Parallel and Distributed Closed Regular Pattern Mining in Large Databases

M.Sreedevi¹, L.S.S.Reddy²

¹School of Computing, K L University, Green fields
Guntur, Andhra Pradesh, India

²Department of Computer Science & Engineering, LBRCE, JNTUK
Mylavaram, Andhra Pradesh, India

Abstract
Due to huge increase in the records and dimensions of available databases pattern mining in large databases is a challenging problem. A good number of parallel and distributed FP mining algorithms have been proposed for large and distributed databases based on frequency of item set. Not only the frequency, regularity of item also can be considered as emerging factor in data mining research. Current days closed itemset mining has gained lot of attention in data mining research. So far some algorithms have been developed to mine regular patterns, there is no algorithm exists to mine closed regular patterns in parallel and distributed databases. In this paper we introduce a novel method called PDCRP-method (Parallel and Distributed closed regular pattern) to discover closed regular patterns using vertical data format on large databases. This method works at each local processor which reduces inter processor communication overhead and getting high degree of parallelism generates complete set of closed regular patterns. Our experimental results show that our PDCRP method is highly efficient in large databases.

Keywords: Regular patterns, Closed regular patterns, Vertical data format, parallel and distributed algorithm, large databases.

1. Introduction
Now a day mining large databases is a challenging area in data mining and knowledge discovery research. Current literature survey shows that Association rule mining algorithms proposed in data mining which are not sufficient on large databases and still new solutions have to be found. Frequent pattern mining was fundamental and important in data mining research. The Apriori algorithm [1], [2] was the first algorithm to find frequent itemsets based on anti monotone property that was introduced by Agarwal et. al in 1993. Han et.al [3] introduced frequent pattern tree (FP tree) and FP growth algorithm to mine frequent patterns without candidate generation in the year 2000. These FP mining algorithms assume the data as centralized, memory resident and static. However, in real world data mining methods require to handle large databases in which these assumptions are no longer valid. Therefore, researchers focused on large scale parallel and distributed FP mining algorithms [4] to improve scalability and response time. The item is said to be frequent when its occurrence frequency is not less than the user specified minimum support threshold. Occurrence behavior is not sufficient and temporal regularity is also needed in data mining research. Regular pattern mining is one of thrust areas in data mining research. Tanbeer et.al [5] proposed an algorithm to discover regular patterns based on temporal regularity of pattern in transactional database. The authors constructed a highly compact tree structure RP tree with support descending order and a pattern growth approach to mine regular patterns in static databases. A pattern is said to be regular pattern if its regularity is less than or equal to user specified maximum regularity threshold. The significance of occurrence behavior of item can be considered in a wide range of real world applications. Vijay Kumar et. al [6] proposed an algorithm to mine regular patterns in transactional databases using vertical data format. Closed item set mining gained lot of attention than traditional mining Methods. Closed item set mining is more appropriate than traditional mining process. Wang et.al Proposed a BIDE (Bi-Directional Extension) algorithm to mine closed sequential pattern without candidate key maintenance. There is some number of parallel and distributed FP mining algorithms [7] which are developed based on Apriori and FP tree algorithms. So their performance also limited by the capabilities of Apriori technique. To the best of our knowledge no algorithm is proposed to mine closed regular patterns in parallel and distributed environment. So in this paper we propose a new method called PDCRP to mine closed regular patterns in parallel and distributed databases with vertical data format on large databases. In this
process first we mine regular itemsets and then mine 
closed regular patterns by considering global 
maximum regularity and minimum support. Our 
method mines the local data base using vertical data 
format to discover all possible closed regular patterns 
globally with input inter process communication 
among processors.
The remaining of this paper is organized as follows. 
Section 2 describes related work section 3 describes 
problem definition section 4 describes process of 
moving closed regular pattern mining and section 5 
describes experimental results and finally in section 6 
we conclude this paper.

2. Related Work

Mining regular patterns is one of the thrust area in 
data mining research. Recently Tanbeer et.al [4] 
introduced a new problem to mine regular patterns in 
transactional database which follows regularity of 
data item in their occurrence behavior. They 
proposed an algorithm called RP-tree algorithm and 
the construction process of RP-tree is similar to 
construction process of FP-tree construction technique, 
in which RP-tree maintains transaction ids at 
each node than support count maintenance in FP- 
tree. In this process it uses two database scans. With 
the first database scan it creates a header table called 
regular table which stores its regularity and support 
values of items. In the second scan RP-tree is 
constructed based on previously R-table for regular 
item sets.

