ISSN 2319-2720

Volume 2, No.4, October - December 2013

International Journal of Computing, Communications and Networking

Available Online at http://warse.org/pdfs/2013/ijccn07242013.pdf

Association rules Mining Using Improved Frequent Pattern Tree Algorithm

Ms.Monal saxena [1], Mr.Alok jain[2], Mr.Navin gupta[3], Ms.Neha Saxena[4]

[1] Asst. Professor, Dept. of CSE., VISM Gwalior (MP)-INDIA,monalsaxena12@gmail.com
[2] Asst. Professor, Dept. of CSE., VISM Gwalior (MP)-INDIA, alok_0208@gmail.com
[3] Asst. Professor, Dept. of CSE., VISM Gwalior (MP)-INDIA, g.navin123@rediffmail.com
[4]Asst. Professor, Dept. of CSE., IITM Gwalior (MP)-INDIA, saxenaneha07@gmail.com

ABSTRACT:

Web mining can be broadly defined as discovery and analysis of useful information from the World Wide Web. Web Usage Mining can be described as the discovery and analysis of user accessibility pattern, during the mining of log files and associated data from a particular Web site, in order to realize and better serve the needs of Web-based applications. Web usage mining itself can be categorised further depending on the kind of usage data considered they are web server, application server and application level data. This Research work focuses on web use mining and specifically keeps tabs on running across the web utilization examples of sites from the server log records. The bonding of memory and time usage is compared by means of Apriori algorithm and improved Frequent Pattern Tree algorithm.

Keywords- Web usage mining, Apriori algorithm, improved Frequent Pattern Tree algorithm

I INTRODUCTION

The Web is a vast, volatile, diverse, dynamic and mostly amorphous data repository, which stores incredible amount of information/data, and also enhance the complexity of how to deal with the information from the different opinion of view, users, web service providers and business analyst. The users wish for the effective search tools/engine to locate related information easily and accurately [1]. The Web service providers desire to find the technique to guess the user's behaviors and personalize information to shrink the traffic load and create the Web site suited for the different set of users [2]. The business analysts want to have tools to learn the consumer's needs. All of them are expecting equipment or techniques to help them satisfy their needs and solve the problems encountered on the Web. Therefore, Web mining becomes a trendy active area and is taken as the research topic for this analysis [3].Web usage mining is the process of finding out what users are looking for on the internet. Few users might be looking at only documented data, whereas some others might be interested in multimedia data. It is the submission of facts and figures mining techniques to find out interesting usage patterns from World Wide Web facts and figures in alignment to realise and better serve the desires of Web-based applications. Usage facts and figures hold the persona or source of World Wide Web users along with their browsing demeanour at a World Wide Web site. Web usage excavation itself can be categorized farther counting on the kind of usage facts and figures considered [4]:

Web Server Data: The client logs are assembled by the Web server. Usually facts and figures include Internet Protocol address, sheet quotation and get access to time.

- Application Server Data: financial submission servers have significant characteristics to endow e-commerce submissions to be built on peak of them with tiny effort. A key feature is the proficiency to pathway diverse kinds of enterprise events and logs them in application server logs.
- Application Level Data: New types of events can be characterised in an application, and logging can be twisted on for them therefore generating histories of these particularly characterised events. It should be noted however, that numerous end submissions need a combination of one or more of the methods directed in the classes above.

II. APRIORI ALGORITHM

The current Research work is planned to work on log files. Apriori is a typical algorithm for frequent item set mining and association rule learning over transactional databases. It is proceed by recognize the frequent individual items in the database and extend them to big and big item sets as long as those item sets appear sufficiently often in the database [5].

The frequent item sets find out by Apriori can be used to find out association rules which highlight general trends in the database: this has applications in domains such as market basket analysis.

The system operates in the following three modules.

- Preprocessing module
- Apriori or FP Growth Algorithm Module
- Association Rule Generation
- Results

The preprocessing module converts the log file, which normally is in ASCII format, into a database like format, which can be processed by the Apriori algorithm [6].

