An Artificial Bee Colony Algorithm to Mine Periodic High Utility Itemsets

S. Viveka, B. Kalaavathi

Abstract: ***High utility itemset mining is one of the key research areas in data mining in recent years. The main challenge in high utility itemset mining is the exponential growth space for finding the high utility itemsets. Several algorithms were proposed to mine high utility itemsets and all of them suffer because of the huge search space. Genetic Algorithms (GA) are now attracting researchers since it reduces the search space tremendously. In this paper, a novel algorithm based on Artificial Bee Colony PHUIM-ABC is proposed to mine the periodic high utility itemsets. During initial phase, the transaction database is scanned to find the 1-High Transaction Weighted Utility Itemsets (1-HTWUI). The 1-HTWUI is used as measured as the parameter in assigning the onlooker bee. Five real life data sets are used to evaluate the performance of the proposed algorithm with the existing state-of-art algorithms .The execution time and memory usage are the parameters used to measure the performance. Experimental results show that the proposed PHUIM-ABC algorithm performs better than the state-of-art non heuristic algorithms.***

Index Terms: Artifical bee colony, Genetic algorithm, High utility itemset, Itemset mining, Periodic high utility itemset, , Utility mining,

I. INTRODUCTION

Knowledge discovery in database (KDD) is one of the major research areas. It is because the data contains many hidden and highly useful information. Many organizations have huge data, and these data can be analyzed to extract the useful information which is hidden inside. These data exponentially increases day by day. Association Rule Mining (ARM), Clustering, Classification, Linear Regression are various data mining technique to extract the information from hidden dataThe Frequent Pattern Mining (FPM) is a part of data mining which finds the frequently occurring patterns in the databases. ARM finds the associations that exist between the frequent patterns. Both the frequent pattern and ARM finds the information based on the threshold value given as the user input.

Out of the many data mining algorithms ARM attracted many researchers, and number of algorithms has been developed in ARM. ARM considers only the number of times the item appears in the transaction but does not consider the importance of it. In marketing, web stream databases the profit of the items also plays an important role. An item may not appear frequently in a database but the infrequent item may earn more profit. These infrequent high profit item needs to be considered in finding the itemset. High Utility Itemset Mining (HUIM) is a successor of ARM which finds the high profitable items. Instead of finding the frequent pattern above the user specified threshold HUIM finds the High Utility Itemsets (HUIs). High Utility Itemset (HUI) is the item whose profit in the database is greater than or equal to the user specified threshold profit. The utility measurement can be in terms of utilization, cost, profit, and others. The major challenge faced in this is that the downward closure property is not satisfied. The downward closure property is satisfied in FPM and ARM. During the first step, the 1-candidate itemset is found, and the i+1-candidate patterns are found from the i-candidate patterns. The search space of i+1-candidate itemset is limited to i-candidate itemset but this property is not satisfied in the HUIM. The search space of finding HUIM grows exponentially with the length of the patterns. Many algorithms have been introduced to find the HUIM efficiently. The two phase miner is a state of art algorithm which mines the HUIM effectively in two phases. Downward closure property using transaction weight has been used in this two phase miner to effectively prune the search space. An enhanced version of the algorithm FHM uses a structure EUCS based on co-occurrence pruning to trim down the search space. UP tree and UPtree + which are tree based algorithms have been proposed by reducing the search space by upper bounds.

Another tree based algorithm is proposed by Lin et al. which used condensed high-utility pattern (HUP)-tree. FP tree and TWU based algorithm for mining HUI is proposed by Han et al. (2000)[1]. Lan et al. (2014)[2] proposed an indexed projection of the dataset for mining HUI, this algorithm used a different approach of projecting pseudo database for every iteration instead of depth first approach used in tree based algorithms. These algorithms do not consider the period of occurrence of an itemset and these HUI mining algorithms cannot be directly useful for PHUI mining. To address these issues periodic high utility mining is proposed. In this proposed work a new algorithm PHUIM-ABC is proposed to mine all the high profit items that occurred in the transaction during the specified period.

II. LITERATURE SURVEY

Liu et al. (2005)[3] proposed Mining using Expected Utility MEU algorithm for HUIM. The downward closure property is not satisfied in HUIM. A novel approach to include downward closure property in HUIM is proposed by using over estimation method.

Revised Manuscript Received on April 17, 2019

S.Viveka, Assistant Professor, Department of Information Technology, Velalar College of Engineering and Technology, Erode, India.

Dr.B.Kalaavathi, Professor & Head, Department of Computer Science, K.S.R. Institute for Engineering and Technology, Tiruchengode, India.

The algorithm runs in two phases. During the first space the utility of single itemset and Transactional Weight Utility of a 1-candidate itemset {I} is calculated. The Transactional Weight Utility of an itemset {I} TWU(I) is calculated by summing up the transaction utility of all transactions which contains {I}. A given itemset {I} is said to be High Transactional Weight Utilization Itemset HTWUI if the TWU(I) > = minimum utility threshold is said to HTWUI. TWU(I) follows the downward closure property and in second phase all the item which are not part of HTWUI are discarded. HUI are mined from the HTWUI.

A tree based algorithm based on FP-tree to mine HUI is proposed by Tseng et al. (2010)[4]. Utility Pattern tree (UP-tree) is used to record the transaction details. In Phase I, the transaction utility and TWU(I) are calculated. The promising items {I} have TWU(I) >= minimum utility. The other non-promising items are discarded from the database before the second phase. In second phase UP-tree is constructed. Each node of UP-tree contains three values, item name, item count, estimated utility value and two links, one link is to the parent node and the other is the horizontal link, A header table maintains link for the first node in UP-tree for all the promising items. It uses two strategies to maintain the tighter bound of promising items. Discarding global unpromising items (DGU) strategy is followed to eliminate the non promising item and discarding global node utilities (DGN) is done to discard the descendants of non promising items during global UP-tree construction. HUI are mined recursively constructing condition pattern based subtrees.

