Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 4, April 2014, pg.996 – 1001

RESEARCH ARTICLE

An Approach for Finding Frequent Item Set Done By Comparison Based Technique

Ms. Ankita Parmar[1], Mr. Kamal Sutaria[2], Mr. Krutarth Joshi[3]

[1]Student, Computer Engineering, Gujarat Technological University, India

[2]Assistant Professor, Computer Engineering, Gujarat Technological University, India

[3]Systems Engineer, TATA Consultancy Services, Pune, Maharashtra, India

[1] ankita.parmar2188@gmail.com; [2] kamal.sutaria@gmail.com; [3] joshi.krutarth@gmail.com

Abstract— Frequent pattern mining has been a focused theme in data mining research for over a decade. Abundant literature has been dedicated to this research and tremendous progress has been made, ranging from efficient and scalable algorithms for frequent itemsets mining in transaction databases to numerous research frontiers, such as sequential pattern mining, structured pattern mining, correlation mining, associative classification, and frequent pattern-based clustering, as well as their broad applications. In this paper, we develop a new technique for more efficient pattern mining. Our method find frequent 1-itemset and then uses the heap tree sorting we are generating frequent patterns, so that many. We present efficient techniques to implement the new approach.

Keywords— Data mining; Frequent Pattern mining; Support; Min Heap; Data structure

I. INTRODUCTION

Data mining is the process of analysing the data from different perspectives and going over the useful information – information that can be used to increase revenue, cuts costs, or both [2]. Precisely data mining is the process of finding correlation or patterns among dozens of fields in large relational database. One of the most important data mining applications is that of mining association rules. Data mining has many virtues and vices which involves in many fields. Some of the examples are (1) Bank – to identify patterns that help be used to decide result for loan application to the customer, (2) Satellite research – to identify potential undetected natural resources or to identify disaster situations, (3) Medical fields – to protect the patients from infectious diseases, (4) Market strategy – to predict the profit and loss in purchase.

Agrawal et al. (1993) was the first man who has proposed frequent pattern mining for market basket analysis in the form of association rule mining [1]. In this he analyses customers shopping basket to finding buying habits by finding associations between the different purchased items by customers. In this paper, we perform a high-level overview of frequent pattern mining methods, extensions and applications. We proposed new method in the place of FP-Growth tree using MINHEAP tree, in this tree we are trying to find most frequent item which is already sorted and present in root of heap tree.

II. FP – GROWTH METHOD FOR MINING FREQUENT PATTERNS

An FP-growth (frequent pattern growth) uses an extended prefix-tree (FP-tree) structure to store the database in a compressed form. FP-growth adopts a divide-and-conquer approach to decompose both the mining tasks and the databases. It uses a pattern fragment growth method to avoid the costly process of candidate generation and testing used by Apriori [2].

FP-Growth adopts a divide-and-conquer strategy as follows. First, it compresses the database representing frequent items into a frequent-pattern tree or FP-tree, which retains the itemsets association information. It then divides the compressed database into a set of conditional databases. Each associated with one frequent item and mines each such database separately [3].

Definition 1 (FP-tree) A frequent pattern tree is a tree structure defined below.

A. It consists of one root labelled as "root", a set of item prefix sub-trees as the children of the root, and a frequent-item header table.

B. Each node in the item prefix sub-tree consists of three fields: item-name, count, and node-link, where item-name registers which item this node represents, count registers the number of transactions represented by the portion of the path reaching this node, and node-link links to the next node in the FP-tree carrying the same item-name, or null if there is none.

C. Each entry in the frequent-item header table consists of two fields, (1) item-name and (2) head of node-link, which points to the first node in the FP-tree carrying the item-name.[3]

The actual algorithm is:

Algorithm (FP-tree construction)
Input: A transactional database DB and a minimum support threshold.
Output: Its frequent pattern tree, FP-tree
Method: The FP-tree is constructed in the following steps:

1) Scan the transaction database DB once. Collect the set of frequent items F and their supports. Sort F in support descending order as L, the list of frequent items.

2) Create the root of an FP-tree, T, and label it as "root". For each transaction Trans in DB do the following.

