Journal of Intelligent Information Systems manuscript No.
(will be inserted by the editor)

Mining Coverage Patterns from Transactional Databases

P.Gowtham Srinivas* · **P.Krishna Reddy** · **A.V.Trinath** · **S.Bhargav** · **R.Uday Kiran**

the date of receipt and acceptance should be inserted later

Abstract We propose a model of coverage patterns (CPs) and approaches for extracting CPs from transactional databases. The model is motivated by the problem of banner advertisement placement in e-commerce web sites. Normally, an advertiser expects that the banner advertisement should be displayed to a certain percentage of web site visitors. On the other hand, to generate more revenue for a given web site, the publisher makes efforts to meet the coverage demands of multiple advertisers. Informally, a CP is a set of non-overlapping items covered by the certain percentage of transactions in a transactional database. The CPs do not satisfy the downward closure property. Efforts are being made in the literature to extract CPs using level-wise pruning approach. In this paper, we propose CP extraction approaches based on pattern growth techniques. Experimental results show that the proposed pattern growth approaches improve the performance over the level-wise pruning approach. The results also show that CPs could be used in meeting the demands of multiple advertisers.

Keywords coverage patterns · online advertising · projected pattern growth · banner advertisement · click stream mining · computational advertising

P.Gowtham Srinivas
*Corresponding Author
E-mail: gowtham.srinivas@research.iiit.ac.in

P.Krishna Reddy
E-mail: pkreddy@iiit.ac.in

A.V.Trinath
E-mail: venkatatrinath.atmakuri@research.iiit.ac.in

S. Bhargav
E-mail: sripada.b@samsung.com

R.Uday Kiran
E-mail: udayrage@tkl.iis.u-tokyo.ac.jp

1 Introduction

We have proposed a model of data mining pattern, called 'coverage pattern (CP)', and approaches for extracting CPs from transactional databases. A *CP* is a set of non-overlapping items covered by certain percentage of transactions in a transactional database.

The research in this paper is motivated with the problem of banner advertisement placement. A banner advertisement is described as a hyper-text link that is associated with a box containing graphics which is redirected to a particular web page when a user clicks on the banner [4]. The following three entities are involved in banner advertising: advertiser, publisher, and visitor. An advertiser is interested in endorsing products through banner advertisement. A publisher manages a web site that sells banner advertisement space. Finally, a visitor visits the web pages of a web site which contains banners. An advertiser has the goal of spreading the advertisement to a certain percentage of visitors. The goal of the publisher is to get more revenue by efficiently managing the advertising space available in the web pages of a web site and meeting the demands of multiple advertisers. For a given web site, one could analyze the visitors' behavior for a certain period by processing the transactions generated based on click stream data set and identify the sets of web pages that cover a given percentage of visitors. Such a knowledge could be used to place the banner advertisements by assuming similar visitors' behavior. However, the research issue here is to investigate the approaches for discovering the multiple sets of web pages which could cover a given percentage of visitors, based on the transactions extracted from the click stream data.

The proposed model of CPs could help the advertiser by making his advertisement visible to certain percentage of visitors. It also helps the publisher to meet the coverage demands of multiple advertisers by utilising the advertisement space available in all the potential web pages.

In the proposed model, the CPs are specified by defining two measures: *coverage support* and *overlap ratio*. Extracting the complete set of CPs is a challenging task, as the CPs do not satisfy the *downward closure property.* As a result, the computational cost of extracting CPs is prohibitively very high. In the literature [15, 17], by proposing improvements to the definition of CPs, we have proposed a level-wise pruning approach based on *downward-closure property.* We have also proposed a coverage pattern projected growth (CPPG) approach [16] based on the notion of *non-overlap pattern projection.* In this paper, after explaining the model of CPs, level-wise pruning approach, and CPPG, we have proposed an enhanced coverage pattern projected growth approach (ECPPG) which employs the notion of *minimal coverage patterns* in addition to the notion of *non-overlap projection.* We have explained the dynamics of CPs by conducting experiments on two real world click stream data sets and one synthetic data set. The results show that the ECPPG is more efficient than the level-wise pruning approach and CPPG. The results also show how the proposed approach could be used in meeting the demands of multiple advertisers.

In the literature, the notion of coverage is being used for solving the set cover problem [7] in set theory and node cover problem [10] in graphs respectively. In [18], the notion of coverage and overlap is used to examine the creation of a tag cloud for exploring and understanding a set of objects. In [5], the notion of coverage and overlap is used to solve the problem of topical query decomposition. The notion of Apriori principle [2] and database projection [1, 11, 12, 14] were explored to achieve high performance in frequent pattern mining and frequent sequence mining approaches.

The rest of the paper is organized as follows. In the next section 2, we discuss the model of CPs. In section 3, we present the overview of level-wise pruning and *CPPG* approaches for extracting CPs. In section 4, we present the proposed *ECPPG* approach. The experimental results are presented in section 5. The last section 6 contains summary and future work.

2 Model of Coverage Patterns

We consider the banner advertisement scenario and the transactions generated from click stream data of a web site to present the model. However, the model can be extended to any transactional data set. The model of CPs is as follows [15]. Let $W = \{w_1, w_2, \cdots, w_n\}$ be a set of identifiers of 'n' web pages and D be a set of transactions, where each transaction T, which is represented by transactional identifier TID, is a set of web pages such that $T \subseteq W$. A set of web pages $X = \{w_p, \cdots, w_q, w_r\} \subseteq W$, $1 \leq p, q, r \leq n$, is called the pattern. A pattern containing 'k' number of web pages is called a *k-pattern*. Let $|D|$ be the number of transactions in database D and T^{w_i}, $w_i \in W$ be the set of all $TIDs$ in D that contain web page w_i and $|T^{w_i}|$ denotes the number of transactions containing w_i.

Table 1: Transactional database. Each transaction contains the identifiers of the web pages.

TID	1	2	3	4	5	6	7	8	9	10
Web pages	a,b,c	a,c,e	a,c,e	a,c,d	b,d,f	b,d	b,d	b,e	b,e	a,b

The web pages which appear in large number of transactions are more interesting as these are potential pages to place advertisement. The terms *relative frequency* and *frequent web page* captures this aspect and are defined as follows.

Definition 1 *(Relative frequency (RF)) and Frequent web page) The RF of a web page w_i, denoted by $RF(w_i)$, is equal to the ratio of number of transactions that contain w_i to D, i.e., $RF(w_i) = \frac{|T^{w_i}|}{|D|}$. Let the term 'minimum relative frequency (minRF)' indicate user-specified threshold value. A web page w_i is frequent if $RF(w_i) \geq minRF$.*

We define the notion that how many users visit at least one web page in the set of web pages. It means that if we place an advertisement on all pages in the set, it will guarantee the delivery of the advertisement to the users who visit at least one page. We capture this notion with the terms *coverage set (CSet)* and *coverage support (CS).*

Definition 2 *(Coverage set and* coverage support (CS) *of a pattern* $X = \{w_p, \cdots, w_q, w_r\}$, $1 \leq p, q, r \leq n$*) The set of distinct TIDs containing at least one web page of* X *is called coverage set of pattern* X *and is denoted as* $CSet(X)$*. Therefore,* $CSet(X) = T^{w_p} \cup \cdots \cup T^{w_q} \cup T^{w_r}$*. The ratio of the size of the* $CSet(X)$ *to* D *is called the coverage-support of pattern* X *and is denoted as* $CS(X)$*, i.e.,* $CS(X) = \frac{|CSet(X)|}{|D|}$.

