**Indian Journal of Science and Technology,** *Vol 9(17), DOI: 10.17485/ijst/2016/v9i17/93106, May 2016*

*ISSN (Print) : 0974-6846*
*ISSN (Online) : 0974-5645*

# Mining Popular Patterns from Multidimensional Database

**G. Vijay Kumar*, T. Krishna Chaitanya andM. Pratap**

Department of Electronics and Computer Engineering, School of Computing, K L University, Vaddeswaram, Guntur - 522502, Andhra Pradesh, India; gvijay_73@kluniversity.in, tkchaitu.123@gmail.com, pratapchowdarymullapudi@gmail.com

## Abstract

**Objectives:** To extract popular patterns from multidimensional database. Design an efficient algorithm to find frequency and maximum transaction length of a pattern for mining popular patterns from multidimensional database. **Analysis:** Earlier to mine required patterns from database Apriori algorithm is used. After the frequent patterns, they have been extended to a many interesting patterns. However, to mine required patterns from a multidimensional database FP-growth algorithm have been extensively in use. Here we have implemented pop-growth technique to mine popular patterns from multidimensional database based on their popularity values. **Findings:** In this paper, we studied about popular patterns which give the popularity of each item or events in the entire database. Whereas Apriori and FP-growth algorithm depends upon the support or frequency measure of a itemset. Therefore, to obtain required patterns using these techniques one to mine FP-growth tree recursively this involves in more time consumption. In this paper, we have implemented a mining technique, which is prominent for multi-dimensional popular patterns. It overcomes the limitations of existing mining techniques. It implements lazy pruning technique and exhibits downward closure property. **Improvement:** Till date, mining of popular patterns based on their popularity measure is implemented only on transactional database and incremental database. But we have implemented this technique on a dynamic multidimensional database in which popular patterns can be mined in two dimensions. It involves in two steps: 1. The Pop-tree structure, which catches the vital information for the mining process of popular patterns. 2. The Pop-tree development calculates for mining popular patterns.

**Keywords:** Down Ward Closure Property, Lazy Pruning, Multidimensional Database, Popular Patterns, Popularity, Support, Pop Tree

## 1. Introduction

Mining required patterns from a database is the programmed examination of expansive amounts of information to remove already obscure, fascinating itemsets or patterns which includes information records, inconsistency recognition, and conditions. Such examples, which were found by mining, can be a sort of outline of the information, and can be utilized for further examination. Data mining is seen as a necessity, which transforms data into business intelligence giving information to various organizations. It is currently used in a wide range of profiling practices, such as marketing, surveillance, crime detection, education and social systems. After the frequent patterns are introduced, they have been extended to a variety of interesting patterns to keep away from the applicant era and-test methods of the Apriori calculation, a tree-based calculation named FP-growth was executed to manufacture a FP-tree, primarily to get the substance of value-based database so that incessant examples are recursively mined from the FP-tree with a confined test-just approach. The mining of these examples depend on the bolster/recurrence tally. While bolster/recurrence is a valuable measure, support-based continuous example mining may not be proficient to get data, for example, relationship, regularity, periodicity, frequency among examples in a TDB. The popular pattern mining techniques are focused on

*Author for correspondence

mining knowledge or necessary information at single concept levels i.e., it can be basic or high concept level. However, it is often desirable to discover knowledge at multiple concept levels.

