International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

An Efficient Pruned Matrix Aided Utility Tree for High Utility Itemset Mining from Transactional Database

V. Jeevika Tharini[1], B.L.Shivakumar[2]

Submitted: 03/11/2022 **Accepted:** 08/02/2023

Abstract: High Utility Itemset Mining (HUIM) is the progression of identifying highly profitable items by considering the unit profit of the item from the huge transactional database. HUIM is an essential subject with broad applications in recent years. HUIM paves the way to know the profitable items using factors namely profit and quantity. Until today, abundant algorithms have been found to mine High Utility Itemset (HUI) and it is entirely different from the conventional mining algorithms. Most of the utility mining algorithms generate the itemset recurrently and scan the database redundantly, which leads to computational complexity. To overcome this issue, a pruning technique is introduced with a matrix and Frequent Pattern (FP) tree is constructed with the pruned matrix whereby the complexity in HUI identification is minimized. Experimental results are investigated using a benchmark dataset and the outcome depicts that the proposed pruned matrix-aided utility tree (PMAUT) outperforms the existing state of art techniques in terms of time consumption and memory usage.

Keywords: *Internal utility, unit profit, external utility, pruned matrix, frequent pattern, transaction weighted utility model, threshold value, and tree.*

1. Introduction

Traditional mining algorithms like Association Rule Mining (ARM) and Frequent Itemset Mining (FIM) are developed to mine the frequencies of every individual item in the transaction database by considering their confidence value or minimum threshold value [1, 2]. FIM and ARM are insufficient in finding highly profitable itemsets, which may have less frequency in the transaction. For example, in a supermarket sugar may be sold hundred or more than a hundred kilograms per day while fewer kilograms of cashew nuts may be sold for a day or month. Sugar has a greater incidence value but is sold with less rate of profit while cashew nuts have a lesser incidence with a higher profit value for traders. Both FIM and ARM find the frequent item sugar with lower profit from the transaction data and to overcome this limitation of conventional algorithms, HUIM approaches are developed. Several researchers proposed diverse approaches and mechanisms for HUIM identification [3, 4].

Utility mining is one of the newest forms of mining items with high utility in data mining. The technique of extracting HUIs from a database is known as utility mining. Each object in utility mining has its utility function. The utility value is determined by the dataset's characteristics. HUIs are items with a utility value greater than the minimal utility criterion. Utility mining considers all the factors like the number of items and unit profit of itemsets [5]. Utility mining can be applied to a quantitative database, i.e. a transaction database with the number of items sold [6]. HUIM is developed to find the items having high utility value from the quantitative database. In utility mining, the number of items occurred in a transaction is the internal utility and profit of individual items as an external utility [7, 8].

The most provoking task in recent research is efficiently mining HUIs. User preference determines the utility rate of the product. Several algorithms were projected to spot the most profitable items. Still, the profitable item discovery process met with various limitations likely higher time and memory utilization, generation of various and repeated candidate items, and in certain cases, some of the needed items may be left from the item generation. Hence, this research is formulated with pruned matrix and Frequent Pattern (FP) tree [9] to overcome the mentioned limitations. The main aim of the research is to develop an approach that can effectively handle candidate generation and database scans.

The major contribution of the research work is

- To construct a dynamic matrix, a utility matrix pruning technique is incorporated that can minimize the recurrent database scan.
- Subsequently, a utility-based FP tree is generated from the pruned matrix and it can avoid redundant candidate itemset generation.

Research Scholar, Sri Ramakrishna College of Arts & Science, Coimbatore
Principal, Sri Ramakrishna College of Arts & Science, Coimbatore

The rest of the research work is organised as follows: Section 2 details the recent research in utility mining, section 3 depicts the proposed PMAUT with its running example, section 4 presents the results and performance of comparisons of existing and proposed algorithms, and section 5 explains the conclusion with future recommendations.

2. Related Works

The idea of HUIs was initially put out in 2003 [10], and a potent foundation for mining HUIs was unveiled in 2004 [11]. After that, other researchers have suggested quick and memory-effective ways to mine HUI data using several techniques. The most significant subcategories of these methodologies include utility-list-based techniques, pattern-growth-based techniques, projection-based techniques, hybrid techniques, and apriori-like techniques [12, 13]. In addition, several variations and extensions have been developed in the literature to mine beneficial versions of HUIs, including top-k HUI mining [14], EUP Growth+ [15] and HUP Miner [16].

