Extracting Support Based *k* most Strongly Correlated Item Pairs in Large Transaction Databases

S Roy[1] and D K Bhattacharyya[2]
[1]Department of Information Technology, North Eastern Hill University,
Umshing, Shillong 793022, Meghalaya, INDIA

[2]Department of Computer Science & Engineering, Tezpur University,
Napaam, 784 028, Assam, INDIA

Abstract

Support confidence framework is misleading in finding statistically meaningful relationships in market basket data. The alternative is to find strongly correlated item pairs from the basket data. However, strongly correlated pairs query suffered from suitable threshold setting problem. To overcome that, top-*k* pairs finding problem has been introduced. Most of the existing techniques are multi-pass and computationally expensive. In this work an efficient technique for finding *k* top most strongly and correlated item pairs from transaction database, without generating any candidate sets has been reported. The proposed technique uses a correlogram matrix to compute support count of all the 1- and 2-itemset in a single scan over the database. From the correlogram matrix the positive correlation values of all the item pairs are computed and top-*k* correlated pairs are extracted. The simplified logic structure makes the implementation of the proposed technique more attractive. We experimented with real and synthetic transaction datasets and compared the performance of the proposed technique with its other counterparts (TAPER, TOP-COP and Tkcp) and found satisfactory.

Keywords: *Association mining, correlation coefficient, correlogram matrix, top-k correlated item pairs.*

1. Introduction

Traditional support and confidence measures [17] are insufficient at filtering out the uninteresting association rules [1]. It has been well observed that item pairs with high support value may not imply statistically highly correlated. Similarly, a highly correlated item pair may exhibit low support value. To tackle this weakness, a correlation analysis can be used to provide an alternative framework for finding statistically interesting relationships. It also helps to improve the understanding of meaning of some association rules. Xiong et. al., introduced the notion of strongly correlated item pairs in their work TAPER [2,15], which retrieves all strongly correlated item pairs from transaction database based on user specified threshold θ. A number of techniques have already been proposed [3, 12, 13, 15, 16] to handle this problem. However, setting up an appropriate value for θ is the most challenging task, which requires a prior knowledge about the data distribution. To address this issue an alternative top-*k* correlated-pairs query problem has been introduced recently to mine *k* topmost strongly correlated pairs instead of computing all strongly correlated pairs based on θ [14]. Top-*k* query could play a major role in answering how sales of a product is associated to sales of other product which in turn may help in designing sales promotions, catalog design and store layout. Besides providing a statistical meaning to the traditional association mining problem, top-*k* query could be helpful in efficient finding of co-citation and term occurrence during document analysis. Current research on computational biology reveals that a simple pair-wise correlation analysis may be helpful in finding new gene-gene relationship [8] which again in turns useful in discovering gene regulatory pathways or gene interaction network. Functional relationship [9, 10] between pairs of genes based on gene expression profiles and their changes in different diseases and conditions may be indicative in determining disease mechanism for diseases like cancer.

Recently, a number of techniques have been proposed to compute top-*k* correlated pairs. Existing techniques require multi passes over the database which is too costly for large transaction database. It would be more effective if one can develop an algorithm, which extracts top-*k* strongly correlated item pairs using single pass over the database and without generating large tree or candidate itemsets.

This paper presents a one pass technique *k*-SCOPE (***k*-S**trongly **CO**rrelated item **P**air **E**xtraction) which extracts top-*k* strongly correlated item pairs in only single scan over the database and without generating any candidate sets. A preliminary version of this work can be found in [7]. *k*-SCOPE uses a correlogram matrix for capturing the support of all the 1- and 2-itemsets. Later, it generates a list of *k* top most strongly correlated item pairs from the matrix. The performance of the proposed technique has been found satisfactory in comparison to other counterparts, in light of several real and synthetic transaction datasets.

The rest of the paper is organized as follows: s*ection 2* reports the background of the work and also discusses some of the related works in *section 3*. The proposed technique is described in s*ection 4 & 5*. *Section 6* shows the performance evaluation of the proposed technique and finally in *section 7*, the concluding remarks are given.

2. Background

Association mining [1] is a well studied problem in data mining. Starting from market basket data analysis, now it spreads its spectrum of applications in different domains like machine learning, soft-computing, computational biology and so on. Standard association mining technique is to extract all subsets of items satisfying minimum support criteria. Unlike traditional association mining, the all-pair-strongly correlated query is to find a statistical relationship between pairs of items from transaction database. The problem can be defined as follows.

Definition 1: Given a user-specified minimum correlation threshold θ and a market basket database with I={I_1 I_2, I_3,…. I_N, } set of *N* distinct items and *T* transactions in database *D* is a subset of *I*, a all-strong-pairs correlation query finds collection of all item pairs *(I_i, I_j)* with ϕ correlations above the threshold θ. Formally, it can be defined as:

$$SC(D,\theta)=\{(I_i,I_j)\mid\forall(I_i,I_j)\subset I, I_i\neq I_j\wedge\phi(I_i,I_j)\geq\theta\} \quad (1)$$

To determine appropriate value of θ, a prior knowledge about data distribution is required. Without specific knowledge about the target data, users will have difficulties in setting the correlation threshold to obtain their required results. If the correlation threshold is set too large, there may be only a small number of results or even no result. In which case, the user may have to guess a smaller threshold and do the mining again, which may or may not give a better result. If the threshold is too small, there may be too many results for the users; too many results can imply an exceedingly long time in the computation, and also extra efforts to screen the answers.

