I. J. Computer Network and Information Security, **2023, 5, 81-95**
Published Online on October 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2023.05.08

Ford Fulkerson and Newey West Regression Based Dynamic Load Balancing in Cloud Computing for Data Communication

Prabhakara B. K.*
Department of Information Science and Engineering, A J Institute of Engineering and Technology, Mangaluru, Affiliated to VTU Belagavi, India
E-mail: prabhakarabk@ajiet.edu.in
ORCID iD: https://orcid.org/ 0000-0001-6001-7259
*Corresponding Author

Chandrakant Naikodi
Department of Studies and Research in Computer Science (PG), Davangere University, Davangere, India
E-mail: chandrakant.naikodi@davangereuniversity.ac.in
ORCID iD: https://orcid.org/ 0000-0001-60420-0448

Suresh L.
Department of Information Science and Engineering, RNS Institute of Technology, Bengaluru, Affiliated to VTU Belagavi, India
E-mail: drsureshl@rnsit.ac.in
ORCID iD: https://orcid.org/ 0000-0001-9513-1940

Received: 27 December 2022; Revised: 27 February 2023; Accepted: 06 May 2023; Published: 08 October 2023

Abstract: In Cloud Computing (CC) environment, load balancing refers to the process of optimizing resources of virtual machines. Load balancing in the CC environment is one of the analytical approaches utilized to ensure indistinguishable workload distribution and effective utilization of resources. This is because only by ensuring effective balance of dynamic workload results in higher user satisfaction and optimal allocation of resource, therefore improve cloud application performance. Moreover, a paramount objective of load balancing is task scheduling because surges in the number of clients utilizing cloud lead to inappropriate job scheduling. Hence, issues encircling task scheduling has to be addressed. In this work a method called, Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) in CC environment is proposed. The FF-NWRDLB method is split into two sections, namely, task scheduling and dynamic load balancing. First, Ford Fulkerson-based Task Scheduling is applied to the cloud user requested tasks obtained from Personal Cloud Dataset. Here, employing Ford Fulkerson function based on the flow of tasks, energy-efficient task scheduling is ensured. The execution of asymmetrical scientific applications can be smoothly influenced by an unbalanced workload distribution between computing resources. In this context load balancing signifies as one of the most significant solution to enhance utilization of resources. However, selecting the best accomplishing load balancing technique is not an insignificant piece of work. For example, selecting a load balancing model does not work in circumstances with dynamic behavior. In this context, a machine learning technique called, Newey West Regression-based dynamic load balancer is designed to balance the load in a dynamic manner at run time, therefore ensuring accurate data communication. The FF-NWRDLB method has been compared to recent algorithms that use the markov optimization and the prediction scheme to achieve load balancing. Our experimental results show that our proposed FF-NWRDLB method outperforms other state of the art schemes in terms of energy consumption, throughput, delay, bandwidth and task scheduling efficiency in CC environment.

Index Terms: Cloud Computing, Task Scheduling, Ford Fulkerson, Load Balancing, Machine Learning, Newey West Regression.

1. Introduction

Cloud Computing environment is a very sought after internet-based technique that bring forth both the resources and computer services on demand to customers with distinct requirements. Service provisioning are split into three types, namely, Infrastructure as a Service (IaaS), Software as a Service (SaaS) and Platform as a Service (PaaS). Load balancing (LB) refers to the workload allocation in a distributed fashion, in such a manner that the system resources are said to be loaded in an equal format, or, no resource is either said to be over-loaded or under-loaded. In this way, the entire performance of the system is said to be determined by performance metrics like the makespan, average response time, or total execution time is said to be enhanced significantly.

A fair task distribution method was designed in [1] with the purpose of enhancing the average response time and makespan in the CC environment. The fair task distribution was said to be achieved via finite state Markov process that possess the advantage of possessing a balance equation between states. By considering the balance state probabilities the anticipated virtual machine utilizations were analyzed that in turn played the major role in task allocation approach. Also, the Load Balancer (LBer) acted as the central server that utilized fair task allocation scheme for fair distribution of incoming tasks between the virtual machines, by considering both the current state and their processing potentialities. With this, makespan, average response time, resource utilization were said to be improved with lower degree of imbalance.

Despite improvement observed in several performance factors, energy consumption and delay involved during load balancing was not focused. To address on this aspect, in this work, first, Ford Fulkerson-based Task Scheduling algorithm is proposed that with the aid of Ford Fulkerson function based on the task flow, scheduling is performed that in turn reduces the delay involved.

Differences in provisioning of resource results in energy wastages, and moreover, it has negative influences on the Quality of Service (QoS), therefore generating huge violations as far as Service Level Agreement (SLA) is concerned. But, the cloud providers face with optimal resource management for distinct cloud workloads in heterogeneous environment. In [2], Predictive Priority-based Modified Heterogeneous Earliest Finish Time (PMHEFT) method was proposed to evaluate the required resource necessitate. In this work, an efficient and dynamic resource prediction model in a heterogamous environment was proposed to satisfy the user's requirements. With this type of predictive design the efficiency was said to be attained with improved makespan and power consumption.

Though significant amount of power consumption and makespan were improved with optimal resource management, however, the task scheduling efficiency along with the bandwidth consumption was not focused. To address on this aspect, in this work, Newey West Estimator and Linear Regression-based Dynamic Load Balancing algorithm is designed that with the aid of Newey West Estimator addresses the aspect of over-load by performing migration, therefore improving the task scheduling efficiency significantly.

Over the past few years, with an increase in online applications huge voluminous data are said to be accumulated extensively. Though paramount evolution of CC has been designed to handle such types of diversified data, still it faces several issues in real-time processing and load balancing of resources.

A dynamic load balancing algorithm was designed in [3] based on an election mechanism. With this the delay involved during the load balancing was found to be reduced. Global schedulers support dynamic workload without the necessity of off-line task-allocation. However, this type of schedulers outputs poor worst-case performance when employed in parallel with smart task splitting and partitioning mechanisms, therefore increasing run-time overhead. An approach to significantly schedule dynamic real-time workloads by employing semi-partitioned scheduling was presented in [4]. With this design complexity was said to be reduced.

Recent research exhibited that load balancing mechanisms designed on the basis of metaheuristics ensure better solutions for smooth scheduling and resource allocation in the CC environment. However, most of the prevailing methods only consider single or few QoS metrics and discard several paramount characteristics.

In [5], a load balancing algorithm, called, Data Files Type Formatting (DFTF) was proposed that employed novel Cat Swarm Optimization (CSO) in addition to support vector machine. Due to this novel optimization, several time factors involved in balancing load like, migration time, response time, optimization time and overhead time were found to be reduced significantly. Yet another method to reduce cost and delay involved in load balancing employing cooperative mechanism was proposed in [6].

The scalability of present day datacenter networks (DCNs) are said to be expanding rapidly owing to the extensive cloud service deployment. This is due to the surge in both the bandwidth consumption and DCN cost with the increase in the overall network size. Hence, there requires mechanisms in keeping the traffic balanced.

In [7], a port-based forwarding load balancing scheduling (PFLBS) mechanism for Fat-tree based DCNs with several new aspects that in turn addressed the limitations of the prevailing load balancing mechanisms was proposed. With this there not only saw a dip in the flow completion time but also resulted in the improvement of throughput. Yet another load balancing method integrating support vector machine (SVM) with adaptive genetic algorithm (AGA) was proposed in [8].

Motivated by the above papers, in this work, a method called, Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) in CC environment are proposed. The main objective of this research is to propose a new optimized task scheduled and dynamic load balancing model in CC environment that performs task scheduling and load balancing separately in an efficient manner. This optimization method addresses the limitations of the earlier load balancing methods in CC environment by its multi-objective model i.e., optimal task scheduling and dynamic load balancing.

