Journal of Theoretical and Applied Information Technology
15th December 2022. Vol.100. No 23

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

IS COMPUTATIONAL ALGORITHM FOR HIGH MINING ITEMSETS EFFECTIVE?

[1]SIVA S, [2]SHILPA CHAUDHARI, [3]S. PRAVEEN KUMAR

[1]School of Computing and Information Technology, REVA University, Bangalore, India

[2]Department of Computer Science and Engg., M S Ramaiah Institute of Technology, Bangalore, India

Post-doc Research assistant, University of Trento, Italy

Email: [1]sivaraju.reva@gmail.com, [2]shilpasc29@gmail.com, [3]pk.sekharamantry@unitn.it

ABSTRACT

Every day the number of datasets is increasing, and the advancement also leads to different research algorithms to extract meaningful information about personal interests. The problem of frequent itemsets from analysing different periodical itemsets from the large chunks of data has always been a keen research area for data analysts. Therefore, the most valuable fields of data mining are computational high-utility itemset mining i.e., used to obtained valuable knowledge from a homogeneous mixture of data. In the past years, various algorithms and works are proposed by different researchers to solve the issues of high-utility or frequent itemsets mining in transactional databases; however, an essential limitation of the algorithms is static nature [14]. In real-life applications, datasets are often dynamic and change according to different parameters such as price, weight, quality etc. Therefore, a transformation of the existing database and redefinition of parameters is required for gaining maximum profits from the high-utility itemsets mining. In this research, a novel approach is designed and developed i.e., a computational intelligence approach based on high-utility itemset mining algorithm for efficient analysis of periodical itemsets from the transactional database [19]. The proposed MFHUIM algorithm is further experimented on retail BMS database using Java-based environment and results of both the algorithms based on runtime, memory usage and scalability are detailed in the research work. MFHUIM outperforms EFIM algorithm in extraction of the accurate high utility itemsets in a large set of databases.

Keywords: *Periodical Itemsets, High-Utility Itemsets, Transactional Database, Homogeneous Dataset, Pattern Mapping*

1. INTRODUCTION

The problem of extracting meaningful or valuable information from the large chunks of transactional databases is a typical task; therefore, big data mining technology comes into action for identifying useful information for the users. The research is focused on one particular problem of data mining techniques such as Frequent Itemset Mining (FIM). The problem was first introduced by [1] for the purpose of analysing the customer transactional information in a retail store such that sales can be fostered for the store. The prime objective of most of FIM algorithms is to extract the various products or items that are most purchased, so generally, users rely on the downward closure property that states all irregular items are subsets of multiple items and all frequents items are a superset of all the items. Logically, the property holds meaning in real-world as support or in general terms frequency of items is an anti-monotonic quantity which means it could be equal to less than that of the superset value of the itemsets. All in all, the property of downward closure is a useful technique, as it could aid in reducing the search space for FIM. For instance, if there are total n different items in a dataset, then every iteration of downward closure will reduce the search space by 2(n-k) – 2 supersets of datasets.
However, this technique is easy to understand and inspired many Patten mining techniques such as Apriori, FP-growth etc. but it suffers from a significant drawback, as it only measures the frequency of itemsets in a database and ignores the other essential parameters such as profit or quantity of sales. Therefore, considering other metrics are equally important, as sometimes FIM algorithms are prominent for many items frequency but might result in low yield profit for the users. Furthermore, rare item problem in association rule technique is

another major drawback of FIM algorithms that it considers the utilities to be equal to that of itemsets. The utilities of the itemset could vary, yet in many applications, some items repeat very frequently than others. Hence, it will lead the mining analysis to possibly two results such as: (1) if the minimum support is considered to be very high value, then the rare items in the dataset will be ignored or (2) To find the mixed itemset, minimum support will be considered very low, which will raise the issues of combinational explosion. In many real-world applications, the rare combinations usually turn out to be more profitable for the HUIM results as well as provide better insights to the users.

Thus, in order to find the most useful and profitable itemset from a large chunk of data, a High-Utility Itemset Mining (HUIM) technique is being introduced, which could be assumed as generalisation class of FIM technique. Unlike the FIM technique, HUIM focuses on multiple parameters that are associated with frequent itemsets such as unit sold or profit from each item etc. For instance, if any transactional database contains give different attributes for a transaction such as transaction ID, price, quantity or itemsets, then it can be possible that most frequent itemsets don't turn out to be most the profitable ones using FIM algorithms. Nonetheless, on the other hand, it could be the possibility that using HUIM algorithms will aid the users to find the most optimal datasets and items that could be more frequent as well as the most profitable. The frequency itemset is an itemset, which has maximum frequency support that should be greater than that of a minimum threshold value.