HUI-Miner is proposed by Liu et al. (2012) [5] to mine HUI introduced list based data structure. It runs in two phases, in phase-I TU, TWU are calculated. Items in transactions are ordered according to TWU. In phase-II., list data structure is built. The list data structure has three attributes TID, iutility and rutility, where itutility is the utility of an item {I} in the transaction and rutility is calculated by adding the utility of all items present after {I} in TWU ordering in the transaction. Two pruning strategies based on ituility and rutility are proposed. HUI are mined by a recursive procedure by adding the extension to the candidate itemset.

FHM algorithm is proposed by Fournier-Viger et al. (2014)[6] is an improvement of HUI-Miner. It uses the utility list structure and three pruning strategies, TWU pruning, iutility pruning and iutity and rutility pruning. The k-candidate itemset are generated from k-1 candidate itemset by join operation on utility list data structure. The join operation is costly and consumes more resources. Estimated Utility Co-occurence Structure (EUCS) is proposed to maintain the utility of 2-candidate itemset. EUCS reduces the number of join operation performed to find HUI. The utility list for extension are constructed for every extension of candidate itemset. The HUI are generated by using a recursive procedure.

An another extension of HUI-Miner is proposed by Krishnamoorthy & S (2015)[7] , it partitions the utililty list in k- partitions and k is decided by the user. The partition utility list structure maintains the utility information as in HUI miner but partitioned into k lists. A partitioned based pruning and pruning for the next candidate itemset is implemented. The results of the miner depends on the selection of K.

Fournier-Viger et al. (2016)[8][9] proposed an algorithm for mining Periodic High Utility Itemset (PHUI) called PHM. It uses list data structure to record the TID, utility and remaining utility. Average periodicity a new measure is introduced to enhance the results of periodic mining. New pruning strategies to abonden the candidate itemset are introduced. The construction procedure to construct candidate itemset from previous candidate itemset is based on the construct procedure of HUI-Miner algorithm.

Due to the development of detecting advancements, collecting object moving data has twisted out to be quicker or less demanding. Cyclicity could be a typical marvel in moving items and developments adjust cyclicity. For instance, people move to work in weekdays, and creatures move from one place to an alternate showing a specific cyclicity. Periodic patterns uncover repetitive exercises at consistent time interims for a specific place. Mining spatio-worldly periodic patterns has pulled in consideration as of late (Li et al. 2012)[10]. There's a strong interest to dissect expansive spatio-fleeting mechanical wonder data to look out covered up and profitable periodic patterns to know the practices of moving items (Cao et al. 2007)[11].

Antiquated Periodic Pattern Mining (PPM) calculations for occasion/image successions can't be specifically connected to seek out periodic patterns to spatio-worldly directions in view of the particular qualities of spatio-transient directions: the vulnerability of spatio-fleeting directions and in addition locational (spatial) vagary and worldly vagary, gradable nature of spatio-worldly marvels, and anomalies of your chance interims (Zhang et al. 2015)[12]. To be specific, for the vulnerability of spatial areas (x, y), it's to a great degree unrealistic that partner protest visits decisively a similar area each time in each sum. An approach to comprehend this disadvantage is to utilize pack system to trade the exact areas of the articles by bunched reference spots (Li et al. 2011)[10]. In this way, as divergent to in PPM for time-arrangement data, finding interesting spots for spatio-transient periodic patterns is of decent significance in spatio-fleeting PPM. In this exploration work, it has been anticipated that a calculation to mine periodic high utility itemset by utilizing pruning systems to viably prune the hunt space. In such a large number of uses like cross-showcasing in retail locations, online web based business organization, site click stream investigation and articulating the essential example in biomedical applications have been generally utilized in high utility mining. The incessant thing set mining distinguishes a lot of continuous thing. In data mining, from database framework it has been comprehended that more prominent the high utility thing in the calculation create the more has been the preparing. Consequently, it devours the execution of the mining errand to diminish significantly while managing thick, complex database.

III. PROBLEM STATEMENT

The basic terms used in the HUIM are defined below using the standard conventions as in the previous work (Ahmed *et al.* 2009[13]; Liu &Qu 2012[14]; Liu *et al.* 2005[15]; Yao & Hamilton 2006[16], Lin *et al.* 2015, 2016[17,18]; V.Kavitha et al.(2016)[19][20]).

Let C = {$c_1, c_2, c_3, \ldots c_n$) be set of distinct commodities or items in the dataset D. A transaction in a dataset can be defined as set of commodities that belongs to C given by T_i={c_n| n=1,2,3…N, $c_k \in C$} where N is the number of commodities that participate in the T_i transaction. A dataset D is defined as the set of transaction that are present D ={$T_1, T_2, \ldots T_m$} where the total number of transactions in dataset D is given by m.

Let D be a small transaction dataset and the transaction history of D is given in Table 1.

Definition 1.

For each item $c_i \in C$ is associated with a profit value / external utility value referred as P(c_i). From Table 1, the P(A)=5

Definition 2.

For each item $c_i \in C$ is associated with a purchase quantity / internal utility value referred as I(c_i, T_i). From Table 1, the I(A, T_1)=1

Definition 3.Utility/Value of an item in a transaction.

The item c_i utility for a given transaction in dataset is calculated by using two factors profit and quantity. The utility of an item is denoted by V(c_i)

$$V(c_i, T_{id}) = P(c_i) * Q(c_i, T_{id}) \quad (1)$$

For the example in table 1 the utility of item B in transaction T_3 is calculated as V(B, T_3) = 10

Table.1 Transaction Database for PHUIM - ABC

T_{id}	Transaction	Profit / Utility (P)	Transaction Utility (TU)
T_1	A1,C1	5,1	6
T_2	E1	3	3
T_3	A1,B5,C1,D3,E1	5,10,1,6,3	25
T_4	B4,C3,D3,E1	8,3,6,3	20
T_5	A1,C1,D1	5,1,2	8
T_6	A2,C6,E2	10,6,6	22
T_7	B2,C2,E1	4,2,3	9

Definition 4.Utility/Value of an itemset in a transaction.

