Research Article

Cognitive Analysis on Web Server Log Metaheuristic Algorithm and Distributed ARM

Naresh Kumar Kar[1], Subhash Chandra Shrivastava[2], Megha Mishra[3]

[1]Dept. of CSE, RCET, Bhilai, Chhattisgarh, India
[2]Dept. of Mathematics, RCET, Bhilai, Chhattisgarh, India
[3]Dept of CSE, SSGI, Bhilai, Chhattisgarh,India

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021

Abstract: Web use is increasing daily, Web use mining (WUM) and frequent pattern mining make it easier to assess cognitive evaluation from Web server log. This cognitive analysis is helping the organization decision makers to take on strategy decisions. Association rule mining (ARM) is considered one the most excellent method for identifying frequent patterns from data source. Here our aim is to find frequently used elements such as urls, the greatest number of used urls, set of urls. These elements are browsed together in the way as we can identify website consumer behavior. In this paper we have proposed an algorithm that are a comprehensive approach of Metaheuristic (Genetic algorithm) and Multicore processing named as GMARM (Genetic Multi-core Association Rule Mining). Here we enter web usage data to a preprocessing tool which is Genetic algorithm-based preprocessing, then we apply association rule mining to find out user behavior pattern using MARM algorithm. MARM algorithm finds gain of multi-core processor, preprocessed data passed to distributed processors.
Keywords—Genetic, GMARM, Support, ARM, Confidence**.**

1. Introduction

Business information can be obtained from the web server log. Suppose the site owner wants to find out which URL's are extremely lesshave been visiting and which URL's are extremely visited. He can use a web usage mining algorithm to discover this fascinating pattern.Following that, the site owner would be able to make the right decision. Web usage mining is helping to know web user conduct, their best interest area. The preceding factors influenced us to conduct this study as a cognitive analysis of web server logs.Discovery of useful knowledge from web is referred to as web content mining. Web structure mining finds the link structure of web. Web usage mining have various stages as shown in fig.-1.

Using association rule mining we can find out frequentitemset. Let us understand ARM with the help of Market basket data fragment. Set of transactions are given, we must find the rules. This One will forecast the incidence of an item in accordance with the principle of occurrences of other items in the transaction. Set of transactions T given, the ultimate goal of association rule mining is to find all rules that are as follows:

1. support(s)≥ minsup threshold
2. Confidence(c)≥ minconf threshold

Terms used in ARM are as follows:

- Itemset: Collection of items Example: {Milk, Diaper, Bread}
- k-itemset: which keeps k items
- Support count: Number of occurrence of an itemset E.g. ({Milk, Bread,Diaper}) = 3
- Support: Fraction of transactions that include an itemset, E.g. s({Milk, Bread, Diaper}) = 3/5
- Frequent Itemset: whose support is greater than or equal to a minsup threshold
- Association Rule: Implication expression A ® B, here A,B are itemsetsfor Example: {Beer}, {Milk, Curd}
- Support (s): Fraction of transactions contain both X and Y
- Confidence (c) A ® B

Confidence(A ® B) = support(A U B)/support(A)

Table 1. Market Basket Transaction

TID	*Items*
1	Item1, Item2
2	Item1, Item3, Item4, Item5
3	Item2, Item3, Item4, Item6
4	Item1, Item2, Item3, Item4
5	Item1, Item2, Item3, Item6

