International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 23s, 2025
https://theaspd.com/index.php

An Efficient Load Balancing Algorithm for Mining Frequent Item sets for Large Transactional Data

Rashmi V[1], Dr. Divvela Srinivasa Rao[2], Dr. S. Sagar Imambi[3], M. Lakshmi Narayana[4], Kishore Reddy Aduri[5], Dr. R. Venkata Lakshmi[6], Narendra Babu Pamula[7*]

[1]Assistant Professor, Department of Information Technology, Prasad V Potluri Siddhartha Institute of Technology,
Vijayawada, Andhra Pradesh, India.

[2]Associate Professor, Department of AI & DS, Lakireddy Bali Reddy College of Engineering, Mylavaram, Krishna District, Andhra Pradesh, India.

[3]Professor, Department of CSE, Koneru Lakshmaiah Educational Foundation, Guntur, Andhra Pradesh, India.

[4]Assistant Professor, Department of Information Technology, S.R.K.R Engineering College, China Amiram, Bhimavaram West Godavari District, Andhra Pradesh, India.

[5]Professor in ECE & Vice Principal (Diploma), Sree Venkateswara Engineering College, Nellore, Andhra Pradesh, India.

[6]Asst.Professor in Department of Computer Science and Engineerining, Bapatla Engineering College (A), Andhra Pradesh, India

[7]Sr. Assistant Professor, Department of Artificial Intelligence & Data Science, Lakireddy Bali Reddy College of Engineering(A), Mylavaram, Andhra Pradesh, India.

*Corresponding author: Narendra Babu Pamula, E-mail: naren.pamula@gmail.com

ABSTRACT

Frequent Itemset Mining (FIM) is critical in many data-intensive applications such as customer behavior analysis, recommendation systems, and fraud detection. Nevertheless, while the size and complexity of transactional databases increase explosively, classical FIM methods face tremendous challenges in scalability, processing efficiency, and system reliability. The growing amount of data can result in loaded servers and virtual machines (VMs), which cause performance reduction, signal loss, and longer execution time. Such problems not just hinder system stability but also lead to a bad user experience. This research proposes a new approach known as Grey Wolf-based Adaptive Frequent Itemset Mining (GW-AFIM), combining intelligent load balancing with frequent itemset mining to address current limitations. The model applies Grey Wolf Optimization (GWO) to automatically evaluate and assign data loads between processing units. It draws inspiration from the manner in which grey wolves hunt and live in packs. The fitness function of the GWO algorithm is responsible for minimizing data traffic, enhancing load balancing, and accelerating computation.GW-AFIM aims at mining transaction-based frequently preferred itemsets with balanced utilization of resources. Rigorous experimental analyses prove that GW-AFIM is remarkably better than current algorithms with respect to processing time, speedup, accuracy, scalability, and error rate. The outcomes show that the proposed model is an efficient and scalable solution for mining frequent itemsets from enormous transactional data, which is very much applicable in contemporary big data environments where real-time analytics and fault tolerance are of utmost importance.

Keywords: *Frequent Itemset Mining (FIM), Load Balance, Data Traffic, Frequent Itemset, Big Data, Execution Time, Adaptive Algorithms, Pattern Discovery.*

1. INTRODUCTION

Fast data mining has gained greater significance in the big data age with the compound growth of digital service-generated transactional data from retail systems, the Internet of Things (IoT), and online platforms. Among these techniques, frequent itemset mining (FIM) is still important. The main thing it does is find important correlations, patterns, and associations in big datasets. There are many applications in practice for FIM because it can find sets of items which occur in transactions together in frequent patterns. These systems help find fraud, keep track of inventory, make recommendations, and guess what customers will do. Even though FIM is of great importance, conventional FIM algorithms such as Apriori, Eclat, and FP-Growth have serious challenges in dealing with big-size and high-dimensional data. Their challenges are high computational expense, poor scalability, inefficiency in memory usage, and rising processing time. Furthermore, in cloud and distributed platforms, data load imbalance among large

numbers of computing nodes usually leads to suboptimal resource consumption, system bottlenecking, and even VM crashes or signal losses. Such problems seriously undermine the efficiency and reliability of FIM systems, rendering them ineligible for real-time or large-scale data analysis applications. To solve these important problems, we need smart load balancing systems that not only make good use of computing resources but also help us find frequent itemsets quickly and accurately. In this context, metaheuristic optimization algorithms have become more popular because they can solve hard optimization problems in environments that are always changing and not always clear. The Grey Wolf Optimization (GWO) algorithm has shown good results in many areas because it is simple, converges quickly, and can balance exploration and exploitation. This paper presents a new method referred to as Grey Wolf-based Adaptive Frequent Itemset Mining (GW-AFIM) that combines the concepts of GWO with FIM. GW-AFIM is used to overcome the shortcomings of traditional methods. The algorithm of GWO is utilized in the model for dynamic load balancing where the fitness of each node is determined by parameters such as the amount of data, processing power, and frequency of transactions. This balances the workload in the system so that it is spread across the system and does not overload it, hence maximizing execution time and system stability. Besides load balancing, GW-AFIM emphasizes the adaptive extraction of frequently preferred itemsets, identified through user transaction behavior and frequency thresholds. The model efficiently eliminates low-utility patterns and highlights high-impact itemsets, which are more pertinent for decision-making in real-world contexts. The implementation of an adaptive thresholding mechanism improves the quality and precision of the extracted itemsets. When compared to previous iterations of the frequent itemset mining algorithm, the new GW-AFIM model is light years ahead. By combining adaptive pattern mining with an intelligent load balancing system inspired by grey wolf behavior, it offers a scalable, efficient, and reliable solution for handling complex large-scale transactional data. This improves mining efficiency and makes it easier to build sophisticated analytics systems that can quickly process massive amounts of data.

