



Selective Database Projections Based Approach for Mining High-Utility Itemsets

¹A V Murali Krishna, ²Dhanyamraju Anantha Prathyusha, ³Sandineni Divya Sree, ⁴Nagarkunta
Karishma

¹Assistant

Professor, ²Student, ³Student, ⁴Student

^{1,2,3,4}Department of Computer Science and
Engineering, ^{1,2,3,4}Matrusri Engineering
College, Hyderabad, India.

Abstract: High-utility itemset mining (HUIM) is an emerging area of data mining and is widely used. HUIM differs from the frequent itemset mining (FIM), as the latter considers only the frequency factor, whereas the former has been designed to address both quantity and profit factors to reveal the most profitable products. The challenges of generating the HUI include exponential complexity in both time and space. Moreover, the pruning techniques of reducing the search space, which are available in FIM because of their monotonic and anti-monotonic properties, cannot be used in HUIM. In this paper, we propose a novel selective database projection-based HUI mining algorithm (SPHUI-Miner). We introduce an efficient data format, named HUI-RTPL, which is an optimum and compact representation of data requiring low memory. We also propose two novel data structures, viz, Selective database projection utility list and Tail-Count list to prune the search space for HUI mining. Selective projections of the database reduce the scanning time of the database making our proposed approach more efficient. It creates unique data instances and new projections for data having fewer dimensions thereby resulting in faster HUI mining. We also prove upper bounds on the amount of memory consumed by these projections. Experimental comparisons on various benchmark data sets show that the SPHUI-Miner algorithm outperforms the state-of-the-art algorithms in terms of computation time, memory usage, scalability, and candidate generation.

Index Terms – Itemsets, Frequent itemset mining, High-utility itemset mining, SPHUI-Miner.

I. INTRODUCTION

Frequent itemset mining (FIM) is one of the fundamentals of data mining and has many real-life applications. FIM is mainly used to find concomitantly occurring items in the transactions. FIM techniques depend on a support confidence framework where the frequency of items should not be less than the minimum support threshold. The high utility itemsets rarely appear but have high utility values and are often ignored in the FIM algorithms as these consider only occurrence frequencies of itemsets. Therefore, an important limitation of FIM is its assumption that gives equal importance to all items irrespective of their value to the organization. Generally, these assumptions do not hold in real-world applications. For example, bread is purchased in hundreds or thousands per day while fewer diamonds are bought in a week or month. The former has a higher frequency but lower profit value while the latter has a lower frequency with higher profit value for retailers. FIM mining discovers frequent itemsets generating a low profit while it fails to discover the less frequent itemsets that generate a high profit. To solve the limitation of FIM or association rule mining (ARM), high-utility itemset mining (HUIM) was designed to discover the useful and profitable itemsets from the quantitative databases. These databases contain utility values of itemsets. An itemset is considered a high utility itemset (HUI) if its utility value is no less than the user-specific, minimum utility threshold. In real-world practice, the utility of an itemset can be measured by several factors, such as weight, profit, or cost, which can be defined via the user's preferences. HUI can help managers to carry out accurate financial analyses and make informed decisions. HUIM is used in a wide range of applications such as website clickstream analysis, mobile computing, top-k HUI mining, and biomedical applications. HUIM also motivates and inspires several data mining tasks such as high-utility sequential pattern mining, high average utility itemset mining, and high utility stream mining.

II. LITERATURE SURVEY

2.1 Staged approach for grammatical gender identification of nouns using association rule mining and classification:

In some languages, gender is a grammatical property of the noun. Grammatical gender identification enhances the machine translation of such languages. This paper reports a three-staged approach for grammatical gender identification that makes use of the word and morphological features only. A Morphological Analyzer is used to extract the morphological features. In stage one; association rule mining is used to obtain grammatical gender identification rules. Classification is used in the second stage to identify grammatical gender for nouns that are not covered by grammatical gender identification rules obtained in stage one. The third stage combines the results of the two stages to identify gender. The staged approach has better precision, recall, and F-Score compared to machine learning classifiers used on complete data sets. The approach was tested on Konkani nouns extracted from the Konkani WordNet and an F-Score of 0.84 was obtained.

The grammatical gender of a noun in Konkani can be determined based on morphological and word features using a staged approach, with a weighted average F-Score of 0.84 thus supporting our hypothesis. On examination of grammatical gender identification rules, we found that some morphological paradigms suggest grammatical gender. For some words like lawyer two paradigms get assigned suggesting that the noun has non-neuter grammatical gender. Some association rules example Begin Vowel = NULL End Vowel = o \rightarrow P ID = two zero Gender = masculine acc :(0.99315) generated can also be used to automatically assign morphological paradigms to new noun entries based on word features. Future work will be focused on finding a word, morphological, or context features that give more clues to determine masculine and neuter gender.

2.2 Frequent Itemset-based Hierarchical Document Clustering using Wikipedia as External Knowledge:

High dimensionality is a major challenge in document clustering. Some of the recent algorithms address this problem by using frequent itemsets for clustering. But, most of these algorithms neglect the semantic relationship between the words. On the other hand, some algorithms take care of the semantic relations between the words by making use of external knowledge contained in WordNet, Mesh, Wikipedia, etc but do not handle the high dimensionality. In this paper, we present an efficient solution that addresses both these problems. We propose a hierarchical clustering algorithm using closed frequent itemsets that use Wikipedia as external knowledge to enhance the document representation. We evaluate our methods based on F-Score on standard datasets and show our results to be better than existing approaches.

In this paper, we presented a hierarchical algorithm to cluster documents using frequent itemsets and Wikipedia as background knowledge. We used generalized closed frequent itemsets to construct the initial clusters. Then we proposed two methods using, 1) TF-IDF and 2) Wikipedia as external knowledge, to remove the document duplication and construct the final clusters. In addition to these, we proposed a method for labeling clusters. We evaluated our approaches on five standard datasets and found that our results are better than the current state-of-the-art methods. In the future, these ideas can be extended to other areas like evolutionary clustering, data streams, etc by using incremental frequent itemsets.