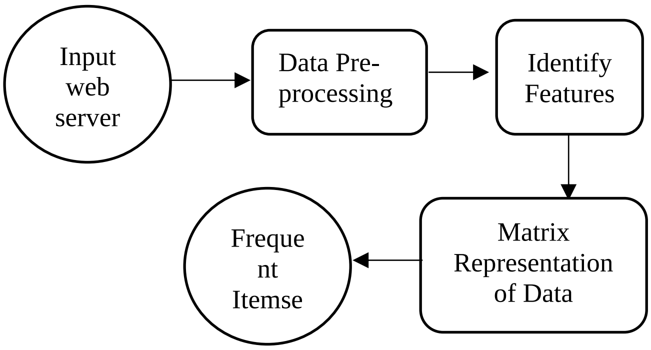

Fig.1. Web Usage Mining Steps

2. **Example of Rules:**

{Item2,Item3}→{Item4}(s=0.5,c=0.67), {Item2,Item4} → {Item3} (s=0.5, c=1.0), {Item3,Item4} → {Item2} (s=0.5, c=0.67), {Item4}→{Item2,Item3}(s=0.5,c=0.67) {Item3} → {Item2,Item4} (s=0.5, c=0.5) ,{Item2} → {Item3,Item4} (s=0.5, c=0.5)

- Examples:

—Rules and regulations provided previously are binary partitions of no differentitemset: {Item1, Item2, Item4}

—The support for rules generated from the same itemset is similar, but we'll have different confidence. As a result, the support and confidence requirements can be separated.

In this paper we have taken the advantage of multicore processing to parse the input retrieved in the form of matrix after preprocessing phase. Fig.-2 depicts the logical processor available and core available. Further in section 2 we will explain some research which aretaken in this filed, in section 3 we will explain some known problem from literature survey, in section 4we will discuss about our proposed methodology, in section 5investigational result and dataset used and conclusivelyin section 6 we will accomplish our investigation.

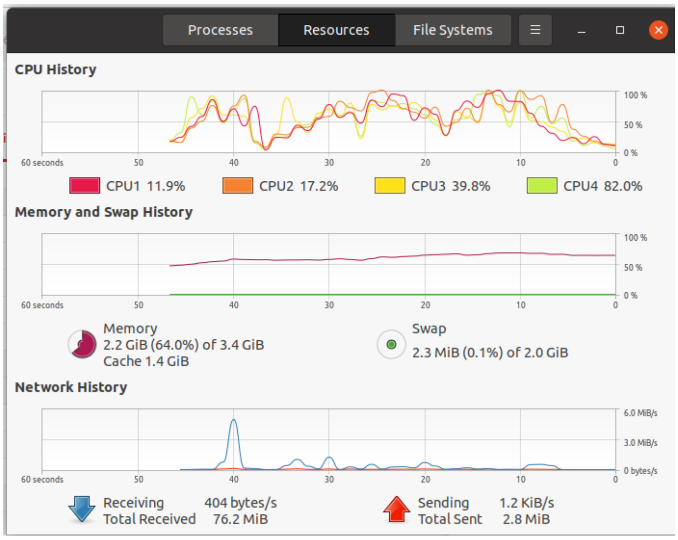

Fig.2. Multi-core Processor

3. **Literature Survey**

Web usage mining has been the subject of several studies. Rash-mi is a term used to describe a person who is In 2018, JayathirthaRao et al. proposed a cognitive bias for an educational assessment. The author used data from a smart school room web log to create multiple models based on mouse clicks and movement. For discovering learning behaviour, the author used statistical data and data mining techniques [1]. In 2005, VishwaVinay, Ingemar J. Cox et al. compared four different dimensional reduction techniques and evaluated their efficacy, concluding that PCA and ICA have less significant precision unless applied to larger datasets [2]. JayantiMehra suggested a web usage mining algorithm for server web log statistics preprocessing in 2018. The author detailed every aspect of the information that is being pre-processed, as well as the frequency of page access [3]. Following our research into the different literature, we discovered the following states:

Table 2. Comparison

Dimension reduction technique					
Database	**ICA**	**PCA**	**RM**	**NR**	**Genetic**
MED	0.254	0.254	0.175	0.198	0.32
CRAN	0.185	0.185	0.112	0.132	0.28

4. Problem Identification

After reading different literature we found some shortcoming in this research are as follows:

- Due to hefty size of web log server need high computation time.
- Existing ARM algorithm consumes much I/O time for scanning input dataset for calculation of support for each candidate set.
- Noise present in input dataset reduces the algorithm performance.
- Existing research may not concentrate over preprocessing of input dataset.
- The web access dataset contains a large number of features that are either irrelevant or redundant.

