R-Diffset vs. IR-Diffset: Comparison Analysis in Dense and Sparse Data

Julaily Aida Jusoh, Sharifah Zulaikha Tengku Hassan, Wan Aezwani Wan Abu Bakar,
Syarilla Iryani Ahmad Saany, Mohd Khalid Awang, Norlina Udin @ Kamaruddin
Faculty of Informatics and Computing, University Sultan Zainal Abidin, Besut, Terengganu, 22200, Malaysia.

***Abstract*—The mining of concealed information from databases using Association Rule Mining seems to be promising. The successful extraction of this information will give a hand to many areas by aiding them in the process of finding solutions, economic projecting, commercialization policies, medical inspections, and numbers of other problems. ARM is the most outstanding method in the mining of remarkable related configurations from any groups of information. The important patterns encountered are categorized as recurrent/frequent and non-recurrent/infrequent. Most of the previous data mining methods concentrated on horizontal data set-ups. Nevertheless, recent studies have shown that vertical data formats are becoming the main concerns. One example of vertical data format is Rare Equivalence Class Transformation (R-Eclat). Due to its efficacy, R-Eclat algorithms have been commonly applied for the processing of large datasets. The R-Eclat algorithm is actually comprised of four types of variants. However, our work will only focus on the R-Diffset variant and Incremental R-Diffset (IR-Diffset). The performance analysis of the R-Diffset and IR-Diffset algorithms in the mining of sparse and dense data are compared. The processing time for R-Diffset algorithm, especially for sequential processing is very long. Thus, the incremental R-Diffset (IR-Diffset) has been established to solve this problem. While R-Diffset may only process the non-recurrent itemsets mining process in sequential form, IR-Diffset on the other hand has the capability to execute sequential data that have been fractionated. The advantages of this newly developed IR-Diffset may become a potential candidate in providing a time-efficient data mining process, especially those involving the large sets of data.**

Keywords*—R-Diffset; IR-Diffset; dense data; sparse data; comparison analysis

I. INTRODUCTION

The main objective of data mining (DM) is the process of identifying patterns for useful information in large data repositories [1-3, 36, 26, 30]. It plays a key role in the oversized data approach to acquire meaningful knowledge from a complex system. The approach includes collaborative studies in statistics, machine learning, data science, and database theory. Its goal is to learn about the past or predict the future by studying the present data. In addition, this approach, also known as "Knowledge Discovery in Databases" (KDD), focuses on finding patterns in databases, resulting in an association rule that may disclose important information. Some common patterns are found in the databases, such as clusters, sets of items, trends, and outliers [4]. In data mining, there are two types of tasks: prediction tasks and descriptive tasks. The prediction tasks attempt to determine the value of one attribute depending on the value of another. These tasks incorporate techniques like statistics, categorization, regression, and forecasting [19]. Meanwhile, the descriptive tasks are to generate patterns in the database in order to extract the underlying relationships [29]. Some of the patterns generated are anomalies, clusters, correlations, and trends. Clustering, summarizing, association rules, pattern identification, and sequence discovery are some of the approaches used in this task [22]. There are two types of itemset mining in the database: frequent itemset and infrequent itemset. From the records of the previous studies, in frequent itemset mining, there are three well-known algorithms: Apriori [5, 6, 32], FP-Growth [7], and ECLAT [8, 18]. This research focuses on the vertical format by looking deeper into the equivalence class transformation (ECLAT) algorithm [8]. Tidset, Diffset, Sortdiffset, and Postdiffset are four extension variations introduced in ECLAT. In 2018, Jusoh et al. [9-10] introduced the R-ECLAT algorithm, particularly useful for mining infrequent itemsets. This algorithm is based on the ECLAT algorithm for mining infrequent itemsets [11-13]. In previous research, the R-Diffset was developed and executed in sequential processing for infrequent item mining as an extension of the Diffset algorithm. However, the current execution of data processing often faces the constraint of slowness, especially when dealing with large datasets.

In 2022, Man et al. [19-20, 27, 43] introduced a new incremental of rare pattern mining approach using R-Eclat named as Incremental Rare Equivalence Class Transformation, (IR-Eclat). The IR-Eclat, which was created especially for mining rare patterns is advantageous for dynamic databases because the volume of data increase linearly with time. For dynamic databases that are subject to the addition or deletion of items or records of transactions in the database, the incremental approach is advantageous. To apply mined knowledge of previously and scanning of a basic incremented database, it is necessary to use implementation of incremental mining.