Other expansions of the HUIM problems include the concept of fuzzy [17], and finding HUIs utilising evolutionary algorithms and optimization strategies [18]. High utility pattern mining has also been employed in several applications in the literature to address a variety of issues. Discovery of weblog data [19], extraction of weblog topic [20], topic extraction from social network [21], customer purchase behaviour analysis and identifying marketing solution [22], forecasting of revenue [23], manufacturing plan establishment [24], service-oriented computing [25], revenue management by utility-based suggestion [26], and retrieval of user behaviour patterns in the context of mobile commerce [27] are some examples of applied research.

The most significant suggested techniques for the HUI mining problem, as well as its various variations and expansions, are briefly summarised in this part, which also looks at these methods' mining strategies. The prominent drawbacks identified in the literature namely database handling issues, candidate generation, and repetitive computation of similar itemset. By considering these drawbacks, an effective Pruned Matrix Aided Utility Tree (PMAUT) for HUIM is developed and discussed in the subsequent section.

3. Proposed Methodology - Pruned Matrix Aided Utility Tree (Pmaut)

This section explains the illustrative example with preliminaries for the proposed methodology and the process of pruning matrix-aided tree. Let $I = \{m_1, \mathrm{m}_2, \mathrm{m}_3, \dots \dots, \mathrm{m}_\mathrm{n}\}$is a set of n individual items in a data store with a transaction$T = \{t_1, t_2, t_3 \dots \dots \dots \dots t_n\}$ where every transaction in the data store lies within $T_r \in DB$. In this T_r is a subgroup of I and has a distinctive identifier r called TID. The profit of the item is indicated as external utility values depicted in Table 2 and the count of the purchased products is indicated as an internal utility in Table 1. A minimal utility value is assigned based on the user's preferences. This work, elucidated a running example with twelve transactions and eight objects A to I, as depicted in Table 1.

Table 1. Transaction Database (DatastoreDB)

Item	A	B	C	D	E	F	G	H	I
T1	1	2	0	2	0	1	2	0	0
T2	0	0	0	1	0	0	0	1	0
T3	2	2	0	0	0	2	1	0	0
T4	0	1	1	1	2	2	0	0	0
T5	0	0	0	2	0	1	0	0	0
T6	0	1	1	0	2	1	0	0	0
T7	3	0	0	0	0	0	2	0	0
T8	0	0	2	0	1	0	3	0	0
T9	2	1	2	1	1	0	3	0	0
T10	0	2	0	1	2	0	2	0	0
T11	0	0	0	0	0	1	1	0	1
T12	0	1	5	0	0	0	0	0	0

Table 2. Unit Profit

Item	A	B	C	D	E	F	G	H	I
Profit	1	2	1	5	4	3	1	2	1

Definition 1: The utility of an item m$_j$ in the transaction T$_r$ is signified as u(m$_j$, T$_r$) and equated as $u(m_j, T_r) = \mathrm{r}(m_j, T_r) \times \mathrm{pr}(m_j)$. For example, an item A's utility value in a transaction T$_1$ is estimated as u(A,T$_1$)=1×1=1, where u(A, T$_2$)=0,u(A, T$_3$)=2,u(A, T$_4$)=0,u(A, T$_5$)=0,u(A, T$_6$)=0,u(A, T$_7$)=3,u(A, T$_8$)=0,u(A, T$_9$)=2,u(A, T$_{10}$)=0,u(A, T$_{11}$)=0,u(A, T$_{12}$)=0.

Definition 2: The utility of an item m$_j$ in the data store DB is signified as u(m$_j$) and equated as $u(m_j) = u(m_j, A_p) + \cdots + u(m_n, A_n)$. For example, the utility of item A in the whole transaction is estimated as u(A)=u(A,T$_2$)+u(A,T$_3$)+u(A,T$_4$)+u(A,T$_5$)+u(A,T$_6$)+u(A,T$_7$) +u(A, T$_8$)+u(A,T$_9$)+u(A,T$_{10}$)+u(A,T$_{11}$)+u(A,T$_{12}$)=8, where u(B)=18, u(C)=11, u(D)=40, u(E)=32, u(F)=24, u(G)=14, u(H)=2, u(I)=1.

