Optimization of High Utility Itemset Mining from Large Transaction Databases on multi-core processor

Ms. Yogita S. Khot
Department of Computer Engineering
PES Modern College of Engineering
Pune, India
yogita.p.narwadkar@gmail.com

Prof. Mrs. Manasi K. Kulkarni
Department of Computer Engineering
PES Modern College of Engineering
Pune, India
mkkulkarni.manasi@gmail.com

***Abstract*—**High utility itemset mining is an emerging era that extends frequent itemset mining to identify itemsets in a transaction database with utility values associated with every item above a given threshold. Researchers recently proposed algorithm TWU (Transaction Weighted Utility) has anti-monotone property for pruning the datasets, but it is an overestimate of itemset utility that leads to more search space.

In this paper we present an algorithm that takes features of CTU-PROL which is proposed by Researchers. It uses TWU with pattern growth based on a compact utility pattern tree data structure. Our algorithm runs on multi-core processor when the main memory is insufficient to deal with large datasets. An experimental result shows a remarkable speedup for large datasets than the previous algorithms. It can mine large data set more efficiently of both dense and sparse data.

Keywords*-High Utility Itemset Mining;CUP-Tree;Multicore Apporach.*

*****

I. INTRODUCTION

The goal of traditional data mining technique is focused largely on finding the items that are more frequent in the transaction databases, without considering the quantity, weight, profit and other user's interest. Itemsets which appear more frequently in the database must be of more meaning to the user from the business point of view. However, profit, quantity and weight are more significant in many areas of business like retail, inventory etc for decision making. This value associated with every item in a database is called the utility of that itemset. Those itemsets having utility values greater than given threshold are called high utility itemsets. This problem can be identified as mining high utility itemsets from transaction database.

In this paper, we present an algorithm that executes on multi-core processor for mining high utility itemsets from large transaction databases using pattern growth approach [6]. The algorithm uses features of CTU-PROL which is proposed by Researchers. It first discovers the large TWU items in the transaction database using user specified threshold and then it creates Compressed Utility Pattern Tree (CUP-Tree) for mining high utility itemsets [14]. If dataset is small then the algorithm executes serially and CUP-tree is created to mine the complete set of high utility itemsets. If the dataset is too large to be held in main memory, the algorithm creates subdivisions using multi-core processors available in the system that can be subsequently mined independently. For each subdivision, a CUP-tree is created to mine the complete set of high utility itemsets [14]. For pruning the search space of subdivision TWU is used.

The performance of our algorithm using multi-core approach is compared with CTU-PROL algorithm [14]. The result shows a remarkable speedup for large datasets than the previous algorithms. It can mine large data set more efficiently of both dense and sparse datasets at most support levels.

II. RELATED WORK

In this section we present a brief review of the different algorithms, techniques, concepts and approaches that have been defined in various research journals and publications. Yao, H., Hamilton, H.J., Buzz, C.J. [2] proposed a framework for high utility itemset mining. They generalize previous work on itemset share measure [2]. This identifies two types of utilities for items, transaction utility and external utility. They identified and analyzed the problem of utility mining. Along with the utility bound property and the support bound property. They defined the mathematical model of utility mining based on these properties. Yao, H., Hamilton, H.J., Buzz, C.J. [3] proposed an algorithm named Umining and another heuristic based algorithm UminingH to find high utility itemsets. They apply pruning strategies based on the mathematical properties of utility constraints. Algorithms are more efficient than any previous utility based mining algorithm. Liu, Y., Liao, W.K., Choudhary A. [4] proposed a two phase algorithm to mine high utility itemsets. They used a transaction weighted utility (TWU) measure to prune the search space. The algorithms based on the candidate generation-and-test approach. The proposed algorithm suffers from poor performance when mining dense datasets and long patterns much like the Apriori [1]. It requires minimum database scans, much less memory space and less computational cost. It can easily handle very large databases. Erwin, A., Gopalan, R.P., N.R. Achuthan [5] proposed an efficient CTU-Mine Algorithm based on Pattern Growth approach. They introduce a compact data structure called as Compressed Transaction Utility tree (CTU-tree) for utility mining, and a new algorithm called CTU-Mine for mining high utility itemsets. They show CTU-Mine works more efficiently than TwoPhase for dense datasets and long pattern datasets. If the thresholds are high, then TwoPhase runs relatively fast compared to CTU-Mine, but when the utility

