Journal of Theoretical and Applied Information Technology
15th November 2020. Vol.98. No 21
© 2005 – ongoing JATIT & LLS
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

OBHUSPC - OPTIMAL BIG HIGH UTILITY SEQUENTIAL PATTERNS MINING WITH CUCKOO OPTIMIZATION USING HADOOP MAP REDUCE FRAMEWORK

[1]V. MALSORU, [2]A. R. NASEER, [3]G. NARSIMHA

[1] CSE, JITS Karimnagar, Affiliated to JNTUH, Telangana State, India

[2]SGCS, INHA University, Incheon, South Korea & SOCIE INHA University, Tashkent

[3]CSE, JNTUH College of Engg., Hyderabad, Telangana State, India

E-mail: [1]malsoru@gmail.com, [2]dr_arnaseer@hotmail.com, [3]narsimha06@gmail.com

ABSTRACT

The unprecedented explosion of information and technology areas in recent years has resulted in the generation of massive amount of data and extracting necessary and sensitive information from these huge amount of data has become a critical and challenging task. In this direction, several data mining approaches are proposed which lay the foundation for faster and efficient knowledge discovery. Big data High Utility Sequential Pattern (HUSP) Mining has turned out to be an important and essential data mining area where several investigations are carried out for effectual information retrieval. In this work, an effectual procedure for sequential pattern mining of Big High Utility Sequential Patterns along with the Binary Cuckoo Search Optimization (OBHUSPC) is proposed. Here Big High Utility Sequential Patterns mining approach is deployed to find the High Utility Sequential Patterns from big data and the Binary Cuckoo Search Optimization (BCSO) technique is used additionally to determine efficiently the finest high utility sequences. The proposed parallel method is implemented in Hadoop disseminated atmosphere to resolve the scalability difficulty and a transformed database is implemented to diminish the scanning time. The performance of the approach used in this work is evaluated using JAVA platform based Hadoop and Map-reduce framework with various big datasets.

Keywords: *Map-Reduce Framework, Data Mining, Big High Utility Sequential Patterns, Binary Cuckoo Search Optimization, Sequential Pattern Mining*

1. INTRODUCTION

Data mining has been an extensive field of research for exploring the data and understanding the highlights of various segments [1]. It is for the most part utilized for advertising examination, creation control, logical research, and soothsaying. It is the way toward removing fascinating (unimportant, verifiable, already obscure and valuable) structures from huge volumes of data [2] [3]. Elective names for data processing incorporate learning disclosure, learning extraction, framework examination, data archaic exploration, data mining, data gathering, and business insight [4]. The principle reason for data mining in information revelation is to separate valuable examples or rules from data sets [5]. Affiliation rule mining is broadly utilized for down to earth applications, for example, market publicizing, biomedical data applications, versatile data applications, and numerous different applications on account of the idea of the co-event of articles in the dataset. Constant mode mining is a technique for mining that breaks down a lot of successive arrangements of time characteristic occasions inside a data occasion [6][7]. A consistent strategy for data mining is a significant data mining practice that is generally utilized in numerous applications, for example, bioinformatics, client conduct suppositions in the exchange history, web mining and system attack interruption recognition [8]. Arrangement mining has been demonstrated beforehand as a constant computational model for profound learning [9].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

This is a developing field of data mining used to separate applicable models from huge data sets, where examples are exhibited in a solitary request [10]. Grouping, characterization, outer investigation, and technique mining are the most essential data mining errands [11]. Pattern mining includes finding intriguing, helpful and startling organizations in databases and in databases that can frequently discover an assortment of configurations, for example, things, affiliations, subcategories, arrangement rules, and occasional configurations [12]. Succession technique mining is a fundamental data mining issue that every now and again discovers sub-lines in a line database. Practically speaking, for an enormous number of structures in a database, a client may just be keen on a subset of structures. SPM is the way toward discovering successions in an arrangement database [13] [14].

Big Data HUSPs mining has been a challenging task as it requires the addressing of several prominent issues [15][16][17][18]. First, execution of HUSP mining approaches on a single machine is highly impossible due to the exponential growth of data in most of the application domains. Hence there is stronger need to design and implement parallel and distributed approaches which consider the task of decomposing the search space, minimizing the communication overhead and tackling the scalability difficulties. In this direction, important issue to be considered here is the computation of utility of a potential pattern over a set of all sequences in the given sequence database. The given huge sequence database needs to be divided into a number of smaller database partitions and each partition needs to be mapped to a node in a distributed platform in order to compute the local utility of a sequence in the partition database. Next we need to design a suitable strategy to combine the local utility of a sequence in all the partitions to generate efficiently a global utility of that sequence in the sequence database. Since we are dealing with massive databases containing huge amount of data termed as Big Data, HUSP mining of sequences in the sequence database encounters a combinatorial explosion of the search space which needs to be reduced drastically by using efficient search space pruning mechanisms.

Another interesting and fascinating direction of data mining research has been the use of efficient bio-inspired approaches such as genetic algorithms-GA[19][20][21][22], particle swarm optimization-PSO[23][24][25][26], ant colony optimization-ACO[27], bat algorithm-BA[28][29] and artificial bee colony optimization-ABC[30] for extracting essential and high value patterns. Recently, a meta-heuristic optimization algorithm termed as Cuckoo Search optimization (CSO) algorithm is developed to solve the problems related to optimization [31][32][33]. This algorithm is a bio-inspired algorithm based on the brood parasitism of certain cuckoo species along with random walks by Levy flights. It represents nature based simulation of laying and protecting of eggs of Cuckoo birds. In this process the cuckoo bird identifies a nest of a host bird which is safe and secure to lay eggs. The cuckoo birds lay eggs with a similar pattern of host birds' eggs where the host bird cannot identify the eggs laid by the cuckoo bird. In this algorithm, the selection of the cuckoos with eggs and performing random L'evy flights to lay the eggs will attract those initial host nests randomly. The quality of the nests will then be evaluated and compared to another random host nest. This would replace the old host nests, if the host nest is better considering the fitness levels.

In this paper, we propose a method OBHUSPC which utilizes the Big High Utility Sequential Patterns to store the database efficiently and reduce the scan time and a binary cuckoo search algorithm is used along with Big High Utility Sequential Patterns extraction to select the best set of items or rules. The remainder of the paper is organized as follows. Section 2 provides a brief review on HUSP approaches and the Map Reduce paradigm based distributed approaches. The proposed method OBHUSPC along with binary Cuckoo search optimization used to retrieve the finest high utility sequences from the given sequence database is presented in Section 3.. Section 4 presents results and discussion along with comparative analysis of the approaches. Finally, Conclusions and future directions are presented in Section 5..