1.1. Contributions of the Work

The contributions of this paper include the following.

- To propose a method called, Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) in CC environment which is a combination of Ford Fulkerson-based Task Scheduling and Newey West Regression-based dynamic load balancer model.
- To design a novel Ford Fulkerson-based Task Scheduling algorithm. During algorithm optimization, multi-objective optimization is carried out according to task flow separately for dependent and independent task so as to obtain significant cloud task-scheduling. Our model can improve the task scheduling efficiency, energy consumption and bandwidth via Ford Fulkerson functions.
- To present a Newey West Estimator and Linear Regression-based Dynamic Load Balancing model that ensures dynamic load balancing during run time according to volume and load of virtual machine in the CC environment.
- The proposed FF-NWRDLB method in CC has provided improved results for task scheduling efficiency, bandwidth, energy consumption, delay and throughput as performance evaluation measures.

1.2. Organization of the Paper

The rest of the paper is organized as: The discussion about numerous scheduling techniques in cloud computing environment is presented in Section 2. In Section 3, the proposed Multi-objective Auto-encoder Deep Neural Network-based (MA-DNN) method has been elaborated. Experimental setup for validating the MA-DNN method is provided in Section 4. Comparative analysis with an elaborate discussion is described in Section 5 and finally, the conclusions are presented in Section 6.

2. Related Works

Cloud computing latency is said to be high owing to the reason that it is far from the actual terminal users. Also, hence, the issue of load balancing between distinct nodes still requires to be addressed. In [9], a load balancing technique by allocating task on the basis of intermediary nodes was proposed. Also by classifying the nodes into light load, normal load and heavy load, completion time of task allocation was found to be reduced.

A review of cloud load balancing mechanisms was presented in [10]. An elaborate and holistic review pertaining to load balancing algorithms were investigated in [11]. The advantages and drawbacks of prevailing load balancing methods were also elaborated in detail with crucial issues being highlighted with the purpose of designing significant load balancing algorithms.

A survey on the intelligent load balancing method that have been designed in heterogeneous networks those based on the machine learning (ML) techniques were presented and investigated in [12]. Yet another systematic literature review to addressing response time and power consumption employing fault tolerant-based load balancing in CC environment were presented in [13]. Dynamic load balancing approaches were using optimization techniques were elaborated in detail in [14]. The need for fault tolerance in CC environment for load balancing to ensure efficient administration and minimize response time was detailed in [15].

A new task scheduling method called, Total Resource Execution Time Aware Algorithm (TRETA) was proposed in [16] that took into consideration the overall execution time in obtaining an optimal schedule. Taxonomy for the load balancing algorithms was presented in [17]. A token-based algorithm was proposed in [18] for dynamic load balancing employing Markov optimization. With this design the resource optimization was said to be achieved.

A novel multiple linear regression analysis was designed in [19] with the objective of reducing the computing time and improving the overall performance. Yet another optimal resource utilization model employing best-fit decreasing-based bin-packing was proposed in [20]. By utilizing this integrated mechanism, energy consumption and makespan along with resource utilization were said to be improved.

Motivated by the above materials in this work, a novel method called, Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) in CC environment is proposed. The elaborate description of FF-NWRDLB is provided in the following sections.

3. Methodology

In this section, an optimal dynamic load balancing method called, Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) in CC environment is proposed. The FF-NWRDLB method consists of two parts: a task scheduling model and dynamic load balancing. The former is responsible for the energy-efficient scheduling of incoming cloud user request tasks obtained from the Personal Cloud Dataset. The latter is responsible for the optimal load balancing by ensuring migration in case of over-load. Figure 1 shows the block diagram of Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) method.

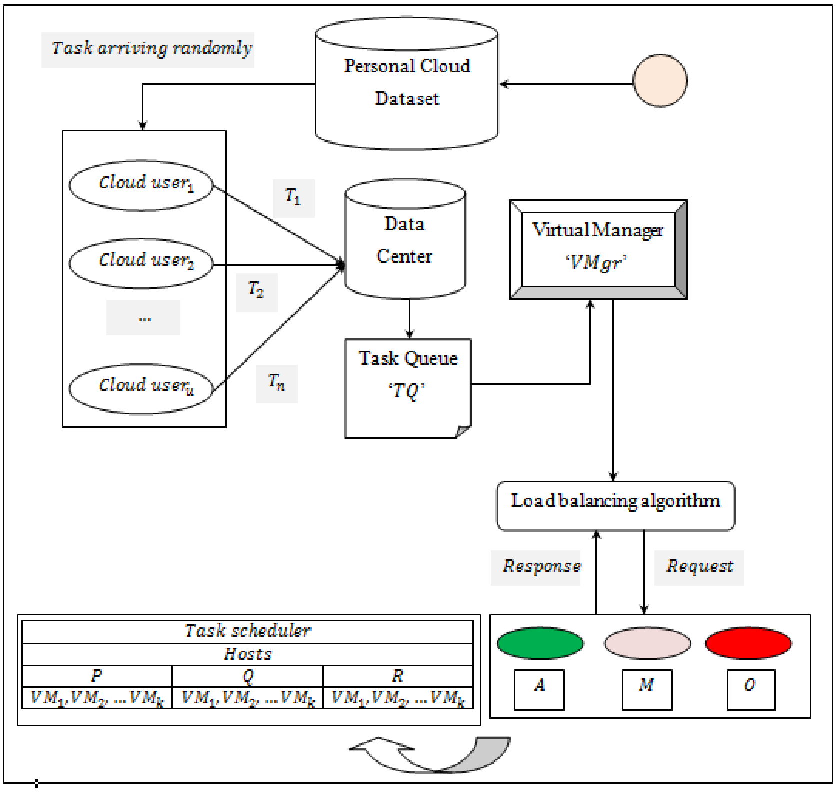

Fig.1. Block diagram of proposed method

As shown in the above Fig. 1. a clear picture of cloud services rendering for data communications based on demand and response process to create task scheduling and balance load optimally. On receiving the cloud user request (i.e., tasks) from the task queue '*TQ*', the Virtual Manager submits the tasks requests between the virtual machine '*VM*' to identify the VM availability. Upon successful identification of available '*VM*' by the Virtual Manager *'VMgr'*, then the available VM based on dependent tasks '*DT*' and independent tasks '*ID*' with fewer dependencies will be the answer to the cloud user requests demand, with representations of available *'A'*, overloaded *'O'* or migrated *'M'*. As a consequence, in our work, a machine learning algorithm called, Newey West Estimator and Linear Regression-based Dynamic Load Balancing to execute load balancing are designed. The elaborate description of the Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) method is provided in the following sections.

3.1. Ford Fulkerson-based Task Scheduling Model

In this section the problem definition and prerequisites for task scheduling in CC for our proposed method is presented. Let us consider a set of cloud user requested tasks '$T=\{T_1,T_2,\ldots T_n\}$' and a set of virtual machines '$VM=\{VM_1,VM_2,\ldots VM_k\}$' in the CC environment, denoted as weighted graph '$WG=(T, ES_t)$'. Here, 'ES_t' refers to the edge set with '$ES_t=\{(T_i,T_j)|$ *where task* T_j *is dependent on* $T_i\}$', '$W(T_i)$' denotes the independent tasks and '$W(T_i,T_j)$' represents the task 'T_j' dependent on task 'T_i' respectively. The task scheduling problem in CC environment can then be defined as the allocation of each cloud user requested tasks 'T_i' where '$T_i \in T$' to a suitable virtual machine 'VM_j' where '$VM_j \in VM$' to improve energy, throughput and minimum delay with maximum flow. Hence, with the objective of improving the energy consumption, throughput and delay with maximum flow (i.e., maximum flow of cloud user requested tasks) a Ford Fulkerson-based Task Scheduling model is designed. Fig.2 gives the structure of Ford Fulkerson-based Task Scheduling model.

