International Journal of Future Generation Communication and Networking
Vol. 13, No. 4, (2020), pp.4617–4631

Energy Aware Resource Allocation Based On Optimization and Minimum Migration Time

S.Rekha[1], Dr.C.Kalaiselvi[2]

[1]Ph.D. Research Scholar, Department of computer science, Tirupur kumaran college for women. Assistant Professor, Department of IT, Dr.N.G.P. Arts and Science college.
E-mail: rekhaishanth@gmail.com
[2]Head and Associate professor, Department of Computer Applications, Tirupur kumaran college for women,
E-mail: kalaic29@gmail.com

Abstract

Recently Cloud Computing is an emerging arena of research where virtual machine has an eminent role for affording service to the clients through computing resources all over the world on demand basis. Virtual machine (VM) availability awareness is one of the prominent tasks scheduling technology which is greatly utilized in cloud platform. The major challenge involved in VM availability is that dynamic change and uncertainty involved in task scheduling along with task services quality necessities which in turn extremely impacts on cloud task scheduling capability. The optimal scheduling currently utilizes Multi-level queue scheduling Particle Swarm Optimization algorithm (MQPSO) while it does not focus on host utility detection. Various failures such as VM failure, Connection failure, and Response failure may occur due to the greater exploitation of host which in turn causes greater energy consumption. Hence improved cuckoo search optimization is greatly involved for Multi-level queue scheduling through validating the combination of Shortest-Job-First (SJF) buffering in addition to Min-Min Best Fit (MMBF) scheduling algorithms (SJF-MMBF). Adaptive Neuro Fuzzy Inference System (ANFIS) algorithm is additionally utilized for over utilized host detection and Virtual Machines (VMs) migration from over-utilized hosts to the other hosts is accomplished and thereby mitigating energy consumption. The various assessment parameters, like Throughput, Delay, Cost, and Energy consumption are assessed for the projected load balanced task scheduling approach by means of Experimental analysis and compared with the prevailing approaches.

Keywords: *Task scheduling, Particle Swarm Optimization, Energy consumption, cuckoo search optimization, over utilized host and Virtual Machines migration.*

1. Introduction

Progressively more computing tasks have been performed on virtual machine (VM) resources in the cloud platform, by development and implementation of cloud computing. The Quality of Service (QoS) relies on the quantity of VM resources obtainable. A significant way of enhancing task processing efficiency of the VM in the cloud platform and fulfilling QoS needs is a predominant task scheduling algorithm entitled as VM accessibility awareness. To determine whether the accessibility of VM resources can fulfil task QoS demands, has been the prime goal of VM availability awareness. The task scheduling will be implemented to enhances the efficiency of application programs operating in the VM system on the basis of this assessment [1,2].

Nevertheless, the conventional assessment of computer system accessibility and resource workload is not able to fulfil the active environment of the cloud platform, on considering of the dynamics and complexity of accessibility of VM resources in the cloud. The prime issue that demands to be solved immediately, in the scheduling process is how to merge accessibility assessment of VM resources and dynamic server workloads. Through the Particle Swarm Optimization algorithm (MQPSO), multi-level queue scheduling is executed in the recent research works [3, 4, 5].

ISSN: 2233-7857 IJFGCN

Shortest-Job-First (SJF) as well as Min-Min Best Fit (MMBF), have been advised for deciding the solutions i.e., it examine both of the aforementioned scheduling algorithms. SJF- Extreme Learning Machine (ELM) is another strategy which combines the ELM-based scheduling with SJF buffering algorithms, which has recommended additionally for eluding the chances of job scarcity in SJF-MMBF. And adding to that, there should be scheduling among the queues, which is generally applied as fixed-priority pre-emptive scheduling. It is evident that SJF- ELM is appropriate for the environments, like heavy-load and high dynamic, besides it is highly favourable in granting the average job hosting rate, through the outcome of simulation. But it failed to focus on host utility identification. Potential use of host demands more energy consumption and it may further result in VM failure, connection failure and response failure [6, 7].

An amplified cuckoo search optimization has been utilised for Multi-level queue scheduling, to prevent this issue in this research. Moreover, both SJF buffering and MMBF scheduling algorithms have examined here. To prevent the chances of facing the job scarcity in SJF-MMBF, another scheme merges SJF buffering and Extreme Learning Machine (ELM)-based scheduling algorithms. For detecting the over-utilized host, an algorithm named Adaptive Neuro Fuzzy Inference System (ANFIS)is presented, besides migrating VMs from over-utilized hosts to other hosts was handled through Minimum Migration Time (MMT) strategy, concerning the reduction of energy consumption [8] [9].

2. Literature Review

For cloud load balancing, various methods have comprehensively conferred in this segment. Rodrigues, et al [2016] tend to reduce the Service Delay in such scenario, for which they framed a model using two cloudlet servers, which aims on both the elements of communication and computation. In this model, the implementation of VM migration helps to restrain the Processing Delay, besides the Transmission Power Control takes place for up surging Transmission Delay. A mathematical model has considered foe deriving an analysis for being utilized to compare the proposed methodology with other traditional approaches. Through these traditional approaches, either one of the Processing Delay or Transmission Delay can be enhanced, since these models does not have the ability to focus on both, hence these have known to be mono focused. Certifying our outcome that a dual focus approach proves its efficiency to overcome the Service Delay issue over Edge Cloud Computing, the proposal states the lowest Service Delay in all study cases, as anticipated.

Yadav and Kushwaha [2014] discussed about a combined task scheduling algorithm which considers the issues like VM management and Datacenter management. On the CloudSim platform this algorithm has been applied and the acquired outcome of the evaluations denotes the effective performance, minimized average execution time, average waiting time of tasks, as well as turnout enhancement over recommended algorithm Cloud system. Moreover, it has known for its fault tolerance and can proficiency to handle energy consumption issue, and compatibility with the procedures of VM management.

Ramezani and Hussain [2014] have introduced a Task-based System Load Balancing technique exploiting Particle Swarm Optimization (TBSLB-PSO). Instead of moving all over-loaded VM, it has the ability to transmit solely an additional tasks to attain system load balancing. An optimization model has also been designed to transfer these extra labours to the new host VMs through implementing PSO. The cloud simulator (Cloudsim) package has been expanded, through which the PSO has been utilised as task scheduling model, to assess the advised technique. The outcomes of the simulation reveal the efficiency of the advised TBSLB-PSO approach to diminish the time consumption of the load balancing process, which is superior to the conventional methods of load balancing. Adding to that, the overloaded VMs will resume to work even in the migration process, also it has not necessitated for implementing the pre-copy process of VM in our advised technique. Apart from the prevention of VM downtime, the TBSLB-PSO

ISSN: 2233-7857 IJFGCN

method also prohibits the loss of customer's last activity, besides enhancing the experience of cloud customers as regards Quality of Service.

