Frequent Itemset Mining using Parallel Processing on Temporal Data

Ms. Jayashri. N. Hajare, Prof. Mr. S. M. Rokade

[student], *Department of Computer Engineering, Pravara Rural Engineering Collage Loni, Maharastra, India*
[Prof], *Department of Computer Engineering, Pravara Rural Engineering Collage Loni, Maharastra, India*

ABSTRACT

Frequent pattern extraction is important technique in data mining domain. Lot of work has been done on pattern mining in literature. The work focuses on pattern extraction on static dataset. The whole dataset is used and frequent itemset are extracted. Temporal data is the data containing timestamp information. Timestamp based pattern extraction affects the results. The proposed work focuses on frequent itemset extraction on temporal data. The whole data is partitioned in time cubes based on the timestamp information. For itemset extraction apriori algorithm is applied on each time cube. A density threshold is used to overcome the overestimation problem of time period. The technique enlists the patterns those are occurred in particular time interval. The work improves the system efficiency with parallel thread working. Comparative results are taken on the basis on time and memory.

Keyword: - *Data mining, Pattern extraction, frequent itemset mining, temporal data, multithreaded programming*

1. INTRODUCTION

Frequent pattern extraction is important technique in data mining domain. Transactions history analysis in shopping mall, banks, stock exchange, etc. is important application in data mining. Frequently occurred patterns in transactions of shopping mall represents frequently sold items. Analysis of such frequent item helps in making decisions such as product promotion, stock maintenance, stock placement, etc. Frequent patterns occurred in bank transactions or in stock exchange helps to predict future patterns. Frequent pattern analysis is also important in intrusion detection, web browsing history, bioinformatics, etc. To define frequent pattern, minimum occurrence count i.e. support value should be mentioned by the user. Itemset occurred more than minimum occurrence count is called as frequent itemset. Itemset is nothing but a co-occurrence of patterns.

All transactions are related to time. Dataset containing time information is called as temporal dataset. Along with occurrence count of transaction the occurrence time i.e. temporal information is again an important aspect. For e.g. transactions containing bread and butter are occurred frequently in the morning time than the whole day transactions. Such interesting patterns are time dependant. Temporal analysis of patterns extracts more useful information. This information includes frequent itemset purchase time span, lifespan of itemset, peak period of purchase, etc.

For frequent pattern extraction, algorithms such as apriori, FP-Growth, eclact, etc. are used. Agrawal et al. [10]. Initially proposes the co-occurrence pattern extraction technique. Apriori technique works in 2 phases. In first phase it identifies the frequent itemset candidate using download closure property and in second phase it filters the candidate patterns based on support value and generates the frequent pattern set. `FP Growth is tree based algorithm. It constructs conditional frequent item tree and uses depth first search for tree traversal and itemset extraction. Eclact algorithm also uses tree structure and depth search traversal technique for itemset extraction. These algorithms do not consider temporal aspect and works on complete dataset.

Sequential and time interval association rules, calendar rules are temporal frequent pattern analysis technique proposed in literature. In the following section related work to pattern discovery is discussed in detail with its techniques and limitations.

2. LITERATURE SURVEY

An association rules define the frequent patterns. Association rule discovery technique was first proposed by Agrawal et al.[10]. Association rules were extracted from 2 view data. Initially frequent items were extracted from each view and then dependency of co-occurrence of frequent pattern in each view defines the association rules. The experiment was carried out over purchase history and association among products was identified.
Patterns those are occurred after every specific time interval is called as cyclic patterns. The patterns should occur without any exception. Ozden et al.[9] proposes a cyclic patterns extraction technique. This is a temporal technique. It identifies daily, monthly, quarterly and yearly patterns. The analysis of such cyclic pattern helps to analyze market trends and also helps in sale forecasting. For cyclic pattern extraction sequential algorithm and interleaved algorithm are used.

The extended version of cyclic pattern extraction is proposed by Ramaswamy[7]. This technique finds the user defined patterns in association rules. Calendar algebra technique is used to detect patterns. But for execution calendar expression rules must be user defined.