Given a pattern X, adding a new web page to this pattern which co-occurs with any of web pages belongs to X may not increase the *coverage support* significantly. From the advertisement point of view, such a pattern can be uninteresting to the advertiser. This is because the same user visits the web pages as there is an overlap of *coverage set* of X and the *coverage set* of new single web page. On the other hand, a new pattern is interesting if there is a minimum overlap of coverage sets of existing pattern and the new web page. To capture this aspect, we have introduced the measure *overlap-ratio (OR).*

Definition 3 *(Overlap ratio (OR) of a pattern* Y*.) The OR of a pattern* $Y = X \cup \{w_r\}$*, where* X *can be empty or* $X = \{w_p, \cdots, w_q\}$, $1 \leq p, q, r \leq n$ *is defined as the ratio of the number of transactions common in* $CSet(X)$ *and* $CSet(\{w_r\})$ *to* $CSet(\{w_r\})$*, i.e.,* $OR(Y) = \frac{|(Cset(X)) \cap (Cset\{w_r\})|}{|Cset\{w_r\}|}$.

For a pattern Y, $OR(Y) \in [0, 1]$. If $OR(Y) = 0$, there exists no common transactions between X and $\{w_r\}$. If $OR(Y) = 1$, w_r has occurred in all the transactions where at least one web page $w_j \in X$ has occurred. It can be noted that OR(Y) is zero if Y contains single page.

Note that a *pattern* is interesting if it has high *CS* and low *OR*. As a result, an advertisement is exposed to more number of users by reducing the repetitive display of the advertisement.

Definition 4 *(Coverage pattern (CP))A pattern* X *is said to be a* CP *if* $CS(X) \geq minCS$*,* $OR(X) \leq maxOR$ *and* $RF(w_i) \geq minRF$*,* $\forall w_i \in X$*. The variables,* $minCS$ *and* $maxOR$ *represent the user-specified* minimum coverage support *and* maximum overlap ratio*, respectively. A coverage pattern* X *having* $CS(X) = a\%$ *and* $OR(X) = b\%$ *is expressed as* $X : [CS = a\%,\ OR = b\%]$.

Example 1 *For the database in Table 1,* $T^a = \{1, 2, 3, 4, 10\}$*,* $T^b = \{1, 5, 6, 7, 8, 9, 10\}$*, and* $RF(a) = \frac{|T^a|}{|D|} = \frac{5}{10} = 0.5$*. If the user-specified* $minRF = 0.5$*, then 'a' is called a frequent web page as* $RF(a) \geq minRF$*.* $CSet(\{a, b\}) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$*,* $CS(\{a, b\}) = \frac{|CSet(\{a,b\})|}{|D|} = \frac{10}{10} = 1$ *and* $OR(\{a, b\}) = \frac{|CSet(\{b\}) \cap CSet(\{a\})|}{|CSet(\{a\})|} = \frac{2}{5} = 0.4$*. If* $minRF = 0.4$*,* $minCS = 0.7$ *and* $maxOR = 0.5$*, then the pattern* $\{a, b\}$ *is a* CP*.*

Problem Statement The problem statement of mining CPs is as follows. Given a set of web pages W (or items), transactional database D, $minRF$, $minCS$ and $maxOR$, discover the complete set of CPs.

Coverage patterns and sorted closure property Extracting the complete set of CPs for a given $minRF$, $minCS$ and $maxOR$ is a challenging task, because the measure CS does not satisfy the *downward closure property.* Although a pattern satisfies $minCS$, it is not necessary that all its non-empty subsets will also satisfy $minCS$ value. For example, consider the patterns $\{a\}$, $\{e\}$ and $\{a, e\}$ from Table 1. The coverage-supports of these patterns are 0.5, 0.4 and 0.7, respectively. If the user specifies $minCS = 0.7$, the pattern $\{a, e\}$ satisfies $minCS$ value. However, the patterns $\{a\}$ and $\{e\}$ do not satisfy $minCS$ value. It can also be noted that the preceding definition of OR measure also does not satisfy *downward closure property.* From Table 1, $OR(\{a, c, b\}) = \frac{|CSet(\{a,c\}) \cap CSet(\{b\})|}{|CSet(\{b\})|} = \frac{2}{7}$ and $OR(\{a, c\}) = \frac{|CSet(\{a\}) \cap CSet(\{c\}|)}{|CSet(\{c\})|} = \frac{4}{4} = 1$. If the user specifies $maxOR = \frac{2}{7}$, $OR(\{a, c, b\}) \leq maxOR$ value. However, $OR(\{a, c\}) \nleq maxOR$ value.

It was observed that the OR satisfies *downward closure property*, if the items are ordered in descending order of their frequencies. This property is called *sorted-closure property* [13] which is given in Property 1 and the correctness is shown in Lemma 2.

Property 1 *Sorted Closure Property: Let $X = \{w_p, \cdots w_q, w_r\}$ be a pattern such that $RF(w_p) \geq \cdots RF(w_q) \geq RF(w_r)$ and $1 \leq p, q, r \leq n$. If $OR(X) \leq maxOR$, then all its non-empty subsets containing w_r and having size $k \geq 2$ will also have* overlap ratio *less than or equal to maxOR.*

Lemma 1 *If $X \subset Y$, then $CSet(X) \subseteq CSet(Y)$.*

Lemma 2 *If $OR(X) \leq maxOR$, then $OR(Y) \leq maxOR$. $\forall\ Y \subset X, w_r \in Y$.*

Proof Let w_a, w_b and w_c be the web pages having $RF(w_a) \geq RF(w_b) \geq RF(w_c)$. If $OR(\{w_a\} \cup \{w_c\}) > maxOR$, then $OR((\{w_a\} \cup \{w_b\}) \cup \{w_c\}) > maxOR$ because from Lemma 1, $\frac{|CSet(\{w_a\}) \cap CSet(\{w_c\})|}{|CSet(\{w_c\})|} \leq \frac{|CSet(\{w_a \cup w_b\}) \cap CSet(\{w_c\})|}{|CSet(\{w_c\})|}$.

Next, we redefine the OR which was defined in Definition 3 by ordering the web pages of pattern in the descending order of frequencies.

