Information Sciences 496 (2019) 343–362

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

An efficient parallel row enumerated algorithm for mining frequent colossal closed itemsets from high dimensional datasets☆

Manjunath K Vanahalli*, Nagamma Patil

Department of Information Technology, National Institute of Technology Karnataka, Surathkal, Mangalore 575025, India

ARTICLE INFO

Article history:
Received 12 February 2018
Revised 25 July 2018
Accepted 2 August 2018
Available online 3 August 2018

Keywords:
Bioinformatics
High dimensional datasets
Parallel preprocessing
Parallel algorithm
Colossal closed itemsets
Rowset cardinality table

ABSTRACT

Mining colossal itemsets from high dimensional datasets have gained focus in recent times. The conventional algorithms expend most of the time in mining small and mid-sized itemsets, which do not enclose valuable and complete information for decision making. Mining Frequent Colossal Closed Itemsets (FCCI) from a high dimensional dataset play a highly significant role in decision making for many applications, especially in the field of bioinformatics. To mine FCCI from a high dimensional dataset, the existing preprocessing techniques fail to prune the complete set of irrelevant features and irrelevant rows. Besides, the state-of-the-art algorithms for the same are sequential and computationally expensive. The proposed work highlights an Effective Improved Parallel Preprocessing (EIPP) technique to prune the complete set of irrelevant features and irrelevant rows from high dimensional dataset and a novel efficient Parallel Frequent Colossal Closed Itemset Mining (PFCCIM) algorithm. Further, the PFCCIM algorithm is integrated with a novel Rowset Cardinality Table (RCT), an efficient method to check the closeness of a rowset and also an efficient pruning strategy to cut down the mining search space. The proposed PFCCIM algorithm is the first parallel algorithm to mine FCCI from a high dimensional dataset. The performance study shows the improved effectiveness of the proposed EIPP technique over the existing preprocessing techniques and the improved efficiency of the proposed PFCCIM algorithm over the existing algorithms.

1. Introduction

The basic and major step of Association Rule Mining (ARM) is Frequent Itemset Mining (FIM). FIM has a great and vast range of applications, including customer analysis, software bug detection, web analysis, event detection, spatiotemporal analysis, text analysis, toxicological analysis, Ribo Nucleic Acid (RNA) analysis and chemical compound predication [1–12]. FIM helps in mining crucial motifs in a variety of biological and chemical applications [1,6]. It is useful in designing methods for clustering high dimensional data [1,7]. The associative classification, which is the integration of ARM and classification helps in the prediction of the subcellular location of proteins and in achieving high accuracy [6,13]. ARM plays a vital role

☆ The research work is about first parallel row enumerated algorithm for mining large cardinality itemsets called as Colossal Itemsets from high dimensional datasets.

* Corresponding author.
E-mail addresses: manjunath.k.vanahalli@gmail.com (M.K. Vanahalli), nagammapatil@nitk.ac.in (N. Patil).

https://doi.org/10.1016/j.ins.2018.08.009
0020-0255/

in providing visual representations of the underlying text collection [1,5,11]. The ARM has been extensively used in gene expression data analysis to uncover the gene networks [6].

An extensive research in the field of itemset mining led to the development of efficient sequential FIM algorithms [14–19]. The inefficiency of sequential FIM algorithms in handling the large transactional datasets led to the development of parallel FIM algorithms [20–24]. The generation of redundant association rules from a large number of mined frequent itemsets resulted in the proposal of Frequent Closed Itemsets (FCI). Mining FCI has received great interest over the past two decades. Conventional parallel algorithms [25–28] and sequential algorithms [29–33] focus on mining FCI from transactional datasets consisting of a large number of rows (sample) and less number of attributes (features). These conventional parallel and sequential algorithms are feature enumeration based algorithms as they tend to mine FCI by searching the itemset space. There will be an exponential increase in the running time of these conventional parallel and sequential algorithms as the average transaction length increases. In the modern era, the abundant data across a variety of domains, including bioinformatics has led to the new form of dataset known as a high dimensional dataset, whose data characteristics are different from that of transactional datasets. The high dimensional datasets consist of less number of rows and considerably large number of features. The amount of information that can be extracted from high dimensional datasets is potentially huge, but extraction of information and knowledge from these datasets is a non-trivial task.

The conventional parallel algorithms [25–28] and sequential algorithms [29–33] face an uphill task in mining FCI from the high dimensional datasets. To overcome the inefficiency and the uphill task of these algorithms, sequential row enumerated algorithms were proposed to mine FCI from high dimensional datasets [34–38]. This problem of inefficiency can be solved to a greater extent by parallel row enumerated algorithms. However, to the best of our knowledge, there is no parallel row enumerated algorithm proposed till date to mine FCI from high dimensional datasets. The result of FCI mining algorithms includes small and mid-sized itemsets, which do not enclose valuable and complete information in many applications [6,13,39–49]. In applications dealing with high dimensional datasets such as bioinformatics, ARM gives greater importance to the large-sized itemsets known as colossal itemsets [6,39,43–49]. The colossal itemsets are more influential in decision making and are significant for many applications [13,40–42]. The importance of discovering the colossal itemsets from high dimensional datasets such as gene expression data was shown by Ronnie Alves et al. [39] and Stefan Naulaerts et al. [6]. The existing sequential, parallel FIM algorithms [14–24] and sequential, parallel FCI mining algorithms [25–38] are inefficient in mining Frequent Colossal Closed Itemsets (FCCI) from high dimensional dataset, as they expend an exponential time in mining a large number of small and mid-sized itemsets.

Zhu et al. [48] were the first to introduce the concept of colossal itemsets in an algorithm based on Pattern Fusion. The existing frequent colossal itemset mining algorithms [43–48] and FCCI mining algorithms [47–49] are all sequential and do not enclose an effective preprocessing technique. The preprocessing techniques of these existing algorithms [43–49] are sequential and fail to prune the complete set of irrelevant features and irrelevant rows, which leads to an exponential increase in the mining search space. Algorithms [47,48] mine limited set of FCCI leading to the generation of an incomplete set of association rules, which consequently affects decision making. Many of the mined FCCI from the BVBUC algorithm [47] tend to provide incorrect support information leading to the generation of an incorrect set of association rules, which ends up affecting decision making. The closeness checking method of rowset and pruning strategy to cut down the mining search space of the existing algorithms [43–49] are inefficient. This highlights the inefficiency of these existing sequential algorithms in mining FCCI from high dimensional datasets. In this paper the problem of inefficiency was solved by proposing the first parallel row enumerated algorithm for mining FCCI from high dimensional datasets.

To surmount the drawbacks, a novel efficient Parallel Frequent Colossal Closed Itemset Mining (PFCCIM) algorithm integrating the following proposed techniques was designed.

1. An Effective Improved Parallel Preprocessing (EIPP) technique was proposed, which prunes the complete set of irrelevant features and irrelevant rows by an effective utilization of minimum support threshold and minimum cardinality threshold respectively.
2. An efficient Rowset Cardinality Table (RCT) based closeness checking method was proposed to check the closeness of a rowset. The proposed method checks the closeness of newly mined frequent colossal itemsets irrespective of the previously mined FCCI. It is not required to store the complete set of previously mined FCCI in the main memory as the closeness checking of a rowset indicates the closeness of an itemset mined at that particular rowset. The advantage of not having a dependency to check the closeness of a rowset by the proposed RCT based closeness checking method will help in great extent for designing the proposed parallel row enumerated algorithm for mining FCCI from high dimensional datasets.
3. An efficient RCT based pruning strategy was proposed to cut down the mining search space by efficient utilization of minimum cardinality threshold. The RCT at the row enumerated node provides prior information regarding the cardinality of the itemsets to be mined at descendant nodes without traversing them, unlike existing FCCI mining algorithms, which do not provide any prior information regarding the same.

The proposed PFCCIM algorithm mine FCCI from the high dimensional dataset by utilizing the parallel bottom-up row enumeration approach as the large cardinality itemsets are present at the initial levels of the bottom-up row enumerated tree. The proposed algorithm utilizes the bitset approach as it is computationally fast. The proposed PFCCIM algorithm is the first parallel bottom-up row enumerated algorithm to mine FCCI from a high dimensional dataset. Results show that the proposed EIPP technique prunes the complete set of irrelevant features and irrelevant rows from the high dimensional

dataset, unlike the existing preprocessing techniques. The proposed PFCCIM algorithm outperforms the existing algorithms in terms of runtime with the help of the proposed efficient RCT based closeness checking method and the proposed efficient RCT based pruning strategy.

The remaining paper is organized as follows. Section 2 highlights the related work. Section 3 describes the preliminaries with respect to itemsets. The proposed parallel preprocessing technique is emphasized in Section 4. Parallel bottom-up traversal of a row enumerated tree is highlighted in Section 5. Section 6 highlights the proposed parallel row enumerated algorithm for mining FCCI. Results are discussed in Section 7, followed by conclusion and future work in Section 8.

2. Related work

An extensive literature survey was carried in the field of itemset mining. FIM and ARM are used to address the unique problems across a wide variety of data domains [1–12]. The ARM is used to review the gene interaction networks to capture the relationship between a group of genes [6]. ARM helps in discovering the enormous amount of knowledge contained by gene ontology annotations [42]. FIM is used to discover the combination of structural features and also to help in identifying the strong associations between allelic combinations associated with diseases [1,6]. The disadvantage of phrase-based approaches in text mining is solved using FIM [12]. The performance of information filtering during the analysis of text is significantly improved by pattern taxonomy. The larger patterns in the taxonomy are more specific and useful during text analysis.

Many efficient sequential FIM algorithms [14–19] were developed over the years to mine the frequent itemsets from transactional datasets. The exponential increase in the running time due to the average increase in the transactional length and the inefficiency of sequential FIM algorithms in handling the large transactional datasets led to the development of parallel FIM algorithms [20–24]. But the generation of redundant rules from a large number of mined frequent itemsets led to the proposal of FCI. Sequential algorithms [29–33] and parallel algorithms [25–28] were developed to mine FCI from the transactional dataset. These conventional parallel and sequential algorithms are feature enumeration based algorithms, which are computationally expensive in mining FCI from high dimensional datasets due to the data characteristics of the high dimensional datasets. Row enumerated FCI mining algorithms [34–38] were developed to handle the computational complexity. These algorithms adapt the sequential row enumeration technique to mine FCI from the high dimensional dataset by searching the rowset space. The row enumerated algorithms [34–38] make use of either the bottom-up or top-down search strategy to navigate through the mining search space.

Row enumerated FCI mining algorithms [34–38] focus on the transposed table approach and dummyTXdummy- generate a large number of conditional transposed tables. F Pan et al. [37] proposed an algorithm with the combination of row and feature enumeration methods to mine FCI from a dataset consisting of a large number of rows and a large number of features. All the existing row enumerated FCI mining algorithms are sequential, unfortunately, no parallel row enumerated FCI mining algorithm has been proposed till date. The parallel row enumerated algorithms are best suited for high computationally expensive problem and will be more efficient in mining FCI from high dimensional datasets. The existing FCI algorithms mine a huge number of small and mid-sized closed itemsets from high dimensional datasets, which have limited information for many applications. Large cardinality itemsets known as colossal itemsets, have greater importance in decision making and provide more suitable information for many applications [6,39,43–45,47–49]. Alves et al. [39] and Naulaerts et al. [6] showed the importance of discovering the colossal itemsets from high dimensional datasets such as gene expression data.

The existing frequent colossal itemset mining algorithms [43–48] and FCCI mining algorithms [47–49] are classified into feature enumeration and row enumeration based algorithms. Feature enumeration based frequent colossal itemset mining algorithms [46,48] and FCCI mining algorithm [48] are best suited for transactional datasets due to their data characteristics. The existing feature enumeration based frequent colossal itemset mining algorithms and FCCI mining algorithms are sequential. The preprocessing techniques of these algorithms [46,48] are sequential and fails to prune the complete set of irrelevant rows and irrelevant features.