An incremental approach is a new approach that has been executed sequentially on the splitted data. This incremental approach will complement the IR-Diffset algorithm to ensure that it can mine the infrequent itemsets faster. This new approach is introduced as IR-Diffset, where IR represents incremental-rare. As part of the R-Diffset algorithm, this study introduces an incremental approach to deal with the processing time issue. To simplify, this research is significant in determining the following aspects: How to reduce speedy processing time in mining infrequent pattern especially in huge dataset? The rest of the components are then organized as

follows: Section IIA discusses sparse and dense data. The basics of mining rare items are covered in Section IIB. R-Diffset and IR-Diffset are described in Section III. Section IV clarifies the discussion of the study. The study's results and the discussion are also described in Section IV. The research was concluded in Section V.

II. Related Works

This section concerned on discussion regarding database, data mining and infrequent itemset mining. A database is known as a structured collection of data, dataset or record. It can be effortlessly rapid accessed, managed and updated data in conjunction with various data processing operations. Data mining is the process of extracting useful unknown knowledge from large datasets. However, mining infrequent (rare) itemsets may be more interesting in many real-life applications. However, this section also covers about sparse and dense data.

A. Sparse vs Dense Data

Sparse data is a matrix [14] in numerical analysis, where the majority of the elements are zero. This section will discuss about sparse data, which are quicker to be processed and figured, compared to dense data. Although the number of zero-valued elements are not specified for a sparse matrix, the number of non-zero elements are standardized to be equal to the number of rows or columns, due to their similarities. However, if most of the elements are non-zero, the matrix will be identified as dense. The sparsity of a matrix is calculated by dividing the zero value elements with those of non-zeros (e.g., p x q for a p x q matrix) [15]. The sparse matrix can be described in the following ways: array and linked list representation. In the array representation, the 2D array can be used to represent a sparse matrix in which there are three elements named as follows:

- Row: a row index containing a non-zero element.
- Column: a column index containing a non-zero element.
- Value: The non-zero element's value is placed at the index (row, column).

A sparse matrix is applied in linked list to portray the list of data structures. While there are three elements in the array representation (i.e., row, column, and value), the linked list representation on the other hand, comprises of four elements. The elements in the linked list are as follow:

- Row: a row index containing a non-zero element.
- Column: a column index containing a non-zero element.
- Value: The non-zero element's value is placed at the index (row, column).
- Next node: It records the next node's address.

When storing and processing sparse matrices on a computer, it is preferable to utilize specific algorithms and data structures that take advantage of the matrices' sparse structure. Sparse matrices, which are common in machine learning, have given rise to specialized computers. When applied to vast sparse matrices, typical dense-matrix structures and techniques are sluggish and ineffective, as computation and storage are squandered on the zeros.

Sparse data compresses more effectively and hence take up less storage space. Some outsize sparse matrices are unmanageable using typical dense-matrix techniques.

The matrix *sparsity* is like density of zero elements in the sparse matrix and is defined as the number of zero elements divided by total elements in the matrix, which computed as:

sparsity = number of zero elements/number of total elements

The sparsity of a sparse matrix is always greater than 0.5.

In a dense matrix, most of the elements are non-zero. The dense matrix can be stored in a fixed-size array.

The sparse matrix, on the other hand, recorded only non-zero elements in a dictionary mapping an index (i, j) to its entry. As a result, you should only describe a matrix as sparse if the number of non-zero elements is relatively tiny in comparison to the total number of entries. This is generally advantageous if the non-zero elements are large (there are about n non-zero entries compared to about n^2 zero entries). However, generating dense matrices is usually faster for extremely tiny matrices (e.g., 5 x 5), even with a large number of zero entries.

Fig. 1 depicts the characteristics of sparse and dense data. Dense data is the nature of the data itself, which is dense, which causes it to be slow. The data is dense and takes up a lot of space, and it is extremely difficult to process. In order to find other data using the dense data, it must first check each column one by one; without skipping a column. Sparse data indicates the nature of the data itself, which is sparse. Processing time for sparse data is faster compared to dense data because sparse data is not dense or compact; in the sparse data column, there is still an empty space, so processing the data will be faster and will not glitch in comparison to the dense data.

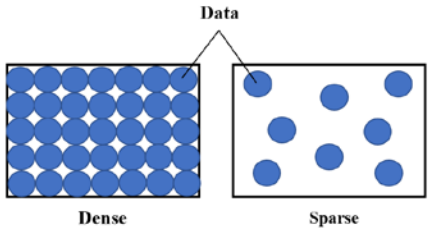

Fig. 1. Features of data-sparse and data-dense.