As shown in the below Fig. 2., the instinct behind the design of Ford Fulkerson is that higher the task dependency on its parents (i.e., other tasks), lesser the bias for that task to be scheduled and simultaneously, parent task possessing higher number of dependent child tasks is provided with the higher bias and finally, bias is designated to the task to execute in which its utilization is maximum. Moreover, we want to identify the maximum flow between two tasks 'T_i' and 'T_j', i.e., while there exists a task queue 'TQ' from 'T_i' to 'T_j' such that '$c_f(T_i,T_j) > 0$' then, the flow of tasks obtained using Ford Fulkerson is mathematically formulated as given below.

$$c_f(T_i) = \min\{c_f(T_i, T_j) : (T_i, T_j) \in TQ\} \quad (1)$$

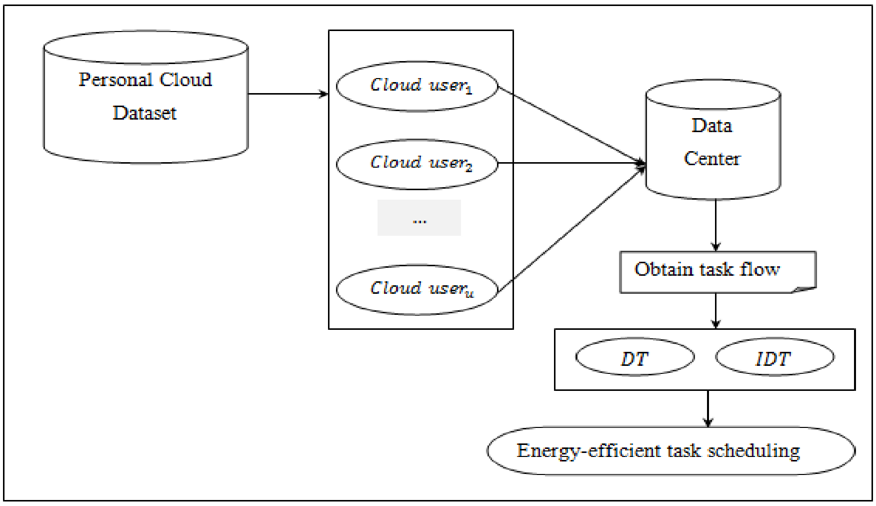

Fig.2. Structure of Ford Fulkerson-based Task Scheduling model

From the above equation (1), the computation flow of tasks '$c_f(T_i)$' is arrived at based on the minimum dependencies and maximum flow (i.e., subject to maximum numbers of inward cloud user requests). With the above maximum flow '$c_f(T_i)$' objective, the tasks scheduling for both dependent and independent tasks are executed separately as given below. The task scheduling with dependent task is mathematically formulated as given below.

$$DT[\eta(T_i, VM_k)] = \frac{1}{[W(T_i,T_j)*W(T_i,T_j)]} + W(T_j, T_j) \quad (2)$$

From the above equation (2), the dependent task scheduling '$DT[\eta(T_i, VM_k)]$' is modeled employing the dependency task of 'T_j' on 'T_i'. In a similar manner, the task scheduling with independent task is mathematically stated as given below.

$$IDT[\eta(T_i, VM_k)] = Inf * W(T_i, T_i) \quad (3)$$

From the above equation (3), the independent task scheduling '$IDT[\eta(T_i, VM_k)]$' is designed by utilizing very high value 'Inf' so that during the process of task selection, higher preference is said to be given to independent tasks. Now, employing the equations (2) and (3), the process of arriving at scheduling based on greedy model is formulated as given below.

$$S[DT] \rightarrow [\eta(T_i, VM_k)] = \sum_{i=1}^{n} T_i * [W(T_j, T_j)] \quad (4)$$

$$S[IDT] \rightarrow [\eta(T_i, VM_k)] = \sum_{i=1}^{n} T_i * [W(T_i, T_i)] \quad (5)$$

From the above equations (4) and (5), for every ordered pair '(T_i, VM_j)', where 'i' varies from '1 to n' and 'j' varies from '1 to k', the task scheduling performance metric is designed on the basis of '$\eta(T_i, VM_k)$'. Depending on the virtual machine availability and task dependencies, at most 'n' number of tasks is selected and scheduled at time intervals 'τ_0' and in a similar manner, remaining tasks are scheduled at time intervals '$\tau_1, \tau_2, \ldots, \tau_{max}$'. Here, '$\tau_{max}$' represents the maximum time interval at which all task execution are accomplished according to number of tasks and virtual machines respectively.

$$TSM \rightarrow S[DT] \cup S[IDT] \quad (6)$$

Finally, from the above equation (), tasks scheduled are listed in the form of matrix represented as 'TSM'. In this way, tasks are scheduled both for dependent and independent tasks with minimum energy consumption, delay and maximum throughput. The pseudo code representation of Ford Fulkerson-based Task Scheduling is given below.

Algorithm 1. Ford Fulkerson-based Task Scheduling

Input: Dataset 'DS', Task Scheduler 'TS', Virtual Manager '$VMgr$', Task Queue 'TQ', Cloud User '$CU = \{CU_1, CU_2, \ldots, CU_m\}$', Task '$T = \{T_1, T_2, \ldots, T_n\}$', Virtual Machine '$VM = \{VM_1, VM_2, \ldots, VM_k\}$'
Output: Energy-efficient task scheduler

1: Initialize 'm', 'n', 'k'
2: Begin
3: For each Dataset 'DS', Task Scheduler 'TS' Task Queue 'TQ', Cloud User 'CU', Task 'T'
4: Generate cloud user requested task flow via Ford Fulkerson as given in (1)
5: Formulate dependent task scheduling as given in (2)
6: Formulate independent task scheduling as given in (3)
7: Perform task scheduling based on greedy model as given in (4) and (5)
8: Return task scheduled results 'TSM' as given in (6)
9: End for
10: End

As given in the above algorithm, with the Personal Cloud Dataset provided as input, cloud user requested task flow are first obtained via Ford Fulkerson. With this Ford Fulkerson function measures the maximum possible flow (i.e., maximum cloud user requested tasks) in CC environment. In this manner, the delay involved in first reduced. Second, with the generated maximum possible flow of cloud user requested tasks, mathematical formulates for dependent task scheduling and independent task scheduling is performed separately. This in turn reduces the energy consumption involved in task scheduling. Finally, with the greedy approach, task scheduled results is obtained that in turn ensures bandwidth significantly.

3.2. Newey West Estimator and Linear Regression-based Dynamic Load Balancing

In accordance with the cloud infrastructure and the user requests, the CC environment is allocated with certain amount of load (i.e., either under-loaded, or over-loaded or balanced). Circumstances like under-loaded or over-loaded give rise to numerous system failure with regard to the power consumption, time factor delay and so on. Hence, dynamic load balancing is indispensable to control all stated issues. This dynamic load balancing of cloud user requested tasks on VMs is an important characteristic of task scheduling in CC environment. The dynamic load balancing of cloud user requested tasks measures the resource utilization and requirements at runtime in dynamic fashion and performs load balancing accordingly. In this work, a Newey West Estimator and Linear Regression-based Dynamic Load Balancing is designed. Figure 3 shows the structure of Newey West Estimator and Linear Regression-based Dynamic Load Balancing.

