Distributed Mining of High Utility Sequential Patterns with Negative Item Values

Manoj Varma[1], Saleti Sumalatha[2], Akhileshwar reddy[3]
School of Engineering and Sciences
Computer Science and Engineering
SRM University AP
India

***Abstract*—The sequential pattern mining was widely used to solve various business problems, including frequent user click pattern, customer analysis of buying product, gene microarray data analysis, etc. Many studies were going on these pattern mining to extract insightful data. All the studies were mostly concentrated on high utility sequential pattern mining (HUSP) with positive values without a distributed approach. All the existing solutions are centralized which incurs greater computation and communication costs. In this paper, we introduce a novel algorithm for mining HUSPs including negative item values in support of a distributed approach. We use the Hadoop map reduce algorithms for processing the data in parallel. Various pruning techniques have been proposed to minimize the search space in a distributed environment, thus reducing the expense of processing. To our understanding, no algorithm was proposed to mine High Utility Sequential Patterns with negative item values in a distributed environment. So, we design a novel algorithm called DHUSP-N (Distributed High Utility Sequential Pattern mining with Negative values). DHUSP-N can mine high utility sequential patterns considering the negative item utilities from Bigdata.**

Keywords*—High utility sequential pattern mining; big data; utility mining; negative utility; distributed algorithms

I. Introduction

These days we can't imagine the volume of data that is produced every day in the form of sequences [14] [15]. Mining high utility patterns is a prominent job in data mining that discovers the itemsets which appear frequently in sequences. Many current algorithms [5] [16] [24] only take into account the frequency of each object in a sequence and presume that the importance of items is the same for various items. In [6], the authors depicted such approaches are not sufficient for industry needs. In reality, these algorithms mined patterns are not especially related to business needs, so they don't really know the patterns are interesting for their business. For example, in a retail market analysis, each item possess its own profit value, and an item will exist in the purchasing record of a customer many times. Utility was introduced to mine frequent patterns to resolve this issue by considering the profit (quality) and quantity of products. This introduce a novel field of study, namely, high utility itemset mining and high utility sequential pattern mining (HUSP), these are able to mine insightful knowledge, given a minimum utility defined by the user instead of minimum support. Utility model-based knowledge can supply more useful and applicable decision-making information than those based on a conventional support framework. High utility sequential pattern (HUSP) mining [2] [23] is used to extract profitable and more beneficial sequential patterns from databases. It considers a business intention such as profit, user interests, value, etc. A sequence mined from a sequence database is said to be a high utility sequential pattern only if it is having an utility not less than the minimum utility threshold supplied by the user. So, we came up with a new method for mining sequential patterns with high utility that includes negative item values using a distributed approach. Here we use algorithms like Hadoop map reduce [9] to operate data quickly in parallel. We suggest few pruning strategies to eliminate unpromising items that leads to minimize the search space in distributed circumstances.

The following are the contributions of the current work:

1. We made a complete overhaul to HUSP-NIV [21] algorithm and studied the distributed solution to the problem of HUSP-N [22] mining.

2. MapReduce algorithm is proposed for extracting HUSPs with negative item values.

3. Proposed a distributed utility upper bound that supports global mining of HUSP-N's.

4. Several experimental evaluations have been accomplished on the real as well as synthetic datasets to assess the efficiency of DHUSP-N algorithm.

The remaining sections in the paper are composed as follows: Description of related work is mentioned in Section II. Section III provides a detailed description of problem definition. The details of DHUSP-N are given in Section IV. The performance details of DHUSP-N obtained from the experimental results are noted in Section V. The enhancements of the current work and its conclusion is given in Section VI.

II. Related Work

Sequential pattern mining became a buzzword and many algorithms [1] [4] [7] [8] [17] have been proposed. Sequential pattern mining is an extenison to frequent itemset mining based on support framework that was firstly introduced by Srikant and Agrawal [1] in their studies. He gave a new definition by adding different time constraints and other attributes like sliding time window, user-defined taxonomy, and introduced a generalised sequential pattern (GSP) algorithm. Wang et al. [20] proposed novel pruning strategies namely, RSU and PEU to remove the sequences with less utility and designed HUS-Span algorithm to efficiently extract HUSPs. Truong-Chi & Fournier-Viger [8] has described about high utility