An alternative solution to this problem could be to change the task of mining correlated item pairs under pre-specified threshold to mine top-*k* strongly correlated item pairs, where *k* is the desired number of item pairs that have largest correlation values. The problem of top-k correlated- pairs query problem can be defined as follows:

Definition 2: Given a user-specified k and a market basket database with I={I_1 I_2, I_3,…. I_N } set of *N* distinct items and *T* transactions in database *D* is a subset of *I*, a top-*k* correlated-pairs query finds the ordered list of *k* item pairs with top most ϕ correlations. Formally, it can be defined as:

$$TK(D,k)=\left\{\begin{matrix}(I_i,I_j)...(I_{i+k},I_{j+k})\mid\phi(I_i,I_j)\geq ...\\ \geq\phi(I_{i+k},I_{j+k})\geq\phi(I_{i+k+1},I_{j+k+1})\end{matrix}\right\} \quad (2)$$

Next we discuss Pearson correlation measure in order to compute the correlation coefficient between each item pair.

2.1 Support based Correlation Coefficient ϕ

In statistics, relationships among nominal variables can be analyzed with nominal measures of association such as Pearson's correlation coefficient and measures based on Chi Square [5]. The correlation coefficient [5] is the computational form of Pearson's correlation coefficient for binary variables. An equivalent support measure based ϕ correlation coefficient computation technique has been introduced in [2, 15] to find correlation of item pairs in a transaction database based on their support count. For any two items I_i and I_j in a transaction database, the support based ϕ correlation coefficient can be calculated as:

$$\phi(I_i,I_j)=\frac{Sup(I_i,I_j)-Sup(I_i)*Sup(I_j)}{\sqrt{Sup(I_i)*Sup(I_j)*(1-Sup(I_i))*(1-Sup(I_j))}} \quad (3)$$

where *Sup(I_i)*, *Sup(I_j)* are the support of item I_i and I_j and *Sup(I_i, I_j)* is the joint support count of (I_i, I_j).

Adopting the support measures, used in traditional association mining technique for computing correlation coefficient ϕ, the task of top-*k* correlated-pairs finding from transactional database is to generate a sorted list of *k* pairs in the order of ϕ from the database. An illustration of the top-*k* correlated-pairs query problem has been depicted in *Fig 1*. The input to this problem is a market basket database containing 8 transactions and 6 items. The value of *k* is set to 7 as input. Since the database has six items, there are ${}^6C_2$ = 15 item pair for which correlation coefficient ϕ is to be calculated. To compute ϕ (4,5) using equation (3), we need the single element support *Sup*(4)=4/8 and *Sup*(5)=3/8 and joint support *Sup*(4,5)=3/8, in order to compute exact correlation ϕ(4,5)=0.77. Finally the pairs are ordered based on the value of ϕ and a list of *k* sorted pairs are generated as output of the problem.

Top-k Correlated-Pairs Query

Input:
a) Market Basket

TID	Items
1	1,2,4,5,6
2	2,4
3	2,3,6
4	1,2,4,5
5	1,3,6
6	2,3
7	1,3
8	1,2,3,4,5

b) *k* =7

Pair	Support	Corr
{1,2}	0.37	-0.44
{1,3}	0.37	-0.66
{1,4}	0.37	0.25
{1,5}	0.37	0.6
{1,6}	0.25	0.06
{2,3}	0.37	-0.44
{2,4}	0.5	0.57
{2,5}	0.37	0.44
{2,6}	0.25	-0.14
{3,4}	0.12	-0.77
{3,5}	0.12	-0.46
{3,6}	0.25	0.06
{4,5}	0.37	0.77
{4,6}	0.12	-0.25
{5,6}	0.12	-0.06

Output

{4,5}
{1,5}
{2,4}
{2,5}
{1,4}
{1,6}
{3,6}

Fig 1. Illustration of top-*k* correlated pairs query problem

Next we discuss and analyze few of the promising techniques on top-*k* strongly correlated item pair findings.

3. Related Works

Extraction of top-*k* correlated-pair from large transaction database has gained considerable interest very recently. Top-*k* problem is basically an alternative solution for the all-pair strongly correlated pairs query problem. A very few techniques have been proposed so far to address the problem of answering top-*k* strongly correlated pairs query. A brief discussion on different key techniques has been presented below.

3.1 TAPER

TAPER [15] is a candidate generation based technique for finding all strongly correlated item pairs. It consists of two steps: *filtering* and *refinement*. In filtering step, it applies two pruning techniques. The first technique uses an upper bound of the ϕ correlation coefficient as a coarse filter. The upper bound *upper(ϕ(X,Y))* of ϕ correlation coefficient for {X, Y} as follows:

$$\phi(X,Y) \leq upper(\phi(X,Y)) = \sqrt{\frac{Sup\ (Y)}{Sup\ (X)}}\sqrt{\frac{1 - Sup\ (X)}{1 - Sup\ (Y)}}$$

If the upper bound of the ϕ correlation coefficient for an item pair is less than the user-specified correlation threshold θ, right way the item pair is pruned. In order to minimize the effort for computation of upper bound of all possible item pairs, TAPER applies second pruning technique based on the conditional monotone property (1-D) of the upper bound of the ϕ correlation coefficient. For an item pair $\{X,Y\}$, if upper bound is less than θ, then all the item pairs involving item X and the rest target items having support less than Y will also give upper bound less than θ.

