International Journal of Engineering & Technology, 7 (4.1) (2018) 134-137

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paperer

Performance of IF-Postdiffset and R-Eclat Variants in Large Dataset

Julaily Aida Jusoh[1], Mustafa Man[2]*, Wan Aezwani Wan Abu Bakar[1]

[1]Faculty of Informatic & Computing, Universiti Sultan Zainal Abidin, Tembila Campus, 22200 Besut, Terengganu, Malaysia
[2]School of Informatics and Applied Mathematics, Universiti Malaysia Terengganu, 21030 Kuala Nerus, Terengganu, Malaysia
**Corresponding author E-mail: mustafaman@umt.edu.my*

Abstract

Pattern mining refers to a subfield of data mining that uncovers interesting, unexpected, and useful patterns from transaction databases. Such patterns reflect frequent and infrequent patterns. An abundant literature has dedicated in frequent pattern mining and tremendous efficient algorithms for frequent itemset mining in the transaction database. Nonetheless, the infrequent pattern mining has emerged to be an interesting issue in discovering patterns that rarely occur in the transaction database. More researchers reckon that rare pattern occurrences may offer valuable information in knowledge data discovery process. The R-Eclat is a novel algorithm that determines infrequent patterns in the transaction database. The multiple variants in the R-Eclat algorithm generate varied performances in infrequent mining patterns. This paper proposes IF-Postdiffset as a new variant in R-Eclat algorithm. This paper also highlights the performance of infrequent mining pattern from the transaction database among different variants of the R-Eclat algorithm regarding its execution time.

Keywords: *Pattern mining; Itemset mining; Infrequent itemset mining; R-Eclat algorithm; Large dataset.*

1. Introduction

With the emergence of big data transformation, companies cannot store all their records for long durations and inefficiently manage huge datasets. Past technologies have limited storage capacity, rigid, and expensive management tools. Their lack of scalability, flexibility, and performance needs to be addressed within the big data context. The management of big data requires important resources, new methods, and powerful technologies. In precise, big data requires clean, process and analysis, security, and granular access to the massively evolving datasets ([1,2]). Most industries are aware that data analysis is increasingly important to be competitive in discovering new knowledge and personalizing services.

In pattern mining, association rule mining (ARM) ([3,4]) plays an essential role in mining association and correlations among itemsets in the dataset. The ARM determines association rules that satisfy predefined minimum support and confidence from a given database. Support and confidence are a measures parameter of the rule. Support indicates the frequency of pattern, while confidence specifies the strength of rule implication.

Two major patterns can be found in the datasets which are frequent and infrequent itemsets. Both patterns present dissimilar information about the datasets. The frequent itemset deals with frequent occurrences of data items, while the infrequent itemset looks into rare occurrences of data items. Recently, infrequent itemset mining has garnered much interest in pattern mining. The objective is to seek infrequent grouping of items that rarely occurs in databases that contain a series of item transactions. The ARM algorithms extract frequent itemsets to display high frequency/support in the transaction database. Infrequent itemsets with low support can yield potential and important association rules (ARs) vital for building a reliable decision support system.

To date, infrequent pattern mining has been widely used in computer science studies. In fact, 25 articles found on Scopus in the year 2008 explicitly mentioned "infrequent mining" in their abstracts, title or keywords, wherein this number had increased rapidly in the year 2017. Figure 1 illustrates the trends in infrequent pattern mining domain for a decade between 2008 and August 2018.

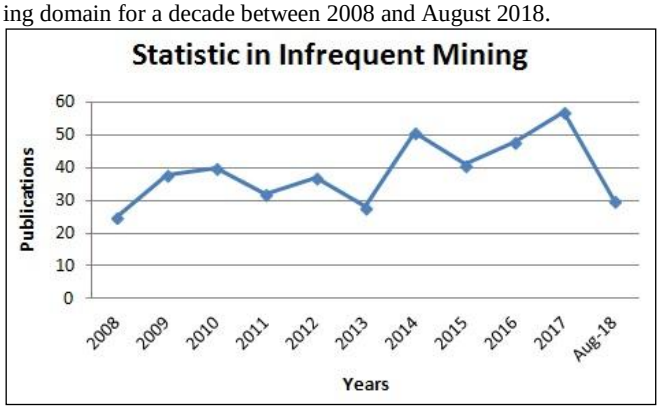


Fig. 1: Trends of Infrequent Pattern Mining Domain for A Decade

The task of infrequent mining patterns has become prominent and has intensively studied issues in data mining. Many algorithms have been designed and developed to address issues related to computational analysis and memory requirements. The two basic algorithms in infrequent itemset mining are Apriori ([5,6]) that lies in horizontal format, and FP-Growth ([7,8]) with its database structure in vertical format. Both algorithms have their advantages and disadvantages concerning sparse or dense datasets. The main drawback of Apriori is the recurrent generating candidate sets in seeking the itemsets. Such generation of candidates requires lengthy execution time, more memory usage, and high computational cost. The FP-Growth discovers itemsets by dividing and conquering methods,

as well as building a compact data structure known as FP-Tree. These demands less memory usage due to its compact structure in uncovering itemsets without candidate itemset generation.