3) Select and sort the frequent items in Trans according to the order of L. Let the sorted frequent item list in Trans be [p | P], where p is the first element and P is the remaining list. Call insert_tree([p | P], T).

4) The function insert_tree([p | P], T) is performed as follows. If T has a child N such that N.item-name = p.item-name, then increment N's count by 1; else create a new node N, and let its count be 1, its parent link be linked to T, and its node-link be linked to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert_tree(P, N) recursively.

Example : Assume we are given a transaction database and a minimal support threshold of 2. First, the supports of all items are computed, all infrequent items are removed from the database and all transactions are reordered according to the support descending order resulting in the example transaction database in table.

TABLE I
AN EXAMPLE PREPROCESSED TRANSACTIONAL DATABASE

Tid	List of Items
100	{Pen, Ink, Rubber}
200	{ Pen, Ink, Pencil }
300	{Pencil, Rubber}
400	{Ink, Pencil}
500	{Register, Ink, Pen}

The FP-tree for this database is shown in Figure 1

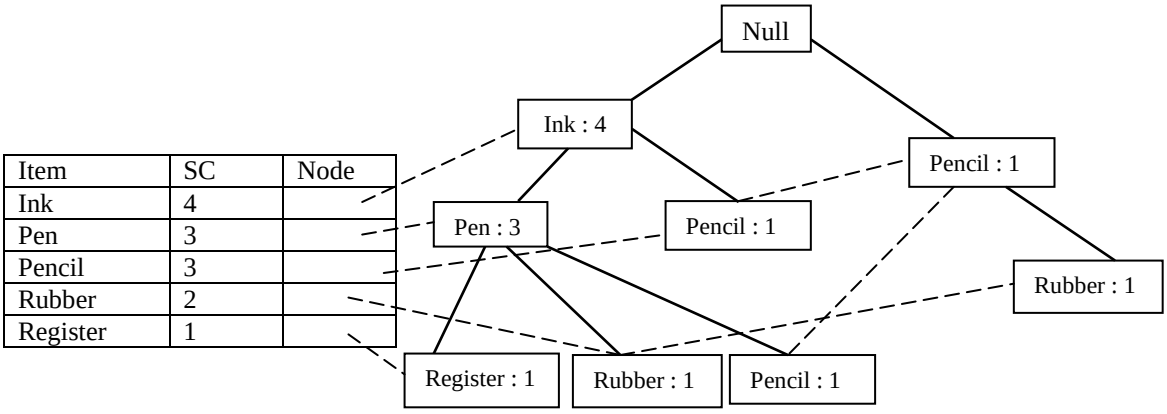


Fig. 1 FP-tree for the above given transactional database

TABLE III
RESULTS OF FP-GROWTH TREE

Item	Conditional Pattern Base	Conditional FP-tree	Frequent Pattern Generates
Register	{Ink , Pen :1 }	-	-
Rubber	{Ink, Pen : 1} {Pencil :1)	-	-
Pencil	{Ink , Pen :1 } {Ink : 1}	{Ink :2}	{Ink , Pencil :2}
Pen	{Ink :3 }	{Ink :3 }	{Ink , Pen :3 }

III. DATA STRUCTURES

The data structures are user defined data types specifically created for the manipulation of data in a predefined manner. There are two types of data structure Linear and Non Linear. Array, stacks, queues and Linked List are Linear Data structure whereas trees and graphs are non linear data structure.

A. Tree

Trees are very flexible, versatile and powerful data structures .A tree is a non linear data structure in which items are arranged in a sorted sequence. It is used to represent Hierarchical relationship existing amongst several data structures.

There are different types of trees like Binary tree, B-Tree, B+Tree ,AVL Tree, Extended Tree , Heap Tree etc. All trees manage data in different types. Here we will explain the functioning of Heap Tree.

B. Heap

The Heap is used in an elegant sorting algorithm called heapsort. Suppose H is a complete binary tree with n elements is maintain in memory using the sequential representation of H. Then H is called the Heap.