In other way, for item pair $\{X,Y\}$, if sup(X)>sup(Y) and fix item A, the upper bound *upper(ϕ(X,Y)),* is monotone decreasing with decreasing support of item Y. Based on that 1-D monotone property, straightway one can avoid such computation of upper bound for other items. In the refinement step, TAPER computes the exact correlation for each surviving pair and retrieves the pairs with correlations above θ.

Discussion

TAPER is candidate generation based, all strongly correlated item pair finding technique. It is well understood that TAPER in its original form cannot find top-*k* item pairs. TAPER suffers from drawbacks of candidate generation step. It is well known that, in comparison to single element item sets, usually the two element candidate sets are huge. The upper bound based pruning technique is very effective in eliminating such large number of item pairs during candidate generation phase. However, when database contains large number of items and transactions, even testing those remaining candidate pairs is quite expensive.

3.2 TOP-COP

TOP-COP [11] is an upper bound based algorithm for finding top-*k* strongly correlated item pairs and extended version of TAPER. TOP-COP exploits a 2-D monotone property of the upper bound of ϕ correlation coefficient for pruning non-potential item pairs i.e. pairs which do not satisfy the correlation threshold θ. The 2-D monotone property is as follows:

For a pair of items (X,Y), if *Sup(X)>Sup(Y)* and fix item Y, the *upper(ϕ(X,Y))* is monotone increasing with decreasing support of item X. Based on the 2-D monotone property a diagonal traversal technique, combined with a refine-and filter strategy has been used to efficiently mine top-k strongly correlated pairs.

Discussion

Like TAPER, TOP-COP is also a candidate generation based technique. The 1-D monotony property, used in TAPER provides a one dimensional pruning window for eliminating non-potential item pairs. Moving one step further, TOP-COP exploits the 2-D monotone property, which helps further in eliminating non-potential pairs from two dimensions instead of one dimension. Compare to 1-D monotone based pruning, the 2-D pruning technique is more effective in eliminating large number of item pairs during candidate generation phase. Like TAPER, TOP-COP also starts with sorted list of items based on support in non-increasing order, which need a scanning of the database once for creating such list. Since it is candidate generation based technique and having structural similarity with TAPER, it also suffers from the drawbacks of expensive testing of remaining candidates after pruning and filtering steps.

3.3 Tkcp

Tkcp [14], an FP-tree [4] based technique, for finding top-*k* strongly correlated item pair. The top-*k* strongly correlated item pairs are generated without any candidate generation. Tkcp includes two sub processes: (*i*) construction of the FP-Tree, and (*ii*) computation of correlation coefficient of each item pairs using the support count from FP-tree and extraction of all the top-*k* strongly correlated item pairs based on correlation coefficient value ϕ. The efficiency of FP-Tree based algorithm can be justified as follows: (*i*) FP-Tree is a compressed representation of the original database, (*ii*) the algorithm scans the database twice only and (*iii*) the support value of all the item pairs is available in the FP-Tree.

Discussion

Although the algorithm is based on efficient FP-tree data structure, yet it also suffers from the following two significant disadvantages.

(i) Tkcp constructs the entire FP-tree with initial support threshold zero. The time taken to construct such huge *FP-tree* is quite large..

(ii) Moreover, it also requires large space to store the entire FP-Tree in the memory; particularly when the number of items is very large.

The techniques discussed above are either generates a large number of candidates or generates large tree. They also need multiple passes over the entire the database. It will be more expensive when the database contains large numbers of transactions or rows. The next section discusses an efficient one pass top-*k* correlated pairs extraction technique that addresses the shortcomings of the algorithms reported above.

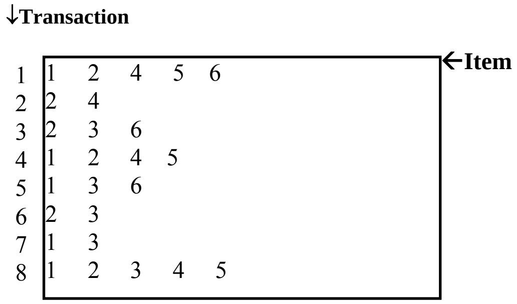
4. Correlogram matrix based technique

The problem of finding support based top-*k* strongly correlated item pairs basically is a problem of computing the support count of 1- and 2- element item sets and uses the count to calculate ϕ (correlation coefficient) of all the item pairs and extracts the *k* most strongly correlated pairs. A correlogram matrix based technique has been discussed in this section for capturing frequency count of 1- and 2-element item sets for finding support based top-*k* strongly correlated item pairs from transaction database using single scan over the entire database and without generating any candidates.

Next we provide the background of the technique.

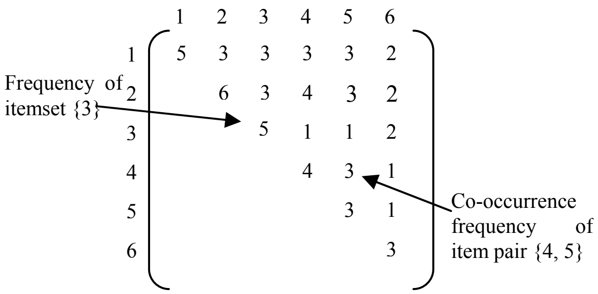
4.1 Correlogram Matrix

A co-occurrence frequency matrix of size: $n \times (n+1)/2$, where, n is the number of items present in the database. Each cell of the matrix contains the frequency of co-occurrence of item pairs. Item pairs are specified by row index and column index of the cell. For example, to specify the frequency of co-occurrence of item pair {4, 5}, corresponding to sample market basket dataset depicted in *Fig* 2 (a), the content of the cell (4,5) in the matrix (see *Fig 2(b)*) with an index of row 4 and column 5 will indicate the co-occurrence frequency of the item pairs {4, 5}.