This paper focuses on the third algorithm in itemset mining known as Eclat algorithm. Zaki *et al*. ([9, 10]) proposed the ECLAT algorithm to mine frequent itemsets. ECLAT algorithm is renowned for its rapid intersection of its tid-list, where the resulting number of tids is referring to the support (frequency) of each itemset. The latest variant in ECLAT algorithm is Postdiffset. The Postdiffset was developed by Bakar *et al*. ([11, 12]) for frequent itemsets mining to minimize the requirement of checking the switching condition. Although this variant suffers from lengthy time processing when compared to other variants in ECLAT algorithm, it does not require much memory.

The first algorithm that inspired by the conventional ECLAT algorithm ([9-10,13]) for infrequent itemset mining is R-Eclat algorithm. The R-Eclat algorithm was initiated by Jusoh *et al*. ([14, 15]) to distinguish infrequent patterns in a large dataset. The R-Eclat algorithm adopts a vertical layout to represent the transaction database and uses a depth-first search to achieve a condensed representation of the transaction database. Three variants have been proposed in the current R-Eclat algorithm; IF-Tidset, IF-Diffset, and IF-Sortdiffset ([14, 15]). However, these variants have different accomplishments, especially regarding time processing during the experiment. Motivated by the good performance of Postdiffset, this paper introduces IF-Postdiffset to mine infrequent itemsets. This paper also ran the comparative analysis of runtime processing among dissimilar R-Eclat format variants.

2. Overview of Infrequent Itemset Mining

In the traditional itemsets mining problem, items that belong to the transaction data are equally treated. In order to allow items differentiation based on their interest or intensity within each transaction, many researches have focused on discovering more informative association rules. A transaction refers to a record of one or more items collected from a finite item domain, while a dataset is a collection of transactions, and itemset is a non-empty subset of items.

For infrequent itemsets mining from transactional datasets, let's assume itemset $I=\{i_1, i_2, \ldots, i_m\}$ is a set of data items. The *k*-itemset is denoted as a set of *k* items in *I*. A transactional dataset $T= \{t_1, t_2, \ldots, t_n\}$ represents a set of transactions, where each transaction t_α ($\alpha \in [1, n]$) is a set of items in *I* and is characterized by a transaction ID (tid). The support of an itemset *X* is defined as the number of transactions satisfied by the itemset, the itemset being frequent if $\{support\ (X) \geq minimum\_support \mid support\ (X) = |S|, S \subseteq T\}$, $|S|$ is the number of transactions satisfied by the itemset. The support (frequency of occurrence of an itemset) of an itemset reflects the number of transactions that contain *I* in *T*. Itemset *I* is infrequent if its support is less than or equal to the minimum support threshold. Otherwise, it is called frequent. Given a transactional dataset *T* and a minimum support threshold, the infrequent itemset mining problem determines all infrequent itemsets from the transactional dataset.

The Apriori algorithm is the most common ARM algorithm. It discovers valid rules using support and confidence requirements while the minsup thresholds are used to prune the search space. However, it is inefficient to seek low-support rules. In Apriori, thousands of itemsets need to be combed through (with high support) to identify infrequent itemsets. Based on the Apriori features, a few extensions need to be enhanced to assure that it is suitable to adopt the infrequent mining. Rahman ([5]) introduced the Online Apriori-Infrequent algorithm that does not use the confidence value parameter in measuring infrequent itemsets. It efficiently uses support to calculate an *anomaly score* for the record to determine if the record is anomalous or not on the fly. An *anomaly score* is assigned to each packet (record) based on the frequent or infrequent patterns of the. This algorithm recovers the join and prune steps of the traditional Apriori algorithm with a constraint. The constraint hinders the joining of itemsets and does not generate frequent itemsets as the output, thus enhancing efficiency and run times significantly.

