International Journal of Data Science and Analytics manuscript No.
(will be inserted by the editor)

An Incremental Framework to Extract Coverage Patterns for Dynamic Databases

Komallapalli Kaushik* · P. Krishna Reddy · Anirban Mondal · Akhil Ralla

Received: date / Accepted: date

Abstract Pattern mining is an important task of data mining and involves the extraction of interesting associations from large transactional databases. Typically, a given transactional database D gets updated due to the addition and deletion of transactions. Consequently, some of the previously discovered patterns may become invalid, while some new patterns may emerge. This has motivated significant research efforts in the area of *incremental mining*. The goal of *incremental mining* is to *efficiently* mine patterns when D gets updated with additions and/or deletions of transactions as opposed to mining all of the patterns from scratch. Incidentally, active research efforts are being made to develop incremental pattern mining algorithms for extracting frequent patterns, sequential patterns and utility patterns. Another important type of pattern is the *coverage pattern (CP)*, which has significant applications in areas such as banner advertising, search engine advertising and visibility mining. However, none of the existing works addresses the issue of incremental mining for extracting CPs. In this regard, the main contributions of this work are three-fold. First, we introduce the problem of incremental mining of CPs. Second, we propose an approach, designated as Comprehensive Coverage Pattern Mining (CCPM), for *efficiently* extracting CPs under the incremental paradigm. Third, we performed extensive experiments using two real clickstream datasets and one synthetic dataset to demonstrate the overall effectiveness of our proposed approach.

Keywords Data Mining · Pattern Mining · Coverage Patterns · Dynamic Databases · Incremental Mining

Komallapalli Kaushik*
Kohli Centre on Intelligent Systems
IIIT, Hyderabad, India.
E-mail: kaushik.k@research.iiit.ac.in

P.Krishna Reddy
Kohli Centre on Intelligent Systems
IIIT, Hyderabad, India.
E-mail: pkreddy@iiit.ac.in

Anirban Mondal
Ashoka University, Sonipat, India.
E-mail: anirban.mondal@ashoka.edu.in

Akhil Ralla
Kohli Centre on Intelligent Systems
IIIT, Hyderabad, India.
E-mail: ralla.akhil@research.iiit.ac.in

1 Introduction

The task of pattern mining involves the extraction of interesting associations from large transactional databases [3, 4, 21, 37]. Pattern mining has several important and diverse applications in areas such as market basket analysis, recommendation systems and Internet advertising. Typically, a given transactional database D gets updated due to the addition and deletion of transactions. Consequently, some of the previously discovered patterns may become invalid, while some new patterns may also emerge.

The naïve approach for pattern extraction would be to run a given pattern mining algorithm *from scratch* whenever D encounters any addition and/or deletion of transactions. However, we can intuitively understand that this naïve approach would suffer from the drawback of high computational overhead. This has motivated significant research efforts in the area of *incremental mining*. The goal of *incremental mining* is to *efficiently* mine patterns when D gets updated with additions and/or deletions of transactions as opposed to mining all of the patterns from scratch. Given the

knowledge of patterns from D, a set Δ^+ of transactions that was added to D and a set Δ^- of transactions that was deleted from D, the goal of incremental mining is to develop *efficient* pattern mining approaches for generating knowledge of patterns from the the new database, say D', which comprises D and the effect of Δ^+ and Δ^-, as opposed to extracting the patterns from scratch.

Incidentally, the paradigm of *incremental pattern mining* is an active research area. Hence, several incremental approaches have been proposed to extract the knowledge of frequent patterns [8, 15, 33], sequential patterns [28, 30, 41] and utility patterns [6, 31, 45]. Observe that if we were to run any existing pattern mining approach from scratch on the new database (i.e., D'), the computational workload would be likely to be significantly higher than if we were to extract the new patterns and validated the already extracted patterns to remove the invalid patterns. Hence, there is an opportunity to improve the performance of pattern extraction by using an incremental approach.

Another important type of pattern is the *coverage pattern (CP)* [37, 38]. A given CP represents the knowledge of the extent of coverage of transactions (in a given transactional database) by a set of items or a pattern. The process of coverage pattern mining extracts the transactional coverage value of all patterns. Notably, coverage pattern mining has significant applications in areas such as banner advertising [26, 40], search engine advertising [9–11] and visibility mining [18]. Further details concerning CPs are provided in Section 2.2.

Incidentally, the issue of extracting CPs under the incremental mining paradigm has not been investigated in the literature. Notably, the existing incremental frameworks cannot be easily extended to develop an incremental mining approach for extracting CPs. This is because the extraction of CPs involves the sorting of a given list of items based on item frequencies. Hence, in contrast with existing incremental mining approaches, we additionally need to address the issue of change in the sorted order of items when D is updated due to the addition and/or deletion of transactions.

In this paper, given the CPs of D, we propose a novel approach, which we designate as **Comprehensive Coverage Pattern Mining (CCPM)**. In particular, CCPM extracts CPs when a set Δ^+ of transactions is added to D and/or a set Δ^- of transactions is deleted from D. As previously discussed, due to the addition and/or deletion of transactions from D, some of the existing CPs may become invalid and some new CPs may emerge. The existing patterns in D are validated by scanning Δ^+ and Δ^-. We generate candidate patterns in a level-wise manner by scanning Δ^+ and Δ^- and validate the new patterns in D.

Recall that the extraction of CPs needs to address the issue of change in the sorted order of items when D is updated due to the addition and/or deletion of transactions. Let the list of items sorted based on their frequencies in D be represented as Frequency list ($Flist$). Under our proposed CCPM approach, the number of patterns to be validated depends upon the extent of change from the $Flist$ of D to the $Flist$ of the new database (i.e., D'). In this regard, the proposed CCPM approach is developed by considering the effect of the number of additional patterns due to the change in the $Flist$.

The main contributions of this work are three-fold:

1. We introduce the problem of incremental mining in the context of coverage patterns.
2. We propose an approach, designated as Comprehensive Coverage Pattern Mining (CCPM), for *efficiently* extracting CPs under the incremental paradigm.
3. We performed extensive experiments using two real click-stream datasets and one synthetic dataset to demonstrate the overall effectiveness of our proposed approach.

In this paper, we have proposed an incremental approach for extracting CPs by considering a level-wise CP mining algorithm [37]. As a part of future work, we also plan to investigate an incremental approach for extracting CPs by considering a pattern growth approach [38].

We have made a preliminary effort in [35] by considering *only* the case of the addition of transactions to D. In this paper, we have made significant extensions to the approach proposed in [35] primarily in the following ways. First, we have extended the approach and the corresponding algorithms by considering (i) the case of deletion of transactions from D and (ii) the case of simultaneously adding a set of new transactions to D and deleting a set of transactions from D. Second, we have added a much more comprehensive set of experiments. Furthermore, we have considerably improved the presentation as well as the overall readability of the paper.

The remainder of this paper is organized as follows. In Section 2, we discuss existing works as well as background information concerning coverage pattern mining. In Section 3, we present the proposed framework of the problem. In Section 4, we discuss our proposed approach. In Section 5, we report the performance evaluation. Finally, we conclude in Section 6 with directions for future work.

2 Related Work and Background

This section discusses existing works as well as background information about coverage pattern mining.

2.1 Related work

In the literature, mining of incremental databases was first studied in the context of frequent patterns by means of the FUP algorithm [15]. In FUP algorithm the existing knowledge is reused to extract new association rules in updated database. The work in [16] extended the FUP algorithm to additionally consider the deletion of transactions. Additionally, the *Borders* algorithm [7] has been proposed for generating associations in an incremental manner for dynamic databases. The work in [2] proposed an approach that extends the FP-tree to incrementally mine frequent patterns in dynamic databases. Moreover, a Dynamic Frequent Pattern Growth algorithm [1] was proposed to mine the frequent patterns from dynamic databases. In particular, it focuses on building a new Frequent Pattern tree purely based on the new transactions. The work in [43] has discussed the mining of weighted maximal frequent itemsets from dynamic databases.

Given that data streams can also be perceived as a type of dynamic database, efforts have been made towards mining frequent patterns in data streams. The work in [39] discussed a technique to discover the complete set of recent frequent patterns from a high-speed data stream over a sliding window by means of the Compact Pattern Stream tree. Moreover, the work in [27] discussed weighted maximal frequent pattern mining over data streams based on the sliding window model for obtaining weighted maximal frequent patterns that reflect recent information about data streams. The work in [25] discussed ways to mine maximal frequent patterns with Apache Spark. The work in [17] discussed the mining of frequent patterns in data streams using a Trie data structure.

Incremental mining has also been studied in the context of sequential patterns and utility patterns. An iterative algorithm has been proposed in [30] for extracting sequential patterns in incremental databases. Further improvements in efficiency were performed by means of the IncSpan algorithm [14, 32]. The IncSpan algorithm improves efficiency by maintaining almost frequent sequences and also by using optimization techniques. A utility sequential pattern mining algorithm, which uses a candidate pattern tree data structure, has been proposed in [41]. The work in [46] proposed a fast updated sequential pattern tree to update the discovered sequential patterns associated with sequence insertion or deletion. Moreover, the work in [24] discussed how to handle the incremental nature of big data and mine regular high-utility sequential patterns from dynamic databases.

Moreover, the work in [23] discussed an approach to maintain the set of sequential patterns from itemset-sequence streams with a transaction-sensitive sliding window. Furthermore, the work in [29] discussed an algorithm based on sequence alignment for mining approximate sequential patterns in web usage data streams. The work in [12] discussed an algorithm to mine closed sequential patterns in stream windows incrementally. The work in [22] discussed a Batch-Free Sequential Pattern Miner algorithm for mining patterns in streams as opposed to traditional batch-based processing. Moreover, the work in [42] discussed the updating of sequential patterns such that only the added transactions need to be processed for updating the existing sequential patterns.

The work in [6] discussed three types of tree structures, namely the Incremental High Utility Pattern (HUP) lexicographic tree, the Incremental HUP transaction frequency tree and the HUP transaction weighted utilization tree. These tree structures are used for mining high utility patterns in incremental databases. Moreover, the one-pass algorithm proposed in [45] uses list-based data structures, which work without generating candidates. Furthermore, the algorithm in [31] proposed a novel structure for reducing the number of transaction scans and to speed up the pattern mining process.

The work in [36] introduced two techniques for reducing overestimated utilities and suggested a tree-based data structure to maintain information of stream data and high-utility patterns. Moreover, the work in [44] discussed a new list-based data structure for erasable pattern mining. In particular, this list-based data structure is able to efficiently store and process stream data on sliding window-based data streams. The approach in [20] discussed the global revision header table, which is used to store the items and the transaction utilities of the current data domain that need to be processed. Here, a utility tree is used to store all of the utility information on the itemsets in the transactions in order to avoid multiple scans of the dataset.

However, *none* of these existing works has addressed incremental mining in the context of coverage patterns.

2.2 Overview of Coverage Pattern Mining

First, we explain the model of coverage patterns. Next, we provide an overview of a level-wise pruning-based coverage pattern mining (CPM) algorithm.

2.2.1 Model of coverage patterns

Let $I = \{i_1, i_2....., i_n\}$ be the set of items. Let D be the transactional database. Each transaction $T \in D$ comprises a set of items, i.e., $T \subseteq I$. The set of transactions, which contain the item i_p, is denoted by T^{i_p}. A pattern is a set of items. We shall now explain the notions of *relative frequency (RF)*, *coverage support (CS)* and *overlap ratio (OR)* in the context of coverage patterns.

Relative frequency (RF) of item i_p: The RF of item i_p, denoted by $RF(i_p)$, is equal to the ratio of the number of transactions that contain item i_p to D, i.e., $RF(i_p) = \frac{|T^{i_p}|}{|D|}$. An item is considered to be frequent if its $RF \geq minRF$, where $minRF$ is a user-specified threshold value.

