A Hybrid Algorithm Using Apriori Growth and Fp-Split Tree For Web Usage Mining

Dr. Parvinder Singh[1], Vijay Dahiya[2]
[1]*(Department of Computer Science, DCRUST, MURTHAL, India)*
[2]*(Department of Computer Science, DCRUST, MURTHAL, India)*

Abstract : *Internet is the most active and happening part of everyone's life today. Almost every business or service or organization has its website and performance of the site is an important issue. Web usage mining based on web logs is an important methodology for optimizing website's performance over the internet. Different mining techniques like Apriori method, FP Tree methodology, K-Means method etc. have been proposed by different researchers in order to make the data mining more effective and efficient. Many people have modeled Apriori or FP Tree in their own way to increase data mining productiveness. Wu proposed Apriori Growth as a hybrid of Apriori and FP Tree algorithm and improved FP Tree by mining using Apriori and removed the complexity involved in FP Growth mining. Lee proposed FP Split Tree as a variant of FP Tree and reduced the complexity by scanning the database only once against twice in FP Tree method. This research proposes a new hybrid algorithm of FP Split and Apriori growth which combines the positives of both the algorithms to create a new technique which provides with a better performance over the traditional methods. The new proposed algorithm was implemented in java language on web logs obtained from IIS server and the computational results of the proposed method performs better than traditional FP Tree method, Apriori Method.*

Keywords: *Apriori Growth, FP Split, Frequent Patterns*

I. Introduction

Now days, the most dynamic place in world on the earth before land and sea is internet. Internet is having a huge, volatile, varied, heterogeneous, semi-structured, and ever progressing data. The whole businesses, government schemes, services which are provided over the internet are mainly dependent on efficiency of the data mining techniques. The applied data mining techniques can provide for an increased profit, competitiveness among the I.T firms which is a prerequisite in this 21st century of science and technology.

II. Related work

Data mining which is still in the arena of discovery is being the hot topic of researchers in the past. Various algorithms have been proposed with numerous modifications in the field of data mining. Some of the notable works are as follows:-

A. APRIORI Algorithm: Apriori algorithm which is one of the widely and the oldest used algorithm in data mining is based on the algorithm of breadth first search algorithm where it makes use of a data tree structure in order to count candidate item sets in an efficient manner based on the user desired support count. The algorithm generates the candidate item sets of length x from item sets of length x-1. Once, after the generation of candidate item sets it prunes those candidates which do not satisfy minimum support count and which have an inappropriate sub pattern.
Limitations:
Though Apriori algorithm is one of the simplest algorithm, still it suffers with certain limitations.

1. It is very expensive to grasp a huge number of candidate sets. The amount of candidates to be generated increases exponentially with increasing n-item set.
2. It is very difficult and a tiresome task for scanning the database repeatedly and looking for a greater number of candidates by matching their pattern, which is very necessary for mining large patterns.

B. FP Growth Algorithm: In order to get the mostly used item sets without any generation of candidate sets, FP-Growth Algorithm provides with the best option in order to search for the suitable candidates based on the support count, which also enhances the performance and efficiency . The FP Growth Algorithm core is based on the storing of the frequent data items into a special type of data structure i.e. Frequent Pattern Tree(FP-tree). Here along with frequent items, a mapping of the items is also stored for faster access and better results.
Limitations

DOI: 10.9790/0661-17633943

Despite of being one of the time efficient algorithm, this algorithm suffers from following limitations.

a) The database is scanned twice for two times for the construction of FP Tree.
b) The making of FP Tree using this algorithm takes a lot of time.

III. Proposed Methodology

The proposed algorithm is a hybrid of a modified FP Tree creation algorithm i.e FP Split and Apriori Growth mining algorithm. This proposed algorithm can be explained using two phases.

The first phase constructs the FP Split Tree which is more efficient way to create candidate sets than FP Tree since the latter involves two complete scans of the database while the former does it once. This impacts the efficiency almost 2 times better. More over the FP Split Tree created by proposed hybrid algorithm involves lesser use of pointers as here each node is not linked in the Tree to its predecessor and successor. Rather here a header list is maintained separately which maintains a list separately for each of the pages which points to the occurrence of these items in the final tree created.

The second phase involves mining the FP Split tree created using the Apriori growth algorithm. This algorithm is more efficient than FP Growth as it does not involve recreating the FP Split trees repeatedly every time in recursion as in FP Growth algorithm thereby reducing the time involved.

Phase 1: Tree construction using FP-Split Tree Algorithm.

Step-1. The database is scanned to create an equivalence class of items. Let the equivalence class of item be ECi= {Tid | Tid are the identifier of transaction ti; i is an item of ti).

Step-2. In step 2, support is calculated in order to filter out the non-frequent items**.** The support of each item I- refers to the number of records contained in the equivalence class ECi. Let |ECLi| denote the support of the equivalence class ECi. After support calculation, items having supports below the predefined minimum support are deleted from the set.