In[1] provides a brief introduction over a study of Data Mining and Social Network Analysis. A new data model had been developed using new agent based intelligent systems in the form of 3d cubes for different classifications[2]. That helps in developing different theories that aims at extracting information using data mining technique[3]. Frequent patterns have been extended to popularity patterns which overcome the limitations of existing patterns. It captures the popularity values of each itemsets in a database[4]. Apriori Hybrid is the combination of best features of both Apriori and Apriori tid algorithms that discovers association rules between datasets in large databases[5]. Regular patterns which can also be mined in vertical format from a transactional database using transaction id's and this technique is further extended to mine parallel and distributed frequent regular patterns in large databases[6,7]. Maximal Regular Frequent Itemset Mining has been introduced using a pair of Transaction-ids instead of using itemsets[8]. And to store information about these patterns without candidate generation a novel frequent pattern tree is developed[9]. In addition to these statistical, machine learnable and graphical techniques and utility mining can also be implemented for extracting information which is very much useful in real time applications[10]. Frequently occurring periodic patterns can be mined with a predefined gap as a requirement[11]. Hyperclique patterns overcome this disadvantage of inefficient mining of interesting patterns at low levels of threshold[12]. Multidimensional database approach over comes the inefficient data storage in relational databases and can mine all possible association rules between item sets in a transactional database[13,14]. An efficient algorithm is proposed for mining regular patterns from a database by considering temporal regularity of patterns, which uses variance interval time between patterns and RFPID for incremental database applications[15,16].

## 2. Materials and Methods

### 2.1 Problem Description

A pattern or an item-set, p, is one dimension Dj or one item Ak, or a set of conjunctive items and dimensions Di^…^Dj^Ak^…^Al , where Ai^…^Aj∈ τ . The support of an itemset is the number of transactions that contain it in the total transactions. An itemset is said to be frequent if its support is greater than the minimum threshold. The problem of mining multi-dimensional database is solved by implementing the proposed algorithm. The proposed algorithm is able to discover associations between items and dimensions as well as associations among items. The proposed algorithm improves the effectiveness of frequent pattern mining by pushing various support constraints inside mining process. A Multi Dimensional database is clearly shown in Table 1.

Formula 1. he popularity of an itemset X P op(X, tj) in a transaction tj implies its degree of X in tj. For simplicity, we compute the membership degree based on the difference between the length |tj| and the size of itemset |X|:

$$P\ op(X, tj) = |tj| - |X|.$$

Formula 2. P op(X, tmaxT L(X)) of an itemset X in transaction tmaxT L(X) implies the degree of X in tmaxT L, where tmaxT L(X) is the transaction having the maximum length in DBX:

$$P\ op(X, tmaxT\ L(X)) = \max_{tj \in DBX} |tj| - |X|.$$

Formula 3. The popularity P op(X) of an itemset X in the TDB is measured as

**Table 1.** Multi dimensional database

| S.NO | STORE | TID | TRANSACTION ITEMS |
|---|---|---|---|
| 1. | BC | 001 | TV, COMPUTER, PRINTER, FRIDGE, AC, WASHING MACHINE |
| 2. | ON | 001 | PRINTER, AC, WASHING MACHINE |
| 3. | BC | 002 | TV, COMPUTER, FRIDGE, PRINTER |
| 4. | ON | 003 | COMPUTER, TV, PRINTER, FRIDGE |
| 5. | BC | 003 | TV, COMPUTER, AC, PRINTER |
| 6. | ON | 003 | TV. COMPUTER |
| 7 | BC | 004 | COMPUTER, PRINTER, WASHING MACHINE |
| 8. | ON | 004 | TV, COMPUTER, PRINTER |
| 9. | BC | 005 | TV, PRINTER |
| 10. | ON | 005 | PRINTER, TV, AC, COMPUTER |

P op(X) = 1/ |DBX| tj∈ DBX P op(X, tj).

Formula 4. P op(X) ≥ min-pop).
Formula 5.

P op(X) = 1/ |DBX| tj∈ DBX P op(X, tj )
= 1 /|DBX| tj∈ DBX (|tj |−|X|)
= (sumT L(X)/ |DBX|) − |X|.