↓Transaction
←Item

1	1	2	4	5	6
2	2	4			
3	2	3	6		
4	1	2	4	5	
5	1	3	6		
6	2	3			
7	1	3			
8	1	2	3	4	5

Fig 2. (a) Sample market basket



$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 5 & 3 & 3 & 3 & 3 & 2 \\ 2 & & 6 & 3 & 4 & 3 & 2 \\ 3 & & & 5 & 1 & 1 & 2 \\ 4 & & & & 4 & 3 & 1 \\ 5 & & & & & 3 & 1 \\ 6 & & & & & & 3 \end{array}$$

Fig 2. (b) Sample correlogram matrix

On the other hand, the diagonal cells having same indices i.e. row and column indices are same, specifies the occurrence frequency of the 1- item set. In *Fig 2.(b)*, the cell (3, 3) indicates the occurrence frequency of the single itemset {3}.

4.2 Construction Correlogram Matrix

To construct the correlogram matrix, we visualize the situation as graph. All the items participated in a particular transaction are considered as nodes. As the items are appeared in the transaction in a lexicographical order, so we can say that they form a directed graph involving all those items as nodes of the graph. All the items are linked by a single link or edge. Thus, there is only one directional path exists between any two nodes. To illustrate the fact let us consider the transaction number four in *Fig 2.(a)*, where item number 1, 2, 4 and 5 participated in the transaction. To count the co-occurrence frequency of all the item pairs participated in a particular transaction; we count the links among all the pairs of nodes and correspondingly increments the content of a cell in the correlogram matrix with index from both the participating nodes. Thus, if we consider the above example, we increment the content of cell (1, 2), (1, 4), (1, 5), (2, 4), (2, 5) and so on. We also increment the count of first node of a pair i.e. during increment of the count for pair (1, 2), we also increment the content of the cell (1, 1). The intension behind such increment is to keep track of the frequency of occurrence of 1-itemset. Thus by following this procedure, one can construct the correlogram matrix by scanning the database once only.

4.3 Extracting top-*k* strongly correlated item pairs

From the correlogram matrix it is very simple to extract the support of 1- and 2-itemsets. Using these support counts, next we computed the correlation coefficient of all the item pairs using equation (3) and created a sorted list of top-*k* item pairs based on their correlation coefficient.

The advantages of this technique can be stated as:

- It is scalable in terms of number of database instance, since it is one pass.
- No requirements for candidate generation;
- Unlike other techniques discussed so far, it requires only one scan over the database;
- It gives a facility of interactive mining, i.e. compute the correlogram matrix once and generate different top-*k* list based on different *k* values.
- Since it is memory based, it can be found very fast, and
- Easy to implement due to its simplified logic structure.

5. *k*-SCOPE: The Algorithm

The stepwise algorithmic representation of the proposed approach (see *Fig* 3) has been presented in this section. The algorithm accepts the market-basket database i.e. *D* and *k* as input and it generates list of top-*k* strongly correlated item pairs, *L*, as output. *Step 1* is dedicated to the first phase of the approach, i.e. construction of the correlogram matrix using single scan of the original database. In *step 2a*, the correlation coefficient of each item pair is computed and in *step 2b*, top most *k* correlated item pairs are extracted and added to the top-*k* list. Top-*k* list *L* is a sorted list (descending order) of item pairs based on the correlation coefficient. For any pair whose correlation coefficient less than the *k*-th pair's correlation coefficients are straightway pruned. Otherwise, it updates the list by eliminating the *k*-th pair and inserting the new pair in its appropriate position in the list. Finally, the algorithm returns top-*k* list *L*.

```
Input  : $D$    // Transaction database
         $k$    // No. of strongly correlated pairs
Output:
      $L$  // Sorted list of $k$ most strongly correlated item pairs

k-SCOPE Algorithm:
    $L = \varnothing$
1.  Generate Correlogram Matrix from $D$
2.  For each item pair $(i, j) \in D$ do
    a. Compute Pearson Correlation coefficient, $\phi(i, j)$ by
       using support count from the correlogram matrix.
    b. If $|L| < k$ then
           $L = L \cup (i, j)$
       Else
           i.  Sort the list $L$ in descending order based on $\phi$
               of each pair.
           ii. If $\exists\, (i, j) \in D$ such that $\phi(i,j) \geq \phi(L[k])$ then
                   Begin
                       $L = L - L[k]$
                       $L = L \cup (i, j)$
                   End
3.  Return L
```

Fig 3. *k*-SCOPE Algorithm

5.1 Analysis of *k*-SCOPE algorithm

In this section, we analyze the proposed technique in terms of *completeness, correctness* and the *computation savings*.

