International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 12, December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

ACCURATE AND EFFICIENT MINING FOR CONFIDENCE COLOSSAL PATTERNS FROM HIGH DIMENSIONAL DATASETS: CDFP-MINE

J. Krishna[1], Dr. P. Suryanarayana Babu[2]

[1] CSE Department, Research Scholar RU Kurnool, Assistant Professor, AITS, Rajampet, AP, (India)
[2]Director SVDDC, RAYALASEEMA UNIVERSITY, KURNOOL, AP,(India)

ABSTRACT

*CDFP-Mine, a novel approach for finding huge Colossal Pattern Sequences (CPS) from High Dimensional Biological Datasets is talked about in this paper. CDFP-Mine has successfully found Determinate Frequent Patterns (DFP) which is additionally advanced into a DFPT$^+$ tree to produce CPS with vector intersection operator. CDFP-Mine influences utilization of a novel incorporated data structure called Hyperstructure 'H-struct', as a blend of a data matrix and one-dimensional arrays exhibit as a pair to powerfully find DFP from Biological High Dimensional Datasets. DFPT+ tree is developed as Bitwise Top-Down Column identification tree. H-struct has an assorted element to encourage is, it has amazingly restricted and precisely predictable primary memory and runs rapidly in memory based requirements. The algorithm is planned such that it takes just a single scan at the database to find CPS. The exact investigation on CDFP-Mine demonstrates that the proposed approach achieves a superior mining effectiveness on different high dimensional datasets and beats Colossal Pattern Miner(CPM) and BVBUC in various settings. The execution of CDFP-Mine on the high dimensional dataset is assessed with Accuracy and Frequency measure*s.

Keywords: Bitwise Vertical Bottom Up Colossal mining(BVBUC), Colossal Pattern Miner(CPM), Colossal Pattern Sequences, Determinate Frequent Patterns(DFP), DFPT$^+$ tree

I. INTRODUCTION

Present day Computational Biology known as Bioinformatics investigation is increasing much significance in the extraction of learning from biological high dimensional datasets. The Bioinformatics has created different vital algorithms for biological information examination. The best part is its solid association with Medicine. The advancement in Medical innovation in a decade ago has presented another type of datasets called biological datasets understood as Microarray and Gene Expression Datasets. In contrast to transactional datasets, those unreasonable dimensional databases normally have few rows(samples) and an enormous wide assortment of columns(genes). Truth is told, from the genome arrangements or biology framework, the fundamental endeavor is too aware of useful qualities for intense assessment. In bioinformatics, the researcher can utilize the advances in computational biology to explore enormous and complex datasets. KDD and Data Mining have worried as an unavoidable need to separate valuable data and information from these datasets.

Next in Data Mining introduction, Frequent Pattern Mining(FPM) picked up as an unmistakable information mining paradigm that helps to separate patterns that thoughtfully symbolize relationship among discrete attributes and plays out a basic part in data mining and information investigation assignments and in additional applications. Fundamentally in view of the multifaceted design of those relations, select sorts of patterns can emerge. The most widely recognized sort of patterns has a tendency to mine successive patterns[5][6], association rules[1][2], correlations[4], episodic[3], clustering[11], classification[10][12], and maximal patterns and frequently closed patterns[7]–[9].

There are various algorithms produced for frequent patterns quick and proficient mining, which are classified into two classes. The first class candidate era strategy, including apriori[2] and its observations, are in perspective of apriori-property[2]: if a specimen altogether isn't normal, at that point it's incredible example can't visit. The apriori-based arrangement of standards accomplished proper lessening around the estimated hopeful sets. In spite of the way that, when there are many frequent patterns or possibly lengthy patterns, it will take multiple scans over the colossal database to develop candidate sets. The second class, pattern-growth strategy, which incorporates FP-Growth, furthermore makes utilization of the apriori-property. Be that as it may, it recursively partitions the database into sub-databases to generate candidate sets. It makes restricted scans over the database.

II. LITERATURE SURVEY

Within the literature, numerous algorithms had been developed under pattern growth method for discovering frequent patterns and closed patterns[8][14][15]. It uses enumeration primarily based approaches[8][15][16], wherein object mixtures are searched for frequent colossal patterns. In view of this, their running time increases exponentially with growing the average duration of the data and makes minimal two scans over the database. These will consume huge extent of memory utilization and predictably takes sufficient time when memory primarily based constraints are present. Those algorithms are rendering to be impractical on high dimensional microarray datasets. The entire sets of frequent closed patterns have received the use of row enumeration space turned into first shown in[16], which was also found in[13].

Although, the present frequent pattern mining strategies nonetheless encounter the subsequent difficulties.