© 2005 – ongoing JATIT & LLS
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2. LITERATURE REVIEW

High utility sequential pattern (HUSP) mining has garnered huge attention of the researchers quite recently and several approaches/techniques are proposed in the literature to extract high utility sequential patterns of Interest. UMSP approach proposed in [34] retrieves high utility mobile sequential patterns in mobile environments using an efficient MTS-tree structure for faster searching. It uses single value utility of a mobile sequential pattern corresponding to a location identifier associated with each itemset in a pattern. But it suffers from the limitation of handling only patterns of single items and single utility value per item. Mining of HUSPs from web log sequences proposed in [35] uses multiple value utility of a pattern and employs two tree structures UWAS-tree and IUWAS-tree for utility representation. But the sequencing of itemsets is not supported in this work. In order to satisfy the downward closure property for mining HUSPs, a metric Sequence Weighted Utility (SWU) defined as the sum of the utilities of all the sequences in the database containing the pattern is introduced in [36]. It proposes Utility Level (UL) algorithm for candidate generation and UtilitySpan (US) using pattern growth approach for HUSP mining, but does not provide a general framework. A general framework for mining HUSP from sequence databases is provided in [37]. USpan algorithm presented in this work uses Sequence Weighted Utility (SWU) and Sequence Weighted Downward Closure (SDCP) to retrieve patterns. Authors in [38] proposed a High utility itemsets mining from big data but this approach does not consider the sequential ordering of itemsets. A novel pruning approach based on the Cumulated Rest of Match (CRoM) based upper bound using High Utility Sequential Pattern Extraction (HuspExt) algorithm with efficient data structures is discussed in [39]. This technique allows higher conservative pruning before candidate pattern generation by defining tighter upper bound on the utility of the candidates. Simulations carried-out on huge data sets from various areas demonstrated that the proposed method productively finds high utility sequential patterns under low utility thresholds with various information qualities.

Mining Big data has been an emerging area of research and several distributed approaches based on Map Reduce paradigm are proposed in the literature to mine frequent sequential patterns. Novel approaches to parallelize projection based algorithms for identification of patterns on distributed-memory parallel computers are investigated in [40]. In this work, efficient task decomposition mechanisms using bipartite graph partitioning techniques are developed to achieve simultaneously load balancing and reduction in data sharing overheads among processors involved. The BIDE-MR presented in [16] is the parallel implementation of BIDE algorithm using Map Reduce paradigm on the Apache Hadoop cluster. A parallel implementation of apriori based approach on Hadoop platform for extracting high utility itemsets is proposed in [17]. PHUI-Miner proposed in [18] is a distributed parallel technique for mining high utility itemsets from huge datasets using pattern compression and sampling techniques to retrieve high utility itemsets. BigHUSP presented in [15] uses a distributed and parallel spark based platform to extract HUSPs from big data. A Map-Reduce framework based approach MR-INCSPM to perform the incremental sequential pattern mining is proposed in [41]. The technique proposed involves backward mining to detect stable sequences and introduces an efficient Co-occurrence Reverse Map (CRMAP) data structure to reduce the size of the search space thereby tackling the performance bottleneck by including efficient candidate generation rules and pruning properties

3. HIGH UTILITY SEQUENTIAL PATTERN MINING USING OBHUSPC

In this section, we present our proposed Optimal Big data High Utility Sequential Pattern mining methodology OBHUSPC used to find and determine the finest long high utility sequences in a given sequence database. The OBHUSPC approach uses BigHUSP technique alongwith Binary Cuckoo search optimization algorithm to generate and determine the best high utility long sequences. This OBHUSPC approach employs numerous Map Reduce steps to mine the Big data in a parallel manner to extract and identify the sequential patterns of high utility from big data efficiently. Hadoop is a disseminated master-slave structural design that encompasses the Hadoop distributed file system (HDFS) for storage and the map-reduce programming framework for computational competence. Here, the execution of the map-reduce framework is performed on apache based great clusters construct of service

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

hardware. The HDFS is used to accumulate the data on computing nodes that are offering high combined bandwidth in the cluster. The overall flow diagram of the proposed method is depicted in figure 1. The novelty of the proposed method is to perform Optimal Big High Utility Sequential Patterns (OBHUSPC) based sequential pattern mining algorithm with the help of the Binary Cuckoo search optimization approach. The OBHUSPC algorithm finds the set of rules or items then the best set of rules is selected by the binary cuckoo search algorithm.

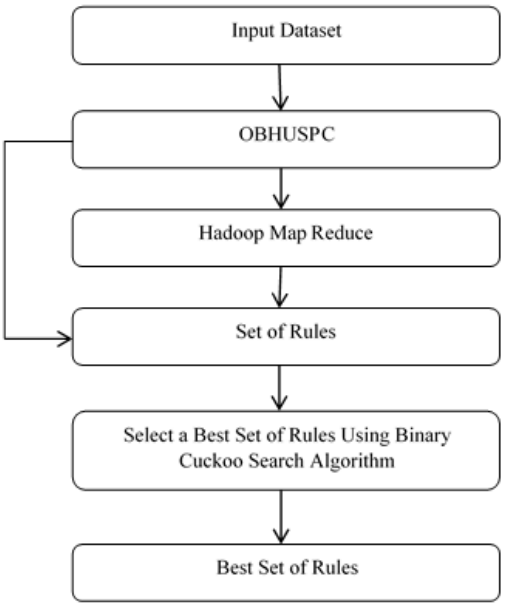

Figure 1: Proposed OBHUSPC based Hadoop Map Reduce Framework

3.1 Proposed Obhuspc Approach

The technique **OBHUSPC** used in this work is an improved version of the approach proposed in [15], but uses additionally the Binary Cuckoo Optimization procedure to determine the finest Global High Utility Sequential Patterns termed as GHUSPs.

In this approach, the given Sequence Database ***QSD*** is divided into '***n***' disjoint smaller database partitions ***QSD1, QSD2 , …., QSDn*** such that ***QSD = Union of QSDi where i=1, ..,n.QSDi != QSDj , for all i != j.*** In this Sequence Database ***QSD***, each item ***t*** in an itemset ***ts*** of a sequence ***p*** is associated with two positive values - internal utility value that is nothing but the quantity of that item and external utility value, that is nothing but price/unit profit. Then the Utility value (***UV***) of an item ***t*** in an Itemset ***ts*** of a sequence ***p*** is computed as the product of internal and external utility values of ***t***. The utility value of an Itemset ***ts*** in a super Itemset ***sts*** of a sequence ***p*** is computed as the sum of the utility values of all the occurrences of the Item ***t*** in the Itemset ***ts*** of a super itemset ***sts*** in a sequence ***p***. Hence, the utility of a sequence ***p*** in a super sequence ***Sp*** is computed as the maximum utility value in the set of all occurrences of ***p*** in ***Sp***, i.e., maximum utility value from the set of utility values of all occurrences of ***p*** in ***Sp*** is taken. The local utility value of a sequence ***p*** in a database partition ***QSDi*** is defined as the sum of the utility values of the sequence ***p*** in all the super-sequences ***Sp*** containing this sequence ***p*** in the database partition ***QSDi.*** The global utility of a sequence ***p*** in a Sequence database ***QSD*** is now defined as the sum of the local utility values of a sequence ***p*** in all the database partitions ***QSDi (where i=1 to n)*** of the Sequence database ***QSD***. Lastly, total utility ***TU*** of a database .partition, TU_{QSDi} is defined as the sum of the utility values of all the super-sequences in the database partition, ***QSDi***. Similarly, Total utility of ***QSD,*** TU_{QSD} is defined as the sum of the total utilities of all the database partitions ***QSDi (where i= 1 to n)***.