The theoretical background for sparse and dense data has been thoroughly discussed in this section. The next section will focus on one of the infrequent itemset mining.

B. Infrequent Item Set Mining

Up to this date, the mining of non-recurrent data have been considerably developed. Non-recurrent or infrequent patterns are usually utilized in many disciplines such as biology, health, and security. In medical sector, the analysis of clinical conditions of patients are carried out to detect irregular patterns

or trends. The results will help the medical officers in deciding the needed treatments or prescriptions.

Infrequent itemset mining from a database is not as prominent as frequent itemset mining [39-40]. However, there are few areas where infrequent item mining is more applicable than frequent item mining [16]. The objective of IIM is to discover unusual, but informative, relationships between entries in a dataset. In [16] the concept of itemset for frequent pattern mining, were established based the pattern mining concept. The set of items were denoted as $I = \{I_1, I_2, \ldots, I_m\}$. The itemset I will be accounted as frequent if and only if the occurrence frequency in the database equals or exceeds the minimum support threshold defined by the users [24-25]. In contrast, infrequent itemset mining requires that the frequency of occurrence in the database be equivalent to or lesser than the user-defined minimum support threshold. But, if the threshold is too low, infrequent itemset mining using the traditional frequent itemset mining method could be unsuccessful [33].

Infrequent itemset mining are receiving tremendous concern principally in the fields of networking and medicine. In networks, it is used to investigate any atypical incidences in any networks which are usually indicators of network breakdown or security breach. In medical, infrequent itemset mining may assist for the discovery of treatments for uncommon cases. The objective of infrequent itemset mining is to mine patterns with a support value smaller than the minimum threshold [34, 35]. Yet, the process of extracting rare patterns from the database, appears to be challenging.

In 2019, Man et al. [4, 28] proposed algorithm to address the discovery of infrequent itemsets mining from the transactional database based on Eclat algorithm. In support measure, IF-Diffset was proved to perform better upon encountering the infrequent itemsets of the transactional database.

An algorithm for extracting infrequent itemsets from weblog has been introduced by Bakariya et al. in 2019 [17]. Infrequent Itemset Mining for Weblog (IIMW) algorithm is a phased, top-down, broad-first algorithm for the recovery of infrequent item sets. A power set and lattice traversal approach had been utilized for this purpose. This approach followed the top-down mechanism, begun from top and carried out down to the bottom. By applying this approach, the computation of support for smaller itemsets were simpler than the larger itemset. This problem has been acknowledged as the challenges for the further application of this method.

To overcome the above-mentioned obstacles, Lu et al., later in 2020, [42] suggested an infrequent pattern mining algorithm using a top-down and depth-first traversing strategy. A negative itemset tree was applied to speed up the mining procedure, by compressing the datasets for a quicker counting process. Their itemset miner, denoted as NIIMiner had been verified to solve the problems of rare itemset mining.

In 2021, Darrab et al. [21] performed an extensive search of latest methods of rare itemset mining. It has been done by inspecting another efficient data structure with mining rare itemsets, the mining of most interesting rare itemsets, as well as the mining rare of itemsets with multiple minimum support thresholds.

In 2021, Abu bakar and his fellow researchers [31, 37] developed a performance enhancement in Incremental Eclat (iEclat) model by embedding Critical Relative Support (CRS) in the mining of infrequent itemset. This was done in line to the increment of itemsets which produced higher cardinality of intersection between each item hence required the method of vertical mining.

In 2022, Cui et al. [41] introduced a novel fuzzy-based rare itemset mining algorithm called FRI-Miner, and succesfully discovered valuable and interesting fuzzy rare itemsets in a quantitative database by applying fuzzy theory with linguistic meaning. The established algorithm was a success and has been shown to have an improved overall mining quality compared to the existing algorithm.

The previous paragraph of this section summarized the latest algorithm for infrequent itemset mining which are infrequent itemsets mining from the transactional database based on Eclat algorithm [4], Infrequent Itemset Mining for Weblog (IIMW) algorithm [17], Incremental Eclat (iEclat) model by embedding Critical Relative Support (CRS) in mining of infrequent itemset [31], Infrequent Pattern Mining Using Negative Itemset Tree (NIIMiner) algorithm [42] and Fuzzy-based Rare Itemset Mining (FRI-Miner) algorithm [41]. The next section will pay a focus on R-Diffset and IR-Diffset. The constraint of time, due to the large size of data, is among the main disadvantages often faced by the R-Diffset. For improvements, a new algorithm, known as incremental R-Diffset (IR-Diffset), which could help to fasten the process has been introduced.

