Analysis of Frequent Item sets and Pattern Sets Mining Algorithms

Javeriya Naaz Ishtiyaque Syed
Computer Science and Engineering
Shree HVPM's COET
Amravati, Maharashtra, India
e-mail:javeriyaisyed@gmail.com

Rajeshri R.Shelke
Computer Science and Engineering
Shree HVPM's COET
Amravati, Maharashtra, India
e-mail:rajeshrishelke@rediffmail.com

Abstract -Frequent itemsets play an essential role in many data mining tasks that try to find interesting patterns from databases, such as association rules, correlations, sequences, episodes, classifiers, clusters and many more. Many researchers invented ideas to generate the frequent itemsets. The time required for generating frequent itemsets plays an important role. Some algorithms are designed, considering only the time factor. Our study includes depth analysis of algorithms and discusses some problems of generating frequent itemsets(pattern sets) from the algorithm. We have explored the unifying feature among the internal working of various mining algorithms. Some Frequent pattern mining often produces a large number of frequent patterns, which imposes a great challenge on visualizing, understanding and further analysis of the generated patterns. This emerges the need for finding small number frequent occurring patterns. In this paper, we explain the basic frequent itemset, pattern sets mining problems. We describe the main techniques used to solve these problems and give a comprehensive survey of the most influential algorithms that were proposed during the last decade.

Keywords*: Data mining, itemsets, mining algorithms, episodes.*

I. INTRODUCTION

The data collected from different applications require proper mechanism of extracting knowledge/information from large repositories for better decision making. Knowledge discovery in databases (KDD), often called data mining, aims at the discovery of useful information from large collections of data. Data mining (sometimes called data or knowledge discovery) is the process of analysing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analysing data. It allows users to analyse data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases. The development of Information technology has generated large amount of databases and huge data in various areas. The research in databases and information technology has given rise to an approach to store and manipulate this precious data for further decision making. Data mining is a process of extraction of useful information and patterns from huge data. It is also called as knowledge discovery process, knowledge mining from data, knowledge extraction or data /pattern analysis. Data mining is a logical process that is used to search through large amount of data in order to find useful data. The goal of this technique is to find patterns that were previously unknown. Once these patterns are found they can further be used to make certain decisions for development of their businesses.

Many of the data mining techniques such as Classification, Clustering, Regression, Artificial Intelligence, Neural Networks, Association Rule, Decision Trees, Genetic Algorithm, Nearest Neighbor method the basic techniques which are implemented without considering time and space constraints[1]. For mining constrained frequent itemsets from distributed uncertain data method is a non-trivial integration of constrained mining, parallel and distributed mining, uncertain data mining, and tree-based frequent itemset mining. The technique handles different types of user-defined constraints[2]. For mining frequent items and itemsets from distributed data streams techniques such as Sketch algorithms and Data Cachers can be used[3]. For mining probabilistically frequent sequential patterns in uncertain databases techniques such as the sequence-level uncertain model, the element-level uncertain model, prefix-projection method of PrefixSpan, U-PrefixSpan Algorithms can be used[4]. Frequent pattern mining is a very important problem within the data processing space. Frequent pattern mining is typically performed on a dealings information D = {$t1$, $t2$, . . . , tn}, wherever tj could be a dealings containing a group of things, j ∈ [1, n]. Let I = {$i1$, $i2$, . . . , im} be the set of distinct things showing in D. A pattern X could be a set of things in I, that is, X⊆ I. If dealings t ∈ D contains all the things of a pattern X, then we are saying t supports X and t could be supporting dealings of X. Let T(X) be the set of transactions in D supporting pattern X. The support of X, denoted as supp(X), is outlined as |T(X)|. If the support of a pattern X is larger than a user-specified threshold min_sup, then X is termed a frequent pattern. Given a dealings information D and a minimum support threshold min_sup, the task of frequent pattern mining is to seek out all the frequent patterns in D with relevance min_sup. Several economical algorithms are developed for mining frequent patterns [5]. Here we have a tendency to analyze the bottlenecks for locating a minimum representative pattern set and develop MinRPset rule to unravel the matter. MinRPset is analogous to RPglobal, however it utilizes many techniques to cut back time period and memory usage. Specially, MinRPset uses a tree structure known as CFP-tree [6] to store frequent patterns succinctly. The CFP-tree structure conjointly supports economical retrieval of patterns that are δ-covered by a given pattern. RPglobal is commonly many orders of magnitude slower than RPlocal. In MinRPset, a representative pattern will represent its subsets solely. To any scale back the amount of representative patterns, we have a tendency to drop this condition to permit a representative pattern to represent additional patterns.