A sequence ***p*** in partition ***QSDi*** is said to be a Local High Utility Sequential Pattern(LHUSP) in the database partition ***QSDi*** if and only if Local utility of a sequence ***p*** in a database partition ***QSDi*** is greater than or equal to ***utp *** TU_{QSDi} where ***utp*** is the given minimum utility threshold percentage and TU_{QSDi} is the Total Utility of a database partition ***QSDi.*** Similarly, a sequence ***p*** in Sequence Database ***QSD*** is said to be a Global High Utility Sequential Pattern(GHUSP) in the ***QSD*** if and only if Global utility of a sequence ***p*** in Sequence Database ***QSD*** is greater than or equal to ***utp *** TU_{QSD} where ***utp*** is the given minimum utility threshold percentage and TU_{QSD} is the Total Utility of a sequence database ***QSD.*** The sum of the utility values of all the sequences having the sequence ***p*** in a database partition ***QSDi*** is termed as Local Sequence weighted Utility of a sequence ***p*** in a database partition ***QSDi.*** Similarly, the sum of the all the local sequence weighted utility of a sequence ***p*** in all the database partitions ***QSDi (where i=1 to n)*** belonging to the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

original Sequence Database ***QSD*** is termed as the Global Sequence weighted utility of a sequence ***p*** in a Sequence Database ***QSD***.

The maximum utility of a sequence pattern ***p*** in a database partition ***QSDi*** is defined as the local utility of a pattern ***p*** in ***QSDi***, if the local utility value of a pattern ***p*** in ***QSDi*** is greater than or equal to ***utp * TU_{QSDi}*** where ***utp*** is the given utility threshold percentage and ***TU_{QSDi}*** is the total utility of a partition ***QSDi.*** Similarly, the maximum utility of a sequence pattern ***p*** in a Sequence Database ***QSD*** is defined as the global utility of a pattern ***p*** in ***QSD***, if the global utility value of a pattern ***p*** in ***QSD*** is greater than or equal to ***utp * TU_{QSD}*** where ***utp*** is the given minimum utility threshold percentage and ***TU_{QSD}*** is the total utility of the Sequence Database ***QSD***.

The **OBHUSPC** implementation consists of three phases as depicted in Figures 2 & 3: Utility Value Matrix (UVM) generation phase, Local High Utility Sequence Pattern (LHUSP) Mining & Potential Global High Utility Sequence Pattern Generation phase and finally Global High Utility Sequence Pattern (GHUSP) mining phase. In the first phase, i.e., Utility Value Matrix generation phase, partitioning of the input sequence database ***QSD*** is carried out to generate ***n*** disjoint smaller databases or partitions ***QSD1, QSD2, …., QSDn*** such that ***QSD = Union of QSDi where i=1, ..,n.*** Each database partition ***QSDi*** is given to a mapper which transforms a sequence ***p*** in the partition to an efficient utility matrix data structure containing utility values that can be used by OBHUSPC in later phases without the need to consider the original Sequence Database ***QSD***. For each input sequence in the given database partition, each mapper generates a utility value matrix containing essential information required to mine Local High Utility Sequential Patterns ***(LHUSPs***). In this phase unhopeful items ***UHIs*** which cannot yield a HUSP are identified and are recorded for later pruning and will be removed in the second phase i.e., LHUSP Mining phase to reduce efficiently the search space. The utility value matrix structure plays an important role in speeding up the mining process as each element in the matrix has an item and one tuple per itemset in the sequence. A tuple per itemset consists of utility value of the item in the itemset and the balance utility of the remaining items in the sequence with respect to the item under consideration***.*** ***LHUSP*** mining phase uses the balance utility values to reduce the search space by removing unhopeful items. The Local Sequence weighted Utility value of each item ***k*** in a database partition ***QSDi*** is computed by each mapper and will generate an item ***k***-value pair (item ***k***, local sequence weighted utility of the item ***k*** in ***QSDi)*** which is then given as input to a reducer in the reduce stage. The corresponding reducer in the reduce stage now computes the Global Sequence weighted utility value of the item by summing up the local Sequence weighted utility value of the same item. In this fashion, each reducer generates the Global Sequence weighted utility value of all the items whose item-value pairs are received from the corresponding mapper. Now the reducer compares Global Sequence weighted utility value of these items with the given minimum threshold ***utp*** to determine the unhopeful items. If the Global Sequence weighted utility value of an item is less than the given minimum threshold ***utp***, then it is marked as unhopeful item UHI and recorded separately to be used in the second phase for pruning to reduce the search phase.

In the second phase of the approach, ***LHUSPs*** are mined in the Map stage using mappers and Potential ***GHUSPs*** are generated in the reduce stage using reducers. The first phase, i.e., utility value matrix generation phase has transformed each database partition ***QSDi*** to a set of updated Utility Value Matrices. The inputs to each mapper in the second phase are: given minimum utility threshold ***utp***, a set of updated Utility Value Matrices corresponding to each database partition ***QSDi*** which also includes information about unhopeful items ***UHIs***. and ***TU_{QSDi},*** the total utility of a database partition ***QSDi.*** In this mapping phase, as a first step, each mapper prunes all unhopeful items ***UHIs*** to reduce the local search space. As a second step, each mapper uses the pattern growth algorithm to mine the set of ***LHUSPs***. It generates a set of Local High Utility Sequential Patterns***(LHUSPs***) in the partition ***QSDi*** if and only if the Local utility of a sequence ***p*** in a partition ***QSDi*** is greater than or equal to ***utp * TU_{QSDi}*** where ***utp*** is the given utility threshold percentage and ***TU_{QSDi}*** is the total utility of a database partition ***QSDi.*** In this step of the mapping phase, each mapper generates local ***HUSPs*** as tuples where each tuple containing a

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

pair - (***p*** sequence pattern, ***uvQSDi*** utility value of the pattern in the database partition ***QSDi***). In order to identify ***Potential GHUSP***, it computes the maximum utility of every sequence ***p, muvQSDi*** in every database partition ***QSDi***. In the reduce stage of this phase, all the ***LHUSPs*** having the same pattern is given to the same reducer. It computes the maximum utility of every sequence ***p*** in a Sequence Database ***QSD***. For every pattern ***LHUSP p*** in the reducer, if there exists a tuple (***p, muvQSDi***) then the maximum utility of that sequence ***p*** in the Sequence Database ***QSD*** is increased by **m*uvQSDi***, otherwise it adds ***utp * TU_{QSDi}*** as the maximum utility of that pattern ***p*** in the database partition ***QSDi***. The reducer then generates ***Potential GHUSPs*** which are nothing but patterns whose maximum utility value in the Sequence Database ***QSD*** is greater than or equal to ***utp * TU_{QSD}*** where ***utp*** is the given minimum utility threshold percentage and ***TU_{QSD}*** is the total utility of a Sequence Database ***QSD.***

In third phase also known as ***GHUSPs*** mining phase, ***GHUSPs*** are identified from the given set of ***Potential GHUSPs*** generated by the second phase by computing the global total utility of each pattern in the set of ***potential GHUSPs***. In the map stage of this phase, each mapper computes the local utility of all the patterns in the input set of ***potential GHUSPs***. If the pattern ***p*** in the input set of ***Potential GHUPs*** is a ***LHUSP*** in database partition ***QSDi,*** then the mapper returns the utility of pattern ***p*** already computed in second phase. Otherwise, utility of the pattern ***p*** in that database partition ***QSDi*** is computed by the mapper using a pattern-growth algorithm that traverses the minimum search space represented by a lexical sequence tree where the nodes in the tree represent two types of patterns - I-concatenate and S-concatenate sequences generated by the sequence-extension step(S-step) and itemset-extension step(I-step) respectively. In the reduce stage of this phase, Global High Utility Sequence Patterns(***GHUPS***) are generated by considering all the patterns that have total utility greater than or equal to the threshold. Here the inputs to the reducers are the set of ***Potential GHUPs*** and their utility values. In this reduce stage, the same reducer is given the pairs with same pattern, pair pattern and its local utility generated by the mappers. The task of the reducer in this stage is to add up the utility of each pattern and select all the patterns having total utility value greater than or equal to the threshold and generate them as Global High Utility Sequence Patterns. In our ***OBHUSPC*** approach, Binary Cuckoo Optimization procedure is employed to determine the finest ***GHUSP*** sequences.