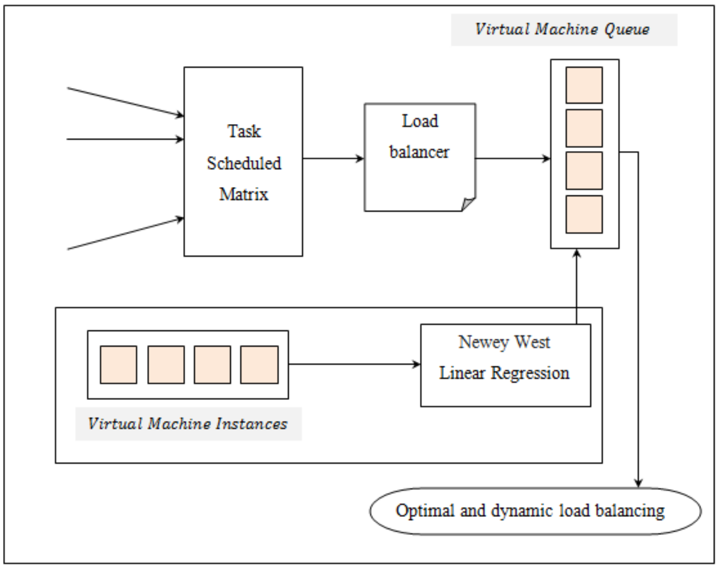

Fig.3. Structure of Newey West Estimator and Linear Regression-based Dynamic Load Balancing

As shown in the above Fig. 3., let us consider a CC environment hosting a set of virtual machines VMs 'VM' instances with each instance responsible for certain scheduled task 'TSM' and cooperatively work in conjunction maintaining the cloud user requests on the network. In a dynamic load balancer model as shown above, the task scheduler schedules the incoming requests and maintains a queue. The queue scheduling is then handled using linear regression function. By employing the linear regression model the resource queue is updated on the basis of the response time performance metric. The dynamic load balancer contains information about the 'k' virtual machine. First, volume of discrete virtual machine is measured and then, the volume of overall virtual machine is obtained as given below.

$$\mathrm{Vol}\,[\mathrm{VM}]_i = \mathrm{CPU}_{PS} * \mathrm{Number}\left(\mathrm{CPU}_{busy}\right) \tag{7}$$

$$\mathrm{Vol}\ [\mathrm{VM}] = \sum_{i=1}^{k} \mathrm{Vol}\ [\mathrm{VM}]_i \tag{8}$$

From the above equations (7) and (8), the volume of discrete virtual machine '$\mathrm{Vol}\ [\mathrm{VM}]_i$' and volume of overall virtual machine '$\mathrm{Vol}\ [\mathrm{VM}]$' is evaluated based on the processing speed of CPU 'CPU_{PS}' and number of CPU that are busy '$\mathrm{Number}\ (\mathrm{CPU}_{busy})$' respectively. Also every virtual machine possesses a queue called Virtual Queue to store the load. Here, the total span of queue in virtual machine denotes the load on that virtual machine. Then, the load at a virtual machine and total load is obtained as given below.

$$\mathrm{Load}\ [\mathrm{VM}]_{i,t} = [\mathrm{TSM}]_i * \frac{\mathrm{TS}[\mathrm{VM}]_{i,t}}{\mathrm{SR}[\mathrm{VM}]_{i,t}} \tag{9}$$

$$\mathrm{TLoad} = \sum_{i=1}^{k} \mathrm{Load}\ [\mathrm{VM}]_{i,t} \tag{10}$$

From the above equations (9) and (10), the load at a virtual machine '$\mathrm{Load}\ [\mathrm{VM}]_{i,t}$' and total load '$\mathrm{TLoad}$' is evaluated based on the task scheduled 'TSM', total span of queue in virtual machine '$\mathrm{TS}[\mathrm{VM}]_{i,t}$' at time '$\mathrm{t}$', service rate (i.e., based on CPU processing speed and number of CPU busy) of queue in virtual machine '$\mathrm{SR}[\mathrm{VM}]_{i,t}$' at time '$\mathrm{t}$' respectively. With the above volume and total load, line regression assumes that the correlation between the dependent variable 'Y' and the '$\mathrm{k} - \mathrm{vector}$' of regressor '$\mathrm{X}$' is linear. Here '$\mathrm{Y}$' denotes the load (i.e., over-load, under-loaded or balanced) and perform migration in case of over-load whereas 'X' represents the task scheduled 'TSM' along with the capacity and total load respectively. The linear regression function for dynamic load balancing is mathematically formulated as given below.

$$\mathrm{Y}_i = \begin{bmatrix} 1 & \mathrm{Vol}[\mathrm{VM}]_{11} & \mathrm{Vol}[\mathrm{VM}]_{12} & \dots & \mathrm{Vol}[\mathrm{VM}]_{1k} \\ 1 & \mathrm{Vol}[\mathrm{VM}]_{21} & \mathrm{Vol}[\mathrm{VM}]_{22} & \dots & \mathrm{Vol}[\mathrm{VM}]_{2k} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \mathrm{Vol}[\mathrm{VM}]_{n1} & \mathrm{Vol}[\mathrm{VM}]_{n2} & \dots & \mathrm{Vol}[\mathrm{VM}]_{nk} \end{bmatrix} \begin{bmatrix} \mathrm{TLoad}_1 \\ \mathrm{TLoad}_2 \\ \dots \\ \mathrm{TLoad}_k \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \dots \\ \epsilon_l \end{bmatrix} \tag{11}$$

From the above equation (11), dynamic load balancing is said to be optimized based on the volume '$\mathrm{Vol}[\mathrm{VM}]_{ij}$' and the total load '$\mathrm{TLoad}_k$' in addition to the error variance 'ϵ_l' respectively. The error variance 'ϵ_l' as said is not static owing to the address of dynamic load balancing during run time, here, Newey–West estimator is employed that according to the task scheduled 'TSM' obtained at that particular time instance 't' and available virtual machine (i.e., virtual machine found with under-loaded) is evaluated.

$$\mathrm{NWE}[\epsilon_l] = \mathrm{TSM}\ [\mathrm{Y}_i > 1] \tag{12}$$

With the above result the load balancing here is said to be optimized because of the learning of resource utilization only at the runtime. Accordingly, with the results obtained from 'Y_i', the number of virtual machine that are in under-loaded stage, or over-loaded stage or balanced are identified. In case of the virtual machine being over-loaded, migration is said to take place, therefore optimizing dynamic load balancing. The pseudo code representation of Newey West Estimator and Linear Regression-based Dynamic Load Balancing is given below.

Algorithm 2. Newey West Estimator and Linear Regression-based Dynamic Load Balancing

Input: Dataset 'DS', Task Scheduler 'TS', Virtual Manager 'VMgr', Task Queue 'TQ', Cloud User '$\mathrm{CU} = \{\mathrm{CU}_1, \mathrm{CU}_2, \dots, \mathrm{CU}_m\}$', Task '$\mathrm{T} = \{\mathrm{T}_1, \mathrm{T}_2, \dots, \mathrm{T}_n\}$', Virtual Machine '$\mathrm{VM} = \{\mathrm{VM}_1, \mathrm{VM}_2, \dots, \mathrm{VM}_k\}$'
Output: Dynamic and optimal load balancing

1: Initialize 'm', 'n', 'k'
2: Initialize task scheduled 'TSM'
3: Begin
4: For each Dataset 'DS', Task Scheduler 'TS' Task Queue 'TQ', Cloud User 'CU' and task scheduled 'TSM'
5: Measure volume of discrete virtual machine and volume of overall virtual machine as given in (7) and (8)
6: Measure load at a virtual machine and total load as given in (9) and (10)
7: Evaluate linear regression function as given in (11)
8: If '$\mathrm{Y}_i \geq 0$ and $\mathrm{Y}_i < 0.5$'
9: Then VM is under-loaded
10: Proceed with data communication
11: End if
12: If '$\mathrm{Y}_i \geq 0.5$ and $\mathrm{Y}_i < 1$'
13: Then VM is balanced
14: Proceed with data communication

15: End if
16: If '$Y_i \geq 1$'
17: Then VM is over-loaded
18: Select the VM for migration
19: End if
20: End for
21: End

As given in the above algorithm, with the task scheduled provided as input, first, volume of discrete virtual machine and volume of overall virtual machine are evaluated. Second, load at a virtual machine and total load is measured. Based on these two results, a Linear Regression function is applied with the objective of ensuring optimized load balancing. Moreover, Newey–West estimator is applied to arrive at the error variance that in turn ensures migration in case of virtual machine being over-load therefore not only improving throughput but with minimum data loss.