TABLE I. SAMPLE DATASET

S_id	$Q-sequence$
1	$\langle(I_5,2)[(I_1,4)(I_2,2)](I_4,4)\rangle$
2	$\langle[(I_1,3)(I_2,1)][(I_1,1),(I_3,1),(I_4,3)][(I_1,2),(I_4,3)(I_5,3)]\rangle$
3	$\langle[(I_3,4)(I_6,6)][(I_2,3)(I_4,3)]\rangle$
4	$\langle[(I_2,1)(I_3,6)][(I_1,3)(I_4,3)][(I_1,4)(I_2,1)(I_3,2)]\rangle$
5	$\langle[(I_2,2)(I_5,3)][(I_1,3)(I_6,2)][(I_1,2)(I_2,1)]\rangle$

sequence mining. Many other pattern mining problems were generalized by the authors, such as frequent itemset mining in transaction databases, sequential pattern mining in sequence databases, and high utility itemsets in databases of quantitative transactions. The sequential order between the items and their utility has been considered to mine high utility sequences from a quantitative sequence database. Guha et al. [11] used the regular expressions as a constraint for user-controlled focus on mining sequential patterns. Some more algorithms like USpan [23], HUS-Span [20] and HuspExt [3] algorithms have been designed to extract high utility patterns based on utility concept but they are not designed to use negative patterns. Negative sequential patters (such as missing medical check-ups) are crucial and more useful than positive sequential patterns (e.g. visiting a medical check-up) in many intellectual systems and applications such as healthcare analysis and risk management. However, exploring sequential patterns with negative item values is considerably more complex than sequential pattern mining including positive item values because of acute time complexity occurred by non-repeating elements, high time complexity and large search space in finding negative sequential patterns. Xu et al. [21] came up with HUNSPM. This algorithm considers the items that do not occur into attention. These are the first studies to mine HUNSPM (high utility negative sequential pattern mining). These algorithms can mine HUSPs efficiently from a non-decentralized database using a single machine, however, they cannot handle big data [12]. Also, their proposed pruning techniques cannot be applied in a distributed environment. Mining patterns from big data on a single machine is very costly to execute the mining algorithms. Developing a distributed algorithm that mines HUNSPs is a key to handle the problem. Recently, Lin et al. [13] introduced an algorithm for high utility itemset mining which is applicable for handling big data. The approach proposed in [13] do not consider the sequential ordering of itemsets. Adding the sequential order of itemsets makes it more challenging to mine. Recently, we proposed a distributed MapRedcue algorithm that can mine high utility time interval sequential patterns [19]. However, we do not include the negative item values. This motivates us to study a novel approach of mining HUSP that includes negative item values from a distributed environment.

No approach has been introduced till now for high utility sequential pattern mining that can consider both the utilities and negative values in a distributed environment, to the best of our understanding. So, we design a novel algorithm called DHUSP-N that can extract all sequential patterns with high utility and negative item values that appear in bigdata.

III. PROBLEM DEFINITION

Given a sequential database D, the problem of mining sequential patterns of high utility with negative item values from large databases in a distributed way is described here. Let a set of distinct items be $I = \{i_1, i_2, i_3, i_4, \ldots, i_n\}$. A positive or negative number $p(i_k)$, called its external utility is associated with each item i_k. The quantity or internal utility of I is called a q-item (i,q), where $i \in I$ and q denotes the purchased amount of i. Our problem is to mine all high utility sequential patterns with negative item values (DHUSP-N) in a distributed environment with a minimum utility threshold δ.

TABLE II. QUALITY TABLE

Item	I_1	I_2	I_3	I_4	I_5	I_6
Quality	5	-3	1	2	4	1

Example: Consider a Q-sequence database as in Table I. Each entry in the Q-sequence database is said to be a q-sequence. The q-sequence S_1 depicts the items I_5, I_1, I_2 and I_4 with internal utility of 2, 4, 2 and 4. Table II gives us the external utilities of these items respectively 4, 5, -3 and 2. So the item I_2 is sold at loss. $[(I_1,4)(I_2,2)]$ is the itemset with two q-items.

Definition 1: Sequence $\alpha = k_1, k_2, \ldots, k_i$ is a subsequence of sequence $\beta = k_1, k_2, \ldots, k_j (i \leq j)$ or the other way β is a super-sequence of α.

Definition 2: Every element in the itemset consists of positive number $p(I)$, called the external utility (e.g. price/profit per unit). Every item I in itemset X_d of particular sequence S_r (i.e., S_r^d) has a positive number $q(I, S_r^d)$, called as its internal utility (e.g., quantity) of I in itemset of particular sequence.