The key function of this research is described as follows

- ✓ Initially, the large data transactional itemset is arranged in tree structure model
- ✓ A novel GW-AFIM is developed to balance the data load and to specify the frequently used items.
- ✓ Now, tree structure is constructed for the each types of itemset
- ✓ The process of first fitness model is functioned as monitoring frame to main the data load by ordering the request based on priority request.
- ✓ The second fitness model of grey wolf is updated in threshold to specify the client votes of each item
- ✓ Finally, an efficient transaction is done and its performance is estimated with existing works and attained better result and less run time and high accuracy.

The paper is sketched as **Section 2** elaborates relevant literatures, System model and problem definition is detailed in **Section. 3**, consequently, **Section 4** defined a novel work, **Section 5** drawn the achieved results and its comparison, finally the paper ends with conclusion in **Section.6.**

2. RELATED WORK

Frequent itemset mining (FIM) plays a pivotal role in data mining processes, as it is essential for identifying interesting patterns such as sequences, clusters, classifiers, correlations, and association rules [1], [2]. Initially introduced for market basket analysis [3], FIM helps uncover user behavior patterns and symmetries in domains like supermarkets, online shopping, and mail-order companies [4]. In the context of Big Data (BD), frequent pattern mining (FPM) becomes especially challenging due to the massive volume and unpredictable nature of data, which introduce high space and time complexity [5], [6]. FPM is utilized in mining graphs, sequences, and itemsets [7], and finds applications in clustering, indexing, and association rule mining [8]. The extraction of frequent itemsets is a prominent area of research [9], often focusing on events that occur frequently in databases [10]. However, many traditional approaches struggle with large-scale databases and are inefficient in terms of memory and processing [11], [12], [13]. Big data's structural and unstructured formats further complicate FPM, necessitating efficient algorithms to handle frequency thresholds and pattern extraction [14], [15].

Existing FPM techniques often rely on column enumeration methods, where each column represents a search space [16]. These methods are generally ineffective for high-dimensional data [17] and face challenges in load balancing, synchronization, and transaction management, particularly in distributed and parallel environments [18]. Load balancing strategies based on static and dynamic models have been explored [8], and machine learning techniques have been proposed to improve load distribution.

However, many of these approaches still fall short in achieving optimal classification and balance [19]. Hybrid machine learning algorithms have also been utilized for frequent itemset extraction [20], though challenges persist. For instance, while SVMs have been used for multimedia classification [21], ant colony optimization (ACO) models tend to suffer from low classification accuracy. Deep reinforcement learning (DRL) models such as the MLB framework have shown promise in mobility load balancing in ultra-dense networks [22], yet they lack classification capabilities.

Traditional algorithms like Apriori are frequently used for pattern classification, but they are not well-suited for large datasets and parallel processing. To address this, Raj et al. proposed the Efficient Apriori-based FIM (EAFIM), which improves classification by reducing input data size [23]. However, load balancing remains a limitation. Vanahalli and Patil developed the Balanced Distributed Parallel Frequent Colossal Closed Itemset Mining (BDPFCCIM) mechanism to handle high-dimensional biological datasets, achieving better distribution but with high processing time [24]. To enhance performance, Devi and Sabrigiriraj introduced the Levy Flight Bat Algorithm (LFBA), integrating online feature selection to filter low-quality features [25]. Additionally, the Weighted Entropy Frequent Pattern Mining (WEFPM) model was introduced to improve classification accuracy. Despite its effectiveness, WEFPM's complexity increases due to uncertainty in data.

3. SYSTEM MODEL AND PROBLEM STATEMENT

The system model of big transactional data and its data traffic problem is shown in **Figure 1.** Usually, the big data contains huge amount of statistics with different applications. To make the efficacy of frequent item classification all parameters should be enhanced. Also the function process of big transactional model is if the request is send through the server then it searches the big data. Immediately, the search engine search for the reference and send through the server connected virtual machine. At that time the high load data damage the virtual serer and tends to signal fault, these issues take more time complete the process. For that, the load balancing is the key strategy to maintain the load balance during data broadcasting.

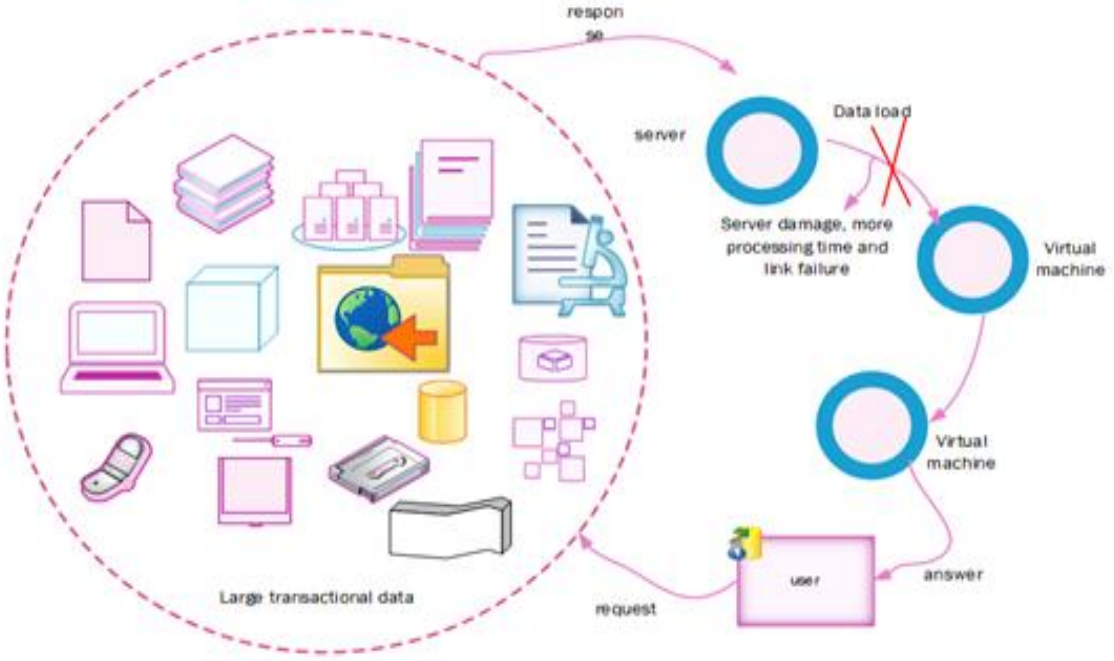

Figure 1: System Model

4. PROPOSED GW-AFIM FOR LOAD BALANCING AND FREQUENT ITEM CLASSIFICATION

In the big transactional data, the FIM is applied to select the frequent used items. Moreover, once the frequently used things are classified then data transmission is processed to answer the client response. In this model the login strategy is utilized, when the owner enters the clients or customers name then frequent used items by that particular client is displayed. In addition, this process should be executed in less execution time. So that a novel GW-AFIM is introduced to balance the data load, best function of

grey wolf model is processed to manage the data load. The proposed architecture is shown in **Figure 2.** Finally, the run time is calculated and compared with recent existing models attained less execution time.