5.1.1 Completeness and Correctness

***Lemma* 1**: *k*-SCOPE is complete i.e. *k*-SCOPE finds top-*k* strongly correlated pairs.

Proof: Since *k*-SCOPE is based on exhaustive search and computes correlation coefficients of all pairs without pruning any item pair, so, *k*-SCOPE extracts *k* top most strongly correlated item pairs based on the value ϕ. This fact ensures that *k*-SCOPE is complete in all respects. □

***Lemma* 2**: *k*-SCOPE is correct i.e. correlation coefficients of the extracted pairs are the *k* top most correlation coefficients.

Proof: The correctness of *k*-SCOPE can be guaranteed by the fact that, *k*-SCOPE calculates exact correlation of each pair present in the database and creates a sorted list (descending order) of item pairs based on the correlation coefficient and discards all those pairs whose correlation coefficient lower than the *k*-th pair's correlation coefficient. □

5.1.2 Space Complexity

We have considered the modified version of TAPER to generate top-*k* strongly correlated pairs, as TAPER in its original form unable to do so.

TAPER and TOP-COP needs memory space for keeping top-*k* list and support count of all items and also to store a huge number of candidates item pairs, depending on the θ upper bound value. TOP-COP maintains a matrix for keeping the pruning status of all item pairs out of *n* item, which in turn occupies memory space of order (n^2). Tkcp creates an entire FP-tree in the memory with initial support threshold zero (0), which is normally very huge when the number of transactions as well as the dimensionality is large and also depends on the number of unique patterns of items in the database. Sometimes it is difficult to construct such tree in the memory. However, *k*-SCOPE always requires a fix memory of size, $n*(n+1)/2$, for *n* dimensional market basket database to construct correlogram matrix and array of size *k* to store top-*k* strongly correlated item pairs. The total space requirement could be:

$$Space_{k\text{-SCOPE}} = O(n*(n+1)/2)+O(k)$$

5.1.3 Time Complexity

The computational cost of *k*-SCOPE consists of two parts: Correlogram matrix construction cost (C_{CM}) and cost for extraction of top-*k* strongly correlated item pairs (C_{EX}).

(a) *Construction of correlogram matrix***:** Let us consider that the database contains *T* number of transactions and maximum *N* number of items in each transaction. So, to scan the database once it requires *(T*N)* times. For storing and updating support count in correlogram matrix with respect to each transaction it requires to find out the all pairs of combinations of the items present in that transaction. Thus, the time requirement for updating the count is N^2. The total time complexity for construction of correlogram matrix could be : $C_{CM} = O(T*N^2)$.

(b) *Extraction of top-k strongly correlated item pairs:* Since for calculating the correlation of each pair, *k*-SCOPE has to traverse the correlogram matrix once, thus the time requirement for computing correlation coefficient of all item pairs with *n* numbers of item, is $O(n*(n+1)/2) \approx O(n^2)$. In order to create the top-*k* list, for each item pair it compares the correlation coefficient (ϕ) of the new pair with (*k-1*)th pair in the list. If ϕ of new pair is greater than the *k*-th pair, then the *k*-th pair is eliminated from the list and new pair is inserted and placed in the list according to the descending order of the ϕ. Thus, for placing a new pair it requires at most *k* times comparison and swapping. Since, the problem is to find *k* top most item pairs out of *n(n-1)/2* item pairs, thus considering worst case scenario, the time requirement for creating list of top *k* item pair can be denoted as:

$$C_{EX} = O(n^2)+O(k*(n*(n-1))/2).$$

Thus the total cost incurred by *k*-SCOPE is:

$$\begin{aligned} COST_{k\text{-SCOPE}} &= C_{CM} + C_{EX} \\ &= O(T*N^2)+ O(n^2)+ O(k*(n*(n-1))/2) \\ &= O(T*N^2)+ O(n^2)+ O(k*n^2) \end{aligned}$$

The computational cost of TOP-COP and TAPER algorithms are almost similar, except the cost of computing the exact correlations for remaining candidates, which may be less in case of TOP-COP, as it prunes more number of non-potential pairs based on 2-D monotone property. The cost of TOP-COP algorithm includes following parameters:

$$COST_{\text{TOP-COP}} = C_{Sort} + C_{Bound} + C_{Exact} + C_{k\text{-list}}$$

Where C_{Sort}, C_{Bound}, C_{Exact} and $C_{k\text{-list}}$ are cost of creating sorted list of items on non-increasing order of support, cost of computing upper bounds, cost of computing exact correlation of remaining pairs and *k*-top list maintenance cost respectively. On elaborating and simplifying the above cost parameters it could become:

$$COST_{\text{TOP-COP}} = O(n \log n) + O(n^2) + O(n^2) + O(k^2)$$

However, the above cost model, does not consider the cost of scanning the database. It requires one scan for creating the initial sorted itemlist and at least another whole scan (when any hash based data structure is used) of the database for computing exact correlation of existing pairs after pruning. On incorporating such cost it would be:

$$COST_{TOP\text{-}COP} = O(T*N)+O(n \log n) + O(T*N)+O(n^2) + O(n^2) + O(k^2)$$
$$\approx 2*O(T*N)+O(n \log n) + 2*O(n^2) + O(k^2)$$

Similarly the cost of *Tkcp* algorithm could be modeled as:

$$COST_{Tkcp} = C_{Sort} + C_{D\_Sort} + C_{FP} + C_{k\text{-}list}$$
$$= (O(T*N)+O(n \log n)) + O(T*N^2) + (O(T*N) + C_{FPTree}) + (O(n)*C_{Cond\_base} + O(P*k^2))$$

where C_{Sort} is the cost of creating initial sorted list of items based on support by one pass of the database, C_{D_Sort} is the cost incurred during sorting the database based on descending order of item support and C_{FP} is the total cost of creating FP-tree. Creation of complete FP-tree requires one complete scan over the database and cost of creating pattern tree (C_{FP_Tree}). Computing correlation of each pair and for maintaining the *k*-top list it requires additional cost C_{Cond_base} for creating of conditional pattern base (P) for each item. As the cost of scanning a database is much larger than the other computational parameters, the computational savings of *k*-SCOPE will be quite large ($O(T*N)$) when the number of records in a transaction database is very high.