The utility of an itemsetco_k in a transaction is calculated as sum of the utility of each item in the itemset in a transaction, provided if all the item in the itemset participates in the transaction. The utility of an itemset is denoted by V(co_k, T_{id})

$$V(co_k, T_{id}) = \sum_{c_i \in co_k \wedge co_k \subseteq T_{id}} V(c_i, T_{id}) \quad (2)$$

For the example in table 1 the utility of item B in transaction T_3 is calculated as

V(B, T_3) = 10

Definition 5.Utility/Value of an itemset in a dataset D.

The utility of an itemsetco_k in a dataset is calculated as sum of the utility of each item in the itemset in all the transaction in the dataset, provided if all the item in the itemset participates in the transaction. The utility of an itemset is denoted by V(co_k)

$$V(co_k) = \sum_{co_k \subseteq T_{id} \wedge T_{id} \subseteq D} V((co_k, T_{id}) \quad (3)$$

For the example in table 1 the utility of item B in transaction D is calculated as V(B) = 22

Definition 6.High Transaction Weighed Utility/Value of an itemset in a dataset D

An itemsetco_k is said to be in HTWUI if the sum of the transaction utility of itemset in all the transaction is greater than or equal to the user specified threshold (Mthreshold). The HTWUI can be represented as follows

$$HTWUI((co_k) \leftarrow \{(co_k \mid \sum_{(co_k \subseteq T_{id} \wedge T_{id} \in D} TU(co_k, T_{id}) \geq Mthreshold\} \quad (4)$$

For the given example in table 1, let the user specified minimum threshold mthreshold = 24. For the itemset AC in D, the utility of itemset AC is calculated as

TU(AC,D) = TU(T_1) + TU (T_3) + TU (T_5) + TU (T_6)

TU(AC,D) = 6 + 25 + 8 + 22 = 61

TU(AC,D) ≥ Mthreshold i.e., 61 ≥ 25, Hence the itemset AC in D is a HTWUI.

Definition 7. High Utility Itemset (HUI) in a Dataset D

An itemsetco_k is said to be a HUI if the sum of the utility of itemset in all the transaction is greater than or equal to the user specified threshold (Mthreshold). The HUI can be represented as follows

$$HUI((co_k) \leftarrow \{(co_k \mid \sum_{(co_k \subseteq T_{id} \wedge T_{id} \in D} V((co_k, T_{id}) \geq mthreshold\} \quad (5)$$

For the given example in table 1, let the user specified minimum threshold mthreshold = 24. For the itemset AC in D, the utility of itemset AC is calculated as

V(AC,D) = V(AC,T_1) + V(AC,T_3) + V(AC,T_5) + V(AC,T_6)

V(AC,D) = 6 + 15 + 6 + 16 = 43

V(AC,D) ≥ Mthreshold i.e., 43 ≥ 25, Hence the itemset AC in D is a HUI.

Definition 8.Transaction Weighted Utility (TWU) of an itemset in Dataset D

Transaction Weighted Utility (TWU) of an itemsetco_k in Dataset D can be defined as the sum of the transaction utility of the itemsetco_k in the all the transactions of the transaction database D.

$$TWU(co_k) = \sum_{co_k \subseteq T_{id} \wedge T_{id} \in D} TU(T_{id}) \quad (6)$$

TWU (AC) = 6+25+8+22 = 61

Based on the behaviour of honey for collecting the nectar, Artificial Bee Colony algorithm proposed. In this model, there are three (3.2)

components, the food sources, employed and unemployed bees. The unemployed bee are the one which find out the food sources. They are classified into two categories, scout bees, and onlooker bees. Scout bee goes in search to find the food sources. After watching the dance of the employed bee, the onlooker bee decides the food source based on the amount of the food available.

Employed bees are the one which goes to the food position and the information regarding the food position and the amount of nectar is shared with the onlooker bee. When the nectar in the food source is exhausted it becomes scout bee and goes in search of a new food source. The number of employed bee will be equal to the number of food sources available.

IV. PROPOSED SYSTEM

A. Pre-processing

The Transaction Weighted Utility (TWU) (Liu *et al.* 2012)[14] model is used in the pre-processing phase of the proposed PHIM-ABC algorithm. First the transaction utility (tu) is found by adding the utility value of all items present in the transaction. This tu is calculated for every transaction of the dataset. At the same time during the initial scan of the dataset TWU of each item X is calculated by adding tu of all the transaction in which the item X participates. The TWU(X) gives the 1-High Transaction Weighted Utility Itemset (1-HTWUI).

Based on the TWU pruning property the item whose TWU is less than the mthresholdi.e non-promising item are removed from the transaction. If any empty transactions are there because of removing a non-promising item, the empty transactions are also deleted. 1-HTWUI satisfies the transaction weighed downward closure property (TWDC). The TWDC states that if an element is not an promising item than all the item which add to non-promising item to form an itemset will not be HUI. Thus the search space and the number of candidate set generated can be effectively reduced by using the TWDC property. The 1-HTWUI gives the upper bound on the utility of the itemset.

The transactions in the database are arranged according the TWU of the first item to enable effective exploration of PHUI, based on the fact that if a item has larger value of TWU, then the probability of being a PHUI is high.

PHUIM Set

The itemset of the PHUIM is associated with the minPeriod, maxPeriod, avgPeriod, fitness value.

Fitness function

Fitness function is calculated using utility value of each itemset. Fitness value (X)=V(c_i,T_{id}) which is same as defined in the definition 3. The fitness value of the itemset is given as the nectar amount of the employed bee.

Fitness value (X) = V(ci,Tid) = P(ci) * Q(ci,Tid)

B. Evaluation Phase

Evaluation of fitness function

The onlooker bee evaluates the fitness function in order to decide the food source, the decidability is done based on the probability function given by the equation below

$$Prob(x) = \frac{f(x)}{\sum_{n=1}^{SN} f(x)} \qquad (7)$$

Where f(X) is the value of fitness or amount of nectar given by the employed bee and the number of food sources, Source Node known to the bee is given by the SN.

C. PHUIM Discovery

The steps in the artificial bee intelligence algorithm are given below:

Step 1: initialize the fitness function and find the transaction utility of all items present in D.

Step 2: Find the number of employed bees and put the found employed bees in the food source.