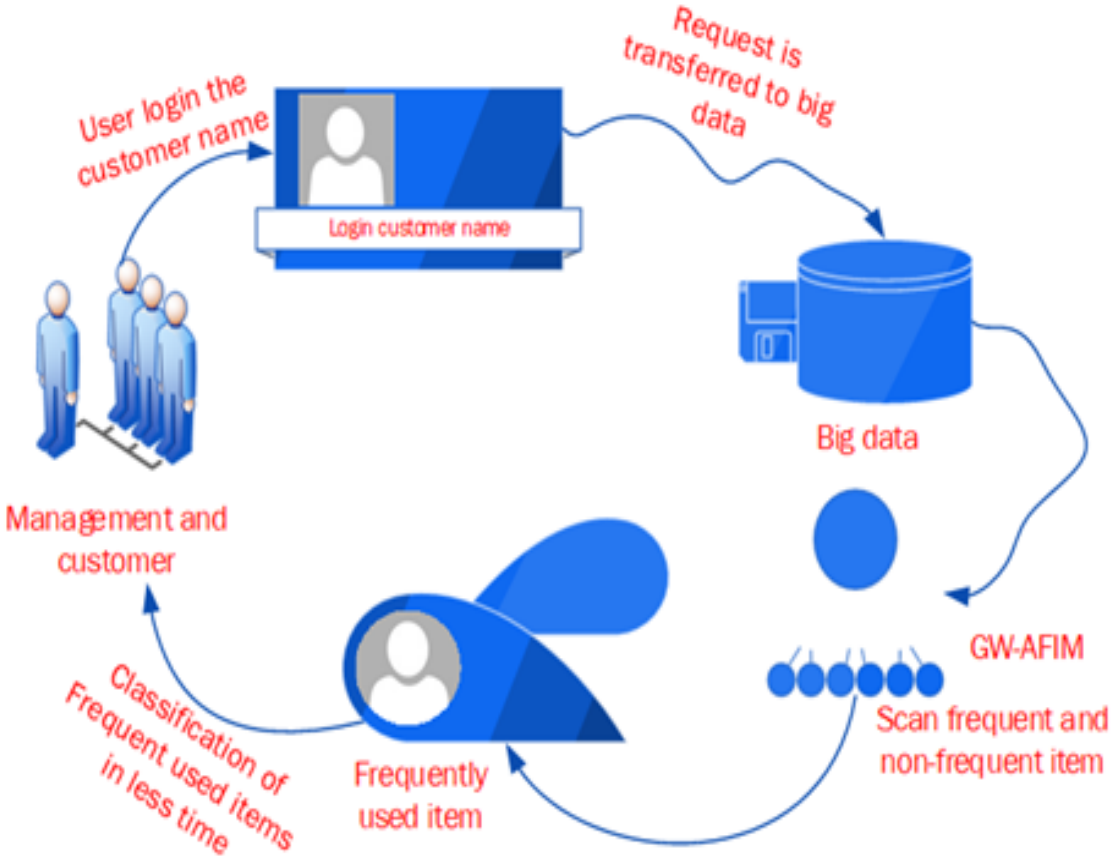

Figure 2: Proposed GW-AFIM

4.1 Process of GW-AFIM

The procedure model is initiated by selecting the frequent paths of an item, which means initially, the location is allocated for all items at that time paths are defined. Moreover, in this path the grey wolf fitness function of monitoring element is updated in path to track the path function. Moreover, the proposed approach is the scanning model to categorize the each type like frequent items and non-frequent items. The proposed scheme is initiated with the selection frequent path. To identify the frequent path, types of all items are described: after that it is split up and separated in tree subset. Moreover there is no requirement of mining the tree branches because already the trees are built with all possible patterns. In the second stage, a database is created to store the un frequent item during the monitoring process. Also the Grey wolf model has two fitness functions, which is determined as head and sub head wolf. Here, it is utilized load balancing and item classification. In addition, the branch of tree is worn to find the vote occurrence of each item. The target of this developed tree based transactional model is to make the patterns and clustering with other items. Moreover, the item patterns that have identical length is grouped together and arranged in same location one by one. In other word, it is the structure of array has different items with various length dimensions. The commonly utilized items and rarely used items are separated by the support customers vote. The items, which have less vote count is considered as rarely used items also in this proposed model, rarely used item sets are removed from the tree structure to reduce the data load and to enhance the specification process.