2. RELATED WORKS

Various data mining techniques and association rules concepts are designed and developed by multiple researchers over the period of time for extracting meaningful information from a large number of datasets. In this section, a brief overview of the concepts and techniques given in different research papers are evaluated.

According to Wu et al. [2], mining and data association rules are standard types of data mining techniques such as Apriori algorithm [22]. The method of Apriori algorithm basically works on two standard parameters such as minimum support and maximum confidence that involves generating association rules for extracting frequent itemsets mining. Similarly, FP-growth algorithm is also used to find out the frequent itemsets from large transactional datasets, but it only includes the confidence parameter for evaluating recurring patterns based on the FP-tree based analysis or diagrammatic association rules [16].

M. Zihayat et al. [3], explains the limitations of the traditional data mining algorithms, as dynamic transactional datasets and multi-valued information are generally ignored while considering the frequent itemsets, however, rare itemsets values can also lead to providing optimal results for the high-utility frequent itemsets. The author suggested that minimum utility mining can be stated as a mathematical model for finding minimum utility bound and support properties for extracting the high-utility frequent itemsets. Therefore, Zihayat, Kargar & Szlichta, explained only about the theoretical model that laid the foundation for understanding the importance of utility mining algorithms.

Fournier-Viger, Gu & Tseng [4], formulated a semantic concept for the semantic-based analysis of the utility-based measures to classify the most frequent patterns from the transactional database. The utility function defined in the research is defined as a general unified framework for the purpose of unifying view of itemset based on frequent itemset mining [18]. The experimental results reflect the mathematical properties of frequent utility-based measures for understanding the significance of predefined utility functions. The performance metrics and frequency from the semantic transaction shifting technique contribute to analysing real-world applications. For example, in order to generate effective marketing strategy high-utility frequent itemsets mining satisfy the significant objectives of the company's bottom line requirements.

Another algorithm supported by Su et al. [5] explains about high-utility item combination and association rules development based on the demographic mathematical standards and frequent items mining to find segments of data defined for combinational of frequent items. The research suggests that high-utility frequent, the itemset mining problem is comparatively different from trivial issues, as it is based on the prior rule discovery and then finding individual attributes as well as predefined objective functions. However, predefining the objective functions. The overall objective behind the development of predefined function is to find such groups or patterns in the given item sets that could ease in ensuring pre-classified information of frequent itemsets mining.

In another article, Bermingham & Lee [6], illustrate two adequate datasets representation

techniques that could ease in development of transaction sensitive sliding window technique for analysing all the necessary information related to the structural architecture of tree-based architecture. Thus, the results of the MHUI-TID or temporal high-utility frequency itemsets mining reflects the transactional downward closure property for generating fewer temporal high utility itemsets in generating data stream in an effective and efficient manner [11]. Another algorithm proposed in the research divides the frequent itemsets mining process into two phases for efficient and effective temporal datasets generation that could lead to optimizing the final search space of the data streams. Nonetheless, the idea of comparing both the upper bounds or predefined functions are based on the heuristics data mining process but consume a lot of memory space and time complexity to prune out the most feasible solution. Hence, it could fail to extract the complete high-utility itemsets as these algorithms are based on the heuristic parameters only.

In order to find out all the relevant frequency itemsets from the transactional database, Reddy et al. [7] proposed a novel approach called transaction weighted utility that could satisfy all the significant properties of downward closure as well as safely reduces the search space. The researcher introduced a two-phased approach which can combine Transaction weighted utility with Apriori algorithm based on the high utility patterns or frequent itemset candidates [23]. The algorithm suggests that items having transactional weighted utility values are not less than that of the minimum utility threshold value. Although, the two-phase approach is valuable for extracting transactional database evaluating results is based on the computationally expensive problem for further frequent utility itemsets [17]. Another strategy based on tree models such as MIQ-tree (Maximum quantity item tree), local UP-tree (local utility path tree) and SIQ-tree (Sum of items quantities tree) is helpful in speeding up the mining process for findings frequent utility itemsets by considering the co-occurrences of the items in the datasets. Another study extending to the new tree-based algorithm is based on the time decaying model that could record the time required to mine the frequent patterns in the computational model. Zhang, Zhang, Niu & Qiu [8] proposed an incremental mining algorithm for effectively mining high-utility itemsets in dynamic transactional databases that are generally used in real-world applications [21]. Furthermore, an indexed list-based data structure is prepared that could filter the low weighted or utilised itemsets and optimise the mining pattern efficiently without storing extra information on transaction item datasets records.

