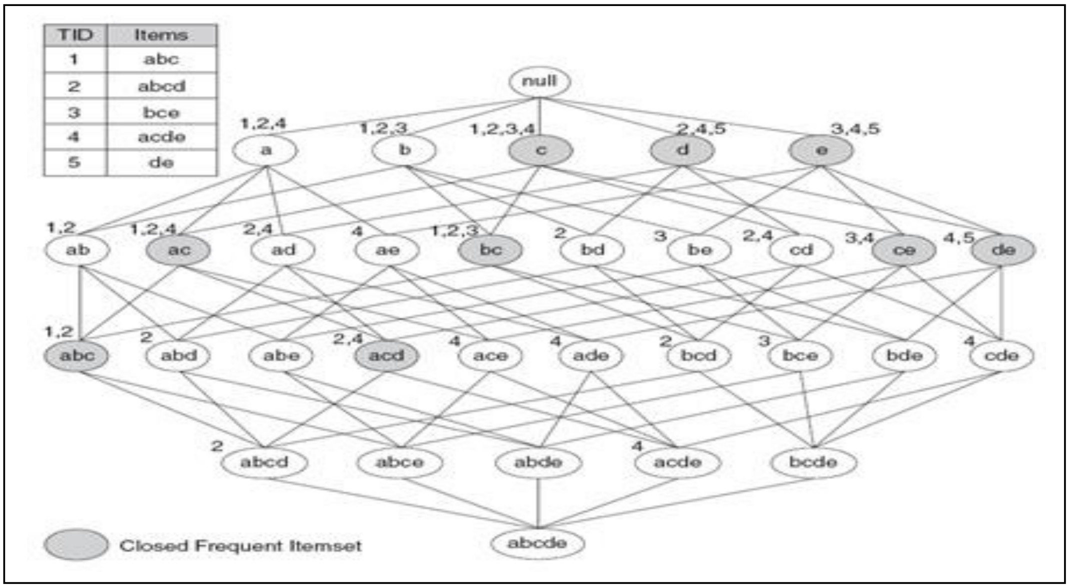


Fig.1. Lattice structure for closed frequent itemset.

4. Methods and Models

The different steps of methodology of mining frequent itemsets are discussed in this section. The methodology uses Massive Online Analysis (MOA) framework for generation of frequent itemsets using the algorithm INC_MINE. The data set generators are Electricity, Airline and Our_Generator.

4.1 Massive Online Analysis framework(MOA): Massive Online Analysis framework[11-13] is a open source software environment for mining massive, potentially infinite, evolving data streams. The experiments can be performed on MOA framework with different configurations, machine learning methods and evaluation methods on data streams. MOA is designed in such a way that it can handle the challenging problems of data streams. MOA is written in JAVA, taking advantage of its portability and well developed libraries. It consists of offline and online algorithms for classification and clustering. It also consists of tools for evaluation. It is also provided with regression analysis and outlier mining methods. Also specific extensions are provided in which a frequent pattern mining can also be conducted on data streams. MOA mainly permits the evaluation of data stream learning algorithms on large streams under explicit memory limits with high speed, without storing any intermediate data and only storing the summary data.

4.2 Data set generators: From the literature review it is found that there is a shortage of data sources for DSM. For the purpose of analysis the authors have selected the *Electricity* dataset and *Airline* data set from the available sources from MOA frame work. The authors also generated their own data set generator called as *our_generator.* The brief details about the data set generators are discussed in this section.

Electricity data set. It is a popular benchmark dataset[12] for testing adaptive classifiers. It has been used in over 40 concept drift experiments. The dataset covers a period of two years (45312 instances recorded every half an hour, 6 input variables). A binary classification task is to predict a rise (UP) or a fall (DOWN) in the electricity price in New South Wales (Australia). The prior probability of DOWN is 58%. The data is subject to concept drift due to changing consumption habits, unexpected events and seasonality. This dataset has an important property not to be ignored when evaluating concept drift adaptation.