CP-Miner [45] and PCP-Miner [45] are sequential row enumerated based algorithms to mine frequent colossal itemsets from high dimensional datasets. Generation of redundant rules from frequent colossal itemsets resulted in the proposal of FCCI. The sequential row enumeration approach of CP-Miner and PCP-Miner algorithm highlights the inefficiency of these algorithms. These algorithms also lack the capability to mine FCCI from high dimensional datasets. DisClose [49] and BVBUC [47] are sequential row enumerated based algorithms to mine FCCI from high dimensional datasets. BVBUC algorithm [47] mine FCCI from the rowsets which are represented by the minimum support threshold level in row enumerated tree and prune the remaining mining search space. BVBUC utilizes the bitset approach for faster computation, but fails to mine the complete set of FCCI leading to the generation of an incomplete set of association rules, which affects decision making. Many of the mined FCCI from the BVBUC algorithm tend to provide incorrect support information leading to the generation of an incorrect set of association rules. The Disclose algorithm [49] proposed by Zulkurnain et al. uses sequential row enumerated Compact-Row tree (CR-tree) data structure to mine FCCI. The existing FCCI mining algorithms are sequential and adopt the sequential row enumeration based approach. The preprocessing techniques of the existing FCCI mining algorithms are ineffective as they fail to prune the complete set of irrelevant rows and irrelevant features, which exponentially increases the mining search space. Moreover, existing FCCI mining algorithms fail to enclose efficient closeness checking

Table 1
High dimensional dataset *D*.

Row id (*rid*)	Features
1	$f_1, f_2, f_4, f_6, f_{10}$
2	f_1, f_2, f_4, f_7, f_8
3	f_2, f_4, f_7, f_8
4	$f_1, f_2, f_6, f_8, f_9, f_{10}$
5	$f_1, f_3, f_4, f_7, f_8, f_{10}$
6	f_2, f_4, f_9
7	f_5, f_7
8	f_5, f_{11}

method of a rowset and efficient pruning strategy, which highlight the inefficiency of these algorithms in mining FCCI from high dimensional datasets.

3. Preliminaries

Let the high dimensional dataset D (R,F) consist of m number of rows, R = $\{r_1, r_2, \ldots, r_m\}$ and n number of features F = $\{f_1, f_2, \ldots, f_n\}$. Each r_i consists of a set of features and has a unique row identifier *rid*. A non-empty subset of features, $X \subseteq$ F is defined as an itemset. An itemset consisting of l features is defined as l-itemset. Let r(f_j) signify the rows in which the j^{th} feature of the dataset is present. A non-empty subset of *rid*s, Y $\subseteq$ R is defined as a rowset. A rowset consisting of l *rid*s is defined as l-rowset. Let f(r_i) signify the features occurring in the i^{th} row of the dataset.

Example 1. Table 1 shows an example of a high dimensional dataset D consisting of 8 rows, where each row is described with unique row identifier (*rid*), R= {1, 2, 3, 4, 5, 6, 7, 8} and 11 features, F= $\{f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}\}$.

Definition 1 (Support)**.** The number of rows in which an itemset X occurs is called the support of an itemset, denoted by *sup(X)*.

Example 2. In Table 1, the support of an itemset X= $\{f_2, f_4, f_7, f_8\}$, *sup(X)* is 2.

Definition 2 (Support Set)**.** The rows in which an itemset X occurs is called support set of an itemset, denoted by *supset(X)*.

Example 3. In Table 1, the support set of an itemset X= $\{f_2, f_4, f_7, f_8\}$, *supset(X)* is *23*.

Definition 3 (Cardinality)**.** The number of items in an itemset X is known as the cardinality of an itemset, denoted by *card(X)*.

Example 4. In Table 1, the cardinality of an itemset X= $\{f_2, f_4, f_7, f_8\}$, *card(X)* is 4.

Definition 4 (Frequent Itemset)**.** An itemset X is called frequent itemset if and only if $sup(X) \geq minsup$, where *minsup* is user specified least support threshold.

Example 5. In Table 1, the itemset X= $\{f_2, f_8\}$ is frequent itemset with minimum support threshold set to 2, $sup(X) \geq 2$.

Definition 5 (Frequent Closed Itemset)**.** An itemset X is called frequent closed itemset if and only if it is frequent and there exists no proper superset X'', ($X \subset X''$) such that support of X is same as the support of X'', $sup(X)$=$sup(X'')$.

Example 6. In Table 1, the itemset X= $\{f_4, f_7, f_8\}$ is frequent closed itemset with minimum support threshold set to 2 because $f_4f_7f_8$ is frequent and there exists no proper superset X'' with the same support as X.

Definition 6 (Frequent Colossal Itemset)**.** An itemset X is called frequent colossal itemset if and only if it is frequent and $card(X) \geq mincard$, where *mincard* is user specified least cardinality threshold.

Example 7. In Table 1, the itemset X= $\{f_1, f_2, f_6, f_{10}\}$ is frequent colossal itemset with minimum support threshold set to 2 and minimum cardinality threshold set to 4, $sup(X) \geq 2$ and $card(X) \geq 4$.

Definition 7 (Frequent Colossal Closed Itemset)**.** An itemset X is called frequent colossal closed itemset if and only if it is frequent closed and $card(X) \geq mincard$, where *mincard* is user specified least cardinality threshold.

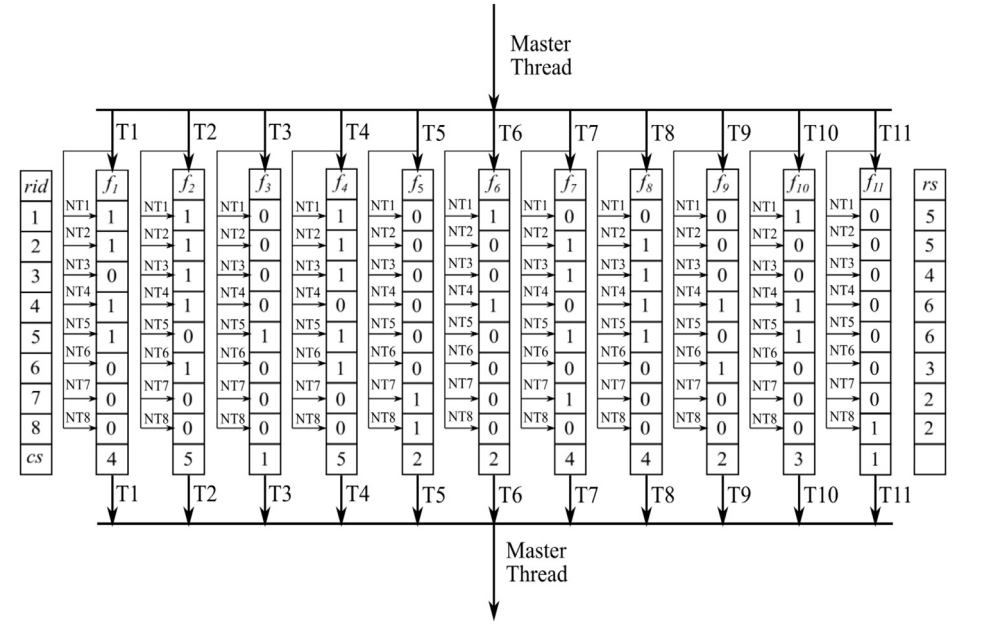
Example 8. In Table 1, the itemset X= $\{f_2, f_4, f_7, f_8\}$, is frequent colossal closed itemset with minimum support threshold set to 2 and minimum cardinality threshold set to 4, $sup(X) \geq 2$ and $card(X) \geq 4$.

Definition 8 (Closure)**.** Given an itemset $X \subseteq$ F and a rowset $Y \subseteq$ R in a high dimensional dataset D (R,F), we define

$$r(X) = \{r_i \in R \mid \forall f_j \in X,\ f_j \textit{ is present in } r_i \textit{ of } D\} \tag{1}$$

$$f(Y) = \{f_j \in F \mid \forall r_i \in Y,\ f_j \textit{ is present in } r_i \textit{ of } D\} \tag{2}$$

Table 2
bitTable Corresponding to High Dimensional Dataset *D*. Parallel Pruning of Irrelevant Features.



The closure of an itemset X, C(X) and closure of a rowset Y, C(Y) is defined as follows

$$C(X) = f(r(X)) \tag{3}$$

$$C(Y) = r(f(Y)) \tag{4}$$

4. Proposed parallel preprocessing technique

The preprocessing technique of existing frequent colossal itemset mining algorithms and FCCI mining algorithms are sequential and fail to prune the complete set of irrelevant rows and irrelevant features, which leads to an exponential increase in the search space. The proposed effective improved preprocessing technique parallelizes the pruning of irrelevant rows and irrelevant features. The proposed work is the first parallel effective preprocessing technique which prunes the complete set of irrelevant rows and irrelevant features. The proposed effective improved parallel preprocessing technique incorporates the bitset approach, which is computationally fast. The bitTable as shown in Table 2 is constructed with the same dataset characteristics as that of a high dimensional dataset *D*, as shown in Table 1. If a row of high dimensional dataset *D* consists of feature f_j then the j^{th} bit of the corresponding row in bitTable is set to 1, else set to 0. The *rs* in Table 2 indicates the number of features present in the respective row. For example, the number of features present in 5th row is 6. The *cs* in Table 2 indicates the support of respective feature in the bitTable. For example, the support of feature f_4 in bitTable is 5. The features of the high dimensional dataset, which do not satisfy the criteria of *minsup* are described as irrelevant features. The rows of the high dimensional dataset which do not satisfy the criteria of *mincard* are described as irrelevant rows. The preprocessing technique of the existing FCI algorithms prunes the irrelevant features before proceeding with mining of FCI. Let F′ be the set of irrelevant features in the dataset. The features $\{f_3, f_{11}\}$ are the irrelevant features with *minsup* set to 2, $F' = \{f_3, f_{11}\}$.

$$F_1 = (F - F') = \{f_1, f_2, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}\} \tag{5}$$

The Eq. (5) highlights the pruning of irrelevant features $\{f_3, f_{11}\}$ from the bitTable, as shown in Table 2. Table 3a shows the bitTable after pruning the irrelevant features $\{f_3, f_{11}\}$. Table 2 shows parallelising the process of pruning irrelevant features. Multiple threads (T1, T2, T3, T4, T5, T6, T7, T8, T9, T10, T11) are forked from the master thread to process the large number of features simultaneously. These child threads are forked again as Nested Threads (NT1, NT2, NT3, NT4, NT5, NT6, NT7, NT8) to increase the computation of calculating the support of the respective features. An effective improved parallel preprocessing technique has two level parallelism to prune the irrelevant features.