Heap is of Two types Max Heap and Min Heap. If root is greater than or equal to their left and right child then it is called Max Heap.If root is less than or equal to their left and right child then it is called Min Heap.

1) Operations on Min Heap:

We can perform different types of operations on Min Heap i.e. Insertion, Deletion, Sorting, Traversing and Searching.

Once a heap has been built, Heap sort can simply remove the minimum value (root node) and create the output, sorted array one item at a time. This process is somewhat akin to the Selection Sort but much more efficient.

When the root node in a heap is removed to become part of the final, ordered data set, the last item on the heap is promoted to fill the vacancy at the root position. Clearly, in many cases, this last item will now be out of place. To ensure that the modified heap retains the min-heap property it becomes necessary to ``push down" the newly promoted root item until it is back in the right place. This pushing down process entails examining the node's key value and comparing it with the key value of the node's smallest child. If the node's smaller child is small in value than the node itself, a swap is performed. The process repeats, following the node from the root

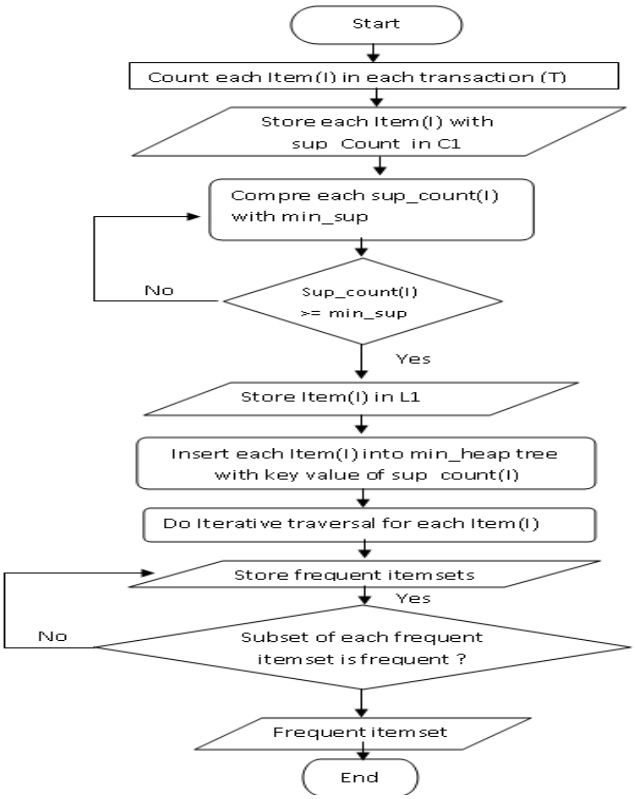
down through demotion, until no swap is needed. At this point the heap is back in order, the new root may be popped off, and the sorting process can continue.

The process of removing the root, promoting the last node, and re-heapifying continues until the heap is exhausted.

IV. PROPOSED METHOD

We are introducing a new method for finding the frequent pattern mining which is based on MINHEAP.

A. Flowchart



B. Iterative Traversal

The term traversal means visit/read each node at least ones. In iterative traversing we will traverse the tree and scan he frequent item set according to the iteration.

Consider the following transactional database having transaction id (tid) and List of items.

TABLE IIIII
AN EXAMPLE PREPROCESSED TRANSACTIONAL DATABASE

Tid	List of Items
100	{Pen, Ink, Rubber}
200	{ Pen, Ink, Pencil }
300	{Pencil, Rubber}
400	{Ink, Pencil}
500	{Register, Ink, Pen}

As our algorithm first we need to scan database for the Support Count, that will generate following output.

TABLE IV
ITEMS WITH SUPPORT COUNT

Item	Support Count
Pen	3
Ink	4
Rubber	2
Pencil	3
Register	1

Now we will discard the Items that have support count less than Minimum Support Count. In our case Minimum Support Count is 2.

So after discarding Item that has less than 2 support count, the output will be as follows;

TABLE V
ITEMS WITH SUPPORT COUNT

Item	Support Count
Pen	3
Ink	4
Rubber	2
Pencil	3

Now we insert each item as node in Heap to create Min-Heap.