Luna, Fournier-Viger & Ventura [9] propose a framework to damped window model to evaluate without candidate patterns to generate a tree based algorithm to mine recent used item package from the database. However, the previous data mining algorithms at least need to scan for two major sets of data items to find the feasible items that might be used high-utility itemsets, but in time scan damped window model a single indexed list-based data structure is developed that could reduce the scan time as well as memory utilisation of the process. One major trivial problem of dynamic transactional datasets is still not resolved in the proposed solution. When considering the weighted edges of the prefix tree structure are less efficient to identify transactional itemsets mining.

As stated above, none of the identified algorithms or concept is viable for yielding correct results on profits of itemsets that are dynamic in nature for high-utility itemset mining. Hence, applying these algorithms in transactional databases of the modern age will lead to yield incorrect results, or these algorithms can ignore the necessary itemsets that could contribute to high-utility itemsets mining.

3. PROBLEM DEFINITION

In this section, different problems and challenges of high-utility frequent itemsets mining are discussed and all the relevant explanation and profitable itemsets are detailed [10]. The High-Utility itemsets in transactional databases where the unit-profits are dynamic in nature and definition of related itemsets and frequent itemset mining.

Let suppose, the transactional database can be denoted as D and l be the set of all the items available in the transactional database. Every element i is a subset of l and goals to discover all the profitable components involved in high-utility itemsets. The utility of itemsets in the transactional database for high-utility is defined as follows:

3.1 Definition 1:

Consider an item x available in the transactional database T. The purchase quantity of x in T can be represented as function q(x, T) which implies the quantity of item x in the transactional database entry. Moreover, the unit profit of the itemsets in the database is represented as p(i) that provide the amount of profit generated in each sale unit item. Hence, the utility item set can be evaluated as given below as follow:

$$u(i) = q(i, T) \times p(i)$$

3.2 Definition 2:

Given a user defined datasets based on minimum utility threshold value such minutil that could discover high-utility itemsets threshold value. Thus,

$$u(x) \geq \text{minutil}$$

Where it states that itemset x is greater than or equal to the minimum threshold value that could significantly explain dynamic profit functions involved in the traditional database schema [25].

Nevertheless, the traditional database has a vital database technique that it could not handle the changes profit over time. The traditional high-utility itemset mining is inapplicable to find the dynamic profits changes which lead to ignore significant values in high-utility itemsets and extract the inaccurate results. Therefore, in order to address the issues of high-utility itemsets mining, a novel problem statement is defined in the research. Furthermore, redefine values and utility measures are precisely below:

3.3 Definition 3:

(Standard deviation of periods)

The term standard deviation can be referred to as the average period of the pattern repeating in an itemset l can be represented as function avgp(x, l) and can be calculated as the sum of all the possible periods of patterns divided k+1 elements, where k is the superset of x element in l itemset [24]. Hence, the standard formula for the standard deviation of periods of patterns is given as:

Standard deviation is given as

$$\boldsymbol{stanDev(X, s)} = \sqrt{\frac{\sum_{i=1}^{k+1}(per_i + avgPr(X,s))^2}{k+1}}$$

In the given sequence formula, l can form conductive support itemsets that could ensure at least one item exists from the X dataset and other elements will be included in the disjunctive support of the given itemsets l.

3.4 Definition 5:

The novel definition of the weighted profit of items in the real-life transactional datasets using p(i, T) that could aid in redefining the above equation as

$$U(i, T) = qu(I, T)\, x\, p(i, T)$$

The main objective of the proposed solution is to mine the high-utility itemsets from transactional databases that can be used for high-utility itemsets mining. Further, proposing an effective high-utility itemsets mining that can aid in converting two different types of database models in a compact format and equivalent expanded format of u(i, T) utility function of each item in every transaction T. The contact database representation of the proposed utility measures can be designed and developed on the basis of high-utility itemset mining algorithms for handling dynamic parameters from profit databases. Several other parameters included in the upper bounds of frequent itemsets mining in a transactional database. The weighted transactional database is considered to be an anti-monotonic in nature, which make it an upper bound element and thus, reduce the search space in an effective manner.

4. PROPOSED MODEL

In this section, a proposed algorithm named as MEFIM, i.e., Modified Frequent High Utility Itemsets Mining algorithm is explained. The proposed algorithm can outperform the limitations of the traditional FIM algorithms and could enhance the computational performance capabilities. The pseudo-code of MEFIM algorithm is detailed below as input transaction dataset contains a predefined minimum utility threshold and generate the output of all the relevant high-utility itemsets with respect to the threshold value for search item.