The Airline data : This data set[12] consists of flight arrival and departure details for all commercial flights from 1987 to 2008. The approximately 120MM records occupy 120GB space. Few of the important aspects of this data set are : When is the best time of day/day of week/time of year to fly to minimize delays? Do older planes suffer more delays? How does the number of people flying between different locations change over time? How well does weather predict plane delays? Can you detect cascading failures as delays in one airport create delays in others?. This is a large dataset with nearly 120 million records. The dataset was cleaned and records were sorted according to the arrival/departure date and time of flight. Its final size is around 116 million records and 5.76 GB of memory. There are 13 attributes, each represented in a separate column: *Year, Month, Day of Month, Day of Week, CRS Departure Time, CRS Arrival Time, Unique Carrier, Flight Number, Actual Elapsed Time, Origin, Destination, Distance, and Diverted*. The target variable is the *Arrival Delay*, given in seconds.

Our_Generator: Generates the data streams using the random number function. The data generation method is based on the customer buying pattern in the market basket data. The number of items is assumed as 26. The data is outputted in the form of flat file which mainly includes maximum column width (≥ 5 and ≤ 26) minimum column width(assumed as 4 always). Number of items is constant (26). The total number of items in one transaction vary from minimum column width to maximum column width. Using the function {elements_count = random( ) % maxi_trans + mini_trans} the required number of transactions are generated. Once the data stream is generated the concept drift is introduced artificially by adding noise (40%) for evaluation purpose. An effective code is designed in order to generate the data stream of 'n' number of instances, where n=200,000 in the present work.

4.3 INC_MINE algorithm

INC_MINE[15] is an algorithm for incremental update of frequent closed itemsets(FCIs) over data streams. The algorithm is provided as an extension in MOA framework.

```
Algorithm    Closed_Subpattern_Mining_Add(T1, T2, minsup, T)
T ← T1
foreach t in T2 in size-ascending order do
    if t is closed in T1 then
        supT(t) ← supT(t) + supT2(t)
        foreach t' that is a subpattern of t do
            if t' ∈ T1 then
                if sup(t') is not updated then
                    insert t' into T
                    supT(t') ← supT(t') + supT2(t')
                else
                    skip processing t' and all its subpatterns
    else
        foreach t' that is a subpattern of t do
            if sup(t') is not updated then
                if t' ∈ T1 then
                    supT(t') ← supT(t') + supT2(t')
                if t' is closed then
                    insert t' into T
                    supT(t') ← supT(t') + supT2(t')
            else
                skip processing t' and all its subpatterns
delete from T patterns with support below minsup
```

Table 1. Transactions in a stream of 3 time units

T1	T2	T3
pqrs	pqr	pqrs
pqrx	pqrs	pz
qry		Apq
...	.....	.....
....	....	....

Fig.2: Algorithm to generate closed patterns.

4. Experiments , Results and Discussions

The experiment was conducted in MOA frame work using INC_MINE algorithm. The different settings for the experiment are shown in the configuration model in MOA frame work in fig.3 which is self explanatory. The results are tabulated in table 3 and 4 respectively. The parameters appearing in the experimental analysis are Window

Size(window_size), Minimum Support (min_sup) and maximum instance Size (max_instance_size). In the case of varying window_size the FCIs are determined for a fixed min_sup. In the case of varying min_sup(0.1,0.2,0.3,0.4 and 0.5) the FCIs and semiFCIs are determined for a variable window size. In the experiments 5 samples of instance size (10000,15000,25000,35000,50000) are used. For the experimental purpose three generators viz., airline, electricity and our generators are used.

The observations made from table 2 are: The window size is kept constant = 10 and the min_sup is varied from 0.1 to 0.5. All the three generators except the electricity behave in a similar fashion. The variation in min_sup causes a drastic change in the values of time. Our generator also shows considerable difference in the values of FCI and semiFCI. The observations made from table 3 are: Irrespective of the max_inst_size all the three generators except 'electricity' predict the same FCI and semiFCI values (16, 17 ; 10,14) respectively. For the size of 25000 airline generator behaves in a slightly different manner(14,17). But the electricity generator shows considerable variations as the instance size is increased. In the case of our generator, the effect of the increase in the window_size has its influence. The performance of INC_MINE algorithm is graphically shown in the figures 5a, 5b, 5c, 5d, 5e and 5f respectively and are self explanatory.