- All object enumeration based totally mining strategies are based on singleton patterns and take lots time to compute those patterns.
- Massive primary memory is required for powerful mining. While memory constraints are present, an apriori-like set of rules will no longer be powerful because it produces large candidates for lengthy patterns. Sufficient memory space is needed to keep candidate sets for discovering frequent patterns of various lengths. Fp-growth evades candidate generation with the aid of condensing into an FP-tree.

- Real-time databases keep all of the instances. Most of the datasets in real time programs are either sparse or dense. It's far hard to pick a right mining method on the fly which fits all instances.
- Real-time programs require high dimensional and scalable. Numerous current procedures are powerful for smaller size data sets. However, as the dataset size will increase the existing strategies suggests fit falls on core data structures and requires sufficient memory.
- Multiple scans over the database. Most of the existing Apriori and FP-Growth strategies make numerous scans over the databases. Efficient data garage systems are needed to keep intermediate results.

Row enumeration search can be explored by way of building projected database recursively[17]. The vertical bottom-up method is enabled to mine efficient huge styles of large size[18]. However, there is a want to consider column enumeration algorithms certain in lots of algorithms are proposed to mine frequent colossal patterns. However, for high dimensional datasets, the pattern mining problem consumes greater time and space. If a dataset is with one hundred rows and one thousand columns, the present enumeration algorithms work properly if a threshold is set to low while discovering colossal patterns and frequently generates a big variety of discovered patterns without an appropriate data. However, traditional fpm techniques are having suit falls in handling high dimensional datasets due to its dimensionality, length and primary memory utilization. Those pretenses a novel mission on design and developing a new method which is efficient in pattern mining on big databases in which space requirement is restricted. For that reason, Determinate Frequent Pattern(DFP) mining is taken into consideration for studying biological datasets.

Table-1. Sample database

Row_{id}	Genes
S_1	G_1, G_2, G_3, G_5, G_7, G_8, G_9
S_2	G_1, G_3, G_4, G_5, G_6, G_8, G_{10}
S_3	G_2, G_5, G_6, G_7, G_8
S_4	G_1, G_2, G_3, G_4, G_5, G_6, G_7, G_{11}
S_5	G_1, G_2, G_4, G_6, G_7
S_6	G_2, $G_{5,}$ G_7, G_8, G_9, G_{10}, G_{11}

III.MOTIVATION AND CONTRIBUTION

The improvements in Bioinformatics added to the advancements of new datasets called High Dimensional Datasets. Investigation of Genetic structures like DNA, RNA, and Protein groupings from Biological datasets will grow new advancements in diagnosis of medical. To do this CPSs are to be found and Determinate Frequent Pattern(DFP) mining is considered as extremely useful for these datasets analysis. The issue of DFP mining is to locate the entire arrangement of Colossal Pattern Sequences in a given biological high dimensional dataset. The fundamental goal is to find all Colossal Pattern Sequences in a given biological high dimensional dataset D with respect to user min_sup threshold value.

An effective new algorithm CDFP-Mine that is uniquely intended to find vast Colossal Pattern Sequences over biological high dimensional datasets is portrayed, in this paper. CDFP-Mine influences utilization of another data structure called Hyperstrucure(H-struct) which can be utilized to find Colossal Pattern groupings by performing attribute enumeration as depth-first row-wise, and proficiently decreases the searching time over the dataset. CDFP-Mine has the accompanying stages; initial, a determinate frequent pattern revelation algorithm is proposed for the diminished datasets utilizing H-struct that can fit into the memory. Second, CDFP-Mine utilizes another property list column vector based intersection operator to find pattern arrangements efficiently by decreasing the search time and database scans. The test comes about demonstrate that this approach creates better outcomes when mining biological high dimensional datasets and outflanks Colossal Pattern Miner on various settings.

IV.BASIC PRELIMINARIES

Let G = {G_1,$G_{2,}$….,G_m} be an arrangement of m gene attributes. An X is a subset of attributes with the end goal that X ⊆ G. So, G = {G_1,G_2 ,… ..,G_m} is also denoted as G = G_1,G_2, … ,G_m. Let S = {S_1, S_2, … .., S_n} be a set rows speaking to trial conditions opposed over organic dataset, where every S_i is an arrangement of n subsets called genes. Each row in S distinguishes a subset of things. S = (Row_{id}, X) is a two-tuple, where R_{id} is a row-id and X is an attribute. S = (Row_{id}, X) is said to contain Y attribute if and just if Y ⊆ X. Table-1 demonstrates a case of the dataset in which the genes are spoken to from g_1 to g_{11}. Give the initial two sections of Table-1 a chance to be our example data collection. Table-I demonstrates a database is the arrangement of trial conditions. Every S_i contains a subset of genes spoke to in lexicographic request. The main objective is to discover all Colossal Pattern Sequences in a given biological high dimensional dataset DB with regard to user minimum support threshold.