2.3 GenMiner: Mining nonredundant association rules from integrated gene expression data and annotations:

GenMiner is an implementation of association rule discovery dedicated to the analysis of genomic data. It allows the analysis of datasets integrating multiple sources of biological data represented as both discrete values, such as gene annotations, and continuous values, such as gene expression measures. GenMiner implements the new NorDi algorithm for normalizing and discretizing continuous values and takes advantage of the Close algorithm to efficiently generate minimal non-redundant association rules. Experiments show that execution time and memory usage of GenMiner are significantly smaller than those of the standard Apriori-based approach, as well as the number of extracted association rules.

The GenMiner ARD approach was developed for processing high dimensional biological datasets integrating both gene expression measures and biological annotations. It was designed considering the specific characteristics of such biological data that are noisy, dense, and highly correlated. It uses a frequent closed itemsets-based approach that allows to discover of low-support, or rare, rules representing associations between very small groups of genes and reduces the number of extracted association rules without information loss. The application of GenMiner to a dataset integrating the well-known Eisen=et al. (1998) gene expression dataset with corresponding gene annotations demonstrated its capability to extract meaningful associations between gene expression profiles and annotations. Furthermore, it showed the potential of this approach to integrate several heterogeneous sources of information. This is only an example of GenMiner's capabilities, as it can easily integrate every kind of gene annotation obtained from any source of biological information. With these features, GenMiner is an ARD approach that is adequate to biologist requirements for the analysis of genomic data.

2.4 Mining frequent patterns without candidate generation: A frequent-pattern tree approach:

Mining frequent patterns in transaction databases, time-series databases, and many other kinds of databases have been studied popularly in data mining research. Most of the previous studies adopt an Apriori-like candidate set generation-and-test approach. However, candidate set generation is still costly, especially when there exist a large number of patterns and/or long patterns. In this study, we propose a novel frequent-pattern tree (FP-tree) structure, which is an extended prefix-tree structure for storing compressed, crucial information about frequent patterns, and develops an efficient FP-tree-based mining method,

FP-growth, for mining the complete set of frequent patterns by pattern fragment growth. The efficiency of mining is achieved with three techniques: (1) a large database is compressed into a condensed, smaller data structure, FP-tree which avoids costly, repeated database scans, (2) our FP-tree-based mining adopts a pattern-fragment growth method to avoid the costly generation of a large number of candidate sets, and (3) a partitioning-based, divide-and-conquer method is used to decompose the mining task into a set of smaller tasks for mining confined patterns in conditional databases, which dramatically reduces the search space. Our performance study shows that the FP-growth method is efficient and scalable for mining both long and short frequent patterns and is about an order of magnitude faster than the Apriori algorithm and also faster than some recently reported new frequent-pattern mining methods.

We have implemented the FP-growth method and studied its performance in comparison with several influential frequent pattern mining algorithms in large databases. Our performance study shows that the method mines both short and long patterns efficiently in large databases, outperforming the current candidate pattern generation-based algorithms. The FP-growth method has also been implemented in the DBMinersystem and has been tested in large industrial databases, such as a retail chain database, with satisfactory performance.

2.5 Mining high utility quantitative association rules:

Mining weighted association rules consider the profits of items in a transaction database, such that the association rules about important items can be discovered. However, high-profit items may not always be high revenue products, since purchased quantities of items would also influence the revenue for the items. This paper considers both profits and purchased quantities of items to calculate the utility of the items. Mining high utility quantitative association rules are to discover that when some items are purchased in some quantities, the other items in some quantities are purchased too, which have high utility. In this paper, we propose a data mining algorithm to find high utility itemset with purchased quantities, from which high utility quantitative association rules also can be generated. Our algorithm needs not generate a candidate itemset and just needs to scan the original database twice. The experimental results show that our algorithm is more efficient than the other algorithms which only discovered high utility association rules.

Mining weighted association rules [2, 4, 6] only considers purchased frequency and profit for an itemset; however, a high-profit item may not always be a high utility item, since the purchased quantities for the itemset also need to be considered. Therefore, this paper defines the utility support of an itemset, which is calculated by weights, purchased quantities, and purchased frequency for the itemset. We also propose an efficient algorithm HUQA to discover high utility quantitative itemset from a transaction database. High utility quantitative association rules can be generated from the high utility quantitative itemset. For the HUQA algorithm, a method for combining adjacent quantities for an item and a method for reducing the number of conditional databases generated by HUQA are proposed. HUQA algorithm does not generate a candidate q-itemset and just needs to scan the original database twice, which is very efficient. For the experiments, we compare our algorithm with the Two-Phase algorithm which is the most efficient algorithm for mining high utility itemset. The experimental results show that our algorithm outperforms the Two-Phase algorithm under the same number of generated high utility (quantitative) itemsets.

III. EXISTING SYSTEM

Traditional HUIM algorithms such as PB, TWU, UP-Growth, and UP-GrowthC carry out mining in two phases. However, the two-phase model suffers from the issues of generating an enormous number of candidates and repeated scanning of the database which makes the algorithms inefficient. The FHM, HUP-Miner, HUI-Miner, EFIM, and d2HUP algorithms mine a complete set of HUI in a single phase without generating candidates. EFIM is the current state-of-the-art algorithm that outperforms all algorithms for HUIM. EFIM proposes database projection on all promising items during the depth-first search and prunes the search space using local and sub-tree utilities. EFIM evaluates the utility of every itemset while exploring the search space of all items. It does this firstly by generating itemsets and then by performing database projection for all promising itemsets. EFIM requires less memory and time as compared to the above-mentioned algorithms. It performs well on dense datasets. However, the method of high-utility itemset mining remains costly in terms of runtime and memory usage because it creates projections on all promising itemsets. There is a scope for improvement by designing more efficient algorithms for this task. In this paper, we address this challenge by providing efficient data representation and selective database projection which improves the pruning technique of the search space.