International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 23s, 2025
https://theaspd.com/index.php

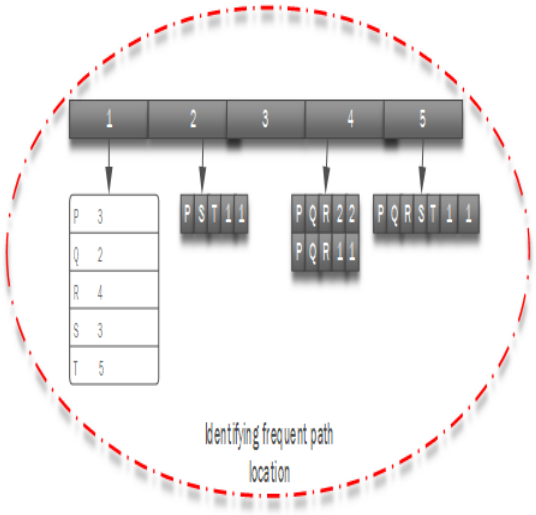

Figure 3(a): a frequent path finding

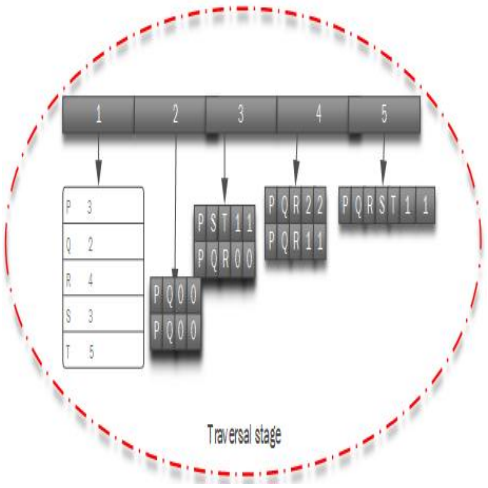

Figure 4: Traversal Stage

The commonly utilized items and rarely used items are separated by the support customers vote. The items, which have less vote count is considered as rarely used items also in this proposed model, rarely used item sets are removed from the tree structure to reduce the data load and to enhance the specification process. After the specification frequent item set, the non-used items are classified and removed. When the non-used things are removed from the dataset then the size dataset get reduced and made the classification process easier. In this planned model there is one free database in the last step to put the non-frequent items. The process steps are detailed in **Figure 3, 4, 5 and 6.**

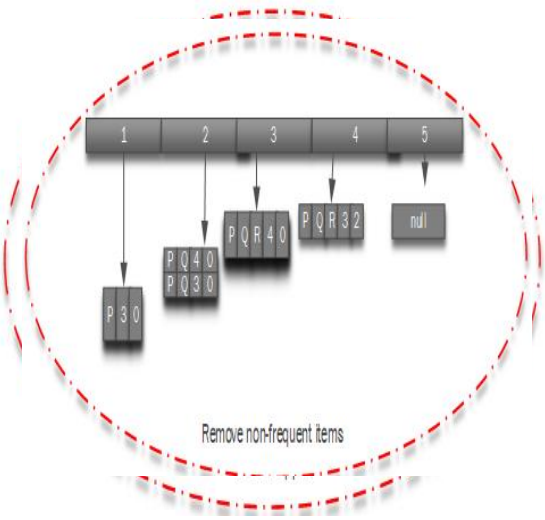

Figure 5: Support Count Figure

Figure 6: Removing Non Frequent Items

Furthermore, The Grey wolf heuristic model is worn to specify the frequent items vote and maintain the load balance. Here, the threshold range is updated with grey wolf second fitness model that is sub leader wolf. Also the fitness function G is updated as the monitoring function to maintain the load balance. Moreover, the data load is balanced by arranging the request based on priority based. So if the data is processed one by one the data load is controlled. The work flow model of proposed approach is shown in **Figure 7**, the main goal of this suggested research is to provide the proficient data transmission for large volume transactional data also, for processing the big data the classification of frequent item is more important to manage the online business without any difficulties.

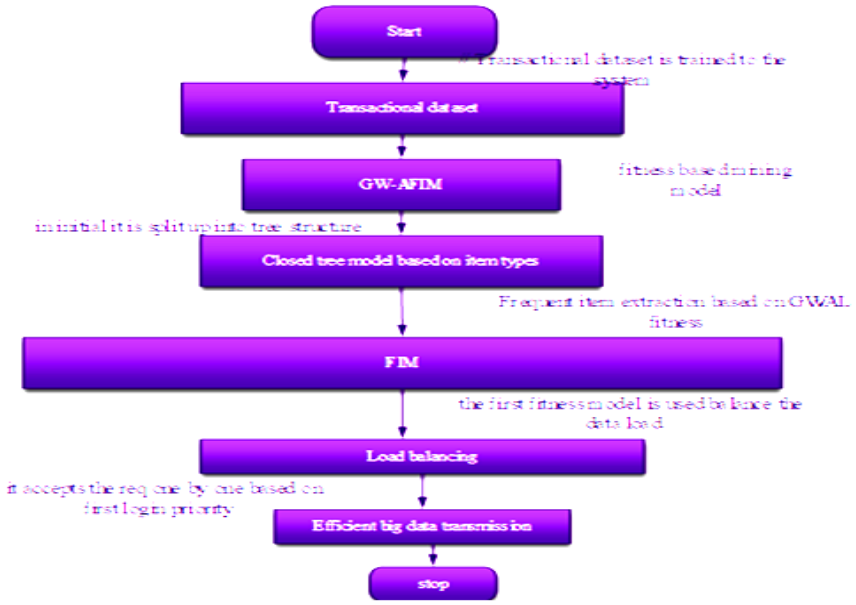

Figure 7: Work Flow of GW-AFIM

5. RESULTS AND DISCUSSION

The planned model is elaborated in java and running in windows 10 platform. Moreover, to carry on the load balancing model, login model is utilized. Based on priority registration the grey wolf send the response, other subsequent are wait for previous task completion. Furthermore, the fitness grey wolf model is processed for continuous monitoring and balance data with scalability.

Algorithm.1 Pseudo code for application based GW-AFIM

Initialize with large transactional frequent dataset $T^*(i=1,2,\ldots n)$

Begin item types a,b,c

Construct tree subset S^*

Set the monitoring element G

// Here, G is the monitoring element to track the data traffic

validate the fitness of all connected hubs

$$T^* = G_1(T^*)$$

// G_1 is second fitness process to track the frequent items for each users

while $(e \leq 3)$: classify as frequent item

//here 3 is the threshold range as support count, e is specific item, if it is greater than or equal to 3

then it is classified as frequent item set.

end for

G=data ze> e

Update reference node

Data transfer $\rightarrow$ *reference* node

Update T^*

end while

return G

5.1 Case Study

The planned schedule is elaborately described in this case study. For that one of the large transactional database is taken, the pattern of transactional data base structure is drawn in **Figure 8**.