Definition 3: The utility of a q-item is defined as the product of internal utility and external utility. The utility of a q-itemset is defined as the sum of each item utility in the q-itemset. The q-sequence utility is defined as the sum of each item utility having positive external utility. For example, utility of $(I_1, 4)$ in itemset 2 of sequence S_1 is $4 \times 5 = 20$. The utility of itemset $[(I_1, 4)(I_2, 2)]$ is S_1 is $4 \times 5 + 2 \times -3 = 20 - 6 = 14$. The q-sequence utility of $S_1 = \langle(I_5, 2)[(I_1, 4)(I_2, 2)](I_4, 4)\rangle$ is $2 \times 4 + 4 \times 5 + 2 \times 4 = 8 + 20 + 8 = 36$.

Definition 4: The sequence local utility in partitioned database D_i is defined as $su_L(\alpha, D_i) = \sum_{S_r \in D_i} su(\alpha, S_r)$ for sequence α in the partition D_i. The sum of local utilities of α in each partition D_i is defined as its global utility and denoted as su_G.

Definition 5: The total utility of a partition D_i is denoted as U_{D_i} and is defined as sum of sequence utility of each S_i, where S_i is an input sequence in D_i. The total utility of sequence database D is denoted as U_D and is defined as the sum of total utility of each D_i.

Definition 6: Given a sequence α, it is called a local high utility sequential pattern with negative value (L-HUSP-N), iff $su_L(\alpha, D_i) \geq \delta \cdot U_{D_i}$, where δ is the minimum utility threshold.

Definition 7: Given a sequence α, it is called a global high utility sequential pattern with negative value (G-HUSP-N), iff $su_G(\alpha, D) \geq \delta \cdot U_D$, where δ is the minimum utility threshold.

Definition 8: Given a sequence dataset D and a minimum utility threshold δ, then the sequence α is said to be a high

TABLE III. UTILITY MATRIX FOR SEQUENCE S_1

Item	q-itemset1	q-itemset2	q-itemset3
a	(0,36)	(20,8)	(0,8)
b	(0,36)	(-6,8)	(0,8)
d	(0,36)	(0,8)	(8,0)
e	(2,28)	(0,8)	(0,8)

GSWU sequence if and only if $GSWU(\alpha, D) \geq \delta \cdot UD$.

Definition 9: Mining of HUSP with/without Negative Item Values - Selecting the highest utility from the utility estimations of every q-sequence and add them together to address the sequence's utility in a given sequence database.The max utility is used to denote the utility of a sequence t and it is characterized as $u_{max}(t) = \sum_{s \in S} max\{u(t, s)\}$.

HUSP mining is to mine all the HUSPs from the database where each item possess only positive external utilities, whereas HUSP-N mining is to mine all the HUSPs from the database where each item in the database may have either positive or negative external utilities. The sequence t is called as HUSP in Q-sequence Database S if and only if $u_{max}(t) \geq \delta$. The properties defined for HUSP-N mining are as follows [21]:

Property 1: HUSP can have items with negative external utilities.

Property 2: No less than one item having a positive external utility must be included in a high utility sequential pattern.

Definition 10: Sequence-weighted Utilization (SWU) of a particular sequence t in q-sequence database S is defined as sum of q-sequence utilities where t is a subsequence to q-sequence.

Definition 11: Given a bunch of sequences D_i, the Local Sequence-Weighted Utility (LSWU) of a sequence α in D_i, indicated as $LSWU(\alpha, D_i)$, is characterized as the amount of the utilities of sequences containing α in D_i, where $\alpha \leq S$ implies α is a subsequence of S. In like manner, the Global Sequence-Weighted Utility (GSWU) of a sequence α in information base D is characterized as: $GSWU(\alpha, D) = \sum_{(D_i \subset D)} LSWU(\alpha, D_i)$.

Property 3: Sequence-weighted Downward Closure Property- Given the S database of q-sequences, and two t_1 and t_2 sequences, where t_2 contains t_1, then t_2 contains t_1, then $SWU(t_2) \leq SWU(t_1)$.

Definition 12: Utility matirx (UM)- It is a data structure introduced in USpan [23] algorithm to store the q-sequence utility. Each element in the matrix stores two values, the first one is item's utility, where as the second is item's remaining utility.

Definition 13: The remaining utility in the Utility Matrix is only the sum of all remaining q-sequence items' positive external utility values.

From Definitions 12 and 13, the utility matrix created for sequence S_1 in our sample database is shown in Table III.

Definition 14: Given a sequence pattern α, an I-concatenate pattern β is a sequence obtained by including an item I to the last itemset α.

Definition 15: Given a sequence α, S-concatenate pattern β denotes a sequence obtained by including a 1-Itemset $\{I\}$ after the last itemset of α.