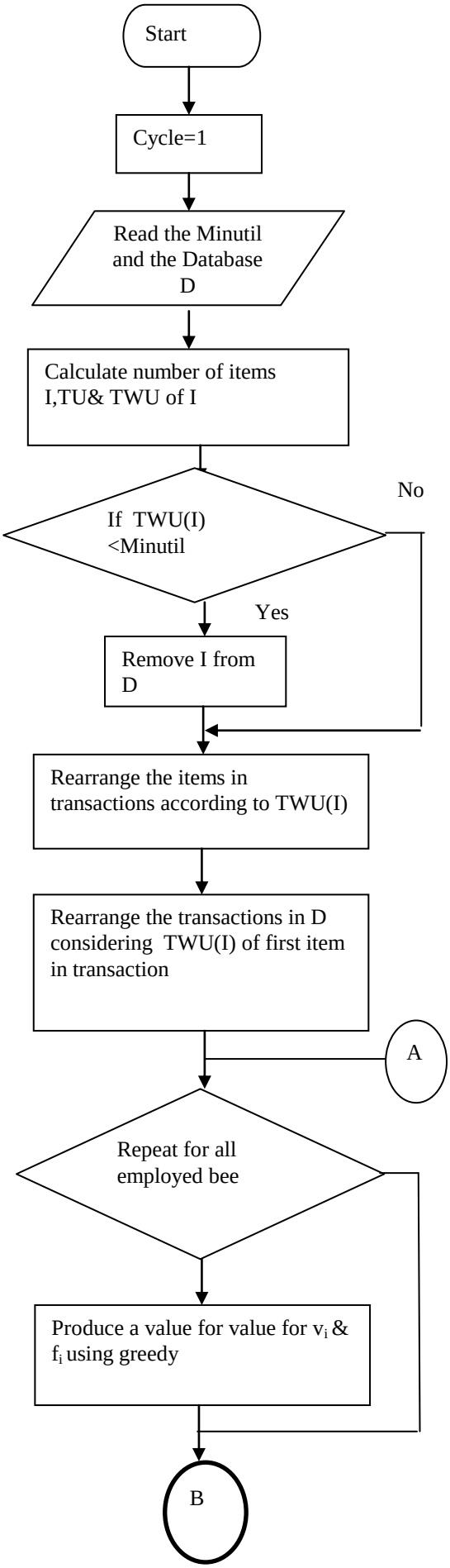
Step 3: Also place the onlooker bee on the food source in the memory

Step 4: Discover the new food sources by searching using the scout bees.

Step 5: Repeat from step 2 until the termination condition is met.

In PHUIM discovery based on ABC algorithm, the fitness function is evaluated for each distinct item found in the transaction database D. The transaction utility for each item is also found during the initial step. The performance of the ABC algorithm depends upon the number of employed bee and the number of cycle the algorithm to be repeated. Decide the number of employed bee and find the fitness value of the each employed bee. The onlooker bee evaluates the fitness value of the employed bee and checks with previous value. If the found value is better than the previous value it updates with the new value and forgets the previous. The onlooker bee will remember only the latest fitness value. Then the found food source is checked for the satisfying criteria for the PHUI. If satisfied the found food source is added to the existing set of PHUI. The process is repeated for the defined number of cycles.

The process of finding PHUI is explained in the figure 1. First the non promising item are removed based on the HTWUI. The items in every transaction is rearranged according to TWU. Then the transactions are rearranged according the TWU of the first item in the transaction. Select the number of employed bee. Calculate the fitness function. The number of onlooker bee will be equal to the number of 1-HTWUI. From the fitness function and the probability the onlooker bee selects the best fitness function. The best item is added to the PHUI if the condition is met. The cycle is repeated for the selected number of times.


Start
Cycle=1
Read the Minutil and the Database D
Calculate number of items I,TU& TWU of I
If TWU(I) <Minutil
No
Yes
Remove I from D
Rearrange the items in transactions according to TWU(I)
Rearrange the transactions in D considering TWU(I) of first item in transaction
A
Repeat for all employed bee
Produce a value for value for v_i & f_i using greedy
B