Mining closed item sets has been gained lot of 
attention in present days than traditional frequent 
mining techniques. Shengnan Cong et.al[8] proposed 
algorithm called Par-CSP to accomplish parallel 
moving of sequential patterns on distributed memory 
systems. In this process they adopt the divide and 
conquer method to minimize the inter process 
communication overhead and also use selective 
sampling technique to address load imbalance 
problem. Mafruz Zaman et.al [9] proposed an 
algorithm called ODAM (Optimized Distributed 
Association Rule Mining) algorithm to minimize 
communication cost. Tanbeer et.al [10] proposed a 
novel algorithm based on tree structure called PP-tree 
(parallel pattern tree) to mine frequent patterns in 
algorithm MLFPT(multiple local frequent patterns) 
for parallel mining of frequent patterns based on FP-
growth algorithm by having two data base scans by 
eliminating the candidate items need. MLFPT 
approach implemented in two stages: In the first 
stage parallel frequent pattern trees are constructed 
and mining process will proceed on these constructed 
trees in the second stage. In this paper we used 
vertical data format to mine closed regular items in 
parallel and distributed environment of large 
databases. The advantage of vertical data format [12] 
[13] is require only one database scan, it uses simple 
operations like union, intersection, deletion etc. Non 
regular items are pruned in this format only.

3. Problem Definition

In this section we describe the concepts of period of 
item, regular pattern mining, closed regular pattern 
moving and also define the problem to obtain 
complete set of closed regular patterns in parallel and 
distributed environment.

Let \( I = \{ i_1, i_2, i_3, \ldots, i_n \} \) be a set of items. A set \( X = \{ i_1, i_2, i_3, \ldots, i_j \} \subseteq I \), where \( j \leq k \) and \( j, k \in [1, n] \) is 
called a pattern or an item set and \( T = (\text{tid}, X) \) where 
\( T \) is a transaction in database DB, \( \text{tid} \) is unique 
transaction identifier and \( X \) is a pattern. The size of 
the database DB is noted as \( m = |DB| \), transaction set 
\( T \) over the database DB is denoted as 
\( T = \{ t_1, t_2, t_3, \ldots, t_m \} \).

3.1 Definition 1 (Period of \( X \))

Assume \( t_j^x \) and \( t_{j+1}^x \) are two consecutive transaction 
in database at one processor. The period of item \( x \) can 
be defined as number of transactions between \( t_j^x \) and 
\( t_{j+1}^x \), \( p^x = t_{j+1}^x - t_j^x \), we consider the first transaction is 
\( t_{\text{first}}^x \) which is null transaction i.e \( t_{\text{first}} = 0 \) and last 
transaction is \( t_{\text{last}}^x \) which is last transaction of the 
database at one processor. Period of item \( X \) can be 
defined as the number of times item \( X \) appears in 
different transactions.

3.2 Definition 2 (Regularity of \( X \))

if regularity of \( X \) is less than or equal to user given 
regularity threshold then item \( X \) is said to be regular 
itemset.

3.3 Definition 3 (Closed Regular Itemset)

Assume \( X = \{ x_1, x_2, x_3, \ldots, x_n \} \) be a set of regular 
itemsets and \( Y = \{ y_1, y_2, y_3, \ldots, y_n \} \) be other set of 
regular itemsets, where \( X \subseteq Y \) that is \( X \) is subset of \( Y \) 
and \( Y \) is super set of \( X \). Support count of \( Y \) must not 
be greater than support count of \( X \) then \( X \) is called 
closed regular itemset.
3.4 Definition 4 (Parallel and Distributed Closed Regular Pattern)

Assume DS = \{p_1, p_2, p_3, \ldots, p_k\} be a number of partitions in parallel and homogeneous distributed system. The database \( DB \) is divided into \( n \) number of equal partitions as \( db_1, db_2, db_3, \ldots, db_n \) and each partition \( db_i \) is assigned to each individual processor \( p_i \). Let regularity of item \( X \) is represented as \( \text{reg}_i(X) \) in \( db_i \) and support count of item \( X \) is represented as \( \text{sup}_i(X) \) in \( db_i \). We describe \( \text{reg}_i(X) \) and \( \text{sup}_i(X) \) as global regularity threshold and global support count of item \( X \) in database \( DB \) respectively. \( \lambda \) is user given minimum regularity threshold and \( \delta \) is user given minimum support count. We accumulate all \( \text{reg}_i(X) \) and \( \text{sup}_i(X) \) from each processor to find global regularity \( \text{reg}(X) \) and global support count \( \text{sup}(X) \) respectively. Closed regular pattern \( X \) is mined which satisfies user given global regularity and global minimum support.