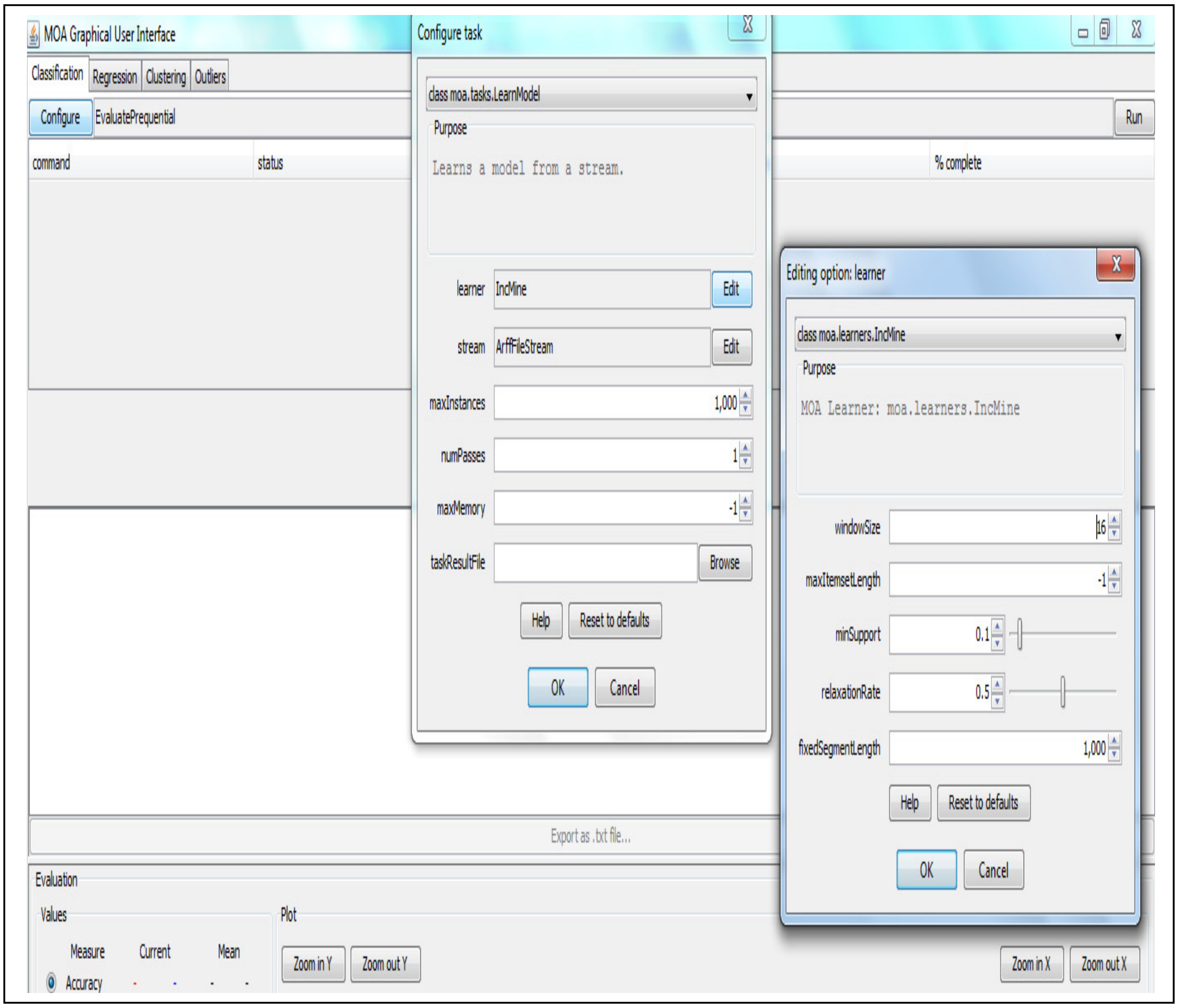


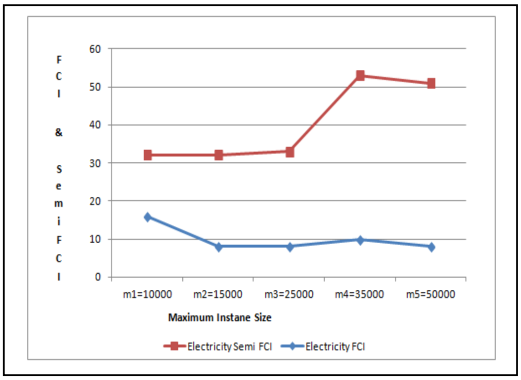
Fig.3. MOA configuration for INC_MINE.