Definition 5 *(Overlap ratio (OR) of a pattern.) OR of a pattern $X = \{w_p, \cdots, w_q, w_r\}$, where $1 \leq p, q, r \leq n$ and $|T^{w_p}| \geq \cdots \geq |T^{w_q}| \geq |T^{w_r}|$, is the ratio of the number of transactions common in $X - \{w_r\}$ and $\{w_r\}$ to the number of transactions in w_r, i.e., $OR(X) = \frac{|(Cset(X - \{w_r\})) \cap (Cset\{w_r\})|}{|Cset\{w_r\}|}$.*

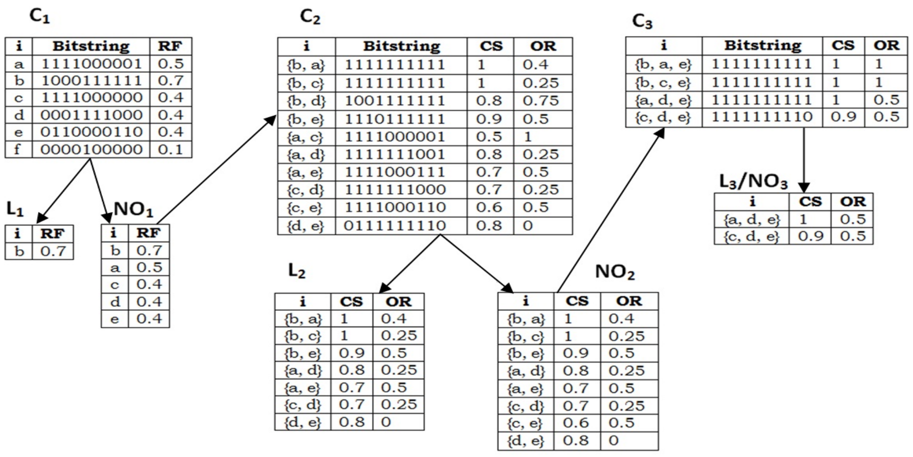
An item 'a' is said to be a non-overlap item with respect to a pattern X if $OR(\{X, a\})$ is no greater than $maxOR$ and $RF(w_i) \geq minRF, \forall w_i \in \{X, a\}$. The notion of *non-overlap pattern* is defined as follows.

Definition 6 *(Non-overlap pattern X.) A pattern X is said to be non-overlap if $OR(X) \leq maxOR$ and $RF(w_i) \geq minRF, \forall w_i \in X$.*

3 Overview of CP extraction approaches: level-wise pruning and projected pattern growth

A naive approach to find the complete set of CPs for a dataset consisting of 'n' web pages is to generate all $(2^n - 1)$ combinatorial patterns and select those patterns that satisfy *minRF*, *minCS*, and *maxOR* constraints. The major issue with this approach is a large search space leading to huge computational cost. In this section we give the overview of two effective approaches to extract CPs: One approach is multiple pass, candidate generation and test approach similar to Apriori based on the *OR* definition satisfying *sorted closure property* and the other approach based on the notion of *non-overlap pattern projection*.

3.1 Coverage Pattern Mining Method (CMine)



C_1

i	Bitstring	RF
a	1111000001	0.5
b	1000111111	0.7
c	1111000000	0.4
d	0001111000	0.4
e	0110000110	0.4
f	0000100000	0.1

L_1

i	RF
b	0.7

NO_1

i	RF
b	0.7
a	0.5
c	0.4
d	0.4
e	0.4

C_2

i	Bitstring	CS	OR
{b, a}	1111111111	1	0.4
{b, c}	1111111111	1	0.25
{b, d}	1001111111	0.8	0.75
{b, e}	1110111111	0.9	0.5
{a, c}	1111000001	0.5	1
{a, d}	1111111001	0.8	0.25
{a, e}	1111000111	0.7	0.5
{c, d}	1111111000	0.7	0.25
{c, e}	1111000110	0.6	0.5
{d, e}	0111111110	0.8	0

L_2

i	CS	OR
{b, a}	1	0.4
{b, c}	1	0.25
{b, e}	0.9	0.5
{a, d}	0.8	0.25
{a, e}	0.7	0.5
{c, d}	0.7	0.25
{d, e}	0.8	0

NO_2

i	CS	OR
{b, a}	1	0.4
{b, c}	1	0.25
{b, e}	0.9	0.5
{a, d}	0.8	0.25
{a, e}	0.7	0.5
{c, d}	0.7	0.25
{c, e}	0.6	0.5
{d, e}	0.8	0

C_3

i	Bitstring	CS	OR
{b, a, e}	1111111111	1	1
{b, c, e}	1111111111	1	1
{a, d, e}	1111111111	1	0.5
{c, d, e}	1111111110	0.9	0.5

L_3/NO_3

i	CS	OR
{a, d, e}	1	0.5
{c, d, e}	0.9	0.5

Fig. 1: Working of CMine algorithm. The term 'i' is an acronym for the item set (or web pages). Bitstring represents bitmap representation of transactions of Table 1 for CSet(i)

The coverage pattern mining (CMine) approach proposed in [15] is as follows. Here, the *sorted closure property* of *non-overlap patterns* is exploited for minimizing the search space while mining the complete set of CPs by designing an algorithm *CMine* similar to Apriori algorithm [2] by adopting a multiple-pass, candidate generation and test approach.

Let F be a set of frequent items, C_k be a set of candidate k-patterns, L_k be a set of coverage k-patterns and NO_k be a set of non-overlap k-patterns. The algorithm *CMine* begins with a scan of database and discovers set of all frequent web pages (denoted as F) and coverage 1-patterns (denoted as C_1). Non-overlap patterns (denoted as NO_1) will be the set of all frequent 1 items. Next, the items in NO_1 are sorted in descending order of their frequencies. Using NO_1 as the seed set, candidate patterns C_2 are generated by computing $NO_1 \bowtie NO_1$ (self-join). From C_2, the patterns that satisfy *minCS* and *maxOR* are generated as L_2. Simultaneously, all candidate

2-patterns that satisfy *maxOR* constraints are generated as non-overlap 2-patterns, NO_2. Since *OR* satisfies *sorted closure property*, C_3 is generated by computing $NO_2 \bowtie NO_2$. This process is repeated until no new CPs are found or no new candidate patterns can be generated. The process of extraction of CPs using *CMine* [15] for the dataset given in Table 1, for the user-specified *minRF*, *minCS*, and *maxOR* as 0.4, 0.7, and 0.5 respectively is depicted in Figure 1.

3.2 Coverage Pattern Projected Growth Method (CPPG)

Similar to Apriori [2], $CMine$ generates potentially huge set of candidate non-overlap patterns and requires multiple scans of database. Also, it takes several iterations to mine long non-overlap patterns. It can be observed that the complexity of *CMine* is due to its step-wise candidate non-overlap pattern generation and test. An effort has been made to extend the notion of *projection database* to improve the performance of *CMine*. An improved approach called Coverage Patterns Projected Growth (CPPG) [16] has been proposed in which the CPs are extracted in an item-wise manner based on the notion of *non-overlap projected database*. For computing CPs, we build the non-overlap projected database for each frequent item. The process partitions both the data set and the set of *non-overlap patterns* to be tested, and confines each test being conducted to the corresponding smaller projected database. The performance could be improved with the significant reduction in the database size due to the projection. We now explain the notion of *non-overlap projected database* after explaining the terms *prefix*, *postfix*, *f-list*, *non-overlap transaction* and *non-overlap projection*.