4. PDCRP Method

In this section we describe our proposed PDCRP method to mine closed regular patterns in parallel and distributed environment. We consider distributed environment which contains different locations where each location contains resources like processor, memory, etc. Consider the large database and divide this database into small number of partitions equal in size, non-overlapping partitions in order to distribute for \( n \) number of processor. We consider the instance database [14] in the process of mining closed regular patterns in the parallel and distributed environment on large databases.

PDCRP method is implemented in two phases. In the first phase regular patterns are mined in parallel based on user given regularity threshold and closed regular patterns are mined from previously mined regular patterns based on user given minimum support threshold.

**Phase I**:
Input: \( DB, \lambda = 5 \)
Output: Set of regular patterns

**Procedure**:

1. Let \( X_i \subseteq I \) a \( k \)-item set at one processor
2. \( P_i^\lambda = 0 \) for all \( X_i \)
3. For each \( X_i \)
4. Find the period of \( X_i \)
5. \( P_i^{\lambda} = P_{i+1}^{\lambda} - P_i^{\lambda} \)
6. \( \text{reg}(X_i) = \max(P_i^{\lambda}) \)
7. if \( \text{reg}(X_i) \leq \lambda \)
8. \( X_i \) is regular itemset
9. Else
10. Delete \( X_i \)
11. Repeat the steps 2 to 10 for \( P_i^{\lambda} \) items.

In phase I we find set of regular patterns at each local processor based on regularity threshold. For example periodicity of itemsets and regular itemsets are shown in table 4 at two local processors.

Table 1 contains database at two processors \( P_1 \) and \( P_2 \) having nine transaction each. The horizontal database is converted into vertical format at each processor using one database scan which are shown in table 2. In this table the item set \( \{d\} \) is appeared in the transactions \( \{1, 3, 4, 5, 6, 7, 8\} \) at processor \( P_1 \) and also appeared in the transactions \( \{1, 2, 3, 4, 5, 6, 7, 8\} \) at processor \( P_2 \). Each processor contains equal number of transactions which are collected from large database every time. So in this method we also maintain the load balancing property. We consider global maximum regularity \( \lambda = 5 \) and global minimum support \( \delta = 10 \), two measures to mine closed regular patterns.

<table>
<thead>
<tr>
<th>Tid</th>
<th>Transaction at P1</th>
<th>Transaction at P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a, b, c, d</td>
<td>b, c, d, e</td>
</tr>
<tr>
<td>2</td>
<td>a, b</td>
<td>a, c, d</td>
</tr>
<tr>
<td>3</td>
<td>a, c, d, e</td>
<td>a, d, e</td>
</tr>
<tr>
<td>4</td>
<td>b, d, e</td>
<td>a, b, c, d, e</td>
</tr>
<tr>
<td>5</td>
<td>a, c, d, e</td>
<td>a, c, d, e</td>
</tr>
<tr>
<td>6</td>
<td>b, c, d</td>
<td>c, d</td>
</tr>
<tr>
<td>7</td>
<td>a, d, e</td>
<td>b, c, d</td>
</tr>
<tr>
<td>8</td>
<td>a, b, c, d, e</td>
<td>b, d, e</td>
</tr>
<tr>
<td>9</td>
<td>a, b, c</td>
<td>a, b, c</td>
</tr>
</tbody>
</table>
Table 2: vertical data format.