6. Performance Evaluation

The performance of *k*-SCOPE is evaluated in comparison to its other counterparts, and tested in light of synthetic as well as real-life datasets. Several synthetic datasets generated according to the specifications given in *Table 1* for testing purpose. The synthetic datasets were created with the data generator in ARMiner software (http://www.cs.umb.edu/~laur/ARMiner/) which also follows the basic spirit of well-known IBM synthetic data generator for association rule mining. The size of the data (i.e. number of transactions), the number of items, the average size and number of unique patterns in transactions are the major parameters in the synthesized data generation.

We also used real life *Mushroom dataset* from UCI ML repository(*http://www.ics.uci.edu/~mlearn/MLRRepository.html*), *Pumsb* from IBM. *Pumsb* is often used as the benchmark for evaluating the performance of association mining algorithms on dense data sets. The *Pumsb* data set corresponds to binarized versions of a census data set from IBM (available at http://fimi.cs.helsinki.fi/data/) is used for the experiments (see *Table 2*).

We used Java for implementation of *k*-SCOPE and modified TAPER. We used code of TOP-COP as provided by the original author. Since performance of Tkcp highly dependent on FP-tree implementation, we adopted third party FP-tree implementation [6] to avoid implementation bias.

Since, TAPER in its original form cannot generate top-*k* list, we modified TAPER, so that it can generate such top-*k* strongly correlated item pair list. As TAPER is dependent on the correlation threshold θ, in order to generate same result by modified TAPER we set the θ with the correlation coefficient of the *k*-th pair from the top-*k* list generated by *k*-SCOPE. Ideal θ values for modified TAPER for different datasets are presented in *Table 3*.

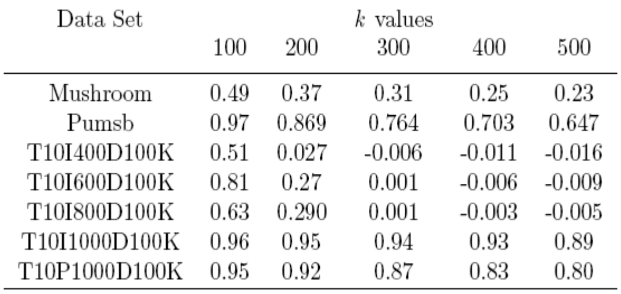
Table 1 : Synthetic Dataset

Data Set	No of Transaction	No of Items	Avg size of transaction	No of Patterns
T10I400D100K	100,000	400	10	20
T10I600D100K	100,000	600	10	20
T10I800D100K	100,000	800	10	20
T10I1000D100K	100,000	1000	10	20
T10P1000D100K	100,000	1000	10	1000

Table 2 : Real Dataset

Data Set (Binarized)	No of Transaction	No of Items	Source
Mushroom	8124	128	UCI
Pumsb	49046	2113	IBM Almaden

Table 3: Suitable θ value for different dataset



Data Set	*k* values				
	100	200	300	400	500
Mushroom	0.49	0.37	0.31	0.25	0.23
Pumsb	0.97	0.869	0.764	0.703	0.647
T10I400D100K	0.51	0.027	-0.006	-0.011	-0.016
T10I600D100K	0.81	0.27	0.001	-0.006	-0.009
T10I800D100K	0.63	0.290	0.001	-0.003	-0.005
T10I1000D100K	0.96	0.95	0.94	0.93	0.89
T10P1000D100K	0.95	0.92	0.87	0.83	0.80

Next we present our experimental results over the different real and synthetic datasets.

6.1 Experimental Results

Performances of all the algorithms were compared in terms of execution time for different values of *k*. In case of Tkcp, it consumes maximum time compared to other two techniques, since Tkcp generates entire FP-Tree with initial minimum support value zero. As a result it has been found too time consuming to construct and parse the tree in the memory, especially when the number of items is large. It has been observed that Tkcp cannot perform when number of items is more then 1000. In case of T10P1000D100K dataset, Tkcp failed to mine, due to large number of items and unique pattern. From the performance graph it could be easily observed that modified TAPER (TAPER in graph is represents the modified TAPER) performs much better then TOP-COP, even though TOP-COP is an improved and modified version of TAPER. It happens because of use of efficient hash data structure, which is lacking in original TOP-COP implementation. This is further indicates that performance of correlation mining algorithms could be improved through efficient implementation. However, in all cases, *k*-SCOPE exhibits comparatively better performance than modified TAPER, TOP-COP and Tkcp (see *Fig 4 & 5*). With the decrease in θ, the running time requirements of modified TAPER also increases, since low θ value generates large number of candidate sets. Similarly, TOP-COP also exhibits an exponential performance graph in increasing number of items. But *k*-SCOPE and Tkcp maintains stable running time in different datasets, since both algorithms are independent of θ. It further confirms the fact that like Tkcp, *k*-SCOPE is also robust with respect to input parameter *k*.