Marahatta, et al [2019] has introduced an Energy-Efficient Dynamic Scheduling Scheme (EDS for virtualized CDC, in which diversified processes and VM shave initially categorised on the basis of scheduling record from past. And then, for higher utilization of an operational state of the host identical variants of tasks are integrated and scheduled. Through utilizing the optimal operating frequencies and energy efficiencies of heterogeneous physical hosts, the energy has considerably preserved at the time of generating and eliminating the VMs. The outcomes of the evaluations proves that, EDS greatly enhances the whole performance of the scheduling process, utilizes the CDC resource in an efficient way, escalates the guarantee ratio of the process, reduces the mean response time, and minimises energy consumption when compared with traditional models.

Rodrigues et al [2018] has recommended an algorithm which uses VM Migration as well as Transmission Power Control, for stabilising workload within cloudlets in addition accordingly expand cost-effectiveness, alongside a delay mathematical model in Mobile Edge Computing and PSO. Proposal is to contemplate the computation, migration, and communication in our anticipated scale at the same time and, because of that, it surpasses other traditional techniques as regards the number of serviced users.

Wang et al [2014] has introduced an elastic resource contributing framework for fault-tolerance, in order to enhance the efficient use of the resource. A new fault-tolerant elastic scheduling algorithms has been structured in accordance with the elastic resource provisioning system as well as the fault-tolerant strategy, entitled FESTAL, which concurrently focuses over the fault tolerance as well as efficient utilization of resource. In addition, the FESTAL has compared with three standard algorithms (Non- migration-FESTAL (NMFESTAL), Non-Overlapping-FESTAL (NOFESTAL), and Elastic First Fit (EFF)), for which a comprehensive simulation have been carried out through CloudSim by including the random synthetic workloads, and the workload from trace logs of the Google cloud (recent version). It is evident through the outcome of various evaluations, that FESTAL has the ability to greatly improve the efficiency of virtualized clouds.

Aslanzadeh and Chaczko [2015] have promoted a new load balancing method, utilising Endocrine algorithm, which is stimulated from hormone system regulation behaviour of human. By implementing self-organizing technique between overloaded VMs, the recommended algorithm attains system load balancing. And it has been designed on the basis of communications between VMs. By implementing the improved feed backing method utilising PSO, it assists the overloaded VMs in transmitting their additional tasks to another under-loaded VM. To assess the algorithm recommended by our research, the CloudSim has been expanded. The outcome of the simulation shows that load balancing strategy recommended by our work greatly minimises the time span on comparing with conventional load balancing techniques. By reducing the VMs' downtime it maximises the Quality of Service (QOS).

3. Proposed Methodology

This research work concentrates on developing a load balanced task scheduling in cloud computing which greatly encompasses four building blocks. Enhanced cuckoo search optimization based Multi-level queue scheduling is the first most building blocks followed by SJF buffering and MMBF scheduling algorithms for validating the scheduling process. The combination of SJF buffering and ELM-based scheduling algorithms is exploited for preventing the job starvation probability in SJF-MMBF. The third block involves host detection by means of Bat optimization algorithm and the final block comprises MMT strategy for VMs migration from the over-utilized hosts to the further hosts, thereby diminishing energy consumption. Figure 1 describes the structural design of the suggested model.

ISSN: 2233-7857 IJFGCN

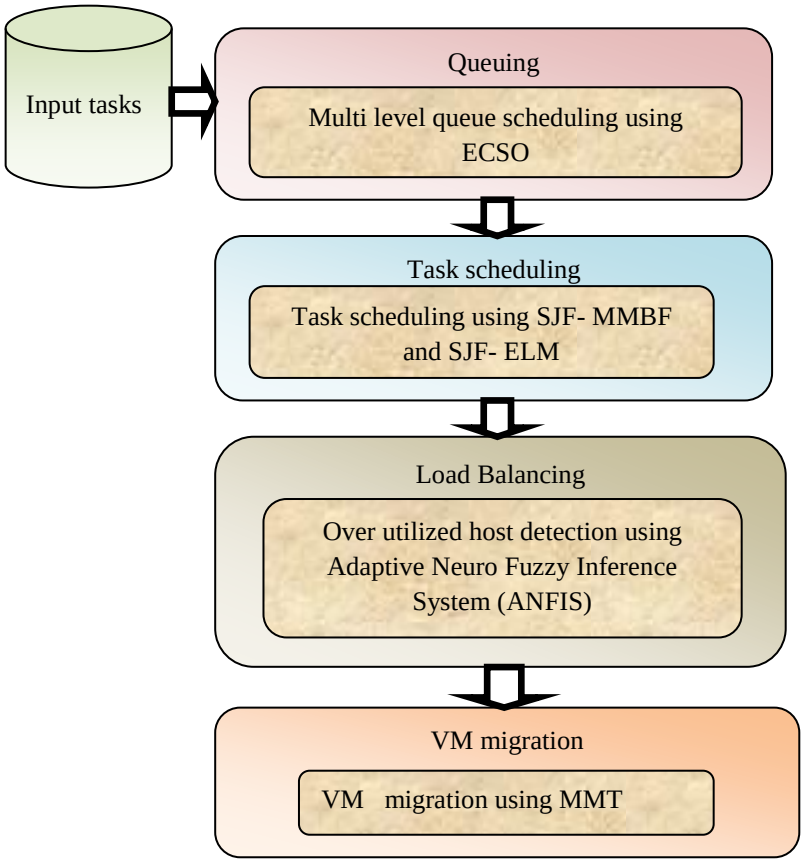

Figure: 1.Proposed work Overall architecture

3.1. Multi-level queue scheduling using Enhanced cuckoo search optimization

The enumeration of initial jobs is accomplished by the clients in terms of burst time. The bursting times like small, middle and high may be the main cause for work segregation by way of multi queues through the Enhanced cuckoo search optimization. The testing of SJF-ELM algorithms along with SJF- MMBF for resource usage is done through cloud in multi queues preference (scheduling). The distribution of network resources is a critical task skilled by the queue manager which utilizes entire network resources. The queue manager is solely responsible for the scheduling and managing the task pertaining to the burst time outcomes in an ascending order.