IV. METHODOLOGY

Mining HUSP with negative values in the big data era is a hectic job to do due to the hurdles of data growth in an exponential way. It is expensive to mine patterns in a single individual machine. Designing a distributed and parallel algorithm is the one solution that we are thinking of. To implement this approach, we need to address a few key issues like decreasing the search space in the data, decreasing the communication overhead between different local machines, and finally the scalability issues to be answered.

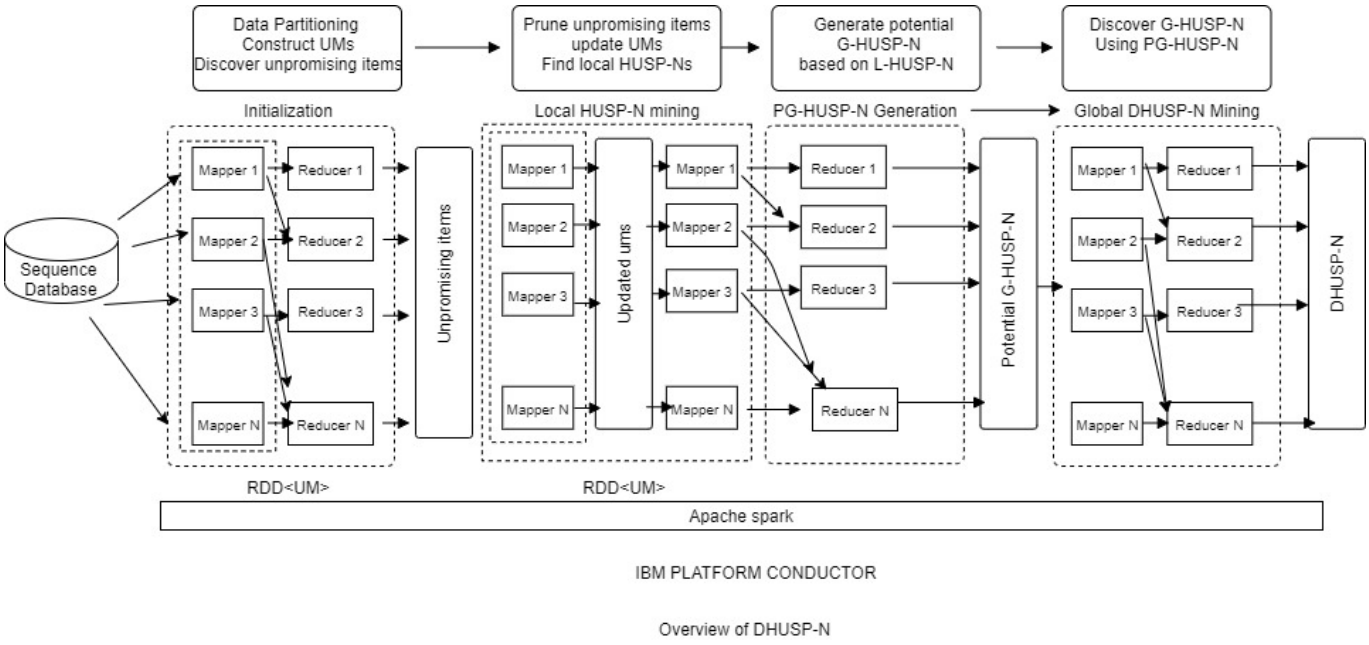
We propose an algorithm namely DHUSP-N (Distributed High utility sequential pattern mining with negative values). This algorithm mines high utility patterns in a distributed approach. Fig. 1 demonstrates the phases of the methodology. In the initialization phase, the sequence database is divided into many partitions. Each partition is given to a mapper which in turn gives a utility matrix [21]. The data structure UM [21] is used in later stages to retrieve utility values. This stage also identifies the items which do not form HUSP, which are pruned by DHUSP-N in the later stage.

1. Initialization phase: This comprises of two stages, namely, map and reduce.

Map stage: In each partition, utility matrix (UM) is constructed by the mapper for every input sequence in the given partition of database. Here UM refers to a data structure which contains utility values and the leftover utility of rest of items. This data structure is used to determine the LHUSP-N from each partitioned node. With this representation of utility matrix the mining takes place even faster. These are stored in Resilient Distributed Dataset (RDD). All elements in the database may not form high utility patterns. We use local sequence weighted utility (LSWU) and global sequence weighted utility (GSWU) to find the unpromising items which may not form good patterns. The pruned items are based on LSWU and GSWU values. The results are stored in a resilient distributed dataset.