II. RELATED WORK

Data mining is the process of finding new patterns from large amount of data. Few of data mining techniques like artificial neural network, decision trees, and nearest neighbor method are frequently used. Each of these techniques analyzes data in different ways. Data mining has importance regarding finding the patterns, forecasting, and discovery of knowledge etc., in different business domains. Data mining techniques and algorithms such as classification, clustering etc., helps in finding the patterns to decide upon the future trends in businesses to grow. Data mining has wide application domain almost in every industry where the data is generated that's why data mining is considered one of the most important frontiers in database and information systems and one of the most promising interdisciplinary developments in Information Technology. But these basic techniques are implemented without considering time and space constraints.

A. Mining constrained frequent itemsets from distributed uncertain data:

Many frequent itemset mining algorithms provide little or no support for user focus when mining precise or uncertain data. However, in many real-life applications, the user may have some particular phenomena in mind on which to focus the mining (e.g., medical analysts may want to find only those lab test records belonging to patients suspected to suffer from asthma instead of all the patients). Without user focus, the user often needs to wait for a long period of time for numerous frequent itemsets, out of which only a tiny fraction may be interesting to the user. Hence, constrained frequent itemset mining which aims to find those frequent itemsets that satisfy the user-defined constraints, is needed. As technology advances, one can easily collect high volumes of massive data from not only a single source but multiple sources. For example, in recent years, sensor networks have been widely used in many application areas such as agricultural, architectural, environmental, and structural surveillance. Sensors distributed in these networks serve as good sources of data. However, sensors usually have limited communication bandwidth, transmission energy, and computational power. Thus, data are not usually transmitted to a single distant centralized processor to perform the data mining task. Instead, data are transmitted to their local (e.g., closest) processors within a distributed environment. As this requires massive computing power, this calls for parallel and distributed mining

.Nowadays, high volumes of massive data can be generated from various sources (e.g., sensor data from environmental surveillance). Many existing distributed frequent itemset mining algorithms do not allow users to express the itemsets to be mined according to their intention via the use of constraints. Consequently, these unconstrained mining algorithms can yield numerous itemsets that are not interesting to users. Moreover, due to inherited measurement inaccuracies and/or network latencies, the data are_often riddled with uncertainty. These call for both constrained mining and uncertain data mining. This propose a data-intensive computer system for tree-based mining of frequent itemsets that satisfy user-defined constraints from a distributed environment such as a wireless sensor network of uncertain data. The key contribution is the non-trivial integration of (i) constrained mining, (ii) parallel and distributed mining, (iii) uncertain data mining, (iv) tree-based mining and (v) frequent itemset mining.

B. Mining Frequent Items and Itemsets From Distributed Data Streams:

Mining data streams is a very important research topic and has recently attracted a lot of attention, because in many cases data is generated by external sources so rapidly that it may become impossible to store it and analyze it offline. In particular, data stream analysis has been carried out for the computation of items and item sets that exceed a frequency threshold. The mining approach is hybrid, that is, frequent items are calculated with a single pass, using a sketch algorithm, while frequent itemsets are calculated by a further multi-pass analysis. The architecture combines parallel and distributed processing to keep the pace with the rate of distributed data streams. In order to keep computation close to data, miners are distributed among the domains where data streams are generated. The paper [3] also reports the experimental results obtained with a prototype of the architecture, tested on a Grid composed of two domains handling two different data streams. Data stream analysis is often performed with randomized and approximated algorithms, since exact and deterministic algorithms would require too much computing time or memory space. Accordingly, data mining algorithms for data stream analysis are generally evaluated with respect to three metrics:

• The processing time of the operations that update the data structures and the mining models after the arrival of a new stream item;
• The storage space used by the algorithm;
• The accuracy of the approximated algorithm, in general specified through two parameters set by the user: the accuracy parameter ϵ and the failure probability δ, which means that the estimation error is at most ϵ with probability $(1 - \delta)$. Of course, processing time and storage size strongly depend on these parameters.