Definition 7 *(Prefix and postfix). Given a pattern* $X = \{i_1, i_2, ..., i_n\}$*, a pattern* $Y = \{i_1', i_2', ..., i_m'\}$ *is called a prefix of* X *if and only if* $i_j' = i_j$ $\forall j \leq m \leq n$*. Here, we call the pattern* $Z = \{i_{m+1}, i_{m+2}, ..., i_n\}$ *as postfix of the pattern* X *with respect to the pattern* Y*.*

Example 2 *For the pattern,* $X = \{b, a, c\}$*, the patterns* $\{b\}$ *and* $\{b, a\}$ *are the prefixes of pattern* X*. The pattern* $\{a, c\}$ *is the postfix of pattern* X *with respect to the prefix* $\{b\}$*, and* $\{c\}$ *is the postfix of pattern* X *with respect to the prefix* $\{b, a\}$*.*

Definition 8 *(f-list). Given a transactional database,* f-list *is the list of all items in the database with decreasing order of their frequencies.*

Example 3 *In Table 1, frequencies of the items in the database are as follows:* $\{a : 5, b : 7, c : 4, d : 4, e : 4, f : 1\}$*. So, the f-list for this database is:* $\{b, a, c, d, e, f\}$*.*

Definition 9 *(Non-overlap transaction). A transaction* T*, in a database* D*, is said to be a non-overlap transaction with respect to a pattern* $X = \{w_p, \cdots, w_q, w_r\}$*, where* $1 \leq p, q, r \leq n$ *and* $|T^{w_p}| \geq \cdots \geq |T^{w_q}| \geq |T^{w_r}|$ *if and only if* T *does not have any of the web page belonging to* X*.*

Example 4 *Consider transaction, $T = \{a, b, c\}$ with TID = 1 from Table 1. Let $X = \{d, e\}$. Here T is a non-overlap transaction with respect to X because T does not have any of the item belonging to X. Suppose if X=$\{a, e\}$, then T is not a non-overlap pattern with respect to X because T contains $'a'$ which belongs to X.*

Definition 10 *(Non-overlap projection). A non-overlap projection of a transaction T, with respect to a pattern $X = \{w_p, \cdots, w_q, w_r\}$, where $1 \leq p, q, r \leq n$ and $|T^{w_p}| \geq \cdots \geq |T^{w_q}| \geq |T^{w_r}|$ is non-empty if T does not have any item belonging to X. Also the non-overlap projection of T with respect to X does not have any item 'w_i' occurring before 'w_r' in the f-list. Finally the items in the non-overlap projection are ordered with respect to the f-list.*

Example 5 *Consider the transaction, $T = \{b, d, f\}$ with TID = 5 from Table 1. Now non-overlap projection of T with respect to $X = \{a, c\}$ is $Y = \{d, f\}$. Here non-overlap projection of T is not empty because T does not have either 'a' or 'c' in it. Also in the non-overlap projection of T with respect to X, we do not have 'b' in Y because 'b' occurs before 'c' in the f-list.*

Definition 11 *(Non-overlap projected database). A non-overlap projected database of a pattern X, with respect to a database D contains the non-overlap projections of all the transactions in D with respect to the patternX.*

Example 6 *For the database D in Table 1, the non-overlap projected database of pattern $X = \{a\}$ with respect to D is shown in Table 2.*

Table 2: Projected database with respect to X={a}

TID	**Pages**	**TID**	**Pages**	**TID**	**Pages**	**TID**	**Pages**	**TID**	**Pages**
5	d, f	6	d	7	d	8	e	9	e

CPPG Method with an example

For the database *D* shown in Table 1, with minRF=0.4, minCS=0.7 and maxOR=0.5, the complete set of CPs can be mined by *CPPG* method in the following steps.

Table 3: Transactional database arranged in the order of f-list

TID	1	2	3	4	5	6	7	8	9	10
Pages	b, a, c	a, c, e	a, c, e	a, c, d	b, d	b, d	b, d	b, e	b, e	b, a

i. Find length-1 frequent patterns and construct the f-list. Scan D once to find all frequent items in the transactions. Each of these frequent items is a length-1 *non-overlap pattern*. They are {a:5}, {b:7}, {c:4}, {d:4}, {e:4}. Where {pattern:count} represents the pattern and its associated count. Now construct the f-list = (b, a, c, d, e).
ii. Arrange the transactions in the f-list order by remove the items that do not satisfy *minRF* from *D*. The transactions in Table 1 arranged in the f-list order is given in Table 3.

iii. Now we partition the complete set of *non-overlap patterns* into the following subsets according to the five prefixes in the f-list. They are the ones having prefix ‘b’, prefix ‘a’, prefix ‘c’, prefix ‘d’ and prefix ‘e’.

iv. For each item in the f-list construct its corresponding non-overlap projected database on D. If the CS of this item is no less than $minCS$ report it as a CP. Now recursively mine the non-overlap projected databases as follows. Find the support count of each item in the projected database. Using this count find the non-overlap patterns. For each non-overlap pattern find the CS and report if it is a CP. Now recursively project the non-overlap pattern on its corresponding projected database and continue the mining process until the projected database size is empty. The *non-overlap projected databases* as well as the *non-overlap* and *CPs* found in them are listed in Table 4, while the mining process is explained as follows.

Table 4: Projected databases, non-overlap and CPs generated in CPPG

Prefix	**Non-overlap projected database**	**Non-overlap patterns**	**Coverage Patterns**
b	{{a,c,e},{a,c,e},{a,c,d}}	{{b},{b,a},{b,c},{b,e}}	{{b},{b,a},{b,c}, {b,e}}
a	{{d},{d},{d},{e},{e}}	{{a},{a,d},{a,e},{a,d,e}}	{{a,d}, {a,e}, {a,d,e}}
c	{{d},{d},{d},{e},{e}}	{{c},{c,d},{c,e},{c,d,e}}	{{c,d}, {c,d,e}}
d	{{e}, {e}, {e}, {e}}	{{d}, {d,e}}	{{d,e}}
e	empty	{{e}}	empty