Reduce Stage: Each reducer receive the output of the same key. So, basically, by adding all the LSWU values of the similar items, the reducer calculates the GSWU values of each item. The reducers emit the items for which the calculated GSWU values are less than the user supplied minimum utility threshold. These are referred as unpromising items. These are also stored and maintained in resilient distributed dataset which is used to update UM's in next phase.

2. Local HUSP-N mining: In this stage, the search space is reduced by pruning all the unnecessary items from each partition. Since it is hard to find global search space initially, we will find local HUSP-N from each partition then we will find DHUSP-N using this stage. Two map transformation stages play a key role in mining local HUSP-N. Map transformation 1 is used to prune unpromising items and map transformation 2 is used to find potential global HUSP. Rather than finding all the patterns in the partition with non-zero utility, we discover HUSPs locally. Pruning of low utility sequential patterns from the sequential patterns will not result in any loss of global



Overview of DHUSP-N

Fig. 1. Workflow of DHUSP-N.

HUSP. Thus, there is no loss of GHUSP while L-HUSP-N mining.

Map Transformation 1: In this map transformation the original UM's which is obtained from the previous stage results us the unpromising items. All the unpromising items in each utility matrix is pruned by mappers. The mappers output the updated utility matrix which will be stored in resilient distributed dataset.

Map Transformation 2: From the given input we have minimum utility threshold δ, partition D_i, updated Utility matrix and total utility, DHUSP-N applies HUSP-NIV [21] algorithm to find the local High Utility Sequential patterns with negative item values whose utility is greater than minimum utility threshold. Each mapper outputs the LHUSP's $< Patt, < D_i, utility >>$, where D_i is the id of partitioned database and utility is the utility of pattern patt in D_i. These are stored in resilient distributed dataset.

3. Discovering potential globally distributed HUSP-N: To find the potential global patterns, global utility of each local HUSP obtained from the L-HUSP-N phase needs to be determined. As the number of local High utility patterns are very large, we only consider potentially global patterns and prune all local HUSP's which are not PG-HUSP-N's. In this stage for whose maximum utility values is less than a certain threshold will be pruned resulting in potential GHUSP-N. Since maximum utility represents the upper bound of utility of pattern, for those patterns where the utility value is less than the threshold limit will be pruned. Continuously pruning with a threshold of maximum utility will not miss any high utility patterns.

Reduce stage: Every L-HUSP-N with same key is collected into same reducer. The reducers in this stage return PG-HUSP-N's whose utility exceeds the user supplied minimum utility threshold.

4. DHUSP-N Mining: The DHUSP-N mining process finds each patterns global utility in the given set. Given the set of PG-HUSP-N, it discovers GHUSP-N's. The reducer finds the sum of each patterns utility in the set after all possible GHUSP-N's are read. All patterns with a total utility greater than the threshold defined are returned as GHUSP-N's.

Map Stage: Given PG-HUSP-N's, each mapper finds the local utility of the patterns as follows: If certain pattern among PG-HUSP-N is local high utility sequential pattern in the given partition D_i, as we already obtained utility from the previous phase, the mapper outputs $< \alpha, < D_i, utility >>$. The mapper otherwise calculates the utility of α. To find α's utility in a partition, we build a pattern-growth approach that passes through the reduced search space. It undergoes both I-concatenate sequence and S-concatenate sequence.

Reduce stage: From the given set of PG-HUSP-N's, utility values are directed to the same reducer which has the same key. The reducer's input is a pattern, an utility in which the utility is the local utility resulting from map stage. Later, after reading each PG-HUSP-N, each pattern utility is added by the reducer. Finally, the patterns whose total utility exceeds the threshold are returned as Global DHUSP-N.

V. Experimental Results

To assess the efficiency of DHUSP-N, experiments have been run on two synthetic datasets and three real-world datasets. As this is the first of this kind there is no suitable algorithm to compare with DHUSP-N. The generic algorithm such as USPAN [23] is not appropriate to compare with DHUSP-N because it does not use negative values and it is a centralized approach. Even we cannot compare it with BIGHUSP [25] as it does not consider the negative values. Hence these algorithms are not suitable to compare with respect to run time or any other parameters.