3.2 Binary Cuckoo Search Optimization (BCSO) Algorithm

The Cuckoo Search algorithm also known as meta-heuristic optimization algorithm which is developed recently to solve the problems related to optimization. This algorithm represents nature based simulation of laying and protecting of eggs of Cuckoo bird. In this process the cuckoo bird identifies a nest of a host bird which is safe and secure to lay eggs. The cuckoo birds lay eggs with a similar pattern of host birds' eggs where the host bird cannot identify the eggs laid by the cuckoo bird. This can be termed as identifying and executing the most frequent patterns in the nature. The Cuckoo search algorithm is used in our work to identify high utility sequence patterns from any big data. This algorithm is a bio-inspired algorithm based on the brood parasitism of certain cuckoo species along with random walks by Levy flights [42]. Cuckoo search uses the following representations - Every egg in a nest is a solution, and a cuckoo egg is a new solution. The aim is to replace a not-so-good solution in the nests with the new and potentially better solutions (cuckoos). Each nest has one egg, in the simplest form. The three simple rules of the cuckoo search algorithm can be stated as follows:

Rule 1: One egg is laid at a time into a nest chosen at random by each Cuckoo bird.

Rule 2: Best nests having high-quality eggs shall be forwarded to the next generations.

Rule 3: The number of available host nests is fixed and the host bird with a probability of $p_r \in [0, 1]$ finds the egg laid by a cuckoo. In such a scenario, either the host bird will ignore or damage the egg or actually leave the nest and create a whole new nest.

In this algorithm, the selection of the cuckoos with eggs and performing random L'evy flights to lay the eggs will attract those initial host nests randomly. The quality of the nests will then be

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

computed and compared to another random host nest. This would replace the old host nests, if the host nest is better considering the fitness levels.

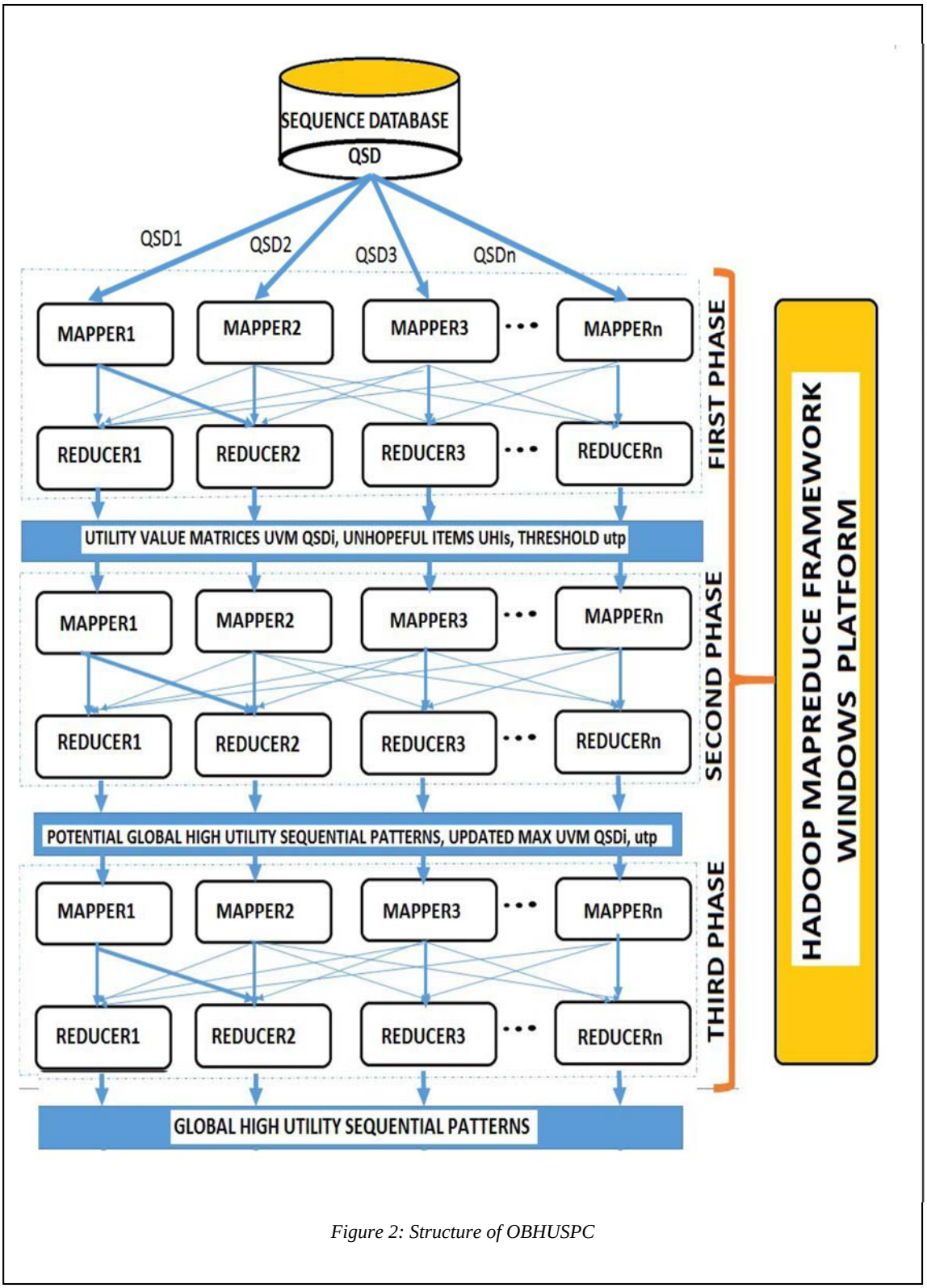

Figure 2: Structure of OBHUSPC

© 2005 – ongoing JATIT & LLS
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

FIRST PHASE of OBHUSPC :

- **PARTITIONING OF SEQUENCE DATABASE *QSD* into *QSDi (i=1,..,n)***

FOR EACH MAPPER IN FIRST PHASE
INPUTS : Sequence Database Partition QSDi
PURPOSE : ***UTILITY VALUE MATRIX GENERATION FOR QSDi***
TASKS PERFORMED : Computation of Local Sequence Weighted Utility value of sequences in ***QSDi,*** Total utility value TU_{QSDi} of a partition ***QSDi***, and generation of corresponding Utility Value Matrix UVM_{QSDi} representing ***QSDi***