## 2.2 Proposed Algorithm

In the initial step, databases are examined once to get the check of each and every thing and each and every measurement. The incessant 1-things or frequent1-measurements are those whose tallies pass their relating bolster limit. In the second step, we get data as{x: support(x), maxT L(x), pop(x)} for each domain items. The items with popularity less than support (less popular) are not eliminated as in frequent pattern instead their super patterns are taken into considerations and checked whether they are popular or not. Henceforth, we can't erase them without performing the super-design prevalence check. Here things or measurements can show up in the pop-tree the length of their tallies pass their comparing edge. Along these lines, a mainstream example which incorporates the entire scientific classification data around a thing is additionally fascinating to the client. At last, all prominent examples are produced by utilizing pop-tree recursively mines. Finally, all popular patterns are generated by using pop-tree recursively mines. This procedure is continued until all popular patterns for all single dimension domains are generated. Then once again this process is repeated to find global popular patterns on all local single dimension domains mined data.

**Pop Tree Algorithm:**

```
Input: DB, λ, K, N
Output: set of popular patterns.
Procedure:
Step 1:  Let $X_i$ subset (I) be a k-itemset
Step 2:      for(i=1; i<=K, i++)
Step 3:          for each $X_i$
Step 4:          Update Pop(Xi)
Step 5:      Pop(Xi)=(Sum(Xi)/|DBxi|)-|Xi|
Step 6:  If Pop($X_i$)>= λ
Step 7:      Popular
         Else
Step 8:      $Pop^{Ub}(Xi^1)$=max(xi)-|Xi|
Step 9:       for($i^1$=i+1;$i^1$<=K;$i^1$++)
Step 10: If ($Pop^{Ub}(X_i^!)$>=λ)
Step 11:     $X_i$ is popular
             Else
Step 12:     Delete $X_i$
 Step 13: Repeat
              for N-DB
```

Here DB = database
K = maximum order of itemset
N = No. of databases
λ = min-pop value

### *2.2.1 Mining Popular Patterns from Local Store BC*

Database of store location BC is shown in Table 2.

In the initial step, databases are examined once to get the check of each and every thing and each and every measurement. The incessant 1-things or frequent1-measurements are those whose tallies pass their relating bolster limit. In the second step, we get data as

{x: support(x), maxT L(x), pop(x)} for each domain items. The minimum popularity taken here is 3.2 and the items whose count is less than min-pop value are not eliminated as in frequent patterns instead their superset is considered. If its superset value exceeds min-pop value then that item is not eliminated. This property is downward closure property. This process is repeated for frequent n-items.

Mining of popular patterns has been proposed by pop-tree algorithm which consists of two steps.

1. The Pop-tree structure which catches the vital information for mining of popular examples and 2. The Pop-development calculation for mining popular examples.

Previously pop-tree was called as popular pattern tree are built by taking only necessary information from the database by filtering it two times. Here as Pop(X) does not proved downward closure property; the item which is not popular should not be removed from pop-tree as their supersets may be popular. But sumTL(x) satisfies the downward closure property so no need to keep all the

**Table 2.** Database of store BC

| S.NO | TID | TRANSACTION |
|---|---|---|
| 1. | 001 | tv, computer, printer, washing machine, frdge, ac |
| 2. | 002 | tv, computer, fridge, priter |
| 3. | 003 | tv, computer, ac, printer |
| 4. | 004 | computer, printer, washing machine |
| 5. | 005 | tv, printer, washing machine, fridge |

unpopular patterns in the tree. Some of the unpopular patterns can be removed.

Let us see the two conditions

Condition 1: The popularity measure is always less than or equal to its long transactional popularity i.e., $Pop(x) \leq Pop(X, t_{maxTL})$

Condition 2: For $X \leq X^{|}$, $Pop(X^{|})$ cannot exceed $maxTL(x) - |X^{|}|$

From given condition a new equation is generated which provides with upper bound popularity Pop(X')

$$Pop^{UB}(X') = maxTL(X) - |X'|$$

With the above equation we remove all the unpopular patterns which is called super-pattern popularity check.