TABLE IV. REAL DATASETS

Dataset	Sequence count	Item count	Average sequence length
Kosarak	990002	41270	8.099
BMSWebview2	77512	3340	4.62
MSNBC	989818	17	5.7

TABLE V. SYNTHETIC DATASET PARAMETERS

Parameter Name	Description
C	Average number of itemsets in each sequence
T	Average number of items in each itemset
D	Number of sequences (in millions)
N	Total number of items

TABLE VI. SYNTHETIC DATASETS

Dataset	C	T	D	N
C10T2.5D5N1000	10	2.5	5	1000
C15T3D10N10000	15	3	10	10000

The Distributed environment is equipped with 1 master node and 6 worker nodes. All the nodes are designated with Intel Xeon 2.6 GHz and 128Gb of RAM and the spark 3.0.0 is employed on the IBM platform conductor. A distributed platform is required for implementation. For this, we use apache spark distributed framework. This runs in a variety of platforms like the IBM platform for Spark [10], Hadoop, and Mesos clusters. We choose IBM Platform Conductor as it permits organizations to execute multiple instances of spark frameworks at the same time on a single infrastructure. It results in best usage of resources along with its efficient resource planning.

In DHUSP-N, we used the following parameters as performance measure: a) Run time: total time to mine DHUSP-N from the data set b) Number of candidates generated with varying utility c) scalability

A. Datasets

For this experiment, we used two synthetic datasets generated by IBM data generator and three real-time data sets, namely, Kosarak, BMSWebView2 and MSNBC. The real datasets are acquired from SPMF data mining libray.[1] The parameters of real datasets are given in Table IV. Table V depicts the parameters of synthetic data and the datasets are given in Table VI.

B. Effect of Minimum Utility

The performance of DHUSP-N is tested on real as well as synthetic datasets. Each experiment is conducted for distinct values of minimum utility threshold and the outcomes are reported in Fig. 2 and Fig. 3. Fig. 2 depicts the results on real datasets, whereas Fig. 3 describes the results on synthetic datasets. From the figures, it is clear that the execution time required for the completion of DHUSP-N is high at low values of minimum utility and tends to fall off with a rise in minimum utility. On Kosarak dataset, the execution time is

[1]https://www.philippe-fournier-viger.com/spmf/

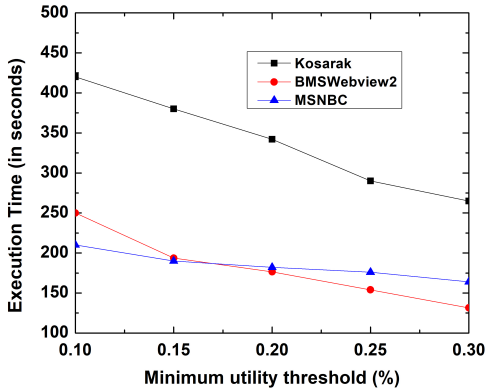


Fig. 2. Run Time Performance of DHUSP-N on Real Datasets.

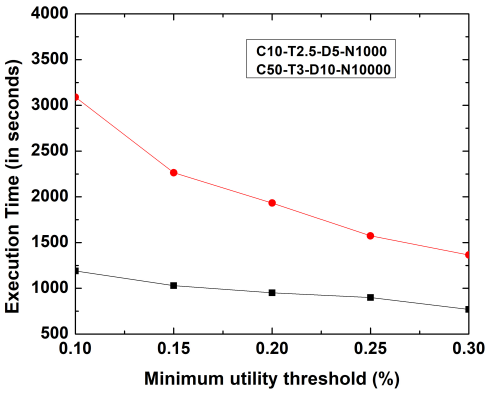


Fig. 3. Run Time Performance of DHUSP-N on Synthetic Datasets.

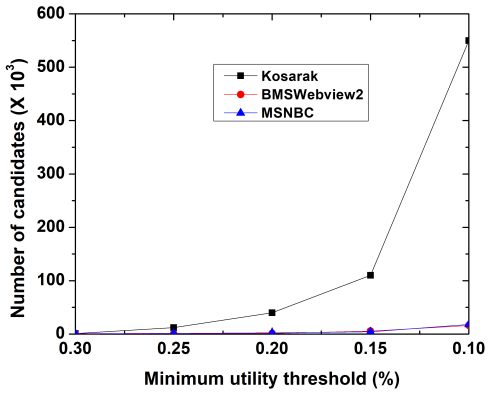


Fig. 4. Number of Candidates Generated on Real Datasets.

420 seconds for 0.1% threshold and it is 265 seconds for 0.3% threshold. Similarly on BMSWebview2 dataset, the execution time is 250 seconds for 0.1% threshold and it is 131 seconds for 0.3% thresholds, and it is 210 seconds and 164 seconds for 0.1% and 0.3% utility respectively on MSNBC dataset.