Let X be a pattern. We now explain the notion of CS of pattern X after defining the notion of *coverage set* of a pattern X.

Coverage set ($CSet$) of a pattern X: Given a pattern $X = \{i_p, i_q, \ldots, i_r\}$, $(1 \leq p, q, r \leq n)$, $CSet(X)$ is the set of all transactions that contains at least one item of pattern X i.e., $CSet(X) = T^{i_p} \cup T^{i_q} \cup \ldots \cup T^{i_r}$.

The CS of X is the fraction of transactions covered by X.

Coverage support (CS) of a pattern X: Given $X = \{i_p, i_q, \ldots, i_r\}$, $(1 \leq p, q, r \leq n)$, $CS(X)$ is the ratio of the size of $CSet(X)$ to $|D|$ i.e., $CS(X) = \frac{|CSet(X)|}{|D|}$.

The notion of Overlap Ratio (OR) is defined to capture the *overlap* of pattern X. For computing the value of $OR(X)$, we sort the items of X in descending order based on item frequencies. We initialize X with the most frequent item and then we add the second-most frequent item. In this manner, we keep adding new items to X based on the value of OR (in descending order of item frequencies). We compute the value $OR(X)$ as follows.

Overlap ratio (OR) of a pattern X. Given $X = \{i_p, \ldots, i_q, i_r\}$, $(1 \leq p, q, r \leq n)$ and $|T^{i_p}| \geq \ldots \geq |T^{i_q}| \geq |T^{i_r}|$, $OR(X)$ is the ratio of the number of common transactions between $CSet(X - i_r)$ and T^{i_r} to the number of transactions having item i_r, i.e.,
$OR(X) = \frac{|CSet(X - i_r) \cap T^{i_r}|}{|T^{i_r}|}$

For a pattern X, $OR(X) \in [0, 1]$. Notably, the value of $OR(X)$ equals zero if X contains a single item. The intuition of OR is as follows. Suppose we want to increase the coverage by adding a new item i_k to the pattern X. Let the coverage set of item i_k be $CSet(i_k)$. Notably, adding i_k to X could result in overlap between $CSet(X)$ and $CSet(i_k)$. If the value of OR is high, $CSet(X \cup i_k)$ may not increase significantly over $CSet(X)$. Hence, in such a case, adding i_k to X would not be interesting. In essence, adding a new item i_k to set X would be interesting only if there is a low value of overlap ratio value. Hence, a pattern is interesting if it has high CS and low OR values, where a high value of CS indicates more coverage of transactions and a lower value of OR indicates less overlap among the transactions covered by the items.

Coverage pattern (CP). A pattern X, where $X = \{i_p, ..., i_q, i_r\}$, $(1 \leq p, q, r \leq n)$ and $|T^{i_p}| \geq ... \geq |T^{i_q}| \geq |T^{i_r}|$ is called a CP if $OR(X) \leq maxOR$, $CS(X) \geq minCS$ and $RF(i_k) \geq minRF \; \forall i_k \in X$, where $minRF$, $minCS$ and $maxOR$ are the user-specified values of minimum RF, minimum CS and maximum OR respectively.

Given a set I of items, a transactional database D and the values of $minRF$, $minCS$ and $maxOR$, the problem of mining CPs is to discover the complete set of CPs from D.

2.2.2 Coverage pattern mining algorithm

Coverage pattern mining (CPM) is an iterative multi-pass algorithm [37] for extracting CPs from a given D.

In the CPM approach, we use the notion of non-overlap pattern (NOP). A pattern X is a CP if it is a non-overlap pattern (NOP) and $CS(X) \geq minCS$.

Non-overlap pattern (NOP): A pattern $X = \{i_p, ..., i_q, i_r\}$, $(1 \leq p, q, r \leq n)$ and $|T^{i_p}| \geq ... \geq |T^{i_q}| \geq |T^{i_r}|$ is called an NOP if $OR(X) \leq maxOR$ and $RF(i_k) \geq minRF \; \forall i_k \in X$.

Notably, the overlap ratio constraint follows the sorted closure property.

Sorted closure property. Let $X = \{i_p, \ldots, i_q, i_r\}$, $(1 \leq p, q, r \leq n)$ be a pattern such that $|T^{i_p}| \geq \ldots \geq |T^{i_q}| \geq |T^{i_r}|$. If $OR(X) \leq maxOR$, all of the non-empty subsets of X containing i_r and having $size \geq 2$ will also have OR less than or equal to $maxOR$.

If an itemset does not satisfy the $maxOR$ constraint, all of its supersets (where each superset has item frequencies in the descending sorted order) do not satisfy the $maxOR$ constraint. This is stated in the following lemma.

Lemma 1 *Let $X = \{i_1, ..., i_{k-1}\}$ such that $|T^{i_1}| \geq \ldots \geq |T^{i_{k-2}}| \geq |T^{i_{k-1}}|$. Moreover, let k and p be integers such that $p > k$. Consider any set $Y = \{i_1, ..., i_{p-1}\}$ such that $|T^{i_1}| \geq \ldots \geq |T^{i_{p-2}}| \geq |T^{i_{p-1}}|$. If $OR(X) \geq maxOR$, $OR(Y) \geq maxOR$.*

Proof The proof follows from the sorted closure property of NOPs.

The description of the CPM approach is as follow. In this approach, NOPs of size k are used to compute

NOPs of size $(k+1)$. Based on Lemma 1, if an itemset does not satisfy the $maxOR$ constraint, all of its supersets (where each superset has item frequencies in the corresponding sorted order) will be pruned. Once we extract NOPs based on the $maxOR$ constraint, CPs are extracted by testing the $minCS$ constraint.

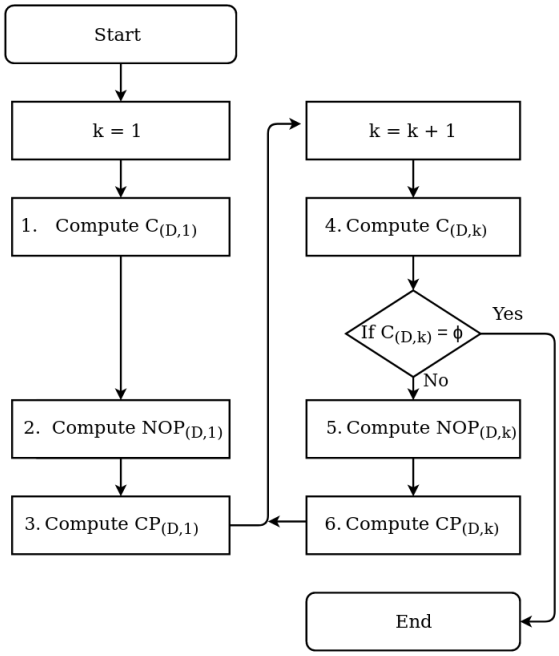


Fig. 1: CPM approach

Let $C_{(D,k)}$, $NOP_{(D,k)}$ and $CP_{(D,k)}$ be the set of candidate, non-overlap and coverage patterns of size k for D respectively. Given the values of $minCS$, $maxOR$ and $minRF$, the steps of CPM to extract CPs from D are depicted in Figure 1. The details of the steps are as follows.

(I) First iteration: Compute $CP_{(D,1)}$

1. **Compute** $C_{(D,1)}$**:** Compute the set C_1 of $< item, frequency >$ pairs by scanning D.
2. **Compute** $NOP_{(D,1)}$**:** For each pair of $< x, frequency > \in C_1$, if $RF(x) \geq minRF$, include it in NOP_1. Sort NOP_1 in the descending order of $frequency$ and copy it to $Flist$.
3. **Compute** $CP_{(D,1)}$**:** For each pair $< x, frequency > \in NOP_{(D,1)}$, if $CS(x) \geq minCS$, include it in CP_1.

(II) Second iteration and beyond: Starting from k=2, the following steps are repeated until $C_k = \phi$. For each iteration k, $CP_{(D,k)}$ is computed.

4. **Compute** $C_{(D,k)}$**:** Generate candidate list $C_{(D,k)}$ of k-items by computing $NOP_{(D,k-1)} \bowtie NOP_{(D,k-1)}$ (self-join).
5. **Compute** $NOP_{(D,k)}$**:** Scan D and compute OR and CS for each $x \in C_k$. For each $x \in C_k$, if $OR(x) \leq maxOR$, include it in $NOP_{(D,k)}$.
6. **Compute** $CP_{(D,k)}$**:** For each candidate $x \in NOP_k$, if $CS(x) \geq minCS$, include it in $CP_{(D,k)}$.

3 Proposed Framework of the Problem

This section discusses the proposed framework of the problem.

Consider a transactional database D and a set NOP of non-overlap patterns extracted based on the user-specified $minRF$, $maxOR$ and $minCS$ threshold constraints. Consider that a set Δ^- of transactions is deleted from D and a set Δ^+ of transactions is added to D. Observe that $\Delta^- \subseteq D$. Let us denote the updated database as $D^{'}$. Note that $D' = (D - \Delta^-) \cup \Delta^+$. The problem is to extract a set of NOPs of the updated database (i.e., D') using an incremental approach.

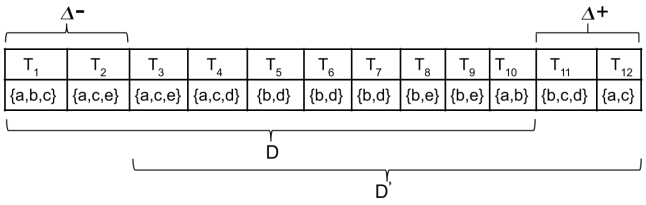


Fig. 2: Transactional database

Illustrative Example: Consider a set of transactions $\{T_1, T_2, \ldots, T_{12}\}$ depicted in Figure 2. In Figure 2, D= $\{T_1, T_2, \ldots, T_{10}\}$, Δ^+=$\{T_{11}, T_{12}\}$ and Δ^-=$\{T_1, T_2\}$. Therefore, D'= $\{T_3, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}, T_{11}, T_{12}\}$. Let the values of $minRF$, $minCS$ and $maxOR$ be 0.3, 0.8 and 0.5 respectively. The following NOPs are extracted from D: {{b}, {a}, {c}, {d}, {e}, {b,a}, {b,c}, {b,e}, {a,d}, {a,e}, {c, d}, {c,e}, {d,e}, {a,d,e}, {c,d,e}}. Furthermore, the following NOPs are extracted from D': {{b}, {d}, {a}, {c}, {e}, {b,a}, {b, c}, {d,a}, {d,c}, {d,e}, {a,e}, {c,e}, {d,a,e}, {d,c,e}}. Observe that there are total of 14 NOPs in D', and 10 of these NOPs are common with D.

4 Proposed Comprehensive Coverage Pattern Mining (CCPM) Approach

This section discusses our proposed CCPM approach and provides an illustrative example.

4.1 Overview of the proposed CCPM approach

As previously discussed in Section 2.2.2, recall that the CPM approach is an iterative algorithm for extracting the CPs of a given transactional database D. Recall that CPM exploits the sorted closure property of the overlap ratio (OR). In CPM, the non-overlap patterns (NOPs) in each iteration are computed and then the CPs are generated based on the coverage support (CS) of NOPs.

Given D and the NOPs of D, we shall now present the Comprehensive Coverage Pattern Mining (CCPM) approach for extracting NOPs (CPs) when a set Δ^+ of transactions is added to D and/or a set Δ^- of transactions is deleted from D. Observe that when D is changed, some of the existing NOPs of D may become *invalid* and some new NOPs may emerge. Notably, CCPM follows a level-wise approach.