Let us understand how repetitive scan required for each candidate sete.g. input dataset as follows:

Table 3. Example Input Dataset

ID	URL ID	Timestamp	URL	URL Visited
1	Url 1	08/Mar/2019:16:05:49	/mailman/bin/view/TWiki/WebTopicEditTemplate	1
2	Url 2	08/Mar/2019:16:06:51	/twiki/listinfo/business	1
3	Url 3	08/Mar/2019:16:11:58	twiki/view/bin/Main/DCCAndPreFix	1
4	Url 3	08/Mar/2019:16:20:55	/twiki/view/bin/Main/DCCAndPreFix	1
5	Url 2	08/Mar/2019:16:23:12	/mailman/listinfo/business	1

Above dataset (D) contains matrix with order 5 x 5. In step-1 we need to calculate support for candidate set {Url1, Url2, Url3}, we need to scan D 5 times for Url1 similarly for each Url, so Total number of scans= 5*5=25. In step-2 calculate support for candidate set-2 {{Url1,Url2}, {Url2,Url3}, {Url1,Url3}} and candidate set-3 {{Url1,Url2, Url3}}, here earlier algorithm scan the dataset repetitively for candidate set1,2,3 respectively even though scanner already visited.

5. Solution Methodology

To overcome the shortcomings of discussed, we have passed input dataset to Metaheuristic algorithm which is Genetic algorithm based preprocessing algorithm. Further for identifying user behavior pattern we will pass preprocessed data to MARM (Multi-core Association Rule Mining). Flow of proposed method depicted in fig.-3. Where the filtered data is stored in a matrix of order N x M, with N denoting number of rows and M denoting number of columns.Matrix must transpose using distributed matrix transposition, further frequent itemset will be generated.

Genetic Algorithm

Dimension reduction in Row Wise:

We wanted to get the URL from the Web log data at the time. The fitness function is D, we get the following string tokenized by "\n" (line feed):

$$D = \sum_{i-1}^{n} S_i \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.(1)$$

n= the total number of lines in the log file.

S_i= string each line

i= line no

$S_i$will provide us URL. \\, a to z, A to Z - As a result or gene, the string formed (URL) by S_i is deliberated. The following are the fitness functions:

$D1 = Si\in$["\\]\\[-] $\ldots\ldots\ldots\ldots\ldots\ldots.(2)$

$$D2 = Si\in[\text{-}] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.(3)$$

If D2 have null (ϕ)value then we should remove it. If output string is O_s then

$$O_s = [D2] \forall\, D2 \neq \phi \ldots\ldots\ldots\ldots\ldots\ldots\ldots.(4)$$

Dimension reduction in column wise:

At this stage, we're doing cognitive analysis on the web log, so we'll need to compute URLs and their frequency, as well as the amount of time users spend on each URL and the response code. As a result, the other features will be removed.

If we transpose any n x m matrix, then diagonal elements remain same after transposition. In parallel transposition data is divided into processor threads (Matrix), each processor thread P(i,j) has three registers, as depicted in fig.-4, fig.-4 shows the how processor threads communicate and exchange data elements.

If we transpose any n x m matrix, then diagonal elements remain same after transposition. In parallel transposition data is divided into processor threads (Matrix), each processor threadP(i,j) has three registers, as depicted in fig.-4, fig.-4 shows the how processor threads communicate and exchange data elements.

Parallel Matrix Transposition

Step 1: do steps (1.a) and (1.b) in parallel
(1.a) for x = 2 to n do in parallel
for j = x to x - 1 do in parallel
C(x- 1, j) (a, j, x)
end for
end for
(1.b) for x = I to n -I do in parallel
for j = x + I to n do in parallel
B(x, rj -1) (ax, j, x)
end for
end for

Step 2: do steps 2.a, 2.b, and 2.c in parallel
(2.a) for x = 2 to n do in parallel
for j = I to x - I do in parallel
while P(x, j) receives input from its neighbors do
(i) if (akx, m, k) is received from P(x + 1, j)
then send it to P(x - 1, j)
end if
(ii) if (ak,, m, k) is received from P(x - 1, j)
then if x = m andj = k
then A(x, j) - a,, {ak, has reached its destination}
else send (akx, m, k) to P(x + 1, j)
end if
end if
end while
end for
end for