The second module is performed in two steps.

- Frequent Item set generation
- Rules derivation

Algorithm: Apriori algorithm Pass 1

```
Produce the candidate itemsets in A1
Save the frequent itemsets in B1
Pass k

Generate the candidate itemsets in Ak from the frequent
itemsets in Bk-1
Join Bk-1 m with Bk-1n, as follows:
insert into Ak
select m.item1, m.item2, m.itemk-1, n.itemk-1
from Bk-1 m, Bk-1n

where m.item1 = n.item1, m.itemk-2 = n.itemk-2, m.itemk-1
< n.itemk-1
Generate all (k-1)-subsets from the candidate itemsets in Ak
Prune all candidate itemsets from Ak where some (k-1)-
subset of the candidate itemset is not in the frequent itemset
Bk-1

Scan the transaction database to findout the support for each
candidate itemset in Ak
Save the frequent itemsets in Bk
```

III. IMPROVED FREQUENT PATTERN TREE ALGORITHM

The FP-Tree Algorithm, suggested by Han in, is an alternative way to find frequent piece groups without utilising applicant generations, therefore advancing performance. For so much it values a divide-and-conquer strategy. The central part of this method is the usage of a specific data structure entitled frequent-pattern tree (FP-tree), which keeps the piece set association information.

In simple words, this algorithm works as follows:

- It compresses the input database conceiving an FP-tree instance to represent common items.
- It divide the compressed database into a set of conditional databases, each one affiliated with one common pattern
- Eventually, each database is extract separately.

Using this scheme, the FP-Tree decrease the enquire charges looking for small patterns recursively and then concatenating then in the long common patterns, proposing better selectivity [7].In large databases, it’s not likely to hold the FP-tree in the major memory. An approach to cope with this difficulty is to foremost split up the database into a group of lesser databases (called projected databases), and then construct a common Pattern-tree from each of these smaller databases.

A. Fp Tree Structure

FP tree is a solid data architecture that retained important, absolutely vital and quantitative information considering common patterns [8].The main attributes of Frequent Pattern tree are:

- It comprises of one root marked as “root”, a set of piece prefix sub-trees as the child of the root, and a frequent-item header chart.
- one-by-one node in the piece prefix sub-tree comprises of three areas:
- ➢ Item-name: It lists which item this node represents.
- Count: It registers the number of transactions represented by the portion of the path coming to this node
- Node-link: It connects to the next node in the FP-tree bearing the identical item-name, or null if there is none.
- one-by-one application in the frequent-item header journal comprises of two area:
- ➢ item-name
- ➢ Header of node- link, which points to the first node in the FP-tree carrying the item-name.

IV. PROPOSED SYSTEM

The aim of the proposed system is to recognize usage pattern from web monitor files of a website. Apriori and FP Tree Algorithm is used for this. Both are prominent algorithms for mining frequent item sets for Boolean association rules. In computer science and data mining, Apriori is a typical algorithm for understand association rules [10]. Apriori

Algorithm follows “bottom-up” technique, used to design to operate on databases containing transactions.

A. Web usage mining:

Web usage mining is a regular detection of patterns in click streams and linked data collected or generated as an outcome of client communications with one or more Web sites. The purpose is to scrutinize the behavioral patterns and profiles of users interacting with a Web site. The discovered patterns are usually symbolized as collections of sheet, objects, or resources that are commonly accessed by collection of users with common interests.