The main steps of the proposed CCPM approach are depicted in Figure 3. Similar to the steps of the CPM approach (see Figure 1), CCPM comprises six steps. During the first iteration, Step 1 is modified as follows in case of CCPM. In this step, to compute the frequencies of the items for D', instead of scanning the entire D', we scan only Δ^+ and Δ^- and compute the counts by utilizing the counts of items for D. There might be additional potential items that may emerge. From these potential items, after pruning the non-potential items by applying pruning rules, $D - \Delta^-$ is scanned to compute the frequencies of those items. Steps 2 and 3 are similar to the corresponding steps in CPM.

From the second iteration onwards, the details of the steps for CCPM are as follows. In Step 4, we compute the OR and CS values for all potential itemsets of size k. In this step, instead of scanning the entire D', we scan only Δ^+ and Δ^- and compute the respective OR and CS values by utilizing the OR and CS values of the itemsets for D. Observe that additional potential items may emerge. From these potential itemsets, after pruning the non-potential itemsets by applying pruning rules, $D - \Delta^-$ is scanned to compute the OR and CS values of those itemsets. Steps 5 and 6 are similar to the corresponding steps of CPM.

In addition to exploiting the OR and CS values of the itemsets in D for computing the final OR and CS values, the important aspect of the proposed CCPM approach is about *reducing the number of additional patterns to be examined*. For such additional patterns, we need to compute the OR and CS values by scanning $D - \Delta^-$. This is done by defining a set of pruning rules, which facilitate the pruning of the non-potential itemsets. We present the details of CCPM after presenting the details of the respective approach for (a) considering the case for the addition of transactions (Δ^+), which we designate as **Incremental Coverage Pattern Mining (ICPM)** and (b) considering the case for the deletion of transactions (Δ^-), which we shall refer to as **Decremental Coverage Pattern Mining (DCPM)**.

Notably, in case of CPM, the candidate NOPs at any given level are generated based on the $Flist$ of D'. If the $Flist$ of D' is the same as that of the $Flist$ of D, only a limited number of patterns need to be validated after subtracting the given NOPs of D. However, if the $Flist$ of D' is different from that of the $Flist$ of D, the number of additional patterns to be validated by scanning $D - \Delta^-$ depends on the degree of change. Hence, in case of CCPM, the number of patterns to be validated depends on the extent of change between the $Flist$ of D and the $Flist$ of $(D + \Delta^+ - \Delta^-)$. The percentage of additional patterns generated due to the change in $Flist$ is designated as Ψ.

In the next three subsections, we present the pruning rules for ICPM, DCPM and CCPM. Next, we explain about the significance and ramifications of Ψ. Finally, we present the proposed CCPM algorithm, which exploits the afore-mentioned pruning rules.

4.2 Case of addition of transactions

Now we shall present the Incremental Coverage Pattern Mining (ICPM) approach by considering the case of addition of transactions. Given a transactional database D, a set of NOPs extracted from D under the user-specified threshold constraints of $minRF$, $maxOR$ and $minCS$, and a set Δ^+ of transactions, the issue is to extract NOPs from $D' = D + \Delta^+$ through an incremental approach.

In addition to Lemma 1, we explain the properties, which we have identified to develop the ICPM algorithm.

We can identify the potential NOPs in D' by measuring the corresponding RF values in Δ^+ as the RF values of the NOPs in D are available as input, as per the following lemma.

Lemma 2 *A 1-itemset X, which is an NOP in D, is an NOP in D', if $RF(X) \geq minRF$ in D'.*

Proof Here, $D' = D + \Delta^+$. According to the definitions of RF and NOP, RF in D' is equal to RF in $(D + \Delta^+)$.

If any given item does not satisfy the $minRF$ threshold in D, it can be pruned if it does not satisfy the $minRF$ threshold in Δ^+, as per the following lemma.

Lemma 3 *A 1-itemset X, which is not an NOP in D, is an NOP in D', iff it is an NOP in Δ^+, i.e., $RF(X_{\Delta^+}) \geq minRF$.*

Proof Let $F(X_D)$ represent the frequency of itemset X in D. Since X is not an NOP in D, $F(X_D) < minRF * |D|$. If $F(X_{\Delta^+}) < minRF * |\Delta^+|$, $F(X_{D'}) < minRF * |D'|$ since $F(X_{D'}) = F(X_D) + F(X_{\Delta^+})$. Hence, proved.

In addition to the NOPs in D, if there is a new NOP in D', it should exist in Δ^+. The following lemma states this rule.

Lemma 4 *A k-itemset $X = \{i_1, ...i_{k-1}, i_k\}$, which is not an NOP in D and not an NOP in Δ^+, will not be an NOP in D' (i.e., $D + \Delta^+$).*

Proof Let $n(OR(X_D))$ and $d(OR(X_D))$ represent the numerator and the denominator of the overlap ratio of a given pattern X in a transactional database D, where $n(OR(X_D)) =$
$|CSet(X - i_r) \cap T^{w_r}|$ and $d(OR(X_D)) = |CSet(i_r)|$. It is given that the pattern is neither an NOP D nor in Δ^+. Hence, we have the following equations:

$$OR(X_D) = \frac{n(OR(X_D))}{d(OR(X_D))} > maxOR \tag{1}$$

$$OR(X_{\Delta^+}) = \frac{n(OR(X_{\Delta^+}))}{d(OR(X_{\Delta^+}))} > maxOR. \tag{2}$$

From Equations 1 and 2, we obtain:

$$\frac{n(OR(X_D)) + n(OR(X_{\Delta^+})}{d(OR(X_D)) + d(OR(X_{\Delta^+}))} > maxOR \tag{3}$$

Equation 3 indicates that the value of OR of X in $D + \Delta^+$ is greater than the value of $maxOR$. Thus, if a given pattern is not an NOP in D and Δ^+, the pattern is not an NOP in $D + \Delta^+$.

Lemma 4 facilitates in pruning the non-potential new NOPs by processing only Δ^+. Hence, by extracting the NOPs from Δ^+, it is ensured that all of the new NOPs in D' are extracted.

Based on the preceding lemmas, the framework of the ICPM approach is as follows. Given $minRF$, $minCS$ and $maxOR$, suppose $NOP_{D'}$ is the complete set of NOPs that are extracted from D'. For each pattern $P \in NOP_{D'}$, the following four possibilities exist w.r.t. its validity in D and Δ^+:

1. $P \in NOP_D$ and $P \in NOP_{\Delta^+}$
2. $P \in NOP_D$ and $P \notin NOP_{\Delta^+}$
3. $P \notin NOP_D$ and $P \in NOP_{\Delta^+}$
4. $P \notin NOP_D$ and $P \notin NOP_{\Delta^+}$

To cover Case 1, we extract NOPs in Δ^+ and add the counts of these NOPs to the corresponding NOPs in D. To cover Case 2, only the patterns in D are validated by adding the counts of the corresponding NOPs in Δ^+. To cover Case 3, we extract additional patterns from Δ^+ and validate these patterns in D. To cover Case 4, no action needs to be taken since such a possibility does not exist (see Lemma 4).

4.3 Case of deletion of transactions

In this section, we present the Decremental Coverage Pattern Mining (DCPM) approach by considering the case of deletion of transactions. Given a transactional database D, a set of NOPs extracted from D at the user-specified threshold constraints of $minRF$, $maxOR$, and $minCS$, and a set Δ^- of transactions, we need to extract NOPs from $D' = D - \Delta^-$ through an incremental approach.

In addition to Lemma 1, we explain the properties, which we have identified to develop the DCPM algorithm.

We can identify potential NOPs in D' by measuring the corresponding RF values in Δ^- as the RF values of the NOPs in D are available as input, as per the following lemma.

Lemma 5 *A 1-itemset X, which is an NOP in D is an NOP in D', if $RF(X) \geq minRF$ in D'.*

Proof Here, $D' = D - \Delta^-$. According to the definition of RF and NOP, RF in D' is equal to RF in $(D - \Delta^-)$.

If a given item does not satisfy the $minRF$ threshold in D, it can be pruned if it does satisfy the $minRF$ threshold in Δ^-, as per the following lemma.

Lemma 6 *A 1-itemset X, which is not an NOP in D, is not an NOP in D', if it is an NOP in Δ^-, i.e., $RF(X_{\Delta^-}) \geq minRF$.*

Proof Let $F(X_D)$ represent the frequency of a given itemset X in D. Since X is not an NOP in D, $F(X_D) < minRF * |D|$. If $F(X_{\Delta^-}) \geq minRF * |\Delta^-|$, $F(X_{D'}) < minRF * |D^{'}|$ since $F(X_{D'}) = F(X_D) - F(X_{\Delta^-})$. Hence, proved.

In addition to the NOPs in D, if there is a new NOP in D', it should not exist in Δ^-. Hence, by extracting the NOPs from Δ^-, we can prune the non-potential patterns. The following lemma states the rule.

Lemma 7 *A k-itemset $X = \{i_1, ...i_{k-1}, i_k\}$, which is not an NOP in D and an NOP in Δ^-, will not be an NOP in D', i.e., $D - \Delta^-$.*

Proof Let $n(OR(X_D))$ and $d(OR(X_D))$ represent the numerator and the denominator of the overlap ratio of X in D, where $n(OR(X_D)) = |CSet(X - i_r) \cap T^{w_r}|$ and $d(OR(X_D)) = |CSet(i_r)|$. It is given that the pattern is not an NOP in D and an NOP in Δ^-. Hence, we have the following equations:

$$OR(X_D) = \frac{n(OR(X_D))}{d(OR(X_D))} > maxOR \tag{4}$$

$$OR(X_{\Delta^-}) = \frac{n(OR(X_{\Delta^-}))}{d(OR(X_{\Delta^-}))} \leq maxOR. \tag{5}$$

From Equations 4 and 5, we obtain:

$$\frac{n(OR(X_D)) - n(OR(X_{\Delta^-})}{d(OR(X_D)) - d(OR(X_{\Delta^-}))} > maxOR \tag{6}$$

Equation 6 indicates that the OR of X in D' is greater than $maxOR$. Thus, if a pattern is not an NOP in D and an NOP in Δ^-, the pattern is not an NOP in D'.

Based on the preceding lemmas, the framework of the DCPM approach is as follows. Given $minRF$, $minCS$, and $maxOR$, suppose $NOP_{D'}$ is the complete set of NOPs extracted from D'. For each pattern $P \in NOP_{D'}$, the following four possibilities exist w.r.t. its validity in D and Δ^-:

1. $P \in NOP_D$ and $P \in NOP_{\Delta^-}$
2. $P \in NOP_D$ and $P \notin NOP_{\Delta^-}$
3. $P \notin NOP_D$ and $P \in NOP_{\Delta^-}$
4. $P \notin NOP_D$ and $P \notin NOP_{\Delta^-}$

To cover Case 1, we extract NOPs in Δ^- and subtract the counts of these NOPs in the corresponding NOPs in D. To cover Case 2, only the patterns in D are validated by subtracting the counts of the corresponding NOPs in Δ^-. To cover Case 3, no action needs to be taken since such a possibility does not exist (refer to Lemma 7). To cover Case 4, we extract the additional non-potential patterns from Δ^- and validate in $D - \Delta^-$.

4.4 Case of both addition and deletion of transactions

Now we shall present the Comprehensive Coverage Pattern Mining (CCPM) approach by considering the case of both addition and deletion of transactions. Given a transactional database D, a set of NOPs extracted from D at the user-specified thresholds constraints of $minRF$, $maxOR$, and $minCS$, a set Δ^+ of transactions and a set Δ^- of transactions, we need to extract NOPs from $((D + \Delta^+) - \Delta^-)$.

In addition to Lemma 1 we explain the properties, which we have identified to develop the proposed Comprehensive Coverage Pattern Mining (CCPM) algorithm.