FOR EACH REDUCER IN FIRST PHASE
INPUTS : Utility Value Matrix UVM_{QSDi} , given Minimum Threshold ***utp***, Total utility value of all partitions TU_{QSDi} ***(where i=1,…,n)***
PURPOSE : ***DETECTION OF UNHOPEFUL ITEMS which cannot form HUPs***
TASKS PERFORMED : Computation of Global Sequence Weighted utility value of sequences in QSDi, Total utility of Sequence Database TU_{QSD}, and Identification of unhopeful Items ***UHIs***

SECOND PHASE OF OBHUSPC

FOR EACH MAPPER IN SECOND PHASE
INPUTS : Utility Value Matrix, UVM_{QSDi} with unhopeful items ***UHIs***, given minimum Threshold ***utp***, total utility of Database partition TU_{QSDi} , Total utility of Sequence Database, TU_{QSD}
PURPOSE : ***MINING OF LOCAL HIGH UTILITY SEQUENTIAL PATTERNS(LHUSPs)***
TASKS PERFORMED: Pruning of unhopeful items, ***UHIs***, Generation of set of Local High utility sequential patterns **LHUSPs** in the partition ***QSDi,*** Computation of maximum utility of every sequence, ***muvQSDi*** in database partition ***QSDi*** , Updating the corresponding UVM_{QSDi}

FOR EACH REDUCER IN SECOND PHASE
INPUTS : Updated Utility Value Matrix UVM_{QSDi}, given minimum Threshold utp, Maximum utility of all sequences ***muvQSDi*** in all partitions ***QSDj where j =1,..,n, j!=i***
PURPOSE : ***GENERATION OF POTENTIAL GLOBAL HIGH UTILITY SEQUENTIAL PATTERNS***
TASKS PERFORMED : Computation of the maximum utility value of all sequences in ***QSD*** corresponding to the sequences in ***QSDi*** and generation of ***Potential GHUSPs***

THIRD PHASE OF OBHUSPC

FOR EACH MAPPER IN THIRD PHASE
INPUTS : Updated Utility Value Matrix UVM_{QSDi}, with Maximum utility value of all sequences in database partition ***QSDi***, ***Potential GHUSPs*** with maximum utility of the corresponding sequences in ***QSD***., given minimum Threshold ***utp***
PURPOSE : COMPUTATION OF TOTAL UTILITY VALUES
TASKS PERFORMED : Computation of the total utility values of all the ***Potential GHUSPs*** sequences in **QSDi**

FOR EACH REDUCER IN SECOND PHASE
INPUTS : All ***Potential GHUSPs*** in partition ***QSDi*** and Total utility values of all the ***Potential GHUPS*** from all the partitions ***QSDj (where j =1,…,n, j!= i)*** corresponding to the ***Potential GHUSPs*** in ***QSDi*** , given minimum Threshold ***utp***
PURPOSE : ***GENEARTION OF GLOBAL HIGH QUALITY SEQUENTIAL PATTERN IN SEQUENCE DATABASE QSD***
TASKS PERFORMED : Computation of the global total utility of all the ***Potential GHUPs*** and selection of all finest ***GHUSP*** sequences

Figure 3 : OBHUSPC Phases and Task Performed

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

This latest approach has one cuckoo's egg laid. If the host bird finds the Cuckoo's egg with a $P_\alpha \in (0 , 1)$, the host bird takes one of the two following actions -either Cuckoo's egg is thrown out of the nest or the host bird builds a new nest leaving the Cuckoo's egg in the old nest. This step is accomplished by replacing many solutions with the new random solutions [43][44]. The steps in the Cuckoo search algorithm can be described as follows:

Step 1: Generation of early population of n host nests N_i (i = 1, . . ., n)

Step 2 : Repeat the following sub steps (i) to (vii) until Max Generation or Stop Criterion is reached

(i) Obtain a cuckoo solution ci randomly by performing Levy flights, generate binary representation of solution and compute its Fitness Fit_{ci}

(ii) Select r nest among n (say, cj) randomly, generate binary representation of solution and compute its Fitness Fit_{cj}

(iii) If fitness of ci, $Fit_{ci} > Fit_{cj}$, fitness of cj then replace cj by the new solution.

(iv) Generate the binary representation of all the nests and compute their qualities

(v) Discard fraction (P_r) of poorer quality nests and build new nests at new locations via Levy flights, generate binary representation of solution and compute its Fitness

(vi) Retain the finest solutions

(vii) Generate ranking of the solutions and identify the best

Step 3: Ouput Best & Finest Solutions

The proposed OBHUSPC method uses Binary Cuckoo search optimization to extract and select the finest high utility sequential patterns which are more than or equal to the minimum threshold thereby forming the best rules like cuckoo bird selects the best nest by considering various parameters such as safe and the best nest, similarity in egg patterns which is in safe to continue its generation further.

4 Results and Discussion

In this work, we proposed OBHUSPC technique using high utility sequence pattern mining based on the BigHUSP along with the binary cuckoo search optimization. The implementation was done using JAVA platform with Hadoop Map reduce framework. The input dataset considered in this work is the big data. The test was carried out using a synthetic database and a real database with 4GB of main memory running the 64-bit version of Windows 10, i7 processor PC cluster. We tested our proposed algorithm with the Cleveland dataset and the retail dataset. The Cleveland dataset is available at UCI repository and the retail dataset is available at SPMF library. Original dataset of heart disease published in UCI repository contains 76 attributes in total. But in the experiments a subset of 14 parameters are referred. The retail dataset contains the total transaction count is 88162 and the individual item count is 16470. Here the average item count per transaction is 1030. The above datasets have been used in sequence pattern mining over streams.

Table I: Threshold Value-based time measures based on the mapper 2

Number of Mapper 2	
Threshold Value	Proposed OBHUSPC Time(msec)
10%	362453
20%	344632
30%	325461
40%	293538

The performance of our proposed study is given in the table I, the time to be measured based on the threshold value for the mapper 2. Here in the proposed method the threshold is varied as 10%, 20%, 30% and 40%. The results are tabulated as shown in table I.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

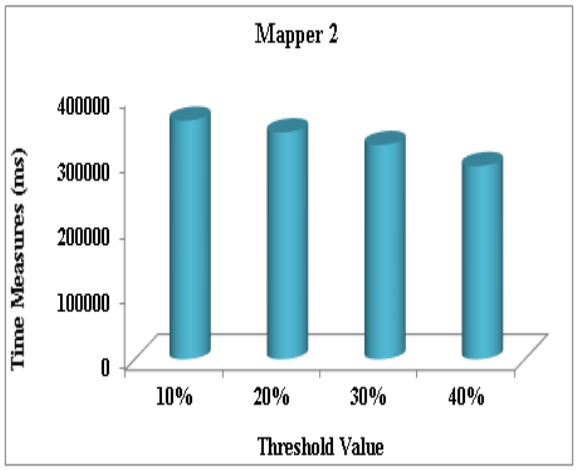

Figure 4: Graphical representation of time measures based on threshold value for mapper 2

Table I shows the threshold value-based time measures for mapper 2. Here we have considered the threshold values 10%, 20%, 30%, and 40% and the corresponding time measures for the proposed OBHUSPC approach are recorded to be 362453ms, 344632ms, 325461ms and 293538ms respectively. A Graphical representation of time measures based on threshold value for mapper 2 is shown in Figure 4. It is clearly seen from the graph that the time taken by our approach decreases as the threshold values are incremented.