4.1 Algorithm: MFHUIM Algorithm

Input	D can be considered as transactional database and alphaD be an itemset, primary item is alpha. Furthermore, mutil be a predefined minimum threshold value for an itemset to be frequent itemset.
Procedure	Start • Arrange all the datasets according to extensions of alpha itemsets in the transactional database. • Alpha = null; • Scan transactional database D and evaluate the value of u(i, T) where all i belongs to I itemsets from database D and T belongs to D. Along with that, a secondary list lu(alpha, i) is also generated for further comparison. • Secondary (alpha) ={ i| i belongs to I intersection

	lu(alpha, i)≥mutil} • Sort all the transaction of secondary (alpha). In an ascending order from the total order received in lu(u, i). • Start scan database D, and start eliminating the elements having lower mutil value and read backwards on itemset scan of the transactional database D. • Evaluate the sub-tree utility set su(i, alpha) with respect to secondary list D using utility bin array. • Primary list(alpha) = {i| i belongs to secondary (beta) intersect su(alpha, i) ≥mutil} • Function call= Search (alpha, primary(alpha), secondary (alpha), mutil) End
Output	The set high-utility items will be projected after the completion of the algorithm.

4.2 Search Procedure

Input	Alpha be the itemset belongs to transactional database D as well as function calls included four parameters such as alpha, primary(alpha) index, secondary (alpha) index and minimum utility threshold.
Procedure	Start • For loop: i to primary (alpha) • Secondary(alpha)= alpha U (itemsets); • Scan, Secondary alpha itemsets to calculate u(beta) and create a secondary dataset records. • Cond: u(beta) ≥ mutil then proceed to output beta • Else: • Calculate secondary itemset and utility list for all the items, including i itemset in dataset scanning should be done. Therefore, a two array sorted utility list will be prepared. • Primary list(beta) = {z belongs to secondary list(alpha) or sublist (beta, z) ≥mutil} • Secondary list(beta) = {z belongs to secondary list(alpha) or lu (beta, z) ≥mutil} • Iterative function call using Search (beta, primary (beta), Secondary (beta), mutil); End
Output	Return the high utility frequent itemset value to the primary function.

4.3 Exemplification

In the following paragraph step wise explanation of the step used to find the frequent itemsets in transactional database D in table [1] is detailed.

Pre-condition: It is assumed that the minimum utility of transactional database D is considered to be 30.

Step 1	The first and foremost step is to initialise variable alpha to the null value.
Step 2	With respect to every value of i present in the itemset lu(alpha, i) corresponding secondary itemset is created, as shown in table [2].
Step 3	The generated itemsets in the secondary list (alpha, i) is arranged in the ascending order as shown in table [3] for the purpose of managing the equivalent order for the transactional weight utility itemset.
Step 4	Therefore, after getting the sorted list of all the itemsets present in the transactional database D, an mutil elements are removed for reducing the search and optimising the high utility itemsets.
Step 5	Using the read backward strategy transaction merging option is used for calculating su(alpha, i) for every value I belongs to transactional database D, until alpha = null.
Step 6	Therefore, after obtaining values from the previous step, primary (alpha) ={e, c, d} from table [4].
Step 7	As a final point, a recursive procedure of Search is called for all the itemset in transactional database D. And the initial call is made using parameters as given below: Search(alpha= null; Primary (alpha) = { e, c, d}, Secondary (alpha) = {a, b, c, d, e}, and mutil= 30}

Therefore, after the complete iteration, the value of beta is returned to {c} and {d}. Moreover, the

utility value of {c} and {d} in the secondary list is equal to 8 and 20 respectively. No others high-utility itemsets are evaluated to fulfil the condition of minimum utility i.e. 30. Hence, transaction item d have is considered to be high-utility frequent itemset i.e. appears almost every transaction that is being made or present in transactional database D.

Table 1. Transactional dataset

Transaction	Itemsets	Quantity	Profit
Tid1	a, b, c, d, e	1, 2, 3, 1, 1	2, 1, 5, 1, 5
Tid2	b, c, g, d	2, 1, 1, 1	2, 1, 3, 2
Tid3	g, c, d	3, 2, 2	1, 2, 4
Tid4	a, f, b	3, 1, 1	3, 2, 1
Tid5	a, b, c, d	3, 4, 1, 2	1, 2, 1, 3

Table 2. Calculating the value of lu(alpha, i),

Item	a	b	c	d	e	f	g
lu(alpha, i)	75	72	60	63	40	20	11

Eliminating the unnecessary items from the transactional database and arranging the transaction in an ascending order based on the lu(alpha, i) value, as shown below in the table 3.

