IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Temporal Co-occurrence pattern Extraction on transactional Dataset

Pratiksha P. Ghule

Student
Department Of Computer Engineering,
Matoshri College Of Engineering & Research Centre, Eklhare, Nashik, India

Prof. Dr. Swati Bhavsar

Associate Professor
Department Of Computer Engineering,
Matoshri College Of Engineering & Research Centre, Eklhare, Nashik, India

Abstract : Frequent itemset mining is important task in data mining domain. This is applicable in variety of applications such as market-basket analysis, browsing history analysis, transaction record analysis, etc. Lot of work has been done in the domain of co-occurrence pattern extraction and association rule mining. The existing system works on static dataset as an input. The proposed system focuses on temporal analysis of data and extracts co-occurrence patterns from dataset based on timestamp information. Each record in a dataset has it is own time information. Based on the time information data is sliced in 3 dimensional cube. The apriori algorithm is extended to work with time cube information data. Using time interval cubes, the co-occurrence of pattern is analyzed periodically and with certain time interval. For processing multiple time cube simultaneously, a multithreaded environment is proposed to improve system efficiency. To solve the overestimation problem, a density threshold value is checked for each time cube. The performance of system is tested on various dataset and its execution time and memory is compared with the existing approach.

IndexTerms **- co-occurrence patterns, time cube, Temporal analysis, apriori, multithreaded application, frequent patterns**

I. INTRODUCTION

In data mining co-occurrence itemset extraction is important branch. For co-occurrence pattern extraction generally transaction dataset is used. Transaction records contain supermarket data selling information. Each record represents the order placed by a customer. Analysis of such order records, frequently sold items can be extracted. The frequently occurred patterns are called as co-occurrence patterns. For example: bread and butter are occurred in multiple transactions at a time hence it{bread, butter} is called as itemset. If this itemset occurs frequently then this is called as co-occurrence pattern. In co-occurrence pattern extraction the order of purchase is not important. The co-occurrence pattern extraction helps to make decision in marketing strategies such as : product placement, product promotion, offer announcement, etc.

For finding co-occurrence patterns, minimum support value should be defined first. Based on the itemset occurrence, the support value of each itemset is calculated. The support value is calculated based on the number of transaction in which itemset belongs. If the itemset follows the minimum support constraint then the item is called as frequent itemset.

The co-occurrence pattern extraction technique can also be applied in variety of domains such as: bank transaction history analysis, intrusion detection, browsing link history, medical records, bioinformatics etc.

Generally, the dataset is stored with timestamp information. Such dataset with time information is called as temporal dataset. The transaction records vary periodically hence pattern extraction should be done by considering not only the occurrence count but also the temporal information.

System is proposed by considering some issues of previous approaches. The record with time stamp entries is called temporal data. Sequential association rules, time interval association rules, calendar specific interval rules are various frequent item set mining technique based on temporal information

This system is designed to mine frequent item set from temporal data. Also as contribution module is added that reduces the time of this frequent itemset mining from static and temporal dataset. Following are the phases of the system execution.

1. Phase of set up

In this phase required transnational dataset is downloaded. If it is not temporal one , it is converted in temporal dataset by adding date time for each transactions and make if temporal one for execution as per requirement. Also by considering slots (day , hour , month)

different data cubes are generated.\\

2. Phase of execution

In this phase data cubes are analyzed first. Any cube with low density is merged with nearest data cube. After that system expects minimum support value from end user and according to that frequent itemset mining is done using Apriori algorithm. As part of contribution, multi-threaded parallel processing is applied. This reduces time for mining frequent itemset from temporal dataset.

Following section includes the related work done in the domain of pattern extraction. In section III problem formulation is stated. In section IV, a proposed system details are discussed followed by its implementation details and results in section V. Section VI concludes the paper.

II. RELATED WORK

Daichi Amagata, et. al. proposes a technique for Mining Top-k Co-Occurrence Patterns. The patterns are extracted by mining multiple data streams. The system works on streaming data. More than one data stream is handled at a time and frequent itemsets are extracted. A CP-graph is generated from transaction records. For each sliding window the CP graph is updated. The patterns that occur in more than one stream and support value is greater than minimum support value then such patterns are called as co-occurrence patterns. The Minimum threshold is calculated automatically based on top K occurrence count of itemset.