(2.b) for x = 1 to n do in parallel

while P(x, i) receives input from its neighbors do
(i) if (ak., m, k) is received from P(x + 1, i)
then send it to P(x, x + 1)
end if
(ii) if (ak., m, k) is received from P(x, x + 1)
then send it to P(x + 1, x)
end if
end while
end for
(2.c) for x = 1 to n - 1 do in parallel
for j = x + I to n do in parallel
while P(x, j) receives input from its neighbors do
(i) if (ak., m, k) is received from P(x, j + 1)
then send it to P(x, j - 1)
end if
(ii) if (akin, m, k) is received from P(x, j -1)
then if x = m andj = k
then A(x, j) +- ak. {ak, has reached its destin ation}
else send (akx, m, k) to P(x, j + 1)
end if
end if
end while
end for
end for

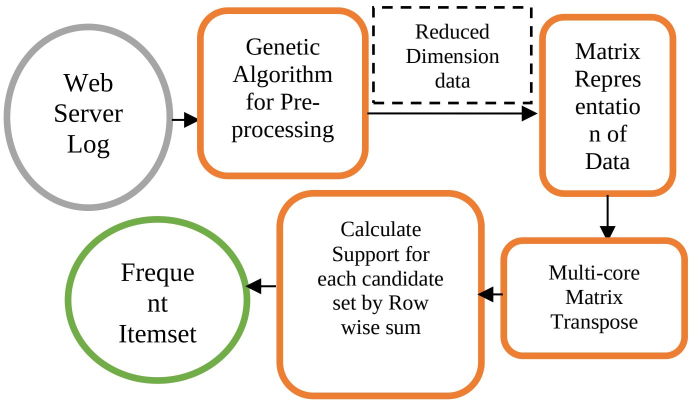

Fig.3. Proposed Flow GMARM

A(i,j) =>to keepAijfirst and Aji when algorithm ends.
B(i,j) used to store data from P(i,j + 1) or P(i - 1,j), that is, from it's top neighbors or its right neighbors.
C(i,j) used to keep data received from P(i + 1,j) or P(i,j - 1), from its bottom or left neighbors.

ID	URL ID	URL Visited
1	Url1	1
2	Url2	1
3	Url3	1
4	Url3	1
5	Url2	1

ID	1	2	3	4	5
URL ID	Url1	Url2	Url3	Url3	Url2
URL Visited	1	1	1	1	1

d Web Log

1. Input Transposed Matrix
2. //Generate Candidate set
   Ck=Gen_candidate_sets (URL ID)
3. Perform row wise sum for itemset-1 and calculate support.
4. Call ck=suppress(ck)
5. while(Ck≠Null)
   Ck=Gen_candidate_itemsets (URL ID)

Perform dot-multiplication of matrix to calculate support.
Ck= suppress (Ck)
6. End while

Gen_candidate_sets (URL ID)

Ck = Φ
L_{k-1}=URL ID
for all itemsets $I_1 \in L_{k-1}$ do
for all itemsets $l_2 \in L_{k-1}$ do
if $I_1[1] = I_2 [1] \wedge I_1 [2] = I_2 [2] \wedge \ldots \wedge I_1 [k-1] < I_2 [k-1]$ then
c = $I_1 [1], I_1 [2] \ldots I_1 [k-1], I_2 [k-1]$
Ck = $C_k \cup \{c\}$

suppress(ck)