Table 3. Re-arranging transaction based on lu(alpha, i)

Transaction	Itemset	Utilities
Tid3	g, c, d	3, 2, 2
Tid4	a, f, b	3, 1, 1
Tid2	b, c, g, d	2, 1, 1, 1
Tid5	a, b, c, d	3, 4, 1, 2
Tid1	a, b, c, d, e	1, 2, 3, 1, 1

The value of lu(beta, i) and su(beta, i) for the initial value of beta= {e}.

Table 4. value of lu(beta, i) and su(beta, i)

Item	c	d	b	a
lu(beta, i)	17	41	28	31
su(beta, i)	17	44	27	29

Hence, in the considered example number of transactions and itemsets are only 5, and only 7 items are available in the transactional database D. Nonetheless, search procedure followed in the above example is a costly operation that could be improved in the proposed modified computational high utility itemset mining algorithm.

5. MODIFIED COMPUTATIONAL HUIM ALGORITHM

In this section, an advanced p-set structured algorithm that could manage the dynamic profit changes made in transactional database D. An improved version algorithm, MFHUIM, i.e. modified frequent high utility itemset mining technique is proposed in the research for easing the database scans as well as calculate the local utility, sub-local utility and utility of each item present in the transactional database. In the proposed algorithm, for every item X in the database an equivalent secondary (X) is evaluated that could be further extended to X set. However, these operations of scanning and calculating the value is a typical task and memory consuming process. Therefore, a novel p-set structured i.e. projected structure itemset approach is proposed for expanding and easing the calculation of high frequent itemset from the broad set of databases [20].

Definition:

Temporary item dataset TID projection: p-set is called as the temporary projection of item dataset when all the set of identifiers of the transaction X i.e. present in transactional database D follows the given underneath condition such as:

P-set= {T.id| T belongs to D intersection X}
Where X is a subset of T

Expansion of TID projection: The expansion of p-set is considered to true when the given underneath condition is satisfied such as:

P-expanded Set (X, i) = {T`.id| T` belongs to DX intersection I belongs to T'}

Therefore, the prime benefit of MFHUIM algorithm via p-set development is that it doesn't reduce the complexity of the main algorithm, but aid in reducing the computational time and memory usage for the Search algorithm [15]. In the search algorithm, it will be only applicable on the present itemsets in the p-expanded set, which reduce the search space for mining high utility itemset as well as reduce the overall run-time of the algorithm.

5.1 Algorithm: MFHUIM algorithm

Input	Let the dataset be D, and minimum threshold be mutil	
Procedure	**Step 1**	Consider X= null;
	Step 2	Calculate the value of lu(X, i) for all the values of i that belongs to X
	Step 3	Sort the secondary list (alpha, X) in increasing order, and scan and remove

		the itemsets minimum utility threshold value. Then, remove all the empty transactions from the database X.
	Step 4	Sort the transaction T in an ascending order based on lu(X,i); while backward reading process.
	Step 5	Scan D for evaluating su(X, i) and P-expanded set for every item present in the secondary (X). Primary(X)= {i | i belongs to secondary (X) intersection su(X, i) ≥ mutil}
	Step 6	Search function call Search(X, D, primary (X), secondary (X), mutil, p-expanded (X, i))
Output	High utility itemset that occurs most number of times in the given transactional database.	

5.2 Search procedure:

Input	Itemset X, transactional database Dcx, primary itemset (x), secondary itemset su(X), minimum utility mutil threshold and extension or expanded projection in the form of p-expanded(X, i)	
Procedure	**Step 1**	Initialise the HUI list to null.
	Step 2	Execute for each loop for i to primary(X) beta = X U {i}; With the help of transaction merging, P-expanded (X, i) can be construed using Dbeta.
	Step 3	If u(beta) ≥ mutil then update HUI= HUI U beta; Scan the Dbeta to calculate su(beta, z), lu(beta, z) and p-expanded (beta, z), so that z belongs to secondary (X) list.
	Step 4	Primary (beta) ={ z belongs to Secondary (X) | all the elements present in secondary (X) are greater than or equal to minimum utility value}
	Step 5	Secondary (beta) = { z belongs to p-expanded(X) | all the elements present in p-expanded (X) are greater than or equal to minimum utility value}
	Step 6	Recursive call for search procedure, Search(z, D, primary (z), secondary (z), mutil, p-expanded (z, i))
Output	Return high utility itemsets list will be returned to the primary function.	