Cuckoo Search (CS) Algorithm

A stochastic population-based swarm intelligence algorithm namely Cuckoo Search (CS) is greatly utilized in this research. The cuckoo bird's parasitic behaviour plays a significant role in this algorithm and the notion is that the eggs of the female cuckoos are laid in the nest of other birds during which two scenarios may occur: The differentiation of the cuckoo's egg from its own egg by the host bird cannot be done and consequently hatching of cuckoo's egg happens for ensuing cuckoos generation or else host bird distinguish cuckoo's egg besides automatically agree for egg disposing or vacating nest for constructing further nest [17] [18].

The origin of the CS exploration method mainly depends on these two natural phenomena encompassing novel elucidations for the succeeding generation expending Levy flights and replacing an egg portion. For these natural phenomena simulation by way of a computer algorithm, the CS is stated through three idealised rules.

ISSN: 2233-7857 IJFGCN

Rule 1: A nest is arbitrarily selected by every cuckoo and its egg is laid in the nest.
Rule 2: The available nests are static and the nests with optimal eggs form the next generations.
Rule 3: If any cuckoo's eggs found by the host bird, it throws the eggs away or leaves nest for new nest construction.

Initialization

On the basis of problem dimension, every host nest (Job) is a probable clarification and one or more eggs may be contained for optimisation issues. Random walk Levy flights is greatly utilized in Equations (1) as well as (2) due to the nest random initialisation in the first step of the algorithm and in every iteration of the algorithm.

$$x_i^j(t) = x_i^j(t-1) + \alpha \oplus Levy(\lambda), \quad (1)$$

$$\text{Levy} \sim u = t^{(-\lambda)}, (1 < \lambda < 3) \quad (2)$$

At this time,the step size has denoted by *s*, the step size scaling weight has represented by $\alpha > 0$, and tuning must be done depending on problem scaling, product refers entry wise multiplications, besides $x^j{}_i$signifiesj^{th} eggs at i^{th} nest. The nests replacement with worst quality of eggs is done in the last step of each iteration, with the probability $= P\alpha \in [0, 1]$.

Improved CSO

A straight flight path with sudden 90 degree turns is done. by exploring the CS Levy flights search stratagem. Moreover random walk search procedure is greatly deployed by the CS and might effortlessly jump from one region to another one deprived of prudently discovering each cuckoo's nest. The challenges in CS are lesser accuracy in optimisation, inefficient local searching, delays in convergence, and these are overwhelmed using the strategy of mutation neighbourhood search by including to the conventional CS. The generation of neighbourhood solution has initiated through modifying the value of a single/many cells from its primary status, through which one candidate solution has received by mutation operator, as the input. For example, a binary vector's neighbour solution would probably be further binary vector that would possess the inverted one cells values in original solution [19][20].

The continuous domain is usually preferred by the CS for its operation and the multi queuing task is considered as a binary discrete issue. This research supports pseudo binary mutation neighbourhood search operator considering both areas. In every iteration of the CS, the operator is considered to be a pre-defined number of solutions and those have arbitrarily chosen as the operator input. Subsequently, arbitrarily chosen cell has established on its own value, the ensuing modification has applied:

- The cell value has inverted to $L_{b,}$ if it has greater value than σ
- The cell value has inverted to $U_{b,}$ if it has lesser value than σ

Further precisely, Boolean boundary in binary converter has denoted by operator σ, besides consequently:

The cells with value lesser than σ characterize 0 in binary mode, beside sit needs to be inverted to 1, if it has nominated by the designed mutation operator; then,

The cells with value greater than σ signify 1 besides consequently; operator inverts them to 0. The designed mutation operator pseudo code besides Extended Binary Cuckoo Search (EBCS) have introduced in (3) respectively.

Proposed Fitness Function φ

Equations (3) and (4) is used to assess the fitness value through solutions equivalent binary vector produced by means of conversion of the solution to binary values, following that the produced binary vector has directed to fitness function,

ISSN: 2233-7857 IJFGCN

$$y_i^j(t) = \begin{cases} 1 \; if x_i^j(t) \geq \sigma \\ 0 \; if x_i^j(t) < \sigma \end{cases} \qquad (3)$$

$$Y_i(t) = \cup_{i=1}^{N} y_i^j(t) \qquad (4)$$

In which, Boolean boundary has indicated by σ (0, 1); the value of jth eggs at i^{th} nest at time t has represented by$x^j_i(t)$, where t=0 indicates initial solutions; the binary value of j^{th} eggs at i^{th}nest *at* time t has signified by $y^j_i(t)$; and equivalent binary vector of i^{th} nest *at* time t has denoted by Y_i (t).

The new solutions obtained are validated and boundary violations are resolved for perpetually maintaining the difference between L_b and U_b in every iteration. The unique solutions persist in the constant field of the CS, besides it possesses the flexibility to get converted into binary, if required.

In order to obtain a subset of jobs from dataset's overall prevailing jobs, the learning algorithm influences to the minimum possible job completion time by means of those designated jobs and this is the main motto of multi-level queuing task.

Fitness function φ is greatly involved for the optimisation algorithms performance evaluation formulti-level queuing. The suggested objective function φ primarily intends to obtain the minimum Delay response time by considering the number of jobs pertaining to heuristic optimisation method.
The suggested objective function φ takes number of jobs to satisfy heuristic optimisation method selecting to reach minimum Delay response time as the regulation term.

Consider total number of jobs encompassed in a dataset as represented by d; s as the number of jobs designated by the met heuristic optimisation algorithm; α signifies weight assigned to number of jobs; in addition define that 1-α signifies the weight given to the average job completion time such as stated in Function (6). Fitness function φ computation *is* expressed in Function (7) and fixing set α = 0.2

$$\Phi = \left(\alpha * \frac{s}{d}\right) - ((1-\alpha) * z) \qquad (5)$$

In function (7), through dividing both terms with their maximum possible values, they have been normalised (the number of assigned jobs has divided by overall jobs d, then the job completion time z has divided by the value of 1.