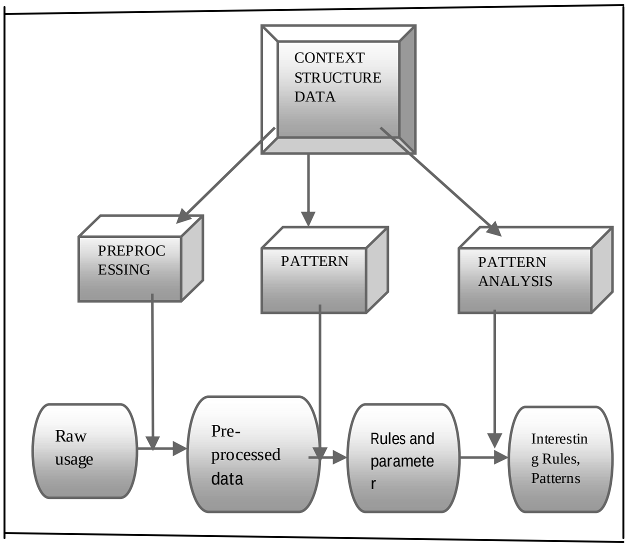


Fig1: web usage mining process

4.1.1 Algorithm for new improved fp –tree web using mining

Variables used in this algorithm are as follows:

- URI Stem: is the field in the log that corresponds to the address of a web-page.
- AtEndOfLog: tells us, whether the log record come to an end.
- Token : is a variable that is initially set to a value of 0
- SID: is the session ID of the record that has been retrieved from the log record.
- Write: function that writes the instructed value in a file.

First an array arr is maintained where a number of unique session ids are stored.

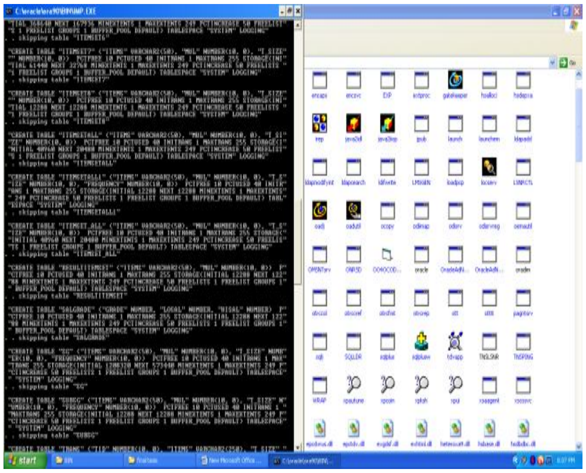
```
While Not At End Of Log
Read Log  Record
Token = 1
If SID = arr then
Token = 0
End if
If Token = 1 then
arr (k)= SID
k=k+1
end
if
Wend
Session Distribution
Session x Starts
For I = 0 to n
While not At End Of Log
Read Log Record


/* only those files that have either .asp or .html extension
name are being selected */

If SID = arr (i) and right (URIStem, 4) = ".asp" or right
(URI Stem, 5) = "html "and

/* repeated occurrence of URI Stem is ignored */

URI Stem # url then
write URIStem
url = URIStem
End if
Wend
Next
Session x Ends
```

Thus a log file that has mega bytes of data can be reduced to a few bytes. The above Algorithm works for a single session, this can be repeated for a desired number of times which is equal to the number of sessions required to analyze.

V. IMPLEMENTATION

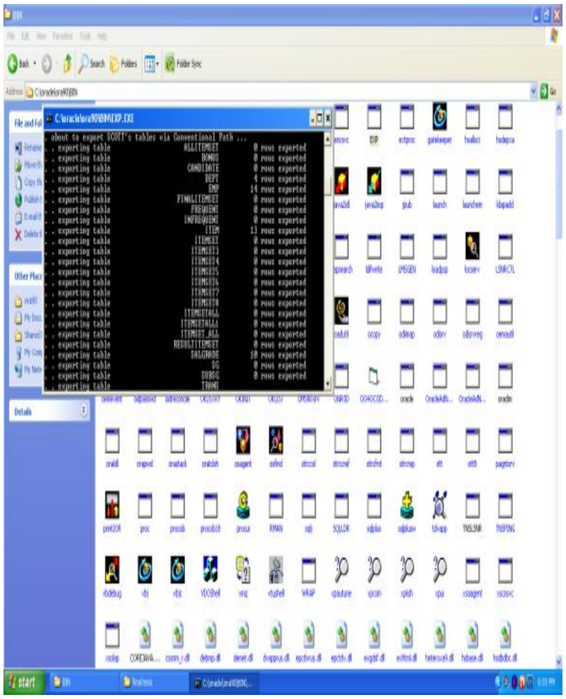
The experiments were performed using a SQL and a windows operating system in a command prompt workstation with only one 32 bits CPU and one giga of RAM memory. Under large minimum supports, improved fp tree runs faster than apriori while running slower under large