Definition 3: The transaction utility of a transaction T$_r$ is signified as tu and equated as $tu(T_r) = \sum_{\mathrm{L} \subseteq T_r} \mathrm{u}(\mathrm{L}, T_r)$. For example, tu(T$_1$)=u(A,T$_1$) + u(B,T$_1$) + u(D,T$_1$) + u(F,T$_1$) + u(G,T$_1$) = 18, where tu(T$_2$)=$_2$)=7, tu(T$_3$)=18, tu(T$_4$)=22, tu(T$_5$)=13, tu(T$_6$)=14, tu(T$_7$)=5, tu(T$_8$)=9, tu(T$_9$)=18, tu(T$_{10}$)=19, tu(T$_{11}$)=5, tu(T1$_2$)= 7.

Definition 4: The total transaction utility of a transaction T$_r$is signified as tu and equated as $TTU = \sum_{T_r \epsilon \mathrm{DB}} \mathrm{Tu}(T_r)$. For example, TTU(T$_r$)=T$_1$+T$_2$+ T$_3$+ T$_4$+ T$_5$+ T$_6$+ T$_7$+ T$_8$+ T$_9$+ T$_{10}$+ T$_{11}$+ T$_{12}$=150 or A+B+C+D+E+F+G+H+I=150.

Table 3. Utility Information of individual item, transaction and total transaction

Item	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**Transaction Utility**
T$_1$	1	2	0	10	0	3	2	0	0	18
T$_2$	0	0	0	5	0	0	0	2	0	7
T$_3$	2	4	0	0	0	6	1	0	0	13
T$_4$	0	2	1	5	8	6	0	0	0	22
T$_5$	0	0	0	10	0	3	0	0	0	13
T$_6$	0	2	1	0	8	3	0	0	0	14
T$_7$	3	0	0	0	0	0	2	0	0	5
T$_8$	0	0	2	0	4	0	3	0	0	9
T$_9$	2	2	2	5	4	0	3	0	0	18
T$_{10}$	0	4	0	5	8	0	2	0	0	19
T$_{11}$	0	0	0	0	0	3	1	0	1	5
T$_{12}$	0	2	5	0	0	0	0	0	0	7
Transaction Weighted Utility	8	18	11	40	32	24	14	2	1	150

The dynamic matrix is a potent tool for efficiently organising notions so it can generate meaningful insightful and act promptly. Initially, the matrix construction is accomplished by utilizing the values from the table 3. The dynamic matrix construction process is given in Equation 1. To increase the effectiveness of the HUI mining process, the proposed technique applies a utility matrix-based pruning strategy to the original utility matrix. To do this, every row in the related utility matrix that does not include an item based on the condition in Equation 2 is eliminated. UM-prune is an effective pruning approach that may identify utility matrices that shouldn't be built during the mining procedure of the proposed methodology, which decreases the execution time and needed memory space of PMAUT.

The procedure of adding the item into the matrix is given in Algorithm 1.

Algorithm 1. Adding Item into Matrix

```
Begin
Temp:=0
for p:= to count rows of mtr do
    mtr[p][0]=:=mtr[p][0]+mtr[p][m]
    if (mtr[p][m]=0)then
        temp:=temp+mtr[p][0];
mtr[0][0]:=mtr[0][0]-temp
end;
```

$$Dynamic\ Matrix\ Construction = \begin{bmatrix} 1 & 2 & 0 & 10 & 0 & 3 & 2 & 0 & 0 & 18 \\ 0 & 0 & 0 & 5 & 0 & 0 & 0 & 2 & 0 & 7 \\ 2 & 4 & 0 & 0 & 0 & 6 & 1 & 0 & 0 & 13 \\ 0 & 2 & 1 & 5 & 8 & 6 & 0 & 0 & 0 & 22 \\ 0 & 0 & 0 & 10 & 0 & 3 & 0 & 0 & 0 & 13 \\ 0 & 2 & 1 & 0 & 8 & 3 & 0 & 0 & 0 & 14 \\ 3 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 5 \\ 0 & 0 & 2 & 0 & 4 & 0 & 3 & 0 & 0 & 9 \\ 2 & 2 & 2 & 5 & 4 & 0 & 3 & 0 & 0 & 18 \\ 0 & 4 & 0 & 5 & 8 & 0 & 2 & 0 & 0 & 19 \\ 0 & 0 & 0 & 0 & 0 & 3 & 1 & 0 & 1 & 5 \\ 0 & 2 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 7 \\ 8 & 18 & 11 & 40 & 32 & 24 & 14 & 2 & 1 & - \end{bmatrix} \text{---------(1)}$$

$$Pruning\ strategy\ (UM_{Pruning}) = \frac{minutil(TU)+\text{maxutil(TU)}}{2} \text{---------(2)}$$

Definition 5: The utility matrix pruning for the transaction Tr is equated as $\boldsymbol{UM_{Pruning}} = \frac{\boldsymbol{minutil(TU)}+\textbf{maxutil(TU)}}{\textbf{2}}$**.** For example, $UM_{pruning}$=(5+22)/2=14.