 ISSN: 2321-8169
4309 - 4313

threshold becomes lower, CTU-Mine outperforms TwoPhase. Erwin, A., Gopalan, R.P., N.R. Achuthan [7] proposed an efficient algorithm called CTU-PRO for utility mining using the pattern growth approach. They proposed a new compact data representation named Compressed Utility Pattern tree (CUP-tree) which extends the CFP-tree of [11] for utility mining.TWU measure is used for pruning the search space but it avoids a rescan of the database. They show CTU-PRO works more efficiently than TwoPhase and CTU-Mine on dense data sets. Proposed algorithm is also more efficient on sparse datasets at very low support thresholds.TWU measure is an overestimation of potential high utility itemsets, thus requiring more memory space and more computation as compared to the pattern growth algorithms. Erwin, R.P. Gopalan, and N.R. Achuthan [14] proposed an algorithm called CTU-PROL for mining high utility itemsets from large datasets. They used the pattern growth approach [6]. The algorithm first finds the large TWU items in the transaction database and if the dataset is small, it creates data structure called Compressed Utility Pattern Tree (CUP-Tree) for mining high utility itemsets. If the data sets are too large to be held in main memory, the algorithm creates subdivisions using parallel projections that can be subsequently mined independently. For each subdivision, a CUP-Tree is used to mine the complete set of high utility itemsets. The anti-monotone property of TWU is used for pruning the search space of subdivisions in CTU-PROL, but unlike TwoPhase of Liu et al. [4], CTU-PROL algorithm avoids a rescan of the database to determine the actual utility of high TWU itemsets. The performance of algorithm is compared against the TwoPhase algorithm in [4] and also with CTU-Mine in [5]. The results show that CTU-PROL outperforms previous algorithms on both sparse and dense datasets at most support levels for long and short patterns.

III. TECHNICAL DETAILS

High Utility itemsets mining algorithm is an extension of frequent pattern mining algorithm to identify high utility itemsets from transaction database. The architecture described here in this paper applies to wide range of transaction datasets. The application and experiment presented here use Java multi-core approach to optimize the code for accelerating the speed of mining.

A. *Terms and Definations*

In this section, we specify basic terms for mining high utility itemsets based on [1, 2, 9]. Let $I=\{i_1,i_2,...,i_m\}$ be a set of items and $D=\{T_1,T_2,...,T_n\}$ be a transaction database where the items of each transaction T_i is a subset of I. The quantity of an item i_p in a transaction Tq is denoted by o (ip, Tq). The external utility s (i_p) is the value of a unit of item ip in the utility table, (e.g., profit per unit). The utility of item ip in transaction Tq, denoted by u (ip, Tq) is defined as o (ip, Tq) x s X (i_p). Utility of item i_p is the number of time the item appears in transaction multiply by profit associated with that item. Transaction utility (TU) of a transaction T_1 is the sum of utilities of all items in that transaction. A set X is called an itemset if X is a subset of I. A set X is called an itemset, if X is a subset of I.

The utility of X in transaction Tq denoted by u (X, Tq) is defined as:

Transaction weighted Utility of an itemset X denoted as TWU (X) is the sum of the transactions utilities (TU) of all the transactions containing X. As shown in [4] any superset of low TWU itemset is also a low TWU itemset and so we can prune all supersets of low TWU itemsets.