Liu *et al*. ([6]) dealt with infrequent itemset using multiple minsup thresholds by designing MSApriori algorithm. In this algorithm, some items can have low support that does not contribute to the rules generated by Apriori, although it may participate in rules with high confidence. This issue can be addressed when each item in the database can have its minimum item support (MIS). By providing a dissimilar MIS for dissimilar items, a higher minsup is set for frequent itemset, while lower minsup for infrequent items.

In the Apriori-based methods, a huge number of candidate sets needs to be produced. Hence, Han *et al*. ([16]) devised a solution based on a compact tree structure called FP-Tree. This algorithm can reduce the database scanning by considering a divide-and-conquer technique. It has been proven to perform faster than Apriori.

The RP-Tree algorithm was developed by Tsang *et al*. ([7]) to avoid itemset generation and pruning steps by replacing into a tree structure. It is designed based on FP-Tree and uses a two-pass approach to identify infrequent patterns. The RP-Tree count the item support by performing a database scanning. In the second scanning, the transactions that include at least a single rare item are used to build the initial tree and to prune the others. Besides, it is efficient in seeking long patterns as the task is fragmented into a series of searches for short patterns.

Gupta ([8]) proposed a technique called minimally infrequent itemsets (MII). It is designed based on Inverse FP-Tree (IFP). In MII, the IFP-Tree is separating into two subtrees called as projected and residual. The projected subtree corresponds to the set of transactions that contains a particular item. From the projected subtree, a potential minimal infrequent itemset did not have any infrequent subset since the itemset itself is a subset. Meanwhile, a residual tree for a particular item is a tree representation of the residual database that corresponds to the item. Residual trees can minimize computation time and suitable for large and dense datasets.

Maximum Constraint-based Conditional Frequent Pattern-growth (MCCFP) is an extension of the FP-growth algorithm. It constructs the initial tree (in descending order) from all items in the transaction dataset by using numerous minimum support. It is aimed to determine a list of frequent items or patterns via a single scan on the dataset. As MCCFP builds a tree in the downward order of items, it needs extra memory. It is still more efficient than Apriori-based approach because it avoids combinatorial explosion of candidate generation and numerous scans on a transactional dataset.

Based on the previous discussion, the Apriori approach demands a set of candidates for a generation, while the tree-based approach requires a tree construction to detect infrequent itemsets. Both techniques are applied in the horizontal data format. Jusoh *et al*. ([14, 15]) proposed R-Eclat algorithm as an alternative for infrequent itemsets mining. A brief description of R-Eclat variants is given in the next section.

3. Proposed IF-Postdiffset Variant in R-Eclat Algorithm

The R-Eclat algorithm is an innovative technique that is specially devised for infrequent pattern mining ([14, 15]). It is motivated based on the traditional ECLAT algorithm, which uses a depth-first search and adopts a vertical layout to represent the transaction database. An important feature of R-Eclat is that it has speedy support counting by determining support of any k-itemset by intersecting tid-lists of its k-1 subsets. The R-Eclat algorithm has different format variants known as IF-Tidset, IF-Diffset, and IF-Sortdiffset. Thus, the size of tidsets is a vital factor that affects execution time and memory usage. IF-Tidset takes more space to store candidate set and needs more time for intersection when tid-list is large. In IF-Diffset, the cardinality of sets representing itemsets is decreased and yields in the faster intersection and less memory usage. But, in IF-Sortdiffset, it requires a high cost in executing the tid-list sorting task. As a continuity to the current format variants of the R-Eclat, this paper suggests the modification of the latest variant on Eclat algorithm known as Postdiffset. In frequent mining, Postdiffset used tidset format at starting the process and later switch to diffset

format when switching condition is met. This paper proposes an improvised Postdiffset format variant to mine infrequent pattern in the transaction database, as shown in Figure 2. This new format is introduced as IF-Postdiffset.