IV. PROPOSED SYSTEM

To reduce the memory and runtime of the EFIM algorithm, we propose an efficient selective database projection-based HUIM algorithm, called SPHUI-Miner. In this algorithm, we create new database projections of smaller size having fewer dimensions and unique data instances which result in faster HUI mining. We also provide upper bounds on the amount of memory consumed by these projections. The SPU-List structure is introduced to directly prune the search space for an itemset having utility less than a loose upper bound. This avoids traversing down to the unpromising itemsets in the search space. Unlike EFIM, SPHUI-Miner uses a Tail-Count list, which maintains a count of each item in the database projection. The Tail-

Count list is used for applying PEP, which avoids the need for exploration of each itemset. Thus, fewer branches need to be traversed to find the high utility itemsets.

V. TECHNOLOGY STACK

5.1 INTERFACE REQUIREMENTS

5.1.1 User Interface

The user interface of this system is a user-friendly Java Graphical User Interface.

5.1.2 Hardware Interfaces

The interaction between the user and the console is achieved through Java capabilities.

5.1.3 Software Interface

The required software is JAVA 1.6

5.1.4 Operating Environment

Windows XP, Linux.

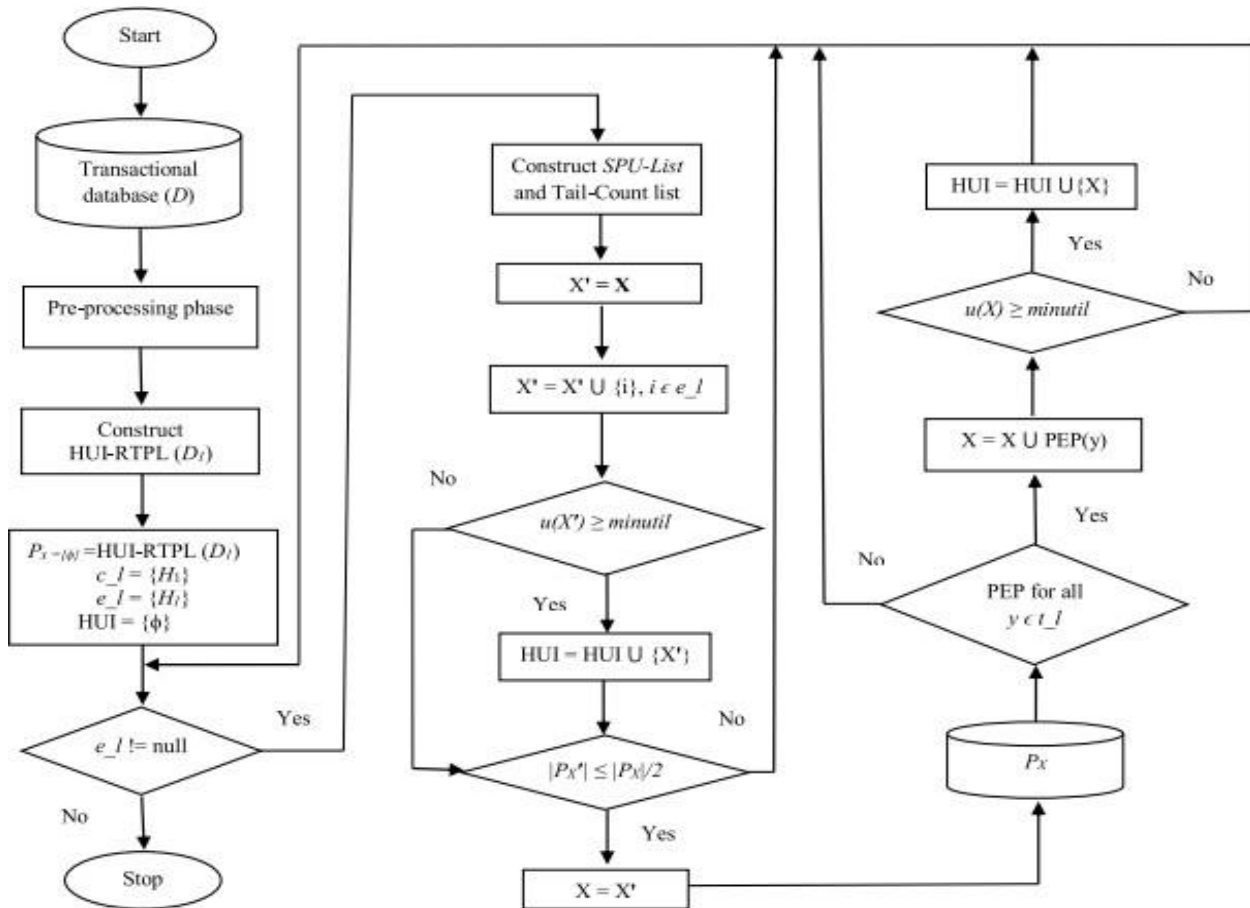
5.2 HARDWARE REQUIREMENTS

| | |
|-------------|---------------------------|
| Processor : | Pentium –IV |
| Speed : | 1.1 GHz |
| RAM : | 256 MB(min) |
| Hard Disk : | 20 GB |
| Key Board : | Standard Windows Keyboard |
| Mouse : | Two or Three Button Mouse |
| Monitor : | SVGA |

5.3 SOFTWARE REQUIREMENTS

| | |
|------------------------|------------|
| Operating System : | Windows XP |
| Programming Language : | Java |

VI. IMPLEMENTATION



System Architecture

Mining high utility items can be beneficial to any product selling company as this will produce product patterns for companies that are high in profit. By seeing this, companies may invest more in such products to increase their revenues. Existing frequent Itemsets help in finding User System products that are high in selling but lacking support of high utility. Later High utility mining algorithms were introduced but they scan databases multiple times to mine high utility patterns which require high memory and computation. To overcome this, Selective Database Projections (estimate or calculation) has been introduced, which will find high utility pattern from the selective database instead of scanning the entire database.