for all c ∈Ck
for all (k-1)-subsets d of c do
if d ∉ Lk-1
thenCk = Ck – {c}

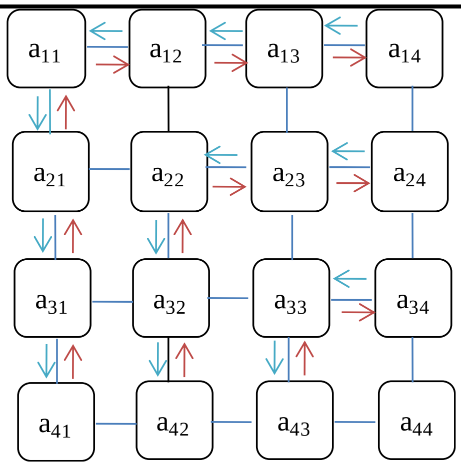

Fig. 5. Parallel Matrix Transpose for 4x4 matrix

Suppose the generated transposed transaction matrix as given below:

T1	10	10	10	10
T2	11	0	0	11
T3	15	15	18	14
T4	11	11	11	12

So to calculate support count for above URL transaction matrix, just need logical row wise some for **1-itemset** as follows:

Itemset	T1	T2	T3	T1		Logical Sum (Support Value)
URL1	10	11	15	11	→	4
URL 2	10	0	15	11	→	3
URL 3	10	0	18	11	→	3
URL 4	10	11	14	12	→	4

For **2-itemset**

URL1,URL 2	URL1,URL 3	URL1,URL 4	URL2,URL 3	URL2,URL 4	URL3,URL 4

URL1	10	11	15	11	Support Value
URL2	10	0	15	11	

URL1*URL2	**1**	**0**	**1**	**1**	**3**
URL1	10	11	15	11	Support Value
URL3	10	0	18	11	
URL1*URL3	**1**	**0**	**1**	**1**	**3**
URL1	10	11	15	11	Support Value
URL4	10	11	14	12	
URL1*URL4	**1**	**1**	**1**	**1**	**4**
URL2	10	0	15	11	Support Value
URL3	10	0	18	11	
URL2*URL3	**1**	**0**	**1**	**1**	**3**
URL2	10	0	15	11	Support Value
URL4	10	11	14	12	
URL2*URL4	**1**	**0**	**1**	**1**	**3**
URL3	10	0	18	11	Support Value
URL4	10	11	14	12	
URL3*URL4	**1**	**0**	**1**	**1**	**3**

Likewise, for Itemset-3,4 can be calculated.

6. Experimental Evaluation

We have measured the performance of Algorithm on a 1.8 GHz Intel core i5 laptop machine with 8.0 GB main memory, running on Ubuntu operating system. All programs were developed under the openclgcc compiler, version 1.5 and jdk 1.5. Opencl used for implementation of parallel matrix transposition and JDK used for frequent itemset generation. For experimental evaluation web log data used table-4 shows the data fragment.

Table 4. Dataset

64.242.88.10			[07/Mar/2004:16:05:49	-0800]	"GET	/twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables	HTTP/1.1"	401	12846
64.242.88.10			[07/Mar/2004:16:06:51	-0800]	"GET	/twiki/bin/rdiff/TWiki/NewUserTemplate?rev1=1.3&rev2=1.2	HTTP/1.1"	200	4523
64.242.88.10			[07/Mar/2004:16:10:02	-0800]	"GET	/mailman/listinfo/hsdivision	HTTP/1.1"	200	6291
64.242.88.10			[07/Mar/2004:16:11:58	-0800]	"GET	/twiki/bin/view/TWiki/WikiSyntax	HTTP/1.1"	200	7352

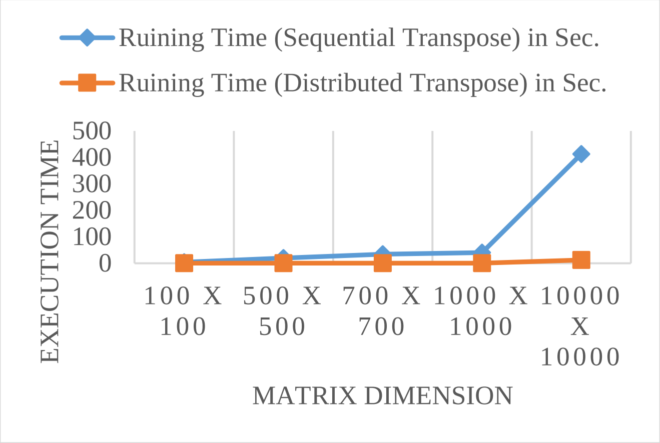

Fig.4. Running Time Comparison (Sequential and Distributed Matrix Transpose)