We can identify potential NOPs in D' by measuring the corresponding RF values in Δ^+ and Δ^- as RF values of the NOPs in D are available as input, as per the following lemma.

Lemma 8 *A 1-itemset X, which is an NOP in D, is an NOP in D' $(D + \Delta^+ - \Delta^-)$, if $RF(X) \geq minRF$ in D'.*

Proof Here, $D' = D + \Delta^+ - \Delta^-$. According to the definition of RF and NOP, RF in D' is equal to RF in $(D + \Delta^- \Delta^-)$.

If a given item does not satisfy the $minRF$ threshold in D, it can be pruned if it *does not satisfy* the $minRF$ threshold in Δ^+ and *satisfies* the $minRF$ threshold in Δ^-, as per the following lemma.

Lemma 9 *A 1-itemset X, which is not an NOP in D, is not an NOP in D', if it is an NOP in Δ^- and not an NOP in Δ^+, i.e., $RF(X_{\Delta^-}) \geq minRF$ and $RF(X_{\Delta^+}) < minRF$.*

Proof Let $F(X_D)$ represent the frequency of a given itemset X in D. Since X is not an NOP in D, $F(X_D) \leq minRF*|D|$. If $F(X_{\Delta^+}) < minRF*|\Delta^+|$ and $F(X_{\Delta^-}) \geq minRF*|\Delta^-|$, $F(X_{D'}) \leq minRF*|D^{'}|$ since $F(X_{D'}) = F(X_D) + F(X_{\Delta^+}) - F(X_{\Delta^-})$.

In addition to the NOPs in D, if there is a new NOP in D', it should exist in Δ^+ and it should not exist in Δ^-. Hence, by extracting the NOPs from Δ^+ and Δ^-, we can prune the non-potential patterns. The following lemma states the rule.

Lemma 10 *A k-itemset $X = \{i_1, ...i_{k-1}, i_k\}$, which is not an NOP in D and an NOP in Δ^- and not an NOP in Δ^+ , will not be an NOP in D', i.e., $D - \Delta^- + \Delta^+$.*

Proof Let $n(OR(X_D))$ and $d(OR(X_D))$ represent the numerator and the denominator of the overlap ratio of a given pattern X in D, where $n(OR(X_D)) = |CSet(X - i_r) \cap T^{w_r}|$ and $d(OR(X_D)) = |CSet(i_r)|$. It is given that the pattern is not an NOP in D and not an NOP in Δ^+ and an NOP in Δ^- . Hence, we have the following equations:

$$OR(X_D) = \frac{n(OR(X_D))}{d(OR(X_D))} > maxOR \tag{7}$$

$$OR(X_{\Delta^-}) = \frac{n(OR(X_{\Delta^-}))}{d(OR(X_{\Delta^-}))} \leq maxOR. \tag{8}$$

$$OR(X_{\Delta^+}) = \frac{n(OR(X_{\Delta^+}))}{d(OR(X_{\Delta^+}))} > maxOR. \quad (9)$$

From Equations 7, 8 and 9 we obtain:

$$\frac{n(OR(X_D)) - n(OR(X_{\Delta^-})) + n(OR(X_{\Delta^+}))}{d(OR(X_D)) - d(OR(X_{\Delta^-})) + d(OR(X_{\Delta^+}))} > maxOR \quad (10)$$

Equation 10 indicates that OR of X in D' is greater than $maxOR$. Thus, if a pattern is not an NOP in D, not an NOP in Δ^+ and an NOP in Δ^-, the pattern is not an NOP in D'.

The framework of the CCPM approach is as follows. Given $minRF$, $minCS$ and $maxOR$, suppose $NOP_{D'}$ is the complete set of NOPs extracted from D'. For each pattern $P \in NOP_{D'}$, the following eight possibilities exist w.r.t. its validity in D, Δ^+ and Δ^-:

1. $P \in NOP_D$ and $P \in NOP_{\Delta^+}$ and $P \in NOP_{\Delta^-}$
2. $P \in NOP_D$ and $P \in NOP_{\Delta^+}$ and $P \notin NOP_{\Delta^-}$
3. $P \in NOP_D$ and $P \notin NOP_{\Delta^+}$ and $P \in NOP_{\Delta^-}$
4. $P \in NOP_D$ and $P \notin NOP_{\Delta^+}$ and $P \notin NOP_{\Delta^-}$
5. $P \notin NOP_D$ and $P \in NOP_{\Delta^+}$ and $P \in NOP_{\Delta^-}$
6. $P \notin NOP_D$ and $P \in NOP_{\Delta^+}$ and $P \notin NOP_{\Delta^-}$
7. $P \notin NOP_D$ and $P \notin NOP_{\Delta^+}$ and $P \in NOP_{\Delta^-}$
8. $P \notin NOP_D$ and $P \notin NOP_{\Delta^+}$ and $P \notin NOP_{\Delta^-}$

To cover Cases 1-4, only the patterns in D are validated by adding and subtracting the counts of corresponding NOPs in Δ^+ and Δ^- respectively. To cover Case 7, no action needs to be taken since such a possibility does not exist (see Lemma 10). To cover Cases 5, 6 and 8, we extract the additional patterns from Δ^+ and Δ^- and validate in $D - \Delta^-$.

4.5 Impact of changes in *Flist*

In the proposed approach, we need to validate the additional patterns that are generated due to the addition of Δ^+ and the deletion of Δ^-. These are the additional new patterns that are generated (i) due to the change in the database and (ii) the change in the sorting order of the items in *Flist*.

Notably, if there is no change in the *Flist* in D and D', the same patterns, which are in D, will be validated in D'. However, if there is a change in the *Flist*, the additional patterns are to be validated and the number of additional patterns depends on the degree of change in the *Flist* order. We designate the set of patterns generated due to the change in the *Flist* of D to the corresponding D' with APF. The $Flist_{D'}$ is extracted by counting item frequencies of D' and forming the $Flist_{D'}$ in the sorted order. If $Flist_{D'}$ differs from $Flist_D$, the new patterns are generated.

Now we shall present an approach to estimate the value of the upper-bound of the number $APFs$ of *additional patterns due to change in Flist*. We compute the value of Ψ by normalization i.e., $\frac{|APF|}{max(|APF|)} * 100$, where $max(|APF|)$ is the maximum value of $|APF|$. The range of Ψ lies between 0 and 100. We shall shortly see how to determine the value of $max(|APF|)$.

Now we shall define the notion of *prefix set*, which is used in the computation of $|APF|$. **Prefix set**: The prefix set $P_D(i_p)$ of a given item i_p in database D is the set of items, whose frequency is greater than that of its own frequency i.e., the set of items that comes before i_p in the *Flist*. For computing the value of $|APF|$, our inputs are the *Flist* of D and the *Flist* of D'. Let $Flist_D$ (i.e., the *Flist* of D) and $Flist_{D'}$ (i.e., the *Flist* of D') be $\{i'_1, i'_2, .., i'_m\}$ and $\{i_1, i_2, ..., i_n\}$ respectively. Consider a pattern X in which i_p is the least frequent item.

We define $commset(i_p)$ as the set of items, which are present in both $P_{D'}(i_p)$ and $P_D(i_p)$ i.e., $P_{D'}(i_p) \cap P_D(i_p)$. The patterns having items in *commset* along with i_p are not additional patterns as those have already been checked in extracting the NOPs of D. The number of additional patterns having i_p as the least frequent item is $(2^{|P_{D'}(i_p)|} - 2^{|commset|})$. The value of $|APF|$ is more influenced by the least frequent item as the size of the prefix set of the least frequent item is high. The value of $|APF|$ is zero when both the *Flists* of D and D' are the same. The estimated value of $|APF|$ is computed using Equation 11 as follows:

$$|APF| = \Big(\sum_{i_p \in Flist_{D'}} 2^{|P_{D'}(i_p)|} - 2^{|commset(i_p)|} \Big) \quad (11)$$

The value of $|APF|$ is maximum when the *Flists* of D and D' are disjoint sets. The value of $max(|APF|)$ is equal to $2^N - N - 1 \approx 2^N$, where N is the size of $Flist_{D'}$. The value of Ψ is computed using Equation 12 as follows:

$$\Psi = \frac{|APF|}{max(|APF|)} \approx \Big(\frac{|APF|}{2^N} \Big) * 100 \quad (12)$$

4.6 CCPM algorithm

Similar to the CPM algorithm, our proposed CCPM algorithm is also an iterative algorithm. At the k^{th} iteration of the CCPM algorithm, we compute the NOPs of size k by performing a single scan of D, Δ^+ and Δ^-.

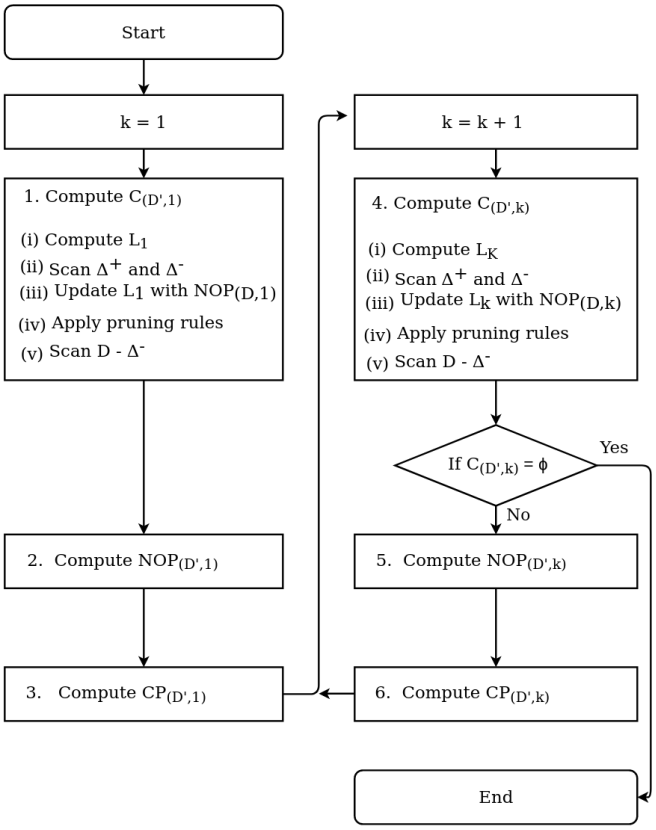


Fig. 3: CCPM approach

The inputs to the algorithm are as follows: (i) transactional dataset D (ii) set of transactions Δ^+ and Δ^- over a set I of items (iii) the values of $minRF$, $minCS$ and $maxOR$ (iv) $NOP_{(D,1)}$ of tuples $< x, v(x) >$ for all single items of D ($v(x)$ denotes the frequency of item x) and $NOP_{(D,k)}$ of triples $< x, NOR(x), DOR(x), NCS(x) >$ for all k-itemsets ($k \geq 2$), where x denotes any itemset of size k and $NOR(x)$ represents the numerator value of $OR(x)$, $DOR(x)$ represents the denominator value of $OR(x)$ and $NCS(x)$ denotes the numerator value of the coverage support of x.

For the sake of convenience, we shall use the following notations. We use D' to denote $D+\Delta^+-\Delta^-$. We use $NOP_{(D,k)}$ and $NOP_{(D',k)}$ to denote the set of NOPs of size k in D and D' respectively. The notations $CP_{(D,k)}$ and $CP_{(D',k)}$ are used to denote the set of CPs of size k in D and D' respectively. Moreover, we employ the notations L_k and R_k to denote a set of patterns of size k. The notation $v_i(x)$, i=1 to 12, denotes the variable, which holds a real value.

The steps of the CCPM approach are depicted in Figure 3. The details of these steps are as follows.