Xiao, et.,al[3] proposes a technique for mining association rules based on temporal information. The paper works on discovery of co-occurrence patterns and association rules from dataset. The system uses temporal dataset. Based on the temporal information, the maximum time frequent window for itemset is identified. This is the window in which the itemset support value crosses the minimum support threshold constraint. The system uses A variable neighborhood search (VNS) algorithm.By mathematical modeling the time window is optimized.

L. T. Nguyen,et.,al [4] proposes a technique for mining association rules with generalization. For mining association rules, the item are arranges in hierarchical manner. The technique focuses on reducing the redundancy in association rule.

For mining class association rule L. T. Nguyen,,et. al. proposes a CAR-Miner algorithm[5]. This method is useful than heuristic and greedy methods. This methods work on removing noise and hence it improves the accuracy of the system. For mining class association rules, system uses tree structure. Each node in a tree has some information for fast computation of support value for next candidate pattern. Node pruning technique is used to find candidate patterns.

D. Nguyen, et.al.[6] proposes a class association rule mining technique named as CCAR. This is an extended version of Car algorithm. This technique is able to find CAR rules based on user defined minimum support threshold value and minimum confidence thresholds from a dataset. To improve efficiency of system it partition the algorithm in two phases. In preprocessing phase candidate rules are generated and candidate list is filtered in the post-processing step.

Saleh and Masseglia [7] proposes a system to find frequent itemset on temporal data. It find the subset of dataset in which the frequent itemset occurs. This technique is useful for seasonal product purchase analysis. The system dynamically finds the period of items in a dataset. The itemset support value should be very less as compared to the other traditional algorithm.

Progressive partitioning[8] technique proposes a solution on 2 problems: 1: Lack of exhibition period of each item 2 : lack of equitable support for each item. The system initially portioned the dataset in equal parts. And then progressively join the candidate items with the itemset size 2.

Matthews et al. [9] proposes a generic algorithm to find association rules from temporal dataset. This technique does not require any prior knowledge for portioning the dataset. Algorithm searches each rule in rule space as well as in temporal space. The dynamic partition of datasets is generated as per each rule.

Mazaher Ghorbani et. al[1] proposes a technique for frequent itemset extraction from temporal dataset. The data is divided in number of time cube blocks. Each block is analyzed separately and frequent itemsets are extracted. The frequent itemsets are extracted based on minimum support value and cube density. The apriori algorithm is used to find frequent itemsets. This is a time consuming process because each time cube block need to be processed separately.

III. ANALYSIS AND PROBLEM FORMULATION

Co-occurrence pattern extraction plays important role in data mining. Lot of work has been done on static dataset. The transaction records vary periodically. The co-occurrence pattern should be extracted based on the temporal information. There is need to develop a system that fiend co-occurrence patterns on temporal dataset efficiently.

IV. PROPOSED METHODOLOGY

A. Architecture

Following figure shows the architecture of the system. Transaction dataset, BTC information and threshold values are input to the system. System finds co-occurrence patterns per time cube and also the final co-occurrence set. The apriori algorithm based mining frequent itemsets with time cubes is applied on the dataset. The system works with multithreaded execution. The multithreading execution increases system efficiency.