Let us consider P,Q,R,S,T,U,V,W,X,Y is the each types of item, now it is arranged in transaction database pattern. Now, the proposed novel algorithm is applied over the transactional database to specify the frequently used item from the large transactional set. The process and tree structure model is detailed in **Figure 9**. Once the item is structured, commonly used itemset are chosen by taking intersection function of each model.

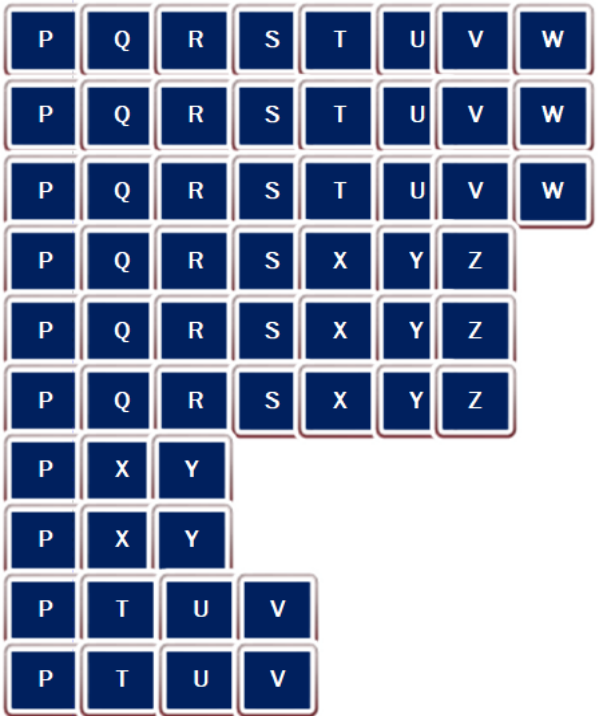

Figure 8: Transactional Database specification

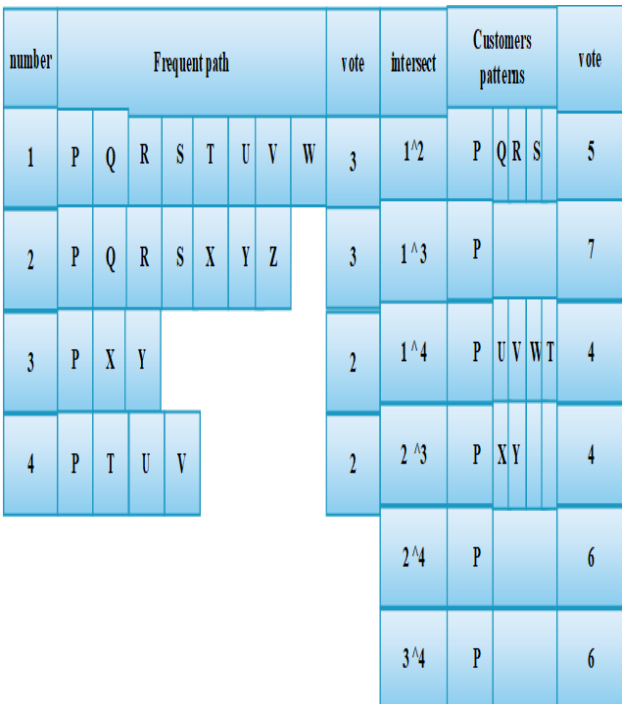

Figure 9: Process of frequent item

5.2 Performance Analysis

The quality of this model is calculated on the basis of validating the key metrics like accuracy, scalability with existing model multilevel parallel frequent item mining (MLPFIM) (Vanahalli, Manjunath K., and Nagamma Patil., 2020) Balanced Dispersed Parallel Frequent Colossal Closed Itemset Mining BDPFCCIM (Huang, Chun-Hong, and Yungho Leu., 2019]

5.2.1 Accuracy

The rate of accuracy count is validated as the accurate classification frequently used dataset. Thus the evaluations of accuracy measure with existing approaches are shown in **Figure 10** and **Table.1** where, $T'N$ is True Negative, $T'P$ for true positive, $F'N$ is false negative and $F'P$ is for false positive the calculation of accuracy is defined in equation 1.

$$Accuracy = \frac{(T'N + T'P)}{(T'N + T'P + F'N + F'P)} \text{----------(1)}$$

The obtained accuracy by the developed approach is 98% for 500 bytes data size. Simultaneously, MLPFIM achieved 92% of accuracy and BDPFCCIM pertained 90% of accuracy.

Table 1: Comparison of accuracy

Data size (bytes)	BDPFCCIM	MLPFIM	Proposed (GW-AFIM)
500	90	92	98
1000	89	90	97.8
1500	88.7	89.5	97.5
2000	88	89	97.3
2500	87.6	88	97