(I) First Iteration: Compute $CP_{(D',1)}$

1. **Compute $C_{(D',1)}$**
   - 1.1 **Compute L_1:** Initialize the set L_1 of $< x, v(x) = 0 >$ pairs for set I of all items. Here, x denotes the item and $v(x)$ denotes the frequency value of x.
   - 1.2 **Scan Δ^+ and Δ^-:** By scanning Δ^+, compute $v(x)$ for each $x \in I$ and store $< x, v(x) >$in L_{Δ^+}. Moreover, by scanning Δ^-, compute $v(x)$ for each $x \in I$ and store $< x, v(x) >$ in L_{Δ^-}. Replace each $< x, v(x) > \in L_1$ with $< x, v1(x) - v2(x) >$. Here, $v1(x)$ is the frequency of x in L_{Δ}^{+} and $v2(x)$ is the frequency of x in L_{Δ}^{-}.
   - 1.3 **Update L_1 with $NOP_{(D,1)}$:** For each $< x, v1(x) > \in NOP_{(D,1)}$, replace $< x, v2(x) >$ of L_1 with $< x, v1(x) + v2(x) >$.
   - 1.4 **Apply Pruning Rules:** Compute $R_1 = L_1 - NOP_{(D,1)}$. Remove $< x, v(x) > \in R_1$, if both conditions (i) and (ii) are satisfied: (i) $< x, v1(x) > \in L_{\Delta^+}$ and $v1(x) < minRF * |\Delta^+|$ (ii) $< x, v1(x) > \in L_{\Delta^-}$ and $v1(x) \geq minRF * |\Delta^-|$.
   - 1.5 **Scan $D - \Delta^-$:** Scan $D - \Delta^-$ and update the frequency values for all $< x, v(x) >$ pairs in R_1. Compute $C_{(D',1)}$ by merging R_1 and L_1.
2. **Compute NOPs:** For each $< item, frequency > \in C_{(D',1)}$, if $RF(x) \geq minRF$, include it in $NOP_{(D',1)}$. Sort $NOP_{(D',1)}$ in descending order and form $Flist(D')$.
3. **Compute CPs:** For each candidate 1-item $< x, v(x) > \in NOP_{(D',1)}$, if $CS(x)(= RF(x)) \geq minCS$, include it in $CP_{(D',1)}$.

(II) Second iteration and beyond: Generation of $NOP_{(D',k)}$ and $CP_{(D',k)}$: Starting from $k = 2$, the following steps (i.e., Steps 4 to 6) are repeated until $NOP_{(D',k)} = \phi$. For each iteration k, the input is $NOP_{(D,k)}$ and the output is $NOP_{(D',k)}$ and $CP_{(D',k)}$.

4. **Compute $C_{(D',k)}$**
   - 4.1 **Compute L_k (generate candidate k-itemsets):** Generate candidate k-itemsets L_k by computing $NOP_{(D',k-1)} \bowtie NOP_{(D',k-1)}$ (self-join). Initialize each element x of L_k with $< x, v1(x) = 0, v2(x) = 0, v3(x) = 0 >$.
   - 4.2 **Scan Δ^+ and Δ^-** : By scanning Δ^+, compute the values of NOR(x), DOR(x) and NCS(x) for each $x \in L_k$ and store $< x, NOR(x), DOR(x), NCS(x) >$ in L_{Δ^+}. Moreover, by scanning Δ^-, compute the values of NOR(x), DOR(x), and NCS(x) for each $x \in L_k$ and store $< x, NOR(x), DOR(x), NCS(x) >$ in L_{Δ^-}.
   - 4.3 **Update L_k with $NOP_{(D,k)}$:** For each x such that $< x, v1(x), v2(x), v3(x) \in NOP_{(D,k)}$, $< x, v4(x), v5(x), v6(x) > \in L_{\Delta^+}$, and $< x, v7(x), v8(x), v9(x) > \in L_{\Delta^-}$, replace $< x, v10(x), v11(x), v12(x)$

Table 1: NOP_D

{b}	{a,d}
{a}	{a,e}
{c}	{c,d}
{d}	{c,e}
{e}	{d,e}
{b,a}	{a,d,e}
{b,c}	{c,d,e}
{b,e}	

Table 2: $NOP_{D'}$

{a}	{d,a}
{c}	{d,c}
{b}	{d,e}
{d}	{a,e}
{e}	{c,e}
{b,a}	{d,a,e}
{b,c}	{d,c,e}

Table 3: $CP_{D'}$

{b,a}
{b,c}
{d,a}
{d,e}
{d,a,e}
{d,c,e}

$>$ in L_k with $< x, v10(x) = v1(x) + v4(x) - v7(x), v11(x) = v2(x) + v5(x) - v8(x), v12(x) = v3(x) + v6(x) - v9(x) >$.

4.4 **Apply Pruning Rules:** Compute $R_k = L_{(k)} - NOP_{(D,k)}$. Remove $< x, v1(x), v2(x), v3(x) > \in R_k$, if all the following three conditions are satisfied. (i) $< x, v1(x), v2(x), v3(x) > \in L_{\Delta^+}$ and $v1(x)/v2(x) > maxOR$ (ii) $< x, v4(x), v5(x), v6(x) > \in L_{\Delta^-}$ and $v4(x)/v5(x) \leq maxOR$ (iii) the items in x follows the order of items in $Flist(D)$.

4.5 **Scan** $D - \Delta^-$**:** Scan $D - \Delta^-$ and update $< x, NOR(x), DOR(x), NCS(x) >$ for all $x \in R_k$. Compute $C_{(D',k)}$ by merging R_k and L_k.

5. **Compute** $NOP_{(D',K)}$
Form $C_{(D',k)}$ by merging L_k and R_k. For each $< x, v1(x), v2(x), v3(x) > \in C_{(D',k)}$, if $OR(x)$ $(= v1(x)/v2(x)) \leq maxOR$, include it in $NOP_{(D',k)}$.
6. **Compute CP**$_{(D',k)}$
For each candidate k-itemset $< x, v1, v2, v3 > \in NOP_{(D',k)}$, if $CS(x)(= v3/|D'|) \geq minCS$, include it in $CP_{(D',k)}$.

Impact of Ψ: As explained in Section 4.5, we note that that the number of additional *NOPs* generated depends on the change in the $Flist(D')$ as compared to $Flist(D)$. We compute Ψ. Let Ψ_{max} be a user-specified threshold. If $\Psi < \Psi_{max}$, we follow the CCPM approach. Otherwise, the CPM algorithm can be applied from scratch.

4.7 Illustrative Example of CCPM

We explain the working of our proposed CCPM algorithm by means of an illustrative example. Figure 1 depicts the original database D ($\{T_1, T_2, \ldots, T_{10}\}$), transactions added Δ^+ ($\{T_{11}, T_{12}\}$) and transactions deleted Δ^- ($\{T_1, T_2\}$). Let the values of $minRF$, $minCS$ and $maxOR$ used for the extraction of the coverage patterns be 0.3, 0.8 and 0.5 respectively. The input to the algorithm also consists of iteration-wise NOPs of D, as shown in Table 2.

In the first iteration, $NOP_{(D',1)}$ and $CP_{(D',1)}$ are extracted. The items of size one, having frequency greater than ($minRF$ * size of the given database), are called NOPs. The NOPs of size one in D along with their frequencies are {a:6, b:7, c:5, d:5, e:4}. First, we initialize set L_1 as {a:0, b:0, c:0, d:0, e:0}. Then we compute the frequencies of items $x \in L_1$ by scanning Δ^- and Δ^+ and store them in L_{Δ^+} and L_{Δ^-} respectively. The set L_{Δ^+} becomes {a:1, b:1, c:2, d:1, e:0}, set L_{Δ^-} becomes {a:2, b:1, c:2, d:0, e:1} and L_1 becomes {a:-1, b:0, c:0, d:1, e:-1}. For each $x \in NOP_{(D,1)}$, we update the frequencies in the set L_1. Hence, L_1 becomes {a:5, b:7, c:5, d:6, e:3}. Then we compute $R_1 = L_1 - NOP_{(D,1)}$. As R_1 is empty, we need not scan $D - \Delta^-$. Finally, $NOP_{(D',1)}$ = {b:7, d:6, a:5, c:5, e:3} and $CP_{(D',1)}$ = {}. The set of CPs is empty as none of the items has frequency greater than 8 ($minCS * |D'|$). The $Flists$ of D and D' are {b,a,c,d,e} and {b,d,a,c,e} respectively.

The number of additional patterns is computed for any given item as follows. As an example, for item a, the number of additional patterns $|APF|$ is $2^{|\{b,d\}|} - 2^{|\{b\}|}$ = 2. For item c, $|APF|$ is $2^{|\{b,d,a\}|} - 2^{|\{b,a\}|} = 4$. For the remaining items, $|APF|$ is 0. In this example, Ψ is 18.75.

We generate candidate k-itemsets L_k by computing $NOP_{(D',k-1)} \bowtie NOP_{(D',k-1)}$. L_k is equal to {{b,d}, {b,a}, {b,c}, {b,e}, {d,a}, {d,c}, {d,e} {a,c} {a,e}, {c,e}}. We initialize the overlap ratio (numerator and denominator), coverage support (numerator) of pattern $x \in L_k$ as 0. We compute the overlap ratio (numerator and denominator), coverage support (numerator) for each $x \in L_k$ in Δ^- and Δ^+ by scanning Δ^- and Δ^+ and store them in L_{Δ^+} and L_{Δ^-} respectively. L_{Δ^+} becomes {{b,d : [1,1,1]}, {b,a : [0,1,2]}, {b,c : [1,2,2]}, {b,e : [0,0,1]}, {d,a : [0,1,2]}, {d,c : [1,2,2]}, {d,e : [0,0,1]} {a,c : [1,2,2]} {a,e : [0,0,1]}, {c,e : [0,0,2]}} and L_{Δ^-} becomes {{b,d : [0,0,1]}, {b,a : [1,2,2]}, {b,c : [1,2,2]}, {b,e : [0,1,2]}, {d,a : [0,2,2]}, {d,c : [0,2,2]}, {d,e : [0,1,1]} {a,c : [2,2,2]} {a,e : [1,1,2]}, {c,e : [1,1,2]}}. The NOPs of size 2 of D are {{b,a : [2,5,10]}, {b,c : [1,4,10]}, {b,e : [2,4,9]}, {a,d : [1,4,7]}, {a,e : [2,4,7]}, {c,d : [1,4,7]}, {c,e : [2,4,6]} and {d,e : [0,4,8]}}. For each $x \in NOP_{(D,k)}$ and $x \in L_k$, we update the overlap ratio and coverage support by adding the respective numerator, denominator of overlap ratio and numerator of coverage support from L_{Δ^+} and adding the respective numerator, denominator of overlap ratio and numerator of coverage support from L_{Δ^-}. After updating L_k becomes {{b,d : [0,0,0]}, {b,a : [1,4,10]}, {b,c : [1,4,10]}, {b,e : [2,3,8]}, {d,a : [0,0,0]}, {d,c : [0,0,0]}, {d,e : [0,3,8]} {a,c : [0,0,0]} {a,e : [1,3,6]}, {c,e : [1,3,6]}}.