Definition-1: A **support(s)** is defined as the number of transactions in database that contains both A and B, represented as its frequency. Support (A◦B) = P(XUY)

Definition-2: Confidence(c) of the Rule A◦B is true in the database, if it contains the number of transactions

containing A that also contains B, represented as Confidence(A◦B) = P(B|A) = P(A∪B)|P(A)

Definition-3: The **Relative frequency** of an attributes, A, B is contained in database, the relative frequency is defined as

Relative frequency (RF) = support(A,B)/support(A)

Definition-4: An **Association Rule** is an inference of the form A → B between two attributes X and Y where A, B ∈ I and A ∩ B = ∅, which satisfies user supplied Support s and Confidence c.

Definition-5: Determinate Frequent Pattern: A determinate frequent pattern can be frequent if both items in the set are frequent by themselves. A determinate frequent pattern set (A, B) is frequent if both A and B in the set is also frequent and it is true in database, if it is having its min_sup above 2.

Definition-6: Colossal Pattern Sequence: An attribute set A $\subseteq$ I, is a pattern sequence, if and only if sup(A) $\geq$ min_sup and must be a determinate frequent pattern.

V.RELATED WORK

For a given set highlights in the biological high dimensional dataset, we have a tendency to characterize a gene articulation matrix(M) with m X n. Table-2 demonstrates a bit framework of M, that is identical to gene articulation matrix of the DB, where 1-signifies 'overexpressed' and 0-signifies 'underexpressed.' A transaction of gene articulation information is identified with 'overexpressed' information.

By performing column-wise prune on gene articulation matrix M in view of minimum support and wipe out the columns whose aggregate occurrences are less than minimum support. Table-3 shows that the pruned gene articulation matrix with the minimum support is three.

Table-2. A sample database of Gene articulation matrix(M) with Support Count(SC)

Row$_{id}$	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8	G_9	G_{10}	G_{11}
S_1	1	1	1	0	1	0	1	1	1	0	0
S_2	1	0	1	1	1	1	0	1	0	1	0
S_3	0	1	0	0	1	1	1	1	0	0	0
S_4	1	1	1	1	1	1	1	0	0	0	1
S_5	1	1	0	1	0	1	1	0	0	0	0
S_6	0	1	0	0	1	0	1	1	1	1	1
Support Count (SC)	4	5	3	3	5	4	5	4	2	2	2

Table-3 Gene articulation matrix pruned with 3 as min_sup

Row$_{id}$	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8
S_1	1	1	1	0	1	0	1	1
S_2	1	0	1	1	1	1	0	1
S_3	0	1	0	0	1	1	1	1
S_4	1	1	1	1	1	1	1	0
S_5	1	1	0	1	0	1	1	0
S_6	0	1	0	0	1	0	1	1
Support Count (SC)	4	5	3	3	5	4	5	4

The support is given as the frequency of the rows in the dataset that contains a set of features G'. The relative frequency of rows in the dataset that contain A is called its support of A, A ⊆ I for a given item set is the recurrence of rows in the dataset that consists A. For a set of patterns, ∃ a colossal pattern sequences with a maximum length.

VI.CDFP-MINE

In this segment, we study efficient mining of colossal pattern sequences from biological high dimensional dataset. The mining process of CDFP-Mine illustrated in first subsection with an example and the next, algorithm of the CDFP-Mine.

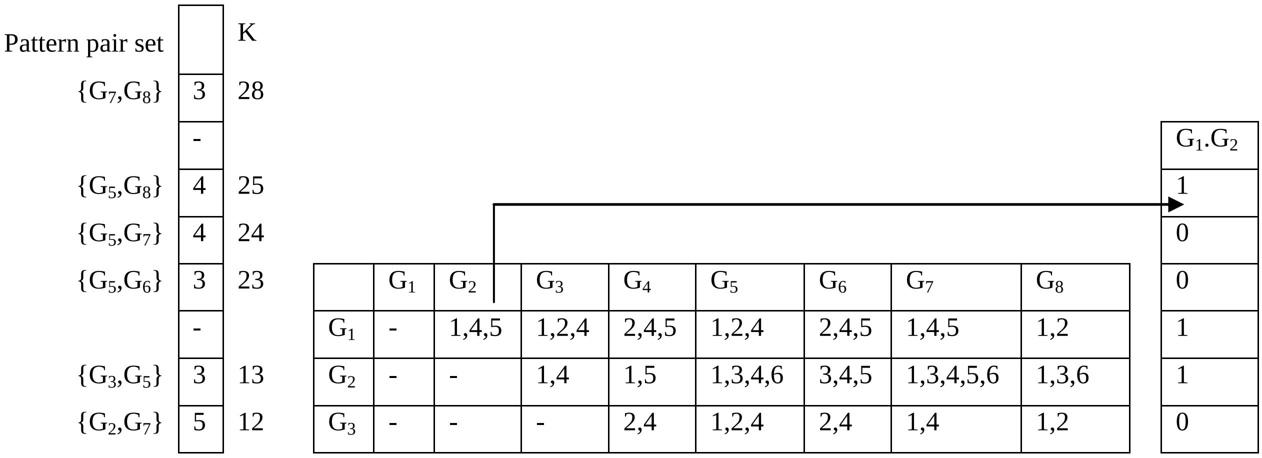
6.1 Discovering colossal pattern sequences and mining process