Ale and Rossi[6] proposes a system that extracts frequent itemset over specific time period. The time period is shorter than the whole transaction history time span. Life time of each item is used to define time interval. To extract temporal pattern a new algorithm is proposed based on apriori algorithm. The concept of temporal support is introduced first time in the algorithm.

In the market, purchase strategy varies very frequently. The product sold in one time span may not be sold with that frequency in next cycle. Hence cyclic patterns discovery over transaction dataset may not give the appropriate patterns. To overcome this issue, Han et al[8] proposes a periodic pattern discovery technique. The technique segments the time span in specific time period. The time period are fixed in length. The frequent patterns extracted from such periodic dataset gives good results for some period but not for all.

In real life scenario the exhibition period varies with respect to item. By considering the exhibition period of item, Progressive-partition-miner- PPM[5] technique is proposed by C. H Lee and M. S. Chen. The technique initially partitions the database in item exhibition period. Then it finds occurrence count of each candidate progressively using intrinsic partitioning characteristics.

Matthews *et al.* [2] proposes a new generic algorithm that traverses whole dataset and find temporal frequent patterns. It uses evolutionary algorithms in searching for finding association rules and defining optimization parameters. The technique simultaneously searches the rule space and temporal Space. Using this technique, temporal frequent patterns are extracted without prior partitioning the dataset.

Xiao *et al.* [3] proposes a technique to find association rule with time windows. The main aim of this system is to dynamically find association rules and its occurrence time windows. The time window is arbitrary in length and varies with respect to each association rule.

Seasonal product purchase behavior analysis technique is proposed by Saleh and Masseglia[4].This technique partition the database and analyze each part separately. The technique works on assumption that some products are sold in specific time period than a whole year like items purchased in winter are different than summer. Based on item frequency in transaction dataset, dataset is get partitioned for further analysis. Itemset support value increases with partitioned dataset as compared to the whole dataset and more frequent itemset are extracted for specific time window.

M. Ghorbani and M. Abessi[1] proposes a temporal frequent itemset extraction based on time intervals. A basic time cubes are defined by taking 3-D parameters such as day-month-year or, hour-day-month. The technique assumes that patterns are found in specific time interval or else it may exist over complete dataset. For temporal pattern extraction apriori algorithm is modified in which time cube based large one itemsets are extracted. The technique is time consuming because each time cube is processed separately.

3. ANALYSIS AND PROBLEM FORMULATION:

There is need to develop frequent itemset extraction from a given dataset with temporal information. The item selling frequency varies with respect to time. The efficient system should be developed that finds frequent itemset over specific time period.

Problem Statement: To design, develop and test efficient frequent itemset extraction technique over temporal dataset.

4. SYSTEM OVERVIEW:

Following fig.1 shows the architecture of the system. The system has 4 inputs: dataset, BTC parameters, minimum density value and minimum support value. Based on the given input system find frequent itemset on periodic basis from whole dataset. The detailed working of the system is described in the following section.