Table II: Threshold value-based time measures for mapper 3

Number of Mapper 3	
Threshold Value	Proposed OBHUSPC Time(msec)
10%	282707
20%	274856
30%	262297
40%	246124

The threshold value-based time measures for mapper 3 are depicted in Table II. Here we considered the threshold values 10%, 20%, 30% and 40% and obtained the performance measures for proposed OBHUSPC approach using the mapper 3. We observed that our proposed approach achieves a time measure of 293538ms for 10% threshold, a time measure of 274856ms for a threshold of 20%, a time measure of 262297ms for a threshold of 30% and finally a time measure of 246124ms for 40% threshold value. Figure 5 shows the graphical representation of time measures based on different threshold values for mapper3. As it is evident from the graph that as the threshold value increases the time taken by our approach decreases to a larger extent.

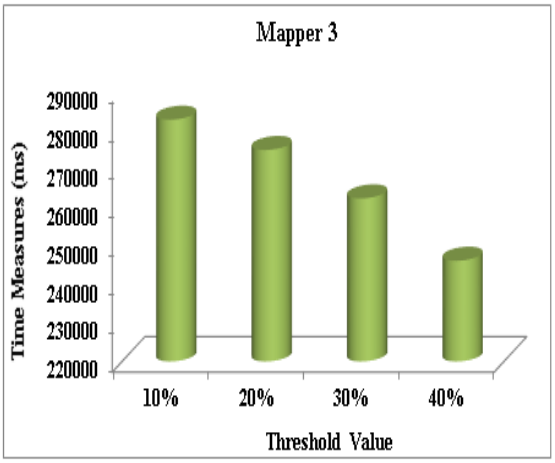

Figure 5: Graphical representation of time measures based on threshold value for mapper 3

Table III: Threshold value-based time measures for mapper 4

Number of Mapper 4	
Threshold Value	Proposed OBHUSPC Time(msec)
10%	253759
20%	242566
30%	237129
40%	216823

The table III depicts the threshold value-based time measures for mapper 4. Here we obtained the performance measures for our proposed

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

OBHUSPC approach using the mapper 4 by considering the threshold values 10%, 20%, 30% and 40%. As shown in the table, our proposed approach has taken an execution time of 253759 ms for 10% threshold, an execution time of 242566 ms for a threshold of 20%, an execution time of 237129 ms for a threshold of 30% and finally an execution time of 216823 ms for 40% threshold. Figure 6 shows the graphical representation of time measures based on different threshold values for mapper3. As it is evident from the below graph that as the threshold value increases the time taken by our approach decreases to a larger extent.

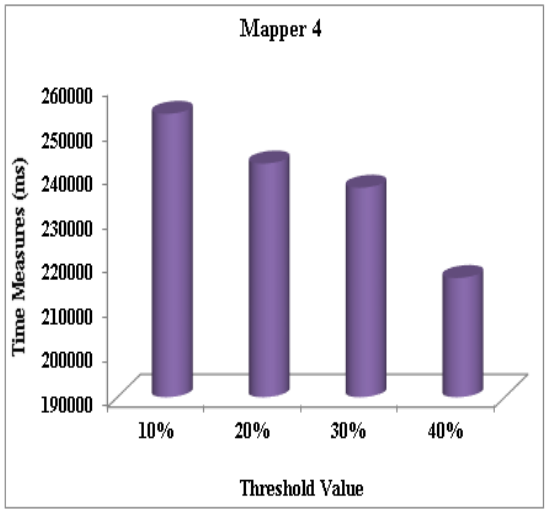

Figure 6: Graphical representation of time measures based on threshold value for mapper 4

4.1 Comparative Analysis

As it is evident from the experimental studies that our proposed pattern mining approach OBHUSPC achieves to a great extent very good results. Our approach uses traditional BHUSP technique along with a binary CSA. Here binary CSA is used for determining the best set of rules. In order to determine the efficiency of our proposed approach, we compared our approach with the existing Updown Directed Acyclic Graph (UDDAG)[45] implemented on Hadoop platform using the same mappers and measuring the time taken for different variations of the threshold. The UDDAG has been developed for efficient sequential pattern mining where UDDAG allows bidirectional pattern growth along both ends of the detected patterns. The UDDAG-based pattern mining approach first transforms a database based on frequent itemsets, then partitions the problem, and finally, detects each subset using UDDAG. The comparison experimental study was carried out using a synthetic database and a real database with 4GB of main memory running the 64-bit version of Windows 10, i7 processor PC cluster. Our proposed algorithm and the existing UDDAG approach were tested using the Cleveland dataset and the retail dataset. The comparison results are presented in the following Tables IV, V & VI.

Table IV: Comparison of OBHUSPC with existing UDDAG time measures for mapper 2

Number of mapper 2		
Threshold Value	Proposed OBHUSPC Time(ms)	Existing UDDAG Time(ms)
10%	362453	365874
20%	344632	345156
30%	325461	326589
40%	293538	297846

Table IV shows the comparison of OBHUSPC approach with existing UDDAG time measures for mapper 2. Here we have used the threshold values 10%, 20%, 30%, and 40% and the corresponding scanning time measures for the proposed OBHUSPC and the existing UDDAG approach are recorded. A Graphical representation of proposed and existing approach time measures based on threshold value for mapper 2 is shown in Figure 7. It is clearly seen from the graph that the scanning time taken by our approach is lower than that of the existing UDDAG approach.

The threshold value-based scanning time measures providing comparison of OBHUSPC with existing UDDAG approach for mapper 3 are depicted in Table V. Here we considered the threshold values 10%, 20%, 30% and 40% and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

obtained the performance measures for proposed OBHUSPC approach and existing UDDAG approach using the mapper 3. Figure 8 shows the graphical representation of proposed and existing UDDAG approach scanning time measures based on different threshold values for mapper3. As it is evident from the graph that the scanning time taken by our approach is much lower than that of the existing UDDAG approach.

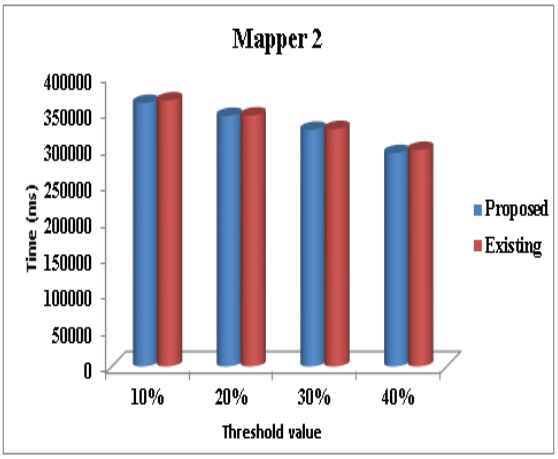

Figure 7: Graphical representation of proposed and existing approach time measures- Mapper2

Table V: Comparison of OBHUSPC with existing UDDAG time measures for mapper 3

Number of mapper 3		
Threshold Value	Proposed OBHUSPC Time(ms)	Existing UDDAG Time(ms)
10%	282707	284596
20 %	274856	278488
30%	262297	265999
40%	246124	248777

The table VI depicts the threshold value-based time measures exhibiting the comparison of OBHUSPC with the existing UDDAG approach for mapper 4. Here we obtained the performance measures for our proposed OBHUSPC approach and the existing UDDAG approach using the mapper 4 by considering the threshold values 10%, 20%, 30% and 40%. Figure 9 shows the graphical representation of proposed and existing approach scanning time measures based on different threshold values for mapper4.