First let us find the *non-overlap patterns* having prefix ‘a’. So, D is projected with respect to {a} to form the {a}’s non-overlap projected database, which consists of five non-overlap projections: {5:{d}}, {6:{d}}, {7:{d}}, {8:{e}}, {9:{e}}. Here {TID: Projection} denotes the TID of the projection in the original database D. By scanning {a}’s non-overlap projected database once, all the length-2 non-overlap patterns having prefix {a} can be found they are {a, d}:[0.25, 0.8] and {a, e}:[0.5, 0.7]. Here {pattern}:[OR, CS] represents the pattern along with its *overlap ratio* and *coverage support* respectively. Recursively all the *non-overlap patterns* having prefix {a} can be partitioned into two subsets: 1. those having prefix {a, d} and 2. those having prefix {a, e}. These subsets can be mined by constructing respective non-overlap projected databases and mining each recursively as follows. The {a, d} non-overlap projected database consists of only two non-overlap projections {{e}, {e}}. By scanning {a, d}’s non-overlap projected database once, all the length-3 non-overlap patterns having prefix {a, d} can be found they are {a, d, e}:[0.5, 1]. Recursively all the *non-overlap patterns* having prefix {a, d} can be partitioned into only one subset those having prefix {a, d, e}. As the non-overlap projection database of {a, d, e} does not have any non-overlap projections in it the mining process for patterns having prefix {a, d, e} will stop here. The {a, e} non-overlap projected database consists of zero non-overlap projections so the mining process for patterns having prefix {a, e} will stop here. Similarly, we can find the *non-overlap patterns* having prefix {b}, {c}, {d} and {e}, respectively, by constructing {b}-, {c}-, {d}- and {e}-non-overlap projected databases and mining them respectively. During the mining process for each *non-overlap pattern* found we compute the *CS* and store the pattern in the *CP* set if it satisfies *minCS* constraint.

4 Enhanced Coverage Pattern Projected Growth Method (ECPPG)

It can be observed that even though *CS* does not satisfy *downward closure property*, it satisfies *upward closure property* [6]. According to *upward closure property* if a pattern satisfies *CS*, all its super-sets satisfy *CS*. We exploit this property to avoid the computation of unnecessary projections, once we find that a pattern satisfies *CS*. We can observe that once a pattern X meet the *CS* and *OR* constraints for the first time, generation of CPs starting with prefix X can be easily found. We call such a pattern as *minimal CP*. Once a minimal CP, X, is found we can find the complete set of CPs starting with prefix X simply by adding iteratively the items with lower precedence in the f-list to the pattern X by checking only *OR*. There is no need for performing any database scans and projections in extracting these patterns, which improves the mining process significantly. The proposed Enhanced Coverage Pattern Projected Growth Method (ECPPG) improves the performance over *CPPG* by using the notion of *minimal CP*. Similar to *CPPG*, it recursively projects the transaction database into a set of smaller projected databases and extract subsequent non-overlap patterns in each projected database. Now, we define the notion of *minimal CP* after explaining the *upward closure property* of *CS*.

Property 2 *(Upward closure property). Let X be a pattern and Y be super set of pattern X. If $CS(X) \geq minCS$, then all its super sets such as Y will also have* coverage support *greater than or equal to $minCS$.*

Lemma 3 *If $CS(X) \geq minCS$, then $CS(Y) \geq minCS$, $\forall\ Y \supset X$.*

Proof Since, $Y \supset X$, $|CSet(X)| \leq |CSet(Y)|$ [From Lemma 1]. From this, $\frac{|CSet(X)|}{|D|} \leq \frac{|CSet(Y)|}{|D|}$. If $CS(X) \geq minCS$, then $CS(Y) \geq minCS$.

A *Minimal Coverage Pattern* is a pattern ‘p’ such that any non-empty subset of ‘p’ has *CS* less than *minimum coverage support (minCS)* and *OR* less than *maximum overlap ratio (maxOR)* thresholds.

Definition 12 *(Minimal coverage pattern (minimal CP)). A pattern $X = \{w_p, \cdots, w_q, w_r\}$, where $1 \leq p, q, r \leq n$ and $|T^{w_p}| \geq \cdots \geq |T^{w_q}| \geq |T^{w_r}|$ is called a minimal CP if no proper subset of X has coverage support $CS \geq minCS$ and overlap ratio $OR \geq maxOR$.*

Example 7 *Consider the database shown in Table 1. From Table 5 we can say that pattern $\{c,\ d,\ e\}$ is not minimal CP for minCS value of 0.7, because its subsets $\{c,\ d\}$ and $\{d,\ e\}$ satisfy constraint on minCS. We can say that the pattern $\{c,\ d\}$ is a minimal CP because no proper subset of $\{c,\ d\}$ satisfy constraint on $minCS$.*

4.1 ECPPG Algorithm with an example

Algorithm 1 ECPPG:Enhanced Coverage Pattern Projected Growth Method

```
Input: A transactional database D, and the user-specified minimum relative frequency
(minRF), minimum coverage support (minCS) and maximum overlap ratio (maxOR)
Output: The complete set of CPs.
Method:
1. Scan database $D$, find all the frequent 1-items and construct f-list.
2. For each frequent 1-item 'x', if $CS(x) \geq minCS$ call $GenerateCovPatterns(x, f\text{-}list)$,
   else construct its non-overlap projected database $NOP(D)\mid_x = ConstructNOP(D, x, f\text{-}list)$
   and call $ECPPGRec(x, 1, NOP(D)\mid_x, f\text{-}list)$.
Subroutine $ECPPGRec(\alpha, l, NOP(D)\mid_\alpha, f\text{-}list)$
Parameters: $\alpha$ : a non-overlap pattern; $l$: the length of $\alpha$; $NOP(D)\mid_\alpha$: the non-overlap
projected database of $\alpha$ with respect to $D$.
Method:
1. Scan $NOP(D)\mid_\alpha$ once, find the set of non-overlap items 'i'.
2. For each non-overlap item 'i', append it to $\alpha$ to form a non-overlap pattern $\alpha'$.
   If it satisfies $minCS$ constraint and no proper subset of $\alpha'$ satisfies $minCS$, call
   $GenerateCovPatterns(\alpha', f\text{-}list)$.
3. else construct non-overlap projected database of $\alpha'$, $NOP(D)\mid_{\alpha'} =
   ConstructNOP(NOP(D)\mid_\alpha, \alpha', f\text{-}list)$ and call $ECPPGRec(\alpha', l+1, NOP(D)\mid_{\alpha'}, f\text{-}list)$
Subroutine $GenerateCovPatterns(X, list)$
Parameters: $X$ : a minimal CP; $list$: f-list.
Method:
1. Get the last item 'i' in minimal CP $X$.
2. For each item having precedence lower than 'i' in f-list, append it to form pattern $\alpha$,
   and calculate $OR(\alpha)$ by computing boolean AND of bit set of pattern, $\alpha$ and item, 'i'. If
   $OR(\alpha) \leq maxOR$, output $\alpha$ and call $GenerateCovPatterns(\alpha, list)$ recursively.
Subroutine $ConstructNOP(D, x, list)$
Parameters: $D$ : transactional database; $x$ : pattern; $list$: f-list.
Method: For every transaction $t$ in transactional database $D$, find non-overlap projection
of $t$ with respect to $x$ and form a new database $D'$. Output $D'$.
```

We explain the Algorithm 1 as follows. First, scan the transactional database and find length-1 frequent patterns and construct f-list (step-1). Each of these frequent patterns is a length-1 non-overlap pattern. Next, transactional database is arranged in the f-list order, by removing the items that do not satisfy *minRF*. Now we partition the complete set of non-overlap patterns into subsets having each item in f-list as prefix. For each item in the f-list, check for its *CS* (step-2). If it is no less than *minCS* report it as a *minimal CP*. Otherwise, if the *CS* is less than *minCS*, the corresponding non-overlap projected database on *D* is constructed by calling *ConstructNOP* method. Now recursively mine the non-overlap projected databases by calling *ECPPGRec* which is given as follows. Find the support count of each item in the projected database. For each non-overlap pattern find the *CS* of the pattern. If it is no less than *minCS*, find the coverage supports of all proper subsets of the pattern. If no subset of pattern exists such that it is no less than *minCS*, report it as *minimal CP*. Now recursively project the non-overlap pattern, only if it is not minimal on its corresponding projected database and continue the mining process until the projected database size is empty. Once the minimal CPs are extracted, we can mine the complete set of CPs having minimal CP as prefix by calling *GenerateCovPatterns* which iteratively adds the items in the f-list with lower precedence and then checking for its *OR*.