Table 2: Results for 3 data generators for Window_size= 10

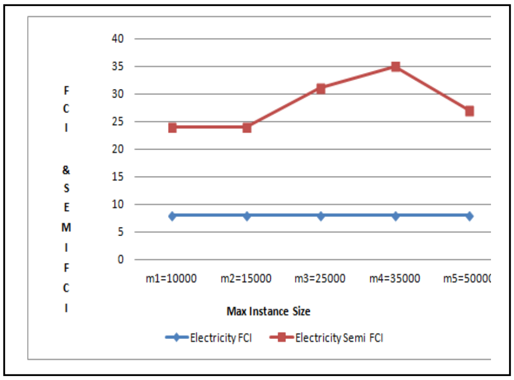
AIRLINE				**ELECTRICITY**				**OUR_GENERATOR**			
Min_sup=0.1, Max_inst_size= 10000				**Min_sup=0.1, Max_inst_size= 10000**				**Min_sup=0.1, Max_inst_size= 10000**			
Window_size	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**
10	**0**	**16**	**17**	**10**	**31**	**16**	**16**	**10**	**9**	**10**	**14**
20	**0**	**16**	**17**	**20**	**31**	**16**	**16**	**20**	**6**	**10**	**14**
30	**0**	**16**	**17**	**30**	**31**	**16**	**16**	**30**	**0**	**10**	**14**
40	**0**	**16**	**17**	**40**	**31**	**16**	**16**	**40**	**0**	**10**	**14**
50	**0**	**16**	**17**	**50**	**31**	**16**	**16**	**50**	**0**	**10**	**14**
Min_sup=0.1, Max_inst_size= 15000				**Min_sup=0.1, Max_inst_size= 15000**				**Min_sup=0.1, Max_inst_size= 15000**			
Window_size	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**
10	**0**	**16**	**17**	**10**	**4**	**8**	**24**	**10**	**14**	**10**	**15**
20	**0**	**16**	**17**	**20**	**4**	**8**	**24**	**20**	**14**	**10**	**15**
30	**0**	**16**	**17**	**30**	**0**	**8**	**24**	**30**	**0**	**10**	**15**
40	**0**	**16**	**17**	**40**	**0**	**8**	**24**	**40**	**0**	**10**	**15**
50	**0**	**16**	**17**	**50**	**0**	**8**	**24**	**50**	**0**	**10**	**15**
Min_sup=0.1, Max_inst_size= 25000				**Min_sup=0.1, Max_inst_size= 25000**				**Min_sup=0.1, Max_inst_size= 25000**			
Window_size	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**
10	**15**	**14**	**17**	**10**	**3**	**8**	**45**	**10**	**8**	**10**	**14**
20	**0**	**14**	**17**	**20**	**0**	**8**	**45**	**20**	**6**	**10**	**14**
30	**15**	**14**	**17**	**30**	**0**	**8**	**45**	**30**	**3**	**10**	**14**
40	**0**	**14**	**17**	**40**	**0**	**8**	**45**	**40**	**0**	**10**	**14**
50	**0**	**14**	**17**	**50**	**0**	**8**	**45**	**50**	**0**	**10**	**14**
Min_sup=0.1, Max_inst_size=35000				**Min_sup=0.1, Max_inst_size=35000**				**Min_sup=0.1, Max_inst_size=35000**			
Window_size	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**
10	**0**	**16**	**17**	**10**	**0**	**10**	**43**	**10**	**5**	**10**	**14**
20	**0**	**16**	**17**	**20**	**0**	**10**	**43**	**20**	**0**	**10**	**14**
30	**0**	**16**	**17**	**30**	**0**	**10**	**43**	**30**	**0**	**10**	**14**
40	**0**	**16**	**17**	**40**	**0**	**10**	**43**	**40**	**0**	**10**	**14**
50	**0**	**16**	**17**	**50**	**0**	**10**	**43**	**50**	**0**	**10**	**14**
Min_sup=0.1, Max_inst_size=50000				**Min_sup=0.1, Max_inst_size=50000**				**Min_sup=0.1, Max_inst_size=50000**			
Window_size	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**	**Window_size**	**Time(ms)**	**FCI**	**Semi-FCI**
10	**0**	**16**	**17**	**10**	**0**	**8**	**43**	**10**	**3**	**10**	**14**
20	**0**	**16**	**17**	**20**	**0**	**8**	**43**	**20**	**0**	**10**	**14**
30	**0**	**16**	**17**	**30**	**0**	**8**	**43**	**30**	**0**	**10**	**14**
40	**0**	**16**	**17**	**40**	**0**	**8**	**43**	**40**	**0**	**10**	**14**
50	**0**	**16**	**17**	**50**	**0**	**8**	**43**	**50**	**0**	**10**	**14**