III. R-DIFFSET VS. IR-DIFFSET

This section will explain the theoretical background of R-Diffset and IR-Diffset. The technique, formula, and illustration model for R-Diffset and IR-Diffset in infrequent itemsets mining will also be presented. Similar techniques, called depth-first search are used for both R-Diffset and IR-Diffset.

A. R-Diffset

In [4], the extension of the diffset algorithm named as R-Diffset, has been introduced. It is executed via sequential processing specifically for infrequent itemset mining. Each item in a vertical database is associated with its corresponding tidset, i.e., the set of all transactions (tids) where it occurs. Thus, the size of tidsets is the primary factor affecting the running time and memory usage. The bigger the tidsets, the more time and memory are needed. However, existing data processing execution is often confronts the issue of large database with single format only. Before this, the execution time data processing in R-Diffset algorithm it is very time-consuming especially in sequential processing [38].

The R-Diffset algorithm concern to keeps track the differences in the tids (transaction IDs) of a candidate pattern from its generating infrequent patterns. It helps to decrease the size of memory required to store intermediate results as depicted in the below pseudocode. The line 6 till 8 in Fig. 2 illustrate that this algorithm will generate all itemsets with their

diffsets and supports. Line number 7 shows where it gets the diffset data, instead of getting intersection of tidsets data between column i and (i + 1) based on minimum support that has been set at line 5. All infrequent itemsets are generated by computing diffsets for all distinct pairs of itemsets and checking the support of the resulting itemset. This process is repetitive until all infrequent itemsets have been enumerated. Since the generated diffsets are a small fraction size of tidsets, so the intersection operations are performed extremely fast. The performance of R-Diffset shows a good improvement in execution time over R-Tidset variant in both dense and sparse databases.

Pseudocode R-Diffset

Input:$E((i_1, t_1), \ldots (i_n, t_n)|P), s_{min}$
Output:$IF(E, s_{min})$

Begin *//get minimum support*
1. arrange data by itemset
2. looping = numOfColumns;
3. min_supp = number_of_rows * percentage_min_support
4. run tidset;
5. for (i=0; i<=min_support)
6. if (**support<=min_support**){
7. get diffset data for column [i] with column [i+1];
8. save to DB;}
9. Set next transaction data;
10. Write to text file the value for the current / last transaction data;}
end

Fig. 2. Pseudocode for R-Diffset.

B. IR-Diffset

This study presents a modified algorithm, which is an improvised version of existing R-Diffset, named as Incremental R-Diffset (IR-Diffset). This new approach serves as an alternative to speed up the processing time in infrequent itemsets mining, particularly those involving large datasets. IR-Diffset could be implemented with the data that has been sequentially splitted, incrementally. Incremental in itemsets refers to any additional new item which is being added or deleted to the existing itemsets in database whereas incremental in records of transaction refers to the additional transactions, added to the existing transaction. The IR-Diffset model is illustrated in Fig. 3.

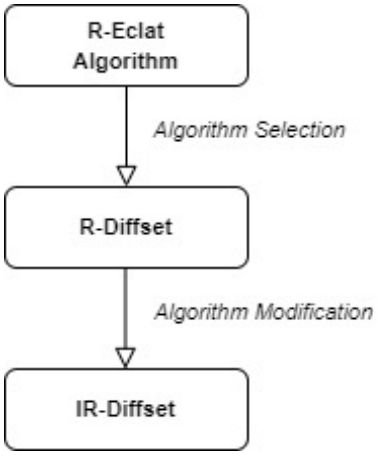

Fig. 3. IR-Diffset Model.