By applying the pruning strategy, the items namely A, C, H, and I are pruned from the dynamic matrix. The pruned matrix is constructed with the items B, D, E, F, and G, which is given in Equation 3. The pruning process with the support of a matrix can eventually minimize the redundant scan of the database scan and the count of the item also minimized that can effectively enhance the performance of the proposed PUMAT technique. The acquired values in the pruned matrix are utilized for FP tree construction.

$$Dynamic\ Matrix\ After\ Pruning = \begin{bmatrix} 2 & 10 & 0 & 3 & 2 \\ 0 & 5 & 0 & 0 & 0 \\ 4 & 0 & 0 & 6 & 1 \\ 2 & 5 & 8 & 6 & 0 \\ 0 & 10 & 0 & 3 & 0 \\ 2 & 0 & 8 & 6 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 4 & 0 & 3 \\ 2 & 5 & 4 & 0 & 3 \\ 4 & 5 & 8 & 0 & 2 \\ 0 & 0 & 0 & 3 & 1 \\ 2 & 0 & 0 & 0 & 0 \\ 18 & 40 & 32 & 24 & 14 \end{bmatrix} \text{---------(3)}$$

The most important information is kept in the FP Tree, which is a pattern of data generation. The main benefit of the FP Tree is that it only has to read the whole set of data twice to build a tree. Efficiency is gained in the FP Tree because memory usage is compact. The minimal support count for the item is considered while building the tree. Candidate items are created from the reduced tree. The items and their support count ordering is given in Figure 1 and the process of tree construction from the ordered item is illustrated in Figure 2

Items and their Support Count

Unordered Item			Order Item	
Item	Count		Item	Count
B	7		B	7
D	6	➡	G	7
E	5		D	6
F	6		F	6
G	7		E	5

Ordered Transaction

Transaction	Items
T_1	B, G, D, E
T_2	D
T_3	B, G, F
T_4	B, D, F, E,
T_5	D, F
T_6	B, F, E
T_7	G
T_8	G, E
T_9	B, G, D, E
T_{10}	B, G, D, E
T_{11}	G, F
T_{12}	B

Fig. 1. Items and their Ordering

The items are ordered with the assistance of support count and it is utilized during the construction of the FP tree. The process of candidate itemset generation from the pruned FP tree is minimized. The null transaction and repeated candidate itemset generation are eliminated by the FP tree. The general form of candidate generation and FP tree-based candidate generations are given in Table 4.

Table 4. Candidate Generation

Itemset	**The general form of generation and count**		**FP tree-based generation and count**	
2-Itemset	BG, BD, BF, BE, GB, GD,GF, GE,DB, DG, DF, DE, FB, FG, FD, FE, EB, EG, ED, EF	20	BG, BD, BF, BE, GD, GF, GE, DF, DE, FE	10
3-Itemset	BGD, BGF, BGE, BDG, BDF, BDE, BFG, BFD, BFE, BEG, BED, BEF,…..FED, FEG,FEB.	60	BGD, BDF, BGF, BGE, BDE, GDF, GDE, BDE, DFE	9
4-Itemset	BGDF, BGDE, GDFE, GDFB, DFEB, DFEG, FEBG, FEBD, EBGD, EBGF	10	BGDF, BGDE, BDFE, BFEG	4
5-Itemset	BGDFE	1	Nil	-

Definition 6: The utility of an itemset L in the data storeDB is signified as u(L), and indicated as $\boldsymbol{u(L)} = \sum_{\mathbf{L} \subseteq \boldsymbol{T_r} \boldsymbol{v} \boldsymbol{T_r} \subseteq \mathbf{DB}} \mathbf{u}(\mathbf{L}, \boldsymbol{T_r})$**.** For example, the utility value of itemset BG in entire transaction is estimated as u(BG)= 4+5+5+6=20, u(BD)=25, u(BF)=38, u(BE)=28, u(GD)=19, u(GF)=16, u(GE)=14, u(DF)=24, u(DE)=35, u(FE)=25. Likewise, the utility value of every itemset is calculated.