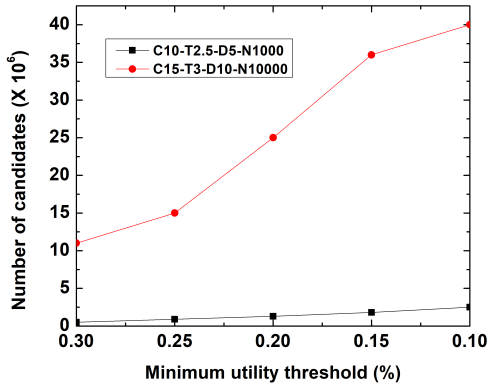


Fig. 5. Number of Candidates Generated on Synthetic Datasets.

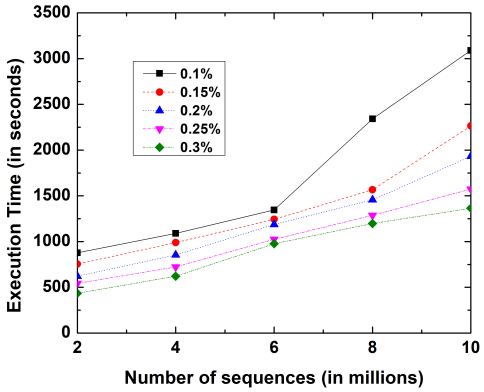


Fig. 7. Scalability Test on C15T3D10N10000.

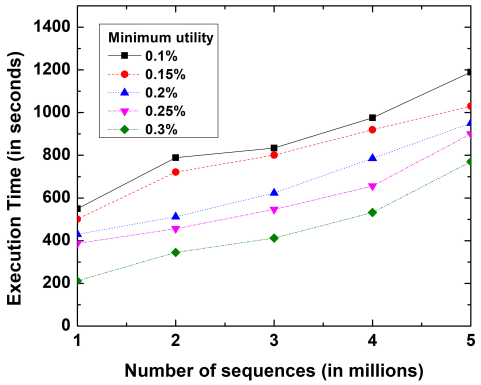


Fig. 6. Scalability Test on C10T2.5D5N1000.

To know the performance on large datasets, we conducted the experiment on two synthetic datasets having 5 million and 10 million sequences respectively. The former dataset require 1190 seconds for completion, whereas later dataset completed its execution in 3090 seconds. This is for 0.1% threshold and the execution times for the remaining thresholds are shown in Fig. 3. The number of candidates generated is also reported in Fig. 4 and Fig. 5. It is clearly noticed in Fig.5 that the candidates generated is high for larger dataset i.e. C15T3D10N10000. The reason is due to the increase in the sequence count from 5 to 10 and increase in the itemsets count per sequence from 10 to 15. Moreover, the difference in the execution time is high for lower values of minimum utility compared to higher values of minimum utility.

C. Scalability

To assess the scalability of DHUSP-N, we conducted the experiments on both the synthetic datasets. In case of C10T2.5D5N1000 dataset, initially we considered the first 1 million sequences and noted the execution time of the algorithm. Later, the database is scaled by 1 million sequences and repeated until 5 million sequences. Similarly, for C15T3D10N10000 dataset, the process is repeated from 2 to 10 million sequences in steps of 2 million sequences. The experiment is conducted for varying minimum utility. The results reported in Fig. 6 and Fig. 7 depicts the scalability of DHUSP-N. It is observed that the time for execution increase with the increase in the sequence count. The processing time increased significantly after 6 million sequences for 0.1% utility as shown in Fig. 7.

VI. Conclusion

This paper introduced a novel algorithm called DHUSP-N for mining high utility sequential patterns with negative values in a distributed environment. To our understanding, no methods were introduced in the utility mining literature to mine high utility sequential patterns with negative values in distributed environment. The performance of DHUSP-N is assessed on real as well as synthetic datasets. As this is the initial step of distributed approach to HUSP-N mining problem, there is a lot for the improvement as future work. More efficient data structures and techniques for pruning can be studied in the future. The current problem can be further extended to incremental mining of high utility negative sequential patterns [18]. Also, time intervals can be included in addition to the order of items purchased which leads to time interval high utility sequential pattern mining [19] with negative values.

Acknowledgment

The authors would like to thank the parent institute for the enormous support.

References

[1] Agrawal.R and Srikant.R, Mining sequential patterns, Mining Sequential Patterns. In: Proceedings of the Eleventh international conference on data engineering, 1995, pp. 3–14.

[2] Ahmed.C.F, Tanbeer.S.K, and Jeong.B, A novel approach for mining high-utility sequential patterns in sequence databases, ETRI Journal, vol. 32, 2010, pp. 676–686.