4. Experimental Setup

Experimental evaluation of the proposed Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) in CC environment carried out using Cloud Simulator via Native Java Codes. The CloudSim network simulation being a simulation and modeling tool performs cloud computing services on a distributed server with numerous Virtual Machines connected to it. The dataset is taken from Personal Cloud Datasets: NEC Personal Cloud Trace (http://cloudspaces.eu/results/datasets). The dataset comprises 17 attributes and 66245 instances. The 17 attributes are row id, account id, file size (i.e. task size), operation_time_start, operation_time_end, time zone, operation_id, operation type, bandwidth trace, node_ip, node_name, quoto_start, quoto_end, quoto_total (storage capacity), capped, failed and failure info. Among the 17 attributes, two columns are not used such as time zone and capped. Dataset is divided in to training dataset and testing dataset. Training dataset is 70% and testing dataset is 30%.Initially 150 data are taken as input for conducting the experiment , Then 300 data is taken for experimental analysis .similarly upto 1500 data are considered to validate the results of proposed work.

Experimental evaluation of proposed FF-NWRDLB method and two existing methods, namely Fair task distribution [1] and Predictive Priority-based Modified Heterogeneous Earliest Finish Time (PMHEFT) [2] are simulated using the different metrics such as task scheduling efficiency, energy consumption, bandwidth, throughput and delay.

5. Discussion

5.1. Case Scenario: Task Scheduling Efficiency

In this section the task scheduling efficiency results is provided. The efficiency of optimal and dynamic resource allocation can be measured on the basis of the number of tasks scheduled by the task scheduler in the task queue with respect to the cloud user requested tasks. Greater the task scheduling efficiency better is the methods performance to be. The task scheduling efficiency is mathematically formulated as given below.

$$TSE = \sum_{i=1}^{n} \frac{T_{CS}}{T_i} * 100 \tag{13}$$

Table 1. Comparison results of task scheduling efficiency using FF-NWRDLB, Fair task distribution [1] and PMHEFT [2]

Number of cloud user requested tasks	Task scheduling efficiency (%)		
	FF-NWRDLB	Fair task distribution	PMHEFT
150	94.66	90	86.66
300	93.25	89	85.25
450	93	88	84.15
600	92.55	87.25	84
750	91	87	83.25
900	90.55	86.55	83
1050	90	86.25	81.55
1200	88.35	85	80
1350	87	84.15	79
1500	85	82	77

From the above equation (13), the task scheduling efficiency 'TSE' is evaluated on the basis of the number of cloud user requested tasks 'T_i' placed in the task queue 'TQ' and the number of cloud user requested tasks correctly scheduled 'T_{CS}'. It is measured in terms of percentage (%). Different numbers of cloud user requested tasks ranging between 150 and 1500 are considered separately. The result of the simulation is listed in table 1. Figure 4 shows the comparison of task scheduling efficiency values of three different methods, FF-NWRDLB, Fair task distribution [1] and PMHEFT [2] respectively. It is observed that the FF-NWRDLB outperforms Fair task distribution [1] and PMHEFT [2].

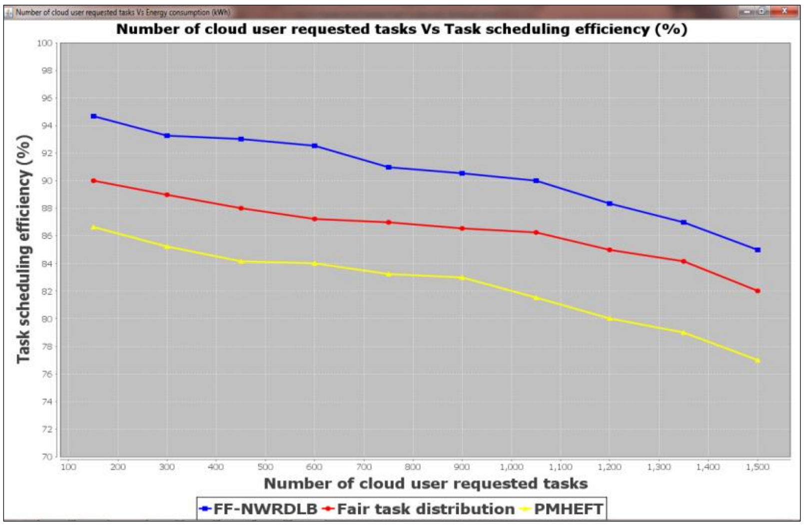

Fig.4. Graphical representation of task scheduling efficiency

Fig. 4. Above illustrates the task scheduling efficiency arrived at by distinct types of cloud user requested tasks when served at the CC environment using three distinct methods. The x-axis shows the number of cloud user requested tasks that generate cloud tasks to be processed, whereas the y-axis shows their corresponding task scheduling efficiency achieved in terms of percentage (%).It represents how distinct types of cloud user requested tasks consume corresponding tasks for performing dynamic load balancing. It is represented that the proposed FF-NWRDLB method exhibits higher improved task scheduling efficiency in the underlined environment when compared with [1,2]. The reason behind the improvement was due to the proper balance in host utilization between cloud users in the CC environment by employing Ford Fulkerson-based Task Scheduling. By applying this algorithm using the Ford Fulkerson function acquires global solution (i.e., optimal utilization of host ensuring task scheduling efficiency) irrespective of the cloud user request tasks in concern, therefore resulting in fast convergence between under-utilization and over-utilization. As a result, the task scheduling efficiency using FF-NWRDLB method is said to be improved by 5% compared to [1] and 10% compared to [2] respectively.

5.2. Case Scenario 2: Energy Consumption

In this section the energy consumption involved during virtual machine allocation for respective cloud user requested task is given. A significant amount of energy is said to be consumed during virtual machine allocation in the data center. Energy consumption is mathematically formulated as given below.

$$EC = \sum_{i=1}^{n} T_i * Energy(VM_{alloc}) \tag{14}$$

Table 2. Comparison results of energy consumption using FF-NWRDLB, Fair task distribution [1] and PMHEFT [2]

Number of cloud user requested tasks	Energy consumption (J)		
	FF-NWRDLB	Fair task distribution	PMHEFT
150	52.5	61.5	72
300	58.35	73.55	80.15
450	65.25	85.25	90.35
600	75	93.15	103.25
750	78.25	100.25	115.55
900	85.55	115.35	130.35
1050	90	125.25	145.55
1200	105.35	135.55	152.35
1350	125.55	150	160
1500	138.35	168.25	180

From the above equation (14), energy consumption 'EC' is measured on the basis of the number of cloud user requested tasks involved in simulation 'T_i' and the actual energy consumed during virtual machine allocation '$\text{Energy}(VM_{alloc})$'. It is measured in terms of joules (J). Number of cloud user requested tasks ranging between 150 and 1500 are considered during the simulation of task scheduling and dynamic load balancing for efficient data communication. Only after the scheduled tasks, the dynamic load balancing is performed for significant data communication. The result of the simulation for energy consumption is listed in table 2. Figure 5 illustrates the comparison of mean values conducted for 10 different simulation runs of three different methods. It is observed that the FF-NWRDLB method outperforms Fair task distribution [1] and PMHEFT [2].