Table 3: Results for 3 data generators for min_sup = 0.1

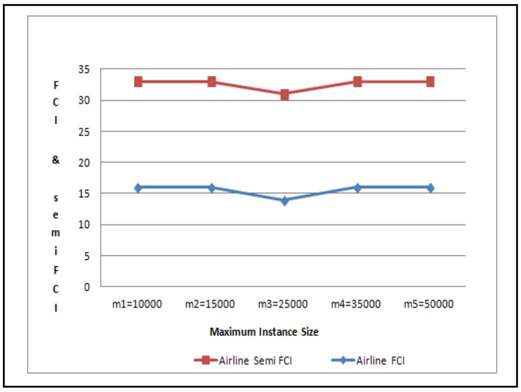
AIRLINE				ELECTRICITY				OUR_GENERATOR			
Window_size=10, Max_inst_size= 10000				Window_size=10, Max_inst_size=10000				Window_size=10, Max_inst_size= 10000			
Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI
0.1	16	16	17	0.1	4	8	16	0.1	16	7	23
0.2	16	16	17	0.2	16	8	16	0.2	16	7	23
0.3	0	16	17	0.3	16	8	16	0.3	0	7	23
0.4	16	16	17	0.4	0	8	16	0.4	0	7	23
0.5	16	16	17	0.5	0	8	16	0.5	0	7	23
Window_size=10, Max_inst_size=15000				Window_size=10, Max_inst_size=15000				Window_size=10, Max_inst_size=15000			
Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI
0.1	0	16	17	0.1	16	8	16	0.1	14	7	16
0.2	4	16	17	0.2	0	8	16	0.2	14	7	16
0.3	16	16	17	0.3	0	8	16	0.3	0	7	16
0.4	0	16	17	0.4	0	8	16	0.4	0	7	16
0.5	0	16	17	0.5	0	8	16	0.5	0	7	16
Window_size=10, Max_inst_size=25000				Window_size=10, Max_inst_size=25000				Window_size=10, Max_inst_size=25000			
Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI
0.1	0	14	17	0.1	0	8	23	0.1	8	7	17
0.2	3	14	17	0.2	0	8	23	0.2	6	7	17
0.3	4	14	17	0.3	0	8	23	0.3	3	7	17
0.4	16	14	17	0.4	0	8	23	0.4	0	7	17
0.5	15	14	17	0.5	0	8	23	0.5	0	7	17
Window_size=10, Max_inst_size=35000				Window_size=10, Max_inst_size=35000				Window_size=10, Max_inst_size=35000			
Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI
0.1	0	16	17	0.1	0	10	27	0.1	5	7	18
0.2	0	16	17	0.2	0	10	27	0.2	0	7	18
0.3	16	16	17	0.3	0	10	27	0.3	0	7	18
0.4	15	16	17	0.4	0	10	27	0.4	0	7	18
0.5	0	16	17	0.5	0	10	27	0.5	0	7	18
Window_size=10, Max_inst_size=50000				Window_size=10, Max_inst_size=50000				Window_size=10, Max_inst_size=50000			
Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI	Min_sup	Time(ms)	FCI	Semi-FCI
0.1	0	16	17	0.1	0	8	19	0.1	3	7	18
0.2	4	16	17	0.2	0	8	19	0.2	0	7	18
0.3	0	16	17	0.3	0	8	19	0.3	0	7	18
0.4	0	16	17	0.4	0	8	19	0.4	0	7	18
0.5	3	16	17	0.5	0	8	19	0.5	0	7	18