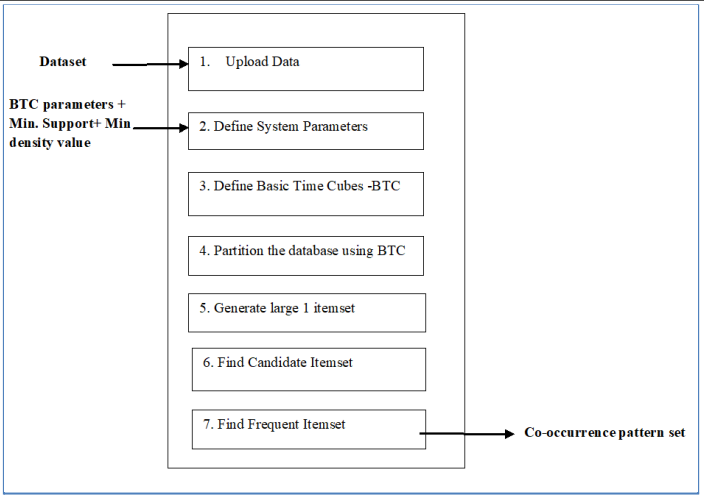


Figure 1 : System Architecture

B. System Working:

The transaction dataset with time information is partitioned in number of time cubes. User defines the time cube division parameters such as: (Hour, Month, Year), (Hour, Day, Month), (Day, Month, Year),etc. User also provides the number of partition count for each parameter. For example if user suggest to partition each parameter in 3 sections then total 27 cubes will be generated. A unique id is assign to each time cube. The input dataset is partitioned in time cube and time cube unique id is assigned to the transaction in which it belongs.

The whole dataset is partitioned in number of time cubes. Some data cube may contain large number of items whereas some data cubes contain very few items. The frequently occurred pattern extraction from data cube containing vary few items may not generate an appropriate result. A cube density constraint is introduced to measure and limit the minimum occurrence of items in a data cube. The density of time cube is calculated as:

Density = α *A (1)

Where, A is the average number of records per cube it is calculated as:

$$A = \frac{N}{N-BTC} \quad (2)$$

Where N = total number of transaction in a dataset,

N-BTC = Number of transaction in BTC

A is user defined threshold value between 0 and 1.

If the time cube density is less than A then the time cube data is merged with next time cube and revised time cubes are generated.

The large one itemset is extracted from each time cube. The large-one itemset represents the set of co-occurrence patterns. The support value for each co-occurrence pattern X is extracted using following formula:

$$\text{Support}(X) = \frac{N(X)-cube}{N-cube} \quad (3)$$

Where N(x)-cube : total number of itemsets containing itemset X in time cube TC

N-cube: Total number of transaction in time cube TC

User defines the minimum support value. If the support value of itemset X is greater than the minimum support value then the itemset is called as frequently occurred itemset. Such itemset is added in co-occurrence pattern list.

For co-occurrence pattern extraction following 2 conditions need to be satisfied:

1. The itemset should belong from a time cube whose density is greater than A
2. The itemset should have support value greater than minimum support value

The mining frequent itemset algorithm process each time cube block To improve system efficiency the parallel processing is introduced. In parallel processing more than one block is executed simultaneously. The collective result is displayed to the user.

C. Algorithms:

Algorithm 1: mining Large1 itemsets

Input: BTC: Basic time cube from Dataset,

Min-Sup: Minimum support threshold,

Den : Min density,

Output: L: Large 1 itemset

Processing:

1. For all items i in time cube D-BTC

Calculate support of i

2. For all BTC

3. If(support (X-BTC) >=min-sup and D-BTC-Density>=den then
4. Update Time cube TC with BTC
5. Add item i in large 1 itemset L
6. Else merge time cube with next time cube
7. Return L with TC

Algorithm 2: candidate generation
Input: L: Large 1 item set
Output: C-set: Candidate items set
Processing:

1. Li,Lj: Generate items pairs in L
2. For all pairs of L
3. Apply join operation over Li and Lj
4. If candidate length is K then
5. update candidate in C-set
6. return C-set

Algorithm 3: mining frequent itemsets with time cubes
Input: D: Dataset,
Min-sup: Minimum support threshold,
Den : Min density,
TI: Basic Time Cube parameter
Output: BTC-F-set: Periodic co-occurrence patterns
F-set: Final co-occurrence patterns
Processing:

1. Generate BTC form Time cube information
2. L,TC: Find large 1 itemset using mining Large1 itemsets algorithm
3. L1, L2 : partition large one itemset for parallel processing
4. Generate 2 thread for processing L1, L2
5. C-set: Generate candidate item form L1 and L2 using candidate generation algorithm
6. Sup: Find min support for each each candidate
7. For each time Cube c
8. If sub >= minsup & BTCden[c] > = den then
9. Add itemset in BTC-F-Set with TC information
10. F-set: Frequent items present in each time cube c
11. Return BTC-F-set, F-set

D. Mathematical Modeling:

The system S can be defined as:
S= {I, O, F} where
I = {I1, I2,I3,I4}, Set of Inputs
I1 = Dataset
I2 = Time cube parameters
I3 = Minimum density Value
I4 = Minimum Support Threshold