Experiments. Both algorithms adopts a divide and conquer approach to decompose the mining problem into a set of smaller problems and uses the frequent pattern (FP-tree) tree and web mining algo or data structure to achieve a condensed representation of the database transactions. Under large minimum supports, resulting table and graph in relatively small size so with this condition based FP tree does not take advantages of small memory space and also page fault for both algorithm is almost equal. minimum supports. Table 1.1 shows what minimum support used in



```
C:\oracle\ora90\BIN\IMP.EXE
"TIAL 368640 NEXT 167936 MINEXTENTS 1 MAXEXTENTS 249 PCTINCREASE 50 FREELIST"
"S 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM" LOGGING"
. . skipping table "ITEMSET6"

"CREATE TABLE "ITEMSET7" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0), "T_SIZE"
"" NUMBER(10, 0))  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 STORAGE(INI"
"TIAL 61440 NEXT 32768 MINEXTENTS 1 MAXEXTENTS 249 PCTINCREASE 50 FREELISTS "
"1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM" LOGGING"
. . skipping table "ITEMSET7"

"CREATE TABLE "ITEMSET8" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0), "T_SIZE"
"" NUMBER(10, 0))  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 STORAGE(INI"
"TIAL 12288 NEXT 12288 MINEXTENTS 1 MAXEXTENTS 249 PCTINCREASE 50 FREELISTS "
"1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM" LOGGING"
. . skipping table "ITEMSET8"

"CREATE TABLE "ITEMSETALL" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0), "T_SI"
"ZE" NUMBER(10, 0))  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 STORAGE(I"
"NITIAL 40960 NEXT 20480 MINEXTENTS 1 MAXEXTENTS 249 PCTINCREASE 50 FREELIST"
"S 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM" LOGGING"
. . skipping table "ITEMSETALL"

"CREATE TABLE "ITEMSETALL1" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0), "T_S"
"IZE" NUMBER(10, 0), "FREQUENCY" NUMBER(10, 0))  PCTFREE 10 PCTUSED 40 INITR"
"ANS 1 MAXTRANS 255 STORAGE(INITIAL 12288 NEXT 12288 MINEXTENTS 1 MAXEXTENTS"
" 249 PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABL"
"ESPACE "SYSTEM" LOGGING"
. . skipping table "ITEMSETALL1"

"CREATE TABLE "ITEMSET_ALL" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0), "T_S"
"IZE" NUMBER(10, 0))  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 STORAGE("
"INITIAL 40960 NEXT 20480 MINEXTENTS 1 MAXEXTENTS 249 PCTINCREASE 50 FREELIS"
"TS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM" LOGGING"
. . skipping table "ITEMSET_ALL"

"CREATE TABLE "RESULTITEMSET" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0))  P"
"CTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 STORAGE(INITIAL 12288 NEXT 122"
"88 MINEXTENTS 1 MAXEXTENTS 249 PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1"
" BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM" LOGGING"
. . skipping table "RESULTITEMSET"

"CREATE TABLE "SALGRADE" ("GRADE" NUMBER, "LOSAL" NUMBER, "HISAL" NUMBER)  P"
"CTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 STORAGE(INITIAL 12288 NEXT 122"
"88 MINEXTENTS 1 MAXEXTENTS 249 PCTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1"
" BUFFER_POOL DEFAULT) TABLESPACE "SYSTEM" LOGGING"
. . skipping table "SALGRADE"

"CREATE TABLE "SG" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0), "T_SIZE" NUMB"
"ER(10, 0), "FREQUENCY" NUMBER(10, 0))  PCTFREE 10 PCTUSED 40 INITRANS 1 MAX"
"TRANS 255 STORAGE(INITIAL 1208320 NEXT 573440 MINEXTENTS 1 MAXEXTENTS 249 P"
"CTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE"
" "SYSTEM" LOGGING"
. . skipping table "SG"

"CREATE TABLE "SUBSG" ("ITEMS" VARCHAR2(50), "MUL" NUMBER(10, 0), "T_SIZE" N"
"UMBER(10, 0), "FREQUENCY" NUMBER(10, 0))  PCTFREE 10 PCTUSED 40 INITRANS 1 "
"MAXTRANS 255 STORAGE(INITIAL 12288 NEXT 12288 MINEXTENTS 1 MAXEXTENTS 249 P"
"CTINCREASE 50 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE"
" "SYSTEM" LOGGING"
. . skipping table "SUBSG"

"CREATE TABLE "TRANS" ("TID" NUMBER(10, 0), "ITEMS" VARCHAR2(250), "T_SIZE" "
```