Within the literature, there are distinctive approaches to investigate the biological high dimensional datasets. High dimensional databases characterized as experimental conditions as rows and huge gene variables as columns. This extraordinary feature will decrease the quantity experimental conditions in pattern mining procedure through building a data matrix search techniques. Row enumeration algorithms work properly when the dataset size is low dimensions.

Horizontal search method can't do efficient mining of patterns for the reason that possibility of discovering the exponential order of items. H-struct matrix is built the use of vertical seek approach as shown in Fig-1. For the identical gene expression data in Table-1 with minimum support = 3, we introduce a determinate frequent pattern mining method for mining colossal pattern sequences. This method explores the concept of vector databases as shown in Fig-1.

6.1.1 Finding determinate frequent patterns

Using vertical search strategies, construct a data matrix such that each attributes is a bitwise column vector and their corresponding genes are in the all rows of this column vector. Now scan the dataset and mark the row number corresponding to each row and column. Each entry in the matrix is a column vector contains set of bit fields and storing with binary values. Its support values are stored in triple count array as shown in the Fig-1.



Pattern pair set		K
$\{G_7,G_8\}$	3	28
	-	
$\{G_5,G_8\}$	4	25
$\{G_5,G_7\}$	4	24
$\{G_5,G_6\}$	3	23
	-	
$\{G_3,G_5\}$	3	13
$\{G_2,G_7\}$	5	12

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8
G_1	-	1,4,5	1,2,4	2,4,5	1,2,4	2,4,5	1,4,5	1,2
G_2	-	-	1,4	1,5	1,3,4,6	3,4,5	1,3,4,5,6	1,3,6
G_3	-	-	-	2,4	1,2,4	2,4	1,4	1,2

$G_1.G_2$
1
0
0
1
1
0

	-	
{G_2,G_5}	4	10
	-	
{G_1,G_7}	3	6
{G_1,G_6}	3	5
{G_1,G_5}	3	4
{G_1,G_4}	3	3
{G_1,G_3}	3	2
{G_1,G_2}	3	1

a) One Dimensional Triple Array Pair Set

G_4	-	-	-	-	2,4	2,4,5	4,5	2
G_5	-	-	-	-	-	2,3,4	1,3,4,6	1,2,3,6
G_6	-	-	-	-	-	-	3,4,5	2,3
G_7	-	-	-	-	-	-	-	1,3,6
G_8	-	-	-	-	-	-	-	-

b) 2-level determinate frequent pattern sequences

G_5.G_8
1
1
1
0
0
1

c) Column vector

Figure-1. Hyper structure Matrix with column vector database and triple pair set array

6.1.2 One dimensional triple array pair set

In general, the ARM algorithms keep up various item count recurrence esteems all through a look over a database. For example, it's fundamental to have sufficient primary memory to store each pattern count that the quantity time's sets of a pattern pair happen inside the database. It's difficult to update a 1 to a count set where the counting groups are held on various locations of memory and troublesome in loading the page to primary memory. In these cases, the algorithms will be ease back to find that count of pattern pair in primary memory since it requires additional overhead on handling time and expands a time to discover set of a frequent pattern. In this manner, it's hard to count an esteem that necessities enough fundamental memory. When it includes high-dimensional datasets, it's hard all to maintain in one memory.

To optimize main memory, a pattern pair*(i, j)* occurrence in the dataset should be counted in one place. If the CPS order is $i < j$, and uses only one entry M[*i*,*j*] in two dimensional array M. This approach makes half of the array as useless. Count Array(CA) is a more efficient way to store CPSs in memory.

A count array is defined as a one-dimensional triple array set which will store a count as *CA*[*k*] for the pair*(i,j),* with $1 \leq i < j \leq n$, where $k = (i-1)\left(n - \frac{i}{2}\right) + (j - i)$

To discover colossal pattern sequences, CDFP-Mine performs an iterative depth first search (DFS) on column enumeration strategy. By imposing backtracking search order on column sets, we are able to perform a systematic search over colossal pattern sequences.

6.1.3 Pruning the search space by creating a DFPT$^+$ tree

The gene sequence be R discovered from data matrix, R-is gene in database which exclusively contains a particular gene and its Row$_{id}$ count must be above the min support threshold as shown in Table 4. The discovered determinate pattern pair sets can be divided into 7 non-overlap subsets based on the data matrix.