B. *Example*

TID	1	2	3	4	5	6	TU.
t_1	2	0	1	1	0	0	80
t_2	2	1	1	0	0	0	195
t_3	0	0	1	1	10	0	110
t_4	0	1	0	0	15	0	225
t_5	1	0	1	0	0	1	37
t_6	2	0	0	1	10	0	105
t_7	2	0	0	0	8	1	62
t_8	1	1	0	1	2	0	205
t_9	1	0	0	1	10	0	95
t_{10}	1	1	0	0	5	0	185
total	12	4	4	5	60	2	1299

Figure 1. Tranasaction Table refer to [14]

		Profit ($)
1	Printer Ink	10
2	Colour Laser Printer	150
3	Bubble Jet Printer	25
4	Digital Camera	35
5	Glossy Photo Paper	5
6	Floppy Disk	2

Figure 2. Utility Table refer to [14]

Suppose we have a small transaction database of an electronic retailer as shown in Figure 1 and Figure 2 shows the profit (external utility) for each item. The values in each row in figure shows the quantity of each item bought in a particular transaction, (i.e. the local transaction utility value). The last column shows the transaction utility for each transaction with total transaction utility of the database in the last row. In transaction t_1, two A (printer ink), one C (bubble jet printer), and one D (digital camera) were bought, yielding transaction utility of $80. The utility of item A, u (A, t_1) is $20 and the utility of item A in the whole database, u (A) = $120. Itemset CD occurs 2 times, in transactions t1 and t_3. Further, u (CD, t_1) = $60, u (CD, t_3) =$60 and u (CD) = $120 and TWU (CD) =$190. The TWU of item B (colour laser printer) is the sum of the transaction utilities of (t_2, t_4, t_8, t_1) = $810 and the TWU of itemset AC is the sum of transaction utilities of (t_1, t_2, t_5) =$312.

IV. HIGH UTILITY ITEMSET MINING

In this section we describe the CTU-PROL algorithm using multi-core processor for mining large data sets [14].

- First calculate values of TU and TWU to identify items of high TWU. By using user specified threshold we prune the items. Rearrange the items with high TWU.
- Subdivide the database by using parallel computing using Java multi-core with high TWU items.

 ISSN: 2321-8169
4309 - 4313

- Construct a Compressed Utility Pattern Tree shown in fig. 3. For each subdivision. It shows high utility itemsets after pruning the items.
- Finally compare the results with previous algorithm.

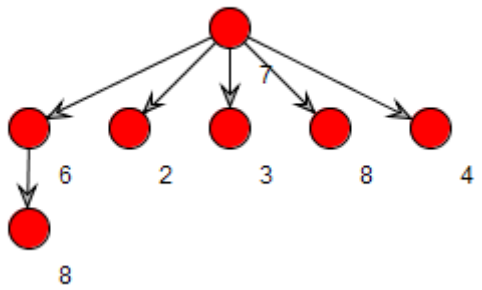

Figure 3. Compressed Utility Pattern Tree

V. EXPERIMENTAL RESULTS

To execute the algorithm on parallel computing we use the system i5, 4GB RAM and JDK 1.8 installed. By performing several experiments, we compare the performance of the serial computing with multi-core described earlier. We compare the result with

- Varying number of transactions with same dataset required. Shown in fig 6.
- Different dataset with same number of transactions. Shown in fig.5.
- Real data sets.

We use real dataset Retails available from FIMI repository. We also generate synthetic datasets 1K (1000 Transactions), 2K, 3K, 4K and 5k using our program to test the scalability of our algorithm. Table 1 shows the characteristics of the datasets. All these datasets are normally used in frequent itemset mining; we had to add only item utility values and price to the databases. We generated utility values from a suitable log-normal distribution and quantities are randomly generated [14].

Results of our experiments are shown in fig.5 and fig. 6. For very low threshold in the small and large Synthetic dataset, CTU-PROL and multi-core runs with same speed, but when the utility threshold becomes higher, multi-core approach runs faster than CTU-PROL.