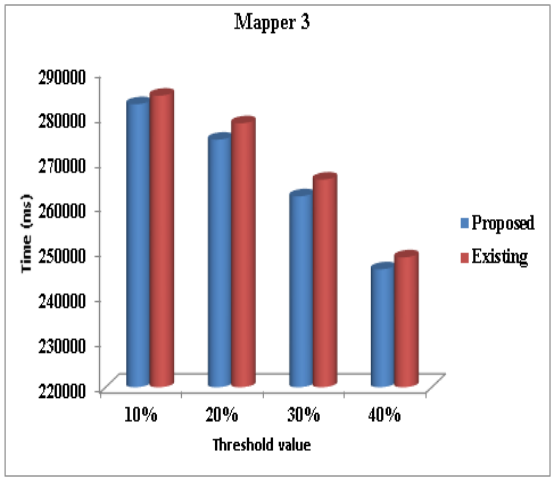

Figure 8: Graphical representation of proposed and existing approach time measures- Mapper3

Table VI : Comparison of OBHUSPC with existing UDDAG time measures for mapper 4

Number of mapper 4		
Threshold Value	Proposed OBHUSPC Time (ms)	Existing UDDAG Time (ms)
10%	253759	254876
20%	242566	245887
30%	237129	241168
40%	216823	219986

As it is evident from the graph that the scanning time taken by our approach is much lower than that of the existing UDDAG approach for varying threshold values. From the above experimental results, it is clear that the proposed OBHUSPC algorithm reduces the scanning time when

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

compared with the existing UDDAG method for varying threshold values and for all the mappers used. Since our approach is implemented in Hadoop distributed environment, it reduces the scalability problem to a larger extent.

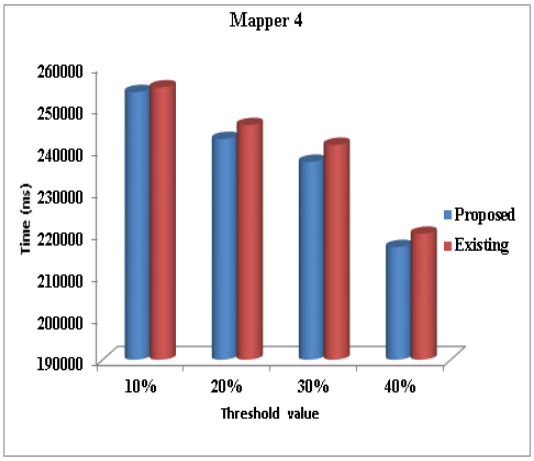

Figure 9: Graphical representation of proposed and existing approach time measures- Mapper4

The main objective of this experimental study is to demonstrate the effectiveness of the Bio-inspired approaches in determining the finest high utility patterns in Big data sets. Our approach is an enhanced version of the technique proposed in [15], but uses the Binary Cuckoo Search Optimization (BCSO) algorithm to extract the best global high utility sequential patterns. The BCSO algorithm is a bio-inspired algorithm based on the brood parasitism of certain cuckoo species along with random walks by Levy flights. It represents nature based simulation of laying and protecting of eggs of Cuckoo birds. The second objective of this study is to perform comparative analysis of this proposed bio-inspired approach with our Two Way Distributed Sequential Pattern Mining methodology employing Updown Directed Acyclic Graph (UDDAG) along with Fruit fly optimization algorithm to extract the sequential patterns proposed in [45]. This distributed model uses the Fruit fly bio-inspired algorithm to extract the enhanced consequences that will take least amount of execution time to complete a progression. In order to carry out proper comparative analysis, both our proposed models are implemented using Windows JAVA platform with Hadoop Map Reduce framework. It is evident from the experimental results that the proposed Binary Cuckoo Search based OBHUSPC algorithm reduces the scanning time when compared with the existing Fruit fly based UDDAG method for varying threshold values and for all the mappers used.

5 CONCLUSION

An innovative approach OBHUSPC of Optimal Big High Utility Sequential Patterns mining using Binary Cuckoo Search Optimization procedure to determine High Utility Sequential Patterns from Big Data efficiently is presented in this paper. Here Binary Cuckoo Search Optimization procedure is exploited to determine the finest high utility sequences. Our approach resolves the scalability difficulty to a larger extent as our procedure is implemented in a Hadoop distributed environment. In this procedure, OBHUSPC employs numerous Map Reduce steps to mine data in a parallel manner. It partitions the input sequence database into numerous divisions and processes every division of the input sequence database separately. Two innovative and disseminated policies are introduced to successfully shorten the search space and significantly improve the performance of OBHUSPC. The performance of the proposed technique is evaluated based on scanning time. From the experimental results, it is clear that the proposed OBHUSPC approach reduces the scanning time when compared with the existing UDDAG method for varying utility threshold values for all the mappers used. The results obtained demonstrated the efficiency and considerable better performance of the proposed technique over the existing UDDAG methods. In the future, the proposed work can be used to implement various other Utility Sequential Pattern mining algorithms for enhancing the performance and capabilities of the proposed method.

REFERENCES

[1] Z. Zeng,, B. Zhou., & C. Li, (2019). Research on Intrusion Data Mining Algorithm Based on Multiple Minimum Support. 2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS).

[2] C. Jalota, & R. Agrawal, (2019). Analysis of Educational Data Mining using

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Classification.2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon).

[3] B. Denham, R. Pears, & M.A. Naeem, (2019). HDSM: A distributed data mining approach to classifying vertically distributed data streams. Knowledge-Based Systems, 105114.

[4] W. Hadi,, N. El-Khalili,, M. Alnashashibi, G. Issa, & A. A. Banna, (2019). Application of data mining algorithms for improving stress prediction of automobile drivers: A case study in Jordan. Computers in Biology and Medicine, 103474.

[5] U. Yun, G. Lee, & E. Yoon, (2019). The advanced approach of sliding window-based erasable pattern mining with a list structure of industrial fields. Information Sciences, 494, 37–59.

[6] D. Zhang, K. Lee, & I. Lee, (2019). Semantic periodic pattern mining from Spatio-temporal trajectories. Information Sciences, 502, 164–189.

[7] U. Yun, H. Nam, G. Lee, & E. Yoon, (2019). An efficient approach for incremental high utility pattern mining with indexed list structure. Future Generation Computer Systems, 95, 221–239.

[8] W. X. Wilcke., V. de Boer, M. T. M.de Kleijn,, F.A.H. van Harmelen., & H.J. Scholten, (2018). User-centric pattern mining on knowledge graphs: An archaeological case study. Journal of Web Semantics.

[9] B. Le, U. Huynh, & D. T. Dinh, (2018). A pure array structure and parallel strategy for high-utility sequential pattern mining. Expert Systems with Applications, 104, 107–120.

[10] Z. Shou, & X. Di, (2018). Similarity analysis of frequent sequential activity pattern mining. Transportation Research Part C: Emerging Technologies, 96, 122–143.

[11] U. Yun, D. Kim, E. Yoon, & H. Fujita, (2018). Damped window-based high average utility pattern mining over data streams. Knowledge-Based Systems, 144, 188–205.

[12] M. Xu, J. Singla., E. I. Tocheva., Y. W. Chang,, R. C. Stevens, G. J. Jensen, & F Alber, (2019). De Novo Structural Pattern Mining in Cellular Electron Cryotomograms Structure.