Each non-cover subset is changed over into DFPT$^+$ tree. The tree is built as a phylogenetic tree. Presently it is conceivable to list gene G_1, with the end goal that a non-covering subset which has a place with pattern G_1. Then again for every gene G_1, if G_1 has a place with all rows of C_i which condition g1 will make gene database about

G_1. A vertical top-down tree is developed in a reality that the size of the corresponding pattern of a node G_1 in a vertical top-down search tree DFPT$^+$ is not as much as the span of any of its siblings' comparing to G_1. So in each branch of this tree, the size of the pattern is more prominent than the size of patterns which are delivered in level i of the branch. Since the level one of a tree contains DFP with min_sup. On the off chance that we investigate the tree just to min_sup level; it can find all the colossal patterns of the dataset. The DFPT$^+$ tree seeks just min_sup level of a tree which is conditioned explicitly on non-covering set and prunes its branch. Fig.2 indicates sub tree for the arrangement of a subset that containing only G_1 and another sub tree for the arrangement of subsets containing only G_2.

Table-4. A gene DFP database

Sl. No.	Genes	Row$_{id}$ numbers	On conditioned
1.	G_1	1,2,4,5	{G_2, G_3, G_4, G_5, G_6, G_7}
2.	G_2	1,3,4,5,6	{G_5, G_6, G_7, G_8}
3.	G_3	1,2,4	{G_5}
4.	G_4	2,4,5	{G_6}
5.	G_5	1,2,3,4,6	{G_6, G_7, G8}
6.	G_6	3,4,5	{G_7}
7.	G_7	1,3,6	{G_8}

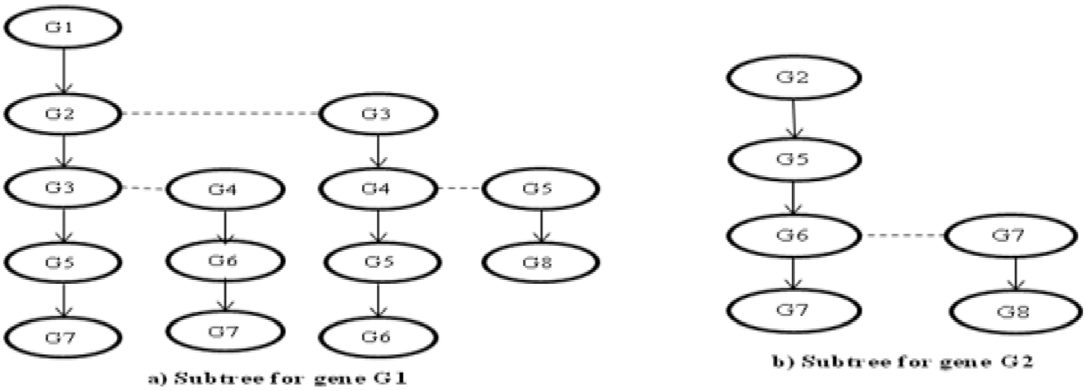


Figure-2. DFPT$^+$ sub trees of gene G_1 and G_2

6.1.4 Discovering (CPS) colossal pattern sequences

We can extend each i-level pattern pair sets to frame a new bitwise column vector to decide the equal to CPS which are frequent or not from the discovered gene DFP databases. In the search strategy of column enumeration, each pattern sequence is a segment set and its supplement gene is those which there are on the whole rows of this segment set. By performing the intersection operation on column vectors to decide the corresponding pattern sequence which results a column bit vector. Pattern sequence discovery contains only gene G_1, and after that containing just G_2 gene and so on. The remaining mining procedure can be performed on H-struct, just without referring the original database. For every DFP there is a k value.

Within the above instance, the pair set $G_1•G_2$ may be explored on G_3 to create a brand new pattern pair set as $G_1•G_2$ and G_3. It plays bitwise vertical intersection on $G_1•G_2$ and G_3 and discovers a brand new DFP pair set. Its corresponding count value is stored on a 3-level triple count array. $G_1•G_2$ and G_3 isn't identical to both $G_1•G_2$ or G_3. Consequently, it's also known as colossal patterns and its count is 2, which is stored in a triple array. Figure-3 shows min_sup pruned DPT+ tree for the dataset of Table-1. Based totally in this tree we will construct a pruned bitwise vector tree for discovering colossal pattern sequences. We will expand every level 2 node of this tree and construct its children and go on increasing to subsequent level with min_sup. By using performing Column vector intersection and expanding to next level, this procedure is repeated recursively with a brute force backtracking and forwarding, we are able to find out long colossal pattern sequences which are colossal patterns as in Table-5.

