A RYU-SDN Controller-Based VM Migration Scheme Using SD-EAW Ranking Methods for Identifying Active Jobs in the 5G Cloud Framework

Grace Shalini T., Velammal College of Engineering and Technology, India
Rathnamala S., Sethu Institute of Technology, India

ABSTRACT

The presented scheme focuses on active jobs live migration among VMs in 5G cloud framework depending on the software defined networks (SDN) to improve QoS in cloud framework. In this approach, RYU SDN controller is employed, which provides software components that allows software developers to extend network management and control applications for utilizing the features of SDN controller. It currently supports variety of southbound protocols such as OpenFlow, OF-Config, NETCONF, etc., whereas the proposed system uses Mininet prototype network. The destination server selection in the data centre is based on the server distinction based equivalent active weights (SD-EAW) ranking methods. The weight computation necessitate was to recognize non-active and active jobs. A presented SD-EAW scheme utilizes Pareto distribution for the recognition of active and inactive jobs in both continuous and discrete intervals of time. The presented SD-EAW algorithm functions well over all traditional approaches and in turn offers an optimum solution through minimizing the cloud environment's make span.

KEYWORDS

5G Cloud Environment, Mininet Prototype Network, SDN-Enabled Cloud Data Centres, Server Distinction Based on the Equivalent Active Weights (SD-EAW) Ranking, Virtual Machine (VM) Migration

1. INTRODUCTION

In today's world, the network solution is being developed for advanced networks at the steady pace. The mobile device proliferation, strategies of server virtualization, and the cloud framework were the finest opinions in a conventional network structure (Varghese, 2018), because of the advanced technologies advancement (Stergiou, 2018). Nowadays, the system links a databases variety of servers over several domains of network. Consequently, multiple server and client scenarios are needed. Accordingly, the traffic in the trends could vary. The enterprise company supplies private and public providers of cloud which should be supple in retrieving storage, software, and further IT

DOI: 10.4018/IJCAC.319031

tools on-demand (Senyo, 2018). This in turn could be resolved by offering infrastructures of network over software defined networks (SDN). This could become prevalent nowadays for the reason of SDN benefits, like reliability, innovation, scalability, and testing (Amoore, 2018). The information plane comprises the network components managed like OpenFlow or Off-switches of a control plane. This in turn permits the shared knowledge in the networks application layer which offers features of network like load balancing, routing, and detection of intrusion (Langmead, 2018).

Presently, cloud computing is developing over several benefactors and the outdated set-ups of cloud that are being replaced through the technology revolution. By the way, this approach focusses on issues of load balancing, one of the major issues in cloud computing. A deprived algorithm of load balancing might outcome in utmost make span. The make span time was distinct as general time of the cloud environment completion. The solutions with respect to the issues of load balancing were given by means of traditional systems. However, only a fewer traditional system. The network objective of 5G cloud was capable of attaining optimal cellular networks performance (Ge, 2014). Several prevailing approaches focused on the virtual machines migration that comes at load balancing problems of the cloud environment. However, it was too essential for considering the migration of jobs over virtual machines in cloud environment.

There were two kinds of jobs in the environment of cloud. One is the active one and the other is the inactive one. The job was supposed to be an active one once their lifetime could not be projected at the environment. The time of those active jobs' completion might fall at some exponential time and thus this becomes a NP-hard issue. The presented research work offers an optimal active jobs migration over the VM in cloud for solving load balancing problems. For performing these active jobs live migration over virtual machines, SD-EAW ranking algorithm was presented. The algorithm in turn hypotheses an instruction queue for storing environment's active jobs and employs Pareto distribution for the intention of jobs lifetime on behalf of recognizing the dynamic jobs weight. The algorithm process and job weights calculation are offered by the proposed scheme. The presented SD-EAW process was compared with traditional outcomes in the optimum outcome on minimizing make span of cloud framework.

The remaining portion of the proposed system is structured as shown: section 2 is the detailed description of related works employed so far. Section 3 is the detailed explanation of the proposed mechanism. The performance analysis of the proposed system is depicted in section 4. Finally, the conclusion part is deliberated in section 5.

2. RELATED WORKS

(Abdelmoniem, 2016) presented the VM and SDN-capable switches along with SDN-driven in cast frame depending on SDN controller and hyper speed/switch package. This in turn estimates the smaller testbed performance, larger and NS2 environment over the usual implementation of presented approach. The Load balancing was regarded as the crucial portion of cloud framework all through the period. There were 'n' literatures number that were available for the issues related to load balancing. The cloud environment concentrates on some more main intentions like the utilization of proper resource, algorithm of optimal scheduling for minimizing the environments make span, efficient utilization of resource clustering policy and the optimum policy of job migration so as to ensure the effective load balancing framework. In one such traditional system, it was said that job migration policy was concentrated on producing the effective cloud environment. From this regard, several literatures were analyzed for the better knowledge of job migration policies and their parameters. Various migration categories include virtual machine migration, resource migration and several other kinds of migrations that were occurred by several other domains.