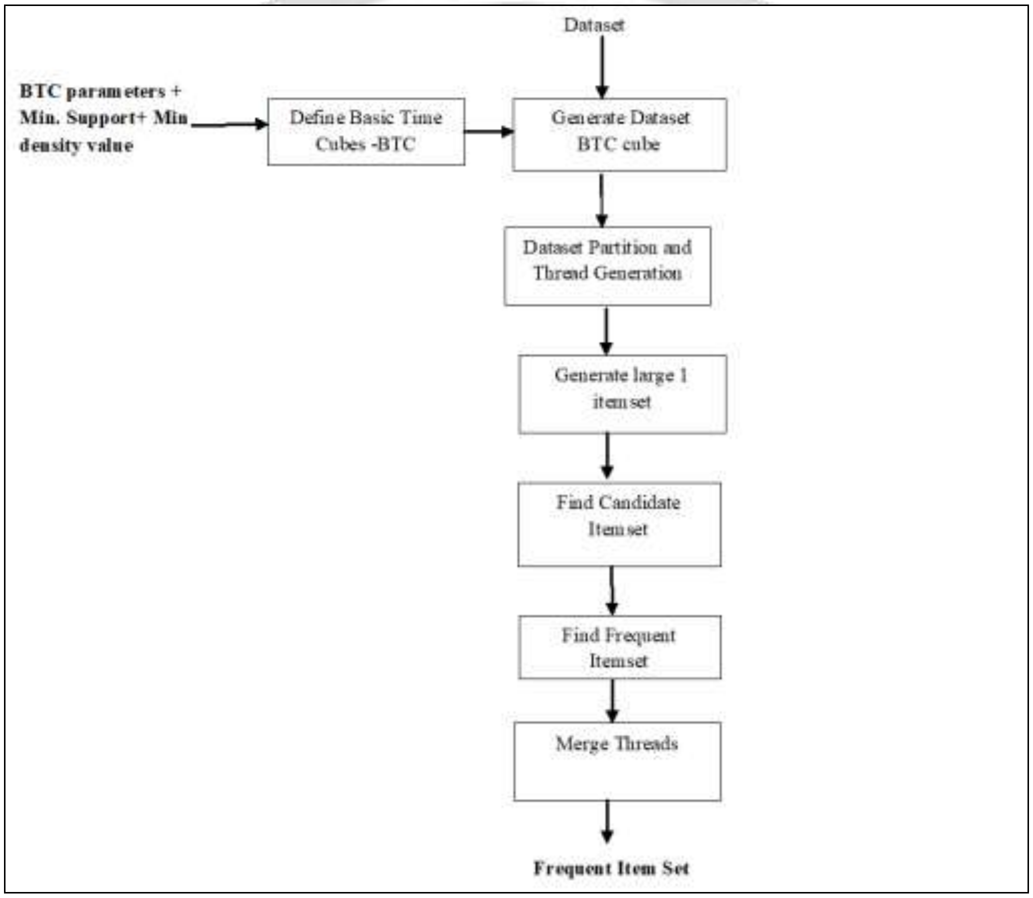
Dataset
BTC parameters + Min. Support+ Min density value
Define Basic Time Cubes -BTC
Generate Dataset BTC cube
Dataset Partition and Thread Generation
Generate large 1 item set
Find Candidate Item set
Find Frequent Itemset
Merge Threads
Frequent Item Set

Fig -1: System Architecture

4.1 System Working

System extracts the frequent itemset from a given dataset with specific time interval .The dataset contains number of transactions with timestamp information. Transaction contains unique ids of items purchased in that transaction.
The dataset is partitioned in number of data cubes. The cubes are called as basic time cubes- BTC . These time cubes are generated based on the timestamp information. The partitioning strategy is initially defined by the user. For cube generation 3 dimensions are selected by user like: day-month-year, hour-month-year, hour-day, month, etc.
The frequent items are extracted from each time cube. A support value of each itemset I is calculated as:

Support(I) = $\frac{\text{N(I)}-\text{cube}}{\text{N}-\text{cube}}$

Where N(I)-cube is the occurrence count of itemset in defined time cube and N-cube is the total number of transactions in that time cube. A minimum support threshold value is required for filtering frequent itemset. The itemset support value is compared with minimum support value.
The data is not equally partitioned in each time cube. Some time cubes may contain dense transaction entries where as some data cubes may contain very few transaction records. Frequent itemset extracted from data cubes containing very few records, generates the overestimated false result. To overcome this issue data cube density concept is used. Frequent itemset are extracted from only those data cubes having cube density greater than the threshold value. If data cube is having lower density then the data cube records are merged with next data cube. The cube density is calculated as:
Density = α *A
Where α is user defined constant value.
A is average number of records per cube it is calculated as:
A = $\frac{N}{\text{N−BTC}}$,
N is total number of records in a dataset
N-BTC is total number of time cubes.

Apriori algorithm is used to find frequent itemset over each time cube. Initially large 1 itemset is generated and then based on support value frequent items are filtered. Processing each time cube is time consuming task. To improve system efficiency parallel processing is used. A multithreaded application simultaneously process time cubes and finally merged the results.