Step-3. In step 3, firstly frequent items are generated; secondly, the equivalence class of item is converted into nodes for the construction of FP-split tree. Moreover, in order to facilitate tree traversal, a header table is built in advanced so that each item can point to its first occurrence in the FP-split tree.

The node structure of FP Split tree is as follows:-

<table>
<tr><td>Content</td><td>Count</td><td rowspan="2">Link_Sibling</td></tr>
<tr><td colspan="2">List</td></tr>
<tr><td colspan="3">Link_Child</td></tr>
</table>

Table 1: Node Structure for FP Split Tree

In Table 1, Count represents the support count, Link_Sibling represents pointer linkage to the sibling nodes, Link_Child represents the pointer linkage to the child nodes and content represents the frequent item set.

Step-4**.** This step starts the beginning of the FP Split tree construction; firstly a dummy root is created.

Step-5. The nodes are added into the FP Split tree on the basis of the four rules. These four rules are , where x stands for a specific node in the FP Split tree.

```
Rule I:
If ( x is root and x.Link-child= null ) Then
x.link-child<= n
Else
Call Compare (x.Link_childList, n.list )
End if

Rule 2:
If ( n.list c x.list and x.Link-child == null ) Then
x.Link-child<= n
else
Call Compare (x.link-child.List, n.List )
End if;

Rule 3:
If ( n.List n x.List== 0 and x.Link-siblings== null) Then
x.Link-sibling <=n
else
Call Compare (x.Link-child.List, n.List )
End if;
```

```
Rule 4:
If (x.List n n.List # 0 and x.List - n.List<#0 ) Then
Call split ( n ) and return two nodes nl and n2
End if
```

On the basis of above four rules the new node is compared different nodes like root node, child node and sibling node and is added accordingly into the FP Split tree.
Finally the Tree so created is ready for mining using Apriori Growth Algorithm.

Phase 2:- Tree Mining using Apriori Growth Algorithm.
In order to perform the mining by Apriori Growth aglgorithm firstly the candidates are generated using a candidate set algorithm.

Candidate set Algorithm

```
Step 1.     List1=k-1 frequent item dataset from Data
Step 2.     N=size(list1)
            Initialize mylist as a blank list to contain generated frequent item dataset
Step 3.     Repeat for I=1 to n-1
Step 4.     Repeat for j=I+1 to n
Step 5      l1=list1[I]
Step 6      l2=list1[j]
Step 7      Remove the last elements from l1 and l2
Step 8      if l2 is a subset of l1 then
                Flist=append last element of l2 at end of l1
            [end of while]
Step 9.     If count(flist)>=supp then
                Add flist to mylist
            Else
                return null
Step 10     end
```

New Apriori Growth Algorithm

Input: pagelist2: list of pages with equivalence class satisfying support count
Tree: Nodes of tree as a list.
Sup: minimum support
Output: frequent item sets ‘data’
The algorithm is implemented with the following steps:-

```
    Step 1.     Repeat following step while scanning pagelist2 till end
          ▪ Create list containing single item from pagelist2
          ▪ Add this list to mylist1
            [End of repeat]
    Step 2.     Add mylist1 to data
    Step 3      Repeat for k=2,3,4,….
    Step 4.     $C_k$=getCandidate(k)
    Step 5.     If [$C_k$]=0 then
                    Goto step 6
               Else
                    Add $C_k$ to data
    Step 6.     End
```

Finally, we have the mined item sets in the pagelist2 data structure.

IV. Implementation Results

Output Generated is as follows:
The Frequent sets generated for threshold value 3 are as follows:
The itemsets generated for the support count 1 for the are:-

```
 [[0], [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21],
[22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41],
[42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60], [61],
[62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73]]
```

The itemsets generated for the support count 2 are:-
[[0, 39], [0, 49], [1, 39], [1, 49], [2, 39], [7, 39], [9, 39], [12, 39], [12, 49], [13, 39], [14, 39], [23, 39], [29, 39], [29, 49], [30, 39], [39, 41], [39, 48], [39, 49], [39, 55], [39, 65], [39, 70], [39, 72], [41, 49], [48, 49], [49, 65], [49, 72]]

The itemsets generated for the support count 3 are:-
[[0, 39, 49], [1, 39, 49], [12, 39, 49], [29, 39, 49], [39, 41, 49], [39, 48, 49], [39, 49, 65], [39, 49, 72]]
The algorithm cannot move further for support count 4, as no more frequent items are observed.

Comparison of proposed algorithm with fp tree.