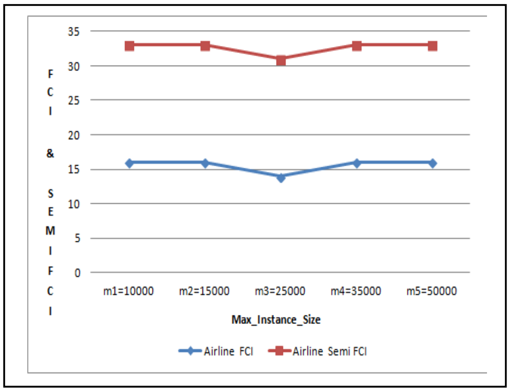
(a)



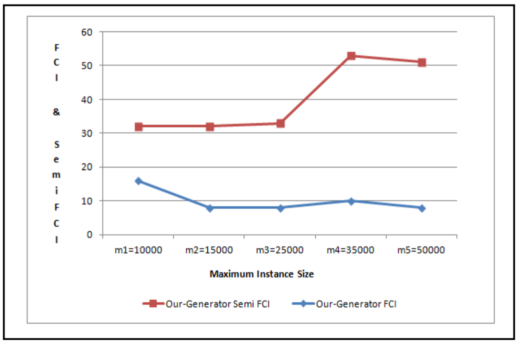
(b)



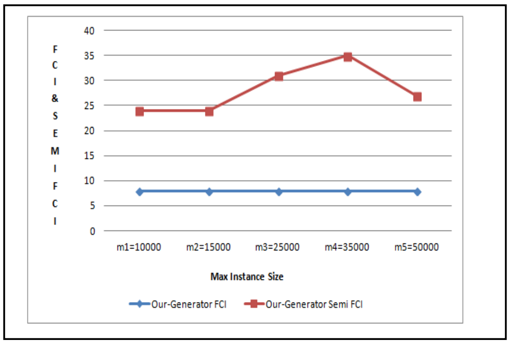
(c)



(d)



(e)



(f)

Fig.4.(a-f.): Graphical representation of performance of INC_MINE algorithm for electricity data set (a & b), airline data set(c &d) and Our_generator (e &f)

6. Conclusion

Some of the important conclusions are as follows: The present work uses massive online analysis framework for the conducting the experimental analysis for determining frequent itemsets In the present investigation three generators (Airline, Electricity, and Our_generator) are used. The algorithm used in determining frequent itemsets is INC_MINE which is an incremental update of FCI over data streams. The parameters appearing in the experimental analysis are Window Size(window_size), Minimum Support (min_sup) and maximum instance Size (max_instance_size). In the case of varying window_size the FCIs are determined for a fixed min_sup. In the case of varying min_sup(0.1,0.2,0.3,0.4 and 0.5) the FCIs and semiFCIs are determined for a variable window size. In the experiments 5 samples if instance size (10000,15000,25000,35000,50000) respectively. The present work establishes that association Rule Mining could be performed even in the case of data stream mining by INC_MINE algorithm using MOA framework which is first of its kind in the literature. One of the important conclusions of the present investigation is that by a suitable choice of the min_sup, max_instance size and the data set generator it is possible to obtain the values of FCI and semiFCI of our interest.

Acknowledgements

One of the authors Mrs. Malini M. Patil acknowledges J.S.S Academy of Technical Education, Bangalore, Karnataka, India. The author also acknowledges Bharathiar University, Coimbatore, Tamilnadu, India for providing the facilities for carrying out the research work. The authors also express their heartfelt gratitude and wish to acknowledge Dr. Albert Bifet for his timely guidelines and valuable suggestions in carrying out the present work.