[3] Alkan.O.K and Karagoz.P, CRoM and HuspExt: Improving Efficiency of High Utility Sequential Pattern Extraction, IEEE Transactions on Knowledge and Data Engineering, vol. 27 (10), 2015, pp. 2645-2657.

[4] Aloysius, G. and Binu, D. An approach to products placement in supermarkets using PrefixSpan algorithm. Journal of King Saud University - Computer and Information Sciences, vol. 25 (1), 2013, pp.77-87.

[5] Ayres J., Flannick J., Gehrke J., and T. Yiu, Sequential pattern mining using a bitmap representation, In Proceedings of Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 429–435.

[6] Cao Y,. Zhao H,. Zhang D,. Luo C. Zhang, and E. Park, Flexible frameworks for actionable knowledge discovery, IEEE Transactions on Knowledge and Data Engineering, vol. 22 (9), 2010, pp. 1299–1312.

[7] Chen Jinlin, An UpDown Directed Acyclic Graph Approach for Sequential Pattern Mining. IEEE Transactions on Knowledge and Data Engineering, vol. 22 (7), 2010, 913–928.

[8] Chi-Truong, T. and Fournier-Viger, P. A Survey of High Utility Sequential Pattern Mining. Lecture Notes in Computer Science, 2019, 97–129.

[9] Dean J., and Ghemawat S. MapReduce: simplified data processing on large clusters. ACM Communications, vol. 51(1), 2008, pp. 107–113.

[10] Fu J., Sun J., and Wang K., SPARK – A Big Data Processing Platform for Machine Learning, In 2016 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), Wuhan, 2016, pp. 48-51.

[11] Guha, S., Rastogi, R. and Shim, K. Rock: A robust clustering algorithm for categorical attributes. Information Systems, vol. 25 (5), 2000, pp. 345–366.

[12] Kitchin R., Big Data. John Wiley and Sons, Ltd, 2016. [Online].Available:http://dx.doi.org/10.1002/9781118786352.wbieg0145.

[13] Lin Y. C., Wu, C.-W. and Tseng, V. S. Mining High Utility Itemsets in Big Data. Lecture Notes in Computer Science, 2015, pp. 649–661.

[14] Mabroukeh N.R., and Ezeife.C.I, A taxonomy of sequential pattern mining algorithms, ACM Computing Surveys, vol. 43 (1), 2010, pp. 3:1–3:41.

[15] Mooney C.H., and Roddick J.F., Sequential pattern mining approaches and algorithms, ACM Computing Surveys, vol. 45 (2), 2013, pp. 19:1–19:39.

[16] Pei J., Han J.W., Mortazavi-Asl B., and Pinto H., PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth, In Proceedings of International Confernece on Data Engineering, 2001, pp. 215–224.

[17] Saleti S., and Subramanyam R.B.V. A novel mapreduce algorithm for distributed mining of sequential patterns using co-occurrence information. Applied Intelligence, vol. 49, 2019, pp. 150–171.

[18] Saleti S., and Subramanyam R.B.V. A MapReduce solution for incremental mining of sequential patterns from big data. Expert systems with applications, vol. 133, 2019, pp.109-125.

[19] Saleti S., and Subramanyam R.B.V. Distributed mining of high utility time interval sequential patterns using mapreduce approach. Expert systems with applications, vol. 141, 2020.

[20] Wang J.Z., Yang Z.H., and Huang J.L., An efficient algorithm for high utility sequential pattern mining, Frontier and Innovation in Future Computing and Communications, Lecture Notes in Electrical Engineering, vol. 301(2014).

[21] Xu T., Dong X., Xu J., and Dong X. Mining High Utility Sequential Patterns with Negative Item Values. International Journal of Pattern Recognition and Artificial Intelligence, vol. 31 (10), 2017, 1750035.

[22] Xu T., Li T., and Dong X. Efficient High Utility Negative Sequential Patterns Mining in Smart Campus. IEEE Access, vol. 6, 2018, 23839–23847.

[23] Yin J., Zheng Z., and Cao L., Uspan: An efficient algorithm for mining high utility sequential patterns, In Proceedings of ACM SIGKDD, 2012, pp. 660–668.

[24] Zaki M.J., SPADE: An efficient algorithm for mining frequent sequences, Machine Learning, vol. 42 (1), 2001, pp. 31–60.

[25] Zihayat M., Hut Z. Z., An, A., and Hut, Y. Distributed and parallel high utility sequential pattern mining. In 2016 IEEE International Conference on Big Data, 2016, pp. 853-862.