For construction of pop tree structure the main things to be calculated for each patterns are

is<x: support(x) , maxTL(x), Pop(x) >. Now calculate popularity of each frequent-1 pattern from the above table in order to check which pattern not popular

<tv:4,6,3.75>, <computer:4,17,3.25>, <printer:5,6,3.4>, <fridge:3,6,4>, <ac:3,6,4>, <washing machine:<3,6,3.6>

Here by calculating popularity values of all items we know that all patterns are popular because every pattern exceeds min-pop = 3.Calculating popularity of frequent-2 pattern from above table. Here we verify popularity of each frequent-2 itemset with min-pop value i.e., 3.

By calculating each pattern we get to know that the patterns {tv, computer}, {computer, printer}, {computer, washing machine}, {tv, printer}, {printer, washing machine} are not popular since their Pop(x) value is less than min-pop. But we cannot remove them without calculating the Pop(X').

Calculating the Pop(X') values of the transactions which has less than min pop value.

$$Pop(X') = maxTL(X) - |X'| \quad (|X'|=|X|+1).$$

Pop({tv, computer}')=6-3=3; Pop({computer, printer}') = 6-3=3 ; Pop({tv, printer}') = 6-3=3; Pop({computer, washing machine}') = 6-3=3, Pop({printer, washing machine}') = 6-3=3.

So by calculating Pop(X') of {tv, computer}, {computer, printer}, {computer, washing machine}, {tv, printer}, {printer, washing machine} all satisfy the value of min-pop so all patterns are considered as popular even calculating popularity of frequent-3 and further itemsets popularity of every itemset satisfies min-pop value so all patterns in store BC are popular.

**Pop-Tree Structure:**

After calculating the pop(X') all patterns are placed in descending order and the H-table is placed as <x: support(x), sumTL(x), maxTL(x)> and the pop-tree is drawn by slowly inserting each and every transaction one by one.

The H-table and Pop tree structures of store BC are clearly shown in Figure 1.

### *2.2.2 Mining Popular Patterns from Local Store ON*

The database of store location ON is clearly shown in Table 3.

In the initial step, databases are examined once to get the check of each and every thing and each and every measurement. The incessant 1-things or frequent1-measurements are those whose tallies pass their relating bolster limit. In the second step, we get data as{x: support(x), maxTL(x), pop(x)} for each domain items. The minimum popularity taken here is 3.2 and the items whose count is less than min-pop value are not eliminated as in frequent patterns instead their superset is considered. If its superset value exceeds min-pop value then that item is not eliminated. This property is downward closure property and the same process is repeated here up-to frequent-n items. Then popular patterns are mined globally from individual popular patterns of each store with certain minimum popularity and pop tree is constructed for global popular items. Finally, popular patterns are mined from pop tree using downward closure property.

In constructing the pop tree structure the main things to be calculated for each patterns are

<x: support(x), maxTL(x), Pop(x)>. The min-pop is given by the users. Let us consider that the user given minpop = 3. Now calculate each pattern from the above table in order to check which pattern is not popular

<printer: 3,4,3.6>, <ac: 2,4,3>, <washing machine: 1,4,3>,<computer:4,4,2.25>,<tv:4,4,2.25>,<fridge:1,4,3>. By calculating each pattern we get to know that the

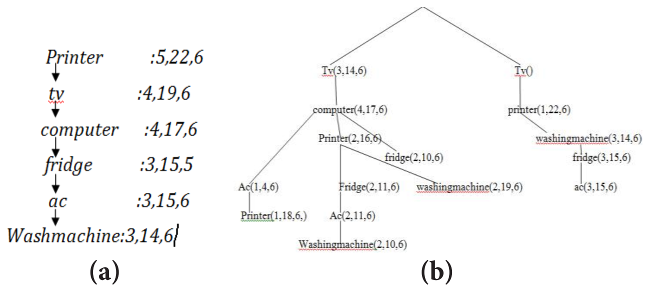


**Figure 1.** **(a)** H-tree of store BC. **(b)** Pop-tree of store BC.

**Table 3.** Database of store ONs

| S.NO | TID | TRANSACTION |
|---|---|---|
| 1. | 001 | printer, ac, washing machine |
| 2. | 002 | computer, tv, printer, fridge |
| 3. | 003 | tv, computer |
| 4. | 004 | tv, computer, printer |
| 5. | 005 | printer, tv, ac, computer |

patterns computer, tv are not popular since their 0Pop(x) value is less than min-pop. But we cannot remove them without calculating the Pop(X').