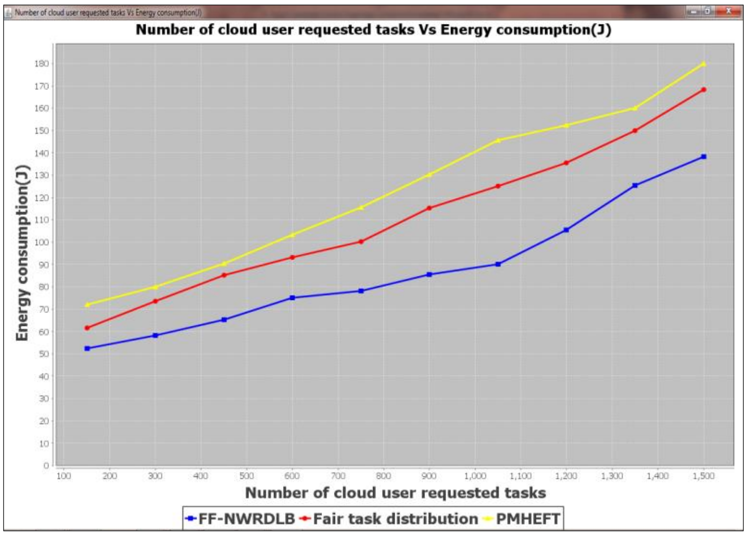

Fig.5. Graphical representation of energy consumption

Fig.5. given above shows the energy consumption or the energy consumed while allocation virtual machine dynamically in CC environment even in case of distinct constraints between resources of the cloud. From the above figurative representation, the energy consumption is found to be in the increasing trend. In other words, increasing the number of cloud user requested tasks results in a proportionate increase in consumption of energy for dynamic virtual machine allocation. Simulation performed with 150 cloud user requested tasks saw 52.5J of energy being consumed using FF-NWRDLB, 61.5J using [1] and 72J using [2] respectively. The energy consumed in the process of dynamic load balancing for virtual machine allocation using FF-NWRDLB method was observed to be comparatively lesser than [1,2]. The reason behind the improvement was due to the application of Ford Fulkerson-based Task Scheduling algorithm. By applying this algorithm with cloud user requested tasks being scheduled provided as input, maximum possible flow or maximum cloud user requested tasks in CC environment was formulated. Next, with the formulated maximum possible flow of cloud user requested tasks, dependent task scheduling and independent task scheduling was modeled separately. This in turn minimized the energy consumption using FF-NWRDLB method by 21% compared to [1] and 29% compared to [2].

5.3. Case Scenario 3: Bandwidth

Bandwidth refers to the data transfer capacity of a computer network and it is measured in terms of bits per second (Bps). Here, bandwidth refers to capacity whereas throughput specifies how much data actually is said to be transmitted. In other words, bandwidth represents the amount of data communication occurred over CC environment at a specified time instance. In addition it denotes the speed of CC network and not the fastness in which the data are moving between two ends or simply between cloud users. Higher the bandwidth, robust task scheduling and dynamic load balancing are made and therefore more numbers of cloud users data are said to be transmitted, and vice versa. To be more specific, bandwidth is measured as the amount of data that can be transferred from one point (i.e., between cloud users) to another within a CC network at a specific amount of time during load balancing Table 3 given below lists the bandwidth measure results obtained using three distinct methods, FF-NWRDLB method, Fair task distribution [1] and Predictive Priority-based Modified Heterogeneous Earliest Finish Time (PMHEFT) [2] respectively.

As illustrated in Fig.6., the bandwidth is a significant performance metric to judge whether the proposed method is desirable or not. Higher value of the bandwidth appropriates optimal dynamic load balancing between cloud users, therefore larger amount of information or data communication is said to be performed at a specific time instance between cloud users. Based on the comparison with FF-NWRDLB method, Fair task distribution [1] and PMHEFT [2], in figure, the outcome of the bandwidth is illustrated. Obviously, when the number of cloud user requested tasks is 1500, the difference between different methods is inconspicuous. On the other hand, with the increasing of number of cloud user requested tasks, the difference between these three methods becomes more prominent that infers FF-NWRDLB method perform better than the other two methods. The reason behind the improvement is due to the application of Ford

Fulkerson-based Task Scheduling. By applying this algorithm, according to the virtual machine availability and task dependencies scheduling of tasks were made that in turn improved the data transfer capacity significantly, therefore ensuring dynamic load balancing in an optimal fashion. As a result, the bandwidth rate using FF-NWRDLB method was found to be comparatively better by 12% compared to [1] and 30% compared to [2] respectively.

Table 3. Comparison results of bandwidth using FF-NWRDLB method, Fair task distribution [1] and PMHEFT [2]

Number of tasks	Bandwidth (Mbps)		
	FF-NWRDLB	Fair task distribution	PMHEFT
5000	20	17	18
10000	23	22	9
15000	25	21	20
20000	28	27	22
25000	31	28	23
30000	35	31	25
35000	38	33	27
40000	40	35	29
45000	42	38	32
50000	45	40	35

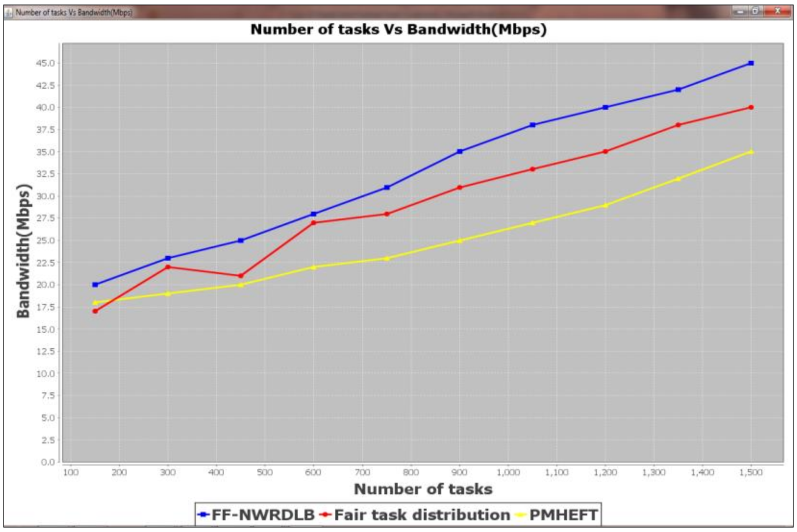

Fig.6. Graphical representation of bandwidth

5.4. *Case Scenario 4: Throughput*

Throughput 'T' indicates how many cloud user requests tasks 'T_i' a virtual machine 'VM_j' executes per unit time. In other words, the value of throughput defines the effectiveness of the overall system. High performance or high throughput indicates better system efficiency. It is measured in terms of percentage (%).

$$Th = \sum_{i=1}^{n} \frac{T_i}{MS} \tag{15}$$

From the above equation (15), the rate of throughput 'Th' is measured based on the number of tasks involved in the simulation process 'T_i' and the makespan 'MS'. It is measured in terms of bits per second (bps). The result of the simulation is listed in table 4. Figure 7 shows the comparison of throughput values of three distinct methods, FF-NWRDLB method, Fair task distribution [1] and Predictive Priority-based Modified Heterogeneous Earliest Finish Time (PMHEFT) [2] respectively.