5. ALGORITHMS

Algorithm 1: BTC Specific mining frequent itemsets
Input: Database -D,
Basic Time Cube-BTC
Minimum support-min_sup,
Density Threshold-den

Output: BTC specific Set of frequent itemsets F-set
Processing:

```
1.  L: find large 1 itemset with TC time cube
2.  Generate 2 parallel thread
3.  Partition large 1 data L in 2 parts
4.  Assign partition to each thread
5.  C: find candidates in each thread
6.  Sup: Calculate support value for each C
7.  If sup>=min_sup and density of (TC)> den
8.  Add itemset in F-set
9.  Merge thread
10. Return F-set
```

Algorithm 2: mining Large1 itemsets
Input: D: Dataset,
Basic Time Cube-BTC
Minimum support-min_sup,
Density Threshold-den
Output: L: Large 1 itemset L
Processing:

```
1.  For all items I in Time cube
    Sup(I):Calculate support value
2.  For all items in BTC
3.  If(support (I) >=sup and D-BTC-Density>=den then
4.  Update Time cube TC  as TC U BTC
5.  Add item I in large 1 itemset L
```

6. Return L with TC

```
Algorithm 3: candidate generation
Input: L: Large 1 item set L
Output: C-set : Candidate items set
Processing:
For all items in L
    1. C(Li,Lj): Generate item pair Li and Lj using joint operation
    2. If C(Li,Lj) length is K then
    3. add C(li,Lj) in C-set
    4. return  C-set
```

6. MATHEMTICAL MODEL

The system can be mathematically represented using set theory. The system set S can be defined as:
S= {I, O, P} where
I = {D, BTC, T, α }, Set of Inputs
D = transaction Dataset
BTC = Basic time cube parameters
T = Minimum Support Threshold
α = min density value

O = {FIM_P, FIM }, Set of Outputs
FIM_P = Periodic frequent itemset
FIM = Final Frequent itemset

F= {F1, F2, F3, F4, F5, F6, F7, F8, F9 }, Set of Functions
F1 = Upload dataset
F2 = generate BTC dataset cubes
F3 = Dataset Partition
F4 = Calculate density
F5 = Generate large1 itemset
F6 = Find Candidate itemset
F7 = Calculate support
F8 = Find periodic Frequent itemset
F9 = Merge Itemset
F11 = Generate Frequent itemset

7. IMPLEMETATION

To implement the system java platform is used. For parallel processing java multi threading is used. The system was implemented and tested on core i3 system with 4 gb ram.

7.1 Datasets:

Following datasets are downloaded from fimi[11] website.

1. Pumb_star
2. Mashroom
3. T10I4D100K

The dataset contains number of transactions. Each transaction is represented by a row containing unique item ids. The dataset do not contain transaction timestamp information.

A custom dataset is generated with timestamp information. To add timestamp information user selects the interval i.e. start date and end date. Based on the user input system randomly generates the dates and assigned to the transaction.

Based on the timestamp information time cubes are generated. User selects the BTC partition type i.e. day-month-year or hour-day-month,etc and defines the number of slots need to be generated. Based on the given input system defines the time cubes and partition the data in multiple files.

7.2 Performance Metric:

The complete system can be evaluated based on following parameters.

1. Variation in minimum support value: The count of temporal frequent itemsets extracted from dataset by changing the minimum support value.
2. Variation in BTC time Cubes: The count of temporal frequent itemsets extracted from various basic time cubes.
3. Time: The time required for execution over for complete system. The time will be compared between single thread and multi thread execution.
4. Memory: The memory requirement will be compared for executing single and multi threaded application.

8. Results

Partial System is implemented. Dataset preprocessing is performed and time information is added to the database. Based on the time information time cubes are also generated.
For time cube generation from raw dataset user selects start and end date, number of slots and BTC type. Following are the details for Cube generation process:
Start Date: 11-03-2018 03:00:10
End Date: 25-06-2018 02:00:00
BTC Type: hour, date, month
Number Of Slots: 3

With slot count 3, hour, date and month data is portioned in 3 slots and finally 27 cubes are generated between the given dates.
Hour slot: 0.0-7.667, 7.667-15.334, 15.334-23.0
Date Slot: 1.0-11.0, 11.0-21.0, 21.0-31.0
Month Slot: 3.0-4.0, 4.0-5.0, 5.0-6.0