Pop(X') = maxTL(X) - |X'| ( |X'|=|X|+1)

Pop(computer') = 4-2=2 ; Pop(tv') = 4-2=2

By calculating this, we came to know that both computer and tv has pop value which is less than min-pop. So finally all are popular patterns apart from computer and tv. So by calculating Pop(X') all satisfy the value of min-pop so all patterns are considered as popular even calculating popularity of frequent-3 and further itemsets popularity of every itemset satisfies min-pop value so all patterns in store ON are popular.

After calculating the pop(X') all patterns are placed in descending order and the H-table is placed as <x: support(x), sumTL(x), maxTL(x)> and the pop tree is drawn by slowly inserting each and every transaction one by one .

The H-table and pop tree of store location ON are clearly shown in Figure 2.

### 2.2.3 *Mining Popular Patterns Globally from Two Local Stores*

Review that, to mine mainstream designs, the Pop-development calculation applies two key techniques: 1. Development of a Pop-tree and 2. Mining of prevalent examples from the Pop-tree. The Pop-development finds prevalent examples from the Pop-tree, in which every tree hub catches its event check, absolute exchange length, and most extreme exchange length. The calculation finds well known examples by developing the anticipated database for potential prevalent itemsets and recursively mining their augmentations While building the restrictive database from an anticipated database, we perform a super-design fame check for expansions of any disliked thing, and erase the thing just when it fizzles the check.

This kind of pruning strategy is known as lazy pruning recall that the Pop-development mines repeatedly the anticipated databases of all things in H-table. Before building the anticipated database for a thing x in H-table, we yield the thing as a well known example if its prominence is at any rate min-pop.

The restrictive example base for the {ac}-anticipated database is developed by collecting the substance in the tree way (tv:4,19,6 computer:4,17,6 fridge:3,15,5, washing machine:(3,14,6) (tv:4,19,6 fridge:4,17,6 washingmachine:3,14,6) ,(tv:4,19,6 computer:4,17,6). The header table for DB{ac}, as demonstrated contains all things that happen simultaneously with air conditioning in the Pop-tree. It additionally contains the comparing support, sumT L and maxT L of everything in DB{ac}. We then register the definite ubiquity of everything in DB{ac}. The H-table and store location of global popular patterns are clearly shown in Figure 3.

The restrictive tree for any patterns base of an itemset X might contain two things: 1. Things that are popular in DBX and 2. Things that are not popular in DBX however having conceivably famous super-designs. To discover the patterns those are not popular that having conceivably prevalent super-designs, we apply the lazy pruning system.

In light of, the popularity of things in the H-table of DB{ac} can be processed: P op({tv, ac}) = (15/3) − 2 = 3,