For the database in Table 1, with *minRF* = 0.4, *minCS* = 0.7 and *maxOR* = 0.5, we explain the extraction of set of minimal CPs and complete set of CPs. Scan the database and find the item frequencies. They are {a:5}, {b:7}, {c:4}, {d:4}, {e:4}. Now construct the f-list = (b, a, c, d, e). Reorder the database as per the f-list order by removing non-frequent items. The new database formed is shown in Table 3. Next we partition the complete set of non-overlap patterns into the following subsets according to the five prefixes in the f-list.

Table 5: Projected databases, non-overlap and minimal CPs generated in ECPPG

Pre-fix	**Non-overlap projected database**	**Non-overlap patterns**	**Minimal set of CPs**	**Complete set of CPs**
b	Not formed	{{b}}	{{b}}	{{b},{b,a},{b,c},{b,e}}
a	{{d},{d},{d},{e},{e}}	{{a},{a,d},{a,e}}	{{a,d},{a,e}}	{{a,d},{a,e},{a,d,e}}
c	{{d},{d},{d},{e},{e}}	{{c},{c,d},{c,e}}	{{c,d}}	{{c,d},{c,d,e}}
d	{{e},{e},{e},{e}}	{{d}, {d,e}}	{{d,e}}	{{d,e}}
e	empty	{{e}}	empty	empty

We now explain the extraction of *non-overlap patterns* with prefix 'a'. *CS* of pattern {a} is 0.5 which is no greater than minCS=0.7. So, D is projected with respect to {a} to form the {a}'s non-overlap projected database, which consists of five non-overlap projections: {5:{d}}, {6:{d}}, {7:{d}}, {8:{e}}, {9:{e}}. By scanning {a}'s non-overlap projected database once, all the length-2 non-overlap patterns having prefix 'a' can be found they are {a, d}:[0.25, 0.8] and {a, e}:[0.5, 0.7]. Recursively all the non-overlap patterns having prefix 'a' can be partitioned into two subsets: 1. those having prefix {a, d} and 2. those having prefix {a, e}. These subsets can be mined by constructing respective non-overlap projected databases and mining each recursively as follows. But, here the *CS* of both patterns {a, d} and {a, e} is greater than *minCS*. They are reported as minimal CPs. Mining of minimal CPs having prefix 'a' is completed. Complete set of CPs having prefix 'a' is generated from minimal CP by adding iteratively the items with lower precedence in f-list to the pattern and checking for *OR*. For the minimal CP {a, d}, 'e' is the only item after 'd' in f-list. Hence, item 'e' is added to pattern {a, d} and the *OR* of new pattern {a, d, e} is computed as 0.5. As $OR(\{a, d, e\}) \leq maxOR$, pattern {a, d, e} is reported as CP. As there is no item with lower precedence than 'e' in f-list, mining of CPs with prefix {a, e} stops here. If we are not using the notion of minimal CPs, we have to generate the non-overlap projected database of {a, d} and scan the same in order to extract the CP {a, d, e}.

Similarly, we can find the non-overlap patterns having prefix {b}, {c}, {d} and {e}, respectively. The non-overlap projected databases, non-overlap patterns, minimal CPs and CPs found by this method are shown in Table 5.

In this process of extracting CPs, it can be observed that non-overlap patterns with prefix 'b', $\{\{b,a\},\{b,c\},\{b,e\}\}$ are not formed as $\{b\}$ satisfies *CS*, instead the patterns $\{\{b,a\},\{b,c\},\{b,e\}\}$ are generated by just comparing the *OR* threshold. The non-overlap patterns with prefix 'a', $\{\{a,d,e\}\}$ is not formed as $\{a,d\}$ and $\{a,e\}$ satisfies *CS* and non-overlap patterns with prefix 'c', $\{\{c,d,e\}\}$ is not formed as $\{c,d\}$ satisfies *CS*.

4.2 Discussion

We explain how the notions of non-overlap projection and minimal CPs could improve the efficiency: From Figure 1, Table 5 and Table 4, one can verify that *CMine*, *CPPG* and *ECPPG* returns the same set of CPs. Also, one can observe that *CPPG* and *ECPPG* does not generate the candidate non-overlap patterns {a, c}, {b, d}, {b, a, c}, {b, a, e}, {b, c, e} during the mining process because it uses only non-overlap projections to find the CPs, whereas *CMine* generates them during the mining process because the *CMine* algorithm is based on candidate generation and test approach. From the mining process of *ECPPG* and *CPPG*, one can observe that the CPs {b}, {b, a}, {b, c}, {b,e}, {a, d}, {a, e}, {a, d, e}, {c, d}, {c ,e}, {c, d, e} and {d ,e} are generated without any database projections and scans in *ECPPG* whereas *CPPG* requires database projections and scans for the same. This demonstrates the importance of using the notion of *minimal CP* along with non-overlap database projections in *ECPPG*.

We prove that *ECPPG* is complete. Let α be a length-l non-overlap pattern and $\{\beta_1, \beta_2, ..., \beta_m\}$ be the set of all length $l+1$ non-overlap patterns having prefix α. The complete set of the non-overlap patterns having prefix α, except for α can be divided into m disjoint subsets. The j^{th} subset $(1 \leq j \leq m)$ is the set of non-overlap patterns having prefix β_j. From this we can say *ECPPG* partitions the problem recursively. That is, each subset of non-overlap pattern can be further divided when necessary. This forms a divide and conquer framework. Using this framework it mines all non-overlap patterns that exists in the transactional database using all the frequent items in the transactional database as non-overlap 1-patterns and grows the non-overlap patterns recursively.

Also the *ECPPG* method correctly identifies the non-overlap patterns. Let α and β be two non-overlap patterns in a database D such that α is a prefix of β. We can easily see that $NOP(D) \mid_{\beta} = (NOP(D) \mid_{\alpha}) \mid_{\beta}$. From this we can say that *ECPPG* mines the set of non-overlap patterns correctly by largely reducing the search space. Now we discuss the advantages of the algorithm as follows.