The irrelevant features and irrelevant rows have to be pruned before proceeding with FCCI mining. Hence, the preprocessing technique of existing colossal itemset mining algorithms [43–45] and FCCI mining algorithm [49] prune the irrelevant

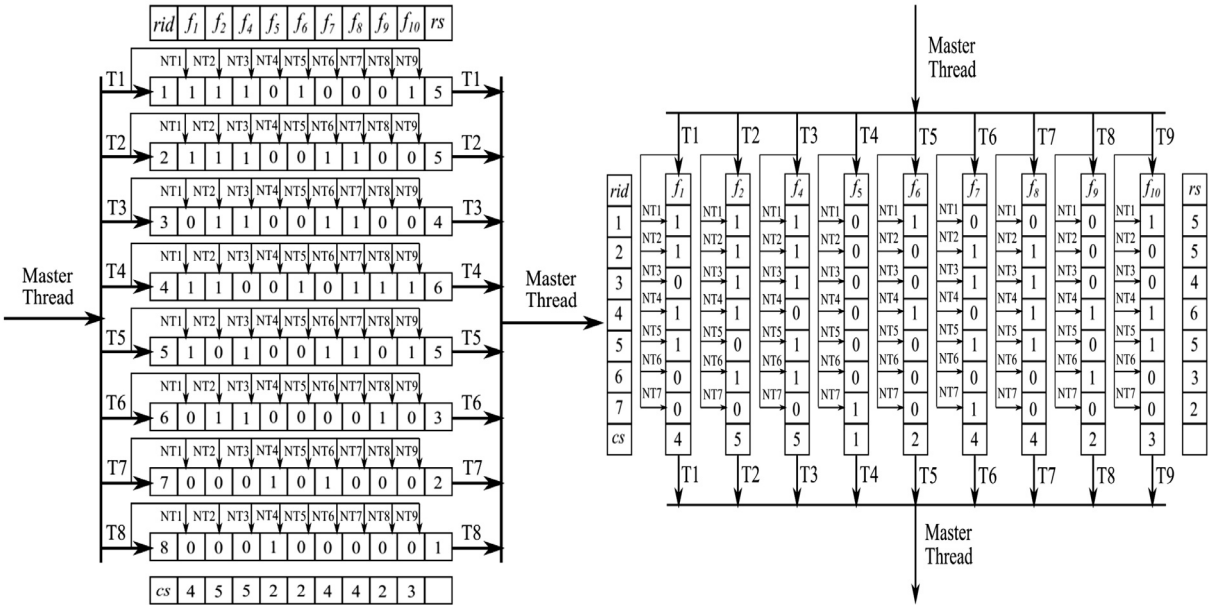
Table 3
Proposed Parallel Preprocessing Technique.

(a) bitTable after pruning irrelevant features $\{f_3, f_{11}\}$.

Parallel Pruning of Irrelevant Rows.

(b) bitTable after pruning irrelevant row *8*.

Parallel Pruning of Irrelevant Features.

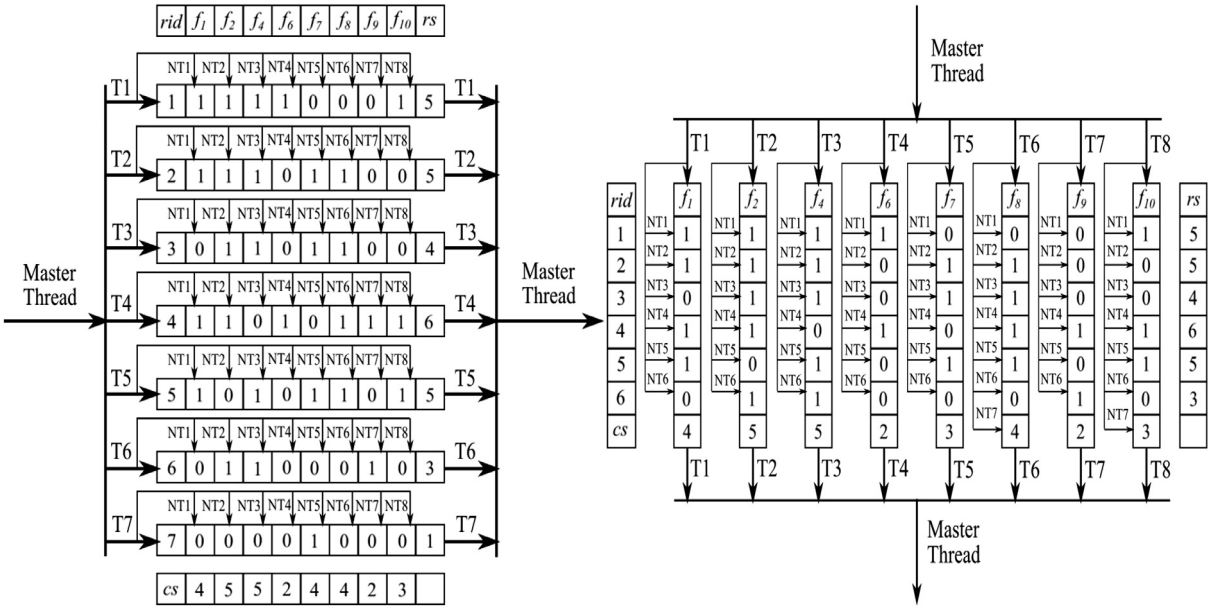


(c) bitTable after pruning irrelevant feature $\{f_5\}$.

Parallel Pruning of Irrelevant Rows.

(d) bitTable after pruning irrelevant row *7*.

Parallel Pruning of Irrelevant Features.



rows after pruning the irrelevant features. Let R′ be the set of irrelevant rows from the dataset. The 8th row is irrelevant with *mincard* set to 2, R′ = {8}.

$$R_1 = (R - R') = \{1, 2, 3, 4, 5, 6, 7\} \tag{6}$$

The Eq. (6) highlights the pruning of irrelevant row 8 from the bitTable, as shown in Table 3a. Table 3b shows the bitTable after pruning the irrelevant row 8. Table 3a shows parallelising the process of pruning irrelevant rows. Multiple

threads (T1, T2, T3, T4, T5, T6, T7, T8) are forked from the master thread to process the rows of high dimensional dataset simultaneously. These child threads are forked again as Nested Threads (NT1, NT2, NT3, NT4, NT5, NT6, NT7, NT8, NT9) to increase the computation of calculating the cardinality of the respective rows. An effective improved parallel preprocessing technique has two level parallelism to prune the irrelevant rows. The support of each feature ($sup(f_j)$, $\forall\ f_j \in F_1$) is affected by the pruning of irrelevant rows. The pruning of irrelevant row 8 will help in reducing the support of the feature $\{f_5\}$ and eventually converting it to irrelevant feature, F′={f_5}. The preprocessing technique of existing colossal itemset mining algorithms [43–45] and FCCI mining algorithm [49] prune the irrelevant features and irrelevant rows just once, and after that fails to take advantage of the reduction in the support of features due to the pruning of irrelevant rows. Hence, the preprocessing technique of the existing colossal itemset mining algorithms and FCCI mining algorithms fail to prune the complete set of irrelevant features and irrelevant rows. To overcome this drawback of existing preprocessing techniques, the proposed effective improved parallel preprocessing technique takes advantage of the reduction in the support of features due to the pruning of irrelevant rows and continues to prune the irrelevant features. The process of pruning the irrelevant features in the proposed effective improved parallel preprocessing technique has been parallelized as shown in Tables 2, 3b and 3d.

$$F_2 = (F_1 - F') = \{f_1, f_2, f_4, f_6, f_7, f_8, f_9, f_{10}\} \tag{7}$$

The Eq. (7) highlights the pruning of irrelevant feature $\{f_5\}$ from the bitTable, as shown in Table 3b. Table 3c shows the bitTable after pruning the irrelevant feature $\{f_5\}$. The cardinality of each row ($card(r_i)$, $\forall\ r_i \in R_1$) is affected by the pruning of the irrelevant features. The pruning of irrelevant feature $\{f_5\}$ will help in reducing the cardinality of the 7th row and eventually converting it to irrelevant row, R′={7}. The proposed effective improved parallel preprocessing technique takes the advantage of reduction in the cardinality of the rows due to the pruning of irrelevant features and continues to prune the irrelevant rows. The proposed effective improved parallel preprocessing technique prunes the irrelevant features and irrelevant rows alternatively in an iterative manner until all the remaining features and rows in the bitTable satisfy the criteria of *minsup* and *mincard* respectively, whereas, the existing preprocessing technique prunes the irrelevant features and irrelevant rows just once.

$$R_2 = (R_1 - R') = \{1, 2, 3, 4, 5, 6\} \tag{8}$$

The Eq. (8) highlights the pruning of irrelevant row 7 from the bitTable, as shown in Table 3c. Table 3d shows the bitTable after pruning the irrelevant row 7. All the remaining features and rows in the bitTable, as shown in Table 3d satisfy the criteria of *minsup* and *mincard*, hence the reduction terminates. Table 3d shows the bitTable after applying the proposed effective improved parallel preprocessing technique when the *minsup* and *mincard* is set to 2. Table 3b shows the bitTable after applying the preprocessing technique of the existing algorithms when the *minsup* and *mincard* is set to 2. It is observed that the proposed effective improved parallel preprocessing technique prunes the complete set of irrelevant features and irrelevant rows, which is the major limitation of the preprocessing technique in the existing algorithms. After the proposed preprocessing, let F_{final} be the set of relevant features and R_{final} be the set of relevant rows.

The proposed effective improved parallel preprocessing technique is divided into two tasks. First, Parallel Minimum Support Threshold Preprocessing (PMSTP) task to prune the irrelevant features parallely. Second, Parallel Minimum Cardinality Threshold Preprocessing (PMCTP) task to prune the irrelevant rows parallely. The proposed effective improved parallel preprocessing technique is highlighted by Algorithm 1. PMSTP task, as shown in Procedure 1 and PMCTP task, as shown in

Algorithm 1 Effective improved parallel preprocessing technique.

Input: bitTable, *minsup*, *mincard*.
Output: preprocessed bitTable
Initialisation : P_flag = 1
1: **while** (P_flag==1) **do**
2: PMSTP_task()
3: P_flag=0
4: PMCTP_task()
5: **end while**

Procedure 2 are invoked by the proposed effective improved parallel preprocessing technique in an iterative manner until all the remaining features and rows in the bitTable satisfy the criteria of *minsup* and *mincard*, respectively.

Procedure 1 PMSTP_task().

```
#pragma omp parallel for
for (j=0;j<no_of_features;);j++ do
    support_j=0
    #pragma omp parallel for reduction (+:support_j)
    for (i=0;i<no_of_rows;i++) do
        if (bitTable[i][j]==1) then
            support_j = support_j + 1
        end if
    end for
    if (support_j<minsup) then
        delete j^th feature
    end if
end for
```

Procedure 2 PMCTP_task().

```
#pragma omp parallel for
for (i=0;i<no_of_samples;) do
    cardinality_i=0
    #pragma omp parallel for reduction (+:cardinality_j)
    for (j=0;j<no_of_features;j++) do
        if (bitTable[i][j]==1) then
            cardinality_i = cardinality_i + 1
        end if
    end for
    if (cardinality_i<mincard) then
        delete i^th sample
        P_flag=1
    end if
end for
```

5. Parallel bottom-up traversal of row enumerated tree

The proposed work mine FCCI from high dimensional datasets by adopting a row enumeration technique due to the data characteristics of the high dimensional datasets. The bottom-up traversing of the row enumerated tree implies that the search starts from the smaller rowset value. The sequential bottom-up traversal expend huge amount of time in the complete traversal of the row enumerated tree. To overcome the computation problem, the proposed work adopts the parallel bottom-up traversal of row enumerated tree. Fig. 1 shows the parallel bottom-up traversal of the row enumerated tree. Fig. 1 shows that the team of threads are forked from the master thread for the parallel bottom-up traversal of the row enumerated tree. The row enumerated tree of Fig. 1 is constructed for the preprocessed bitTable shown in Table 3d.

The bottom-up search strategy is efficient in mining colossal itemsets as the large cardinality itemsets are present at the initial level of the bottom-up row enumerated tree. The bottom-up approach also has a benefit of utilizing the bitset result of the parent node to obtain the bitset result at the child nodes, thus, exponentially reducing the number of bitwise AND operations to be performed. Only one bitwise AND operation is required to obtain the bitset result at each node. For example, in Fig. 1, the bitset result at node 1236 requires 1 bitwise AND operation. The bitset result of node 123 and node 6 are utilized to generate the bitset result of node 1236. The anti-monotone property of the minimum cardinality threshold is utilized by the bottom-up search strategy to cut down the search space. It means that if an itemset at a node represented by *l*-rowset is not colossal, then an itemset at a child node represented by *(l+1)*-rowset is also not colossal. For example, with *minsup* set to 2 and *mincard* set to 3, the descendants of node 123 can be pruned because the itemsets in descendant nodes are not colossal.