References

1. Aggarwal, C.C. (Ed.), Data streams: Models and Algorithms. Series: Advances in Database Systems,2007, Vol. 31, XVIII, 354 p, ebook ,Springer, Berlin Heidelberg.
2. Gaber, M. M , Zaslavsky, A. and Shonali Krishnamurthy, Data Streams: Models and Methods, 2007,Vol. 31, pp., 39-59, Book, Springer., Berlin Heidelberg.
3. Ikonomovska, A., Suzana, L., and Gjorgjevik, D. A Survey of Stream Data Mining. in Mile J Stankovski(eds), Proceedings of 8th National Conference with International participation, ETAI, 2007,pp., I6-2.
4. Guha, S. , Koudas, N.K. and Shim, K. Data Streams and Histograms, Proceedings of thirty-third annual ACM Symposium on Theory of Computing. ,2001,pp., 471-475 , ACM Press.
5. Nan Jiang and Le Gruenwald. Research Issues in Data Stream Association rule mining, ACM SIGMOID. Record. 2006,Vol. 35, No 1.
6. Rakesh Agrawal and Ramakrishnan Srikant,Fast Algorithms for Mining Association Rules', 1994, Proceedings of the 20th VLDB Conference Santiago, Chile.
7. Dora Cai, Davis Clutter Greg Pape Jiawei Han Michael Welge and Loretta Auvil 'MAIDS : Mining Alarming incidents from Data streams'. SIGMOID 2004,Paris,France.
8. Philip. S, Yu and Yun Chi. 'Association Rule Mining on Data Streams'. IBM Research Centre.
9. Hua-Fuli, Suh_yin Lee and Man-Kwan shah. Efficient Algorithm for Mining Frequent items over entire history of data streams.
10. Pang-Ning Tan, Michael Steinbach and Vipin Kumar(Eds) Introduction to Data Mining,2007, Pearson Education.
11. Albert Bifet, Geffrey Holmes, R. Kirkby, B Pfahranger, Massive online Analysis Journal of Machine Learning Research ,2010, 11:1601-1604.
12. Albert Bifet, R Gavalda Mining frequent closed trees in evolving data streams. Intelligent Data Analysis, 2011.
13. MOA: Massive Online Analysis . http://moa.cs.waikato.ac.nz/.
14. Cheng. J. Ke. Y, NG. W. A survey on algorithms for mining frequent itemsets over data streams. Knowledge information systems, 2008
15. Ricard Gavalda and Massimo Quadrana. Methods for frequent pattern mining within the MOA system. University Politecnica, Barcelona.
16. Srimani. P.K. and Malini M. Patil, Performance Analysis of Hoeffding trees in MDM using MOA Framework, International Journal of Data Mining, Modeling and Management. Article is in the press.
17. Srimani. P.K. and Malini M. Patil, Massive Data Mining Using Baysian Approach. International Journal of Conceptions on Computing and Information Technolog ,2014, Vol., 2 Issue 4, pp:27-32, ISSN- 2345-9808.
18. Srimani. P.K. and Malini M. Patil, Regression Modelling using IBLSTREAMS, Indian Journal of Science and Technology.2014, Vol 7(6), 864–870.
19. Srimani. P.K. and Malini M. Patil, Performance of Clustering Evaluation Measures in Massive Online Analysis frame work, European Journal of Scientific Research, 2014, Vol 117, Issue 4, No 10, pp:556-567.
20. Srimani P.K. and Malini M. Patil. Simple Perceptron Model (SPM) on evolving streams in MDM. International Journal of Neural Networks(IJNN),2012, Vol. 2, Issue 1, pp.20-24.
21. Srimani P.K. and Malini M. Patil. Massive Data Mining on data streams using Classification algorithms. International Journal of Engineering Science and Technology (IJEST), 2012, Vol. 4 No.06, pp., 2839-2848.
22. Mohammed J Zaki and Ching-Jui Hsiao. CHARM: An Efficient algorithm for closed itemset mining, Rensselaer Polytechnic Institute, Troy NY.