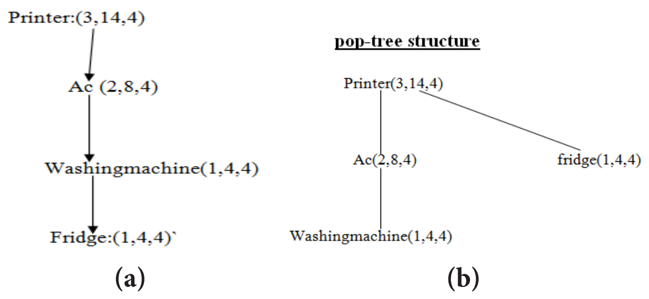


**Figure 2.** **(a)** H-tree of store ON. **(b)** Pop-tree of store ON.

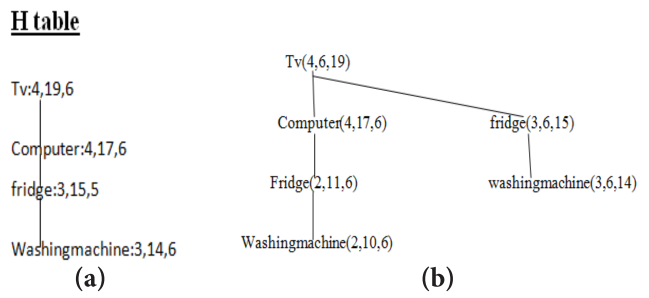


**Figure 3.** **(a)** H-tree of global popular patterns. **(b)** Pop-tree of global popular patterns.

P op({computer, ac}) = (10/2) – 2 = 3, P op({fridge, ac}) = (11/2) –2=3.5, Pop({ac, washing machine}) = 11/2 –2=3.5. All items except are popular together with ac because they satisfy min-pop value = 3.

## 3. Results and Discussions

We have performed experiment on mostly used datasets in mining frequent patterns because the properties of these datasets are known precisely. By using this data we obtained stable results in seconds

The run time (sec) comparison of FP-growth algorithm and proposed algorithm is clearly shown in Table 4.

The variation of runtime with min-pop(%) value is clearly shown in Figure 4.

### 3.1 Runtime

The execution time is the time required for mining popular examples over sorted datasets with which change in min-pop has accounted for. In above figure we speak to the outcomes on one meager dataset and other result on thick dataset.

The reduction in patterns varying with min-pop value for both proposed and existing algorithms are clearly shown in Figure 5.

### 3.2 Compactness of Pop-Tree

In this graph we compared the compactness of pop-tree with pop-tree nodes. With the increase of min-pop the tree size is gradually reduced in T20I3D100K.

**Table 4.** Table showing run time in seconds

| Algorithms | Runtime (in secs) at Different Supports(%) | | | | | | |
|---|---|---|---|---|---|---|---|
| | 0.05 | 0.1 | 0.5 | 1 | 5 | 10 | 15 |
| FP-Growth | 463 | 446 | 154 | 36 | 57 | 8 | 1 |
| Proposed Algorithm | 149 | 66 | 20 | 8 | 3 | 1 | 1 |

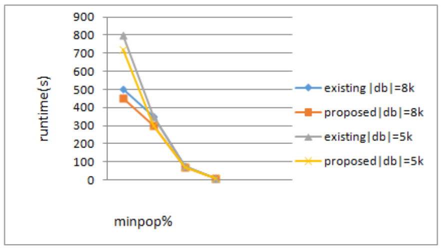
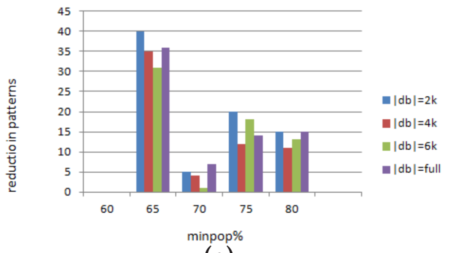
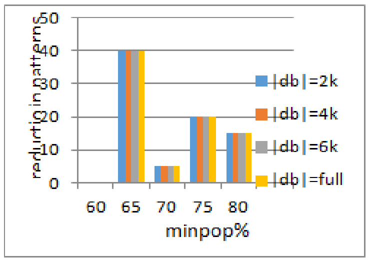


**Figure 4.** Variation of runtime with min-pop(%).



**(a)**



**(b)**

**Figure 5.** **(a)** Reduction in patterns varying with min-pop for proposed algorithm. **(b)** Reduction in patterns varying with min-pop for existing algorithm.