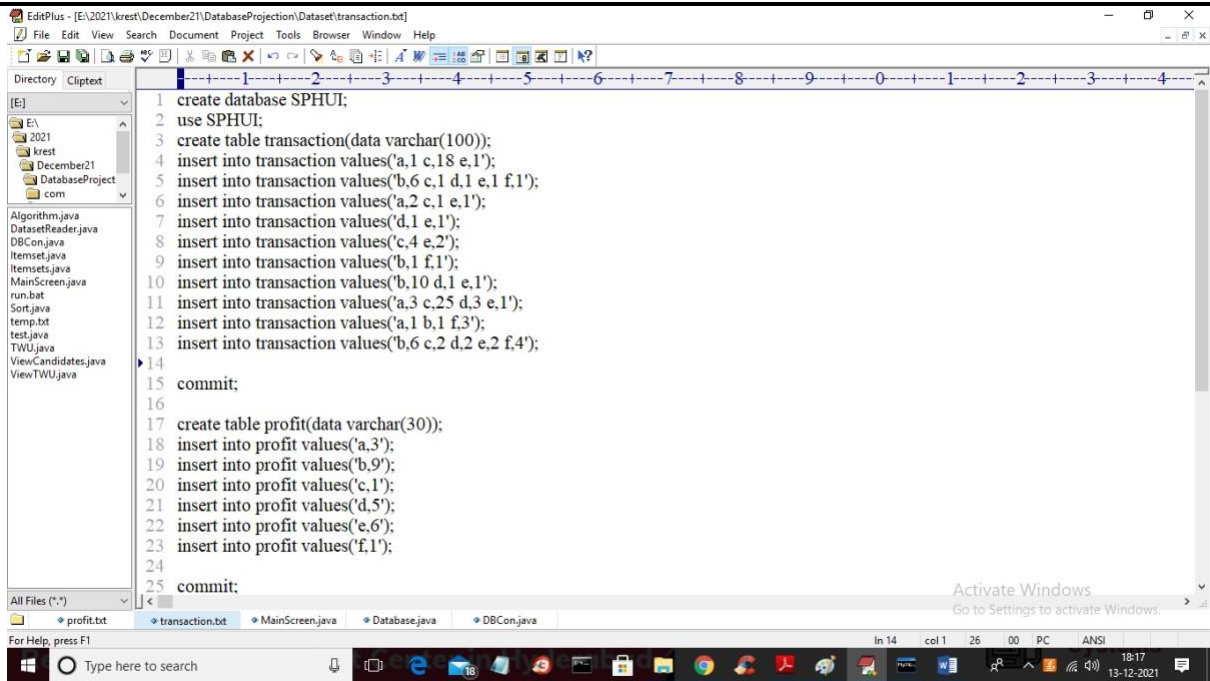
In the proposed paper (SPHUI-Miner) author will calculate Transaction Utility (profit in a single transaction) and Transaction Weighted Unit (profit of a single item in the entire database) and then prune out all those items in a transaction whose TWU does not satisfy MIN UTILITY THRESHOLD (sum of all profit in database * 0.4 thresholds).

Items that are satisfying MIN UTILITY will form a HUI-RTPL (Reduced Transaction Patterns List) also known as the Transaction bitmap matrix. From this matrix we will prune out all common transactions to reduce the pattern list and this reduced list will decrease memory consumption and execution time. The tail count also contains a unique count of each transaction which will reduce scan time as we directly fetch item utility from the tail count by giving the item name.

Selective projection-based utility (SPU-List) will calculate the utility of an itemset Y in the projection PX and if the utility value is greater than MIN UTILITY then it will accept as a high utility item otherwise prune it.

To implement this project we have used the same example given in the paper and we have converted this example to the MYSQL database and then read these records from MYSQL and then run the SPHUI-Miner algorithm on those records.

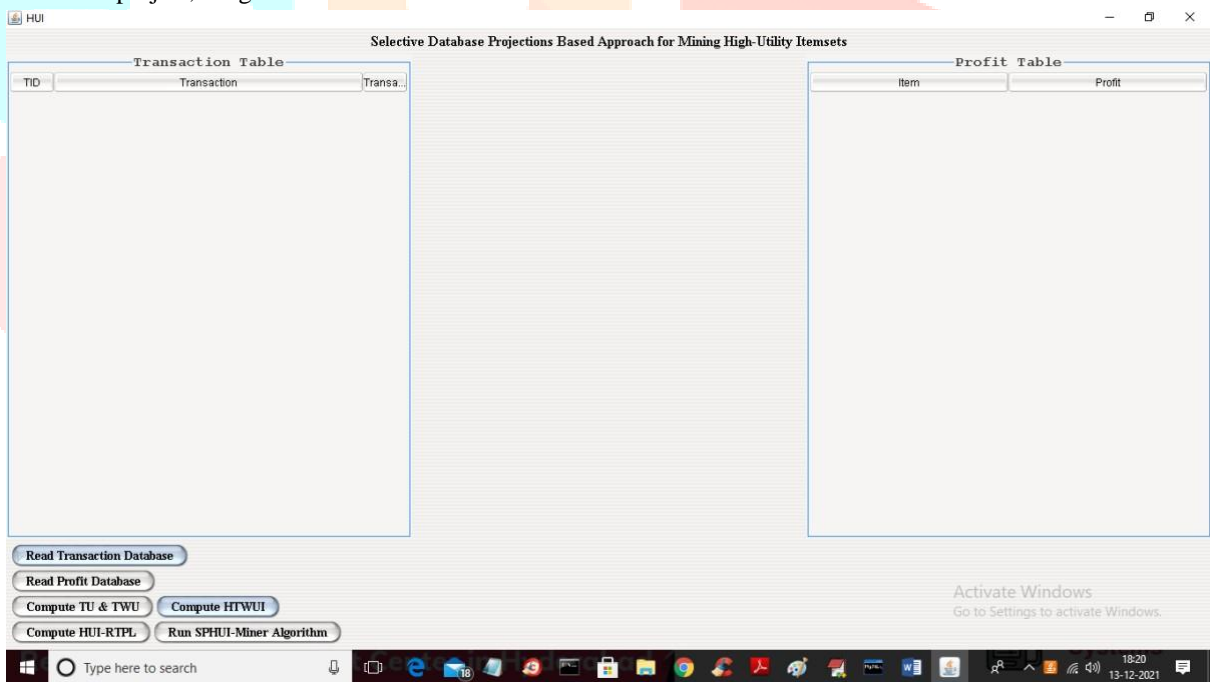
Below screen showing the transaction database used in this project implementation