6. Proposed parallel row enumerated algorithm

6.1. Rowset cardinality table

The proposed closeness checking method takes advantage of the novel Rowset Cardinality Table (RCT) in the row enumeration tree to check the closeness of a rowset. If the rowset is closed then the itemset mined at that rowset is also closed. The RCT helps in closeness checking of a rowset without the need to scan through the previously mined FCCI itemsets. The proposed pruning strategy utilizes the RCT to efficiently cut down the search space. The RCT provides prior information

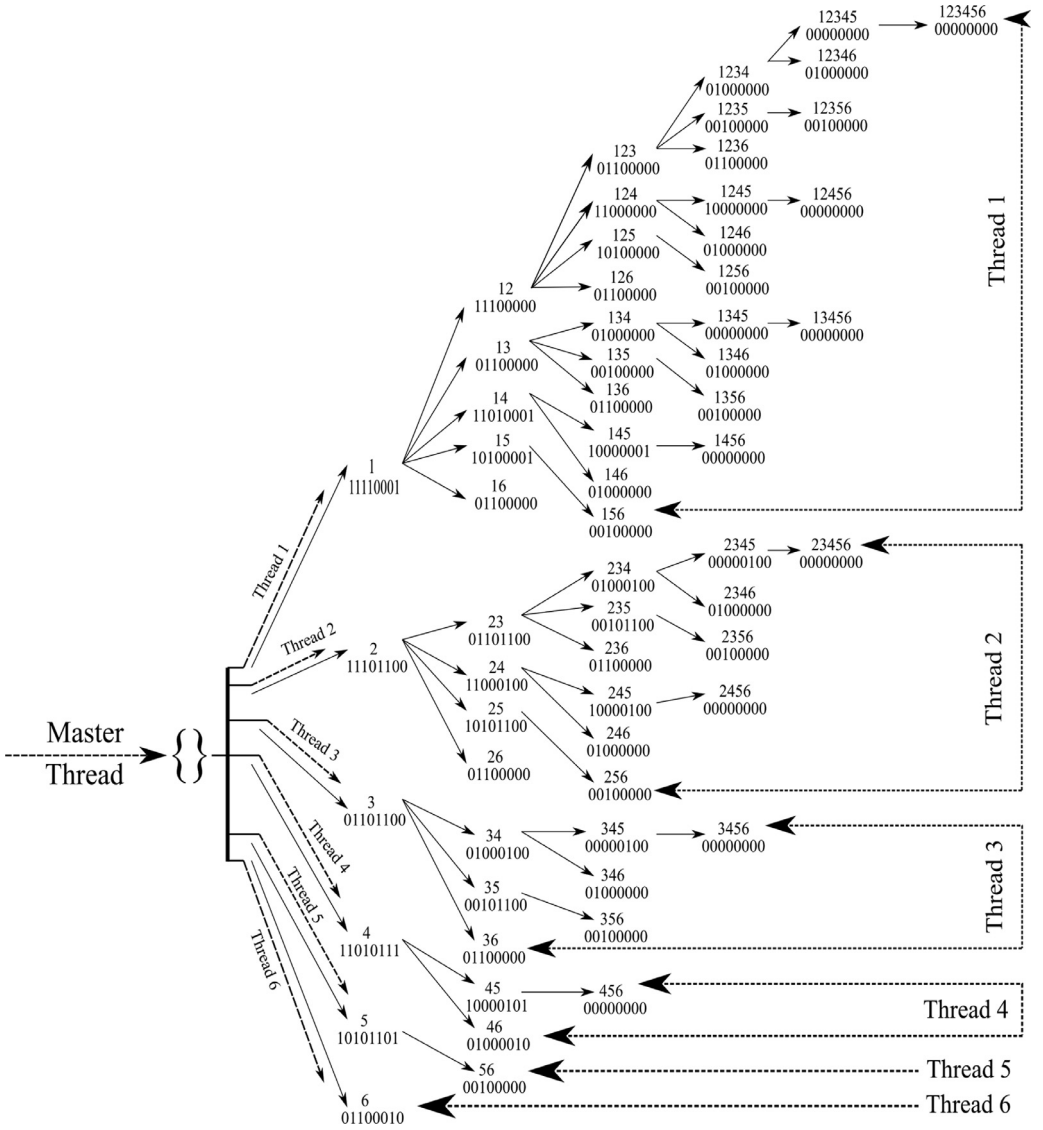


Fig. 1. Parallel bottom-up traversal of row enumerated tree.

regarding the cardinality of the itemsets to be mined at the descendant nodes without traversing them. The RCT for every node is shown in Fig. 2.

Definition 9 (Rowset Cardinality Table)**.** Given a rowset $Y=\{r_{i1}, r_{i2},\ldots\ldots, r_{ik}\}$ representing a row enumerated node in an order such that $r_{i1} < r_{i2} < \ldots. < r_{ik}$, the Rowset Cardinality Table (RCT_Y) at node Y contains the updated cardinality for each row in (R_{final} - Y) depending upon the cardinality of the bitset result obtained at node Y.

The Rowset Cardinality Table (RCT_Y) at node Y is obtained by the following steps.

1. Obtain the indexes of all the ones appearing in the bitset result obtained at node Y.
2. For each row in (R_{final} - Y), calculate the number of ones from the preprocessed bitTable appearing at the indexes obtained from step 1.

Example 9. Obtaining Rowset Cardinality Table for node *14* (RCT_{14}) and *23* (RCT_{23}) in Fig. 2.

- Rowset Cardinality Table for node *14*, RCT_{14} is shown in Table 4c. The indexes of all the ones appearing in the bitset result (11010001) at node *14* are {1,2,4,8}.
- For each row in (R_2 - Y) = {2,3,5,6}, the number of ones appearing at the indexes {1,2,4,8} from the preprocessed bitTable, as shown in Table 3d, are {2,1,2,1}.

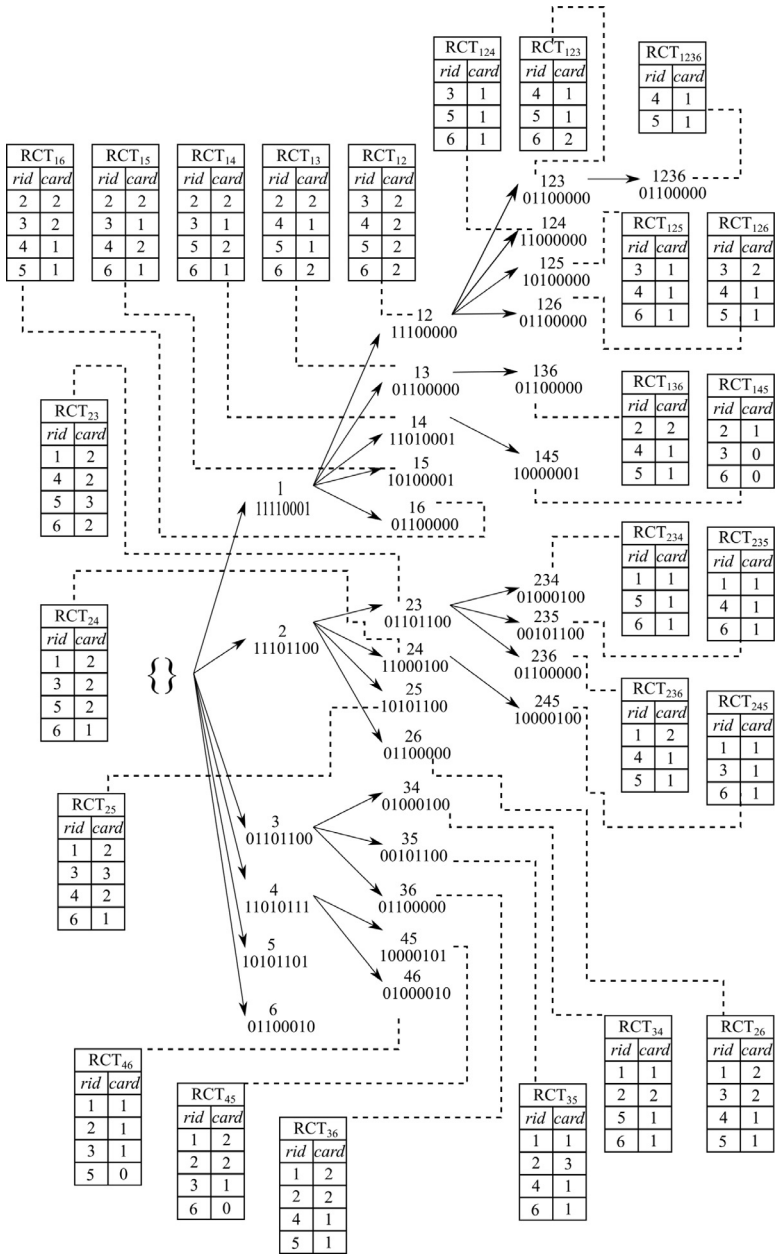


Fig. 2. Bottom-Up Row Enumeration Tree with Rowset Cardinality Table for respective nodes.

- Rowset Cardinality Table for node *23*, RCT_{23} is shown in Table 4d. The indexes of all the ones appearing in the bitset result (01101100) at node *23* are {2,3,5,6}.
- For each row in (R_2 - Y) = {1,4,5,6}, the number of ones appearing at the indexes {2,3,5,6} from the preprocessed bitTable, as shown in Table 3d, are {2,2,3,2}.

Example 10. Tables 4a, 4b 4c and 4d show the Rowset Cardinality Table at node *12, 13, 14* and *23* respectively in Fig. 2.

6.2. Rowset cardinality table based proposed closeness checking method

The RCT based closeness checking method is proposed to speed up the closeness checking of a rowset during the traversal of the row enumerated tree. The closeness of the rowset indicates that the itemset occurring in that rowset is also closed. The proposed efficient closeness checking method will not scan through the previously mined FCCI to check the existence and closeness of newly mined frequent colossal itemset.

Table 4
Rowset Cardinality Table of *12, 13, 14* and *23*.

$(a)RCT_{12}$		$(b)RCT_{13}$		$(c)RCT_{14}$		$(d)RCT_{23}$	
rid	card	*rid*	card	*rid*	card	*rid*	card
3	2	2	2	2	2	1	2
4	2	**4**	1	3	1	**4**	2
5	2	**5**	1	**5**	2	**5**	3
6	2	**6**	2	**6**	1	**6**	2

Table 5
Rowset Cardinality Table of *123*, RCT_{123} .

rid	card
4	1
5	1
6	2

Lemma 1. *A rowset $Y \subseteq R_{final}$ during the row enumeration is closed iff the cardinality of all the rows in the RCT_Y is less than the cardinality of an itemset $X \subseteq F_{final}$ mined at rowset Y.*

Proof. According to Definition 8, if rowset *Y* is closed, then $Y{=}r(X)$. It is necessary to prove that $Y{=}r(X)$ with the help of RCT_Y. The $RCT_Y(rid)$ returns the updated cardinality value corresponding to the *rid* in RCT_Y. (For all)$\forall$ *rid* $\in$ (R_{final} - *Y*), if $RCT_Y(rid)$ is less than the cardinality of an itemset *X* mined at the rowset *Y*, then an itemset *X* has not occurred in (R_{final} - *Y*). This indicates that an itemset *X* has occurred only in *Y*, hence $Y{=}r(X)$ proved. Therefore, rowset *Y* is closed. □

Lemma 2. *A rowset $Y \subseteq R_{final}$ during the row enumeration is not closed iff the cardinality of any one of the rows in the RCT_Y is equal to the cardinality of an itemset $X \subseteq F_{final}$ mined at a rowset Y.*

Proof. According to Definition 8, if rowset *Y* is not closed, then $Y \neq r(X)$. It is necessary to prove that $Y \neq r(X)$ with the help of RCT_Y. The $RCT_Y(rid)$ returns the updated cardinality value corresponding to the *rid* in RCT_Y. According to the steps followed for obtaining RCT_Y at rowset *Y*, the updated cardinality for each *rid* in RCT_Y will never be greater than the cardinality of an itemset *X* mined at rowset *Y*. (For any)$\forall$ *rid* $\in$ (R_{final} - *Y*), if $RCT_Y(rid)$ is equal to the cardinality of an itemset *X* mined at the rowset *Y*, then an itemset *X* has occurred in (R_{final} - *Y*). Hence, $Y \neq r(X)$ proved. Therefore, rowset *Y* is not closed. □

The RCT based closeness checking is on the basis of Lemma 1 and Lemma 2. If the rowset is closed then the itemset occurring in that rowset is also closed. An itemset $f_1f_2f_4$ is obtained from the rowset *12* during row enumeration. The rowset *12* is closed because the cardinality of all the rows in RCT_{12}, as shown in Table 4(a), is less than the cardinality of $f_1f_2f_4$. Hence, $f_1f_2f_4$ is also closed. An itemset f_2f_4 is obtained from rowset *13* during row enumeration. The rowset *13* is not closed because the cardinality of *rid 2* and *rid 6* in RCT_{13}, as shown in Table 4b, is equal to the cardinality of f_2f_4.