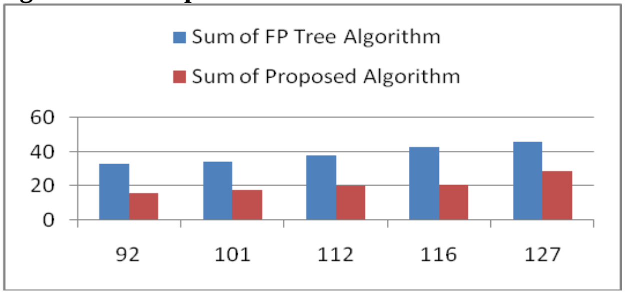

Figure 2: Number of records(number) vs timespan(milliseconds)

The graph in figure 2 shows the comparison between FP Tree and FP Split tree methods for support count 3. We can see that the time taken by proposed algorithm is always better than the traditional method. The difference would be more clearly visible if we get logs of few days time for some website where the incoming traffic is also more frequent. The proposed algorithm has been tested on logs received from Kurukshetra University website for only 29 minutes. The more is the number of records, the better visible is the difference between the efficiency of the two algorithms.

The data from two algorithms when averaged, it demonstrated the effectiveness of our proposed methodology. The efficiency of this technique as against traditional method is found to more efficient according to time consumed for execution.

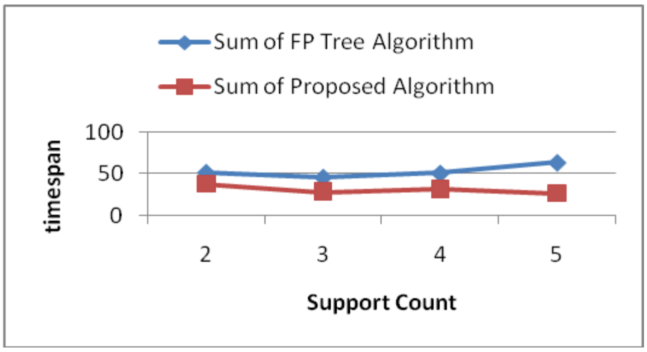

Figure 3: Support count(number) vs timespan (milliseconds)

The graph in figure 3 shows the comparison between FP Tree and FP Split tree methods for different support counts for a fixed no. of records. This comparison can also be seen with the table 2 as follows:-

Support	FP tree time	Proposed Algorithm Time
2	52	38
3	46	29
4	61	32
5	54	27

Table 2: Comparing the FP tree and Hybrid algorithm in time(milliseconds).

V. Conclusion

Web usage mining refers to the use the access logs from a web server to study the usage pattern of clients. This research discusses different techniques for content mining on website logs. The two main methods in this context- Apriori and FP Tree method are the traditional methods. The most commonly used Apriori algorithm had a major disadvantage of performing multiple database scans for candidate set generation. The FP Tree structure solved this problem by restricting the database scans to two times. But the FP growth algorithm was very complicated and time consuming since it recursively created trees at every step during frequent item-set generation. The FP-Split algorithm further improved candidate set generation by doing a single scan of the date for candidate generation. The Apriori growth algorithm when used for mining the FP Tree performed better in frequent item-set generation. So we proposed here a hybrid technique for web usage mining using FP Split Tree and Apriori Growth algorithm. The hybrid algorithm was programmed using Java language and logs from kurukshetra university website were used to demonstrate the validity and effectiveness of proposed technique. The results showed that the proposed algorithm performed more efficiently than the traditional method.

This method can be used in association mining at many other applications like WSN, Social Network behavioral mining etc. In future, we plan to further improve the efficiency by reducing the complexity involved in creating FP Split tree which is certainly better than FP Tree but still can be worked upon for further improvement.

References

[1] Chin Fewng Lee and Tsung-HsienShen, “An FP-split method for fast association rules mining”, IEEE 2005. Pp. 459-464.

[2] Bo Wu, Defu Zhang, QihuaLan, JieminZheng, “An Efficient Frequent Patterns Mining Algorithm based on Apriori Algorithm and the FP-tree Structure”, Third 2008 International Conference on Convergence and Hybrid Information Technology, IEEE 2008.

[3] Anupam Joshi, Tim Finin, Akshay Java, Anubhav Kale, and PranamKolari, “Web (2.0) Mining: Analyzing Social Media”, IEEE 2008.

[4] K. R. Suneetha, Dr. R. Krishnamoorthi, “Identifying User Behavior by Analyzing Web Server Access Log File”, IJCSNS International Journal of Computer Science and Network Security. VOL.9 No.4, April 2009

[5] Mehdi Heydari, Raed Ali Helal, Khairil Imran Ghauth, “A Graph-Based Web Usage Mining Method Considering Client Side Data”, 2009 International Conference on Electrical Engineering and Informatics, 2009.

[6] HuipingPeng, “Discovery of Interesting Association Rules Based on Web Usage Mining”, 2010 - International Conference on Multimedia Communications.

[7] MajaDimitrijevic, TanjaKrunic, “Association rules for improving website effectiveness: case analysis”, Online Journal of Applied Knowledge Management Volume 1, Issue 2, 2013

[8] Kirti S. Patil, Sandip S. Patil, “Sequential Pattern Mining Using Apriori Algorithm & Frequent Pattern Tree Algorithm”, IOSR Journal of Engineering (IOSRJEN) Vol. 3, Issue 1 (Jan. 2013), Pp. 26-30