The architecture present in the paper[3] addresses the issues mentioned above by exploiting the following main features:

• the architecture combines the parallel and distributed paradigms, the first to keep the pace with the rate of a single data stream, by using multiple miners (processors or cores), the second to cope with the distributed nature of data streams. Miners are distributed among the domains where data streams are generated, in order to keep computation close to data.
• the computation of frequent items is performed through *sketch* algorithms. These algorithms maintain a matrix of counters, and each item of the input stream is associated

with a set of counters, one for each row of the table, through hash functions. The statistical analysis of counter values allows item frequencies to be estimated with the desired accuracy. Sketch algorithms compute a linear projection of the input: thanks to this property, sketches of data can be computed separately for different stream sources, and can then be integrated
to produce the overall sketch [7].

• the approach is *hybrid*, meaning that frequent items are calculated online, with a single pass, while frequent itemsets are calculated by a further multi-pass analysis. This kind of approach allows important information to be derived on the fly without imposing too strict time constraints on more complex tasks, such as the extraction of frequent k-itemsets, as this could excessively lower the accuracy of models.

• to support the mentioned hybrid approach, the architecture exploits the presence of *data cachers* on which recent data can be stored. In particular, miners can turn to data cachers to retrieve the statistics about frequent items and use them to identify frequent *sets* of items. To avoid excessive communication overhead, data cachers are distributed and placed close to stream sources and miners.

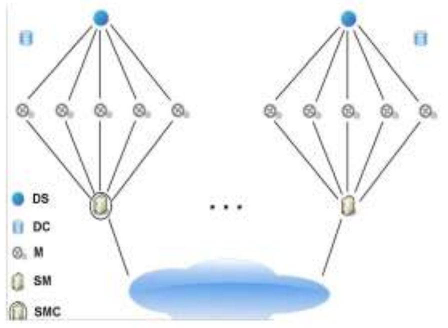

Figure: Distributed architecture for data stream mining.

C. *Mining Probabilistically Frequent Sequential Patterns in Uncertain Databases:*

Data uncertainty is inherent in many real-world applications such as environmental surveillance and mobile tracking. As a result, mining sequential patterns from inaccurate data, such as sensor readings and GPS trajectories, is important for discovering hidden knowledge in such applications. Previous work uses *expected support* as the measurement of *pattern frequentness*, which has inherent weaknesses with respect to the underlying probability model, and is therefore ineffective for mining high-quality sequential patterns from uncertain sequence databases. In paper[4] propose to measure pattern frequentness based on the *possible world semantics*. It establish two uncertain sequence data models abstracted from many real-life applications involving uncertain sequence data, and formulate the problem of mining *probabilistically frequent sequential patterns* (or *p-FSPs*) from data that conform to our models. Based on the *prefix-projection* strategy of the famous *PrefixSpan* algorithm, it develop two new algorithms, collectively called *U-PrefixSpan*, for p-FSP mining. *UPrefixSpan* effectively avoids the problem of "possible world explosion", and when combined with our three pruning techniques and one validating technique, achieves good performance. The efficiency and effectiveness of *U-PrefixSpan* are verified through extensive experiments on both real and synthetic datasets. It is found that the patterns found by *ElemU-PrefixSpan* is accurate in terms of locations on the hallways, although they may not be accurateenough for detecting local events such as entering/exiting a room. The experiments verify that the patterns found by *ElemUPrefixSpan* is useful for trajectory mining tasks in RFID applications. We expect that our *U-PrefixSpan* algorithms would also be useful for frequent sequential pattern mining in many other real world applications involving uncertain data.