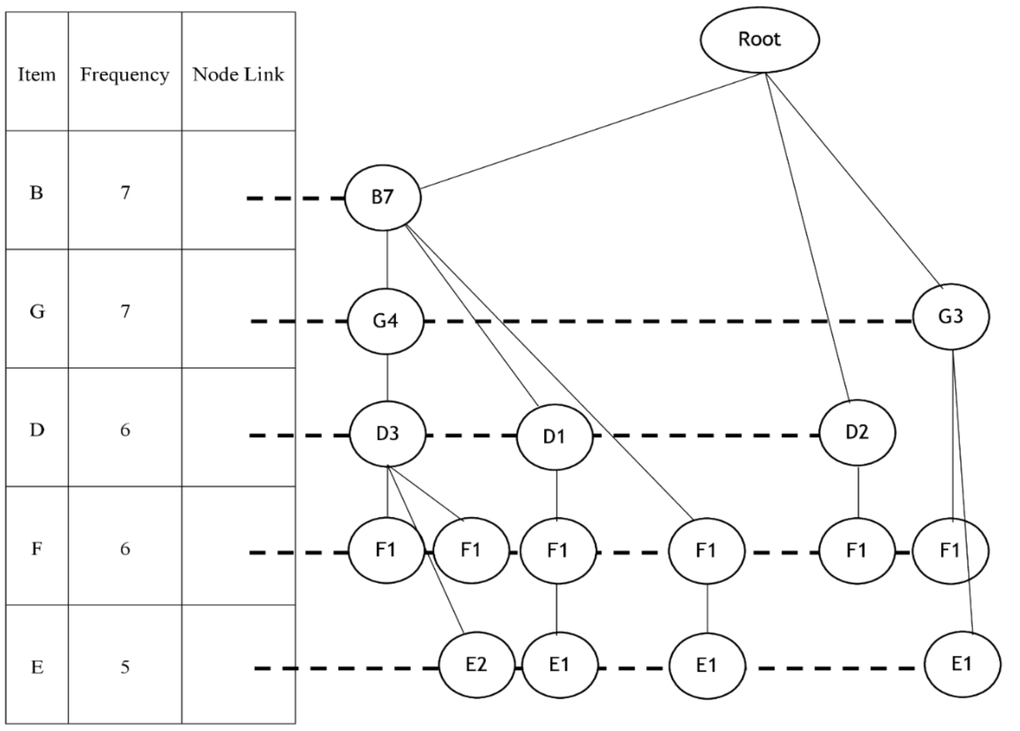

Fig. 2. Pruned FP Tree

The itemset generated from the FP tree is evaluated with the threshold value 21 whereas the attained HUIs are BG, BD, BE, BF, DE, DF, FE, BDE, BGD, BFE, BDF, BGE, GDE, BGDE, and BDFE. The retrieval of the candidate itemset from the transaction by using the proposed technique is highly effective that minimizes the usage of memory and computational time also effective. The HUIs are retrieved for diverse thresholds and the thresholds with a similar count of the itemset are given in this section.

4. Result and Discussion

In this part, the performance of the proposed technique is examined and assessed. On a PC with a core i3 CPU running at 2.8 GHz, 8 GB of RAM, and a 500 GB hard drive, all experiments are carried out. Both processing time and IO time are included in reported runtimes. Based on real-world datasets, PMAUT performance is assessed. From [28], the datasets have been downloaded for this research. A statistical breakdown of the datasets utilised in the research is shown in Table 5.

Table 5. Dataset Information

Dataset Name	Average Transaction Length	Item Count	Transaction Count
Chess	37	75	3196
Mushroom	23	119	8124
Connect	43	129	67557
Foodmart	44	1559	4141

4.1. Comparison of Time Consumption

The time taken to generate the candidate itemset and to identify a profitable itemset from the generated candidate itemset is discussed in this section. In the conducted test, the runtime of five datasets and three existing algorithms are compared. The PMAUT algorithm has shown improved outcomes than the HUIP [16], EUP Growth+ [15], and Top-K algorithms [14]. Since, the PMAUT algorithm creates effective combinations of itemset from the dataset, which can significantly eliminate the computational complexity. The time taken to perform the existing and proposed approach is given in Table 6 and Figure 3.