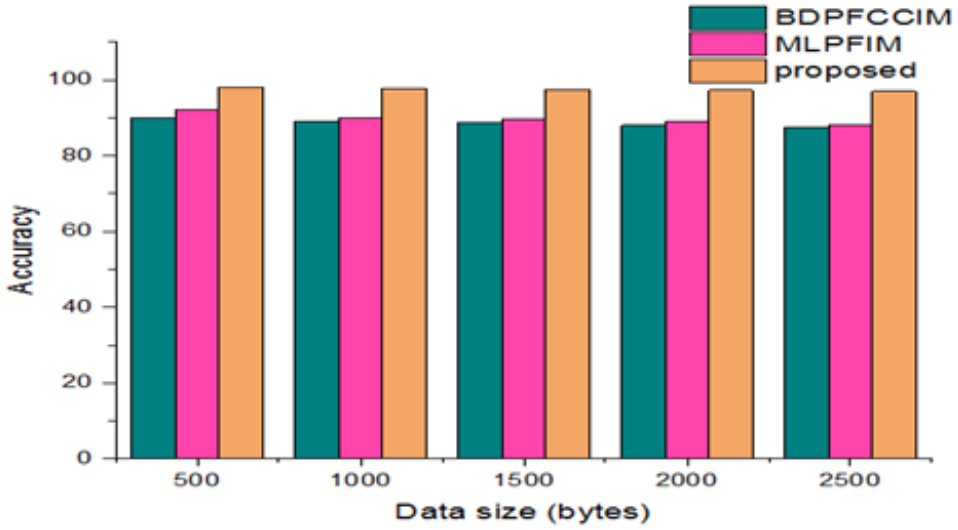

Figure 10: Accuracy Comparison

5.2.2 Scalability

For the load balancing model, scalability is the chief metric to evaluate the balancing capacity. Thus the assessments of scalability with other techniques are drawn in **Figure 11**. and in **Table 2**.
The obtained scalability rate by the developed approach is 50% for 500 bytes data size. Simultaneously, BDPFCCIM achieved 17% of accuracy and MLPFIM pertained 20% of accuracy.

Table 2: Scalability Comparison

Data size (bytes)	BDPFCCIM	MLPFIM	Proposed (GW-AFIM)
500	17	20	50
1000	16	18	46
1500	14	15	44
2000	10	12	40
2500	8	10	35

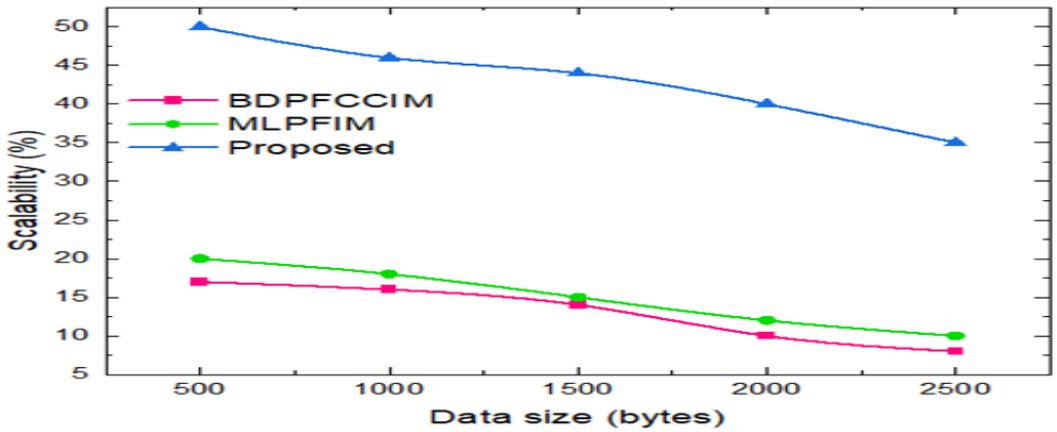

Figure 11: Comparison of Scalability

5.2.3 Run Time

The execution time for the process is termed as run time, an efficient load balancing algorithm can minimize more running time. The time utilized to execute the proposed model is 8ms for 500 bytes of data, simultaneously, MLPFIM consumed 23 ms of time to execute the process and BDPFCCIM needs 20 ms to execute the process. The comparison is detailed in **Table.3** and **Figure 12.**

Table 3: Comparison of Runtime

Data size (bytes)	BDPFCCIM	MLPFIM	Proposed (GW-AFIM)
500	20	23	8
1000	8	5.4	18
1500	7	5	15
2000	6	4	12
2500	5	3	10

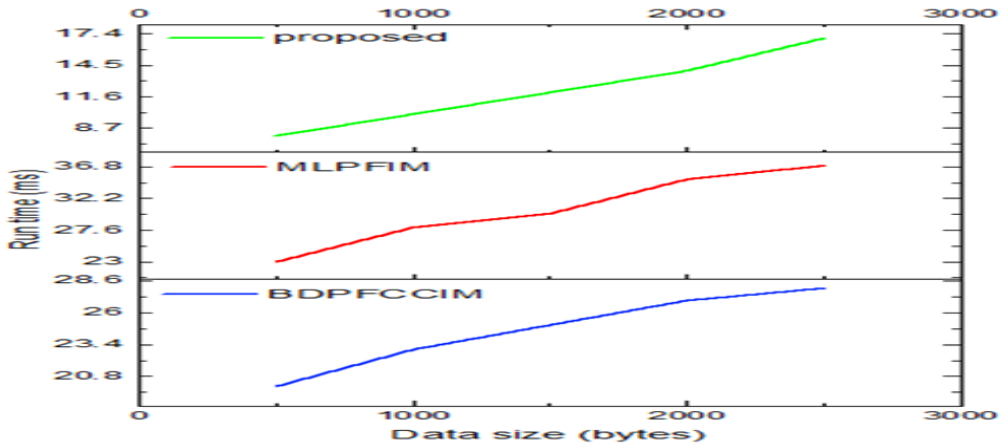

Figure 12: Estimation of Runtime (ms)

5.2.4 Speedup

The speedup is estimated with occurrence of data traffic, when the data load is increases the time taken for normal the process. Consequently, the execution time is evaluated, based on run time the speedup rate is calculated. The drawn estimation graph is detailed in **Figure 13** and **Table 4**.The measured speedup by the developed approach is 20ms for 500 bytes data size for BDPFCCIM. MLPFIM achieved 23ms of speedup measure and proposed approach pertained 9ms of speedup rate for 500 bytes of data.