Algorithm : Extended Binary CS Pseudo code
input: jobs D, *MaxIter*(or stop criteria),CS parameters value, number of nests, number of mutations N_{mut}, number of dimensions
(jobs) d, $L_b$and U_b
Output: *Best _Fitness*, *Best _Nest*
for *each nest* n_i(i=1,. . . ,m) do

$x^j_i(0)$ aarbitrary number drawn from [L_b, U_b]
end
Convert $n_i$to binary using
Evaluate job completion time of equivalent Binary vector of $n_i$in addition store it in f_i
end
[*MaxFit*, *Index*]=max(f)
Best Nest(job)=n_{index}
Best Fitness(job completion time) =MaxFit
while ($t \leq$ *MaxIter or stop criterion*) do
for *each nest(job)*n_i(i=1,. . . ,m) do
Execute Levy flights for new solution generation
Applying Eq. 1 and 2 besides store it in *n new*$_i$
end
for *each nest (job)nnew*$_i$(i=1,. . . ,m) do

ISSN: 2233-7857 IJFGCN

Rectification procedure
if $x^j_i > U_b$ *then*
$x^j_i = U_b$
else if $x^j_i < L_b$ then
$x^j_i = L_b$
else
$x^j_i = x^j_i$
end
end Convert $nnew_i$ to binary applying Eq. 3 in addition to 4
Evaluate job completion time of equivalent binary vector of $nnew_i$ besides store it in *acc*
if ($acc > f_i$) then
$f_i = acc$
$n_i = nnew_i$
end
end
Arbitrarily choose k (k=1,. . . ,N_{mut}) nests as well as store
them in Mut $\_Nest_k$ for *each nest Mut* $\_Nest_k$
(k=1,. . . ,N_{mut}) do
Mutate *Mut*_ $Nest_k$
Convert *Mut*_ $Nest_k$ to binary using Eq. 3 and 4
evaluate job completion time of equivalent binary vector of *Mut*_ $Nest_k$ in addition store it in *acc*
if (*there is a nest in the population with completion time lower thancple*) then
Switch nest with *Mut* $\_Nest_k$
end
End
end

[*MaxFit*, *Index*] = max (*f*)
Best $\_Nest = n_{index}$
*Best*_ *Fitness=MaxFit*
end

The jobs are categorized into multi queues on the basis of above procedure and scheduled in their specific queue depending on the SJF- MMBF and SJF- ELM scheduling algorithms

Based on the above procedure jos were divided in to multi queues and then it will be scheduled in their specific queue based on the SJF- MMBF and SJF- ELM scheduling algorithms.

3.2. Scheduling Policy for Min-Min Best Fit and Shortest-Job-First Buffering

The prediction of future resource requirements is a great challenge because of strong heterogeneity and dynamism encompasses during the process and also it is more costly for precise traffic features prediction (for instance arrival predicted level along with job medium size).

SJF- MMBF

1) Buffering Algorithm (SJF Buffering): The many type-v jobs that arrived in time intervals [t;t + 1) are buffered in v^{th} queue according to buffering strategy in addition described in Algorithm 1, for v V.
2) Scheduling Algorithm (MMBF Scheduling): In result epoch t, do
a) Determine resource array N A_tv.
b) choose action at*At such that Nal*NAtV is determined.

ISSN: 2233-7857 IJFGCN

3) Scheduling Process: In a time interval [t; t + 1), Nvp(t) type-v jobs continue to be functioned, besides Nat *,vNvpt type-v jobs are de-queued from the v^{th} queue in an HOL way as well as start to be aided for v V. The number of jobs to come in the queue in addition to acquisitive workload requirement are efficient, respectively.

SJF- ELM

1) Algorithm of SJF Buffering: A range of [t; t + 1] are buffering time of sort of v jobs consequently on queue as well as buffering policy. It has applied in Algorithm SJF as SJF-ELM, v ∈ V.

2) Scheduling Algorithm (ELM Scheduling): In $\boldsymbol{t_j}$ **of** decision epoch, execute

- $S_t \leftarrow \left(N_v^p(t)\right), Q_v(t), W_v(t)$ State is intellect.
- Huge X actions are computed practically and N A_s×v as array resource.

3) Scheduling Process:

a) Scheduling: jobs of $N_v^p(t)$type 1 are being supported in queue along vϵV, and de-queue $\left(N\left(a_t^*{}_{,}v\right) - N_v^p(t)\right)$ type-1 jobs from 1th queue and begin assisting function. Lot of waiting jobs have hold by queue and the required workload have updated in accumulation are from

$$\begin{cases} Q_v(t+1) = Q_V(T) - \left(N\left(a_t^*{}_{,}v\right) - N_v^p(t)\right) \\ w_v(t+1) = W_v(t) - N\left(a_t^*{}_{,}v\right) \end{cases} \quad (6)$$

b) Time Computation for job accomplishment in time-averaged as given by

$$\boldsymbol{E\left[\tilde{T}(t)\right] = \sum_{v=1}^{V} \alpha E\left(\hat{T}_v(t-1)\right) + (1-\alpha) T_v(t)} \quad \textbf{(7)}$$

Where, a weight parameter has represented by$\boldsymbol{\alpha} \in (\mathbf{0}, \mathbf{1})$

c)Vector of parameter have been revised, if $\boldsymbol{E\left[\widetilde{T(t)}\right] T_j > E[T^*]}$, $\omega *$

d) To store the final number of arrived traffic T, the following function can be applied

$$\{j_v(t-T+1), \ldots . j_v(t)\} \quad (8).$$

3.3. Over utilized host detection using ANFIS

An over-utilized host does not support services for the entire request which are allocated to it during which there is an up surge of response times of the requests. Hence over-utilized hosts are also involved with cloud service provider and ANFIS is greatly utilized for system CPU utilization prediction.

Adaptive Neuro Fuzzy Inference System (ANFIS)

ANFIS input is nothing but all hosts CPU utilization information and ANFIS network is regarded as a sort of neural network that functions on the basis of neuro fuzzy network, besides it comprises two rules as depicted in Fig 2.

The first layer comprises of entire adaptive nodes. The layer 1outputs are theinputs fuzzy membership grade specified by:

$$O_i^1 = \mu_{A_i}(x) \; for \; i = 1,2 \quad (9)$$

ISSN: 2233-7857 IJFGCN

Where x and y are signified as the input nodes, A and B are represented as the linguistic labels, (x) and (y) are signified as the membership functions which frequently assume a bell shape by means of the highest and lowest values equivalent to 1 and 0, correspondingly.

$$\mu(x)= \frac{1}{1+(\frac{x-c_i}{a_i})^{2b_i}} (10)$$

Where , a_i, b_i, and $c_i$represents the premise parameters set.