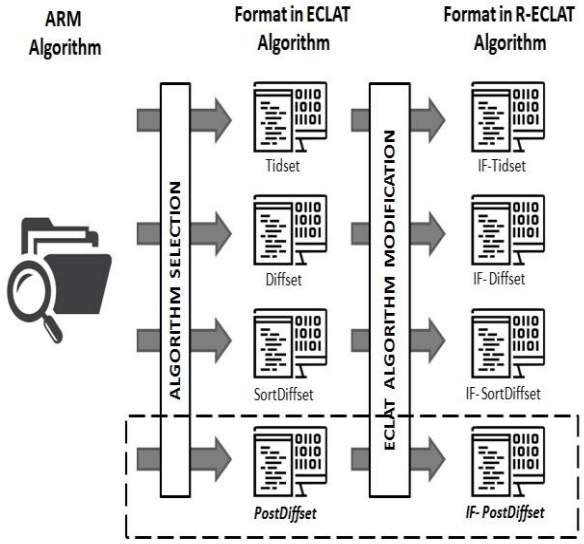


Fig. 2: The proposed IF-Postdiffset format variants in R-Eclat Algorithm

Figure 3 illustrates the step-by-step actions taken in the IF-Postdiffset format. The minsup threshold value is determined regarding percentage, where the user-specified minsup value is divided by 100 and multiplied with the total rows (records) of each dataset. Next, in each loop, starting with the first loop, if the support is less than or equal (≤) to minsup, the first level of looping is based on tidset process, and followed by the second level onwards of looping to gain variance set intersection between i^{th} column and $i+1^{th}$ column. Finally, the values of the itemsets/data transaction are saved to the database.

Pseudocode IF-Postdiffset

```
Input: Transaction data
Output: Infrequent itemsets
Begin   // get minimum support
 1.   min_supp=number_of_rows * percent-
      age_min_support
 2.   Run tidset in first loop;
 3.     if (support ≤ min_support) {
 4.       add data to the next process;
 5.       add data into DB }
 6.   Run diffset in next looping;
 7.     if (support ≤ min_support) {
 8.       add data to the next process;
 9.       add data into DB }
10.   end looping;
11.   end diffset;
12.   Write to file the value for the current / last trans-
      action  data;}
End;
```

Fig. 3: Pseudocode of IF-Postdiffset

4. Experimental Observations

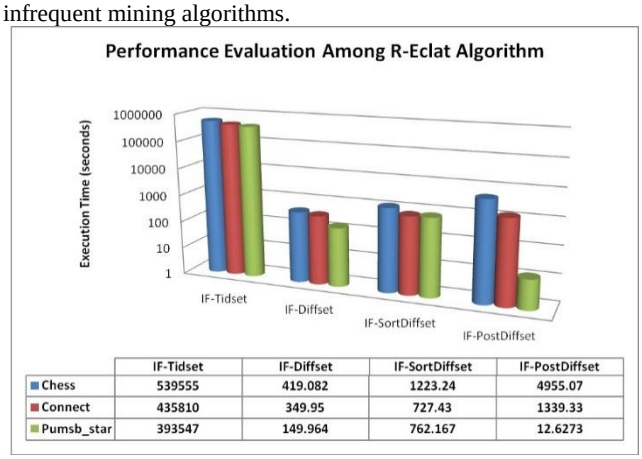
Experiments were run using IF-Tidset, IF-Diffset, IF-Sortdiffset, and IF-Postdiffset algorithms. The raw benchmark datasets were retrieved from Frequent Itemset Mining Dataset Repository (*http://fimi.ua.ac.be/data/*) in a *.dat file format. To simplify, the selected benchmark datasets were transformed into Structured Query Language (SQL) format. Three datasets in the dense category, including chess, pumsb_star, and connect, were used in this experiment. The characteristics of each dataset are summarized in Table 1. All experiments were performed on HP Notepad, Intel ® Core ™ ® i7-4210U CPU @ 2.40 GHz with 8GB RAM, in a Win10 64-bit platform.

Table 1: Dataset Characteristics

Dataset	No. of Transactions	Length (Attributes)	Size (KB)
Chess	3196	37	335
Pumsb_star	49046	57	11526
Connect	65536	43	191134

As for the performance analysis targets, the datasets were modified and randomly selected into three thousand rows (records) of data for mining purposes.

In the chess dataset, the IF-Tidset completely lost its performance over three other formats, where it was executed in 539555 seconds, when compared to IF-Diffset (419.082), IF-Sortdiffset (1223.24), and IF-Postdiffset (4955.07). Upon connection, the IF-Diffset outperformed the 349.95 seconds record, whereas the IF-Tidset scored 435810, IF-Sortdiffset with 727.43, and IF-Postdiffset with 1339.33 execution time (seconds). As for the pumsb_star, the IF-Postdiffset displayed extreme performance with only 12.6273 seconds. Overall, the IF-Diffset completed the experiment with an outstanding performance among chess and connect in mining infrequent itemsets. Nonetheless, the IF-Postdiffset performed better in pumsb_star. The results tabulated in Figure 4 indicate that the nature of datasets concerning the occurrence frequency of itemsets could be a contributing factor to the overall performance of certain association rule infrequent mining algorithms.