Algorithm : *SeqU-PrefixSpan*($\alpha e, D|\alpha, T|\alpha$)

```
Input: current pattern $\alpha e$, projected probabilistic database $D|\alpha$,
element table $T|\alpha$
1: $vec\alpha e \leftarrow \Phi$
2: for each projected sequence $s_i|\alpha \in D|\alpha$ do
3: $pr(s_i|\alpha e) \leftarrow 0$
4: for each instance $s_{ij} |\alpha = <pos, Pr(s_{ij})> \in s_i|\alpha$ do
5: Find its corresponding sequence $s_{ij} \in D$
6: if $e \in s_{ij} [pos + 1, . . . , len(s_{ij} )]$ then
7: $pr(s_i|\alpha e) \leftarrow pr(s_i|\alpha e) + Pr(s_{ij})$
8: $c' \leftarrow \min c \geq pos+1\{s_{ij} [c] = e\}$
9: Append $(c, pr(s_{ij}))$ to $s_i|\alpha e$
10: if $pr(s_i|\alpha e) > 0$ then
11: Append $s_i|\alpha e$ to $D|\alpha e$
12: Append $pr(s_i|\alpha e)$ to $vec\alpha e$
13: $(tag, f\alpha e) \leftarrow PMFCheck(vec\alpha e)$
14: if $tag$ =TRUE then
15: output $\alpha e$
16: $T|\alpha e \leftarrow Prune(T|\alpha, D|\alpha e)$
17: for each element _ $\in T|\alpha e$ do
18: $SeqU\text{-}PrefixSpan(\alpha e\_, D|\alpha e, T|\alpha e)$
19: Free $D|\alpha e$ and $T|\alpha e$ from memory
```

D. *MinRPset Algorithm for finding frequent pattern sets:*

The number of frequent patterns may be terribly massive. Besides frequent closed patterns, many alternative ideas, like generators [8], disjunction-free generators [9], δ-free sets [10], non-derivable patterns [11], maximal patterns [12], top-k frequent closed patterns [13] and redundancy-aware top patterns [14], are planned to scale back pattern set size. the amount of generators is larger than that of close patterns. Moreover, the set of generators itself isn't lossless. It needs a border to be lossless [9], therefore will the set of disjunction-free generators and δ-free sets. the amount of non-derivable patterns also can be larger than that of closed patterns on some datasets. the amount of greatest patterns is far smaller than the amount of closed patterns. All frequent patterns may be recovered from greatest patterns, however their support information is lost. Another work that additionally ignores the support information is [15]. It selects k patterns that best cowl a group of patterns. If the amount of frequent closed patterns is higher than 1,000,000 than another formula known as FlexRPset, that provides one further parameter K to permit users to form a trade-off between result size and potency. Frequent closed patterns preserve the precise support of all frequent patterns. In several applications, knowing the approximate support of frequent patterns is enough. Many approaches are planned to form a trade-off between pattern set size and therefore the exactitude of pattern support. Another approach planned by I. M. Pei et al. [16] uses absolute error sure. It uses heuristic algorithms to mine a token condensed pattern-base, that could be a superset of the greatest pattern set. All frequent patterns and their support may be repaired from a condensed pattern-base with error guarantee. Many enhancements are created to the profile primarily based approach. Jin et al. [17] have developed a regression primarily based approach to reduce restoration error. They cluster patterns supported restoration errors rather than similarity between patterns, so their approach can do lower restoration error. However, there's still no error guarantee on the repaired support. CP-summary [18] uses conditional independence to scale back restoration error. It adds an added part to every profile: a pattern base, and therefore the new profile is named c-profile. The things in a very c-profile are expected to be freelance with relation to the pattern base. CP-summary provides error guarantee on calculable support. However, patterns of a c-profile typically share very little similarity, therefore a c-profile isn't representative of its patterns any further. Profiles may be thought of as generalizations of closed patterns. Wang et al. [19] build generalization on another succinct illustration of frequent patterns non-derivable patterns. They use mathematician Random Field (MRF) to summarize frequent patterns. The support of a pattern is calculable from its subsets, that is comparable to non-derivable patterns. Mathematician Random Field model isn't as intuitive as profiles, and it's additionally costly to find out. It doesn't give errorguarantee on calculable support either. The higher than approaches aim to summarize frequent patterns. Mampaey et al. [20] aim to summarize knowledge instead with a group of non-redundant patterns. A probabilistic most entropy model is employed in their approach.