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Tids at P1</th>
<th>Tids at P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1, 2, 3, 5, 7, 8, 9</td>
<td>2, 3, 4, 5, 9</td>
</tr>
<tr>
<td>b</td>
<td>1, 2, 4, 6, 8, 9</td>
<td>1, 4, 7, 8,9</td>
</tr>
<tr>
<td>c</td>
<td>1, 3, 5, 6,8,9</td>
<td>1, 2, 4, 5,6,7,9</td>
</tr>
<tr>
<td>d</td>
<td>1, 3, 4, 5, 6, 7, 8</td>
<td>1, 2, 3, 4, 5, 6, 7,8</td>
</tr>
<tr>
<td>e</td>
<td>3, 4, 5, 7, 8, 9</td>
<td>1, 3, 4, 5, 8</td>
</tr>
</tbody>
</table>

Table 3 represents PDCRP header table which simulates regularity and support values of each item set at every processor. This table is somewhat similar to pp-tree [15] header table that collects all local regularity and global support values respectively. Our PDCRP method works to mine closed regular patterns in large databases.

Table 3: PDCRP Header table

<table>
<thead>
<tr>
<th>Item s</th>
<th>P1</th>
<th>P2</th>
<th>----</th>
<th>Pn</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>i1</td>
<td>regi1</td>
<td>regi1</td>
<td>----</td>
<td>regmaxi1</td>
<td>max(regi1)</td>
</tr>
<tr>
<td></td>
<td>supi1</td>
<td>supi1</td>
<td>----</td>
<td>supmaxi1</td>
<td>Σsupi1</td>
</tr>
<tr>
<td>i2</td>
<td>regi2</td>
<td>regi2</td>
<td>----</td>
<td>regmaxi2</td>
<td>max(regi2)</td>
</tr>
<tr>
<td></td>
<td>supi2</td>
<td>supi2</td>
<td>----</td>
<td>supmaxi2</td>
<td>Σsupi2</td>
</tr>
<tr>
<td>im</td>
<td>regim</td>
<td>regim</td>
<td>----</td>
<td>regmaxim</td>
<td>Max(regim)</td>
</tr>
<tr>
<td></td>
<td>supim</td>
<td>supim</td>
<td>----</td>
<td>supmaxim</td>
<td>Σsupim</td>
</tr>
</tbody>
</table>

Table 4 represents PDCRP local tables with P^i and reg_i.

<table>
<thead>
<tr>
<th>Items</th>
<th>Processor P1</th>
<th>Processor P2</th>
<th>Max(reg_i, reg_j)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1, 1, 1, 2, 2, 1, 1</td>
<td>2, 1, 1, 1, 4</td>
<td>4</td>
</tr>
<tr>
<td>b</td>
<td>1, 1, 2, 2, 2, 1</td>
<td>1, 3, 3, 1, 1</td>
<td>3</td>
</tr>
<tr>
<td>c</td>
<td>1, 2, 2, 1, 2, 2, 2</td>
<td>1, 1, 2, 1, 1, 1</td>
<td>2</td>
</tr>
<tr>
<td>d</td>
<td>1, 2, 1, 1, 1, 1, 1</td>
<td>1, 1, 1, 1, 1, 1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4 represents regularity of itemsets at each processor. Itemset < a > contains its regularities <1, 1, 1, 1, 1, 1>, its regularity is 2 at processor P1 and itemset < b > contains its regularities <2, 1, 1, 1, 1, 1>, its regularity is 4 at processor P2. Consider the maximum regularity of all regularity values at different processors as the regularity of that itemset. The one itemsets < a, b, c, d, e > are regular itemsets based on global regularity threshold λ = 5. Similarly, we find regular two itemsets, three itemsets and so on.

Phase II

Input: DB with Regular item sets, δ =10 (sup-count)

Output: complete set of closed-regular patterns.

1. Let X_i be a regular k-item set
2. Let X_j be a regular k + m item set
3. m=1,2,3 ………n
4. X_i ⊂ X_j for all i <= j
5. Find Sup(X_i), sup-count of X_i
6. Find Sup(X_j), sup-count of X_j
7. If Sup(X_i) > Sup(X_j)
8. X_i is closed-regular item set
9. Else
10. X_i is not closed-regular item set

In phase II we find complete set of closed regular patterns based on user given support count which is considered globally. That is support count of itemset is sum of support counts of itemset at each processor. The support count of itemset is not less than its immediate super set, the item is said to be closed itemset.
Table 5: One itemsets with local and global supports.