Fig 2: implementation for new improved fp tree algorithms in SQL +



```
C:\oracle\ora90\BIN\EXP.EXE
. about to export SCOTT's tables via Conventional Path ...
. . exporting table                      ALLITEMSET          0 rows exported
. . exporting table                           BONUS          0 rows exported
. . exporting table                       CANDIDATE          0 rows exported
. . exporting table                            DEPT          4 rows exported
. . exporting table                             EMP         14 rows exported
. . exporting table                    FINALITEMSET          0 rows exported
. . exporting table                        FREQUENT          0 rows exported
. . exporting table                      INFREQUENT          0 rows exported
. . exporting table                            ITEM         11 rows exported
. . exporting table                         ITEMSET          0 rows exported
. . exporting table                        ITEMSET3          0 rows exported
. . exporting table                        ITEMSET4          0 rows exported
. . exporting table                        ITEMSET5          0 rows exported
. . exporting table                        ITEMSET6          0 rows exported
. . exporting table                        ITEMSET7          0 rows exported
. . exporting table                        ITEMSET8          0 rows exported
. . exporting table                      ITEMSETALL          0 rows exported
. . exporting table                     ITEMSETALL1          0 rows exported
. . exporting table                     ITEMSET_ALL          0 rows exported
. . exporting table                   RESULTITEMSET          0 rows exported
. . exporting table                        SALGRADE         10 rows exported
. . exporting table                              SG          0 rows exported
. . exporting table                           SUBSG          0 rows exported
. . exporting table                           TRANS
```

Fig 3: implementation for new improved web mining algorithms in sql +

But as minimum supports decrease resulting data structure size rapidly increase, it require more memory space , at this point advantage of improved frequent pattern tree come in existence with less page fault web mining algorithms considerable work well with high dense database along with small minimum supports. And we are implemented both algorithms and finally implemented result are as follows, in shown in figure.

VI. PERFORMANCE RESULTS

Web usage mining has emerged as the essential tool for realizing personalized user-friendly and business-optimal Web services. Web usage mining is used by e-commerce sites to organize their sites and to increase profits.

Apriori -the classical mining algorithm is a way to find out certain potential, regular knowledge from the massive ones. But there are two more serious defects in the data mining process. The first needs many times to scan the database and the second will inevitably produce a large number of irrelevant candidate sets which seriously occupy the system resources. An improved method is introduced on the basic of the defects above. The improved algorithm reduces the database scan, and we prune the candidate item sets according to the minimum supporting degree and we get the frequent item sets. After analysis, the improved algorithm reduces the system resources occupied and improves the efficiency and quality.

Now we try to understand the above defined approach with the help of an example and thereby comparing it with the conventional Apriori algorithm and our new approach.