III. ANALYSIS

A. *Mining constrained frequent itemsets from distributed uncertain data:*

The resulting system uses a tree-based approach to efficiently mine from distributed uncertain data for only those constrained frequent itemsets. It avoids the candidate generate and-test paradigm, handles uncertain data, pushes user constraints inside the mining process, avoids unnecessary computation, and finds only those itemsets satisfying the constraints in a distributed environment. But the drawback of these technique is when the number of sites are increased the runtime increases and it is found that, the higher the selectivity of the constraints, the longer was the runtime for this system.

B. *Mining Frequent Items and Itemsets From Distributed Data Streams*:

The distributed stream mining system presented in paper[3] is a contribution in the field and it aims at solving the problem of computing frequent items and frequent itemsets from distributed data streams by exploiting a hybrid single-pass/multiple-pass strategy. We assumed that stream sources, though belonging to different domains, are homogenous, so that it is useful to extract knowledge from their union. Beyond presenting the system architecture, it described a prototype that implements it and discussed a set of experiments performed in a real Grid environment. The experimental results confirm that the approach is scalable and can manage large data production by using an appropriate number of miners in the distributed architecture. Bt in Experiments with the dataset, results show that the system is stable (i.e., the processing time is shorter than the time period) only when the threshold is as high as 0.03 and at least 5 miners per domain are available.

C. *Mining Probabilistically Frequent Sequential Patterns in Uncertain Databases:*

It formulate and study the problem of mining probabilistically frequent sequential patterns (or *p- SPs*) in uncertain databases. Their study is founded on two uncertain sequence data models that are fundamental for many real-life applications involving uncertain sequence data. They propose two new *U-PrefixSpan* algorithms to mine p-FSPs from data that conform to our sequence level and element-level uncertain sequence models. They also design three pruning rules and one early validating method to speed up pattern frequentness checking, which further improve the mining efficiency. Experiments show that this algorithms effectively avoid the problem of “possible world explosion”, and the trajectory patterns found by *ElemU-PrefixSpan* in an RFID tracking application are shown to be accurate and useful. Bt the patterns found by *ElemU-PrefixSpan* is accurate in terms of locations on the hallways, although they may not be accurate enough for detecting local events such as entering/exiting a room.

D. *MinRPset Algorithm for finding frequent pattern sets:*

MinRPset algorithm first mine frequent patterns, and then find representative patterns in a post-processing step, while RPlocal integrates frequent pattern mining with representative pattern finding. Due to the use of the post-processing strategy, MinRPset have the following additional benefits besides producing fewer representative patterns:

- Users may not know what value should be used for ϵ at the beginning. The post-processing strategy allows users to try different ϵ values without mining frequent patterns multiple times. This is especially beneficial on very large datasets.
- In MinRPset and FlexRPset, it is easy to keep record of the set of patterns covered by each representative pattern. This information is useful for users to inspect individual representative patterns in more details.
- We can relax the conditions on ϵ -covered to further reduce the number of representative.

IV. CONCLUSION

In this paper, a comparison framework has developed to allow the flexible comparison of existing and new frequent itemset(pattern set) mining algorithms that conform to the defined algorithm interface. Using this framework this paper presented the comparative performance study of various algorithms. In this work, an in-depth analysis of few algorithms is done which made a significant contribution to the search of improving the efficiency of frequent itemset mining. The developed framework can be used for comparing the other algorithms, which does not use candidate set generation to discover frequent patterns. And can also lead to several ideas for optimizations, which could improve the performance of other algorithms. MinRPset algorithm for locating minimum representative pattern sets is introduced. It mines frequent patterns, and then find representative patterns during a post-processing step, while RPlocal integrates frequent pattern mining with representative pattern finding. As a result of the utilization of the post-processing strategy, MinRPset have the extra benefits besides giving fewer representative patterns.

REFERENCES

[1] " Data Mining: Tools and Techniques" ,Kumud Sharma, Rajwinder Kaur, Kavita Sharma in "International Forum of Researchers Students and Academician".