Then we compute $R_k = L_{(k)} - NOP_{(D,k)}$. R_k becomes {{b,d}, {d,a}, {d,c}, {a,c}}. Then we prune pat-

terns from the set R_k if overlap ratio of x is greater than $maxOR$ in Δ^+ and is less than or equal to $maxOR$ in Δ^- and the pattern follows the order of item in $Flist(D)$. Two patterns {{b,d}, {a,c}} in R_k follow the order of item in $Flist(D)$ so check $maxOR$ of those patterns in Δ^+ and Δ^- to prune those patterns. Overlap ratio [numerator, denominator] of {b,d} in Δ^+ is [1,1] and in Δ^- is [0,0] and overlap ratio [numerator, denominator] of {a,c} in Δ^+ is [1,2] and in Δ^- is [2,2]. We can prune pattern {b,d} from R_k as it follows the pruning condition. For remaining patterns in R_k scan $D - \Delta^-$ and update the overlap ratio and coverage support. R_k is updated as {{d,a : [1,4,8]}, {d,c : [2,4,7]}, {a,c : [3,4,5]}}. We can compute $C_{(D',k)}$ by merging R_k and L_k. $C_{(D',k)}$ becomes {{b,a : [1,4,9]}, {b,c : [0,4,10]}, {b,e : [2,3,7]}, {d,a : [1,4,8]}, {d,c : [2,4,7]}, {d,e : [0,3,8]} {a,c : [3,4,5]} {a,e : [1,3,6]}, {c,e : [1,3,6]}}.

Now by checking the conditions of overlap ratio we can include the pattern in $NOP_{(D',k)}$. For $x \in NOP_{(D',k)}$ if the coverage support condition satisfies we include it in $CP_{(D',k)}$. We follow a similar procedure for iteration 3 as well. The final sets of NOPs and CPs are shown in Table 2 and Table 3 respectively.

5 Performance Evaluation

This section reports our performance evaluation. We have conducted experiments by implementing our proposed CCPM algorithm as well as the reference CPM algorithm in *Python 3*. Our experiments were conducted using a computer with a fifth-generation Intel Core-i5 2.7 GHz processor and 8GB RAM.

The experiments were conducted on three datasets. The first dataset is **BMS-POS** dataset, which is a click-stream dataset [19] of an e-commerce company; the dataset has 515,596 transactions and 1656 distinct items. The second dataset is **CABS120k08** dataset, which is a click-stream dataset [34]. It is a collection of search queries and documents clicked and user identifiers. The transactions from this dataset are extracted using the 30-minutes rule, i.e., the collection of search queries and documents clicked during the session of 30-minutes duration are considered as one transaction [13]. The CABS120k08 dataset has 402,776 transactions and 572 distinct items. The third dataset is the **T10I4D100K**, which is a synthetic dataset [5] generated by a dataset generator. This synthetic dataset had 100,000 transactions and 871 distinct items. We shall henceforth refer to this dataset as **Synthetic** dataset.

We have also generated another synthetic dataset, designated as **Synthetic-Modified**, to study the impact of change in the *Flist* on the proposed approach. First, we computed the frequency for all the items in T10I4D100K dataset and sorted in the descending order of frequency. Next, for each transaction in T10I4D100K, we have formed the corresponding transaction in Synthetic-Modified by swapping the most frequent item with the least frequent item and vice versa, the second-most frequent item with the second-least frequent item and vice versa, and so on.

The performance metric is execution time ET. ET represents the processing time required for extracting the coverage patterns from the given transactional dataset from the disk. We measure the value of ET in seconds.

As reference, for the purpose of meaningful comparison, we adapt the existing CPM approach [37], which we have previously discussed in Section 2.2.2. Recall that CPM is a level-wise pruning approach, which does not support an incremental approach. Hence, under the CPM approach, whenever a set of transactions are added to or deleted from the transactional database, the CPM algorithm is executed to extract coverage patterns *from scratch*, subject to the constraints of $minRF$, $maxOR$ and $minCS$.

Table 4 summarizes the parameters of our performance evaluation.

5.1 Effect of varying $minRF$, $maxOR$ and $minCS$ values

Figure 4 depicts the effect of variations in $minRF$, $maxOR$ and $minCS$ for three datasets at the corresponding default values of Δ^+.

Figure 4(a) shows the effect of variations of the $minRF$ threshold on ET for three datasets. It can be observed that when the value of $minRF$ is low, ET of CPM in case of BMS-POS is high as an increased number of items will satisfy the $minRF$ threshold. As $minRF$ increases, the number of items, which are involved in the formation of patterns, is reduced. As a result, ET is reduced as the number of patterns is reduced. The results show that the ET performance of CCPM is improved significantly as compared to CPM. This is due to less computation carried out to extract the same number of patterns in CCPM as compared to the amount of computation in CPM. It can be observed that the performance trends of CCPM and CPM are similar for the other two datasets.

Figure 4(b) shows the effect of variations of the $maxOR$ threshold on ET for three datasets. It can be observed that when the value of $maxOR$ is low, ET of CPM on BMS-POS is low as less number of patterns will be computed. As $maxOR$ increases, as expected, ET increases due to increase in the number of patterns to be computed. It can be observed that the ET

Table 4: Parameters of our performance evaluation

*DF - Default Value, *SS - Step-size

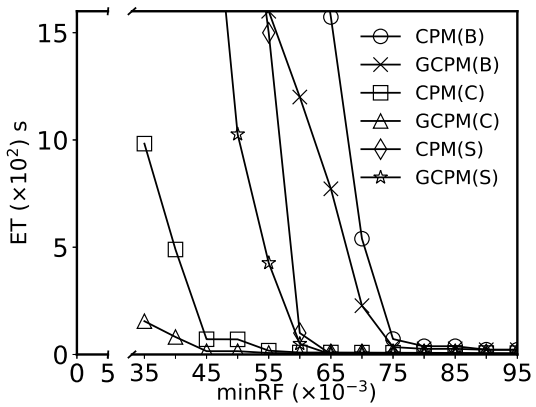
Dataset	Parameters	Addition of transactions	Deletion of transactions	Addition and Deletion of transactions
BMS-POS	Δ^+	[25K,250K]; SS = 25K; DF = 100K	–	[10K,120K]; SS = 10K; DF = 40K
	Δ^-	–	[25K,250K]; SS = 25K; DF = 100K	[10K,120K]; SS = 10K; DF = 40K
	$\mid D \mid$	265,596	515,596	265,596
	$minRF$	[0.035, 0.095]; SS = 0.005; DF = 0.065	0.065	0.065
	$minCS$	[0.1, 0.5]; SS = 0.1; DF = 0.5	0.5	0.5
	$maxOR$	[0.1, 0.8]; SS = 0.1; DF = 0.6	0.6	0.6
CABS120k08	Δ^+	[20K,200K]; SS = 20K; DF = 60K	–	[10K,120K]; SS = 10K; DF = 40K
	Δ^-	–	[20K,200K]; SS = 20K; DF = 60K	[10K,120K]; SS = 10K; DF = 40K
	$\mid D \mid$	202,776	402,776	202,776
	$minRF$	[0.035, 0.095]; SS = 0.005; DF = 0.035	0.035	0.035
	$minCS$	[0.1, 0.5]; SS = 0.1; DF = 0.5	0.5	0.5
	$maxOR$	[0.1, 0.8]; SS = 0.1; DF = 0.4	0.4	0.4
Synthetic	Δ^+	[5K,50K]; SS = 5K; DF = 25K	–	[2.5K,25K]; SS = 2.5K; DF = 10K
	Δ^-	–	[5K,50K]; SS = 5K; DF = 25K	[2.5K,25K]; SS = 2.5K; DF = 10K
	$\mid D \mid$	50K	100K	50K
	$minRF$	[0.035, 0.095]; SS = 0.005; DF = 0.045	0.045	0.045
	$minCS$	[0.1, 0.5]; SS = 0.1; DF = 0.3	0.3	0.3
	$maxOR$	[0.1, 0.8]; SS = 0.1; DF = 0.4	0.4	0.4

performance of CCPM is improved significantly over CPM. This is due to the less computation carried out in CCPM as compared to the computation carried out in CPM for extracting the same number of patterns. It can be observed that the performance trends of CCPM and CPM are similar in case of the other two datasets.

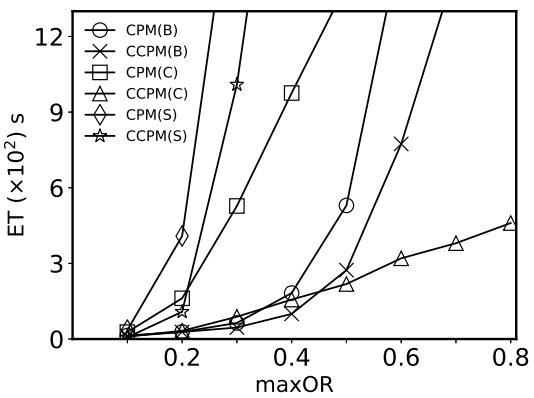
Figure 4(c) shows the effect of variations of the $minCS$ threshold on ET for three datasets. Observe that by varying $minCS$, ET of CPM is reduced slightly. This is due to the fact that less number of patterns are pruned due to the $minCS$ threshold. Due to the less computation carried out in CCPM as compared to the computation carried out in CPM, the performance of CCPM is improved. It can be observed that the performance trends of CCPM and CPM are similar on the other two datasets.

5.2 Effect of variations in Δ^+

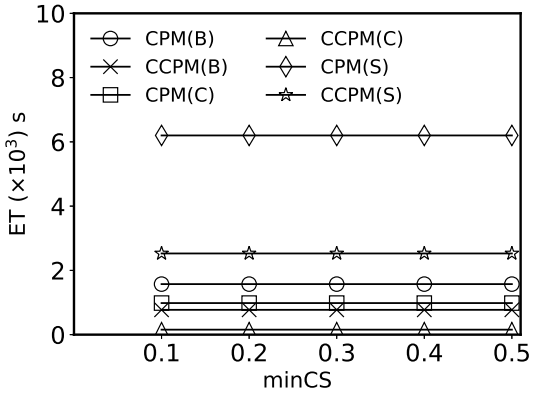
Figure 5(a) depicts the effect of variations in Δ^+ on ET for BMS-POS dataset. It can be observed that when Δ^+ is low, ET of CPM is low and as Δ^+ increases, ET is increased significantly. This is due to the fact that as Δ^+ increases, the size of D' increases. The variation in ET is observed due to the change in the characteristics of the dataset as we add Δ^+. Observe that



(a) ET ($minRF$)



(b) ET ($maxOR$)



(c) ET ($minCS$)

Fig. 4: Effect of variations in $minRF$, $maxOR$ and $minCS$ (B, C, S in parenthesis represent BMS-POS, CABS120k08 and Synthetic datasets respectively)

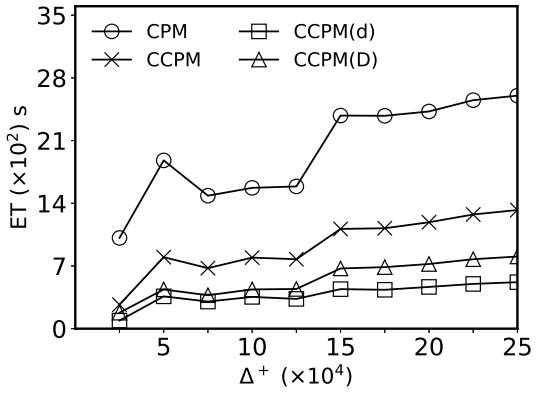
CCPM improves performance significantly over CPM at all values of Δ^+. This is due to the fact that CCPM only validates the available CPs of D in Δ^+, and a relatively small number of additional potential candidate patterns in Δ^+ through a single scan of D. Notably, at relatively low values of Δ^+, CCPM improves the performance over CPM significantly by up to four times. Figure 5(a) also indicates breaking of ET of CCPM into two parts: CCPM(d) and CCPM(D). CCPM(d) is the ET incurred for extracting candidate patterns from d (i.e., Δ^+) and pruning, and CCPM(D) is the ET incurred for scanning D. As expected, as Δ^+ increases, both CCPM(d) and CCPM(D) increase. However, the increase in CCPM(d) is more than that of CCPM(D) as the size of d increases.

Similar trends on CABS120k08 and Synthetic dataset can be observed in Figure 5(b) and Figure 5(c) respectively.