Table 1: data base Table

TID	List of item _IDs
TI0	11,12,15
T200	12,14
T300	12,13
T400	11,12,14
T500	11,13
T600	12,13
T700	11,13
T800	11,12,13,15
T900	11,12,13

After applying phase 1 of our algorithm then above set of pattern items we get:

First we are comparing the support count in apririori algo and our new algo implementation and result we get:

First we are compare support count and time compare result is shown in the table. And graph is following:

Tables 2: compare result between time and support count

Sr.no	Support count	Time in second		
		Apriori algorithms	Improved fp tree	Improved web mining
1	2	43	38	35
2	4	35	28	25
3	6	15	10	08
4	8	15	08	05
5	10	11	05	01

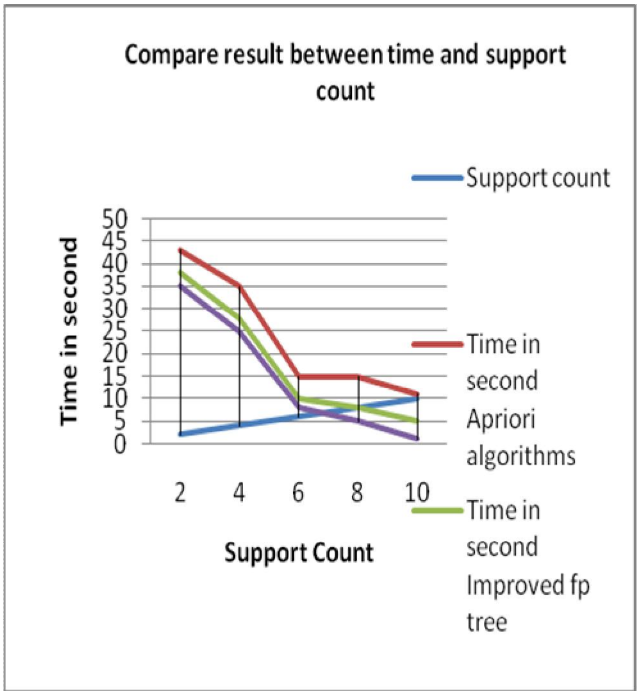


Fig3: Compare result between time and support count

Then applying the II phase algorithms the result se get it that are as flows and we compare the time and no. database are taking the time. And result as shown in table and graph are as flows. For the comparative study of web mining and frequent pattern design, we have taken a database of 1000 transaction of 13 items. In this systematic process we considered 1000 transactions to generate the all frequent itemset and pattern with the support count 2-10%.shown in the table 2and fig.3

Table 3: Compare result between database scan or time

Sr.no	Database scan	Time in second		
		Apriori algorithms	Improved fp tree	Improved web mining
1	5	150	140	130
2	7	330	290	274
3	9	552	489	467
4	11	833	821	794
5	13	957	911	891

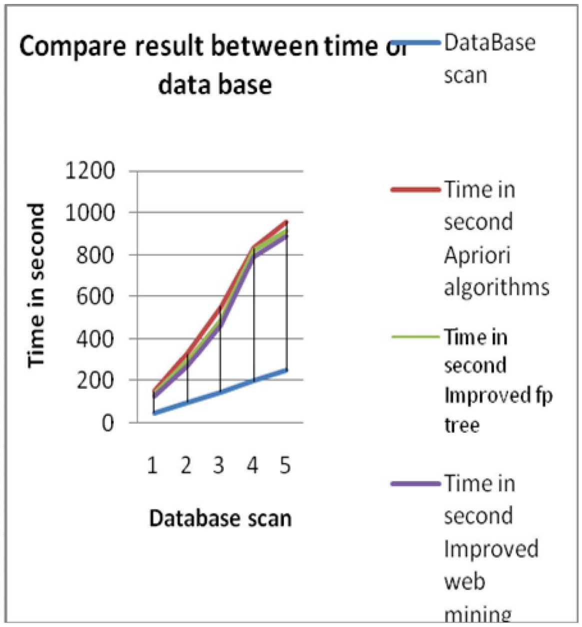


Fig4: Compare result between time or data base

All the Graphs presented in this section were calculated in the same way. After processing each new tuple the following statistics were computed. We present a complete analysis of the experiments carried out in this research as well as, a short conversation about why the results obtained show that our approach is suitable for the web mining scenario.

