FREQUENT UTILITY SEQUENCE PATTERN MINING USING PTREE+

Mrs.C.Sivamathi[1], Dr.S.Vijayarani[2]
[1]Ph.D, Research Scholar, Department of Computer Science, Bharathiar University, Coimbatore, India.
[2]Assistant Professor, Department of Computer Science, Bharathiar University, Coimbatore, India.

Abstract :Sequential pattern mining has various forms in data mining like sequential pattern, constraint-based sequential pattern mining, frequent sequential mining, long sequential pattern mining, utility based sequential mining etc. Among this, recently utility based sequential mining has attracted many researchers. This is because utility based sequence mining considers semantic importance of the sequence, which was not considered in frequent based sequence mining. In this paper a novel algorithm was proposed which considers both frequency and utility of sequence. This algorithm was based on prefix tree structure to retrieve high utility sequences. The algorithm uses the prefix tree to store both support and utility information of sequences. Experimental results were conducted using BMS sequence datasets. Execution time, Memory space and number of patterns retrieved are analyzed in a dataset.

Index Terms: **Sequence mining, Utility sequence mining, High Utility sequences, Frequent high utility sequences, Minimum utility threshold.**

I. INTRODUCTION

Sequential pattern mining plays an important role in many applications like sales of items in retail marketing, purchasing behavior of customers, web log sequence analysis, analysis in protein sequences, analysis in XML query access patterns etc [1]. But traditional frequent based sequential mining considers only the occurrence of sequence, not the semantic importance of the sequence. Hence utility based sequence mining was introduced, to consider the significance of the sequences [2] [3]. This mining results the sequences whose utility value is greater than minimum utility threshold value.

In real time applications, there is need to retrieve patterns both with high utility and high occurrences. For example in retail marketing to promote cross marketing, we need to identify such high utility and frequent sequences. It can also be applicable in web log sequence analysis to identify which websites that was most frequently visited with high utility value. For such cases we need to retrieve patterns by considering both support and utility values. These types of patterns are termed as frequent high utility patterns. These patterns must satisfy two properties: one is , the support of the pattern must be greater than the minimum support threshold and the other is, the utility of the pattern must be greater than minimum utility threshold.

The remaining section of the paper is organized as follows: Section 2 describes related works, proposed work was given under section 3, Section 4 illustrates experimental results and section 5 concludes the work.

II. RELATED WORKS

In 2006, H. Yao et al. proposed UMining [6] algorithm to find almost all the high utility itemsets from an original database. The drawback of this algorithm was that it could not retrieve complete set of high utility itemsets. Later many algorithms were proposed in utility mining to retrieve high utility itemsets.

Yin et al [3] proposed USPAN algorithm to mine high utility sequential patterns. The authors introduced the lexicographic quantitative sequence tree to extract the complete set of high utility sequences. They also introduced two effective pruning strategies. Experimental results on both synthetic and real datasets show that USpan works efficiently, even in a large scale data.

Zihayat et al. [4] proposed an algorithm to retrieve high utility sequence in data streams. The authors proposed two efficient data structures namely ItemUtilLists (Item Utility Lists) and HUSP-Tree (High Utility Sequential Pattern Tree). In addition, a novel utility model called Sequence- Suffix Utility is proposed for effectively pruning the search space in HUSP mining. The authors named the algorithm as HUSP-Miner (High Utility Sequential Pattern Miner) to find HUSPs in static databases efficiently. Then, a one-pass algorithm namely HUSP-Stream (High Utility Sequential Pattern mining over Data Streams) was proposed to incrementally update ItemUtilLists and HUSP-Tree online and find HUSPs over data streams. Experimental results on both real and synthetic datasets show that HUSP-Miner performs well.

Dave[12] proposed an algorithm to retrieve high utility sequences in from Incremental Sequential Dataset. In the process of mining HUS, when new sequences are added into the existing database the whole procedure of mining HUS starts from the scratch, in spite of mining HUS only from incremental sequences. This result in excess of time as well as efforts. So the authors proposed an incremental algorithm to mine high utility sequences from the Incremental Database.