Table 6. Comparison of Time Consumption

Dataset	Algorithm	T-35	T-30	T-25	T-20	T-15
Chess	HUIP	417623	408984	392831	388923	379231
	EUP-Growth+	326304	345799	349508	350280	351838
	Top-K	184311	193474	198372	201685	207791
	PMAUT	160384	169315	198301	201108	218391
Mushroom	HUIP	457796	459467	534397	598615	764682
	EUP-Growth+	39379	82342	86557	8804	104105
	Top-K	43471	34212	35977	36967	37866
	PMAUT	37011	37132	39546	34037	33130
Connect	HUIP	456242	497761	519877	567941	597371
	EUP-Growth+	40968	41873	43983	43317	48364
	Top-K	357163	372872	393877	403873	411198
	PMAUT	35761	39837	40641	41980	43981
Foodmart	HUIP	59431	61977	62928	63977	65431
	EUP-Growth+	45622	46874	47752	48763	49871
	Top-K	37284	37653	38273	40853	43871
	PMAUT	29886	30879	32982	35875	39088

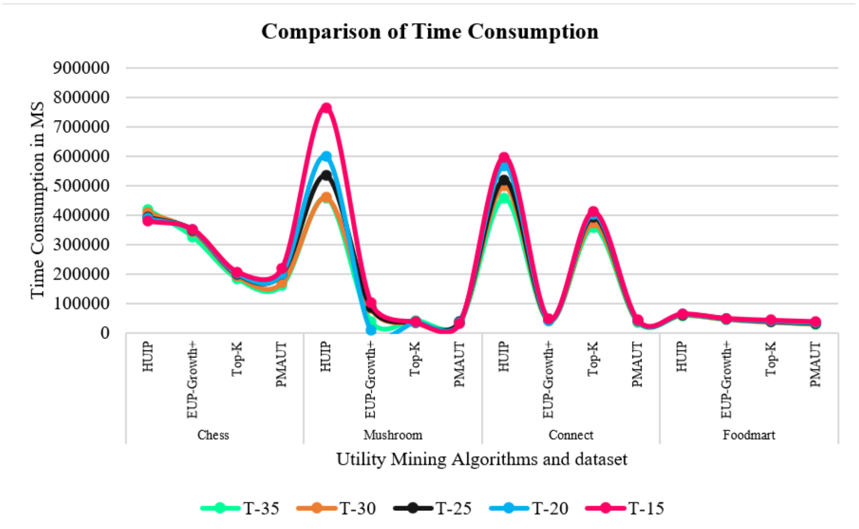

Fig. 3. Comparison of Time Consumption

From the observation of Figure 3, it is found that the PMAUT approach has given better results in terms of time consumption that is minimal memory usage than the HUIP [16], EUP Growth+ [15], and Top-K algorithm [14]. The time consumption for the dataset chess, mushroom, connect, and foodmart with diverse threshold values are given in Figure 3. The time consumption is denoted in millisecond (MS) and the results are compared for diverse threshold value.

4.2. Comparison of Memory Usage

The memory space utilised for storing the generated candidate itemset and other processed information is compared in this section. The experimental investigation of the PMAUT, HUIP [16], EUP Growth+ [15], and Top-K algorithm [14]. algorithms with benchmark datasets with several threshold values given in Table 7 and Figure 5.

Table 7. Comparison of memory usage

Dataset	Algorithm	T-35	T-30	T-25	T-20	T-15
Chess	HUIP	76.18	72.59	69.46	65.24	61.24
	EUP-Growth+	77.49	74.98	62.91	62.13	51.83
	Top-K	36	37.77	39.81	50.68	52.59
	PMAUT	34.3	34.16	33.11	32.64	32.24
Mushroom	HUIP	38.66	42.91	45.89	48.71	51.36
	EUP-Growth+	12.16	12.95	14.24	15.16	15.99
	Top-K	13.16	20.14	22.99	19.57	21.41
	PMAUT	10.36	10.73	12.61	12.71	13.26
Connect	HUIP	42.82	46.76	47.66	49.79	52.76
	EUP-Growth+	39.32	41.78	43.19	48.95	49
	Top-K	31.42	32.77	33.49	34.86	36.59
	PMAUT	30.59	31.32	33.44	34.31	36.11
Foodmart	HUIP	67	68	71	72	72.5
	EUP-Growth+	63	64	64.87	66.32	68
	Top-K	61	63	64.23	65	66
	PMAUT	53	54	56	57	57.35