The incremental itemset will be added to the basic model of R-Diffset, establishing a new IR-Diffset model. The IR-Diffset algorithm will keep track the differences in the tids (transaction IDs) by counting each item for finding support which is less than or equal. As presented in Fig. 4, a slight modification of implemented code is at line 7 (while loop). The while loop functions to control flow statements by allowing itemsets to be executed repeatedly. For example, the itemsets in the loop will be performed repeatedly, as long as the variable *i* is less than looping. The variable '*r*' at line 4, refers to the record of transaction. When variable '*r*' is not equal to zero, the transactions will be run (i.e the condition is true) it will enter the looping. When variable 'r' is equals to zero, the process will be terminated. Then, line 8 refers to (i = 0; i < looping; i++), which is i = 0 means that the variable i is = to 0, i < looping means that the process will be run repeatedly as long as i is smaller than zero and i++ means that each time the looping process take place, the new I will be I = i+1. This incremental approach has been proposed to overcome the limitations of sequential processing in extracting infrequent itemsets. Then, the following are the key steps in IR-Diffset over the dataset:

Support counting is for finding the support through determination of supports of any k-itemsets on the intersecting tid-lists of its k-1 subsets. First, the basic theory for support counting must be less than or equal to MSTV. Second, the result will be written and saved in the text file prior to preparation for the next row of transaction data. The minimum support threshold value (MSTV) is considered as a benchmark to discover a low occurrence in each dataset. MSTV is determined in terms of percentage [23].

$$\frac{\alpha}{100} * \beta \quad (1)$$

Where, α = User specified minimum support value
β = Total of records in datasets.

Pseudocode IR-Diffset

Input:$E((i_1, t_1), \ldots (i_n, t_n)|P), s_{min}$
Output:$IF(E, s_{min})$

start *//get minimum support*
sort data by itemset
looping = num_of_column
r = record_of_transactions
min_supp = num_of_row * percentage_min_supp
run tidset
while r ≠ 0 do
for (i=0; i<looping; i++)
if(support<=min_support)
get diffset data for column [i] with column [i+1];
add next transaction data;
write to text file the value for the current / last transaction data;
end

Fig. 4. Pseudocode for IR-Diffset.

The step-by-step process of IR-Diffset is presented in the above pseudocode. Other process steps in each of these variants are preserved as the original variant. The first dissimilarity is the basic theory for support counting where it will only consider the support that is less than or equal to MSTV. The second difference is the storage of results. The

value/result is written and saved in the text file prior to preparation for the next row of transaction data. This is for the purpose of saving memory space. In each loop, starting with the first loop, if the support is less than or equal (<=) to min_supp, then,

1) Rather than utilizing intersection, IR-Diffset obtains the result of Diffset (difference intersection set) between the k^2 and $k^{th}+1$ columns and stores it in the database.

Table I shows the effectiveness of the comparative analysis and that clearly shows the technique, data format, execution time, process, and execution between R-Diffset and IR-Diffset algorithms with infrequent itemsets mining. R-Diffset and IR-Diffset use the same technique, depth-first search.

TABLE I. FEATURES DIFFERENCES BETWEEN R-DIFFSET AND IR-DIFFSET

	R-Diffset	IR-Diffset
Technique	Depth first search	
Data format	Vertical	
Time	Faster intersection process	Less fast for intersection process
Process	Keeps track of differences in tidsets	-
Execution	Executes the infrequent itemsets mining process in sequential form	Executes sequentially on the data that has been split

Table I summarizes the differences of R-Diffset and IR-Diffset. The details of step-by-step processes for both R-Diffset and IR-Diffset are previously presented in Fig. 2 and Fig. 4, respectively. Line 6 to 8 in Fig. 2 illustrate that all itemsets will be generated by this algorithm together with their diffsets and supports. This algorithm is similar to those of IR-Diffset as portrayed in Fig. 4 (lines 9 to 11). The differences between these two pseudocodes can be seen in line 7 (while $r \neq 0$) is used inside the pseudocode of the IR-Diffset. The term "min_supp" is used in the pseudocode to refer to the minimum support threshold value, which is expressed as a percentage and is calculated by dividing the user-specified min_supp value by 100 and multiplying it by the total number of rows (records) in each dataset. Then, starting with the first loop, if the support is less than or equal (<=) to min_supp, then line number 10 shows where it gets the diffset data, instead of getting intersection of tidsets data between column i and $(i + 1)$ based on minimum support that has been set at line 8 and saved to DB. The benefit of incremental storage structure lies in the fact that every candidate of itemsets in search space has the same itemset in database. Due to this, its support can be computed by a few simple database operations, removing the need of full scan of database. The rest of pseudocode lines for R-Diffset and IR-Diffset are same.