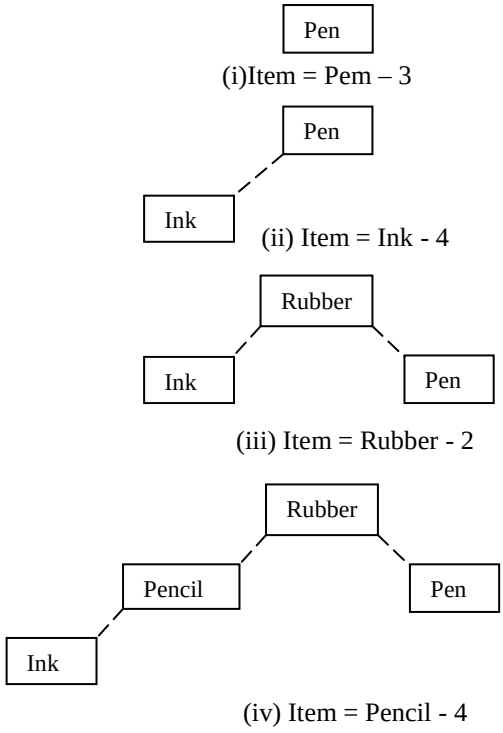


Fig. 2 A Heap Tree

Iterative Traversal: The output of the above Heap Tree by a this method will be as follows & find most frequent item;

1. Iteration 1
Frequent 1 Itemset will generate
{Rubber-2, Pencil-3, Pen-3, Ink-4}

2. Iteration 2
Frequent 2 Itemset will generate
{Rubber-2 Pencil-3 , Rubber-2 Pen -3, Pencil-3 Ink-4}

3. Iteration 3
Frequent 3 Itemset will generate
{Rubber-2 Pencil-3 Pen-3 }

V. CONCLUSIONS

In this paper, we present the FP-Growth tree. We are trying to exploring the new approach on frequent pattern mining. Hopefully, this short overview may give rough outline of our recent work and give just idea of general view of MINHEAP tree by using comparison based technique. In general, we feel as a young research field in data mining, frequent pattern mining has achieved tremendous progress and claimed a good set of applications. However, in-depth research is still needed on several critical issues so that the field may have its long lasting and deep impact in data mining applications.

REFERENCES

[1] Carlos Ordonez, Edward Omiecinski, Levien de Braal, "Mining Constrained Association Rules to Predict Heart Disease".

[2] Saravanan .M.S, Rama Sree .R.J, "Performance study of Association Rule Mining Algorithms for Dyeing Processing System", Innovative Systems Design and Engineering ISSN 2222-1727 (Paper) ISSN 2222-2871 (Online) Vol 2, No 5, 2011

[3] M.Sathish "A Literature Survey on association rule using apriori algorithm in various fields".

[4] Huiying Wang, Xiangwei Liu, "The Research of Improved Association Rules Mining Apriori Algorithm" 2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD).

[5] Huan Wu, Zhigang Lu, Lin Pan, Rongsheng Xu, Wenbao Jiang," An Improved Apriori-based Algorithm for Association Rules Mining", Sixth International Conference on Fuzzy Systems and Knowledge Discovery, IEEE Society community, 2009.

[6] R.Divya , S.Vinod kumar ,"Survey on AIS,Apriori and FP-Tree algorithms",In: International Journal of Computer Science and Management Research Vol 1 Issue 2 September 2012, ISSN 2278-733X.

[7] Agrawal, R. and Srikant, R. 1995." Mining sequential patterns", P. S. Yu and A. S. P. Chen, Eds.In:IEEE Computer Society Pres Taipei, Taiwan

[8] Zhu F, Yan X, Han J, Yu PS, Cheng H (2007) Mining colossal frequent patterns by core pattern fusion. In: Proceeding of the 2007 international conference on data engineering (ICDE'07).

[9] Z. Liu, G. Sang, and M. Lu, "A Vector Operation Based Fast Association Rules Mining Algorithm," in Proc. of Int. Joint Conf. on Bioinformatics, System Biology and Intelligent Computing, pp. 561-564, 2009.