6. COMPARISON

In this section, the comparison between two algorithms is made on the basis of algorithmic complexities since the data is input into the transactional database. In general, the complexity of extended frequent itemset mining will be considered as Big O' O(lnw). The variable l, n, w are the different variables according to time such as:

- l: l is the number of itemsets present in the complete search space
- n: Number of transactions available in the selected database
- w: the average weight and length of transaction possible in the database.

As it is mentioned earlier MFHUIM algorithm is an expanded version of the MEFIM, but it could consider dynamic profit databases that can reduce the overall search space and reduce the complexity of the second phase of the i.e. search procedure [13]. The modification in the search procedure call such as Search(X, D, primary (X), secondary (X), mutil, p-expanded (X, i)) that can append and access the limited number of elements inside the P-set takes only O(1) time to find high utility itemsets. The value of l is considered as the total number of transactions contained for p-set structure i.e., $l = |p - set|$. The complexity of high utility itemset calculation also depends on the vicinity of itemsets diversity available in the transactional database [12]. Therefore, it could be said that if the database is sparse in nature, then overall time complexity reduces and if the datasets are densely populated then overall time complexity is comparatively less high such that $l = |D|$.

Whereas space complexity of both the algorithm is also different because of the various search procedures, in MEFIM holds the complexity of $O(|l| + lnw)$ and the space complexity of MFHUIM depends on the depth-first search that

needs an extra space to store the value at every level. Therefore, the overall complexity of the MFHUIM depends at each recursive call i.e., $O(|l| + lnwk) + O(1)$.

Table 5. Comparison results

Attributes	MFHUIM	EFIM
Runtime comparison	5.00 seconds	0.92 seconds
Transactions investigated	10000 transactions	0.123 transactions
Scanned Transaction	Max: 100 millions & Min: 10 millions	Constant
Memory Usage	9.58 MB	9.63 MB
Scalability	Min: 7 seconds (at 25%) & Max: 1003 seconds (at 100%)	Min: 2.5 seconds (at 25%) & Max: 6.96 seconds (at 100%)

7. RESULTS AND ANALYSIS

On the basis of above algorithms, a benchmark database is considered in order to compare the high utility itemset mining on the different parameters such as runtime, scalability and memory space for each algorithm. The data sources are downloaded from the frequent item transactional database section of fimi.ua.ac websit under the name of BMS database. In order to generate the results, different assumption and changes are made in original datasets for mining frequent itemsets. Following are the assumptions made while evaluating results such as:

- The general interval between each transaction lies between value [1, 10], but in order to obtain results so that frequent changes in the datasets could be noticed then the interval of [2, 50] are considered.
- Each transaction value is lifted from 2% to 10% from the original value for the same reason to randomly generate desired results for frequent utility itemsets.
- Both the algorithms are executed in a java-based environment using a computer having 3rd generation intel i5 processor with 4GB RAM.

7.1 Runtime analysis

The runtime analysis of EFIM and MFHUIM algorithm on BMS database demonstrate that on all the mutil level are equivalent and runs on a constant runtime

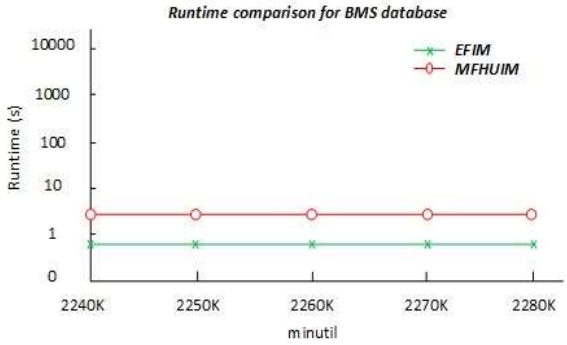

Figure 1: Run-time comparison for BMS database

The prime advantage of MFHUIM algorithm is to pruning search space as it merges and applies local utility bounds for evaluating useful run-time analysis of cost database scans.

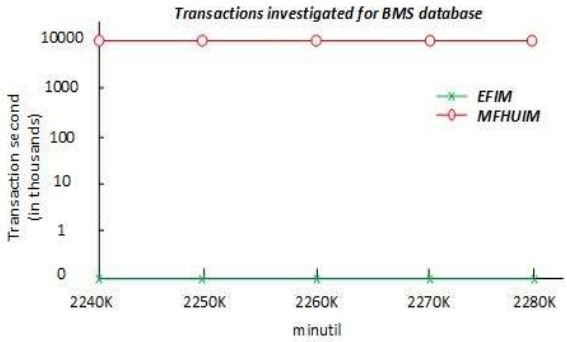

Figure 2. Total transaction investigated for BMS database

The comparison of the number of transactions scanned for mining frequent itemsets from BMS database. Certainly, transaction investigated for EFIM and MFHUIM have comparatively large difference because the number of node located in local utility tree and sub-local utility as there are no such options in MFHUIM algorithm.