IV. DISCUSSION

All experimentations are carried out on an HP Notepad with an Intel ® Core TM ® i7-3520M CPU running at 2.90 GHz and 8GB of RAM running Windows 10 64-bit. Open-source software is used to implement the algorithm development software standard. For our database server, we use MySQL (version 5.6.25 - MySQL community server (GPL)), Apache/2.4.16 (Win32) OpenSSL/1.0.1i PHP/5.6.11, PHP as a programming language, and phpMyAdmin (version 4.8.4) to manage MySQL via the Web are all included in the package. Four datasets including chess, pumsb_star, retails, and T40I10D100K are used in this experimentation as described in Table II. The benchmark datasets were collected in their raw form from the Frequent Itemset Mining Dataset Repository (http://fimi.ua.ac.be/data/). The benchmark is also converted into Structured Query Language (SQL) format to make things easier.

TABLE II. DATASET ATTRIBUTES

Datasets	No. of Transactions	Length (Attributes)	Size (KB)	Category
Chess	3196	37	335	Dense
Pumb_star	49046	57	11526	Dense
Retails	88162	68	5143	Sparse
T40I10D100K	100001	32	15116	Sparse

In order to facilitate and accelerate research, the datasets are limited to a thousand rows of randomly processed item sets for mining purposes. Our research is focused on R-Diffset and IR-Diffset. Fig. 5 depicts the performance assessment graph in terms of execution time (in seconds) for the four (4) datasets: chess, pumsb_star, retails and T40I10D100K.

The experiments on R-Diffset and IR-Diffset is successfully tested to determine the comparative performance between sparse and dense data. Fig. 5 summarizes that the execution time R-Diffset in dense data is 0.0001 slightly faster than IR-Diffset. But, in sparse date, the result shows that the execution time of IR-Diffset is 0.0003 faster over R-Diffset.

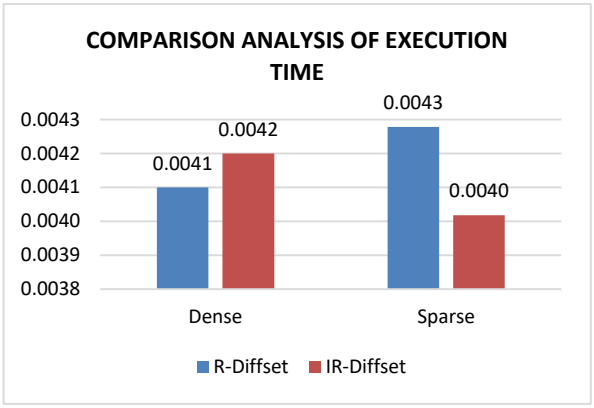

Fig. 5. Performance on R-Diffset and IR-Diffset for chess, pumsb_star, retails, and T40I10D100K

The difference of execution time for both algorithms may be caused by the property's nature of the data itself. Dense dataset is naturally looks like a compressed data where the space between the data is very small. Due to this characteristic, the process of mining infrequent itemset via incremental approach become slower when it needs to sequentially mining the infrequent itemset in several fractions. Finding every single of itemset in the dense data is incredibly challenging due to its dense characteristic. Nevertheless, it is very contrary compared

to the nature feature in sparse dataset. In sparse dataset, the feature between the data itself is sporadic and the space between each data is wide. Due to the wide space between each data, the discovery of the infrequent itemsets is a bit faster and less challenging.

V. CONCLUSION

In this research, the performance comparison between R-Diffset and IR-Diffset has been analysed. The IR-Diffset have a better performance for infrequent itemset mining in sparse dataset although it slightly loses its advantages over R-Diffset in dense dataset. The incremental approach is introduced to overcome the limitations of sequential processing in extracting infrequent itemsets. The complementary of this approach is believed can be a solution of speedy processing time in mining infrequent pattern especially in huge dataset. Looking at the improvement performance of IR-Diffset in sparse dataset, it can be applied to the real dataset from various fields, such as finding a rare disease at certain areas. It might help on how to control and manage this disease from dispersion which might causes the increase number of patients.

ACKNOWLEDGMENT

We would like to express our gratitude to the CREIM of UniSZA for providing financial support under UniSZA internal grant code (UniSZA/2021/GOT/02). Thanks to the corresponding authors, Dr. Julaily Aida Jusoh as the CREIM-UniSZA research project leader, Sharifah Zulaikha Tengku Hassan as the research assistant, and the grant participants. We would also like to thank all of the faculty members who helped us evaluate spelling problems and synchronization consistency and for their thoughtful remarks and recommendations.

REFERENCES

[1] Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996, August). Knowledge Discovery and Data Mining: Towards a Unifying Framework. In *KDD* (Vol. 96, pp. 82-88).