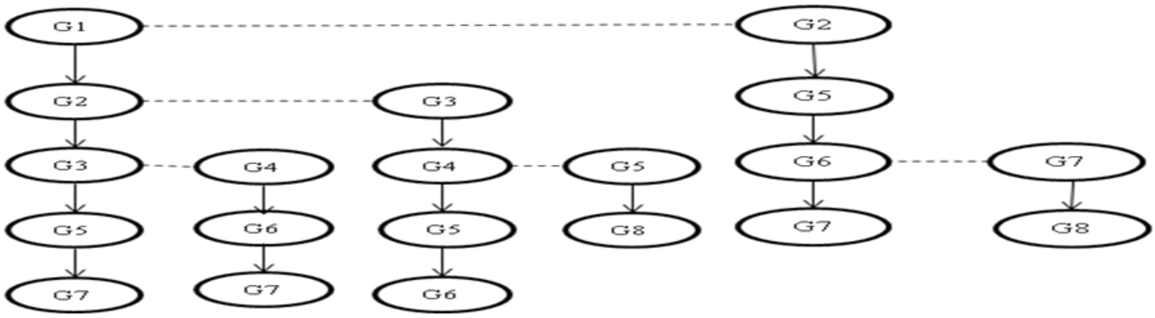


Figure-3. *DFPT*$^+$ Tree for discovering colossal pattern sequences

Table-5. Pattern sequences generated using column vector intersection operator.

Sl. No.	(CPS) Colossal Pattern Sequences
1	$\{G_1, G_2, G_3, G_5, G_7\}$
2	$\{G_1, G_2, G_4, G_6, G_7\}$
3	$\{G_1, G_3, G_4, G_5, G_6\}$
4	$\{G_2, G_5, G_6, G_7\}$
5	$\{G_2, G_5, G_7, G_8\}$
6	$\{G_1, G_3, G_5, G_8\}$
7	$\{G_1, G_2, G_7\}$

8	{G_1, G_4, G_6}
9	{G_2, G_5, G_7}
10	{G_2, G_6, G_7}
11	{G_5, G_6, G_8}
12	{G_2, G_7}
13	{G_5, G_8}

6.2 Accuracy levels are measured for the discovered CPS

In discovering of CPS from DFP mining, the level of the rightness of our algorithm is measured utilizing the "Frequency" and "Accuracy" measures to assess the general execution; these are described by utilizing the formulas.

Let A→B be the found determinate pattern sequence, at that point

$$\text{Accuracy Measure (ACC) of } (A, B) = P(AB) + P(\rightarrow A \rightarrow B)$$

$$Frequency\ Measeure\ (FM)\,of\ (A,B) = \frac{2*P\left(\frac{B}{A}\right)*P\left(\frac{A}{B}\right)}{P\left(\frac{B}{A}\right)+P\left(\frac{A}{B}\right)}$$

6.3 CDFP-Mine algorithm

In a given gene database, a relevance frequency(RF), the problem of mining the set of DFPs can be considered as partitioning into *n*-subproblems. The problem of partitioning can be performed recursively that is each subset of mining can be further divided when necessary. This forms a divide and prune framework. To mine the subsets of mining, we construct corresponding DFP Databases.

For each remaining attribute *i* in *Ai*, starting from it recursively calls CDFPmine(*i X*, *DT*|*i* , *Ai* ,*CPS*) to build its *i -level DFP* Database *DT*|*i* and discover all its patterns using dynamically created count array.

Input: Gene Database and relevance frequency as min support threshold

Output: the complete set of Colossal Patterns

Method:

1. Initialize CP←0, let CP as set of colossal patterns
2. Scan the database and compute data matrix and discover all DFPs pair set and create a DFP database *DB*
3. Call CDFPmine(*0*,*DB*,A_i,*CP*)

Procedure CDFPmine(iX, DT|i, Ai, CP)

Let **iX :** the DFPs if DB is x-DFP database,

DT|i: DFP Database

Ai: Attribute list.

DP: Relevant Frequency (RF) distribution over frequent pattern

```
1.  N← LENGTH($DT|_i$);
2.  CP←empty vector of length N+1;
3.  $iX$← data vector of length N+1;initially all 0.0;
4.  $iX$[0]←2.0
5.          for i = 0 to n do
                    for j=0 to i-1 do
                    CP← $DT|_i$ [j:i]
                    Initialize w←LENGTH(CP)
                    if DP[CP] $\otimes$ $iX$ [i-w] $\geq$ $iX$ [i] then
                    $iX$ [i]← DP[CP] * $iX$ [i-w];
                    CP[i] ← CP;
6.  for i = 0 to N
7.  if (level i is min_sup) then
8.  while i>0 do
            if (CP[i] is colossal) then insert CP[i] onto child of i;
            call CDFPmine($iX$, $DT|_i$, $A_i$, $CP$)
9.  i← i-LENGTH(CP[i])
10. return($iX$ [i], CP)
```