7.2 Memory usage

The space complexity of both the algorithm is also different because of the various search procedures, in MEFIM holds the complexity of $O(|l| + lnw)$ and the space complexity of MFHUIM depends on the depth-first search that needs an extra space to store the value at every level. Therefore, the overall complexity of the MFHUIM depends at each recursive call i.e. $O(|l| + lnwk) + O(1)$. Thus, memory usage for both the algorithm is considered to be equivalent.

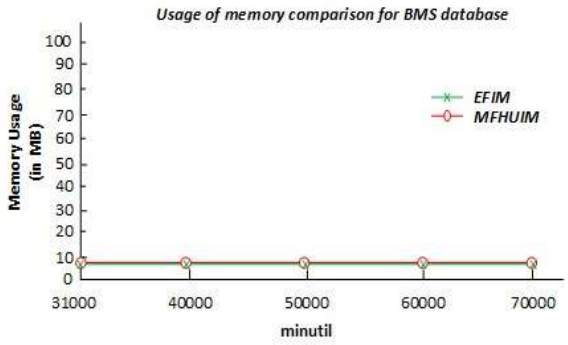

Figure 3. Usage of memory comparison for BMS database

7.3 Scalability

One of the most massive datasets from BMS database such as KOSARAK database is considered for ensuring how frequent the information stored in the state-of-the-art for understanding the scalability of the proposed MFHUIM algorithm in comparison to EFIM algorithm. The runtime of EFIM algorithm differed in an exponential manner, as a number of compared on the memory size from 50% minimum utility value to utmost bottleneck utility threshold value.

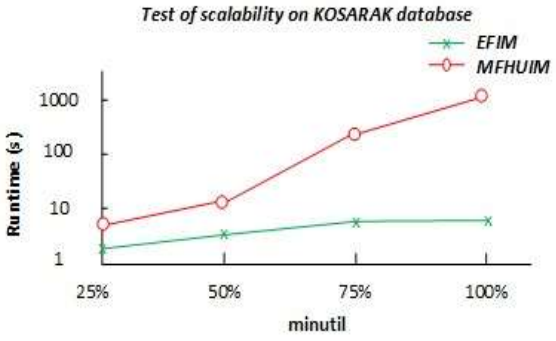

Figure 4. test of scalability on KOSARAK database

8. CONCLUSION

Association rules and frequent itemset mining in data mining is one of the most important fields for real-world big data applications. Various algorithms and techniques are proposed by the number of researchers in a previous time, but only a few algorithms consider the unit profit of the itemset transaction to be dynamic in nature which results in providing inaccurate high utility itemset values. Utility-mining covers all the significant aspects of extracting economic utility mining for rare itemset. The research conducts a literature survey on the previous related works made by different researchers and understands an effective algorithm EFIM is explained in a detailed manner for under possible issues in the existing algorithms. Further, research proposes a novel algorithm MFHUIM which uses an expanded set to extract the accurate high utility itemsets in a large set of databases. Along with that, comparison and experimental results are detailed in the research paper.

9. LIMITATIONS & FUTURE DIRECTIONS

The only limitation of this work is, there is no use of artificial intelligence. There is a scope of integrating machine learning or deep learning in itemset mining. Also, some experiments could be carried to check the efficacy of system using ensemble learning approach.

References

[1] K. Singh and B. Biswas, "Efficient Algorithm for Mining High Utility Pattern Considering Length Constraints", International Journal of Data Warehousing and Mining, vol. 15, no. 3, pp. 1-27, 2019.

[2] C. Wu, P. Fournier-Viger, J. Gu and V. Tseng, "Mining Compact High Utility Itemsets Without Candidate Generation", Studies in Big Data, pp. 279-302, 2019.

[3] M. Zihayat, M. Kargar and J. Szlichta, "A Survey of High Utility Pattern Mining Algorithms for Big Data", Studies in Big Data, pp. 75-96, 2019.

[4] P. Fournier-Viger, J. Chun-Wei Lin, T. Truong-Chi and R. Nkambou, "A Survey of High Utility Itemset Mining", Studies in Big Data, pp. 1-45, 2019.

[5] X. Su, G. Sperli, V. Moscato, A. Picariello, C. Esposito and C. Choi, "An Edge Intelligence Empowered Recommender System Enabling Cultural Heritage Applications", IEEE Transactions on Industrial Informatics, vol. 15, no. 7, pp. 4266-4275, 2019. Available: 10.1109/tii.2019.2908056 [Accessed 10 January 2020].