As compared to *CMine*, *ECPPG* only grows longer non-overlap patterns from the shorter non-overlap ones. The *ECPPG* approach does not generate or test any candidate non-overlap pattern non-existent in a non-overlap projected database. *ECPPG* searches a much smaller space compared to *CMine*, which generates and test a substantial number of candidate non-overlap patterns. Also in *ECPPG*, non-overlap projected databases keep shrinking. The non-overlap projected database is smaller than the original database because only the non-overlap projections of a particular non-overlap pattern are projected into the non-overlap projected database.

5 Experimental Results

We have conducted the experiments on different data sets and analysed how the number of CPs vary with the values of *CS* and *OR*. We have compared the performance of *CMine*, *CPPG* and *ECPPG*. We have also shown the usefulness of CPs in banner advertisement scenario. The experiments were conducted on the following data sets.

i. The synthetic dataset T40I10D100K is generated by the dataset generator [3]. The dataset contains 100,000 transactions and 941 distinct items.
ii. Mushroom dataset is a dense dataset containing 8,124 transactions and 119 distinct items [8].
iii. BMS-POS dataset contains click stream data of a e-commerce company [8]. The dataset contains 515,597 transactions and 1656 distinct items.
iv. MSNBC dataset contains data from Internet Information Sever (IIS) logs for msnbc.com and news related portions of msn.com for the entire day of September, 28, 1999 [9]. Requests are at the level of page category. The number of categories are 17 and the number of transactions are 989,818.

The *CMine*, *CPPG* and *ECPPG* algorithms were written in Java and run with Windows 7 on a 2.66 GHz machine with 2GB memory. To implement *ECCPG* we have used a pseudo-projection technique [14]. When the database can be held in main memory, instead of constructing a physical projection, one can use pointers referring to the sequences in the database as a pseudo projection. Every projection consists of two pieces of information: pointer to the transaction in the database and the index of the first element in the non-overlap projection. The pseudo projection avoids the copying of non-overlap projections physically.

5.1 Generation of Coverage Patterns

The Figure 2(a) shows the number of patterns (y-axis) generated for BMS-POS dataset for different values of *minCS* (x-axis) while *minRF* and *maxOR* are fixed at 0.075 and 0.7 respectively. The total number of CPs generated decrease with the increase in *minCS*. This is because when *minCS* is low, several patterns with smaller size will satisfy *minCS*. As *minCS* value increases, smaller patterns do not satisfy *minCS*. The same behaviour was observed in Figure 2(b) and Figure 2(c).

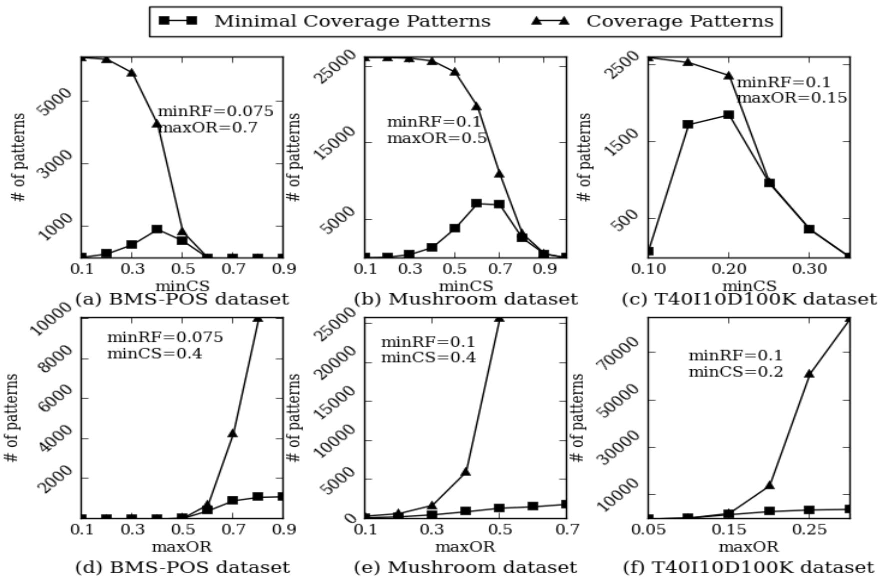


Fig. 2: Number of minimal CPs and total number of CPs.

From Figure 2(a) one can observe that the number of minimal CPs increased till $minCS = 0.4$ and then decreased with increase in $minCS$. This is because once a non-overlap pattern is minimal at one level, further patterns at higher levels are not formed and the number of non-overlap patterns at higher level are more when compared to lower level. Initially, when $minCS$ is low, minimal CPs are formed at lower levels and when $minCS$ is increased till 0.4, minimal CPs are formed at higher levels. Number of minimal CPs are decreased after 0.4 because non-overlap patterns at higher levels are not formed due to fixed $maxOR$. The same behaviour was observed in Figure 2(b) and Figure 2(c).

The Figure 2(d) shows the number of patterns generated (y-axis) for BMS-POS dataset for different values of $maxOR$ (x-axis) while $minRF$ and $minCS$ are fixed at the values 0.075 and 0.4 respectively. One can observe that the total number of CPs and the number of minimal CPs increases with increase in $maxOR$ parameter. This is because more number of items can be grouped to form non-overlap patterns when $maxOR$ is high. The same behaviour was observed in Figure 2(e) and Figure 2(f).

Figure 2 shows the comparison of total number of CPs generated and the total number of minimal CPs generated for three data sets at different $minCS$ and $maxOR$ values. It can be observed that the number of minimal CPs are always significantly less than or equal to the total number of CPs generated. This implies that we are generating 24,357 uninteresting patterns, which eventually shows the importance of mining minimal CPs in extracting the complete set of CPs by reducing database projections and scans. Similar behaviour can be observed with all the datasets.

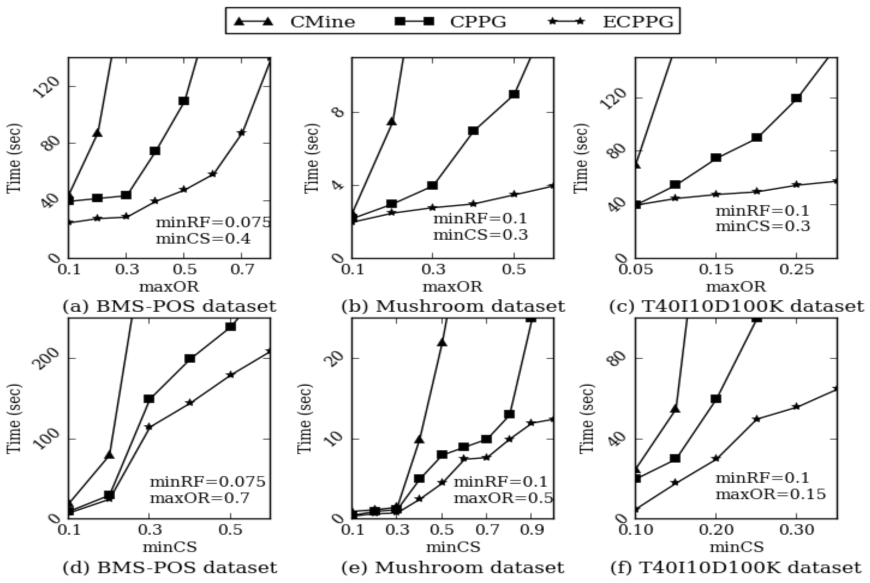


Fig. 3: Performance of CMine, CPPG, and ECPPG on BMS-POS, Mushroom, T40I10D100K datasets at different minCS and maxOR values.