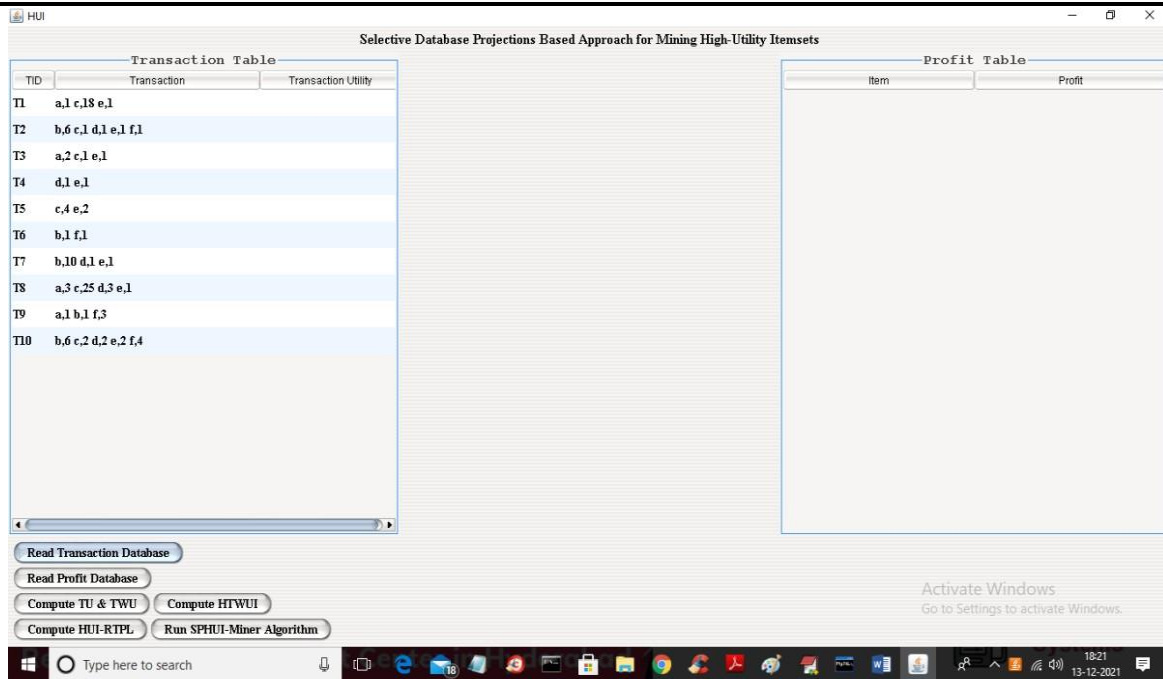
```
1 create database SPHUI;
2 use SPHUI;
3 create table transaction(data varchar(100));
4 insert into transaction values('a,1 c,18 e,1');
5 insert into transaction values('b,6 c,1 d,1 e,1 f,1');
6 insert into transaction values('a,2 c,1 e,1');
7 insert into transaction values('d,1 e,1');
8 insert into transaction values('c,4 e,2');
9 insert into transaction values('b,1 f,1');
10 insert into transaction values('b,10 d,1 e,1');
11 insert into transaction values('a,3 c,25 d,3 e,1');
12 insert into transaction values('a,1 b,1 f,3');
13 insert into transaction values('b,6 c,2 d,2 e,2 f,4');
14
15 commit;
16
17 create table profit(data varchar(30));
18 insert into profit values('a,3');
19 insert into profit values('b,9');
20 insert into profit values('c,1');
21 insert into profit values('d,5');
22 insert into profit values('e,6');
23 insert into profit values('f,1');
24
25 commit;
```

In the above screen, we are using two tables where the first one contains transaction data and the second one contains profit data for each item. By using the above database we will calculate Transaction utility, TWU, HUI RTPL, and then calculate and prune HUI using tail count and SPU LIST.

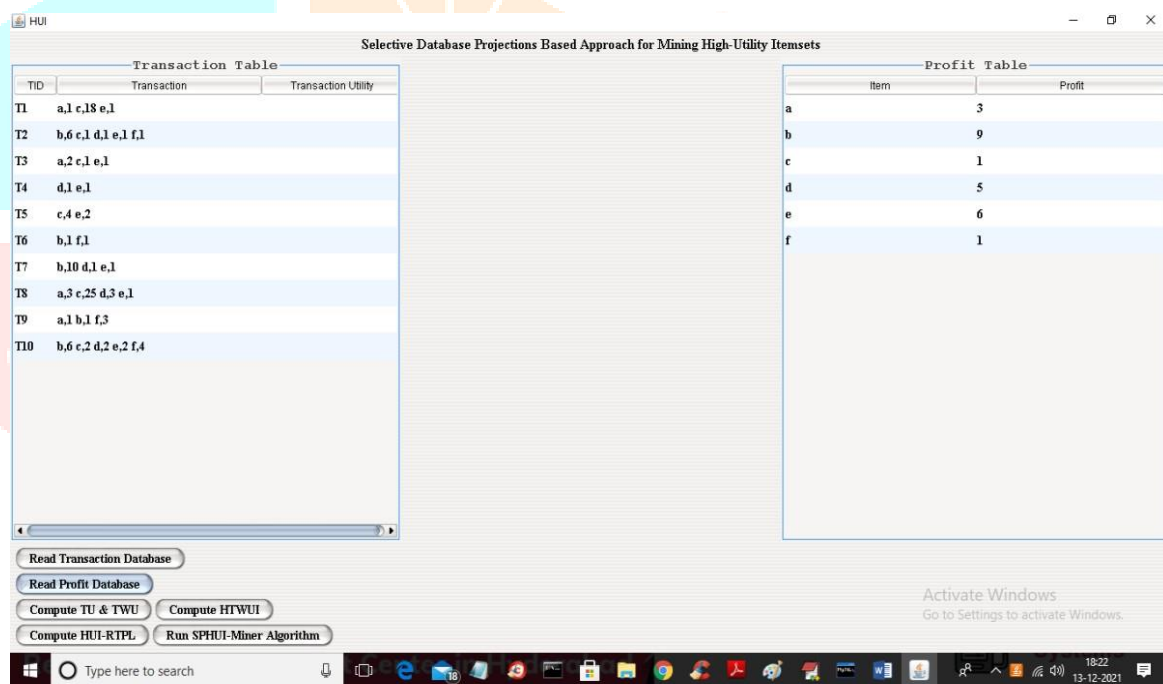
When we run the project, we get the below screen.