The second layer encompasses of fixed nodes. They have tagged with M, signifying that they execute as a basic multiplier. The layer representation is given by,

$$O_i^2 = w_i = \mu_{A_i}(x)\mu_{B_i}(y),\ i = 1,2 \quad (11)$$

The rule firing strength has signified through output w_i. The node outcome signifies rule firing strength.

The third layer includes the static nodes with label N, demonstrating their role of normalization in order to firing strengths from the preceding layer[21] [22]. This layer can be represented as:

$$O_i^3 = w_i = \frac{w_1}{w_1+w_2},\ i = 1,2 (12)$$

This layer's outcomes have termed as normalized firing strengths.

The fourth layer encompasses adaptive nodes. In this layer, each nodes output is merely normalized firing strength output besides a first order polynomial (for a first order Sugeno model). Consequently, this layer output is specified as:

$$O_i^4 = w_i f_i = w_i(p_i x + q_i y + r_i) (13)$$

Where $w$represents the layer 3 output, besides$\{p_i, q_i, r_i\}$is designated as the parameter set. These parameters are referred as the resultant parameters

In the fifth layer, a single fixed node has categorized by S. This node summates overall incoming signals[23,24]. Hence, model overall output is specified by:

$$O_i^5 = \sum_{i=1} w_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} (14)$$

The over-utilized host will be determined during this stage

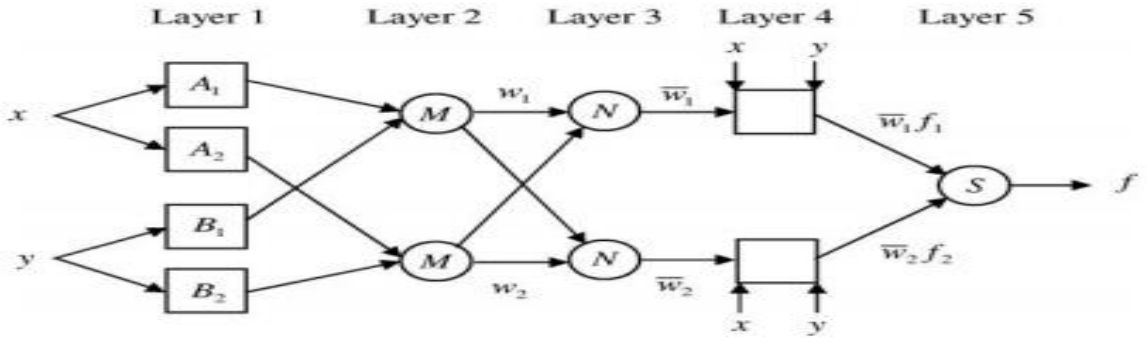

Figure.2. ANFIS architecture.

ISSN: 2233-7857 IJFGCN

Host Under loading Detection

The underutilized host refers to the very low load host along with an allotment of fewer requests. Due to the energy consumption, many requests cannot be responded and required a basic policy to handle the underutilized hosts except over-utilized host. Following which migration of VMs is accomplished for underutilized hosts. In the course of which any host cannot be over-utilized. All the under-utilized hosts switch to several hosts as probable to the sleep mode which in turn mitigates the active host quantity and utilization of energy.

3.4. Virtual machine migration using Minimum Migration Time (MMT)

The over utilized host is identified either by selecting a single or many VM migration occur from these hosts to the others accompanying a constraint that it should carry forward to the over utilized mode. From over utilized hosts, the VMs have selected by the strategy called Minimum Migration Time (MMT) [25] [26]. The minimum time is necessitated for attaining the migration process by this policy.The migration time is assessed as RAM amount utilized through VM divided by the spare network bandwidth existing for the host j. If VM v has subsequent conditions, MMT policy is specified as

$$v \in \frac{V_j}{\forall V_j} \quad \textbf{(15)}$$

$$\frac{RAM_U(v)}{NET_i} \leq \frac{RAM_U(A)}{NET_i} \textbf{(16)}$$

Here, a VMs set that presently allocated to host j has denoted by Vj; The spare network bandwidth existing for the host j has indicated by NETj; and RAM that presently exploited through VM has represented by RAM $_u$ (a).

ALGORITHM

Input: hostList
For each host in the host List do
If host is over loaded then //Part A
Obtain VMs from this host migration // Part B
For each host in host List do
If host is under loaded then // Part C
If it is possible to migrate all VMs which
assigned to his host to the other hosts then
Migrate entire VMs
else
Retain host active

4. Result and Discussion

Throughout this segment, the empirical findings of the proposed model have conferred. The implementation of the overall system has carried out in JAVA platform. By considering the metrics, such as Throughput, Delay, and Cost, the recommended methodology of MQ-SJFELM-MMT has contrasted with the present approaches, such as single queueing SJF-RL (SQ-SJF- RL), single queuing SJF-MMBF (SQ- SJF-MMBF), PLBA, and Multi Queuing SJF-ELM (MQ-SJF-ELM).

ISSN: 2233-7857 IJFGCN

Table: 1.Performance comparison results

METRICS	METHODS				
	SQ-SJF- RL	SQ- SJF-MMBF	PLBA	MQ-SJF-ELM	MQ-SJFELM-MMT
Throughput	0.320	0.570	-	0.698	0.755
Delay(ms)	1102	680	-	530	350
cost($)	145	117	-	89	57
Energy consumption (kwh)	12	9	8	6	4