5.2 Performance of CMine, CPPG and ECPPG algorithms

Figure 3 shows the performance of *CMine*, *CPPG*, and *ECPPG* algorithms at different $minCS$ and $maxOR$ values for these datasets. From the experiments, one can observe that performance of *ECPPG* is better than *CMine* and *CPPG* on all the datasets. Figure 3(a) shows the performance of *CPPG*, *ECPPG* and *CMine* with respect to $maxOR$ for $minRF = 0.075$ and $minCS = 0.4$. From this figure, it can be observed that as $maxOR$ is increased the runtime increases. This is because as $maxOR$ is increased more number of non-overlap patterns are generated which eventually increases the runtime of the algorithm. The same behaviour was observed in Figure 3(b) and Figure 3(c).

From Figure 3(d), we can observe that for all algorithms *CPPG*, *ECPPG* and *CMine* as $minCS$ is increased for a fixed $maxOR$ and $minRF$ the runtime is increased, this is because as $minCS$ is increased more number of non-overlap patterns are required to generate a CP which satisfy a particular $minCS$. This makes the algorithm to generate more number of *non-overlap patterns* which eventually increases the runtime of the algorithms. The same behaviour was observed in Figure 3(e) and Figure 3(f).

5.3 Usefulness of Coverage Patterns

In this experiment, we demonstrate the usefulness of CPs in banner advertisement scenario by extracting a sample of CPs from MSNBC dataset. Table 6 shows the CPs generated for $minCS = 0.4$, $maxOR = 0.5$ and $minRF = 0.02$ for MSNBC dataset. The names of web page categories involved in MSNBC are

Table 6: Sample coverage 3-patterns extracted from MSNBC dataset [9].

S.No.	Coverage Pattern	CS	S.No.	Coverage Pattern	CS
1	$\{local, misc, frontpage\}$	0.42	4	$\{on\text{-}air, news, misc\}$	0.40
2	$\{news, health, frontpage\}$	0.43	5	$\{tech, weather, on\text{-}air\}$	0.41
3	$\{tech, opinion, frontpage\}$	0.41	6	$\{sports, misc, opinion\}$	0.43

"frontpage", "news", "tech", "local", "opinion", "on-air", "misc", "weather", "health", "living", "business", "sports", "summary", "bbs" (bulletin board service), "travel", "msn-news", and "main-sports". From Table 6, it can be observed that any of six CPs ensure about 40 percent coverage. That is, with 40 percent coverage requirement, the publisher meet the requirement of multiple advertisers by placing the each advertiser's banner in different set of web pages.

6 Conclusions and Future work

In this paper we have proposed a new data mining pattern called 'coverage pattern (CP)' and presented approaches to extract the same. We have explained how CPs could be useful by considering the issue of banner advertisement placement. The CP is defined using the notions of *CS* and *OR*. Normally, the number of CPs explode with data size. In this paper, we have presented approaches by defining efficient pruning methods. We have discussed level-wise pruning approach which exploits the *sorted closure property* of *overlap ratio*. We have proposed projected pattern growth approach by exploiting the notions of *non-overlap projection* and *minimal CPs*. By conducting experiments on three kinds of datasets, we have shown that the proposed model and methodology can effectively discover CPs. The experiment results also show how the proposed model can help in banner advertisement placement.

As a part of the future work, we are going to investigate how both frequent and coverage pattern knowledge can be used for efficient banner advertisement placement. We are planning to integrate the topical relevance of web pages with the algorithm for finding CPs of a website to make the results of the CPs applicable in the real scenario of banner advertisement placement. We are also exploring how the notion of CPs can be extended to other domains like bio-informatics for extracting potential knowledge patterns.

Acknowledgements We thank anonymous reviewers for their useful comments.

References

1. Agarwal R, Aggarwal C, Prasad V (2001) A tree projection algorithm for generation of frequent item sets. Journal of parallel and Distributed Computing 61(3):350–371

2. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules in large databases. In: VLDB, pp 487–499
3. Agrawal R, Imieliński T, Swami A (1993) Mining association rules between sets of items in large databases. In: Proc. of ACM SIGMOD, pp 207–216
4. Amiri A, Menon S (2003) Efficient scheduling of internet banner advertisements. ACM, vol 3, pp 334–346
5. Bonchi F, Castillo C, Donato D, Gionis A (2008) Topical query decomposition. In: Proc. of ACM SIGKDD, pp 52–60
6. Brin S, Motwani R, Silverstein C (1997) Beyond market baskets: generalizing association rules to correlations. In: ACM SIGMOD Record, vol 26, pp 265–276
7. Chvatal V (1979) A greedy heuristic for the set-covering problem. Mathematics of operations research pp 233–235
8. Fimi (2010) Frequent itemset mining implementations repository. http://fimi.cs.helsinki.fi/ July 2010
9. Frank A, Asuncion A (2010) UCI machine learning repository
10. Garey MR, Johnson DS, Stockmeyer L (1974) Some simplified np-complete problems. In: Proc. of ACM STOC, pp 47–63
11. Han J, Pei J, Mortazavi-Asl B, Chen Q, Dayal U, Hsu M (2000) Freespan: frequent pattern-projected sequential pattern mining. In: ACM SIGKDD, pp 355–359
12. Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation. In: ACM SIGMOD Record, vol 29, pp 1–12
13. Liu B, Hsu W, Ma Y (1999) Mining association rules with multiple minimum supports. In: KDD, ACM, pp 337–341
14. Pei J, Han J, Mortazavi-Asl B, Wang J, Pinto H, Chen Q, Dayal U, Hsu M (2004) Mining sequential patterns by pattern-growth: The prefixspan approach. TKDE 16(11):1424–1440
15. Srinivas PG, Reddy PK, Sripada B, Kiran RU, Kumar DS (2012) Discovering coverage patterns for banner advertisement placement. In: PAKDD (2), pp 133–144
16. Srinivas PG, Reddy PK, Trinath AV (2013) Cppg: Efficient mining of coverage patterns using projected pattern growth technique. In: Proc. of TAKDD, PAKDD Workshop, Springer, pp 319–329
17. Sripada B, Polepalli K, Rage U (2011) Coverage patterns for efficient banner advertisement placement. In: WWW, ACM, pp 131–132
18. Venetis P, Koutrika G, Garcia-Molina H (2011) On the selection of tags for tag clouds. In: Proc. of ACM international conference on Web search and data mining, WSDM ‘11, pp 835–844