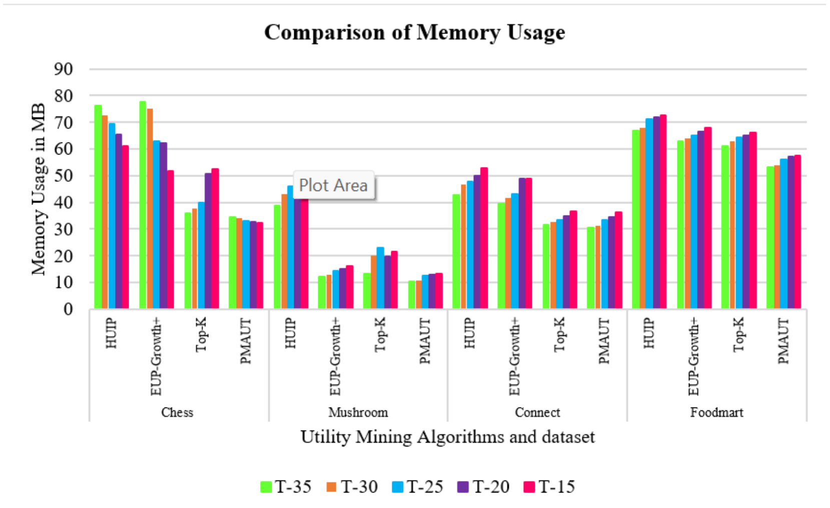

Fig. 4. Comparison of memory usage

From the observation of Figure 4, it is found that the PMAUT approach has given better results in terms of memory consumption which is minimal memory usage than the HUIP [16], EUP Growth+ [15], and Top-K algorithm [14]. The memory usage for the dataset chess, mushroom, connect, and foodmart with diverse threshold values are given in Figure 4. Memory usage is denoted in megabytes (MB) and the results are compared for diverse threshold value.

5. Conclusion

Currently, HUIM is a significant research topic and it can expose extremely profitable products. An abundant number of algorithms have been considered to proficiently mine the HUIs from the quantitative data store and many of the researchers applied statistical analysis to spot the essential information. This research work introduces a pruning strategy based pruned utility matrix and FP tree. An effective pruned matrix is built based on the suggested pruning method, known as UM-prune that enhances the performance of HUI extraction. The utility matrix's item sequencing type balanced the number of items and rows in the constructed matrices, which increased the mining performance of the process. An FP tree is constructed from the pruned matrix and redundant candidate generation is avoided in this research work. The testing results demonstrated that the suggested method uses less memory and executes faster than its predecessors. In future, the optimal HUIs can be retrieved with the assistance of bio-inspired approaches.

Reference

[1] Fournier-Viger, P., Chun-Wei Lin, J., Truong-Chi, T ., & Nkambou, R. (2019). A survey of high utility itemset mining. *High-utility pattern mining: Theory, algorithms and applications*, 1-45.

[2] Gan, W., Lin, J. C. W., Fournier-Viger, P., Chao, H. C., Tseng, V. S., & Philip, S. Y. (2019). A survey of utility-oriented pattern mining. *IEEE Transactions on Knowledge and Data Engineering*, *33*(4), 1306-1327.

[3] Tharini, V. J., & Shivakumar, B. L. High-Utility Itemset Mining: Fundamentals, Properties, Techniques and Research Scope. In *Computational Intelligence and Data Sciences* (pp. 195-210). CRC Press.

[4] Yen, S. J., & Lee, Y. S. (2007, September). Mining high utility quantitative association rules. In *International Conference on Data Warehousing and Knowledge Discovery* (pp. 283-292). Springer, Berlin, Heidelberg.

[5] R. Chan, Q. Yang, and Y. Shen, ―Mining high utility Itemsets,‖ The Third IEEE International Conference on Data Mining, pp. 19-26, 2003.

[6] Liu, Y., Liao, W. K., &Choudhary, A. (2005, May). A two-phase algorithm for fast discovery of high utility itemsets. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*(pp. 689-695). Springer, Berlin, Heidelberg.