Following table shows the transaction partition in respective time cube:

Table 1: Time Cube generation

Slot Information Hour-date- month	Date-Time	Transaction
0-0-1	06-3-2018 01:23:24	1 3 5 7 9 11 13 15 17
1-1-2	11-5-2018 08:42:36	1 3 5 7 9 12 13 15 17
2-2-0	30-3-2018 16:13:45	1 3 5 7 9 12 13 16 17
2-2-1	24-4-2018 20:45:14	1 3 5 7 8 12 13 16 21
2-2-2	23-6-2018 18:43:18	2 3 5 8 9 12 15 16 17

Frequent itemset are extracted from whole dataset based on time cubes. Following table 2 represents average number of items extracted per time cube, total itemset extracted from whole dataset. Along with the itemset count the time and memory required for Frequent itemset extraction from complete dataset based on BTC time cube information is also presented. The table includes the comparative study between existing and proposed system. The time and memory analysis is calculated for various datasets with various minimum support values.

Table 2: Time and memory analysis For Various Datasets

Dataset	Min Support	FIM Execution Time in seconds	Parallel FIM Execution Time in seconds	FIM Memory Required in MB	Parallel FIM Memory required in MB	Avg No of FIM per cube	Total FIM
Pumsb star	0.4	3076.72	3044.891	6.82	6.95	33506	9548
	0.6	18.142	2.438	7.8	8.27	172	122
	0.8	3	1.382	8.35	8.41	0.09	0
Mashroom	0.4	16.05	11.849	7.27	7.56	3358	675
	0.6	2.656	0.969	7.35	7.77	299	63
	0.8	1.211	0.375	7.54	8.25	43	15
T40I10D100K	0.04	12.062	1.75	7.25	7.48	26.42	16
	0.06	6.858	0.891	7.67	8.63	4.57	3
	0.08	3.205	0.734	7.88	8.8	0.33	0

Following chart 1 and 2 represent graphical analysis of system. The analysis includes time and memory analysis of system execution over various datasets. Time and memory are not depends on the dataset size. It depends on the number of itemset found based on a given threshold. It also shows the comparison among single threaded and parallel threaded system processing in terms of time for different da
tasets. Parallel processing requires less time as compared to the single thread application.

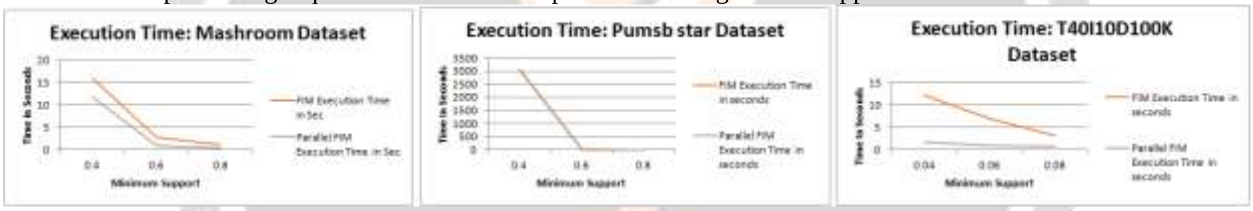

Chart -1 Time Analysis

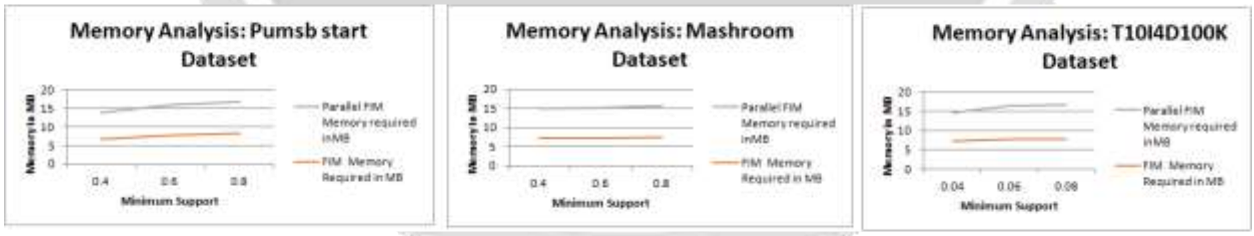

Chart -2 Memory Analysis

chart 3 represents the total number of itemset extracted with respect to various threshold value defined. As minimum support value increases, the count of average number of frequent itemset extracted per cube and total itemset found over complete dataset decreases.