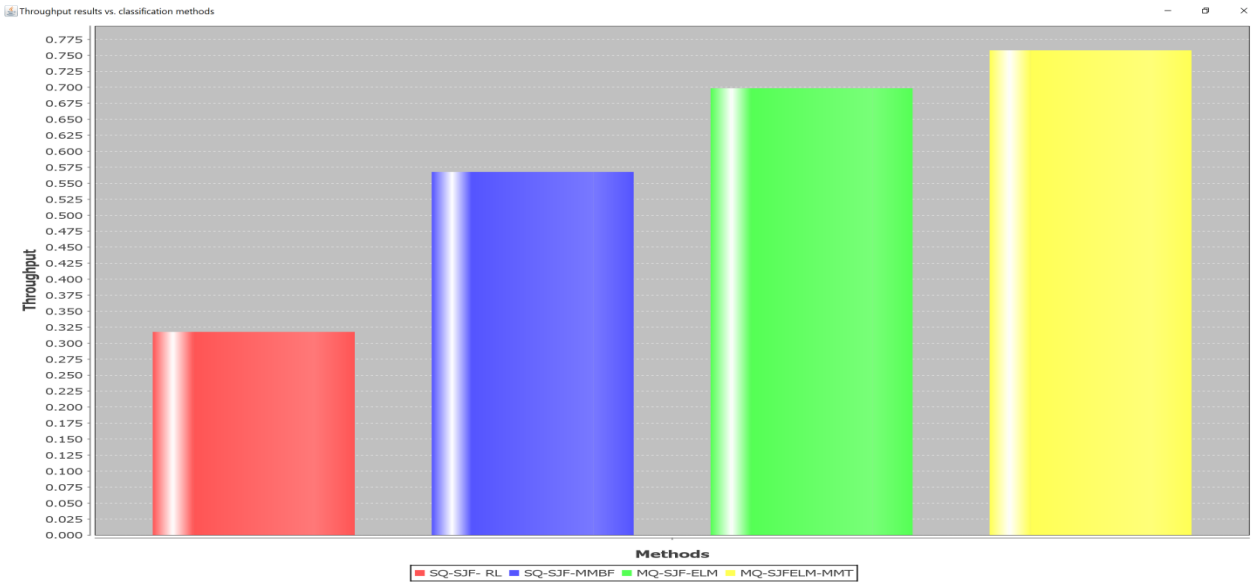

Figure: 3. Scheduling methods Vs results of Throughput

Comparison of Throughput results obtained by the proposed MQ-SJFELM-MMT approach is depicted in figure 3 along with current SQ-SJF-RL,SQ- SJF-MMBF, MQ-SJF-ELM approaches, where X-axis signifies the Scheduling methods, and Y-axis stands for Throughput values. The results demonstrate that the proposed MQ-SJFELM-MMT framework has proficiently secured the Throughput rate of 0.755, which is superior to the results of current methods, such as SJF- RL,SQ-SJF-MMBF, and MQ-SJF-ELM, as they solely attains 0.320, 0.570, and 0.698, respectively.

ISSN: 2233-7857 IJFGCN

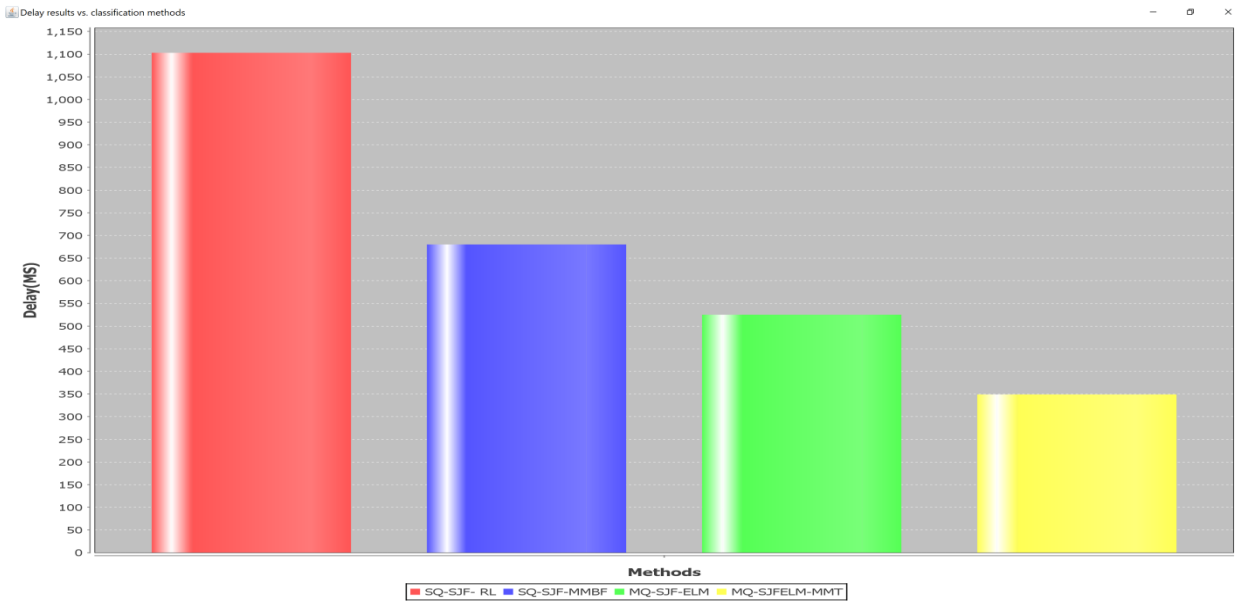

Figure: 4. Delay results vs. scheduling methods

Figure: 4 compares the Job scheduling values as regards the Delay ratio of the proposed MQ-SJFELM-MMT approach, and the current SQ-SJF-RL,SQ-SJF-MMBF, MQ-SJF-ELM approaches, where X-axis represents the Scheduling methods, besides Y-axis stands for Delay values which have taken as the ELM weight matrix that reduces the delay in response time, as it efficiently augments scheduling accuracy. The outcomes reveal that recommended MQ-SJFELM-MMT framework holds the least Delay rate of 350 (ms), which indicates the superior performance of this approach, whereas the current methods, such as SJF- RL,SQ-SJF-MMBF, and MQ-SJF-ELM have solely attains 1102 (ms),680 (ms), and 530 (ms), respectively.