III. Proposed Work

This section illustrates the proposed algorithm to retrieve frequent high utility sequences. The algorithm adopted existing prefix tree structure to retrieve frequent high utility patterns. The tree stores both support and utility information about the item. Using depth first search tree traversal, frequent high utility sequences are retrieved. Initially transaction weighted utility and occurrence of the items are calculated [5] [6] [7]. Then the tree is constructed for high utility items. The items whose utility is greater than transaction weighted utility is known as high utility items [8] [9] [10]. Last depth first search is employed for high utility sequences. This can be illustrated with the following example.

Example:

Table 1. An Example transaction

Trans. id	Transactions	TU
T1	(A,1) (B,3) (C,2)	19
T2	(A,2) (C,3) (D,1)	21
T3	(B,2) (C,1)	10
T4	(B,3) (C,1) (D,3) (E,2)	32
T5	(A,1) (B,3) (C,2) (D,1) (E,2)	28

Table2. Profit table

Item	Unit profit
A	2
B	3
C	4
D	5
E	2

Table3.TWU, support

Item	TWU	Support
A	68	3
B	89	4
C	110	5
D	81	3
E	60	2

From the example, if minimum utility is taken as 80, the items B, C, D is termed as high transaction weighted utility itemsets. Now a prefix tree is constructed with these items. After the construction of prefix tree, frequent high utility sequences, based on minimum threshold values are retrieved directly from the tree.

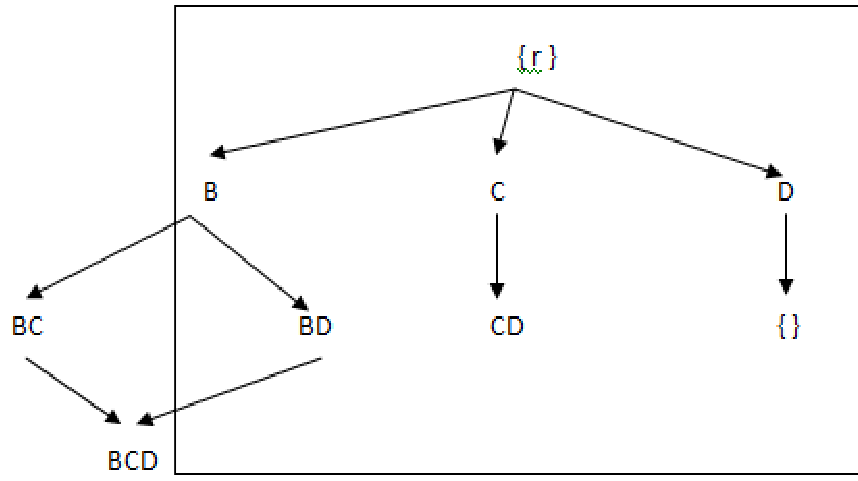

Fig 1.Prefix tree

```
Frequent Utility Sequence Mining – Ptree+
Input: sequence database DB, minimum utility threshold min_util, mecum support threshold min_sup
Output: set of frequent utility sequences, FUS
1: FUS = Ø;
2: Transaction Weighted Utility $TWU = \sum TU$; TU = Trasanction Utility;
3: If (TWU> min_util) return HTWU;
4: Construct Prefix tree(HTWU)
5:for itemsets in HTWU
6: If(HTWU.itemset> min_util && HTWU.itemset> min_sup)
7:FUS = itemset;
8:End for;
9:End if;
```

Pseudocode for Frequent Utility Sequence Mining

IV. Experimental Results

The algorithm is implemented in Java language. The software tool used is NetBeans IDE 8.0. The dataset used in the experiment is BMS sequence dataset. This dataset contains 59,601 sequences of clickstream data from an e-commerce. It contains 497 distinct items. The average length of sequences is 2.42 items with a standard deviation of 3.22. In this dataset, there are some long sequences. For example, 318 sequences contain more than 20 items. This dataset has been taken from SPMF repository. The experiment is conducted with various minim support threshold and minimum utility threshold. Also in retrieving sequences, the length of sequence is also taken as input. This is because, sequence mining takes more space and time complexity, so if maximum sequence pattern is mentioned in advance, the required patterns can be retrieved efficiently. Table 1 shows the execution time, memory space and number of patterns retrieved with different support and utility values. This result displays the number of sequences with maximum length 2. Table 2 shows the same for different length of patterns.