Table 4: Comparison of Speed up Measures

Data size (bytes)	BDPFCCIM	MLPFIM	Proposed (GW-AFIM)
500	20	23	9
1000	23	28	10
1500	25	30	12
2000	27	35	14
2500	28	37	17

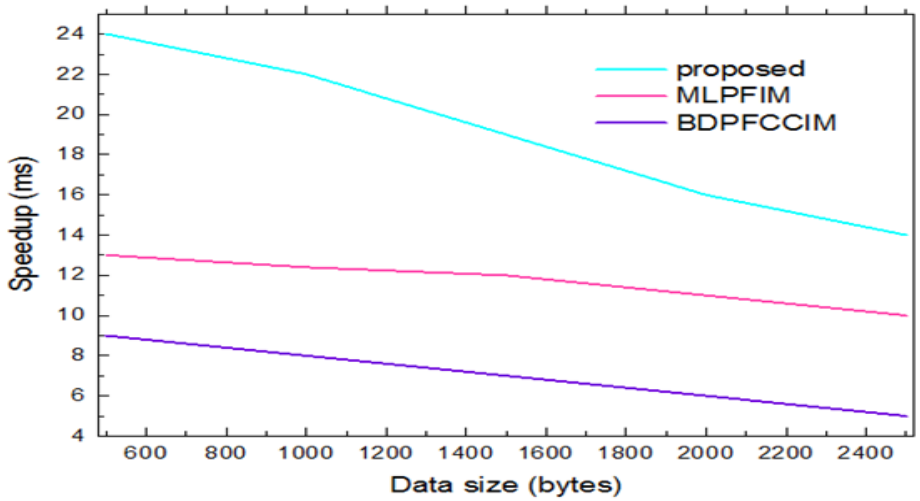

Figure 13: Speedup Comparison

5.2.5 Error Rate

The error rate is evaluated based on miss specification of frequently things. It is estimated as total frequent itemset by classified items, it is elaborated in equation 2. The elaborated validations shown in **Table 5** and Figure **14**.

Error rate= ((Classified frequent data-total frequent data)/ (Total frequent data))*100 ---------- **(2)**

The obtained error rate by the developed approach is 0.8% for 500 bytes data size. Simultaneously, MLPFIM achieved 2% of error measure and BDPFCCIM pertained 3% of error for 500 bytes of data.

Table 5: Comparison of Error Rate

Data size (bytes)	BDPFCCIM	MLPFIM	Proposed (GW-AFIM)
500	3	2	0.8
1000	3.3	2.3	1
1500	3.6	2.7	1.2
2000	3.9	3	1.4
2500	4.2	3.2	1.6

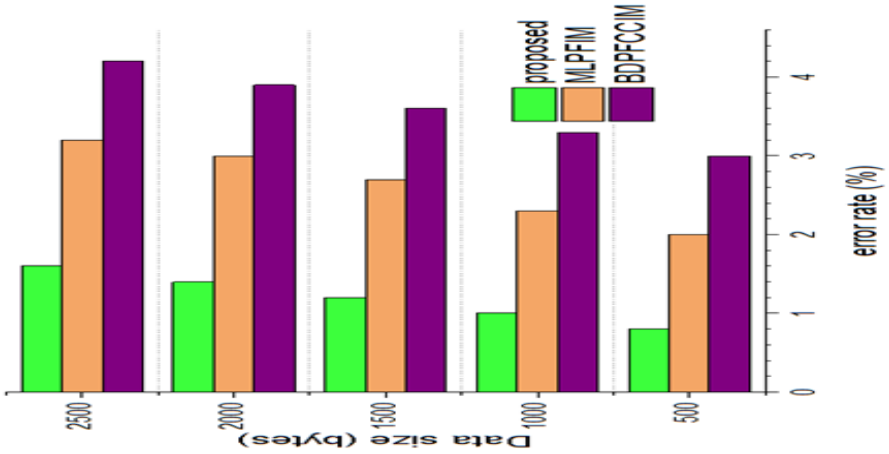

Figure 14: Error Rate Comparison

5.3 DISCUSSION

From all the key metrics evaluation, it is proved that the proposed model is applicable to balance the data load in large volume of transactional dataset. The overall performance of the developed model is shown in **Table 6** and in **Figure 15**. In that the proposed model attained 98% of maximum accuracy, 50% of scalability rate, 20ms of speed up time, 8ms for process execution and also it attained the minimum error rate as 0.8%.

Table 6: Overall Performance

Proposed: GW-AFIM					
Data size (bytes)	accuracy	scalability	speedup	Run time	Error rate
500	98	50	20	8	0.8
1000	97.8	46	18	10	1
1500	97.5	44	15	12	1.2
2000	97.3	40	12	14	1.4
2500	97	35	10	17	1.6

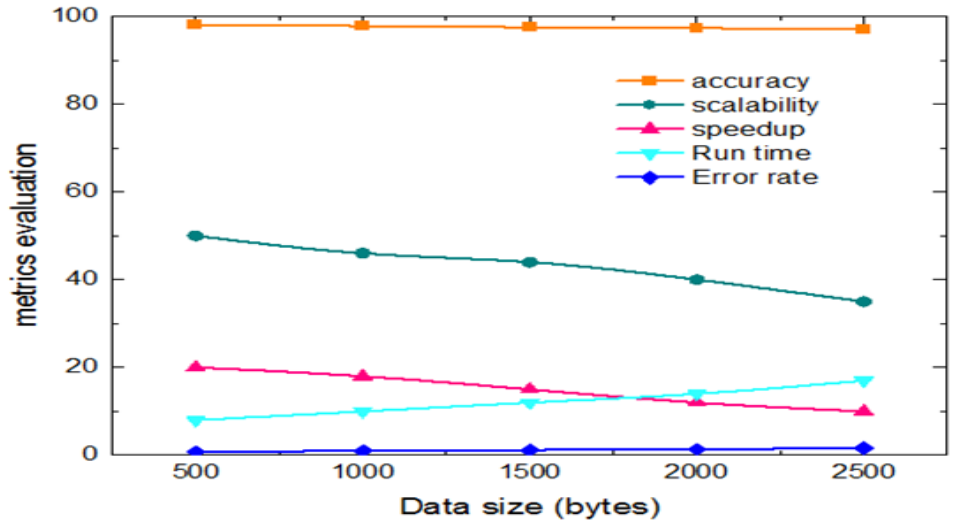

Figure 15: Overall Performance Comparison

6. CONCLUSION AND FUTURE ENHANCEMENT

The Graph-Weighted Adaptive Frequent item set Mining (GW-AFIM) model is a new approach to efficiently find itemsets that appear frequently in large-scale datasets. During big data transformations, this model reduces data traffic and proactively handles data load balancing issues. By incorporating a heuristic-based mechanism, which optimizes item selection through fitness evaluation, the identification of frequently used items is further refined. The model performs effectively, evidenced by experimental results indicating a high accuracy rate of 98% and a minimal error rate of merely 0.8%. Furthermore, the implementation of a dynamic load balancing method reduced the runtime by 8 milliseconds and enhanced scalability by 50%, rendering the model ideal for real-time applications in big data environments. Adaptive learning techniques may be used in future studies to improve heuristic parameters based on the properties of the data. Enhancing the GW-AFIM model's scalability with distributed cloud and edge computing frameworks may enhance performance in a range of scenarios. Mined frequent patterns can be made safer, more understandable, and more practical with the help of real-time visualization tools and privacy-preserving measures.