We have repetitive the similar process by increasing the transaction, after the testing on both technique, so find the finalized this result, we have designed a graph and summarized a result

VII. CONCLUSION

Web usage mining is the procedure of finding out which users are looking for the internet. It can be described as the sighting and scrutiny of user ease of access pattern, during mining of files and its connected data from a Web site, in order to recognize and better serve up the desires of Web-based applications. One of the algorithms which is very simple to use and easy to implement is the Apriori algorithm.

This algorithm is used in the present Research work to generate association rules that associates the usage pattern of the clients for a website. The output of the system was in terms of memory usage and speed of producing association rules.

The main drawback of Apriori algorithm is that the candidate set creation is costly, especially if a large number of patterns and/or long patterns exist. The main drawback of FP-growth algorithm is the explosive quantity of lacks a good candidate generation method. Future research can combine FP-Tree with Apriori candidate generation method to solve the disadvantages of both apriori and FP-growth. In future the algorithm can be extended to web content mining, web structure mining, etc. The work can also be extended to extract information from image files.

REFRENCES

[1] K .S .K .D. **Association Rules Mining: A Recent Overview,** GTS International Tran on Computer Science, Vol.65 (1), 2006, pp.45-65

[2] A R "**Fast Algorithms for Mining Association Rules**", Sep 12-15 1994, Chile, 487-99, pdf, 1-55860-153-9.

[3] Mannila H**,"Efficient algorithms for discovering association rules mining."** conference Knowledge Discovery in Databases (SIGKDD). 181-83.

[4] Tan, P. N., M. St., V. Kumar, **"Introduction to web Mining",** Addison-Wesley, 2013, 769pp.

[5] I. H. Witten and E. Frank, **Data Mining: Practical Machine Learning Tools and Techniques** with Java Implementation, 2nd ed. San Mateo.

[6]Huang, H., Wu, X.. **Association analysis with one scans of web data bases.** Paper submitted at the IEEE On Data Mining, Japan.

[7] R. Jin **"An Efficient Implementation of Apriori Association web mining,"** Proc. Workshop on High Performance Data webMining, Apr. 2011.

[8] J. H and M. Kaber, **"association mining:"** 2014.

[9] Han J **"Mining frequent patterns without candidate rules mining technique,"** in the national seminar of the international web of data, ACM Press, pp. 4-11-2004

[10] E-H. Han, G. Caryopsis "**Scalable Data web mining for Association web Rules,"** IEEE Trans. Eng., vol. 12, no. 3, July 2012.

[11] Brin S., R. Mot, J.D. Ullman, "**web item set counting and implication rules**

[12] Association mining in data base", **in Proceedings of the ACM SIGMOD** International Conference on Management of Data, pp.289-294, 1999.

[13] Masseglia F**., "Using Data Mining Techniques on Web Access Logs to Dynamically Improve Hypertext Structure language"**, In ACCM Web Letters, Vol. 10 No. 9, pp.13-19, 2011.

[14] Stumme.A G., Hotho A.H and Berendt B. **"Semantic Web Mining-A web survey"**MIIT press in Delhi, No.2, pp.124-143, 2014.

[15] Pei J. and Han J**. "Constrained frequent pattern mining: a pattern growth view"** in SIGKDD Explorations, Vol. 4, No. 1, pp. 31-39,2004.

[16] Antunes C. and Olivei A.L.G "**Generalization and association web Pattern-Growth for Sequential attern web Mining with Gap Constraints"** in Int'l Conf Machine Learning and Data in published 2012.