6.4 Scalability of *k*-SCOPE

The scalability of *k*-SCOPE algorithm with respect to the number of transactions and number of items in the databases is shown in *Fig 6*. In this experiment, we used ARMiner to generate synthetic datasets. We generated four data sets with the number of transactions ranges from 1,00,000 to 5,00,000, keeping number of items as 1000 to test the scalability in terms of increasing number of transactions. In order to test the scalability in terms of number of items, we generated another five transaction datasets with number of items ranges from 15000 to 10,000 keeping number of transaction equal to 1,00,000. It has been observed that, the execution time increases linearly with the increase of the number of transaction and items at several different top-k values. Fig 6 reports the scalability test results for *k* equals to 500 and 2500. From the graph it is clear that the performance of *k*-SCOPE is insensitive to different *k* values .Thus *k*-SCOPE is highly robust in handling large transaction databases for different values of *k*.

7. Conclusion

An efficient correlogram matrix based technique, *k*-SCOPE, has been presented in this paper to extract top-*k* strongly correlated item pairs from large transaction database. The technique is capable to generate top *k* strongly correlated item pairs just by scanning the database once. It captures the support count of all the items pairs and stored it in the correlogram matrix. Later, stored support counts are used to compute correlation coefficient all the item pairs and extract *k* most strongly correlated pairs. Advantage of *k*-SCOPE is that, it also supports interactive mining. Experiments have shown that *k*-SCOPE is faster than other counter- parts.

References

[1] J Han and M. Kamber, Data mining Concepts and Technique, Morgan Kaufmann Publishers, San Francisco, CA, 2006.

[2] H Xiong, S Shekhar, S., P-N Tan, V Kumar, Exploiting a Support-based Upper Bound of Pearson's Correlation Coefficient for Efficiently Identifying Strongly Correlated Pairs. Proceedings of SIGKDD'04, pp 334-343, 2004

[3] Z He, S Deng, and X Xu, An FP-Tree Based Approach for Mining All Strongly Correlated Item Pairs, LNAI 3801, pp. 735 – 740, 2005

[4] Han, J., Pei, J., Yin, J, Mining Frequent Patterns without Candidate Generation, Proceedings of SIGMOD'00, pp 1-12, 2000.

[5] T Henry, Reynolds. The Analysis of Cross-classifications. The Free Press, New York, 1977

[6] C. Borgelt, An implementation of the fp-growth algorithm. Workshop Open Source Data Mining Software (OSDM'05, Chicago, IL), pp. 1-5, 2005.

[7] S Roy and D K Bhattacharyya, Efficient Mining of Top-K Strongly Correlated Item Pairs using One Pass Technique,Proc. of ADCOM,pp. 416-41, IEEE 2008.

[8] W P Kuo et.al, Functional Relationships Between Gene Pairs in Oral Squamous Cell Carcinoma, Proc. of AMIA Symposium, pp.371-375, 2003

[9] K Donna, Slonim, From patterns to pathways: gene expression data analysis comes of age, Nature Genetics Supplement, vol 32, pp. 502-508, 2002

[10] A.J. Butte & I S Kohane, Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements, Pac. Symp. Bio-comput.418429,2000.

[11] H Xiong et.al, Top-k ϕ- Correlation Computation, INFORMS Journal on Computing, Vol. 20, No. 4, pp. 539552,Fall 2008

[12] L Jiang et. al., Tight Correlated Item Sets and Their Efficient Discovery, LNCS, Vol 4505, Springer Verlag 2007

[13] S Li, L Robert, L S Dong, Efficient mining of strongly correlated item pairs, Proc. of SPIE, the International Society for Optical Engineering,ISSN 0277-786X.

[14] Z He, X Xu, X Deng, Mining top-k strongly correlated item pairs without minimum correlation threshold, Intl. Journal of Knowledge-based and Intelligent Engineering Systems 10, IOS Press, pp 105-112, 2006.

[15] H Xiong et.al.(2006), TAPER: A Two-Step Approach for All-Strong-Pairs Correlation Query in Large Databases, IEEE Trans. On Knowledge and Data Engineering, vol. 18 no. 4, pp. 493-508.

[16] S Roy and D K Bhattacharyya, SCOPE: An Efficient One Pass Approach to find strongly Correlated Item Pairs, Proc. of ICIT08,pp. 123-126, IEEE CS Press,2008.

[17] R Agrawal , T. Imielinski, A. N. Swami. 1993. Mining association rules between sets of items in large databases. Proc. ACM SIGMOD Internat. Conf. Management Data, ACM Press, New York, 207–216.

Swarup Roy did his M.Tech. in Information Technology and pursuing his Ph.D in Comp Sc & Engg. from Tezpur University. Presently he is an Assistant Professor in the department of Information Technology at North Eastern Hill University, Shillong. He was awarded with gold medal for securing first potion in M.Tech. His research interest includes Data mining and Computational Biology. S Roy has published a number of papers in different refereed Int'l. Conf. Proc. / Journal and also authored a book. He is a reviewer of an international journal.