Dataset	No. Of Transactions	No. Of Items	Size in KB
Real Data (Retail)	881	2959	1023KB
Synthetic 1K	1000	10	139KB
Synthetic 20K	20,000	100	2749KB

Figure 4. Charectristics of Datasets

As shown in fig.7. For varying transactions in different synthetic datasets, multi-core runs faster as compared to CTUPROL. Similarly as shown in fig.7. For varying transactions in Retail dataset multi-core runs faster as compared to CTUPROL. For high threshold in the Retail dataset, CTUPROL runs slightly faster than multicore with minimum transactions, but when the utility threshold becomes lower, multi-core outperforms the CTUPROL.

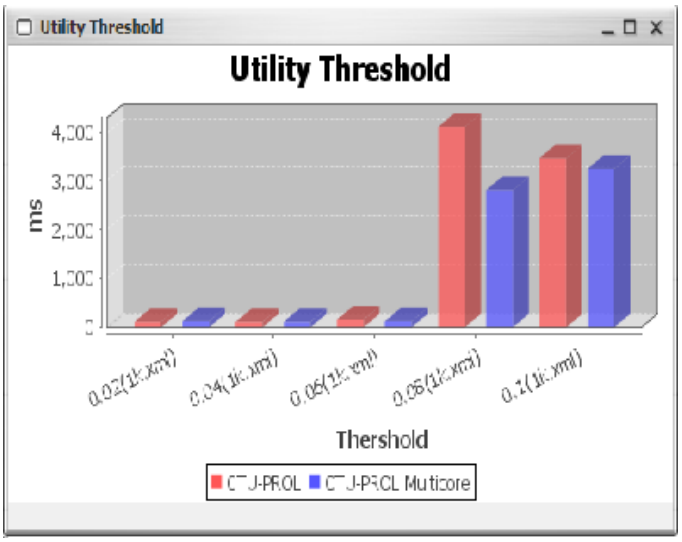

Figure 5. Exexution Time varing with Threshold

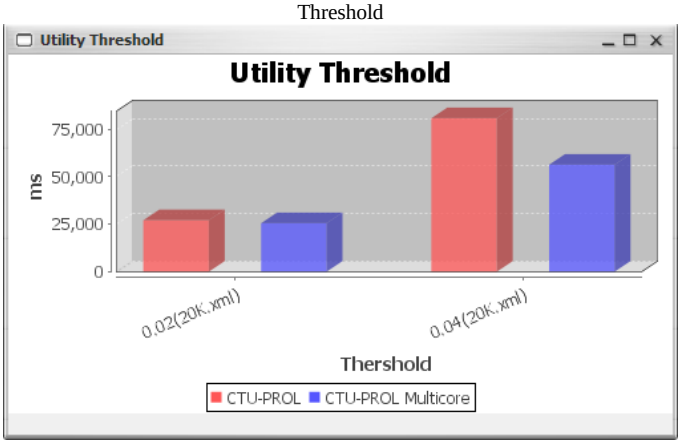

Figure 6. Execution Time varing with Threshold for Large Synthetic Dataset

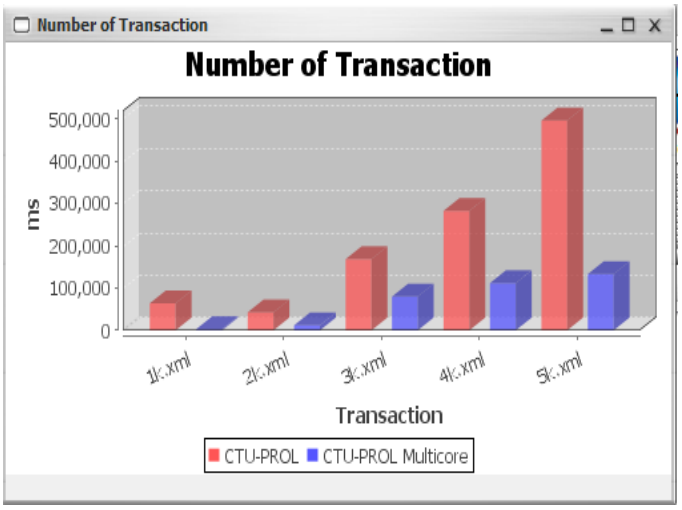

Figure 7. Execution Time varying with number of Transaction

 ISSN: 2321-8169
4309 - 4313

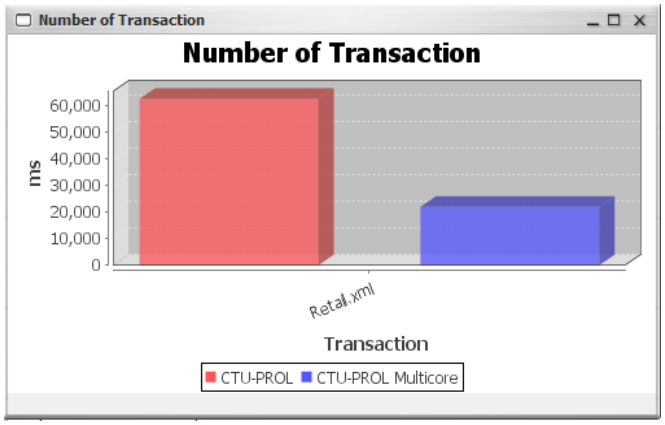

Figure 8. Execution Time varying with number of Transaction for Retail dataset

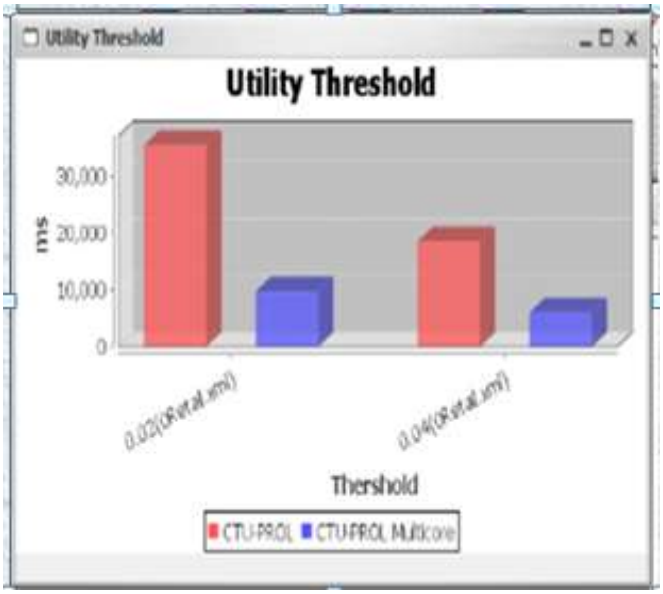

Figure 9. Exexution Time varing with low Threshold for Retail Dataset

VI. CONCLUSIONS

In this paper, we have presented the CTU-PROL algorithm 14] using multi-core processor to mine the complete set of high utility itemsets from large transaction databases. Our multi-core programming extends the pattern growth approach, taking into consideration the lack of anti-monotone property for pruning search space. We have compared the performance of the multi-core CTUPROL with single core CTUPROL algorithm [14] using synthetic and real datasets. The results show that multi-core CTUPROL works more efficiently with graceful up gradation in speed than CTUPROL. Our algorithm adapts to large dataset by constructing parallel subdivision on multi-core available in the system that can be mined independently. The experiments show that our algorithm is scalable for large datasets.

Since resources used in real world are possibly high, TWU is an overestimation the real utility for these pattern growth algorithms. Further research is needed to reduce this overestimation. As the real data for mining is very large in general, we plan to study an efficient Map-Reduce design to reduce the computation.