Fig.7 given below illustrates the throughput rate with respect to 1500 distinct cloud user requested tasks. From the above figure, x axis denotes the number of cloud user requested tasks ranging between 1500 and 1500 and y axis represents the throughput rate measured in terms of bits per second (bps). Also from the above figure it is identified that by increasing the number of cloud user requested tasks results in the increase in the number of cloud users to be concentrated by the load balancer cloud server to overcome the point at issues regarding, over-loading, under-loading and migration in case of over-loading. In all the three methods, increasing the number of cloud user requested tasks results in the increase of throughput also. However, a significant improvement is observed using FF-NWRDLB method. This is because of the application of Newey West Estimator and Linear Regression-based Dynamic Load Balancing algorithm. By applying this algorithm, individual optima (i.e., discrete virtual machine) and overall optima (i.e., volume of overall virtual machine) are obtained for each objective separately by means of linear regression function via load balancer. Then, the load at a virtual machine and total load is evaluated to perform mapping that in turn results in the convergence of obtaining solution set, therefore improving the makespan. As a result, the throughput rate using FF-NWRDLB method is said to be improved by 21% compared to [1] and 48% compared to [2] respectively.

Table 4. Comparison results of throughput using FF-NWRDLB method, Fair task distribution [1] and PMHEFT [2]

Number of cloud user requested tasks	Throughput (bps)		
	FF-NWRDLB	Fair task distribution	PMHEFT
150	12.5	10	8.82
300	13.85	11.55	9.25
450	14.25	12.35	10.55
600	15.35	13.85	11
750	18.25	15.35	11.85
900	20	16.15	13
1050	20.85	18	14.35
120	25	20.25	15.85
1350	28	21.45	17
1500	31	24.35	22

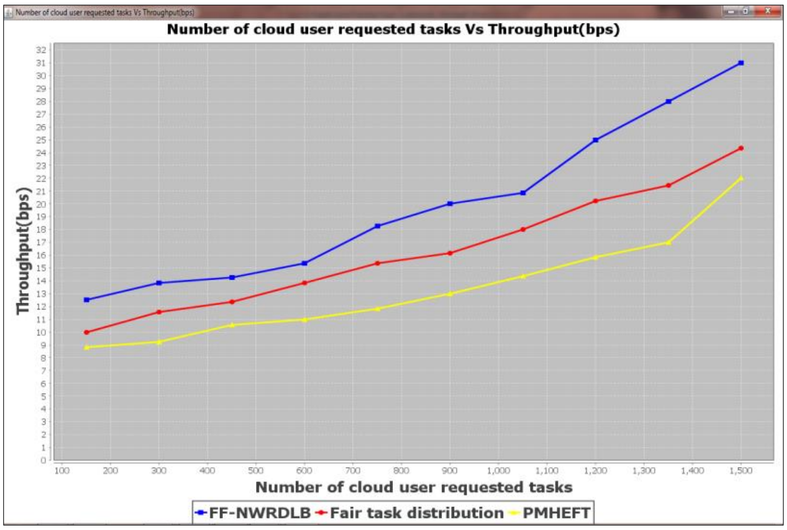

Fig.7. Graphical representation of throughput

5.5. *Case Scenario 5: Delay*

Delay in our work involves transmission delay, '$Trans_d$', propagation delay '$Prop_d$' and the queuing delay '$Queue_d$' incurred during the execution of service requests. It is measured in terms of milliseconds (ms).

$$D = \sum_{i=1}^{n} T_i * [Time\ (Trans_d) + (Prop_d) + (Queue_d)\ ] \quad (16)$$

From the above equation (16), the performance metric delay 'D' is measured based on the delay involved during transmission '$Trans_d$', delay involved during propagation '$Prop_d$' and delay involved in placing the cloud user requests in the task queue '$Queue_d$' respectively. It is measured in terms of milliseconds (ms). Finally, the result of the simulation for delay is listed in table 5.

Fig.8 given below shows the evaluation of performance metrics of underlined task scheduling and dynamic load balancing in CC environment with respect to delay. The performance metric delay is considered as the most critical specifically for dynamic load balancing in CC environment. The abovementioned metric is evaluated in terms of milliseconds. It shows that the time consumed in scheduling cloud user requested task and correspondingly balancing the load in a dynamic fashion during run time contributes in the total time of serving requests. However, time consumed in scheduling cloud user request and balancing of load in dynamic fashion is playing a major role in average response time, i.e., delay. This is owing to the forwarding of user requested tasks towards the adjacent users. In a performance point of view, lower values of all abovementioned delay performance metrics are preferred for better service provisioning and on the other hand, higher makespan may degrade and hence deteriorate the entire performance. The delay was found to be comparatively lesser using FF-NWRDLB method than [1,2]. The reason behind the improvement was owing to the application of Newey West Estimator and Linear Regression-based Dynamic Load Balancing algorithm where mapping tasks with the corresponding load was made by employing Newey West Estimator function. By employing this function resolves load domain information involved in mapping, therefore combining these two characterization together results in the minimization of delay using FF-NWRDLB method by 34% compared to [1] and 39% compared to [2] respectively.

Table 5. Comparison results of delay using FF-NWRDLB method, Fair task distribution [1] and PMHEFT [2]

Number of cloud user requested tasks	Delay (ms)		
	FF-NWRDLB	Fair task distribution	PMHEFT
150	184.5	336	346
300	205.35	365	385.25
450	225.15	390	415.35
600	240.85	410.25	445.85
750	295.35	425.55	510.35
900	315.55	485.35	525.15
1050	335.85	525.15	575.55
1200	415.25	580.25	610.35
1350	485.55	615.35	655.25
1500	525.35	630	675

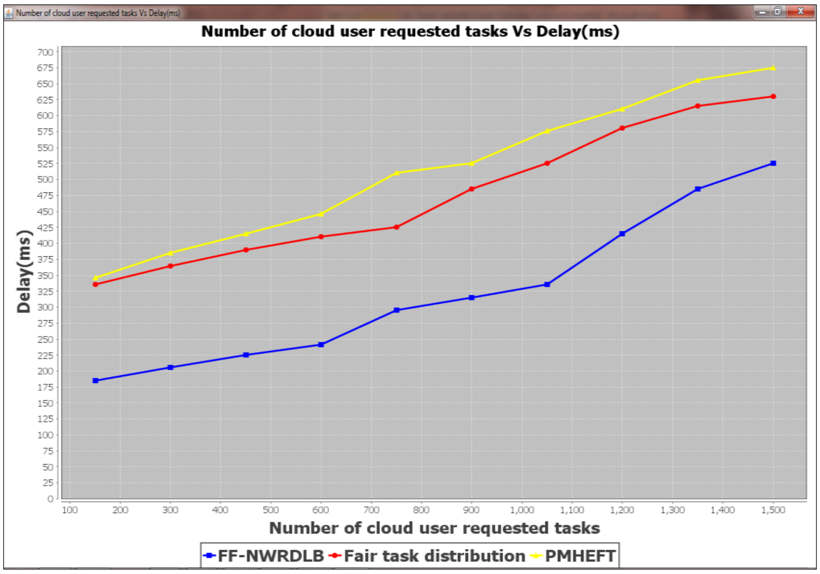

Fig.8. Graphical representation of delay

6. Conclusions

Due to several factors taking into consideration during task scheduling and involvement of distinct constraints during dynamic load balancing, scheduling efficiency and energy consumption in the data center remains to be focused. Also, owing to the reason that similar cloud user request tasks types and resources have to be allocated in an optimal and dynamic fashion, hence task scheduling between cloud user requests is required to reduce task scheduling time, energy consumption and ensure fast processing. In this paper, a method called, Ford Fulkerson and Newey West Regression-based Dynamic Load Balancing (FF-NWRDLB) in CC environment is proposed. The proposed task scheduling and dynamic load balancing method provides information of the cloud user requested tasks to the task scheduler and load balancer by processing the generated data and balancing load accordingly during runtime. After processing, a Ford Fulkerson-based Task Scheduling is designed that schedules the tasks in a computationally efficient manner even in case of distinct flow types by means of Ford Fulkerson function. Finally, employing Newey West Regression-based dynamic load balancer, optimal and dynamic load balancing is ensured. Simulations were performed to evaluate the performance of FF-NWRDLB, Fair task distribution, and PMHEFT for cloud service provisioning. Simulation results revealed that the proposed FF-NWRDLB method out performs Fair task distribution, and PMHEFT for cloud service provisioning implementations, in terms of 8% task scheduling efficiency, 25% energy consumption, 21% throughput, 30% bandwidth and 37 % delay.