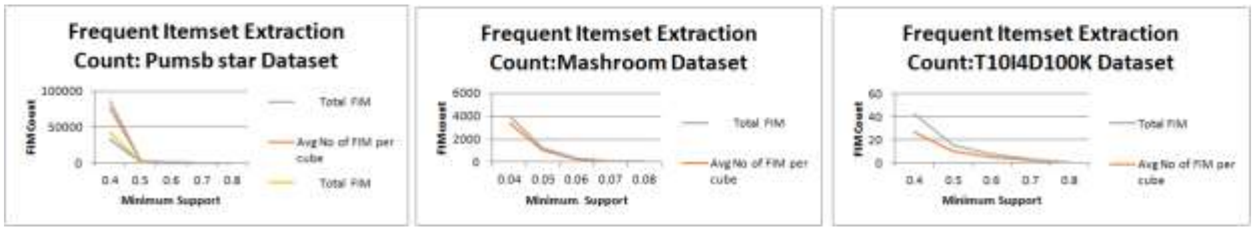

Chart -3 Itemset Count Analysis

9. CONCLUSIONS

System finds frequent patterns over temporal dataset. The Basic time Cubes divides the temporal dataset in number of cubes. To find valid frequent items from time cube data cube density concept is proposed. Due to cube density overestimation of time period is avoided. Multithreaded application improves the system efficiency by parallel processing of time cubes data. The time required for processing dataset decreases with parallel thread execution.

In Future system can be implanted over continuous data stream where dynamic time cubes will be generated for incoming streaming data.

10. REFERENCES

[1] Mazaher Ghorbani and Masoud Abessi, "A New Methodology for Mining Frequent Itemsets on Temporal Data", in IEEE Transactions on Engineering Management, Vol. 64, Issue. 4, pp. 566 - 573, Nov 2017
[2] S. G. Matthews, M. A. Gongora, and A. A. Hopgood, "Evolving temporal association rules with genetic algorithms," in Research and Development in Intelligent Systems XXVII. New York, NY, USA: Springer, 2011, pp. 107–120.
[3] Y. Xiao, R. Zhang, and I. Kaku, "A new framework of mining association rules with time-windows on real-time transaction database," Int. J. Innov. Comput., Inf. Control, vol. 7, no. 6, pp. 3239–3253, 2011.
[4] B. Saleh and F. Masseglia, "Discovering frequent behaviors: Time is an essential element of the context," Knowl. Inf. Syst., vol. 28, no. 2, pp. 311–331, 2011.
[5] C.-H. Lee, M.-S. Chen, and C.-R. Lin, "Progressive partition miner: An efficient algorithm for mining general temporal association rules," IEEE Trans. Knowl. Data Eng., vol. 15, no. 4, pp. 1004–1017, Jul./Aug. 2003.
[6] J. M. Ale and G. H. Rossi, "An approach to discovering temporal association rules," in Proc. ACM Symp. Appl. Comput.-vol. 1, 2000, pp. 294–300.
[7] S. Ramaswamy, S. Mahajan, and A. Silberschatz, "On the discovery of interesting patterns in association rules," in Proc. 24th Int. Conf. Very Large Data Bases, 1998, pp. 368-379.
[8] J. Han, W. Gong, and Y. Yin, "Mining segment-wise periodic patterns in time-related databases," in Proc. Int. Conf. Knowl. Discovery Data Mining, 1998, pp. 214-218.
[9] B.O zden, S. Ramaswamy, andA. Silberschatz, "Cyclic association rules," in Proc. IEEE 14th Int. Conf. Data Eng., 1998, pp. 412-421.
[10] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," ACM SIGMOD Rec., vol. 22, no. 2, pp. 207-216, 1993.
[11] Dataset: http://fimi.ua.ac.be/data/