O = {O1,O2, O3, O4} , Set of Outputs
O1= Periodic co-occurrence patterns
O2 = Final co-occurrence patterns
O3 = Time for processing

O4 = Memory for processing

F= {F1, F2, F3, F4, F5, F6, F7 ,F8, F9, F10, F11, F12, F13 } Set of Functions

F1 = Upload dataset

F2 = Generate time cubes

F3 = partition data in time cubes

F4 = Calculate cube density

F5 = Apply multi-treading

F6 = find large1 itemset

F7 = find candidate set

F8 = Calculate support value

F9 = Find co- occurrence patterns in item cube

F10 = Find co-occurrence patterns in dataset

F11 = View co-occurrence patterns

F12 = Find processing time

F13 = Find memory usage

V. RESULT AND ANALYSIS

The system is implemented in java using jdk1.8. The windows operating system is used for development. The system has 4gb ram and i3 processor.

A. Dataset:

The system is tested on various transaction datasets. The datasets are downloaded from fimi website [10]. The dataset contains only transaction information. The data is in comma separated (csv) format. Each row represents the transactional record. Each row contains number of items defined by its unique item-ids. The temporal information is created synthetically. For temporal information creation user selects the start and end date. Between start and end date random dates are created and assigned to each transaction.

B. Performance Measures:

The system performance is measured in terms of :

1. Time: The time required for processing with single thread and multithread execution is compared.
2. Memory: the memory required for processing single thread and multithread execution is compared.
3. Itemset Extracted: The number of co-occurrence patterns extracted from multiple time cube is extracted.

C. Results:

From the date and time information, the time cubes are generated and whole dataset is divided in number of time cubes. Consider following example: In this example start date and end date is input to the system. To divide the data in time cube, time cube format and number of slots are also input to the system.

Start Date: 11-03-2020 03:00:10
End Date: 25-06-2020 02:00:00
BTC Type: hour, date, month
Number Of Slots: 3
Based on the above information system generates hour, month and day slot as follows:
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. The transaction with timestamp information is present in a dataset. Based on the timestamp information of transaction the slot is identified. The slot numbers are varying from 0-0-0 to 1-1-1. Total 27 possible combinations are generated.

TABLE 1 : TIME CUBE GENERATION

Slot Info. Hour-date -month	Date-Time	Transaction
0-0-1	06-3-2020 01:23:24	1 3 5 7 9 11 13 15 17

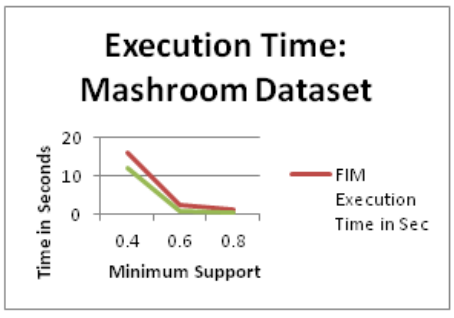
1-1-2	11-5-2020 08:42:36	1 3 5 7 9 12 13 15 17
2-2-0	30-3-2020 16:13:45	1 3 5 7 9 12 13 16 17
2-2-1	24-4-2020 20:45:14	1 3 5 7 8 12 13 16 21
2-2-2	23-6-2020 18:43:18	2 3 5 8 9 12 15 16 17

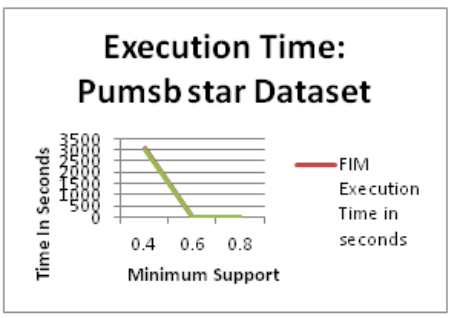
Time and memory analysis:
The following table contains the time and memory analysis for 3 different datasets. The results are collected for different minimum support value. The processing time decreases as we increase the support value. The time required for proposed system is less as compared to the existing system. The proposed system executes the task using multithreaded processing. This reduces the time of execution. For parallel execution, memory required for processing increases.