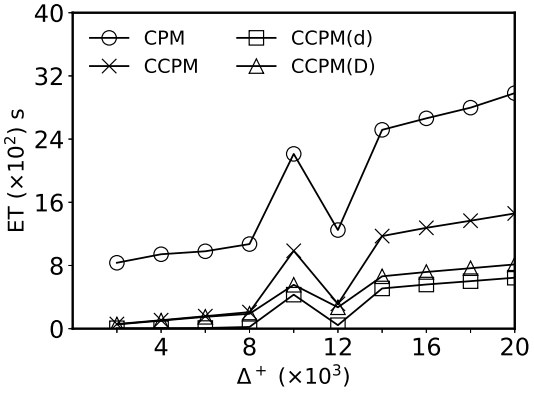
5.3 Effect of variations in Δ^-

Figure 6 depicts the effect of variations in Δ^-. In this experiment, it can be noted that the RF, $maxOR$, and $minCS$ values of a pattern are computed with respect to $D - \Delta^-$. Hence, the value of ET depends on the number of candidate patterns generated.

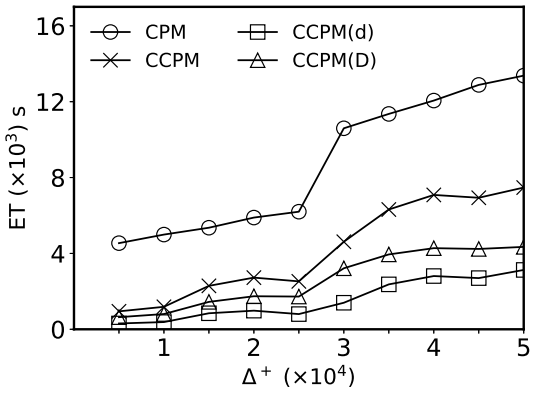
For BMS-POS, the changes in ET are shown in Figure 6(a). It can be observed that when the value of Δ^- is low, ET of CPM is high and as the value of Δ^- increases, ET has shown slightly increasing trend and decreased gradually. Observe that CCPM improves performance significantly over CPM at lower values of Δ^-. This is due to the fact that CCPM only validates the available CPs of D in Δ^-, and a relatively small number of additional potential candidate patterns in Δ^- through a single scan of $D - \Delta^-$. As expected, it can be observed that the performance crossover takes place at higher values of Δ^- and so CPM performs



(a) ET (BMS-POS)



(b) ET (CABS120k08)



(c) ET (Synthetic)

Fig. 5: Effect of variations in Δ^+ (d and D in the parenthesis represents ET of processing patterns in Δ^+ and D respectively)

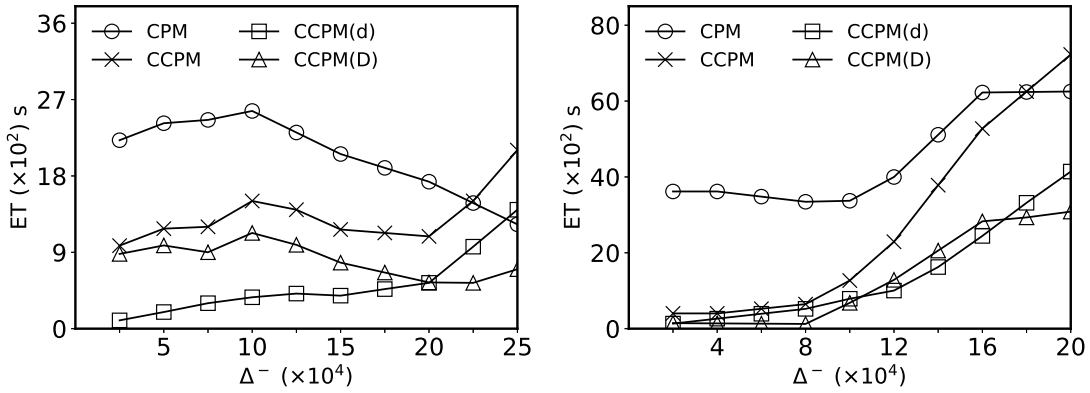
better than CCPM for higher values of Δ^-. Figure 6(a) also indicates breaking of ET of CCPM into two parts: CCPM(d) and CCPM(D). CCPM(d) is the ET incurred for extracting candidate patterns from d (i.e., Δ^-) and pruning, and CCPM(D) is the ET incurred for scanning $D - \Delta^-$. As expected, as Δ^- increases, CCPM(d) increases and CCPM(D) decreases. However, the increase in CCPM(d) is significant as the size of d increases significantly.

Similar performance improvement trends of CCPM over CPM on the CABS120k08 and Synthetic datasets can be observed in Figure 6(b) and Figure 6(c) respectively.

5.4 Effect of variations in Δ^+ and Δ^-

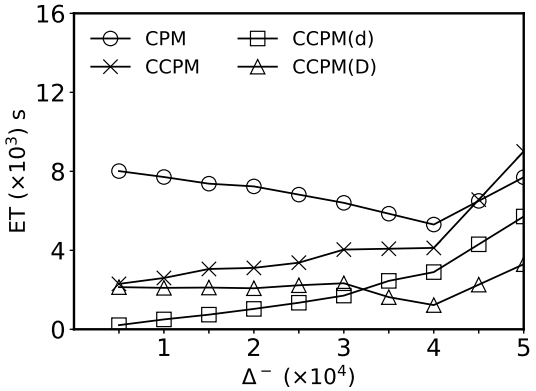
Figure 7 depicts the effect of variations in Δ^+ and Δ^-. For BMS-POS, the changes in ET are shown in Figure 7(a). It can be observed that when Δ^+ and Δ^- are low, ET of CPM is high and as Δ^+ and Δ^- increase, ET shows an overall stable trend. The deviations in the curve are due to the change in the distribution of items in the transactions.

Observe that CCPM improves performance significantly over CPM at lower values of Δ^+ and Δ^-. This is due to the fact that CCPM only validates the available CPs of D in Δ^+ and Δ^-, and a relatively small number of additional potential candidate patterns in Δ^+ and Δ^- through a single scan of $D - \Delta^-$. As expected, it can be observed that the performance crossover takes place at higher values of Δ^+ and Δ^- and so, CPM performs better than CCPM for higher values of Δ^+ and Δ^-. Figure 7(a) also indicates breaking of ET of CCPM into two parts: CCPM(d) and CCPM(D). CCPM(d) is the ET incurred for extracting candidate patterns from d (i.e., Δ^+ and Δ^-) and pruning, and CCPM(D) is the ET incurred for scanning $D - \Delta^-$. As expected, as Δ^+ and Δ^- increases, CCPM(d) increases and CCPM(D)



(a) ET (BMS-POS)

(b) ET (CABS120k08)



(c) ET (Synthetic)

Fig. 6: Effect of variations in Δ^- (d and D in the parenthesis represents *ET* of processing patterns in Δ^- and $D - \Delta^-$ respectively)

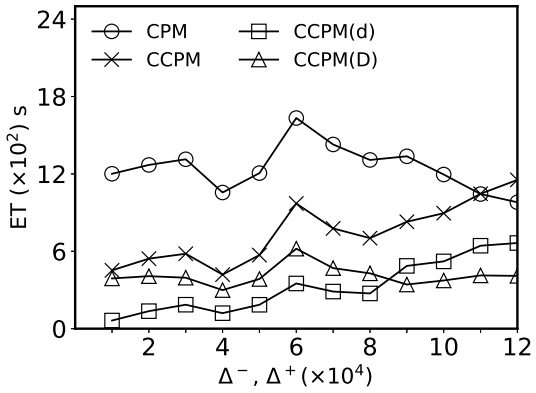
decreases. However, the increase in CCPM(d) is significant as the size of d increases significantly.

Similar performance improvement trends of CCPM over CPM on the CABS120k08 and Synthetic datasets can be observed in Figure 7(b) and Figure 7(c) respectively.

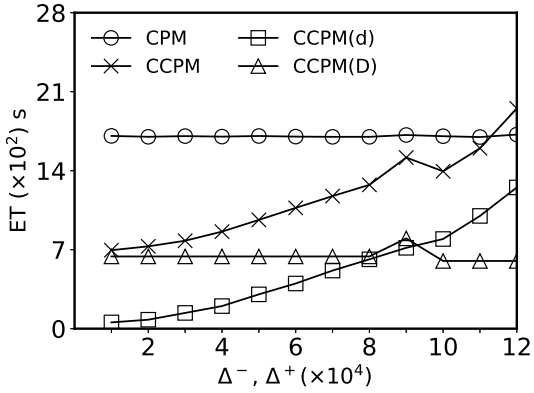
5.5 Effect of Ψ

In this section, we explain the impact of change in the *Flist*. As previously discussed at the beginning of Section 5, recall that we generated the Synthetic-Modified dataset by modifying the Synthetic dataset. We use the Synthetic-Modified dataset to study the effect of Ψ. In case of the Synthetic-Modified dataset, the nature of transactions changes significantly due to the addition of Δ^+ and the deletion of Δ^-. Figure 8 depicts the effect of variations in Δ^+ and Δ^- on *Synthetic-Modified* dataset. The experiments are conducted with the same default values of other parameters as that of synthetic dataset (Table 4).

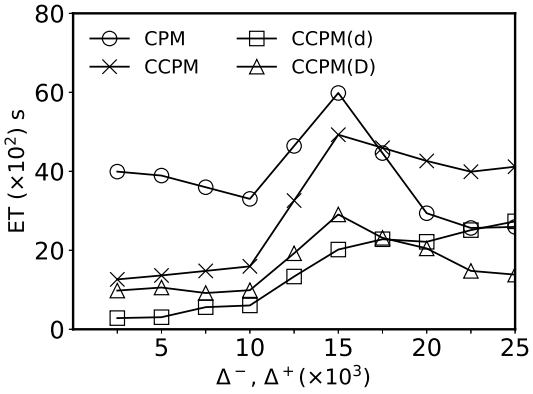
Note that, in the proposed approach, the change in the *Flist* impacts the number of candidate patterns to be validated in CCPM. As expected, the performance of CCPM is impacted with the change in *Flist*. At lower values of Δ^+ and Δ^-, CCPM performs better than CPM. However, as Δ^+ and Δ^- increase, CPM outperforms CCPM. Significant number of additional candidate patterns, which are to be validated, are generated from Δ^+ and Δ^- due to the change in *Flist* of D' as compared to *Flist* of D. As a result, CCPM performance deteriorates significantly over CPM. Figure 8(a) also indicates breaking *ET* of CCPM into two parts: CCPM(d) and CCPM(D). CCPM(d) is the *ET* incurred for extracting candidate patterns from d (i.e., Δ^+ and Δ^-) and pruning, and CCPM(D) is the *ET* incurred for scanning $D - \Delta^-$. As expected, as Δ^+



(a) ET (BMS-POS)

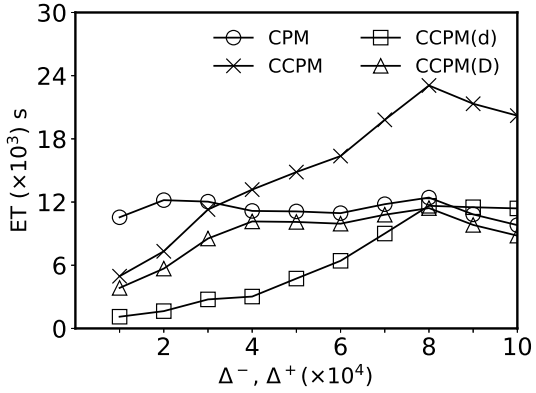


(b) ET (CABS120k08)

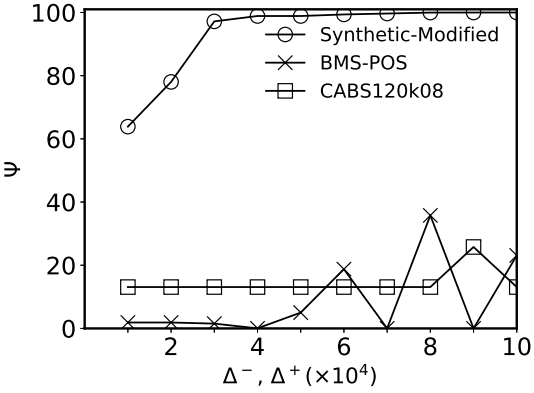


(c) ET (Synthetic)

Fig. 7: Effect of variations in Δ^+, Δ^- (d and D in the parenthesis represents ET of processing patterns in Δ^-, Δ^+ and $D - \Delta^-$ respectively)



(a) ET (Synthetic-Modified)



(b) Ψ

Fig. 8: Effect of Ψ (d and D in the parenthesis represents ET of processing patterns in Δ^-, Δ^+ and $D - \Delta^-$ respectively)

and Δ^- increase, the ET values of both CCPM(d) and CCPM(D) increase.