VII.RESULTS & PERFORMANCE ANALYSIS

In this segment, we will modify the execution of our algorithm with Colossal Pattern Miner(CPM) and BVBUC. The run-time is measured as elapsed time and IO seeks for time. CPM has demonstrated its better execution on finding Colossal Pattern Sequences which is an enumeration based algorithm. We executed this algorithm and contrast our technique with them. In our execution consider, we utilized the various size of the datasets; it is hard to evaluate the min_sup threshold as an absolute value. Rather, the min_sup limit dictated by relative frequency (RF). To compare the algorithm, experiments performed on five real datasets from UCI[19]. Table-6 demonstrates the characteristic data about the datasets.

Table-6. Test datasets and their Characteristics

Name of Dataset	**#genes**	**#samples**
Diabetes	17	768
Heart	28	303
Breast-Cancer	25	699
Prostate-Cancer	12600	102
Lung-Cancer	12533	181

Table-7 and Table-8 demonstrate the consequence of running three algorithms CDFP-Mine, CPM and BUBVC on a real standard dataset Lung-Cancer(LC) and Prostate-Cancer(PC). It seemed that with expanding min_sup all the algorithm executed in the dataset will be diminished.

In regularly FPM algorithms, it is observed that the performance is poor when min_sup is lower value. In this way when the min_sup is small, CDFP-Mine has an effective mining proficiency. From Fig. 4 the performance on Lung-Cancer(LC), the distinction of the effectiveness of CDFP-Mine with CPM and BVBUC is very much when the min_sup is low value. From the Fig. 5 the performance on Prostate-Cancer(PC), it is observed that CDFP-Mine has a most extreme distinction in running time with CPM and least with BVBUC when the support is minimum value.

Table-7. Performance on Lung-Cancer (in sec)

Minimum Support	CDFP-Mine	Colossal Pattern Miner (CPM)	BVBUC
0.1	06	41	21
0.08	09	52	24
0.06	28	65	39
0.05	35	98	58
0.03	49	116	72

Table 8. Performance on Prostate-Cancer (in sec)

Minimum Support	CDFP-Mine	Colossal Pattern Miner (CPM)	BVBUC
0.1	11	67	21
0.08	15	70	26
0.06	31	75	37
0.05	44	86	48
0.03	48	89	50

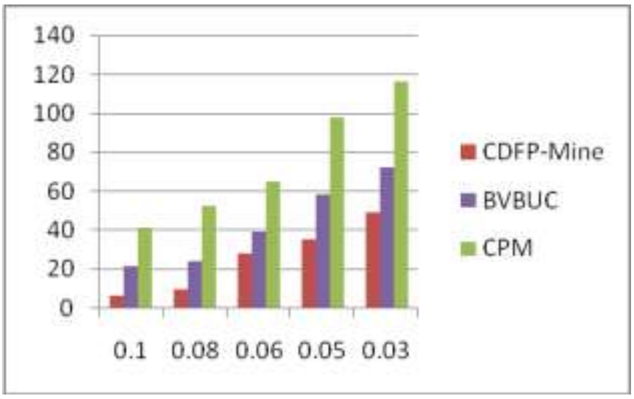


Figure-4. Performance on LC (sec)

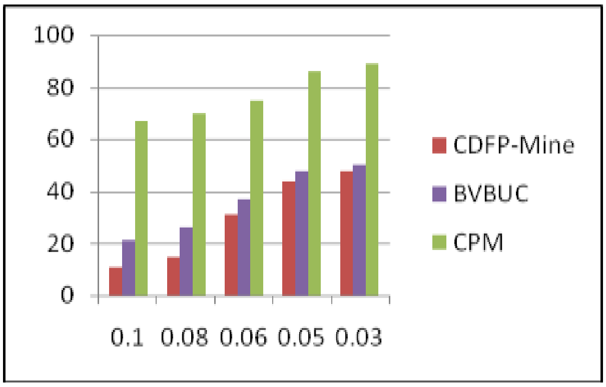


Figure-5. Performance on PC (sec)

Tables-9 and 10 demonstrate the accuracy of Colossal pattern sequences discovered using CDFP-Mine on different datasets is displayed.