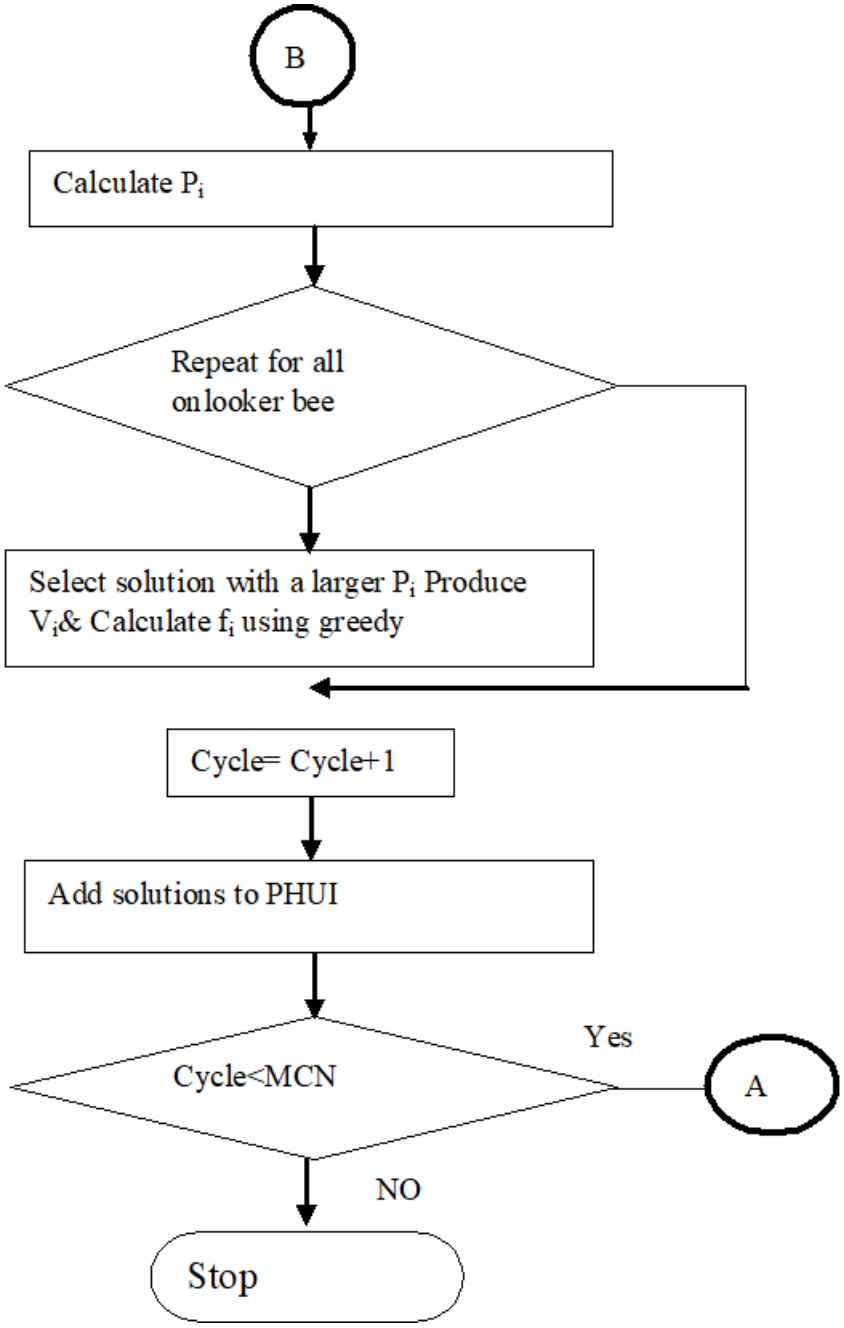


Figure 1: Flow chart of the process

A. ***Algorithm: PHUIM-ABC***

Input: A transaction dataset D, Profit table P, Minimum Utility Threshold Mthreshold

Output: PHUIs, a set of high-utility itemsets.

for every Ti ∈ D do
for every Cij⊆Tq do
TU(Tq) = I(Cij , Ti) × ptable(Cij);
calculate TWU(Cij) = Cij⊆Tq and TU(Tq);
find 1-HTWUIs← { Cij | TWU(Cij) ≥ TU ×δ};
set k = |1-HTWUIs|
set number of employed bee to M;
for i ← 1 to M do
for j ← 1 to k do
initialize pj
calculate the fitness value as f(i)

```
        if fitness(pi (t)) ≥  TU × δ  then
         PHUI s ← (pi (t)) ∪ PHUI s
        find a PHUI
        find best (employed bee) of each M employed
    while cycle count not reached do
    for i ← 1, M do
        update the prob(t + 1) probabilities of  employed bee
        if fitness(pi (t + 1)) ≥  TU × δ  then
            PHUI s ←GetItem(pi (t + 1)) ∪ PHUIs
        find a PHUI
        find best (employed bee) of each M employed
        set t ← t + 1;
    return PHUIs;
```

V RESULTS

All the algorithms are implemented in eclipse IDE with Java API. A core i3 processor with 1 GB is used for the evaluation of algorithm. Three real life datasets retail, foodmart, and mushroom are used to evaluate the performance of the proposed PHUIM-ABC algorithm. Retail dataset is sourced from UCI repository with 541909 transactions. Retail has multivariate integer attributes. It is a sparse dataset. Foodmart is sourced from Microsoft foodmart database. It has 86829 point of sale transaction and around one thousand distinct items participate in the transaction. The mushroom dataset is a drawn from North America in Audubon Society Field Guide, it is a dense dataset. It is a multivariate dataset, which 8125 records with 100 distinct elements. The retail database is a sparse database which contains real data set from the anonymous Belgian retail store, connect is a dense database which has connection data between online users and Accidents is a dense database which contains accident data are obtained from FIMI Repository.

Execution time and memory performance are the metrics used to measure the proposed algorithm performance. To measure the performance the PHUIM-ABC is compared with the PHM miner. The proposed algorithm is compared with the state of art of algorithm HUI miner. The number of candidate set generated by PHUIM-ABC algorithm is half of the candidate set generated by PHM miner. For the three datasets retail, foodmart and mushroom, the TWU based upper bound reduces the candidate set generation by 20%.

From Figure 2, Figure 3 and Figure 4 it can be noted that the PHUIM-ABC performs better than the PHM miner. For all the minimum utility values in mushroom dataset the PHUIM-ABC performs two times better than the PHM. For the foodmart dataset the PHUIM-ABC algorithm performance in terms of running time is improved by five times compared to the PHM algorithm. This is because the foodmart is a dense dataset, the number of candidate set searched is much better reduced using the proposed algorithm. For the chainstore dataset the performance is given in Figure 6. For the chainstore dataset also the performance of the PHUIM-ABC is improved compared to the PHM. The performance is much better for lower minimum utility value. From Figure 6 it can be noted that there is only a slight improvement in the order of 10 % compared to HUI miner.

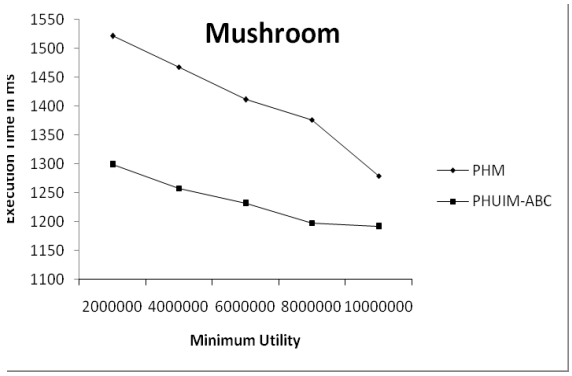


Figure 2 Execution time of mushroom dataset

From the Figure 2 it may be not he that the PHUIMABC algorithms runs 20% faster than PHUIM-ABC algorithm for minimum utility of 2000000 and for the minimum utility there is performance improvement of 10%.

The memory consumption of mushroom dataset is shown in Figure 3For minimum utility memory usage is around 90KB for PHM and around 65 KB for PHUIM-ABC. The overall memory usage of PHUIM-ABC for mushroom dataset is reduced around 30% compared to PHM.