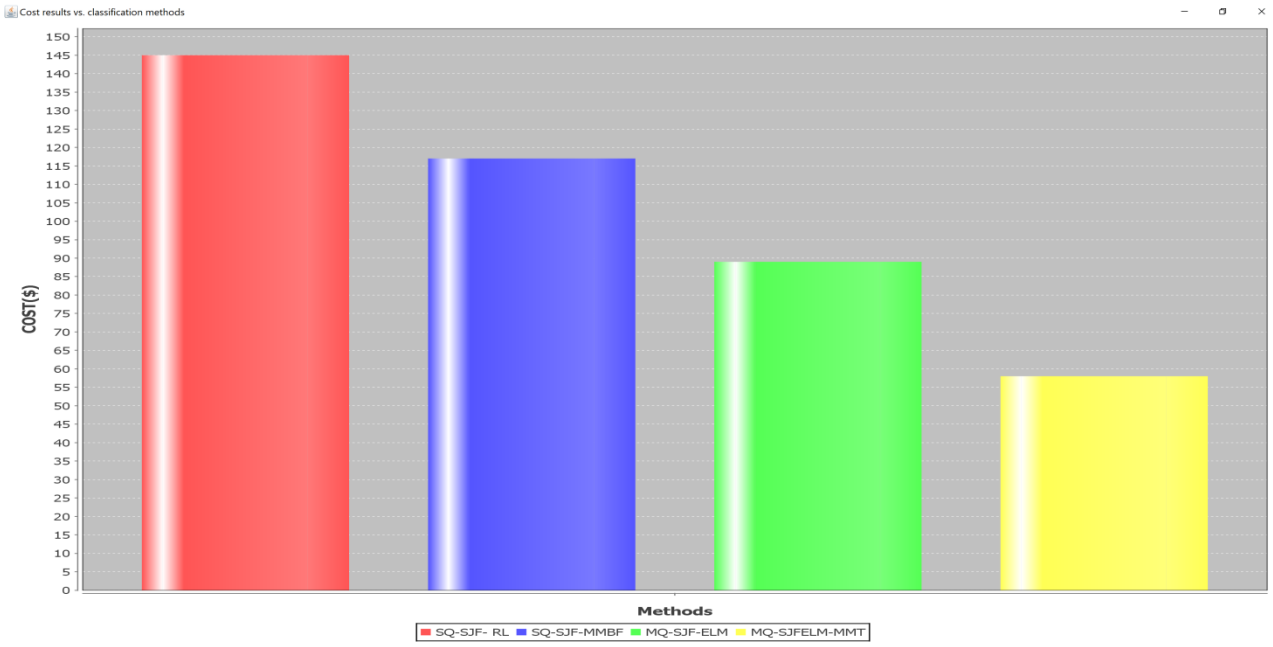

Figure: 5. Cost results vs. Scheduling methods

ISSN: 2233-7857 IJFGCN

In Figure: 5, the chart compares the Job scheduling outcomes on the basis of Cost values of the proposed SJFELM-MMT technique, and the present methodologies of SQ-SJF- RL,SQ- SJF-MMBF, MQ-SJF-ELM, in which the X axis designates Scheduling methods, besides Y axis stands for Cost. The results depict that the proposed MQ-SJFELM-MMT strategy solely requires the least expense with the Cost value of 57$, which is much cheaper than the current methods, such as SJF- RL,SQ-SJF-MMBF, and MQ-SJF-ELM as they demands 145$, 117$, and 89$, respectively.

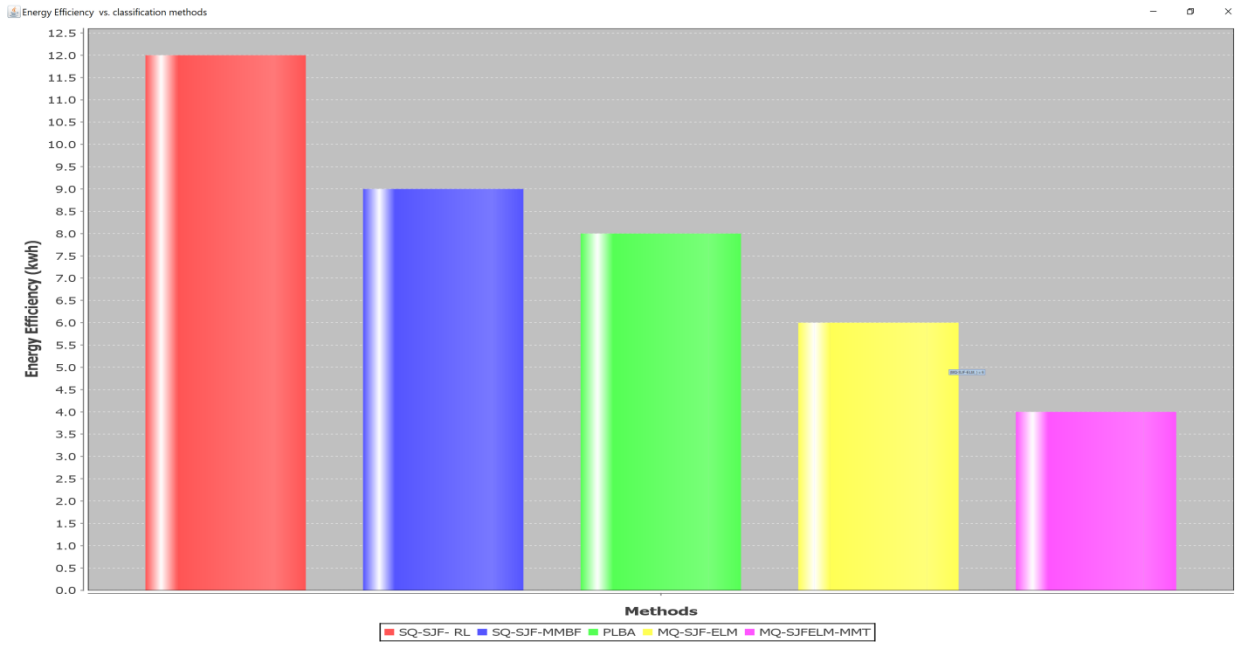

Figure: 6. Energy efficiency results vs. Scheduling methods

The chart in Figure: 6 compares the results of Job scheduling based on the parameter of Energy efficiency for the proposed SJFELM-MMT strategy, and the existing approaches of SQ-SJF-RL,SQ- SJF-MMBF, MQ-SJF-ELM, during which the over utilized hosts have detected for VM migration and energy efficiency optimization of the Cloud using Gaussian radial-based kernel of ANFIS . The results prove that the proposed SJFELM-MMT framework has the efficiency to offer optimum Energy efficiency with the value of 4(kwh), conversely the existing approaches, such as SQ-SJF- RL,SQ- SJF-MMBF, PLBA, and MQ-SJF-ELM deliver 12(kwh), 9(kwh), 8(kwh), and 6(kwh), respectively.

5. Conclusion and Future Work

The growing number of users of the cloud service over the previous years, significantly escalates the challenge in offering the effective customer service. During this study, the Multi-level queue scheduling has carried out using the Enhanced cuckoo search optimization method, which inspects both the scheduling algorithms, namely Min-Min Best Fit (MMBF), and Shortest-Job-First (SJF) buffering. In SJF-MMBF, the evasion of job starvation potentiality has attained by amalgamating the SJF buffering with Extreme Learning Machine (ELM)-based scheduling algorithms through another strategy. The further involvement of Adaptive Neuro Fuzzy Inference System (ANFIS) algorithm based Load Balancing technique helps to identify the over utilized host, besides the Minimum Migration Time (MMT) strategy eventually takes place to diminish the energy consumption by migrating the Virtual Machines (VMs) to the further hosts from the over-utilized hosts. By considering the features of Throughput, Delay, Cost and Energy consumption, the empirical findings ascertain the performance competency of the suggested methodology.