[7] J. Han, J. Pei, Y. Yin, ―Mining frequent patterns without candidate generation‖, in Proceedings of the ACM-SIGMOD Int'l Conf. on Management of Data, pp. 1-12.

[8] Lan, G. C., Hong, T. P., & Tseng, V. S. (2014). An efficient projection-based indexing approach for mining high utility itemsets. *Knowledge and information systems*, *38*(1), 85-107.

[9] Jeevika Tharini, V., & Vijayarani, S. (2020). Bio-inspired High-Utility Item Framework based Particle Swarm Optimization Tree Algorithms for Mining High Utility Itemset. In *Advances in Computational Intelligence and Informatics: Proceedings of ICACII 2019* (pp. 265-276). Springer Singapore.

[10] Chan R, Yang Q, Shen Y-D (2003) Mining high utility itemsets. In: Proceedings of 3rd IEEE international conference data mining, 2003, (Washington, D.C., USA, 2003) pp. 19–22

[11] Yao H, Hamilton HJ, Butz CJ (2004) A foundational approach to mining itemset utilities from databases. In: Proceedings of 3rd SIAM international conference on data mining, 2004, (Orlando, Florida, USA, 2004) pp 482–486

[12] Gan W, Lin JC-W, Fournier-Viger P, Chao H-C, Tseng VS, Yu PS (2018) A survey of utility-oriented pattern mining. arXiv: 1805.10511

[13] Rahmati B, Sohrabi MK (2019) A systematic survey of high utility itemset mining. Int J Inf Technol Decis Mak 18(4):1113–1185

[14] Krishnamoorthy S (2019) Mining top-k high utility itemsets with effective threshold raising strategies. Expert Syst Appl 117:148–165

[15] Sharmila, P., & Meenakshi, S. (2018). AN ENHNACED HIGH UTILITY PATTERN APPROACH FOR MINING ITEMSETS. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, *7*(1).

[16] Fournier-Viger, P., Zhang, Y., Lin, J. C. W., Dinh, D. T., & Bac Le, H. (2020). Mining correlated high-utility itemsets using various measures. *Logic Journal of the IGPL*, *28*(1), 19-32.

[17] Kannimuthu S, Premalatha K (2014) Discovery of high utility itemsets using genetic algorithm with ranked mutation. Appl Artif Intel 28(4):337–359

[18] Wu JM-T, Zhan J, Lin JC-W (2017) An ACO-based approach to mine high-utility itemsets. Knowl-Based Syst 116:102–113

[19] Tamilselvi T, Arasun GT (2019) Handling high web access utility mining using intelligent hybrid hill climbing algorithm based tree construction. Clust Comput 22:145–155

[20] Bakariya B, Thakur GS (2015) An efficient algorithm for extracting high utility itemsets from weblog data. IETE Tech Rev 32(2):151–160

[21] Choi H-J, Park CH (2019) Emerging topic detection in twitter stream based on high utility pattern mining. Exp Syst Appl 115:27–36

[22] Gan W, Lin JC-W, Fournier-Viger P, Chao H-C, Fujita H (2018) Extracting non-redundant correlated purchase behaviors by utility measure. Knowl-Based Syst 143:30–41

[23] Weng C-H (2016) Discovering highly expected utility itemsets for revenue prediction. Knowl-Based Syst 104:39–51

[24] Yun U, Lee G, Yoon E (2017) Efficient high utility pattern mining for establishing manufacturing plans with sliding window control. IEEE Trans Ind Electron 64(9):7239–7249

[25] Kannimuthu S, Premalatha K, Shankar S (2012) Investigation of high utility itemset mining in service oriented computing: deployment of knowledge as a service in E-commerce. In: 2012 fourth international conference on advanced computing (ICoAC), pp 1–8

[26] Yang R, Xu M, Jones P, Samatova N (2017) Real time utility-based recommendation for revenue optimization via an adaptive online Top-K high utility itemsets mining model. In: 13th international conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD), pp 1859–1866

[27] Shie B-E, Yu PS, Tseng VS (2013) Mining interesting user behavior patterns in mobile commerce environments. Appl Intell 38(3):418–435

[28] Fournier-Viger P, Gomariz A, Soltani A, Lam H, Gueniche T (2014) SPMF: open-source data mining platform. http://www.philippe-fournier-viger.com/spmf