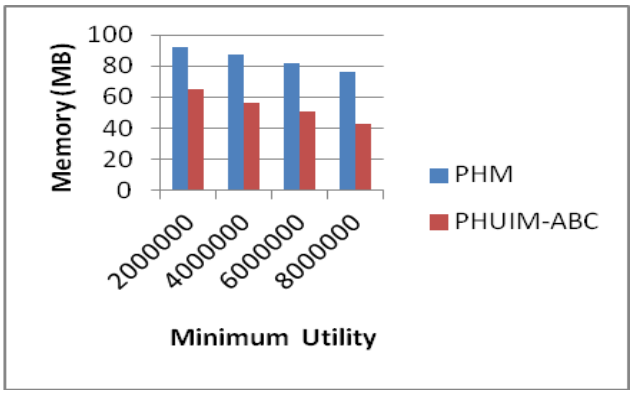


Figure 3 Memory usage of mushroom dataset

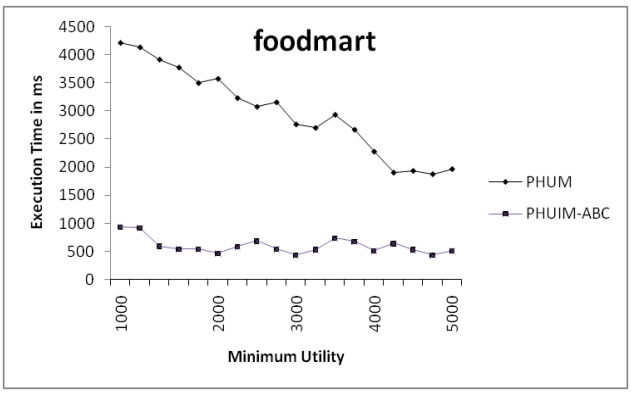


Figure 4Execution time of foodmart dataset

For the thefoodmart dataset shown in Figure 4 PHUIM-ABC shows a good performance improvement. For all the minimum utility values ranging from 1000 to 5000 the execution time is less than 1000 ms and for PHM it is greater than 2000. PHUIM-ABC is atleast 2 times faster than PHM algorithm.

The memory consumption of PHM and PHUIM-ABC is given in Figure 5. The memory consumption for PHM forminimum utility range 1000 to 4000 is in the range of 600MB and for PHUIM-ABC is in the range of 800KB.

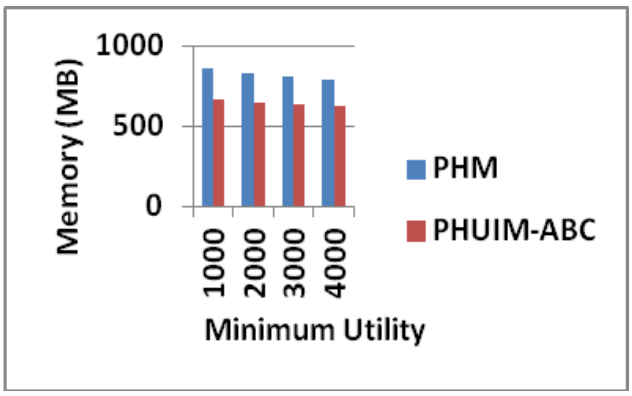


Figure 5 Memory usage of foodmart dataset

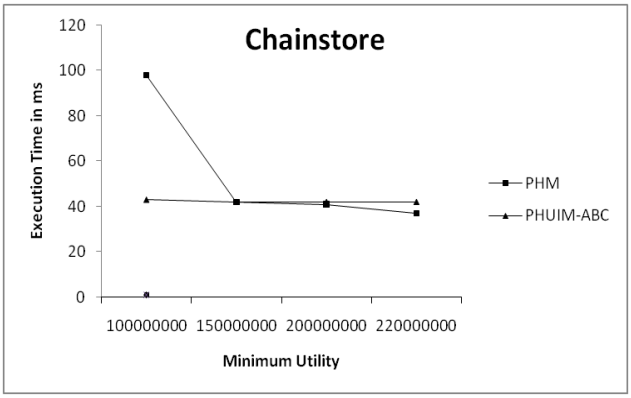


Figure 6 Execution time of chain-store dataset

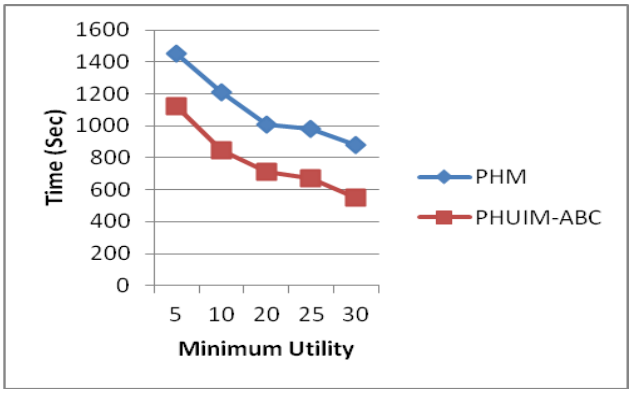


Figure 7 Execution time of accidents dataset

For the accident dataset the execution time for PHM and PHUIM-ABC is shown in figure 3.6 for minimum utility values from 5 to 30. Both the algorithms PHM and PHUIM_ABC execution time linearly decreases with increase in minimum utility value. The PHUIM-ABC shows performance improvement of 20% compared to PHM.

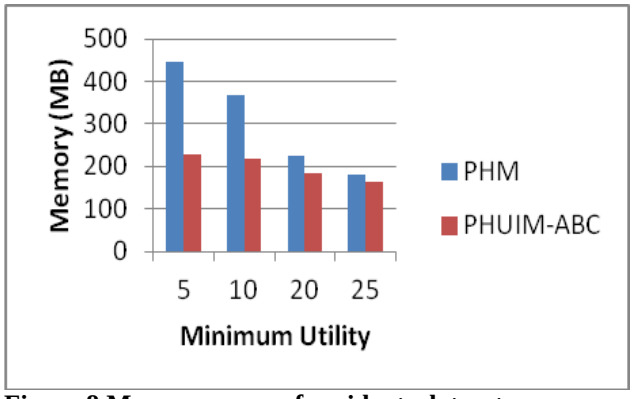


Figure 8 Memory usage of accidents dataset

For the accident dataset the memory usage for PHM and PHUIM-ABC is shown in Figure 8 for minimum utility values from 5 to 25. For lower value of minimum utility 5 PHUIM-ABC takes only 50% of memory of PHM. For higher value of minimum utility PHUIM-ABC memory usage is almost same of PHM shows only a little improvement.