Dhruba K Bhattacharyya did his Ph D in Computer Science in the field of Information security and Error Correction & Detection in 1999 from Tezpur University. Presently he is Professor and Head in the department of Computer Science and Engineering of Tezpur University. Prof Bhattacharyya has more than 100 research publications in the Int'l. Journals and refereed Conference Proceedings and also he has written/co-edited five books. His research interests include Data Mining, Network Intrusion Detection and Content based Image Retrievals. He is a PC/Advisory Committee member of several International Conferences.

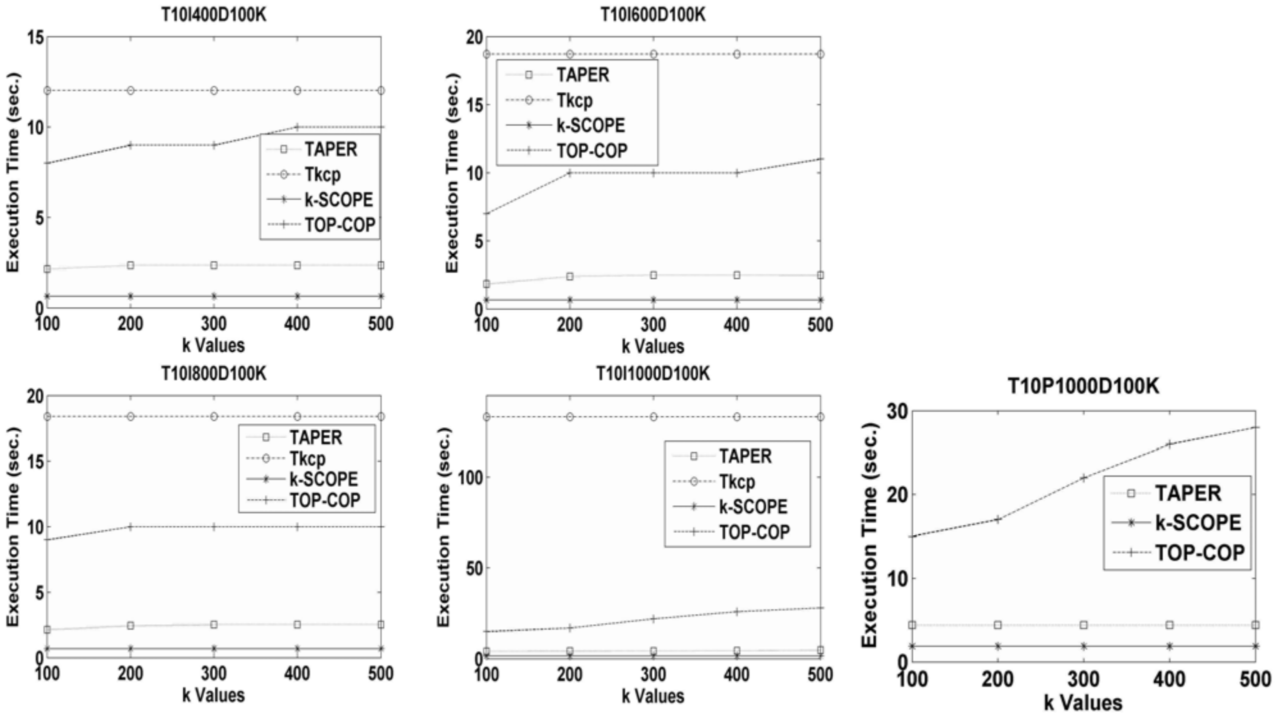


Fig 4: Execution time comparison on Synthetic dataset

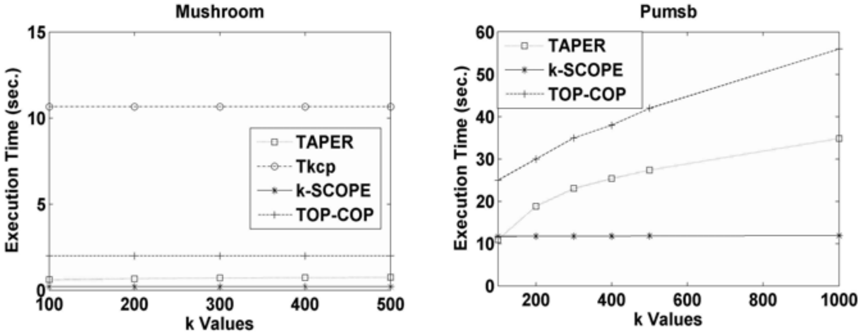


Fig 5: Execution time comparison on Real dataset

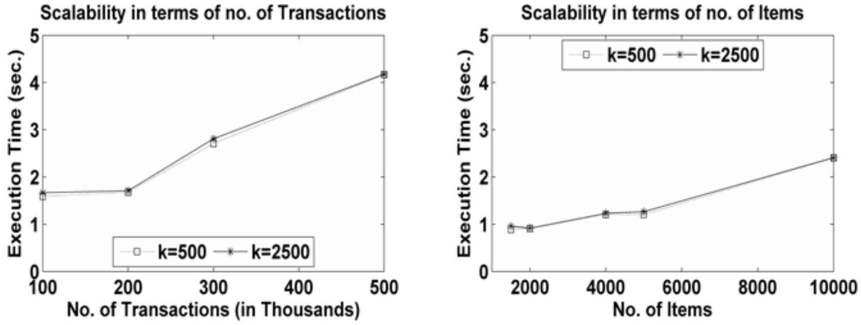


Fig 6: Scalability of *k*-SCOPE algorithm.