[2] Gullo, F. (2015). From patterns in data to knowledge discovery: What data mining can do. *Physics Procedia*, *62*, 18-22.

[3] Tan, P. N., Steinbach, M., & Kumar, V. (2006). Introduction to data mining, addison wesley publishers.

[4] M. Man, J. A. Jusoh, S. I. A. Saany, W. A. W. A. Bakar, and M. H. Ibrahim, "Analysis Study on R-ECLAT Algorithm in Infrequent Itemsets Mining", International Journal of Electrical and Computer Engineering (IJECE), pp. 5446-5453, 2019.

[5] RAgrawal R, Imieliński T, Swami A., "Mining association rules between sets of items in large databases". Proceedings Acm sigmod record, vol. 22, no. 2, pp. 207-216, 1993.

[6] Agrawal, R., & Srikant, R. (1994, September). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB (Vol. 1215, pp. 487-499).

[7] Han J, Pei J, Yin Y. "Mining frequent patterns without candidate generation," Proceedings ACM sigmod record, vol. 29, No. 2, pp. 1-12, 2000.

[8] Ogihara Z. P., Zaki M. J., Parthasarathy S., Ogihara M., Li W., "New algorithms for fast discovery of association rules," 3 rd Intl. Conf. on Knowledge Discovery and Data Mining, 1997.

[9] Jusoh, J. A., Man, M., & Bakar, W. (2018). Mining infrequent patterns using R-ECLAT algorithms. *J Fundam Appl Sci*, *24*.

[10] Jusoh, J. A., & Man, M. (2018). Modifying iEclat Algorithm for Infrequent Patterns Mining. *Advanced Science Letters*, *24*(3), 1876-1880.

[11] Zaki, M. J., Parthasarathy, S., Ogihara, M., & Li, W. (1997, August). New algorithms for fast discovery of association rules. In *KDD* (Vol. 97, pp. 283-286).

[12] Zaki, M. J., & Gouda, K. (2003, August). Fast vertical mining using diffsets. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 326-335).

[13] Trieu, T. A., & Kunieda, Y. (2012, February). An improvement for dECLAT algorithm. In Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication (pp. 1-6).

[14] El Abbadi, N. K., & Al Taee, E. J. (2014). An Efficient Storage Format for Large Sparse Matrices based on Quadtree. International Journal of Computer Applications, 105(13).

[15] Bik, A. J., & Wijshoff, H. A. (1993, August). Compilation techniques for sparse matrix computations. In Proceedings of the 7th international conference on Supercomputing (pp. 416-424).

[16] Borah, A., & Nath. "Rare Pattern Mining: Challenge and Future Perspectives." Springer International Publishing. (2018).

[17] Bakariya, B., Thakur, G. S., & Chaturvedi, K. (2019). An efficient algorithm for extracting infrequent itemsets from weblog. *Int. Arab J. Inf. Technol.*, *16*(2), 275-280.

[18] M. J. Zaki and K. Gouda, "Fast vertical mining using diffsets," Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 326–335, 2003, doi: 10.1145/956750.956788.

[19] Man, M., Ruslan, N. A. B., Bakar, W., Jusoh, J. A., Yusof, M. K., & Josdi, N. L. N. B. (2022). IR-ECLAT: A New Algorithm for Infrequent Itemset Mining. *Journal of Theoretical and Applied Information Technology*, *100*(11).

[20] Man, M., Ruslan, N. A., Jusoh, J. A., & Bakar, W. A. W. A. (2020). Conceptual Model of Incremental R-Eclat Algorithm for Infrequent Itemset Mining. *International Journal Of Engineering Trends And Technology (IJETT)*, *10*, 129-133.

[21] Darrab, S., Broneske, D., & Saake, G. (2021). Modern applications and challenges for rare itemset mining. *Int J Mach Learn Comput*, *11*(3), 208-218.

[22] Fournier-Viger, P., Lin, J. C. W., Vo, B., Chi, T. T., Zhang, J., & Le, H. B. (2017). A survey of itemset mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *7*(4), e1207.

[23] W. A. B. W. A. Bakar, et al., "Incremental-Eclat Model: An Implementation via Benchmark Case Study," Springer International Publishing Switzerland, P.J. Soh et al. (eds.), Advances in Machine Learning and Signal Processing, Lecture Notes in Electrical Engineering, vol. 387, pp. 35-46, 2016.