	IF-Tidset	IF-Diffset	IF-SortDiffset	IF-PostDiffset
Chess	539555	419.082	1223.24	4955.07
Connect	435810	349.95	727.43	1339.33
Pumsb_star	393547	149.964	762.167	12.6273

Fig. 4: Performances on IF-Tidset, IF-Diffset, IF-Sortdiffset, and IF-Postdiffset in chess, connect, and pumsb_star

5. Conclusion

Itemset mining is a dynamic field of research in association rules. Infrequent patterns are considered significant due to the huge impact they have on various domain applications. This paper addressed the issue of discovering infrequent itemsets by using a support measure in differentiating the relevant items from those irrelevant in each transaction. As for the next direction in experimentation, sparse datasets will be tested, such as retail and T10I4D100K, to identify if they display a similar trend of outcomes.

Acknowledgement

We express our gratitude to MyPhD scholarship under SLAB of Kementerian Pendidikan Malaysia (KPM) and Niche Research Grant Scheme (NRGS), Vote 53131 for the financial support for this work. We also wish to thank Mohd Hafizuddin Ibrahim from Politeknik Kuala Terengganu for supporting our work by reviewing spelling errors and synchronizing consistencies, including providing meaningful comments and suggestions.

References

[1] Djenouri Y, Djenouri D, Habbas Z & Belhadi A, "How to Exploit High Performance Computing in Population-Based Metaheuristics for Solving Association Rule Mining Problem", *Distrib. Parallel Databases*, Vol.36, No.2, (2018), pp.369-397.

[2] Yacoob I, Hashem IAT, Gani A, Mokhtar S, Ahed E, Anuar NB & Vasilakos AV, "Big Data: From Beginning to Future", *International Journal of Information Management*, Vol.36, No.6, (2016), pp.1231-1247.
[3] Agrawal R, Imielinski T & Swami A, "Mining Association Rules Between Sets Of Items In Large Databases", *ACM SIGMOD*, Vol.22, No.2, (1993), pp.207-216.
[4] Agrawal R & Srikant R, "Fast Algorithms For Mining Association Rules In Large Databases", *Proc. 20th International Conference on Very Large Data Bases (VLDB)*, September 12–15, Santiago de Chile, Chile, (1994), pp.487- 499.
[5] Rahman A, Ezeife CI & Aggarwal AK, "WiFi miner: An Online Apriori-Infrequent Based Wireless Intrusion System", *Knowledge Discovery from Sensor Data*, Lecture Notes in Computer Science, Vol.5840, Springer, Berlin, (2010), pp.76-93.
[6] Liu B, Hsu W & Ma Y, "Mining association rules with multiple minimum supports", *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (1999), pp.337-341.
[7] Tsang S, Koh YS & Dobbie G, "RP-tree: Rare Pattern Tree Mining", *DaWaK (Lecture Notes in Computer Science), Alfredo Cuzzocrea and Umeshwar Dayal (Eds.), Springer*, Berlin, Vol.6862, (2011), pp.277-288.
[8] Gupta A, Mittal A & Bhattacharya A, "Minimally Infrequent Itemset Mining Using Pattern-Growth Paradigm And Residual Trees", *CoRR abs/1207.4958*, (2012).
[9] Zaki MJ, Parthasarathy S, Ogihara M, Li W *et al*., "New algorithms for fast discovery of association rules", *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (1997), pp.283-286.
[10] Zaki MJ & Gouda K, "Fast Vertical Mining Using Diffsets", *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (2003), pp.326-335.
[11] Bakar W, Jalil MA, Man. M, Abdullah Z & Mohd F, "Postdiffset: an Eclat-like algorithm for frequent itemset mining", *International Journal of Engineering & Technology*, Vol.7, No.2.28, (2018), pp.197-199.
[12] Bakar W, Man M & Jusoh JA, "Comparative performance analysis of postdiffset in frequent vs. infrequent Itemset mining", *International Journal of Engineering & Technology*, Vol.7, No.3.28, (2018), pp.144-148.
[13] Trieu TA, Kunieda Y, "An Improvement For Declat Algorithm", *Proc. 6th International Conference on Ubiquitous Information Management and Communication*, No.54, (2012).
[14] Jusoh JA, Man M & Bakar W, "Mining Infrequent Patterns Using R-Eclat Algorithms", *Journal of Fundamental and Applied Sciences*, Vol.24, No.3, (2018).
[15] Jusoh JA & Man M, "Modifying iEclat Algorithm for Infrequent Patterns Mining", *Advanced Science Letters*, Vol.24, No.3, (2018).
[16] Han J, Pei J, Yin Y & Mao R, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach", *Data Mining and Knowledge Discovery*, Vol.8, (2004), pp.53-87.