Table 1. Performance of proposed algorithm with different support and utility values

Min_support	Min_utility	Exe. Time in ms	Memory space in MB	No. of patterns retrieved
20	30	33	9.48	1359
	40	24	11.34	914
	50	11	15.47	565
	60	9	16.24	353
30	30	29	8.54	1047
	40	21	10.5	748
	50	10	13.2	389
	60	9	14.2	281
40	30	25	7.5	854
	40	19	9.58	587
	50	10	12.5	256
	60	9	12.1	148

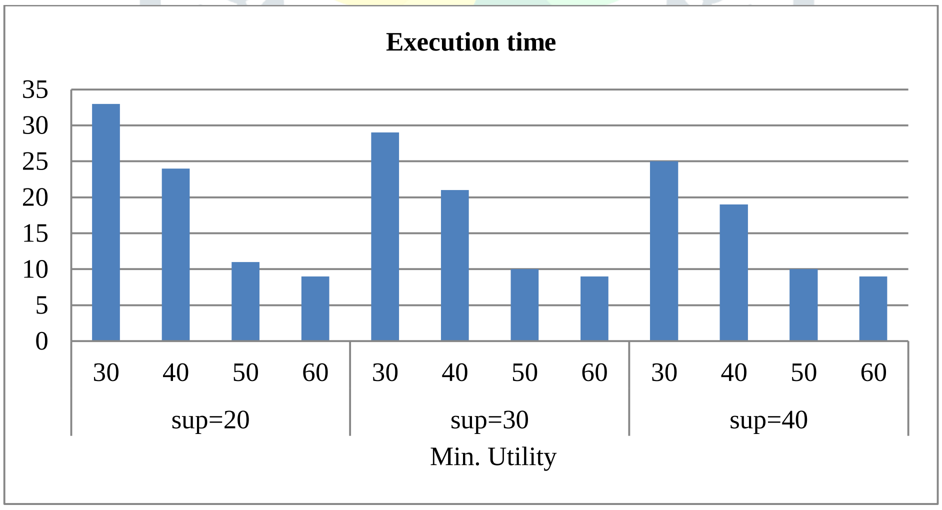

Fig 2.Comparision of Execution time

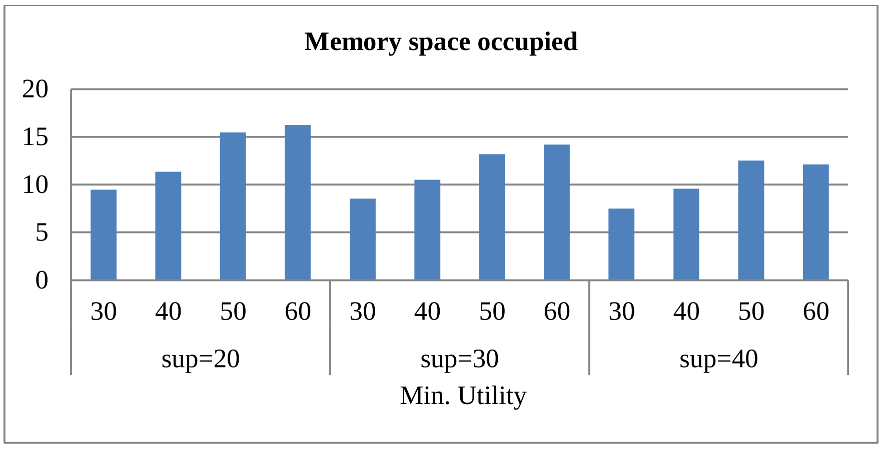

Fig 3.Comparision of Memory space occupied

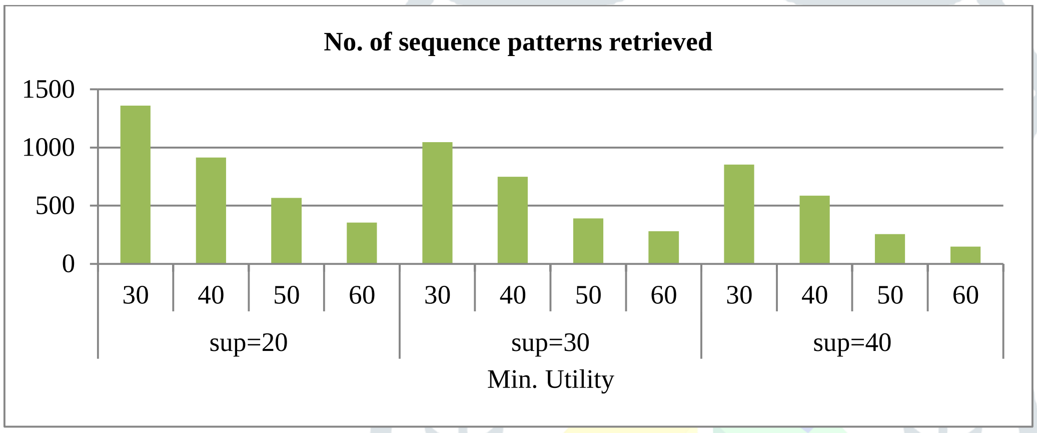

Fig 4.Comparision of Number of patterns retrieved

Table 2 Performance of proposed algorithm with different length of patterns

Length of pattern	**Exe. Time in ms**	**Memory space in MB**	**No. of sequence patterns retrieved**
2	45	9.58	5874
3	84	13.99	10005
4	1125	16.22	205306
5	10240	247.49	2667199