ISSN: 2233-7857 IJFGCN

References:

1. Awad, A.I., El-Hefnawy, N.A. and Abdel_kader, H.M., 2015. Enhanced particle swarm optimization for task scheduling in cloud computing environments. Procedia *Computer Science*, *65*, pp.920-929.
2. Ma, L., Lu, Y., Zhang, F. and Sun, S., 2012, May. Dynamic task scheduling in cloud computing based on greedy strategy. In *International Conference on* Trustworthy Computing and Services (pp. 156-162). Springer, Berlin, Heidelberg.
3. Beegom, A.A. and Rajasree, M.S., 2014, October. A particle swarm optimization based pareto optimal task scheduling in cloud computing. In International Conference in Swarm Intelligence (pp. 79-86). Springer, Cham.
4. Razaque, A., Vennapusa, N.R., Soni, N. and Janapati, G.S., 2016, April. Task scheduling in cloud computing. In 2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT) (pp. 1-5). IEEE.
5. Wang, T., Liu, Z., Chen, Y., Xu, Y. and Dai, X., 2014, August. Load balancing task scheduling based on genetic algorithm in cloud computing. In 2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing (pp. 146-152). IEEE.
6. Tawfeek, M.A., El-Sisi, A., Keshk, A.E. and Torkey, F.A., 2013, November. Cloud task scheduling based on ant colony optimization. In 2013 8th international conference on computer engineering & systems (ICCES) (pp. 64-69). IEEE.
7. Xavier, V.A. and Annadurai, S., 2019. Chaotic social spider algorithm for load balance aware task scheduling in cloud computing. *Cluster Computing*, *22*(1), pp.287-297.
8. Li, J., Qiu, M., Niu, J., Gao, W., Zong, Z. and Qin, X., 2010, August. Feedback dynamic algorithms for preemptable job scheduling in cloud systems. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (Vol. 1, pp. 561-564). IEEE.
9. Agarwal, M. and Srivastava, G.M.S., 2016, April. A genetic algorithm inspired task scheduling in cloud computing. In *2016 International Conference on Computing, Communication and Automation (ICCCA)* (pp. 364-367). IEEE.
10. Rodrigues, T.G., Suto, K., Nishiyama, H. and Kato, N., 2016. Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control. IEEE Transactions on Computers, 66(5), pp.810-819.
11. Yadav, R.K. and Kushwaha, V., 2014, August. An energy preserving and fault tolerant task scheduler in cloud computing. In 2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014) (pp. 1-5). IEEE.
12. Ramezani, F., Lu, J. and Hussain, F.K., 2014. Task-based system load balancing in cloud computing using particle swarm optimization. International journal of parallel programming, 42(5), pp.739-754.
13. Marahatta, A., Pirbhulal, S., Zhang, F., Parizi, R.M., Choo, K.K.R. and Liu, Z., 2019. Classification-based and energy-efficient dynamic task scheduling scheme for virtualized cloud data center. *IEEE Transactions on Cloud Computing*.
14. Rodrigues, T.G., Suto, K., Nishiyama, H., Kato, N. and Temma, K., 2018. Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration. IEEE Transactions on Computers, 67(9), pp.1287-1300.
15. Wang, J., Bao, W., Zhu, X., Yang, L.T. and Xiang, Y., 2014. FESTAL: fault-tolerant elastic scheduling algorithm for real-time tasks in virtualized clouds. *IEEE transactions on computers*, *64*(9), pp.2545-2558.
16. Aslanzadeh, S. and Chaczko, Z., 2015, May. Load balancing optimization in cloud computing: Applying Endocrine-particale swarm optimization. In *2015 IEEE International Conference On Electro/Information Technology (Eit)* (pp. 165-169). IEEE.

ISSN: 2233-7857 IJFGCN

17. Arcanjo, D.N., Pereira, J.L.R., Oliveira, E.J., Peres, W., de Oliveira, L.W. and da Silva Junior, I.C., 2012, November. Cuckoo search optimization technique applied to capacitor placement on distribution system problem. In *2012 10th IEEE/IAS International Conference on Industry Applications* (pp. 1-6). IEEE.
18. Patwardhan, A.P., Patidar, R. and George, N.V., 2014. On a cuckoo search optimization approach towards feedback system identification. *Digital Signal Processing*, *32*, pp.156-163.
19. Valian, E., Mohanna, S. and Tavakoli, S., 2011. Improved cuckoo search algorithm for feedforward neural network training. *International Journal of Artificial Intelligence & Applications*, *2*(3), pp.36-43.
20. Civicioglu, P. and Besdok, E., 2013. A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artificial intelligence review*, *39*(4), pp.315-346.
21. Mishra, R.N. and Mohanty, K.B., 2015, December. Performance enhancement of a linearized induction motor drive using ANFIS based torque controller. In *2015 Annual IEEE India Conference (INDICON)* (pp. 1-6). IEEE.
22. Khan, S.A., Equbal, M.D. and Islam, T., 2014, December. ANFIS based identification and location of paper insulation faults of an oil immersed transformer. In *2014 6th IEEE Power India International Conference (PIICON)* (pp. 1-6). IEEE.
23. Lee, K.C. and Gardner, P., 2006. Adaptive neuro-fuzzy inference system (ANFIS) digital predistorter for RF power amplifier linearization. *IEEE Transactions on Vehicular Technology*, *55*(1), pp.43-51.
24. Kusagur, A., Kodad, S.F. and Ram, B.S., 2010. Modeling, design & simulation of an adaptive neuro-fuzzy inference system (ANFIS) for speed control of induction motor. *International Journal of Computer Applications*, *6*(12), pp.29-44.
25. Mahmud, N., Zahedi, A. and Mahmud, A., 2017. A cooperative operation of novel PV inverter control scheme and storage energy management system based on ANFIS for voltage regulation of grid-tied PV system. *IEEE Transactions on Industrial Informatics*, *13*(5), pp.2657-2668.
26. Qamai, S., Koppanati, R.K. and Kumar, K., 2018, June. VM-MMT: a novel approach for VM consolidation over openstack cloud using linear regression and Minimum Migration Time. In *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1814-1819). IEEE.

ISSN: 2233-7857 IJFGCN