[6] L. Bermingham and I. Lee, "Mining place-matching patterns from spatio-temporal trajectories using complex real-world places", Expert Systems with Applications, vol. 122, pp. 334-350, 2019. Available: 10.1016/j.eswa.2019.01.027 [Accessed 10 January 2020].

[7] P. Reddy, R. Kiran, K. Zettsu, M. Toyoda, P. Reddy and M. Kitsuregawa, "Discovering Spatial High Utility Frequent Itemsets in Spatiotemporal Databases", Big Data Analytics, pp. 287-306, 2019. Available: 10.1007/978-3-030-37188-3_17 [Accessed 10 January 2020].

[8] D. Zhang, Y. Zhang, Q. Niu and X. Qiu, "Mining concise patterns on graph-connected itemsets", Neurocomputing, vol. 336, pp. 27-35, 2019.

[9] J. Luna, P. Fournier-Viger and S. Ventura, "Frequent itemset mining: A 25 years review", Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 9, no. 6, 2019.

[10] J. Wu, J. Lin and A. Tamrakar, "High-Utility Itemset Mining with Effective Pruning Strategies", ACM Transactions on Knowledge Discovery from Data, vol. 13, no. 6, pp. 1-22, 2019.

[11] J. Qu, M. Liu and P. Fournier-Viger, "Efficient Algorithms for High Utility Itemset Mining Without Candidate Generation", Studies in Big Data, pp. 131-160, 2019.

[12] C. Sivamathi and S. Vijayarani, "Multi-level Utility Mining: Retrieval of High Utility Itemsets in a Transaction Database", Computers & Electrical Engineering, vol. 76, pp. 268-282, 2019.

[13] T. Dam, K. Li, P. Fournier-Viger and Q. Duong, "CLS-Miner: efficient and effective closed high-utility itemset mining", Frontiers of Computer Science, vol. 13, no. 2, pp. 357-381, 2019.

[14] A. Renz-Wieland, M. Bertsch and R. Gemulla, "Scalable Frequent Sequence Mining with Flexible Subsequence Constraints", 2019 IEEE 35th International Conference on Data Engineering (ICDE), 2019.

[15] C. Thron and K. Tran, "A Low-Complexity, Low-Memory Frequent Itemset Mining Algorithm for Transactions With Sorted Items", SSRN Electronic Journal, 2019.

[16] M. Cafaro and M. Pulimeno, "Frequent Itemset Mining", Business and Consumer Analytics: New Ideas, pp. 269-304, 2019.

[17] M. Vanahalli and N. Patil, "An efficient parallel row enumerated algorithm for mining frequent colossal closed itemsets from high dimensional datasets", Information Sciences, vol. 496, pp. 343-362, 2019.

[18] S. Pavitra Bai and G. Ravi Kumar, "Subset Significance Threshold: An Effective Constraint Variable for Mining Significant Closed Frequent Itemsets", Advances in Intelligent Systems and Computing, pp. 449-458, 2018.

[19] S. Lessanibahri, L. Gastaldi and C. González Fernández, "A novel pruning algorithm for mining long and maximum length frequent itemsets", Expert Systems with Applications, vol. 142, p. 113004, 2020.

[20] P. Karthik and J. Saira Banu, "Frequent Item Set Mining of Large Datasets Using CUDA Computing", Advances in Intelligent Systems and Computing, pp. 739-747, 2019.

[21] S. Wazir, M. Sufyan Beg and T. Ahmad, "Weighted Frequent Itemset Mining Using OWA on Uncertain Transactional Database", Advances in Intelligent Systems and Computing, pp. 183-193, 2019.

[22] E. Varol Altay and B. Alatas, "Intelligent optimization algorithms for the problem of mining numerical association rules", Physica A: Statistical Mechanics and its Applications, vol. 540, p. 123142, 2020.

[23] P. Arun Kumar, S. Agrawal, K. Barua, M. Pandey, P. Shrivastava and H. Mishra, "Dynamic Rule-Based Approach for Shelf Placement Optimization Using Apriori Algorithm", Frontiers in Intelligent Computing: Theory and Applications, pp. 228-237, 2019.

[24] T. Pan, J. Zhao, W. Wu and J. Yang, "Learning imbalanced datasets based on SMOTE and Gaussian distribution", Information Sciences, vol. 512, pp. 1214-1233, 2020.

[25] W. Gan, J. Lin, J. Zhang, H. Chao, H. Fujita and P. Yu, "ProUM: Projection-based utility mining on sequence data", Information Sciences, vol. 513, pp. 222-240, 2020.