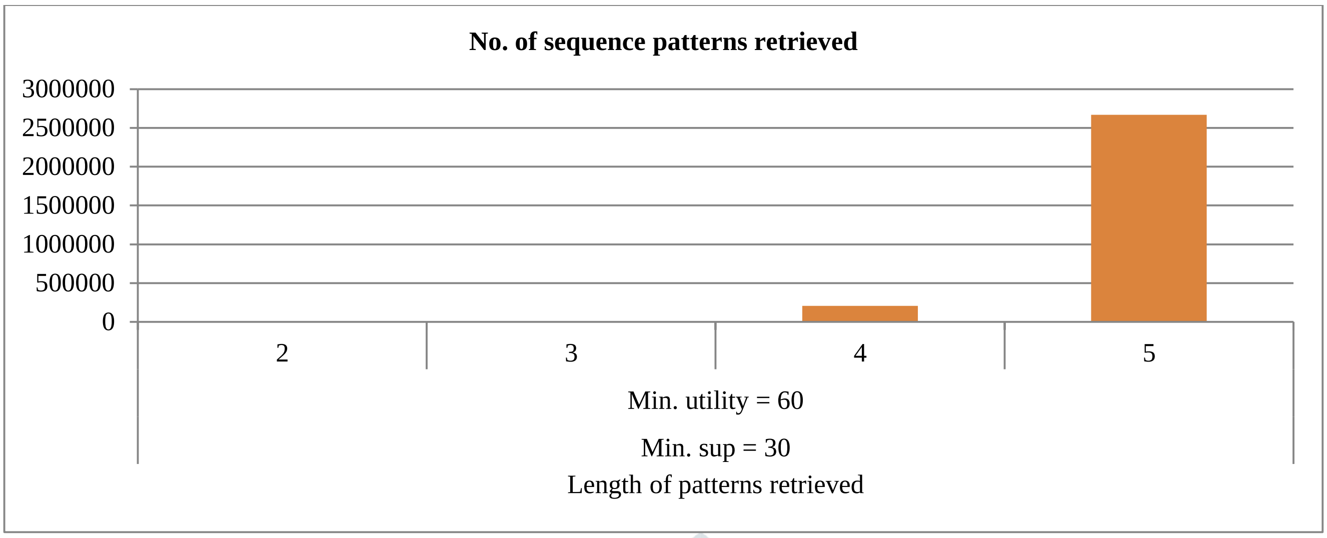

Fig 5.Number of patterns retrieved

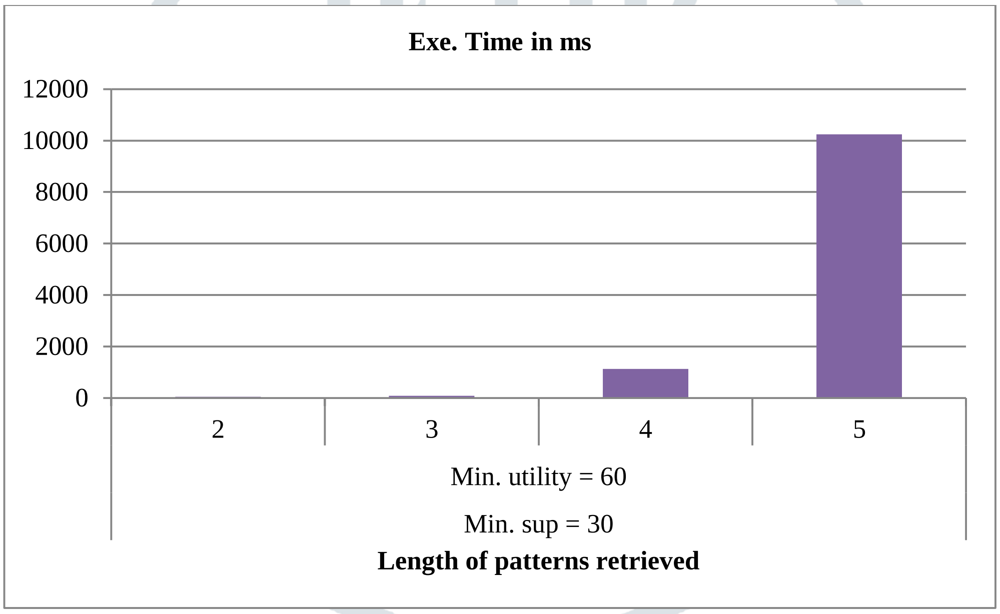

Fig 6.Execution time based on the length of patterns

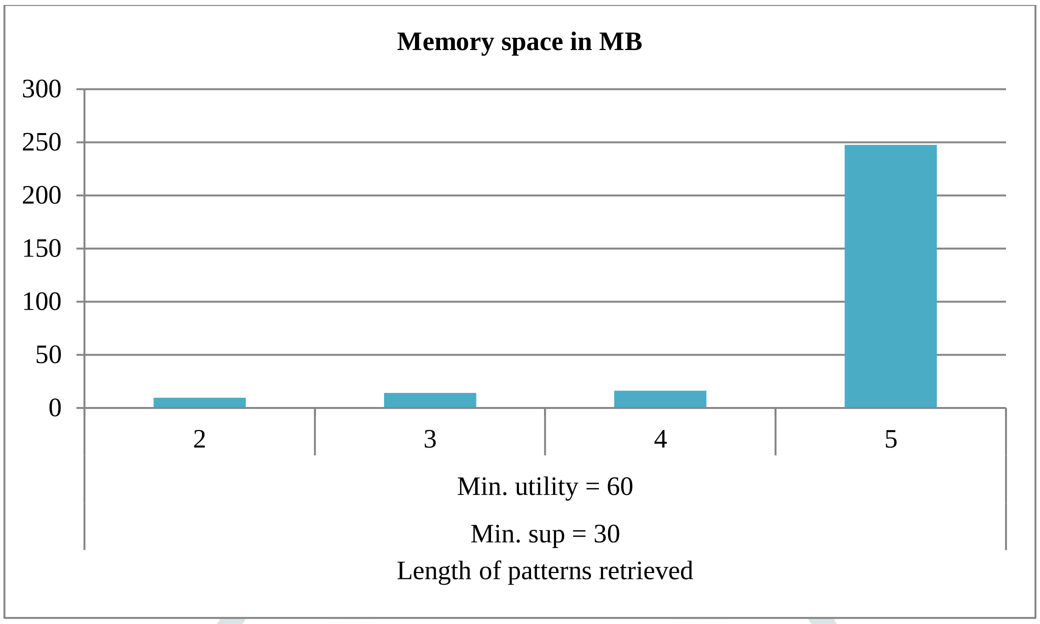

Fig 7.Memory space occupied based on the length of patterns

V. Conclusion

In recent years utility based sequential mining has emerged as an interesting research area. This is because utility based sequence mining considers semantic significance of the sequence, which was not considered in frequent based sequence mining. In this paper a novel algorithm was proposed which considers both frequency and utility of sequence. This algorithm was based on prefix tree structure to mine high utility sequence pattern mining algorithm. Experimental results were conducted using BMS sequence datasets. Execution time, Memory space and number of patterns retrieved are analyzed with different support and utility values. Also the same are retrieved using different length of sequence.

References:

1. Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu, Fellow," Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases", IEEE Transaction on knowledge and data engineering, vol. 25, no. 8, Aug 2013.