For the connect dataset the execution time for PHM and PHUIM-ABC is shown in Figure 9 for minimum utility values from 28 to 40. Both the algorithms PHM and PHUIM_ABC execution time linearly decreases with increase in minimum utility value. The PHUIM-ABC shows performance improvement of when the utility value is 28, 30 and 32. From 34 to 40 execution time of PHM and PHUIM-ABC are almost same.

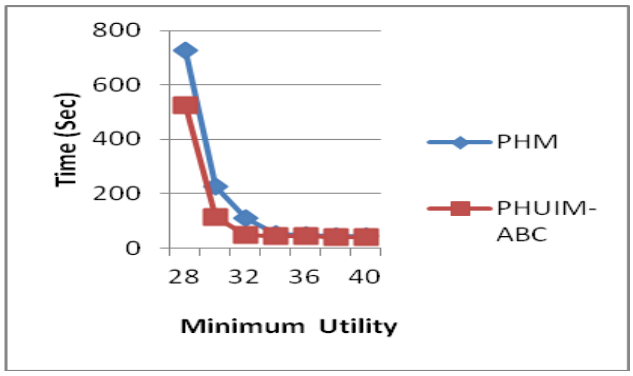


Figure 9 Execution time of connect dataset

For the connect dataset the memory usage for PHM and PHUIM-ABC is shown in Figure 8 for minimum utility values from 28 to 40. For lower value of minimum utility 28 PHUIM-ABC takes only 50% of memory of PHM. For higher value of minimum utility 40 PHUIM-ABC memory usage is PHM shows around 20% improvement

For the retail dataset the execution time for PHM and PHUIM-ABC is shown in Figure 3.10 for minimum utility values from 10000 to 500000. The execution time of both the algorithms PHM and PHUIM_ABC is high for low minimum utility value. The PHUIM-ABC shows performance improvement 10% in execution time compared to PHUIM

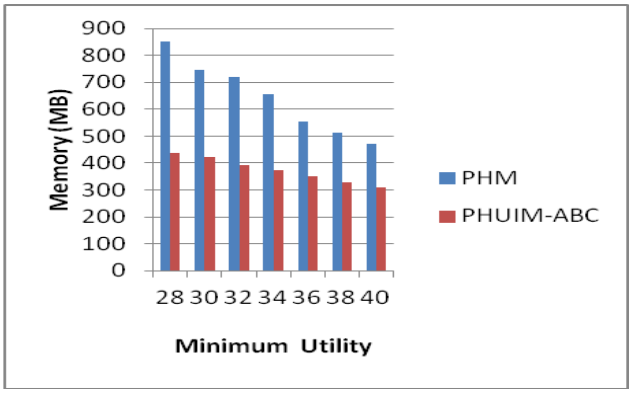


Figure 10Memory usage of connect dataset

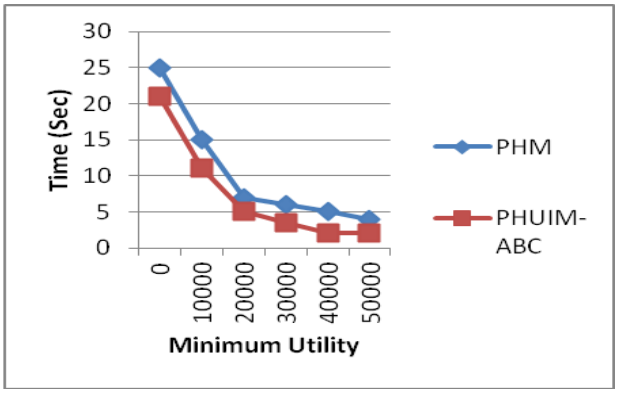


Figure 11 Execution time of retail dataset

For the connect dataset the memory usage for PHM and PHUIM-ABC is shown in Figure 12 for minimum utility values from 10000 to 500000. The PHUIM-ABC takes only 50% of memory of PHM. For higher value of minimum utility PHUIM-ABC memory usage is 2 times less than PHUIM-ABC.

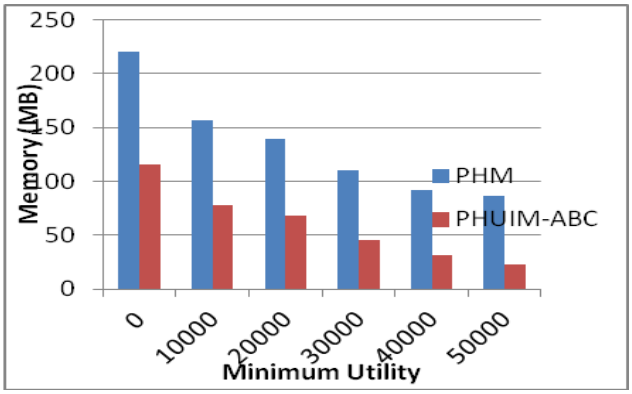


Figure 12 Memory usage of retail dataset

Dataset	Minimum Utility	Best %	Average %	Worst %
Food mart	1000	88.89	83.25	81.72
	2000	90.28	85.36	83.55
	3000	92.78	89.26	87.24
	4000	94.71	94.49	88.75
	5000	97.75	94.82	92.51
Mushroom	2000000	92.32	90.78	88.77
	4000000	98.21	96.54	89.54
	6000000	99.10	97.32	93.54
	8000000	100	100	98.79
	10000000	100	100	100
Chainstore	10000000	88.57	83.29	80.53
	10000000	90.63	84.92	82.51
	15000000	92.47	91.61	85.61
	20000000	95.96	93.81	86.98
	22000000	96.48	94.27	89.81
	10000000	97.56	95.81	90.51
Accidents	5	90.75	85.52	85.52
	10	91.53	87.93	86.85
	20	93.78	92.36	88.11
	25	97.78	95.89	89.97
Connect	28	97.75	95.62	92.25
	30	98.95	96.42	448
	32	100	100	100
	34	100	100	100
	36	100	100	100
	38	100	100	100
	40	100	100	100

Table 2. Percentage of discovered PHUI

From the table 2 it may be noted that, for the connect database all PHUI are identified for all the minimum utility is perfect except for minimum utility 28 and 30. Accidents, chainstore, foodmart and mushroom have more than 80 % accuracy for all the minimum utility value tested.