6.3. Rowset cardinality table based proposed pruning strategy

The RCT based pruning strategy is proposed to efficiently cut down the search space. The proposed pruning strategy utilizes RCT at every node as it provides prior information regarding the cardinality of an itemset to be mined at the descendant nodes without traversing them, unlike the existing FCCI mining algorithms, which do not provide any prior information regarding the same.

Given a rowset *Y*, if the cardinality of any *rids* in RCT_Y is less than the *mincard*, then the descendant nodes with respect to those *rids* can be pruned as they do not generate colossal itemsets. For example, with *minsup* and *mincard* set to 2, the RCT_{123}, as shown in Table 5, gives prior information regarding the cardinality of itemsets to be mined at descendant nodes (*1234, 1235* and *1236*). The cardinality of *rid 4* and *rid 5* in RCT_{123} is less than *mincard*, which leads to the pruning of nodes *1234, 12345, 123456, 12346, 1235* and *12356* as they do not generate the colossal itemsets. The existing algorithms lacks the ability to retrieve the prior information regarding the cardinality of an itemset to be mined at descendant nodes, while the proposed pruning strategy reap the benefit of prior information provided by RCT.

6.4. Parallel frequent colossal closed itesmet mining (PFCCIM) algorithm

The complete set of FCCI are mined in parallel from the high dimensional datasets by PFCCIM algorithm. The PFCCIM algorithm is shown in Algorithm 2. The Procedure 1 highlights the PFCCIM procedure consisting of a novel efficient closeness checking method and a novel efficient pruning stargey. The PFCCIM algorithm mines the FCCI by performing the parallel depth-first traversal of the bottom-up row enumerated tree. The preprocesssed bitTable, *minsup*, and *mincard* are provided as input to the PFCCIM algorithm. The FCCI, set of frequent colossal closed itemsets is initialized to null. R''_{final} is the set

Algorithm 2 PFCCIM algorithm.

Input: Pre-processed bitTable, *minsup*, *mincard*.
Output: Complete set of Frequent Colossal Closed Itemsets,FCCI
Initialisation: FCCI= ∅
R''_{final} = set of rows to be enumerated
rcomb = initial row in R''_{final}

```
#pragma omp parallel
        All bit's in bitset_result are initialized to 1
        #pragma omp for
        for (i=rcomb; i<=no_of_rows; i++)
                PFCCIM(rcomb, bitset_result)
```

Procedure 1 PFCCIM(*rcomb*, bitset_result).

```
1:  Pruning 1: if node rcomb or its descendants does not reach till minsup
2:                  return
3:  calculate bitset_result at the node rcomb
4:  Pruning 2: (for all) ∀ rid ∈ (R''final - rcomb), if RCTrcomb(rid)<mincard
5:                  delete rid from R''final
6:  Optimization: if |rcomb| <minsup
7:                      discard closeness checking
8:                  else
9:                      if Closeness_Checking(itemset,rcomb) == True
10:                         Add itemset to FCCI
11: for each row combination(r_combination) from rcomb
12:   PFCCIM(r_combination, bitset_result)
```

Procedure 2 Closeness_Checking(itemset,*rcomb*).

```
1:  Let Rfinal be set of rows in preprocessed table
2:  (for any)∀ rid ∈ (Rfinal - rcomb), if RCTrcomb(rid)==card(itemset)
3:    flag_closed = false
4:    break
5:  if flag_closed == false
6:    return False
7:  else
8:    return True
```

of rows to be enumerated. The rcomb is the initial row considered during the parallel depth-first traversal of the bottom-up row enumerated tree. The PFCCIM algorithm was implemented using the Open Multi-Processing (OpenMP) application programming interface. Multiple threads forking from the master thread parallely invoke the PFCCIM procedure to mine the FCCI from preprocessed bitTable. The PFCCIM algorithm is the first parallel row enumerated algorithm to mine FCCI from high dimensional datasets.

- **Pruning Strategy 1:** If the node *rcomb* or its descendants do not reach till *minsup* then, the *rcomb* and its descendants if existing, are pruned to cut down the search space. For example, node *6* will be pruned during the mining process if the *minsup* is set to 2 as it will not reach to the *minsup* level and nodes *46, 5, 56, 6* will be pruned during the mining process if the *minsup* is set to 3.
- The bitset_result is calculated at node *rcomb*. For example, the bitset_result at node 12 is 11100000 ($f_1f_2f_4$).
- **Pruning Strategy 2:** Pruning strategy 2 in the PFCCIM algorithm highlights the proposed RCT based pruning strategy. This strategy provides an added computational boost to the proposed PFCCIM algorithm as it provides prior information regarding the cardinality of the itemsets to be mined, whereas the existing FCCI mining algorithms do not provide prior information regarding the same.
- **Optimization:** If the number of *rid*'s in *rcomb* is less than *minsup*, then the closeness checking of the rowset *rcomb* is not required. For example, the closeness checking of all the *2*-rowsets is not required when the *minsup* is set to 3. The optimization in PFCCIM helps to skip closeness checking for (*minsup*-1) number of levels.
- **Closeness Checking:** The procedure 2 and step 9 in procedure 1 highlight the proposed novel RCT based closeness checking method. The proposed novel RCT based closeness checking is on the basis of Lemma 1 and Lemma 2. If the rowset satisfies the closeness checking, then the itemset mined from that rowset is added to FCCI. The algorithm continues with depth-first traversal of the bottom-up row enumeration space.

Table 6
Comparison of Proposed Parallel Preprocessing Technique and Preprocessing Technique used in DisClose, BVBUC, PF, CP-Miner and PCP-Miner for Ovarian Cancer dataset with *minsup* set to 20 and 30.

→ *minsup*		20				30			
↓ *mincard*		EIPP	DisClose	BVBUC	CP-Miner	EIPP	DisClose	BVBUC	CP-Miner
				PF	PCP-Miner			PF	PCP-Miner
2000	Rows	206	220	253	253	200	217	253	253
	Features	13523	13589	13589	13589	12304	12500	12500	12500
3000	Rows	174	192	253	253	166	185	253	253
	Features	13417	13589	13589	13589	11884	12500	12500	12500
4000	Rows	134	152	253	253	126	145	253	253
	Features	12906	13589	13589	13589	11194	12500	12500	12500
5000	Rows	103	125	253	253	0	99	253	253
	Features	12380	13589	13589	13589	0	12500	12500	12500
6000	Rows	0	69	253	253	0	60	253	253
	Features	0	13589	13589	13589	0	12500	12500	12500

7. Results and discussions

This section demonstrates the effectiveness of the proposed EIPP technique and the efficiency of the proposed PFCCIM algorithm. The experiments were carried out on a computer with a specification of 3.4GHz core i7-3770 CPU, 8GB RAM and 1TB hard disk. The proposed EIPP technique was compared with the preprocessing techniques of PCP-Miner, CP-Miner, BVBUC, DisClose and Pattern Fusion (PF). The DisClose algorithm outperforms the other existing FCCI mining algorithms in terms of runtime; hence, the proposed PFCCIM algorithm was compared with the DisClose algorithm. The BVBUC and Pattern Fusion algorithms cannot be considered for the runtime experimental evaluation as it fails to mine the complete set of FCCI and many of the mined FCCI from the BVBUC algorithm tend to provide incorrect support information. The PCP-Miner and CP-Miner cannot be considered for the runtime experimental evaluation as both algorithms fail to mine FCCI from the high dimensional dataset as they are designed to mine only frequent colossal itemsets and not FCCI.

The proposed EIPP technique and the proposed PFCCIM algorithm were implemented in C++ and parallelism was achieved by using the Open Multi-Processing (OpenMP) application programming interface. The preprocessing techniques of the PCP-Miner, CP-Miner, BVBUC, DisClose and PF algorithms were implemented in C++. The DisClose algorithm was also implemented in C++. The experiments were conducted on three high-dimensional datasets: ovarian cancer, lung cancer and prostate cancer [50]. Ovarian cancer consists of 253 samples (rows) and 15154 gene features, lung cancer consists of 181 samples and 12533 gene features, prostate cancer consists of 102 samples and 12600 gene features.

For different values of *minsup* and *mincard*, comparison of the proposed EIPP technique and preprocessing technique of PCP-Miner, CP-Miner, BVBUC, DisClose and PF algorithms for the ovarian cancer, lung cancer and prostate cancer datasets have been tabulated from Table 6 –11. These tables highlight the final set of relevant rows and relevant features after preprocessing. From the experimental results, it can be inferred that the proposed EIPP technique prunes all irrelevant rows and irrelevant features in all cases when compared to the existing preprocessing techniques.

The preprocessing technique of the BVBUC and PF algorithms prune the irrelevant features just once; hence it was observed from the experimental results that for a given dataset, *minsup* and varying *mincard*, the number of irrelevant features pruned by the preprocessing technique of the BVBUC and PF algorithms will remain the same. The experimental results also show that, for a given dataset, *minsup* and varying *mincard*, the preprocessing technique of the BVBUC and PF algorithms fail to prune the irrelevant rows. The preprocessing technique of the PCP-Miner and CP-Miner algorithms prune the irrelevant features and then rows which do not contain any features, only once. The experimental results show that the number of rows pruned by the preprocessing technique of the PCP-Miner and CP-Miner algorithms in all the three high dimensional datasets for a given *minsup* and *mincard* is zero. This illustrates that the number of rows in all the three high dimensional datasets that do not contain any features is zero.

The number of irrelevant features pruned by the preprocessing technique of the PCP-Miner, CP-Miner and DisClose algorithms for a given dataset, *minsup* and varying *mincard* will remain the same. The pruning of irrelevant features remains the same because these existing preprocessing techniques fail to take advantage of the reduction in the support of features due to the pruning of irrelevant rows. The experimental results highlight that the proposed EIPP technique takes the advantage of reduction in the cardinality of rows due to the pruning of irrelevant features and reduction in the support of features due to the pruning of irrelevant rows. The proposed EIPP technique prunes the irrelevant features and irrelevant rows alternatively in an iterative manner until all the remaining features and rows satisfy the criteria of *minsup* and *mincard*, respectively.

Further, the number of relevant features and relevant rows after the proposed parallel preprocessing is zero, when the (*minsup, mincard*) reaches (20, 6000), (30, 5000), (40, 5000), (50, 4000) for the ovarain cancer dataset, as shown in Tables 6 and 7. This indicates that there are no FCCI and the proposed EIPP technique yields the final results without the need for the proposed PFCCIM algorithm. However, the existing algorithms have to go through an enormous number of row

Table 7
Comparison of Proposed Parallel Preprocessing Technique and Preprocessing Technique used in DisClose, BVBUC, PF, CP-Miner and PCP-Miner for Ovarian Cancer dataset with *minsup* set to 40 and 50.