2. C.F. Ahmed, S.K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, "Efficient Tree Structures for High Utility Pattern Mining in Incremental Databases", IEEE Trans. Knowledge and Data Eng., vol. 21, no. 12, pp. 1708-1721, Dec. 2009

3. Junfu Yin, Zhigang Zheng & Longbing Cao, "USpan: An Efficient Algorithm for Mining High Utility Sequential Patterns", in proceeding of KDD'12, August 12–16, 2012,

4. Morteza Zihayat, Cheng-Wei WU , Aijun AN and Vincent S. Tseng, "Efficiently Mining High Utility Sequential Patterns in Static and Streaming Data".

5. J Han, J.Pei, Y.Yin ,R. Mao Mining frequent Patterns without candidate generation:a frequent -pattern tree approach , Data Mining and Knowledge Discovery 8(1)(2004) 53-87

6. Liu. Y, Liao. W,A. Choudhary, A fast high utility itemsets mining algorithm, in: Proceedings of the Utility-Based Data Mining Workshp, August 2005

7. Y.-C. Li, J.-S. Yeh, and C.-C. Chang, "Isolated Items Discarding Strategy for Discovering High Utility Itemsets," Data and Knowledge Eng., vol. 64, no. 1, Jan. 2008.

8. C.H. Cai, A.W.C. Fu, C.H. Cheng, and W.W. Kwong, "Mining Association Rules with Weighted Items," Proc. Int‟l Database Eng. and Applications Symp. (IDEAS ‟98), 1998.