[2] "Mining constrained frequent itemsets from distributed uncertain data" by Alfredo Cuzzocrea, Carson Kai-Sang Leung, Richard Kyle MacKinnon in 2013 Elsevier B.V.

[3] "A Sketch-based Architecture for Mining Frequent Items and Itemsets from Distributed Data Streams" by Eugenio Cesario, Antonio Grillo, Carlo Mastroianni, Domenico Talia in 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing.

[4] "Mining Probabilistically Frequent Sequential Patterns in Uncertain Databases" by Zhou Zhao, Da Yan and Wilfred Ng in EDBT *2012*, March 26–30, 2012, Berlin, Germany.

[5] B. Goethals and M. J. Zaki, "Advances in frequent itemset mining implementations: Introduction to FIMI'03," in *Proc. ICDM*, 2003.

[6] G. Liu, H. Lu, and J. X. Yu, "CFP-tree: A compact disk-based structure for storing and querying frequent itemsets," *Inf. Syst.*, vol. 32, no. 2, pp. 295–319, 2007.

[7] G. Cormode and M. Hadjieleftheriou, "Finding the frequent items in streams of data," *Communications of the ACM*, vol. 52, no. 10, pp. 97–105, 2009.

[8] Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal, "Mining minimal non redundant association rules using frequent closed itemsets," in *Proc. 1st Int. Conf. CL*, London, U.K., 2000, pp. 972–986.

[9] A. Bykowski and C. Rigotti, "A condensed representation to find frequent patterns," in *Proc. PODS*, New York, NY, USA, 2001, pp. 267-273.

[10] J.-F. Boulicaut, A. Bykowski, and C. Rigotti, "Free-sets: A condensed representation of boolean data for the approximation of frequency queries," *Data Mining Knowl. Discov.*, vol. 7, no. 1, pp. 5–22, 2003.

[11] T. Calders and B. Goethals, "Mining all non-derivable frequent itemsets," in *Proc. PKDD*, Helsinki, Finland, 2002, pp. 74–85.

[12] R. J. Bayardo, "Efficiently mining long patterns from databases," in *Proc. SIGMOD*, New York, NY, USA, 1998, pp. 85–93.

[13] J. Wang, J. Han, Y. Lu, and P. Tzvetkov, "TFP: An efficient algorithm for mining top-k frequent closed itemsets," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 5, pp. 652–664, May 2005.

[14] D. Xin, H. Cheng, X. Yan, and J. Han, "Extracting redundancy aware top-k patterns," in *Proc. KDD*, Philadelphia, PA, USA, 2006, pp. 444–453.

[15] F. N. Afrati, A. Gionis, and H. Mannila, "Approximating a collection of frequent sets," in *Proc. KDD*, Washington, DC, USA, 2004, pp. 12–19.

[16] J. Pei, G. Dong, W. Zou, and J. Han, "Mining condensed frequent pattern bases," *Knowl. Inform. Syst.*, vol. 6, no. 5, pp. 570–594, 2004.

[17] R. Jin, M. Abu-Ata, Y. Xiang, and N. Ruan, "Effective and efficient itemset pattern summarization: Regression-based approaches," in *Proc. KDD*, Las Vegas, NV, USA, 2008, pp. 399–407.

[18] A. K. Poernomo and V. Gopalkrishnan, "CP-summary: A concise representation for browsing frequent itemsets," in *Proc. KDD*, New York, NY, USA, 2009, pp. 687–696.

[19] C. Wang and S. Parthasarathy, "Summarizing itemset patterns using probabilistic models," in *Proc. KDD*, Philadelphia, PA, USA, 2006, pp. 730–735.

[20] M. Mampaey, N. Tatti, and J. Vreeken, "Tell me what I need to know: Succinctly summarizing data with itemsets," in *Proc. KDD*,San Diego, CA, USA, 2011, pp. 573–581.

[21] A Flexible Approach to Finding Representative Pattern Set, Guimei Liu, Haojun Zhang, and Limsoon Wongs, IEEE Transactions on knowledge and data engineering, vol. 26, no. 7, july 2014