[24] Vu, L., & Alaghband, G. (2014, January). An efficient approach for mining association rules from sparse and dense databases. In *2014 World Congress on Computer Applications and Information Systems (WCCAIS)* (pp. 1-8). IEEE.

[25] Darrab, S., & Ergenc, B. (2017). Vertical Pattern Mining Algorithm for Multiple Support Thresholds. Procedia Computer Science. 112. 417-426. Elsevier.

[26] M. K. Yusof and M. Man, "Efficiency of JSON for Data Retrieval in Big Data," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 7, no. 1, pp. 250–262, Jul. 2017, doi: 10.11591/IJEECS.V7.I1.PP250-262.

[27] Man, M., Bakar, W. A. W. A., Jalil, M. M. A., & Jusoh, J. A. (2018). Postdiffset Algorithm in Rare Pattern: An Implementation via Benchmark Case Study. *International Journal of Electrical and Computer Engineering (IJECE)*, *8*, 4477-4485.

[28] Bakar, W. A. W. A., Jalil, M. A., Man, M., Abdullah, Z., & Mohd, F. (2018). Postdiffset: an Eclat-like algorithm for frequent itemset mining. *International Journal of Engineering & Technology*, *7*(2.28), 197-199.

[29] Jusoh, J. A., Man, M., & Bakar, W. A. W. A. (2018). Performance of IF-Postdiffset and R-Eclat Variants in Large Dataset. *International Journal of Engineering & Technology*, *7*(4.1), 134-137.

[30] Agrawal, R., & Srikant, R. (1994, September). Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB (Vol. 1215, pp. 487-499).

[31] Abu Bakar, W. A. W., Man, M., Abdullah, Z., & Man, M. B. (2021). CRS-iEclat: Implementation of Critical Relative Support in iEclat Model

for Rare Pattern Mining. *International Journal of Advanced Computer Science and Applications*.

[32] Rahman, A., Ezeife, C. I., & Aggarwal, A. K. (2012). Wifi miner: An online apriori-infrequent based wireless intrusion detection system. *Knowledge Discovery from Sensor Data (Sensor-KDD 2008)*, 76.

[33] D. J. Haglin and A. M. Manning, "On Minimal Infrequent Itemset Mining," Proceedings of the International Conference on Data Mining, DMIN'07, CSREA Press, pp. 141-147, 2007.

[34] A. Gupta, et al., "Minimally Infrequent Itemset Mining Using Pattern-Growth Paradigm and Residual Trees," CoRR abs/1207.4958, 2012.

[35] S.Tsang, et al., "RP-tree: Rare Pattern Tree Mining," DaWaK, Lecture Notes in Computer Science, Alfredo Cuzzocrea and Umeshwar Dayal (Eds.), Springer, Berlin, vol. 6862, pp. 277-288, 2011.

[36] A. Aziz, N. H. Ismail, F. Ahmad, Z. Abidin, K. G. Badak, and M. Candidate, "MINING STUDENTS' ACADEMIC PERFORMANCE," undefined, 2013.

[37] Bakar, W. A. W. A., Man, M., Man, M., & Abdullah, Z. (2020). I-Eclat: Performance enhancement of Eclat via incremental approach in frequent itemset mining. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, *18*(1), 562-570.

[38] Fournier-Viger, P., Lin, J. C. W., Kiran, R. U., Koh, Y. S., & Thomas, R. (2017). A survey of sequential pattern mining. *Data Science and Pattern Recognition*, *1*(1), 54-77.

[39] Shrivastava, S., & Johari, P. K. (2016, May). Analysis on high utility infrequent ItemSets mining over transactional database. In *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)* (pp. 897-902). IEEE.

[40] Keste, P. A., & Shaikh, N. F. (2016, March). Improved approach for infrequent weighted itemsets in data mining. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)* (pp. 2663-2667). IEEE.

[41] Cui, Y., Gan, W., Lin, H., & Zheng, W. (2022). FRI-miner: fuzzy rare itemset mining. *Applied Intelligence*, *52*(3), 3387-3402.

[42] Lu, Y., Richter, F., & Seidl, T. (2020). Efficient infrequent pattern mining using negative itemset tree. In *Complex Pattern Mining* (pp. 1-16). Springer, Cham.

[43] Man, M., Julaily, A. J., Saany, S. I. A., Bakar, W. A. W. A., & Ibrahim, M. H. (2019). Analysis study on R-Eclat algorithm in infrequent itemsets mining. *International Journal of Electrical and Computer Engineering*, *9*(6), 5446.