9. R. Chan, Q. Yang, and Y. Shen, "Mining High Utility Itemsets," Proc. IEEE Third Int‟l Conf. Data Mining, pp. 19-26, Nov. 2003.

10 V.S. Tseng, C.-W. Wu, B.-E. Shie, and P.S. Yu, "UP-Growth: An Efficient Algorithm for High Utility Itemsets Mining," Proc. 16th ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD ‟10), 2010.

11 H. Yao, H.J. Hamilton, and L. Geng, "A Unified Framework for Utility-Based Measures for Mining Itemsets," Proc. ACM SIGKDD Second Workshop Utility-Based Data Mining, Aug. 2006.

12. Jiawei Han, Hong Cheng, Dong Xin and Xifeng Yan, "Frequent pattern mining: current status and future directions," Data Mining Knowledge Discovery, January 2007.

13. M. J. Zaki, SPADE: An Efficient Algorithm for Mining Frequent Sequences, Machine Learning, 2001, vol. 42, pp. 31-60.

14. S. J. Yen and Y. S. Lee. 2007. Mining high utility quantitative association rules." In DaWaK 2007, LNCS 4654, pp. 283-292.

15. Z. H. Deng and Z. H. Wang. 2010. A New Fast Vertical Method for Mining Frequent Itemsets. International Jour-nal of Computational Intelligence Systems, 3(6): 733 – 744.