REFERENCES

[1] Agrawal, R., Imielinski, T., Swami, A., "Mining Association Rules between Sets of Items in Large Database", In: ACM SIGMOD International Conference on Management of Data (1993).

[2] Yao, H., Hamilton, H.J., Buzz, C. J., "A Foundational Approach to Mining Itemset Utilities from Databases", In: 4th SIAM International Conference on Data Mining, Florida USA (2004).

[3] Yao, H., Hamilton, H.J., "Mining itemset utilities from transaction databases", Data & Knowledge Engineering 59(3), 603–626 (2006).

[4] Liu, Y., Liao, W.K., Choudhary, A., "A Fast High Utility Itemsets Mining Algorithm", In: 1st Workshop on Utility-Based Data Mining, Chicago Illinois (2005).

[5] Erwin, A., Gopalan, R.P., N.R. Achuthan, "CTUMine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach", In: IEEE CIT 2007.Aizu Wakamatsu, Japan (2007).

[6] Han, J., Wang, J., Yin, Y., "Mining frequent patterns without candidate generation", In: ACM SIGMOD International Conference on Management of Data (2000).

[7] Erwin, A., Gopalan, R.P., Achuthan, N.R, "A Bottom-Up Projection Based Algorithm for Mining High Utility Itemsets", In: International Workshop on Integrating AI and Data Mining. Gold Coast, Australia (2007).

[8] CUCIS. Center for Ultra-scale Computing and Information Security, Northwestern University, http://cucis.ece.northwestern.edu/projects/DMS/Mine BenchDownloa d.html.

[9] Yao, H., Hamilton, H.J., Geng, L., "A Unified Framework for Utility Based Measures for Mining Itemsets", In: ACM SIGKDD 2nd Workshop on Utility-Based Data Mining (2006).

[10] Pei, J., "Pattern Growth Methods for Frequent Pattern Mining", Simon Fraser University (2002).S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, "A novel ultrathin elevated channel low-temperature poly-Si TFT," IEEE Electron Device Lett., vol. 20.

[11] Sucahyo, Y.G., Gopalan, R.P., CT-PRO: "A Bottom-Up Non Recursive Frequent Itemset Mining Algorithm Using compressed FP-Tree Data Structure", In: IEEE ICDM Workshop on Frequent Itemset Mining Implementation FIMI), Brighton UK (2004).

[12] G. Salton, Automatic Text Processing, Addison-Wesley Publishing, 1989.

[13] J. Pei, J. Han, L.V.S. Lakshmanan, "Pushing convertible constraints in frequent itemset mining", Data Mining and Knowledge Discovery 8 (3) (2004) 227–252.

[14] A. Erwin, R.P. Gopalan, and N.R. Achuthan, "Efficient Mining of High Utility Itemsets from Large Datasets", T. ashio et al. (Eds.): PAKDD 2008, LNAI 5012, pp. 554–561, 2008. © Springer- Verlag Berlin Heidelberg 2008.

ISSN: 2321-8169
4309 - 4313

[15] Bin Chen, Peter Hass, Peter Scheuermann, "A New Two-Phase Sampling Based Algorithm for Discovering Association Rules", SIGKDD '02 Edmonton, Albetra, Canada © 2002 ACM 1 58113 567 X/02/2007.

[16] Ming-Yen lin, Tzer-Fu Tu, Sue-Chen Hsueh, "High utility pattern mining using the maximal itemset property and, lexicographic tree structures", Information Science 215(2012) 1-14.

[17] Sudip Bhattacharya, Deepty Dubey, "High utility itemset mining, International Journal of Emerging Technology and advanced Engineering", ISSN 2250-2459, Volume 2, issue 8, August 2012.

[18] FIMI, Frequent Itemset Mining Implementations Repository.

[19] http://www.cs.uef.fi/~whamalai/datasets.html