VI CONCLUSION

From the experimental results, it clearly shows the ABC based algorithms performs better than the state of art PHUIM algorithms. High utility itemset mining based on artificial bee colony algorithm is proposed. The proposed PHUIM-ABC algorithm mines 50% faster than the state-of-art algorithms. The candidate itemset generated by the proposed PHUIM-ABC has 97% correct value compared to the FHM algorithm. For mushroom data set it performs 50% faster for all the values of minimum utility threshold. The memory usage is reduced by a minimum of 40% for the mushroom, connect and retail datasets. For accident and foodmart the memory usage is reduced by minimum of 20%. Since retail industries need a very fast output than the exact outputs; this algorithm will be best suited. The correctness of the algorithm is above 80 % for all the dataset and for all the minimum threshold value. In the future it is planned to implement genetic algorithms for PHUIM to improve efficiency. This algorithm can be also extended to sequential utility mining algorithms.

REFERENCES

1. Han, Jiawei, Jian Pei, and Yiwen Yin. "Mining frequent patterns without candidate generation." In ACM sigmod record, vol. 29, no. 2, pp. 1-12. ACM, 2000.
2. Lan, Guo-Cheng, Tzung-Pei Hong, and Vincent S. Tseng. "An efficient projection-based indexing approach for mining high utility itemsets." Knowledge and information systems 38, no. 1 (2014): 85-107.
3. Liu, Ying, Wei-keng Liao, and AlokChoudhary. "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, 2005.
4. Tseng, Vincent S., Cheng-Wei Wu, Bai-En Shie, and Philip S. Yu. "UP-Growth: an efficient algorithm for high utility itemset mining." In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 253-262. ACM, 2010.

5. Liu, Mengchi, and Junfeng Qu. "Mining high utility itemsets without candidate generation." In Proceedings of the 21st ACM international conference on Information and knowledge management, pp. 55-64. ACM, 2012.
6. Fournier-Viger, Philippe, Cheng-Wei Wu, SouleymaneZida, and Vincent S. Tseng. "FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning." In International symposium on methodologies for intelligent systems, pp. 83-92. Springer, Cham, 2014.
7. Krishnamoorthy, Srikumar. "Pruning strategies for mining high utility itemsets." Expert Systems with Applications 42, no. 5 (2015): 2371-2381.
8. Fournier-Viger, Philippe, Jerry Chun-Wei Lin, Tai Dinh, and HoaiBac Le. "Mining correlated high-utility itemsets using the bond measure." In International Conference on Hybrid Artificial Intelligence Systems, pp. 53-65. Springer, Cham, 2016.
9. Fournier-Viger, Philippe, Jerry Chun-Wei Lin, Quang-Huy Duong, and Thu-Lan Dam. "PHM: mining periodic high-utility itemsets." In Industrial conference on data mining, pp. 64-79. Springer, Cham, 2016.
10. Li, Z., Han, J., Ding, B. and Kays, R., "Mining periodic behaviors of object movements for animal and biological sustainability studies", Data Mining and Knowledge Discovery, 24(2), pp.355-386,2012.
11. Cao, H., Mamoulis, N. and Cheung, D.W., "Discovery of periodic patterns in spatiotemporal sequences. IEEE Transactions on Knowledge and Data Engineering", 19(4), pp.453-467, 2007.
12. Zhang, Dongzhi, Kyungmi Lee, and Ickjai Lee. "Periodic pattern mining for spatio-temporal trajectories: a survey." In 2015 10th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 306-313. IEEE, 2015.
13. Ahmed, ChowdhuryFarhan, Syed KhairuzzamanTanbeer, Byeong-SooJeong, and Young-Koo Lee. "Efficient tree structures for high utility pattern mining in incremental databases." IEEE Transactions on Knowledge and Data Engineering 21, no. 12, 1708-1721, 2009.
14. Liu, Mengchi, and Junfeng Qu. "Mining high utility itemsets without candidate generation." In Proceedings of the 21st ACM international conference on Information and knowledge management, pp. 55-64. ACM, 2012.
15. Liu, Ying, Wei-keng Liao, and AlokChoudhary. "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, 2005.
16. Yao, Hong, and Howard J. Hamilton. "Mining itemset utilities from transaction databases." Data & Knowledge Engineering 59, no. 3 , 603-626, 2006.
17. Lin, Jerry Chun-Wei, WenshengGan, Tzung-Pei Hong, and Binbin Zhang. "An incremental high-utility mining algorithm with transaction insertion." The Scientific World Journal 2015.
18. Lin, Jerry Chun-Wei, WenshengGan, Philippe Fournier-Viger, Tzung-Pei Hong, and Vincent S. Tseng. "Efficient algorithms for mining high-utility itemsets in uncertain databases." Knowledge-Based Systems 96,171-187,2016.
19. Kavitha, V., and B. G. Geetha. "High Utility Itemset Mining with Influential Cross Selling Items from Transactional Database." Int J AdvEngg Tech/Vol. VII/Issue II/April-June 820-826-2016.
20. Kavitha, V. and Geetha, B.G., "Review on high utility itemset mining algorithms." In 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave) (pp. 1-5). IEEE 2016.

AUTHORS PROFILE

S.Viveka, Working as Assistant Professor, Department of Information Technology, Velalar College of Engineering and Technology, Tamil Nadu India .

Dr.B.Kalaavathi, Working as Professor and Head in the Department of Computer Science and Engineering in K.S.R. Institute for Engineering and Technology, Tiruchengode, India. Published more than 62 paper is international journals and guide Phd Scholars.