→ *minsup*		40				50			
↓ *mincard*		EIPP	DisClose	BVBUC	CP-Miner	EIPP	DisClose	BVBUC	CP-Miner
				PF	PCP-Miner			PF	PCP-Miner
2000	Rows	190	205	253	253	175	202	253	253
	Features	11079	11427	11427	11427	9818	10246	10246	10246
3000	Rows	155	169	253	253	134	160	253	253
	Features	10445	11427	11427	11427	7936	10246	10246	10246
4000	Rows	81	129	253	253	0	122	253	253
	Features	6468	11427	11427	11427	0	10246	10246	10246
5000	Rows	0	89	253	253	0	78	253	253
	Features	0	11427	11427	11427	0	10246	10246	10246

Table 8
Comparison of Proposed Parallel Preprocessing Technique and Preprocessing Technique used in DisClose, BVBUC, PF, CP-Miner and PCP-Miner for Lung Cancer dataset with *minsup* set to 20 and 30.

→ *minsup*		20				30			
↓ *mincard*		EIPP	DisClose	BVBUC	CP-Miner	EIPP	DisClose	BVBUC	CP-Miner
				PF	PCP-Miner			PF	PCP-Miner
2000	Rows	160	180	181	181	154	179	181	181
	Features	7733	7752	7752	7752	6573	6594	6594	6594
3000	Rows	145	177	181	181	140	173	181	181
	Features	7692	7752	7752	7752	6543	6594	6594	6594
4000	Rows	0	136	181	181	0	74	181	181
	Features	0	7752	7752	7752	0	6594	6594	6594
5000	Rows	0	7	181	181	0	0	181	181
	Features	0	7752	7752	7752	0	6594	6594	6594

Table 9
Comparison of Proposed Parallel Preprocessing Technique and Preprocessing Technique used in DisClose, BVBUC, PF, CP-Miner and PCP-Miner for Lung Cancer dataset with *minsup* set to 40 and 50.

→ *minsup*		40				50			
↓ *mincard*		EIPP	DisClose	BVBUC	CP-Miner	EIPP	DisClose	BVBUC	CP-Miner
				PF	PCP-Miner			PF	PCP-Miner
2000	Rows	150	177	181	181	145	176	181	181
	Features	5817	5834	5834	5834	5210	5225	5225	5225
3000	Rows	135	169	181	181	131	166	181	181
	Features	5616	5834	5834	5834	4980	5225	5225	5225
4000	Rows	0	0	181	181	0	0	181	181
	Features	0	5834	5834	5834	0	5225	5225	5225
5000	Rows	0	0	181	181	0	0	181	181
	Features	0	5834	5834	5834	0	5225	5225	5225

combinations to fetch the same final results even after applying their respective preprocessing techniques. Similarly, the same was inferred in comparison of the proposed EIPP technique and existing best preprocessing techniques for the lung cancer dataset (as shown in Tables 8 and 9) and the prostate cancer dataset (as shown in Tables 10 and 11). The proposed EIPP technique outperforms the existing preprocessing techniques effectively by pruning the complete set of irrelevant features and irrelevant rows.

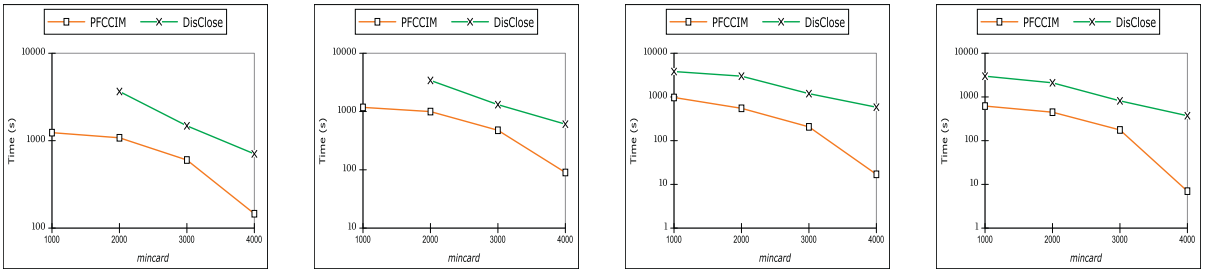
Figs. 3–5 show the runtime comparison of the proposed PFCCIM (2 threads) algorithm and the DisClose algorithm at different values of *minsup* and *mincard* for ovarian cancer, lung cancer, and prostate cancer dataset, respectively. The x-axes in Figs. 3–5 indicate the varying *mincard* and the y-axes indicate the runtime in the logarithmic scale. The Disclose algorithm was not able to record the runtime for the ovarian cancer dataset when the (*minsup, mincard*) was (10,1000) and (20,1000). After administering the proposed preprocessing technique, the number of relevant rows and relevant features in the ovarian cancer dataset is zero when the *minsup* reaches 20 and *mincard* reaches 6000. This indicates that the PFCCIM algorithm is not obligatory to gauge the final result for the ovarian cancer dataset when the *minsup* reaches 20 and *mincard* reaches 6000, whereas the DisClose algorithm has to enumerate through a huge row space to gauge the final result. Fig. 4 indicates that the PFCCIM algorithm is not obligatory to gauge the final result for the lung cancer dataset when the *minsup* reaches 20 and *mincard* reaches 4000. However, Disclose is not obligatory to gauge the final result for the lung cancer dataset

Table 10
Comparison of Proposed Parallel Preprocessing Technique and Preprocessing Technique used in DisClose, BVBUC, PF, CP-Miner and PCP-Miner for Prostate Cancer dataset with *minsup* set to 20 and 30.

→ *minsup*		20				30			
↓ *mincard*		EIPP	DisClose	BVBUC	CP-Miner	EIPP	DisClose	BVBUC	CP-Miner
				PF	PCP-Miner			PF	PCP-Miner
2000	Rows	96	102	102	102	94	101	102	102
	Features	7063	7079	7079	7079	6321	6350	6350	6350
3000	Rows	90	100	102	102	86	95	102	102
	Features	6989	7079	7079	7079	6212	6350	6350	6350
4000	Rows	86	95	102	102	81	92	102	102
	Features	6629	7079	7079	7079	6100	6350	6350	6350
5000	Rows	68	80	102	102	65	78	102	102
	Features	6358	7079	7079	7079	5849	6350	6350	6350
6000	Rows	0	10	102	102	0	3	102	102
	Features	0	7079	7079	7079	0	6350	6350	6350

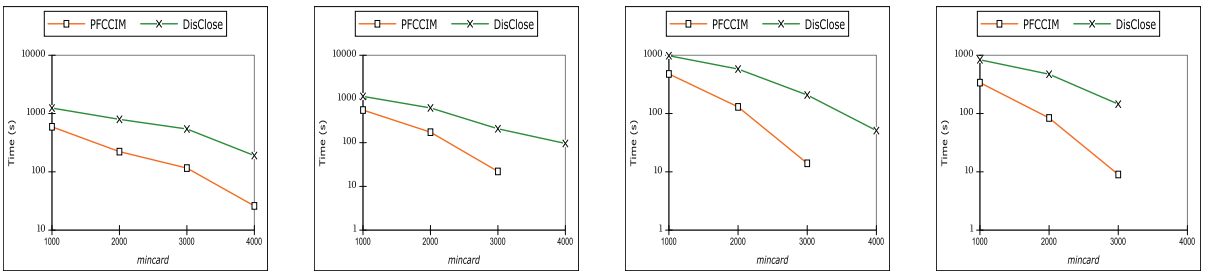
Table 11
Comparison of Proposed Parallel Preprocessing Technique and Preprocessing Technique used in DisClose, BVBUC, PF, CP-Miner and PCP-Miner for Prostate Cancer dataset with *minsup* set to 40 and 50.

→ *minsup*		40				50			
↓ *mincard*		EIPP	DisClose	BVBUC	CP-Miner	EIPP	DisClose	BVBUC	CP-Miner
				PF	PCP-Miner			PF	PCP-Miner
2000	Rows	92	101	102	102	91	100	102	102
	Features	5863	5892	5892	5892	5480	5510	5510	5510
3000	Rows	83	92	102	102	79	89	102	102
	Features	5742	5892	5892	5892	5388	5510	5510	5510
4000	Rows	77	86	102	102	65	83	102	102
	Features	5684	5892	5892	5892	5294	5510	5510	5510
5000	Rows	0	67	102	102	0	61	102	102
	Features	0	5892	5892	5892	0	5510	5510	5510
6000	Rows	0	0	102	102	0	0	102	102
	Features	0	5892	5892	5892	0	5510	5510	5510



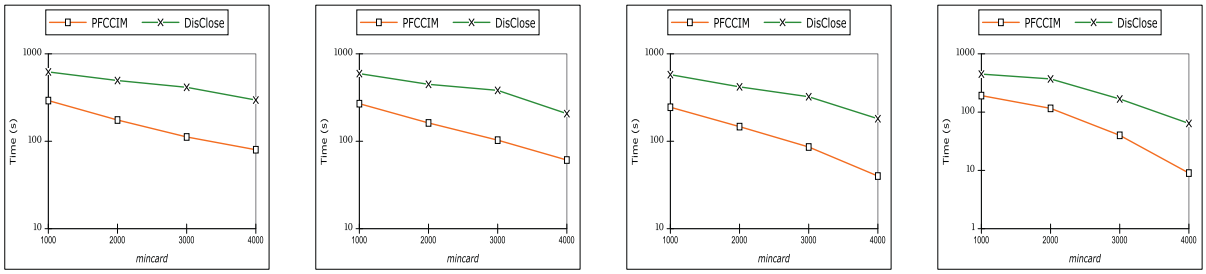
(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

Fig. 3. Runtime of PFCCIM (2 threads) and DisClose for Ovarain Cancer Dataset.



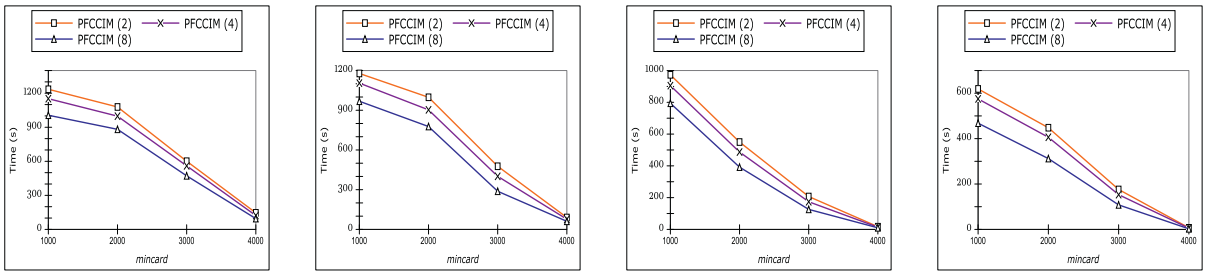
(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

Fig. 4. Runtime of PFCCIM (2 threads) and DisClose for Lung Cancer Dataset.



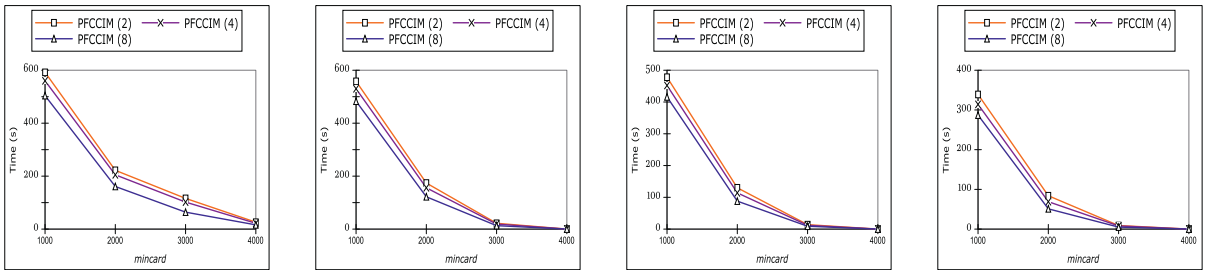
(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

Fig. 5. Runtime of PFCCIM (2 threads) and DisClose for Prostate Cancer Dataset.