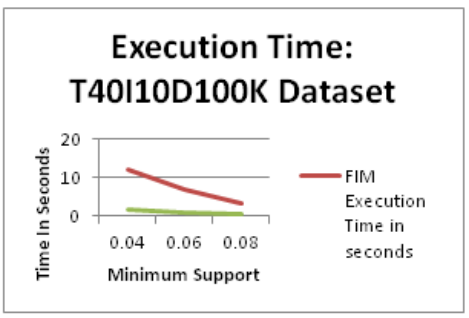
Table 2: Time and memory analysis For Various Datasets

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	3276.72	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	0.0	0
Mashroom	0.4	16.05	11.849	7.27	7.96	3858	675
	0.6	2.656	0.969	7.35	7.75	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

Time Analysis: The following graph shows time analysis for three different datasets. The Results are taken for 4 different minimum support values. The time required for proposed system execution is less as compared to the existing system for all threshold values. As we increase the minimum support value the time required for processing get decreases.

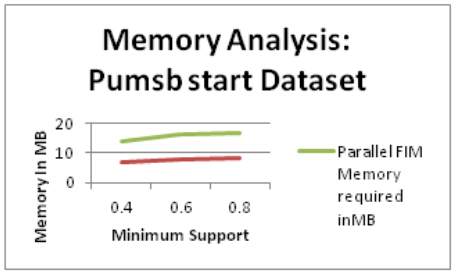


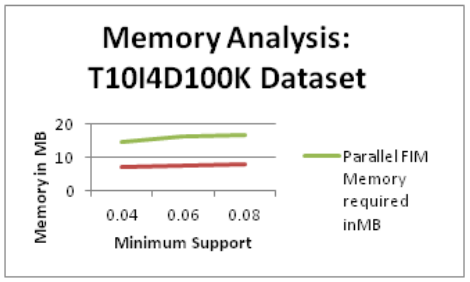


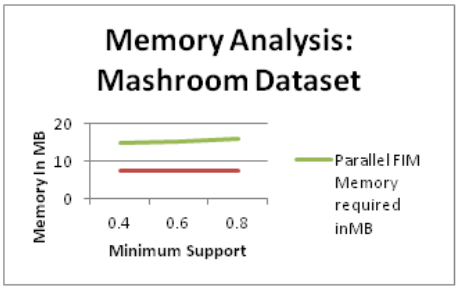


Memory Analysis:
The following graph shows memory analysis for three different datasets. The Results are taken for 4 different minimum support values. The

memory required for proposed system is higher than the existing system for all threshold values.







VI. CONCLUSIONS

The proposed system works on finding co-occurrence patterns on based on its time information. The dataset is divided in number of time cubes based on the time information. Apriori algorithm is used to find co-occurrence patterns in each cube and from the overall dataset. Density of each cube is checked to avoid over estimation problem. For efficiency improvement, parallel processing is introduced. In parallel processing multiple cubes are processed simultaneously. During system execution, following points are noticed:.

- Time and memory are not depends on the dataset size. It depends on the number of itemset found based on a given minimum support value.
- Parallel processing requires less time as compared to the single thread application for all minimum support values.

In Future the system can be implemented on continuous streaming data.

VII. 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] Daichi Amagata, Takahiro Hara,"Mining Top-k Co-Occurrence Patterns across Multiple Streams", in IEEE Transactions on Knowledge and Data Engineering, Vol. 29, Issue 10, pp. 2249 - 2262, Oct 2017

[3] F. Benites and E. Sapozhnikova, "Hierarchical interestingness measures for association rules with generalization on both antecedent and consequent sides," Pattern Recognit. Lett., vol. 65, pp. 197–203, 2015.

[4] F. Benites and E. Sapozhnikova, "Hierarchical interestingness measures for association rules with generalization on both antecedent and consequent sides," Pattern Recognit. Lett., vol. 65, pp. 197–203, 2015.

[5] D. Nguyen, B. Vo, and B. Le, "CCAR: An efficient method for mining class association rules with itemset constraints," Eng. Appl. Artif. Intell., vol. 37, pp. 115–124, 2015.

[6] L. T. Nguyen, B.Vo, T.-P. Hong, and H. C. Thanh, "Car-miner: An efficient algorithm for mining class-association rules," Expert Syst. Appl., vol. 40, no. 6, pp. 2305–2311, 2013.

[7] 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.

[8] 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.

[9] 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.