<table>
<thead>
<tr>
<th>Items</th>
<th>Sup_i at P_1</th>
<th>Sup_i at P_2</th>
<th>( \Sigma(Sup_{P_1, Sup_{P_2}}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>7</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>b</td>
<td>6</td>
<td>5</td>
<td>11</td>
</tr>
<tr>
<td>c</td>
<td>6</td>
<td>7</td>
<td>13</td>
</tr>
<tr>
<td>d</td>
<td>7</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>e</td>
<td>6</td>
<td>5</td>
<td>11</td>
</tr>
</tbody>
</table>

In table 5 the itemsets \(< a, b, c, d, e >\) which satisfies both global regularity \( \lambda \) and global minimum support count \( \delta \).

Table 6: PDCRP header table with two itemsets

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Reg_i x at P_1</th>
<th>Reg_i x at P_2</th>
<th>Max(reg)</th>
<th>Sup_i x at P_1</th>
<th>Sup_i x at P_2</th>
<th>( \Sigma(Sup_{P_1, P_2}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>a,b</td>
<td>6</td>
<td>5</td>
<td></td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>a,c</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>–</td>
</tr>
<tr>
<td>a,d</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>a,e</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>b,c</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>b,d</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>b,e</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>c,d</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>c,e</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>d,e</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 6 contains two itemsets with regularity values and support values at each local processor. Itemset \(< a, b >\) has regularity 6 at \( P_1 \) and it has regularity 5 at processor \( P_2 \). Consider the maximum of the two regularity values i.e 6 which does not satisfies the minimum regularity threshold value, so itemset \(< a, b >\) is not regular itemset. The itemsets \(< (a, c), (a, d), (a, e), (b, c), (b, d), (b, e), (c, d), (c, e), (d, e) >\) are regular two itemsets which satisfied minimum regularity threshold considered at global header table. Regular one Itemsets \(< a >, < b >, < c >, < d >, < e >\) are closed regular itemsets, that is support counts of regular one itemsets is greater than the support counts of immediate regular two itemsets which are represented in table 5 and table 6. So itemsets \(< a, d >, < c, d >, < d, e >\) which satisfies minimum support threshold value.

The itemsets \(< a >\) contains support count of 12 which is greater than the support count 10 of its immediate super set \(< a, d >\). So itemset \(< a >\) is closed regular itemset. Similarly itemsets \(< b >, < c >, < d >, < e >\) also closed regular one itemsets. Similarly we repeat this process for remaining two itemsets, three itemsets and so on.

In this paper PDCRP header table collects all global regularities and global supports to find closed regular itemsets. However this method works in parallel and distributed environment for large databases which does not require inter processor communication, it requires only one communication at the time of constructing PDCRP header table and also minimizes I/O cost. Hence our proposed PDCRP method is highly efficient for large databases.

5. Experimental Results

In this section we describe the results of our proposed method. We implemented this method from real(Kosarak) and synthetic (T1014D100K) datasets which are usually use in frequent pattern mining http://cvs.buu.ac.th/mining/Datasets/synthesis_data/ and UCI Machine Learning Repository (University of California – Irvine, CA), these are used by Almanden Quest research group to develop frequent patterns in mining process. In our PDCRP method the horizontal database at each local processor is converted into vertical data format. Every processor contains equal number of transactions while mining process is going on. So load balancing among all the processors is also one of factor we considered.

![Fig. 1 Execution time on T1014D100K](image)

We used the systems with 2.66 GHz CPU with 2 GB main memory on Windows XP. We had written programs in java. We distribute the database among processors, so every processor has complete access to its database. We account the results on synthetic dataset T1014D100K which contains 100K transactions, 870 items and its average transaction length is 10.10. W represent execution time for different reg() and sup() values in figure 1.
We also account the results on the kosarak real dataset that contains 990 transactions, 41,270 items, and its average length of transaction is 8.10 for different reg() and sup() values which are represented in figure 2.

6. Conclusion

In this paper we introduced a novel approach to mine closed regular patterns using vertical data format in large databases. It takes only one database scan to convert horizontal database to vertical database format. This PDCRP method works in parallel and distributed environment to mine complete set of closed regular patterns based on user given global regularity and support values which minimize I/O cost and no inter processor communication will takes place among processors. So parallel computing is essential component for mining large databases in data mining applications.

References

[8] Shengnan Cong, jiawei Han, David Padua “Parallel Mining of Closed sequential patterns” In ACM KDD, 2005.