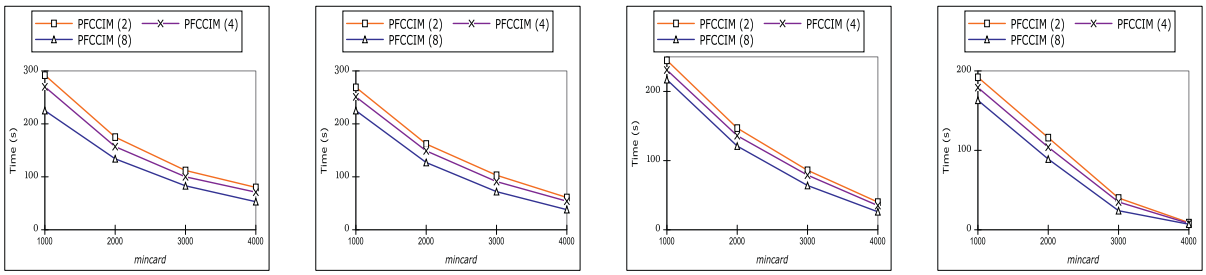
(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

Fig. 6. Runtime of PFCCIM (2 threads), PFCCIM (4 threads) and PFCCIM (8 threads) for Ovarain Cancer Dataset.



(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

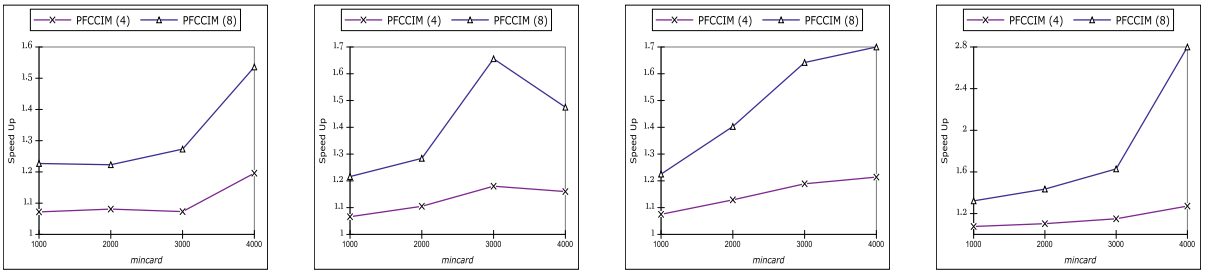
Fig. 7. Runtime of PFCCIM (2 threads), PFCCIM (4 threads) and PFCCIM (8 threads) for Lung Cancer Dataset.



(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

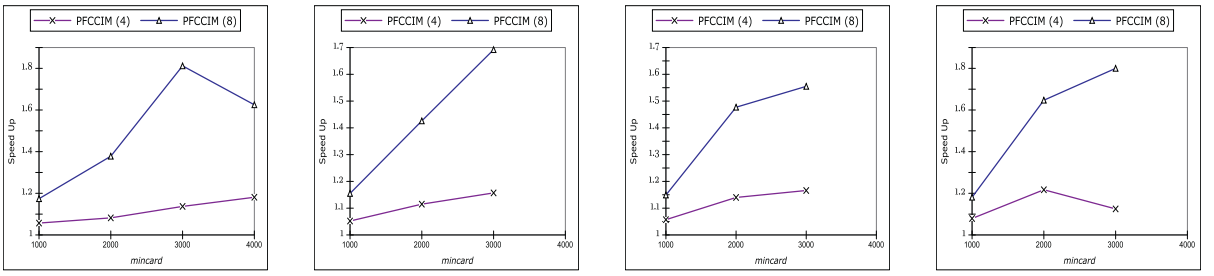
Fig. 8. Runtime of PFCCIM (2 threads), PFCCIM (4 threads) and PFCCIM (8 threads) for Prostate Cancer Dataset.

when the *minsup* reaches 30 and *mincard* reaches 5000. Figs. 3–5 show that the PFCCIM algorithm outperforms the Disclose algorithm. The results also illustrate the efficiency of the proposed RCT based closeness checking method and RCT based pruning strategy. Figs. 6–8 highlight the runtime comparison of the proposed PFCCIM algorithm with the number of threads set to 2, 4, and 8 at different values of *minsup* and *mincard* for ovarian cancer, lung cancer, and prostate cancer dataset respectively. Figs. 6–8 show that PFCCIM (8 threads) outperforms PFCCIM (4 threads) and PFCCIM (2 threads) at different values of *minsup* and *mincard* for ovarian cancer, lung cancer, and prostate cancer dataset respectively.



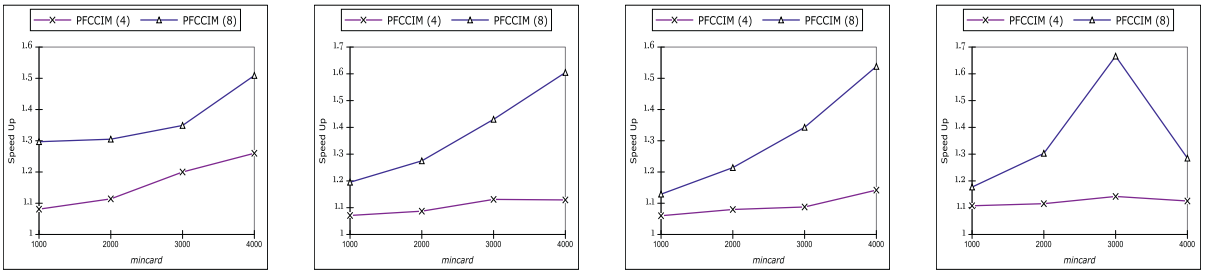
(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

Fig. 9. Speedup of PFCCIM (4 threads and 8 threads) with respect to PFCCIM (2 threads) for Ovarian Cancer Dataset.



(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

Fig. 10. Speedup of PFCCIM (4 threads and 8 threads) with respect to PFCCIM (2 threads) for Lung Cancer Dataset.



(a) minsup=10 (b) minsup=20 (c) minsup=30 (d) minsup=40

Fig. 11. Speedup of PFCCIM (4 threads and 8 threads) with respect to PFCCIM (2 threads) for Prostate Cancer Dataset.

Figs. 9–11 show the speedup of the PFCCIM algorithm (4 threads and 8 threads) with respect to PFCCIM algorithm (2 threads) for ovarian cancer, lung cancer and prostate cancer dataset respectively. The solution of mining FCCI from a high dimensional dataset corresponds to the row enumerated tree, which by nature, as shown in Fig. 1 exhibits the varying number of nodes in each tree path. This nature of the row enumerated tree will lead to the balanced speedup. The proposed PFCCIM algorithm is not obligatory to gauge the final result for the lung cancer dataset when the *minsup* reaches 20 and *mincard* reaches 4000. This indicates that there is no speedup factor when the (*minsup, mincard*) is (20, 4000), (30, 4000) and (40, 4000) as shown in Figs. 10b–d respectively. Figs. 10a and 11d highlights the decrease in speedup of PFCCIM (8 threads) with respect to PFCCIM (2 threads) for the lung cancer and prostate cancer dataset when the (*minsup, mincard*) is (10, 4000) and (40, 4000), respectively. The decrease in speedup is due to the communication overhead created by the processing threads.

7.1. Statistical significance analysis

The experiment results highlight the efficiency of the proposed PFCCIM algorithm over the DisClose algorithm. Figs. 6–8 show that PFCCIM (8 threads) outperforms PFCCIM (4 threads) and PFCCIM (2 threads) at different values of *minsup* and *mincard* for ovarian cancer, lung cancer, and prostate cancer dataset respectively. To further analyze the statistical differences among the performance of PFCCIM (2 threads), PFCCIM (4 threads) and PFCCIM (8 threads), the Wilcoxon Singed-Rank statistical significance test has been performed. This test has been selected for the statistical significance analysis as the runtime for PFCCIM (2 threads), PFCCIM (4 threads) and PFCCIM (8 threads) are not normally disturbed and the runtime

Table 12
Wilcoxon Signed-Rank Test for PFCCIM (2 threads) against PFCCIM (4 threads) and PFCCIM (8 threads) for Ovarian Cancer, Lung Cancer and Prostate Cancer Dataset.

Dataset	Algorithm	p-value[a]	Null hypothesis decision	Significant difference (if $p < 0.05$)
Ovarian Cancer	PFCCIM (4 threads)	< 0.001	Reject	Yes
	PFCCIM (8 threads)	< 0.001	Reject	Yes
Lung Cancer	PFCCIM (4 threads)	< 0.002	Reject	Yes
	PFCCIM (8 threads)	< 0.002	Reject	Yes
Prostate Cancer	PFCCIM (4 threads)	< 0.001	Reject	Yes
	PFCCIM (8 threads)	< 0.001	Reject	Yes

[a] p-values are up to three decimal point.

Table 13
Wilcoxon Signed-Rank Test for PFCCIM (2 threads) against DisClose for Ovarian Cancer, Lung Cancer and Prostate Cancer Dataset.

Dataset	Algorithm	p-value[a]	Null hypothesis decision	Significant difference (if $p < 0.05$)
Ovarian Cancer	Disclose	< 0.001	Reject	Yes
Lung Cancer	Disclose	< 0.001	Reject	Yes
Prostate Cancer	Disclose	< 0.001	Reject	Yes

[a] p-values are up to three decimal point.

has been recorded for varying values of *minsup* and *mincard* for ovarian cancer, lung cancer, and prostate cancer dataset, as shown in Figs. 6–8 respectively.

Table 12 shows the Wilcoxon Signed-Rank test for PFCCIM (2 threads) against PFCCIM (4 threads) and PFCCIM (8 threads) for ovarian cancer, lung cancer, and prostate cancer dataset. Let a null hypothesis indicate that there is no significant difference between the performance of PFCCIM (2 threads) when compared to that of PFCCIM (4 threads) and PFCCIM (8 threads) for a significance level of 5%. When $p \leq 0.05$, the Wilcoxon Signed-Rank test rejects the null hypothesis, indicating that there is a statistically significant differences among samples. When $p > 0.05$, the null hypothesis is retained and it indicates that there is no statistically significant difference among samples. Table 12 highlights that the null hypothesis was rejected for ovarian cancer, lung cancer, and prostate cancer datasets. Hence, the difference between the performance of PFCCIM (2 threads), PFCCIM (4 threads) and PFCCIM (8 threads) is statistically significant. From Figs. 6–8 and Table 12, it is evident that PFCCIM (8 threads) significantly outperforms PFCCIM (4 threads) and PFCCIM (2 threads).

PFCCIM (2 threads), being the least efficient when compared to PFCCIM (4 threads) and PFCCIM (8 threads), is considered for performing statistical significance analysis against the DisClose algorithm. Wilcoxon Singed-Rank Test has been selected for the statistical significance analysis as the runtime for PFCCIM (2 threads) and DisClose algorithm are not normally disturbed and the runtime has been recorded for varying values of *minsup* and *mincard* for ovarian cancer, lung cancer, and prostate cancer dataset as shown in Figs. 3–5 respectively. Table 13 shows the Wilcoxon Signed-Rank Test for PFCCIM (2 threads) against DisClose for ovarian cancer, lung cancer, and prostate cancer dataset. Let a null hypothesis indicates that there is no significant difference between PFCCIM (2 threads) and DisClose for a significance level of 5%. Table 13 highlights that the null hypothesis was rejected for ovarian cancer, lung cancer, and prostate cancer dataset. Hence, the difference between the performance of PFCCIM (2 threads) and that of the DisClose algorithm is statistically significant. From Figs. 3–5 and Table 13 it is evident that PFCCIM (2 threads) significantly outperforms the DisClose algorithm.