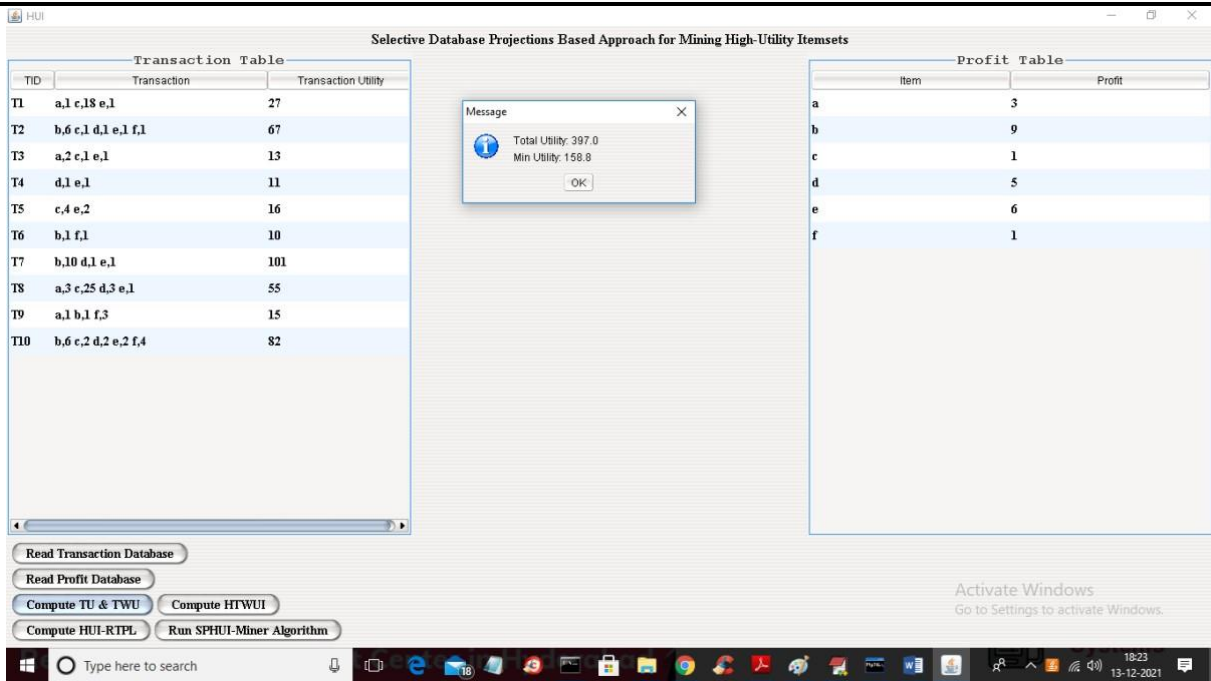
In the above screen, click on the 'Read Transaction Database' button to read transactions from the MySQL database and get the below output.



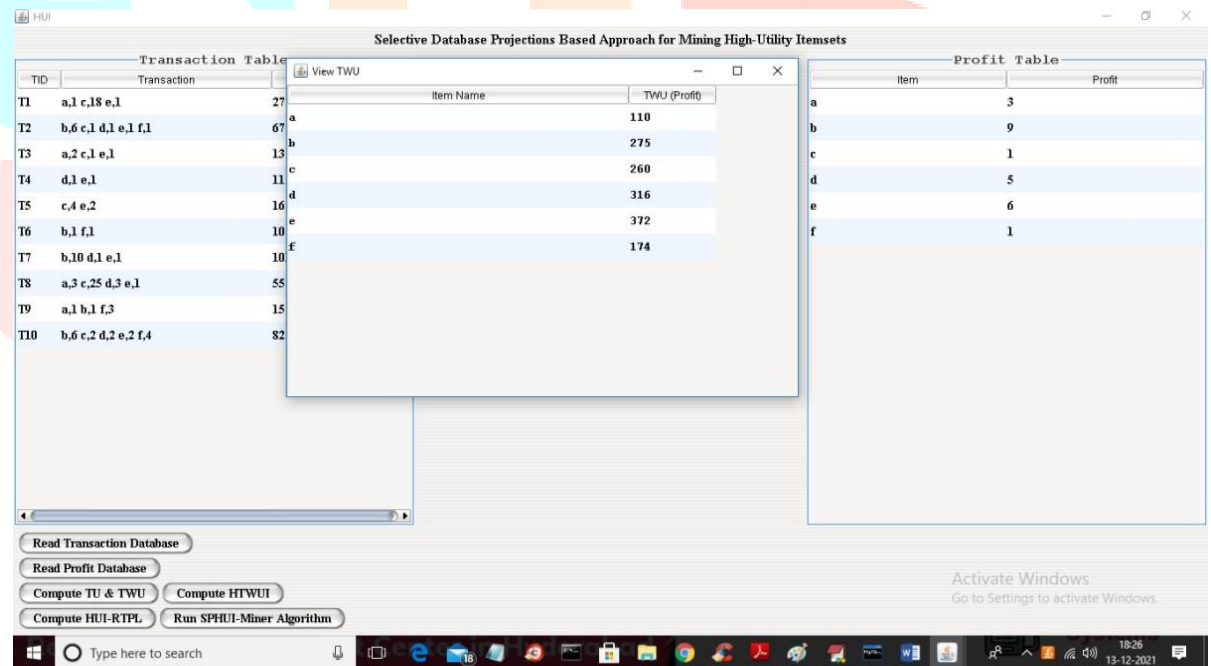
In the above screen, a transaction in the first table, data is loaded from the database but ‘Transaction Utility’ is not found yet, so the column is empty. Click on the ‘Read Profit Database’ button to read the profit table and display the below screen.