The Figure 6 shows the performance comparisons between Apriori and GMARMalgorithm. It compares the execution time taken by the apriori algorithm and GMARMalgorithm for different datasets difference in transaction matrix dimension. GMARMand Aprori applied over different datasets difference in file size, GMARMperform well Apriori.

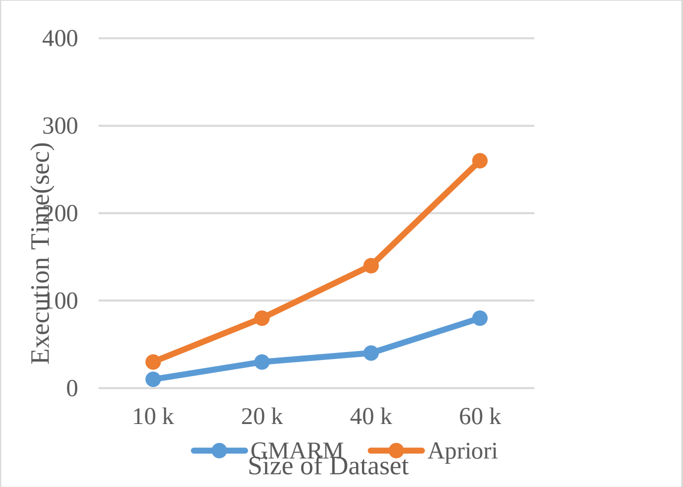

Fig.5. Performance Comparison of Apriori and GMARM

7. Conclusion

Because of great popularity of the internet, we have inspired this research.Doing this kind of research on a web log data can provide information that can be used to better accommodate user needs. In this paper we have passed transposed transaction matrix (dataset) for frequent itemset generation, which is optimal and significant improvement algorithm execution time whereas we have compared GMARM (proposed) with renowned classical Apriori algorithm. In future we can apply distributed processing for GMARM as we saw support calculation for different itemset are independent of each other, this will significantly improve the overall execution time.

References

1. Rashmi Jayathirtha Rao, Christopher Stewart, Arnulfo Perez, and Siva Meenakshi Renganathan, Assessing Learning Behavior and Cognitive Bias from Web Logs 2018 IEEE Frontiers in Education Conference (FIE) DOI: 10.1109/FIE.2018.8658913
2. Vishwa Vinay, Ken Wood, Natasa Milic- Frayling , A Comparison of Dimensionality Reduction Techniques for Text Retrieval, Proceedings of the Fourth International Conference on Machine Learning and Applications (ICMLA'05) 0-7695-2495-8/05 $20.00 © 2005
3. Jayanti Mehra, Dr. R S Thakur , An Effective method for Web Log Preprocessing and Page Access Frequency using Web Usage Mining International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 2 (2018) pp. 1227-1232 © Research India Publications. http://www.ripublication.com

4. Shiming Xiang ; ZishaZhong ; Kun Ding Multicluster Spatial–Spectral Unsupervised Feature Selection for Hyperspectral Image Classification IEEE 2015.
5. P.Miruthula1, S.Nithya Roopa Unsupervised Feature Selection Algorithms: A Survey IJSR 2015.
6. Effect Of Spacing With Different Sowing Method On Growth And Yield Performance Of Green Gram, Ajay Sharma, Aman kumar, Anita Jaswal, International Journal Of Advance Research In Science And Engineering http://www.ijarse.com IJARSE, Volume No. 09, Issue No. 10, October 2020 ISSN-2319-8354(E).
7. Ahmed K. Farahat Ali Ghodsi Mohamed S. Kamel An Efficient Greedy Method for Unsupervised Feature Selection 2011 11th IEEE International Conference on Data Mining.