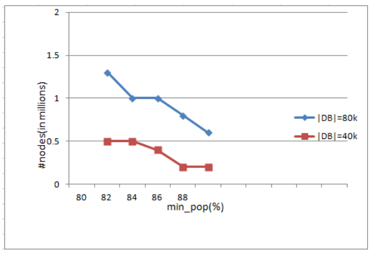


**Figure 6.** Pop-tree node count on T20I3D100K varying with min-pop value.

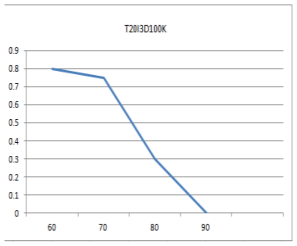


**Figure 7.** Variation of min-pop with mining time

With the increase in database size, the nodes increase in both the datasets.

Pop tree node count varying with minpop value is clearly shown in Figure 6.

### 3.3 Scalability of Pop-Growth

It is observed that as the min-pop value decreases mining time and memory increases in Figure 7.

## 4. Conclusion

In this paper, we perform mining of popular patterns, which catches the prevalence of people, things, or occasions among their associates or gatherings. It implements lazy pruning technique and exhibits download closure property which overcomes the limitations in existing algorithms. We have achieved efficient results in mining popular patterns and have compared runtime and compactness of pop-tree with existing algorithms. Trial results had demonstrated that our pop tree structure is minimal and space productive and our proposed calculation is time proficient.

## 5. References

1. Vedanayaki M. A study of data mining and social network analysis. Indian Journal of Science and Technology. 2014 Nov; 7(S7):185–7.
2. Murugananthan V, Shiva Kumar BL. An adaptive educational data mining technique for mining educational data models in e-learning systems. Indian Journal of Science and Technology. 2016 Jan; 9(3):1–5.
3. Azad N, Ranjbar V, Khani D, Moosavi ST. Information disclosure by data mining approach. Indian Journal of Science and Technology. 2012 Apr; 5(4):1–10.
4. Leung CK-S, Tanbeer SK. Mining popular patterns from Transactional Database. Data ware housing and knowledge discovery. Proceedings of 14th International Conference, DaWaK 2012; Vienna, Austria. 2012 Sep 3-6. p. 291–302.
5. Agarwal R, Srikant R. Fast algorithms for mining association rules. VLDB; 1994. p. 487–99.
6. Kumar GV, Sreedevi M, Kumar NVSP. Mining regular patterns in transactional databases using vertical format. International Journal of Advanced Research in Computer Science. 2011; 2(5):332–5.
7. Kumar GV. Kumari VV. Parallel and distributed frequent–regular pattern mining using vertical format in large databases. ARTCOM 2012, IEEE Xplore, IET; 2012. p. 110–4.
8. Kumar GV, Kumari VV. MaRFI: Maximal regular frequent itemset mining using a pair of transaction-ids. International Journal of Computer Science and Engineering Technology. 2013 Jul; 4(7):**1057–64.**
9. Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. ACM SIGMOD; 2000. p. 1–12.
10. Yao H, Hamilton HJ. Mining itemset utilities from transactional databases. DKE. 2006; 59(3):603–26.
11. Zhang M, Kao B, Cheung DW, Yip KY. Mining periodic patterns with gap requirements from sequences. ACM TKDD. 2007 Aug; 1(2):1–34.
12. Xiong H, Tan P-N, Kumar V. Hyperclique pattern discovery. Data Mining and Knowledge Discovery. 2006; 13(2):219–42.
13. Tanasescu A. The role of multidimensional databases in modern organizations. Economic Insight Trends and Challenges. 2015; 4(67):95–102.
14. Agarwal R, Imielinski T, Swamy A. Mining association rules between sets of items in large databases. ACM SIGMOD Int Conference on Management of Data; 1993. p. 207–16.
15. Rashid MdM, Karim MdR, Jeong BS, Chai HJ. Efficient mining regularly frequent patterns in transactional databases. Springer Lecture Notes in Computer Science. 2012; 7238:258–71.
16. Kumar GV, Kumari VV. Incremental mining for regular frequent patterns in vertical format. International Journal of Engineering and Technology. 2014; 5(2):1–7.