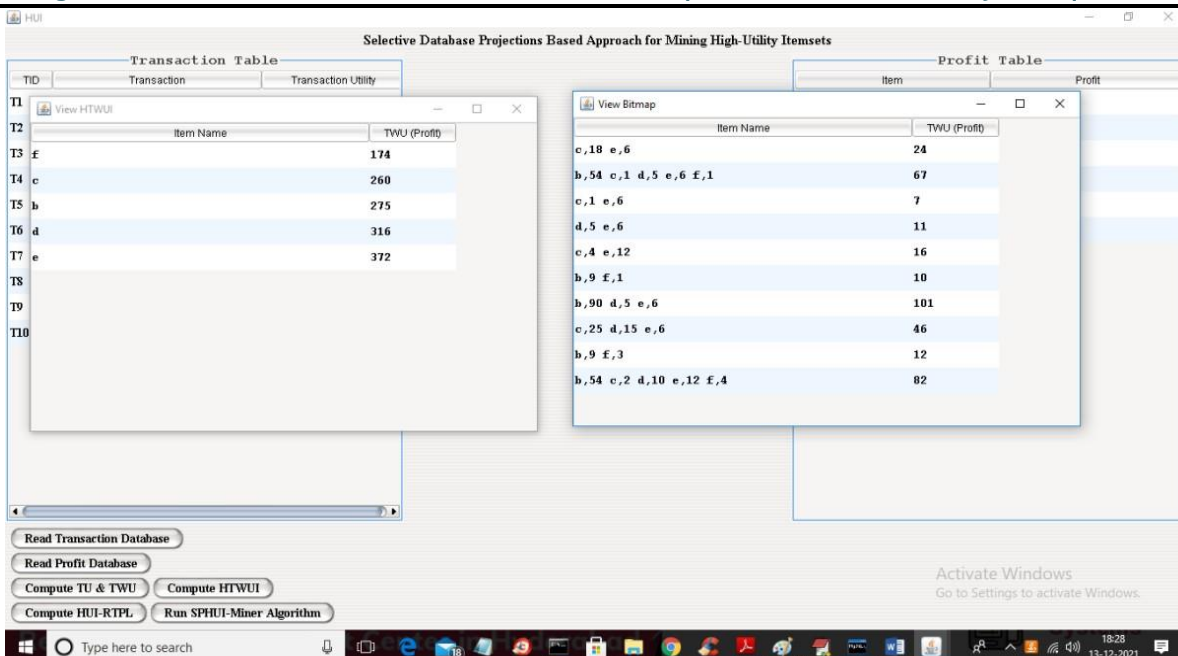
In the above screen in the second table, profit data is also loaded from the database. Click on the ‘Compute TU & TWU’ button to calculate transaction utility and transaction weighted utility and get the below output.



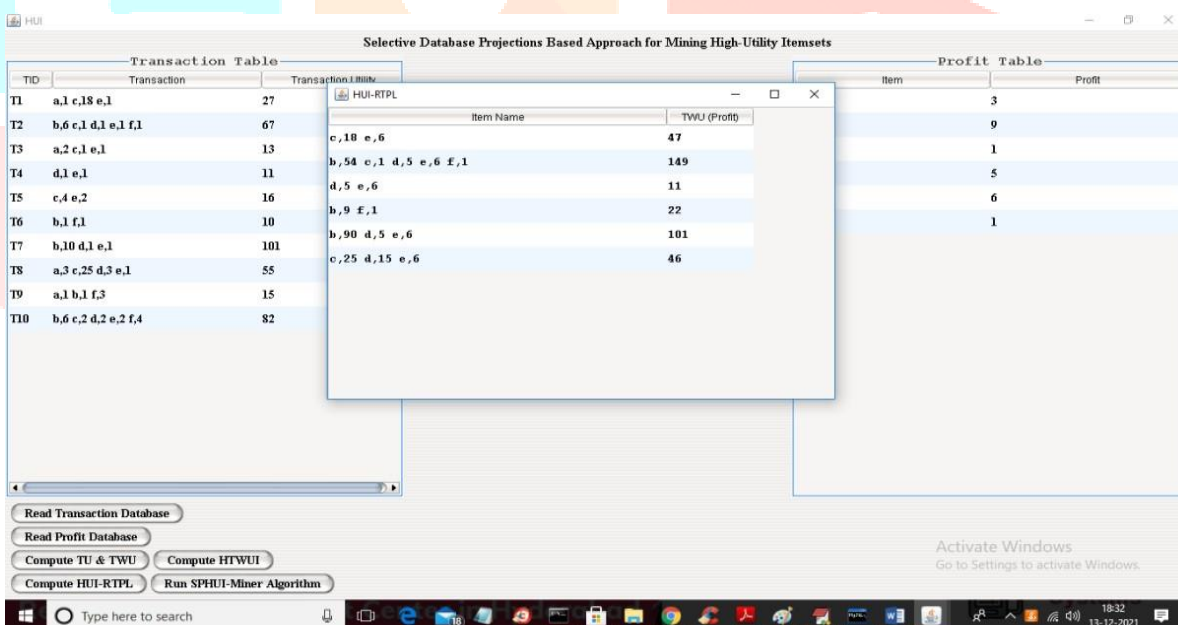
In the above screen, in the first table the last column, we can see that for all the transactions 'Transaction utility' is calculated and if you sum up all the values then the utility value will be 397 and if you multiply that value by 0.4 thresholds then you will get MIN UTILITY value as 158. Click 'OK' in the above dialog box to get the below screen. Now we got min utility value of 158 and all the items whose utility is less than 158 will be removed.



In the above screen, we got TWU values for the items. In all the items only 'a' item value is 110 which is not satisfying MIN UTIL 158 so it will be removed. Close the small screen and click on the 'Compute HTWUI' button to prune 'a' and to get the below output.



In the above screen, the first table shows all the items whose UTILITY values > 158 and 'a' are already removed. The second table shows the binary compact table and in that table, we can see some common transactions, so such common transactions are removed to contain unique transactions. We can also see that 'c, e' appears multiple times, so we sum up all 'c, e' utilities and then remove duplicates to reduce the transaction pattern. The second table consists of a total of 10 transactions and after pruning; it will be reduced to 6. Now close the above two small screens and click on the 'Compute HUI-RTPL' button to get the below output.