Figure 8(b) shows the values of Ψ, which represents the percentage of additional patterns due to change in $Flist$, due to the variation of Δ^+ and Δ^-. It can be observed that for the BMS-POS and CABS120k08 datasets, the values of Ψ are low, but they fluctuate with variations in Δ^+ and Δ^-. However, as expected, the value of Ψ reaches the maximum for the Synthetic-Modified dataset when the values of Δ^+ and Δ^- equal $3 * 10^4$. It can be observed that performance crossover is occurring at the same value in Figure 8(a).

From this experiment, it can be concluded that our proposed CCPM approach exhibits significant performance improvement under normal situations as $Flist$ does not change significantly as can be observed in the results in Figure 8(b) for the BMS-POS and CABS120k08 datasets respectively. However, under extreme and rare circumstances, where a given Δ^+ and/or Δ^- result in significant changes as compared to D w.r.t. the distribution of items in the transactions, it is better to extract coverage patterns from scratch.

6 Conclusion

In practice, pattern mining is performed on huge databases, which tend to get updated over the course of time due to the addition and/or deletion of transactions. Hence, incremental mining aims at *efficiently* and *incrementally* mining patterns when a database is updated. While incremental mining algorithms have been proposed for extracting frequent patterns, sequential patterns and utility patterns, existing works have so far not investigated incremental mining in the context of coverage patterns. Hence, in this paper, we have addressed the problem of incremental mining in the context of coverage patterns and proposed a comprehensive coverage pattern extraction algorithm for *efficiently* extracting the knowledge of coverage patterns when a set of transactions is added to the database and/or another set of transactions is deleted from the database. The results of our extensive performance evaluation using both real and synthetic datasets demonstrate that our proposed algorithm significantly outperforms the existing coverage pattern extraction algorithm. In the near future, we plan to develop incremental algorithms for the pattern growth approach towards the extraction of coverage patterns.

7 Conflict of Interest

On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Abdullah Z, Herawan T, Noraziah A, Deris MM (2012) DFP-Growth: An efficient algorithm for mining frequent patterns in dynamic database. In: Proc. International Conference on Information Computing and Applications, Springer, pp 51–58
2. Adnan M, Alhajj R, Barker K (2006) Constructing complete FP-tree for incremental mining of frequent patterns in dynamic databases. In: Proc. International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Springer, pp 363–372
3. Aggarwal CC, Bhuiyan MA, Al Hasan M (2014) Frequent pattern mining algorithms: A survey. In: Frequent pattern mining, Springer, pp 19–64
4. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules. In: Proc. Very Large Data Bases, Springer, pp 487–499
5. Agrawal R, Imieliński T, Swami A (1993) Mining association rules between sets of items in large databases. In: Proc. Special Interest Group on Management of Data, ACM, pp 207–216
6. Ahmed CF, Tanbeer SK, Jeong BS, Lee YK (2009) Efficient tree structures for high utility pattern mining in incremental databases. Transactions on Knowledge and Data Engineering, IEEE, 21(12):1708–1721
7. Aumann Y, Feldman R, Lipshtat O, Manilla H (1999) Borders: An efficient algorithm for association generation in dynamic databases. Journal of Intelligent Information Systems, Springer, 12(1):61–73
8. Borah A, Nath B (2020) Rare association rule mining from incremental databases. Pattern Analysis and Applications, Springer, 23(1):113–134
9. Budhiraja A, Reddy PK (2015) An approach to cover more advertisers in adwords. In: Proc. International Conference on Data Science and Advanced Analytics, IEEE, pp 1–10
10. Budhiraja A, Reddy PK (2017) An improved approach for long tail advertising in sponsored search. In: Proc. Database Systems for Advanced Applications, Springer, pp 169–184
11. Budhiraja A, Ralla A, Reddy PK (2019) Coverage pattern based framework to improve search engine advertising. International Journal of Data Science and Analytics, Springer, 8(2):199–211

12. Chang L, Wang T, Yang D, Luan H (2008) SeqStream: Mining closed sequential patterns over stream sliding windows. In: Proc. International Conference on Data Mining, IEEE, pp 83–92
13. Chau M, Fang X, Liu Sheng OR (2005) Analysis of the query logs of a web site search engine. Journal of the American Society for Information Science and Technology, Wiley Online Library, 56(13):1363–1376
14. Cheng H, Yan X, Han J (2004) IncSpan: Incremental mining of sequential patterns in large database. In: Proc. Special Interest Group on Knowledge Discovery and Data Mining, ACM, pp 527–532
15. Cheung DW, Wong C, Han J, Ng VT (1996) Maintenance of discovered association rules in large databases: An incremental updating technique. In: Proc. International conference on data engineering, IEEE, pp 106–114
16. Cheung DW, Lee SD, Kao B (1997) A general incremental technique for maintaining discovered association rules. In: Proc. Database Systems for Advanced Applications, World Scientific, pp 185–194
17. Chuang PJ, Tu YS (2019) Efficient frequent pattern mining in data streams. IOP Conference Series: Earth and Environmental Science, IOP Publishing, 234(1):012–066
18. Gangumalla L, Reddy PK, Mondal A (2019) Multilocation visibility query processing using portion-based transactional modeling and pattern mining. Data Mining and Knowledge Discovery, Springer, 33(5):1393–1416
19. Goethals B, Zaki MJ (2004) Advances in frequent itemset mining implementations: report on FIMI'03. Special Interest Group on Knowledge Discovery and Data Mining Explorations Newsletter, ACM, 6(1):109–117
20. Guo F, Li Y, Li L (2020) Research on improvement of high utility pattern mining algorithm over data streams. IOP Conference Series: Materials Science and Engineering, IOP Publishing, 715(1):012–022
21. Han J, Kamber M, Pei J (2011) Data Mining: Concepts and Techniques, 3rd edn. Elsevier, San Francisco, CA, USA
22. Hassani M, Töws D, Cuzzocrea A, Seidl T (2019) BFSPMiner: An effective and efficient batch-free algorithm for mining sequential patterns over data streams. International Journal of Data Science and Analytics, Springer, 8(3):223–239
23. Ho CC, Li HF, Kuo FF, Lee SY (2006) Incremental mining of sequential patterns over a stream sliding window. In: Proc. International Conference on Data Mining-Workshops, IEEE, pp 677–681
24. Ishita SZ, Ahmed CF, Leung CK, Hoi CH (2019) Mining regular high utility sequential patterns in static and dynamic databases. In: Proc. International Conference on Ubiquitous Information Management and Communication, Springer, pp 897–916
25. Karim MR, Cochez M, Beyan OD, Ahmed CF, Decker S (2018) Mining maximal frequent patterns in transactional databases and dynamic data streams: A spark-based approach. Information Sciences, Elsevier, 432:278–300
26. Kavya VNS, Reddy PK (2016) Coverage patterns-based approach to allocate advertisement slots for display advertising. In: Proc. International Conference on Web Engineering, Springer, pp 152–169
27. Lee G, Yun U, Ryu KH (2014) Sliding window based weighted maximal frequent pattern mining over data streams. Expert Systems with Applications, Elsevier, 41(2):694–708
28. Lin MY, Hsueh SC, Chan CC (2018) Mining and maintenance of sequential patterns using a backward generation framework. Journal of Information Science & Engineering, 34(5):1329–1349
29. Marascu A, Masseglia F (2006) Mining sequential patterns from data streams: a centroid approach. Journal of Intelligent Information Systems, Springer, 27(3):291–307
30. Masseglia F, Poncelet P, Teisseire M (2003) Incremental mining of sequential patterns in large databases. Data & Knowledge Engineering, Elsevier, 46(1):97–121
31. Nguyen LT, Nguyen P, Nguyen TD, Vo B, Fournier-Viger P, Tseng VS (2019) Mining high-utility itemsets in dynamic profit databases. Knowledge-Based Systems, Elsevier, 175:130–144
32. Nguyen SN, Sun X, Orlowska ME (2005) Improvements of IncSpan: Incremental mining of sequential patterns in large database. In: Proc. Pacific Asia Conference on Knowledge Discovery and Data Mining, Springer, pp 442–451
33. Nguyen TT (2018) Mining incrementally closed item sets with constructive pattern set. Expert Systems with Applications, Elsevier, 100:41–67
34. Noll MG, Meinel C (2008) The metadata triumvirate: Social annotations, anchor texts and search queries. In: Proc. International Conference on Web Intelligence and Intelligent Agent Technology, IEEE, pp 640–647
35. Ralla A, Reddy PK, Mondal A (2019) An incremental technique for mining coverage patterns in large databases. In: Proc. International Conference on Data Science and Advanced Analytics, IEEE, pp 211–220

36. Ryang H, Yun U (2016) High utility pattern mining over data streams with sliding window technique. Expert Systems with Applications, Elsevier, 57:214–231
37. Srinivas PG, Reddy PK, Bhargav S, Kiran RU, Kumar DS (2012) Discovering coverage patterns for banner advertisement placement. In: Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, pp 133–144
38. Srinivas PG, Reddy PK, Trinath AV, Bhargav S, Kiran RU (2015) Mining coverage patterns from transactional databases. Journal of Intelligent Information Systems, Springer, 45(3):423–439
39. Tanbeer SK, Ahmed CF, Jeong BS, Lee YK (2009) Sliding window-based frequent pattern mining over data streams. Information Sciences, Elsevier, 179(22):3843–3865
40. Trinath A, Srinivas PG, Reddy PK (2014) Content specific coverage patterns for banner advertisement placement. In: Proc. International Conference on Data Science and Advanced Analytics, IEEE, pp 263–269
41. Wang JZ, Huang JL (2018) On incremental high utility sequential pattern mining. Transactions on Intelligent Systems and Technology, ACM, 9(5):1–26
42. Yen SJ, Lee YS (2020) Efficient approaches for updating sequential patterns. In: Proc. Asian Conference on Intelligent Information and Database Systems, Springer, pp 553–564
43. Yun U, Lee G (2016) Incremental mining of weighted maximal frequent itemsets from dynamic databases. Expert Systems with Applications, Elsevier, 54:304–327
44. Yun U, Lee G, Yoon E (2019) Advanced approach of sliding window based erasable pattern mining with list structure of industrial fields. Information Sciences, Elsevier, 494:37–59
45. Yun U, Nam H, Lee G, Yoon E (2019) Efficient approach for incremental high utility pattern mining with indexed list structure. Future Generation Computer Systems, Elsevier, 95:221–239
46. Zhang B, Lin CW, Gan W, Hong TP (2014) Maintaining the discovered sequential patterns for sequence insertion in dynamic databases. Engineering Applications of Artificial Intelligence, Elsevier, 35:131–142