Table-9. Colossal pattern sequences discovered using CDFP-Mine (with uniform RF)

Name of Dataset	#Pattern sequences	Maximum Accuracy (MA)	Highest Frequency Measure (FM)	Accuracy Measure (ACC)	Average Frequency Measure (FM)	Accuracy Measure (ACC)
Diabetes	923	97.79	68.54	74.09	66.87	73.83
Breast-Cancer	6936	100.00	96.08	96.42	94.55	95.12
Heart	41096	100.00	80.37	80.85	66.05	70.27

Table-10 Colossal pattern sequences discovered using CDFP-Mine (with varying RF)

Name of Dataset	#Pattern sequences	Maximum Accuracy (MA)	Highest Frequency Measure (FM)	Accuracy Measure (ACC)	Average Frequency Measure (FM)	Accuracy Measure (ACC)
Diabetes	1133	97.79	68.26	73.70	67.20	73.70
Breast-Cancer	11338	100.00	95.74	96.13	94.22	94.84
Heart	62833	100.00	79.40	79.87	64.74	69.94

VIII.CONCLUSION

CDFP-Mine a new algorithm is presented to mine high dimensional datasets. In this algorithm, we iteratively built a data matrix H-Struct, a bitwise portrayal of the dataset for a powerful revelation of DFPs. To extract CPS, we utilized a vector column intersection bitwise operation to encourage the algorithm. To improve the efficiency of the mining process and memory constraints, we also utilized Triple Pair Count Array alongside H-

Struct. The exact investigation demonstrates that our algorithm has accomplished great mining efficiencies under various settings. Besides, our performance analysis shows that this algorithm additionally accomplishes the most elevated Frequency and Accuracy measures in finding CPSs and significantly the best contrasted with earlier created algorithms.

REFERENCES

[1] R. Agrawal and R. Srikant, Fast Algorithms for Mining Association Rules, In *VLDB'94*, pp. 487–499, (1994).

[2] H. Mannila, H. Toivonen and A. I. Verkamo, Efficient Algorithms for Discovering Association rules, In *KDD'94*, pp. 181–192, (1994).

[3] H. Manila, H. Toivonen and A. I. Verkamo, Discovery of Frequent Episodes in Event Sequences, *Data Mining and Knowledge Discovery*, pp. 259–289, (1997).

[4] S. Brin, R. Motwani and C. Silverstein, Beyond Market Basket: Generalizing Association Rules to Correlations, In *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, pp. 265–276, (1997).

[5] R. Srikant and R. Agrawal, Mining Sequential Patterns: Generalizations and Performance Improvements, In *EDBT'96*, pp. 3–17, (1996).

[6] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal and M.-C. Hsu, PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth, In *ICDE'01*, pp. 215–224, (2001).

[7] R. J. Bayardo, Efficiently Mining Long Patterns from Databases, In *SIGMOD'98*, pp. 85–93, (1998).

[8] J. Pei, J. Han and R. Mao, CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets, In *Proc. 2000 ACM-SIGMOD Int.Workshop Data Mining and Knowledge Discovery (DMKD'00)*, pp. 11–20, (2000).

[9] M. Zaki, Generating Non-Redundant Association Rules, In *KDD'00*, pp. 34–43, (2000).

[10] Y. Cheng and G. M. Church, Biclustering of Expression Data, In *Proc of the 8th Intl. Conf. Intelligent Systems for Mocular Biology*, (2000).

[11] J. Yang, H. Wang, W. Wang and P. S. Yu, Enhanced Biclustering on Gene Expression Data, In *Proc. of the 3rd IEEE Symposium on Bioinformatics and Bioengineering (BIBE)*, Washington DC, March (2003).

[12] G. Cong, A. K. H. Tung, X. Xu, F. Pan and J. Yang, FARMER: Finding Interesting Rule Groups in Microarray Datasets, In *Proc. 23rd ACM Int. Conf. Management of Data*, (2004).

[13] C. Creighton and S. Hanash, Mining Gene Expression Databases for Association Rules, *Bioinformatics*, vol. 19, (2003).

[14] N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal, Discovering Frequent Closed Itemsets for Association Rules, In *Proc. 7th Int'l Conf. Database Theory (ICDT)*, (1999).

[15] M. J. Zaki and C. Hsiao, CHARM: An Efficient Algorithm for Closed Association Rule Mining, In *Proc. SIAM Int'l Conf. on Data Mining (SDM)*, (2002).

[16] F. Pan, G. Cong, A. K. H. Tung, J. Yang and M. J. Zaki, CARPENTER: Finding Closed Patterns in Long Biological Datasets, In *Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD),* (2003).

[17] D. Madhavi and M. Shashi, An Efficient Approach to Colossal Pattern Mining., In *Proceedings of int. J. Computer Sci. Network Security (IJCSNS)*, vol. 6, pp. 304–312, (2010).

[18] Mohammed Karim Sohrabi, Ahmad A Barforoush Efficient Colossal Pattern Mining in High Dimensional Datasets, In *Proceedintgs of Journal of Knowledge Based Systems*, vol. 33, pp. 41–52, (2012).

[19] UCI Machine Learning Data Sets http://archive.ics.uci.edu/ml/datasets/.