In the above screen, we can see that we got a reduced transaction pattern. Now we can find high utility itemsets by clicking on the 'Run SPHUI-Miner Algorithm' button which will find HUI by applying Tail count and SPU LIST.

The screenshot displays the SPHUI-Miner software interface. It features three main windows:

- Transaction Table:** A table listing transactions (T1 to T10) with their itemsets and utility values.

| TID | Transaction | Transaction Utility |
|-----|---------------------|---------------------|
| T1 | a,1 c,18 e,1 | 27 |
| T2 | b,6 c,1 d,1 e,1 f,1 | 67 |
| T3 | a,2 c,1 e,1 | 13 |
| T4 | d,1 e,1 | 11 |
| T5 | c,4 e,2 | 16 |
| T6 | b,1 f,1 | 10 |
| T7 | b,10 d,1 e,1 | 101 |
| T8 | a,3 c,25 d,3 e,1 | 55 |
| T9 | a,1 b,1 f,3 | 15 |
| T10 | b,6 c,2 d,2 e,2 f,4 | 82 |
- View Candidates:** A table showing candidate itemsets with their support counts and TWU (Profit) values.

| Item Name | Support Count | TWU (Profit) |
|-----------|---------------|--------------|
| b d e | 2 | 316 |
| b e | 2 | 276 |
| b d | 2 | 256 |
| b f | 2 | 225 |
| b | 3 | 216 |
- Fit Table:** A table showing profit values for various itemsets.

| Item Name | Profit |
|-----------|--------|
| b d e | 3 |
| b e | 9 |
| b d | 1 |
| b f | 5 |
| b | 6 |
| | 1 |

At the bottom, there are several control buttons: "Read Transaction Database", "Read Profit Database", "Compute TU & TWU", "Compute HITWU", "Compute HUI-RTPL", and "Run SPHUI-Miner Algorithm". The Windows taskbar at the bottom shows the system time as 18:34 on 13-12-2021.

In the above screen, we got final high utility itemsets whose utility values satisfy MIN UTIL value 158.

VII. CONCLUSION

In this paper, we have presented a novel selective database projection-based approach, called SPHUI-Miner, which integrates upper bounds with pruning strategies to efficiently prune unpromising candidates in the search space while mining the HUIs. The proposed algorithm introduces HUI-RTPL to store databases in a compact structure. During selective database projection, it compresses the database relevant to the itemset being investigated. This structure significantly reduces memory consumption. The search over the projections is optimized for faster retrieval of HUIs using Tail-Count and SPU-List. The combined effect of faster computing of utility of itemsets and integrated pruning techniques results in a significant performance increase. We have also shown that at any time during the creation of a projection, the algorithm limits the number of transactions held by newly created projections, which is half the number of transactions in the parent database. For all datasets, the proposed algorithm prunes the tree efficiently and thus requires a very less amount of time and comparable memory. This is the first algorithm that performs selective database projection not only on data instances but also on dimensions for mining HUI. The scalability of SPHUI-Miner is also studied and it is proved experimentally that the algorithm is scalable in terms of memory as well as time. The design of the algorithm is independent of the order in which projections are processed, making it suitable for distributed or parallel implementation. These projections can be stored on a disk and processed one by one. Our linear database representation allows us to estimate a tight bound for efficient pruning and directly find high utility itemsets in an effective and scalable way. The proposed algorithm exhibits high candidate pruning around 2% to 15% as compared to the EFIM algorithm. Extensive experiments on sparse and dense datasets suggest that the proposed algorithm significantly outperforms the state-of-the-art algorithms.

VIII. FUTURE WORK

However, in real life, a database may dynamically change as a result of insertion, deletion, and modification operations and mining can be difficult in an incremental database. To overcome this issue, we need to explore selective database projections for mining high average-utility itemsets for the incremental databases specifically using the HUI-RTPL structure with proposed upper bounds and pruning techniques.

IX. REFERENCES

- [1] S. DESAI, J. PAWAR, AND P. BHATTACHARYYA, "STAGED APPROACH FOR GRAMMATICAL GENDER IDENTIFICATION OF NOUNS USING ASSOCIATION RULE MINING AND CLASSIFICATION," RES. COMPUT. SCI., VOL. 90, PP. 359–371, FEB. 2015.
- [2] G. V. R. Kiran, R. Shankar, and V. Pudi, "Frequent itemset based hierarchical document clustering using Wikipedia as external knowledge," in Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst., 2010, pp. 11–20.
- [3] R. Martinez, N. Pasquier, and C. Pasquier, "GenMiner: Mining nonredundant association rules from integrated gene expression data and annotations," Bioinformatics, vol. 24, no. 22, pp. 2643–2644, 2008.
- [4] R. Agrawal and R. Srikant, "Fast algorithms for mining association rule in large databases," in Proc. 20th Int. Conf. Very Large Data Bases (VLDB), vol. 1215. 1994, pp. 487–499.
- [5] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," Data Mining Knowl. Discovery, vol. 8, no. 1, pp. 53–87, 2004.
- [6] S.-J. Yen and Y.-S. Lee, "Mining high utility quantitative association rules," in Proc. Int. Conf. Data Warehousing Knowl. Discovery, 2007, pp. 283–292.
- [7] C. F. Ahmed, S. K. Tanveer, B. S. Jeong, and Y. K. Lee, "Efficient tree structures for high utility pattern mining in incremental databases," IEEE Trans. Knowl. Data Eng., vol. 21, no. 12, pp. 1708–1721, Dec. 2009.
- [8] H. Yao, H. J. Hamilton, and C. J. Butz, "A foundational approach to mining itemset utilities from databases," in Proc. SIAM Int. Conf. Data Mining, 2004, pp. 482–486.
- [9] H. Yao and H. J. Hamilton, "Mining itemset utilities from transaction databases," Data Knowl. Eng., vol. 59, no. 3, pp. 603–626, 2006.
- [10] C.-W. Lin, T.-P. Hong, G.-C. Lan, J.-W. Wong, and W.-Y. Lin, "Efficient updating of discovered high-utility itemsets for transaction deletion in dynamic databases," Adv. Eng. Inform., vol. 29, no. 1, pp. 16–27, 2015.