References

[1] Stavros Souravlas, Sofia D. Anastasiadou, Nicoleta Tantalak, Stefanos Katsavounis, "A Fair, Dynamic Load Balanced Task Distribution Strategy for Heterogeneous Cloud Platforms Based on Markov Process Modeling", IEEE Access, Mar 2022

[2] Mayank Sohani, S. C. Jain, "A Predictive Priority-Based Dynamic Resource Provisioning Scheme with Load Balancing in Heterogeneous Cloud Computing", IEEE Access, Apr 2021

[3] Xiaoxun Zhong, Lianming Zhang, Yehua Wei, "Dynamic Load-Balancing Vertical Control for a Large-Scale Software-Defined Internet of Things", IEEE Things, Jul 2019

[4] Daniel Casini, Alessandro Biondi, Giorgio Buttazzo, "Task Splitting and Load Balancing of Dynamic Real-Time Workloads for Semi-Partitioned EDF", IEEE Transactions on Computers, Oct 2020

[5] Muhammad Junaid, Adnan Sohail, Rao Naveed Bin Rais, Adeel Ahmed, Osman Khalid, Imran Ali Khan, Syed Sajid Hussain, Naveed Ejaz, "Modeling an Optimized Approach for Load Balancing in Cloud", IEEE Access, Oct 2020

[6] Mirza Mohd Shahriar Maswood, Md. Rahinur Rahman, Abdullah G. Alharbi, Deep Medhi, "A Novel Strategy to Achieve Bandwidth Cost Reduction and Load Balancing in a Cooperative Three-Layer Fog-Cloud Computing Environment", IEEE Access, Jun 2020

[7] Zhiyu Liu, Aqun Zhao and Mangui Liang, "A port-based forwarding load-balancing scheduling approach for cloud datacenter networks", Journal of Cloud Computing: Advances, Systems and Applications, Nov 2021

[8] Eswaran Sivaraman and R. Manickachezian, "Unevenness measurement using the support vector machine and dynamic multiservice load balancing with modified genetic algorithm in cloud-based multimedia system", International Journal of Computer Aided Engineering and Technology, Vol. 10, No. 6, Nov 2018

[9] Guangshun Li, Yonghui Yao, Junhua Wu, Xiaoxiao Liu, Xiaofei Sheng and Qingyan Lin, "A new load balancing strategy by task allocation in edge computing based on intermediary nodes", EURASIP Journal onWireless Communications and Networking, Springer, Jul 2020

[10] Dalia Abdulkareem Shafiq, N.Z. Jhanjhi, Azween Abdullah, "Load balancing techniques in cloud computing environment: A review", Journal of King Saud University – Computer and Information Sciences, Elsevier, Mar 2021

[11] Shahbaz Afzal and G. Kavitha, "Load balancing in cloud computing – A hierarchical taxonomical classification", Journal of Cloud Computing: Advances, Systems and Applications, Jul 2019

[12] Emre Gures, Ibraheem Shayea, Mustafa Ergen, Marwan Hadri Azmi and Ayman A. EL-SALEH, "Machine Learning-Based Load Balancing Algorithms in Future Heterogeneous Networks: A Survey", IEEE Access, Apr 2022

[13] Vahid Mohammadian, Nima Jafari Navimipour, Mehdi Hosseinzadeh, Aso Darwesh, "Fault-Tolerant Load Balancing in Cloud Computing: A Systematic Literature Review", IEEE Access, Dec 2021

[14] Tawfeed Mohmmed Tawfeeg, Adil Yousif, Alzubair Hassan, Samar M. Alqhtani, Rafik Hamza, Mohammed Bakri Bashir, Awad Ali, "Cloud Dynamic Load Balancing and Reactive Fault Tolerance Techniques: A Systematic Literature Review (SLR)", IEEE Access, Jul 2022

[15] Muhammad Asim Shahid, Noman Islam, Muhammad Mansoor Alam, Mazliham Mohd Suud, Shahrulniza Musa, "A Comprehensive Study of Load Balancing Approaches in the Cloud Computing Environment and a Novel Fault Tolerance Approach", IEEE Access, Jul 2020

[16] K.P.N Jayasena, K.M.S.U Bandaranayake, y, B. T. G. S. Kumara, "TRETA - A Novel Heuristic Based Efficient Task Scheduling Algorithm in Cloud Environment", IEEE Xplore, Jun 2021

[17] Sambit Kumar Mishra, Bibhudatta Sahoo, Priti Paramita Parida, "Load balancing in cloud computing: A big picture", Journal of King Saud University – Computer and Information Sciences, Elsevier, Feb 2018

[18] C´eline Comtea, "Dynamic Load Balancing with Tokens", Computer Communications, Elsevier, Sep 2019

[19] Dong-sheng Liu and Xiao-hong Xiao, "Load balancing algorithm based on multiple linear regression analysis in multi-agent systems", International Journal of Computational Science and Engineering, Vol. 16, No. 3, 2018

[20] Neha Gupta, Kamali Gupta, Deepali Gupta, Sapna Juneja, Hamza Turabieh, Gaurav Dhiman, Sandeep Kautish, and Wattana Viriyasitavat, "Enhanced Virtualization-Based Dynamic Bin-Packing Optimized Energy Management Solution for Heterogeneous Clouds", Mathematical Problems in Engineering, Hindawi, Jan 2022

Authors' Profiles

Prof. Prabhakara B. K., Received the Computer Science and Engineering, Bachelor of Engineering degree from Bangalore University, India in 2000 and M. Tech in Computer Science and Engineering from Visvesvaraya University, India in 2006. He is currently Associate Professor in Department of Information Science and Engineering, A J Institute of Engineering and Technology, Mangaluru 575006. He has 22 years of teaching experience. He is served in various capacities in reputed engineering colleges. His research interest includes Cloud Computing and Data Analytics. He can be contacted at email: prabhakarabk@ajiet.edu.in

Dr. Chandrakant Naikodi, Associate Professor, Department of Studies and Research in Computer Science (PG), Davangere University, Davangere, India 577002. He has authored or coauthored more than 50 refereed journal and conference papers. He is author of programming e-books like C, C++, Java and Python. His research interest is Computer Networks, Big Data, IoT. He can be contacted at email: chandrakant.naikodi@davangereuniversity.ac.in

Dr. Suresh L., Professor and Head, Department of Information Science and Engineering, RNS Institute of Technology, Bengaluru, India. He has vast experience of 31 years in the profession, as teacher and administrator. He received best paper awards in IEEE and Science Direct conferences. He has been invited as keynote speaker for the IEEE conferences held in Singapore and Poland. He has authored or coauthored more than 40 refereed journal and conference papers. His research interest is Computer Networks, Big Data, Data Analytics. He can be contacted at email: drsureshl@rnsit.ac.in

How to cite this paper: Prabhakara B. K., Chandrakant Naikodi, Suresh L., "Ford Fulkerson and Newey West Regression Based Dynamic Load Balancing in Cloud Computing for Data Communication", International Journal of Computer Network and Information Security(IJCNIS), Vol.15, No.5, pp.81-95, 2023. DOI:10.5815/ijcnis.2023.05.08