8. Conclusion and future work

Most of the running time of the conventional algorithms is expended in mining an extensive number of small and mid-sized itemsets, which does not enclose valuable and complete information. FCCI enclose valuable information and are significant for many applications dealing with high dimensional datasets. The proposed parallel preprocessing technique prunes complete set of irrelevant features and irrelevant rows, which is a major limitation of the preprocessing technique in the existing algorithms. The PFCCIM algorithm is enclosed with an efficient rowset closeness checking method and an efficient pruning strategy, which enjoys the benefit of prior information regarding the cardinality of an itemset to be mined at descendant nodes without traversing them.

Results and experiments show that the proposed parallel preprocessing technique is more effective in pruning irrelevant features and irrelevant rows than the existing preprocessing technique. The pruning of the complete set of irrelevant rows by the proposed parallel preprocessing technique helps in cutting down the row enumeration mining search space, which provides a major boost to the proposed PFCCIM algorithm. The results and experiment also highlight that the proposed PFCCIM algorithm is very efficient compared to the existing algorithms in mining FCCI from the high dimensional datasets. The maximum speedup of 2.8 is achieved by PFCCIM (8 threads) with respect to PFCCIM (2 threads). The results show a balanced speedup for the PFCCIM (8 threads) with respect to PFCCIM (2 threads) due to the varying number of nodes in each row enumerated tree path. The Wilcoxon Signed-Rank test showed that there is a statistically significant difference between the proposed PFCCIM and the DisClose algorithm for a significance level of 5%. The future work requires the combination of

different enumeration methods to mine FCCI from datasets with a large number of rows and features. Hence, future work involves proposing a parallel dynamic switching algorithm for mining FCCI from datasets consisting of a large number of rows and a large number of features. The dynamic switching algorithm dynamically switches between the row and feature enumeration to handle the changing characteristics of the data subset during the mining process.

References

[1] C.C. Aggarwal, Applications of frequent pattern mining, in: Frequent Pattern Mining, Springer, 2014, pp. 443–467.
[2] D.R. Amancio, A complex network approach to stylometry, PLoS One 10 (8) (2015) e0136076.
[3] D.R. Amancio, Probing the topological properties of complex networks modeling short written texts, PLoS One 10 (2) (2015) e0118394.
[4] Z. Chen, Q. Yan, H. Han, S. Wang, L. Peng, L. Wang, B. Yang, Machine learning based mobile malware detection using highly imbalanced network traffic, Inf. Sci. (2017).
[5] W. Li, J. Han, J. Pei, Cmar: Accurate and efficient classification based on multiple class-association rules, in: Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on, IEEE, 2001, pp. 369–376.
[6] S. Naulaerts, P. Meysman, W. Bittremieux, T.N. Vu, W.V. Berghe, B. Goethals, K. Laukens, A primer to frequent itemset mining for bioinformatics, Brief. Bioinformatics 16 (2) (2015) 216–231.
[7] L. Parsons, E. Haque, H. Liu, Subspace clustering for high dimensional data: a review, Acm Sigkdd Explor. Newslett. 6 (1) (2004) 90–105.
[8] F.N. Silva, D.R. Amancio, M. Bardosova, L.d.F. Costa, O.N. Oliveira Jr, Using network science and text analytics to produce surveys in a scientific topic, J. Informetr. 10 (2) (2016) 487–502.
[9] M.P. Viana, D.R. Amancio, L.d.F. Costa, On time-varying collaboration networks, J. Informetr. 7 (2) (2013) 371–378.
[10] B. Xue, D. Lipps, S. Devineni, Integrated strategy improves the prediction accuracy of mirna in large dataset, PLoS One 11 (12) (2016) e0168392.
[11] X. Yin, J. Han, Cpar: Classification based on predictive association rules, in: Proceedings of the 2003 SIAM International Conference on Data Mining, SIAM, 2003, pp. 331–335.
[12] N. Zhong, Y. Li, S.-T. Wu, Effective pattern discovery for text mining, IEEE Trans. Knowl. Data Eng. 24 (1) (2012) 30–44.
[13] Y. Yoon, G.G. Lee, Subcellular localization prediction through boosting association rules, IEEE/ACM Trans. Comput. Biol. Bioinf. 9 (2) (2012) 609–618.
[14] R. Agrawal, R. Srikant, et al., Fast algorithms for mining association rules, in: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, vol. 1215, 1994, pp. 487–499.
[15] J. Han, J. Pei, Y. Yin, Mining frequent patterns without candidate generation, in: ACM Sigmod Record, vol. 29, ACM, 2000, pp. 1–12.
[16] K.-C. Lin, I.-E. Liao, T.-P. Chang, S.-F. Lin, A frequent itemset mining algorithm based on the principle of inclusion–exclusion and transaction mapping, Inf. Sci. 276 (2014) 278–289.
[17] W. Song, B. Yang, Z. Xu, Index-bittablefi: an improved algorithm for mining frequent itemsets, Knowl. Based Syst. 21 (6) (2008) 507–513.
[18] S.K. Tanbeer, C.F. Ahmed, B.-S. Jeong, Y.-K. Lee, Efficient single-pass frequent pattern mining using a prefix-tree, Inf. Sci. 179 (5) (2009) 559–583.
[19] B. Yildiz, H. Selale, Mining frequent patterns from microarray data, in: Health Informatics and Bioinformatics (HIBIT), 2011 6th International Symposium on, IEEE, 2011, pp. 116–119.
[20] K.W. Lin, D.-J. Deng, A novel parallel algorithm for frequent pattern mining with privacy preserved in cloud computing environments, Int. J. Ad Hoc Ubiquitous Comput. 6 (4) (2010) 205–215.
[21] K.W. Lin, Y.-C. Lo, Efficient algorithms for frequent pattern mining in many-task computing environments, Knowl. Based Syst. 49 (2013) 10–21.
[22] M.K. Sohrabi, A.A. Barforoush, Parallel frequent itemset mining using systolic arrays, Knowl. Based Syst. 37 (2013) 462–471.
[23] K.-M. Yu, J. Zhou, Parallel tid-based frequent pattern mining algorithm on a pc cluster and grid computing system, Expert. Syst. Appl. 37 (3) (2010) 2486–2494.
[24] J. Zhou, K.-M. Yu, Balanced tidset-based parallel fp-tree algorithm for the frequent pattern mining on grid system, in: Semantics, Knowledge and Grid, 2008. SKG'08. Fourth International Conference on, IEEE, 2008, pp. 103–108.
[25] D. Apiletti, E. Baralis, T. Cerquitelli, P. Garza, P. Michiardi, F. Pulvirenti, Pampa-hd: a parallel mapreduce-based frequent pattern miner for high-dimensional data, in: Data Mining Workshop (ICDMW), 2015 IEEE International Conference on, IEEE, 2015, pp. 839–846.
[26] C. Lucchese, S. Orlando, R. Perego, Parallel mining of frequent closed patterns: Harnessing modern computer architectures, in: Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on, IEEE, 2007, pp. 242–251.
[27] B. Negrevergne, A. Termier, J.-F. Méhaut, T. Uno, Discovering closed frequent itemsets on multicore: Parallelizing computations and optimizing memory accesses, in: High Performance Computing and Simulation (HPCS), 2010 International Conference on, IEEE, 2010, pp. 521–528.
[28] S.-Q. Wang, Y.-B. Yang, G.-P. Chen, Y. Gao, Y. Zhang, Mapreduce-based closed frequent itemset mining with efficient redundancy filtering, in: Data Mining Workshops (ICDMW), 2012 IEEE 12th International Conference on, IEEE, 2012, pp. 449–453.
[29] C. Lucchese, S. Orlando, R. Perego, Fast and memory efficient mining of frequent closed itemsets, IEEE Trans. Knowl. Data Eng. 18 (1) (2006) 21–36.
[30] N. Pasquier, Y. Bastide, R. Taouil, L. Lakhal, Discovering Frequent Closed Itemsets for Association Rules, in: Database TheoryICDT99, Springer, 1999, pp. 398–416.
[31] T. Uno, M. Kiyomi, H. Arimura, Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets, FIMI, vol. 126, 2004.
[32] J. Wang, J. Han, J. Pei, Closet+: Searching for the best strategies for mining frequent closed itemsets, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 236–245.
[33] M.J. Zaki, C.-J. Hsiao, Efficient algorithms for mining closed itemsets and their lattice structure, Knowl. Data Eng. IEEE Trans. 17 (4) (2005) 462–478.
[34] H. Liu, J. Han, D. Xin, Z. Shao, Mining frequent patterns on very high dimensional data: a topdown row enumeration approach, in: Proceeding of the 2006 SIAM International Conference on Data Mining (SDM06), Bethesda, MD, SIAM, 2006, pp. 280–291.
[35] H. Liu, X. Wang, J. He, J. Han, D. Xin, Z. Shao, Top-down mining of frequent closed patterns from very high dimensional data, Inf. Sci. 179 (7) (2009) 899–924.
[36] F. Pan, G. Cong, A.K. Tung, J. Yang, M.J. Zaki, Carpenter: Finding closed patterns in long biological datasets, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 637–642.
[37] F. Pan, A.K. Tung, G. Cong, X. Xu, Cobbler: combining column and row enumeration for closed pattern discovery, in: Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on, IEEE, 2004, pp. 21–30.
[38] R. Vimieiro, P. Moscato, Disclosed: an efficient depth-first, top-down algorithm for mining disjunctive closed itemsets in high-dimensional data, Inf. Sci. 280 (2014) 171–187.
[39] R. Alves, D.S. Rodriguez-Baena, J.S. Aguilar-Ruiz, Gene association analysis: a survey of frequent pattern mining from gene expression data, Brief. Bioinformatics (2009) bbp042.
[40] P. Carmona-Saez, M. Chagoyen, A. Rodriguez, O. Trelles, J.M. Carazo, A. Pascual-Montano, Integrated analysis of gene expression by association rules discovery, BMC Bioinformatics 7 (1) (2006) 54.
[41] M. Koyutürk, Y. Kim, S. Subramaniam, W. Szpankowski, A. Grama, Detecting conserved interaction patterns in biological networks, J. Comput. Biol. 13 (7) (2006) 1299–1322.
[42] P. Manda, S. Ozkan, H. Wang, F. McCarthy, S.M. Bridges, Cross-ontology multi-level association rule mining in the gene ontology, PLoS One 7 (10) (2012) e47411.
[43] T.-L. Nguyen, B. Vo, B. Huynh, V. Snasel, L.T. Nguyen, Constraint-based method for mining colossal patterns in high dimensional databases, in: International Conference on Information Systems Architecture and Technology, Springer, 2017, pp. 195–204.

[44] T.-L. Nguyen, B. Vo, L.T. Nguyen, A new method for mining colossal patterns, in: Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on, IEEE, 2016, pp. 003119–003124.
[45] T.-L. Nguyen, B. Vo, V. Snasel, Efficient algorithms for mining colossal patterns in high dimensional databases, Knowl. Based Syst. 122 (2017) 75–89.
[46] Y. Okubo, M. Haraguchi, Finding top-n colossal patterns based on clique search with dynamic update of graph, in: Formal Concept Analysis, Springer, 2012, pp. 244–259.
[47] M.K. Sohrabi, A.A. Barforoush, Efficient colossal pattern mining in high dimensional datasets, Knowl. Based Syst. 33 (2012) 41–52.
[48] F. Zhu, X. Yan, J. Han, P.S. Yu, H. Cheng, Mining colossal frequent patterns by core pattern fusion, in: Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on, IEEE, 2007, pp. 706–715.
[49] N.F. Zulkurnain, D.J. Haglin, J.A. Keane, Disclose: discovering colossal closed itemsets via a memory efficient compact row-tree, in: Emerging Trends in Knowledge Discovery and Data Mining, Springer, 2012, pp. 141–156.
[50] Biological-Datasets, http://datam.i2r.a-star.edu.sg/datasets/krbd/index.html.