REFERENCES:

1. Xiao, Wen, and Juan Hu. "SWEclat: a frequent itemset mining algorithm over streaming data using Spark Streaming." The Journal of Supercomputing (2020), pp: 1-16.
2. Saleti, Sumalatha, and R. B. V. Subramanyam. "A Map Reduce solution for incremental mining of sequential patterns from big data." Expert Systems with Applications 133 (2019), pp. 109-125.
3. Osman, Ahmed M. Shahat. "A novel big data analytics framework for smart cities." Future Generation Computer Systems 91 (2019): pp. 620-633.
4. Cirillo, Davide, and Alfonso Valencia. "Big data analytics for personalized medicine." Current opinion in biotechnology 58 (2019), pp. 161-167.
5. Letras, Martín, et al. "On the Design of Hardware Architectures for Parallel Frequent Itemsets Mining." Expert Systems with Applications (2020).
6. Wei, Tianyou, et al. "FCHUIM: Efficient Frequent and Closed High-Utility Itemsets Mining." IEEE Access 8 (2020), pp. 109928-109939.
7. Wang, Chunxia, and Xiaoyue Zheng. "Application of improved time series Apriori algorithm by frequent itemsets in association rule data mining based on temporal constraint." Evolutionary Intelligence 13.1 (2020), pp. 39-49.
8. Fageeri, Sallam, Rohiza Ahmad, and Hitham Alhussian. "An Efficient Algorithm for Mining Frequent Itemsets and Association Rules." Implementations and Applications of Machine Learning. Springer, Cham, 2020.pp. 229-244.
9. Bhattacharya, Namrata, Sudip Mondal, and Sunirmal Khatua. "A MapReduce-Based Association Rule Mining Using Hadoop Cluster—An Application of Disease Analysis." Innovations in Computer Science and Engineering. Springer, Singapore, 2019, pp. 533-541.
10. Bai, S. Pavitra, and GK Ravi Kumar. "Subset Significance Threshold: An Effective Constraint Variable for Mining Significant Closed Frequent Itemsets." Emerging Technologies in Data Mining and Information Security. Springer, Singapore, 2019. pp. 449-458.
11. Rahimi, Zeinab, Samira Noferesti, and Mehrnoush Shamsfard. "Applying data mining and machine learning techniques for sentiment shifter identification." Language Resources and Evaluation 53.2 (2019), pp. 279-302.
12. Yasir, Muhammad, et al. "D-GENE: Deferring the GENEration of Power Sets for Discovering Frequent Itemsets in Sparse Big Data." IEEE Access 8 (2020), pp. 27375-27392.

13. Bagui, Sikha, Keerthi Devulapalli, and John Coffey. "A Heuristic Approach for Load Balancing the FP-Growth Algorithm on MapReduce." (2020).
14. Bustio-Martínez, Lázaro, et al. "A novel multi-core algorithm for frequent itemsets mining in data streams." Pattern Recognition Letters 125 (2019), pp. 241-248.
15. Oikawa, CR Anna Victoria, et al. "Adaptive Load Balancing based on Machine Learning for Iterative Parallel Applications." 2020 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). IEEE, 2020
16. Jaiswal, Dimple, and Michael Galloway. "Load Balancing of Financial Data Using Machine Learning and Cloud Analytics." 17th International Conference on Information Technology–New Generations (ITNG 2020). Springer, Cham, 2020.
17. Rupani, Kunal, et al. "Dynamic Load Balancing in Software-Defined Networks Using Machine Learning." Proceeding of International Conference on Computational Science and Applications. Springer, Singapore, 2020.
18. Van Huong, Pham, Hoang Van Hiep, and Nguyen Kim Khanh. "Improvement of feature set based on Apriori algorithm in Android malware classification using machine learning method." 2020 RIVF International Conference on Computing and Communication Technologies (RIVF). IEEE, 2020.
19. Wang, Chunxia, and Xiaoyue Zheng. "Application of improved time series Apriori algorithm by frequent itemsets in association rule data mining based on temporal constraint." Evolutionary Intelligence 13.1 (2020): 39-49.
20. Vojiř, Stanislav, et al. "EasyMiner. eu: Web framework for interpretable machine learning based on rules and frequent itemsets." Knowledge-Based Systems 150 (2018): 111-115.
21. Junaid, Muhammad, et al. "A Hybrid Model for Load Balancing in Cloud Using File Type Formatting." IEEE Access 8 (2020): 118135-118155.
22. Xu, Yue, et al. "Load balancing for ultradense networks: A deep reinforcement learning-based approach." IEEE Internet of Things Journal 6.6 (2019): 9399-9412.
23. Raj, Shashi, et al. "EAFIM: efficient apriori-based frequent itemset mining algorithm on Spark for big transactional data." KNOWLEDGE AND INFORMATION SYSTEMS (2020).
24. Vanahalli, Manjunath K., and Nagamma Patil. "Distributed load balancing frequent colossal closed itemset mining algorithm for high dimensional dataset." Journal of Parallel and Distributed Computing (2020).
25. Devi, S. Gayathri, and M. Sabrigiriraj. "Swarm intelligent based online feature selection (OFS) and weighted entropy frequent pattern mining (WEFPM) algorithm for big data analysis." Cluster Computing (2019): 1-13.