[13] B. Peromingo., D. Caballero, A. Rodríguez, A. Caro, & M. Rodríguez, (2019). Application of data mining techniques to predict the production of aflatoxin B1 in dry-cured ham.Food Control, 106884.

[14] H. Y. Lin, & S. Y. Yang, (2019). A cloud-based energy data mining information agent system based on big data analysis technology. Microelectronics Reliability, 97, 66–78.

[15] M. Zihayat, Z.Z. Hut, A. An, Y. Hut, (2016) "Distributed and parallel high utility sequential pattern mining", 2016 IEEE International Conference on Big Data (Big Data).

[16] D. Yu, W. Wu, S. Zheng, and Z. Zhu, *BIDE-Based Parallel Mining of Frequent Closed Sequences with MapReduce*.Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 177–186.

[17] J.-W. Huang, S.-C. Lin, and M.-S. Chen, *DPSP: Distributed Progressive Sequential Pattern Mining on the Cloud*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 27–34

[18] Ying Chun Lin, C. Wu, and V. S. Tseng, "Mining High Utility Itemsets in Big Data", PAKDD 2015, Part II, LNAI 9078, pp. 649–661, 2015. DOI: 10.1007/978-3-319-18032-8_51

[19] W. Song, A.C. Huang, "Mining High Utility Itemsets Using Bio-Inspired Algorithms: A Diverse Optimal Value Framework", IEEE Access, vol.6, pp. 19568-19582, March 20118 DOI 10.1109/ACCESS.2018.2819162

[20] J.-P. Huang, C.-T. Yang, and C.-H. Fu, ``A genetic algorithm based searching of maximal frequent itemsets," in *Proc. Int. Conf. Artif. Intell.*, 2004, pp. 548_554.

[21] S. Kannimuthu and K. Premalatha, ``Discovery of high utility itemsets using genetic algorithm with ranked mutation," *Appl. Artif. Intel.*, vol. 28, no. 4, pp. 337_359, Apr. 2014.

[22] D. Martín, J. Alcalá-Fdez, A. Rosete, and F. Herrera, ``NICGAR: A Niching Genetic Algorithm to mine a diverse set of interesting quantitative association rules," *Inf. Sci.*, vols. 355_356, pp. 208_228, Aug. 2016.

[23] R. Pears and Y. S. Koh, ``Weighted association rule mining using particle swarm optimization," in *Proc. PAKDD Workshop Bio-Inspired Technol. Data Mining*, 2011, pp. 327_338.

[24] J. Gou, F. Wang, and W. Luo, ``Mining fuzzy association rules based on parallel particle swarm optimization algorithm," *Intell. Autom. Soft Comput.*, vol. 21, no. 2, pp. 147_162, Apr. 2015.

[25] J. C.-W. Lin *et al.*, ``Mining high-utility itemsets based on particle swarm

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

optimization," *Eng. Appl. Artif. Intell.*, vol. 55, pp. 320_330, Oct. 2016.

[26] J. C.-W. Lin, L. Yang, P. Fournier-Viger, T.-P. Hong, and M. Voznak, ``A binary PSO approach to mine high-utility itemsets," *Soft Comput.*,vol. 21, no. 17, pp. 5103_5121, Sep. 2017.

[27] R.S. Parpinelli; H.S. Lopes; A. A. Freitas, "Data mining with an ant colony optimization algorithm, "IEEE Transactions on Evolutionary Computation (Volume: 6 , Issue: 4 , Aug 2002), Publisher: IEEE

[28] K. E. Heraguemi, N. Kamel, and H. Drias, ``Association rule mining based on bat algorithm," *J. Comput. Theor. Nanosci.*, vol. 12, no. 7, pp. 1195_1200, Jul. 2015.

[29] K. E. Heraguemi, N. Kamel, and H. Drias, ``Multi-swarm bat algorithm for association rule mining using multiple cooperative strategies," *Appl. Intell.*, vol. 45, no. 4, pp. 1021_1033, Dec. 2016.

[30] M.A.M. Shukran, et. al., "Artificial Bee Colony based Data Mining Algorithms for Classification Tasks", Modern Applied Science, Vol. 5, No. 4; August 201, pp. 217-231, *Published by Canadian Center of Science and Education*

[31] R. Rajabioun, "Cuckoo Optimization Algorithm," *Applied Soft Computing*, vol. 11, pp. 5508–5518, 2011.

[32] X.-S. Yang, S. Deb, "Cuckoo search: recent advances and applications", Neural Computing & Applications, 24 (1) (2014) 169–174.

[33] S.A. Medjahed, A. Benyettou, "Binary Cuckoo Search Algorithm for Band Selection in Hyperspectral Image Classification", IAENG International Journal of Computer Science, July 2015

[34] B.-E. Shie, H.-F. Hsiao, V. S. Tseng, and P. S. Yu, "Mining high utility mobile sequential patterns in mobile commerce environments,"in Proceedings of the 16th International Conference on Database Systems for Advanced Applications - Volume Part I, ser.DASFAA'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp.224–238.

[35] C. Ahmed, S. Tanbeer, and B.-S. Jeong, "Mining high utility web access sequences in dynamic web log data," in Software Engineering AI Networking and Parallel/Distributed Computing (SNPD), 2010 11th ACIS International Conference, June 2010, pp.76–81.

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

[37] J. Yin, Z. Zheng, and L. Cao, "Uspan: An efficient algorithm for mining high utility sequential patterns," in Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD '12. New York, NY, USA: ACM, 2012, pp. 660–668. [Online]. Available: http://doi.acm.org/10.1145/2339530.2339636

[38] Y. C. Lin, C.-W. Wu, and V. S. Tseng, *Mining High Utility Itemsets in Big Data*. Cham: Springer International Publishing, 2015, pp. 649–661.

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

[40] V. Guralnik and G. Karypis, "Parallel tree-projection-based sequence mining algorithms," *Parallel Comput.*, vol. 30, no. 4, pp. 443–472, Apr. 2004. [Online]. Available:http://dx.doi.org/10.1016/j.parco.2004.03.003

[41] Saleti, S. and Subramanyam, R.B.V., 2019. A Map Reduce solution for incremental mining of sequential patterns from big data.Expert Systems with Applications, 133, pp.109-125.

[42] X.-S. Yang, S. Deb, "Cuckoo search via Lévy flights" , in: World Congress on Nature & Biologically Inspired Computing, 2009, NaBIC 2009, IEEE, 2009, pp.210–214.

[43] S. Mahmoudi, R. Rajabioun, S. Lotfi, "Binary Cuckoo Optimization Algorithm", 1st Iranian Conference on Information Retrieval" Oct. 2013

[44] J. García, F Altimiras, A. Peña,G. Astorga,and O. Peredo**, "**A Binary Cuckoo Search Big Data Algorithm Applied to Large-Scale Crew Scheduling Problems", Volume 2018, Article

[45] V. Malsoru, A. R. Naseer A. R., G. Narsimha, "Two Way Distributed Sequential Pattern Mining using Fruitfly Algorithm along with Hadoop and Map Reduce Frame Work", International Journal of Computer Aided Engineering and Technology - Inderscience, DOI: 10.1504/ IJCAET.2021 .10018474,http://www.inderscience.com/jhome.php?jcode=ijcaet