The issues employed over overloaded and underloaded situation were addressed (Mishra, 2020) over the load balancing algorithms classification. The algorithm of load balancing was categorized as dynamic and static strategy. The dynamic approach was classified further as online mode and offline mode. This review classifies the replication outcomes of several algorithms of load balancing depending on parameters like energy consumption, resource utilization and make span. In (Tchernykh, 2015), the issues were discussed in job migration time of completion whereas provisioning the cloud environment resources. A review was carried out on the vital uncertainty causes that lies in the provisioning of resources in the environment of cloud. The algorithm of weighted round robin process (Devi, 2016) was presented for mitigating the un-active tasks for sharing the cloud resource on cloud framework in an effective manner. For doing so, the capabilities like task length depending on the concentrated and categorized the factors of load balancing issues like overloaded, underloaded, and the balanced one. In this approach a new scheme (Deshpande, 2017) was analyzed over a designed prototype for handling traffic. A prototype of this kind was employed intended for VMs only with the virtual machines that have source machines pre-copy and the post-copy at the destination machine. The system's load balancing was carried out by two methodologies (Forsman, 2015) termed pull and push method. Those two methods in turn offer live migration over the systems.

The explicit study on virtual machine migration and the dynamic voltage frequency scaling datacenters of cloud were presented (Shirvani, 2020) so as to enhance the cloud environment performance. The minimized range of power consumption was being utilized for differing virtual machines depending on the incoming jobs. Several issues, challenges, moderations of performance, parameter intended for VM migration were being mentioned. The active live migration depending on the migration of memory was employed for offering the optimal time of migration (Noshy, 2018), transmission of data time and the virtualized cloud environment downtime. A policy of novel resource clustering (Mohan, 2019) termed differentiation of resource depending on the potential of node equivalence algorithm (RDENP) for solving the issues of load balancing in the simulated environment of wireless personal cloud.

An Effectual evaluation model of performance depending on the Poisson process and continuous Markov chain was offered for offering optimal solutions intended for the clusters of resource at the environment of cloud (Kowsigan, 2019). So as to offer enhanced cloud environment reliability the hierarchy-based resource clustering process was employed for minimizing the complexity of time in the selection of resource (Zhao, 2018). So as to enhance the cloud environment scheduling, the issues of resource provisioning were addressed by means of graph model (Kliazovich, 2016). QoS based scheduling and cost algorithm (Wang, 2017) was presented for handling the disputes in the environment of wireless cloud (Madni, 2016). So as to process the data streaming in cloud effectively, Markov chain prediction method was suggested (Zhang, 2015). Markov chain's Hurst exponent was projected for attaining the effective utilization of resource in cloud framework on considering the virtual machines behavior (Lu, 2015).

3. PROPOSED WORK

This section offers the detailed explanation of proposed system. RYU SDN controller is employed in this work that offers software components and allows the software developers for extending the control applications and network management for the utilization of SDN controller features. This in turn supports the diverse range of south band protocols currently like OpenFlow, OF-Config, NETCONF and so on, whereas the presented system employs Mininet network prototype. The selection of destination server in the data center depends on server distinction based Equivalent active weights (SD-EQA) ranking algorithm. The presented SD-EAW algorithm in turn focusses on the weight calculation for each and every job in the cloud framework. A weight calculation necessity was to recognize active and non-active jobs. The proposed SD-EAW scheme employs Pareto distribution for the recognition of continuous and discrete time intervals. The presented algorithm of migration functions well than other traditional techniques and in turn offers the optimal solution on diminishing the make span of the cloud framework.

2.1 Formulation of Parameter

By means of job's speaking lifetime, mentions to the job extent generally. The job's lifetime is intended as overall job utilization presently in addition to in imminent iterations. The future lifetime calculation of job is a challenging procedure, as a lifetime falls under the exponential distribution. At those situations, the active jobs movement turns out to be a non-deterministic polynomial concern. The SDN dependent method for cloud is effectual on keeping QoS. The jobs lifetime calculation outcomes in an increasing order for example, in the future jobs of the environment having the similar lifetime entirely. This might limit jobs to migrate or wander from one processor to the other one in the server. Thus, the agent migrating in turn comes under the impasse condition for selection of job that were to be migrated. Because of this, the entire traditional techniques concentrated only on the non-active job migration. For the reason of exponential distribution, lifetime might gradually enhance for the part of active job that are unexecuted. By the way, the exponential distribution turns out to be a pareto distribution in the presented approach. The pareto distribution in turn creates an exponential distribution as restrained distribution, because of this active job's lifetime are calculated easily. The job's lifetime comes in two types; one is job lifetime which in turn surpasses the limit and the job's lifetime is another one that comes under the limit. In this mentioned limit, roughly for instance 'Y'. at this time, exponential distribution initially is then expressed as shown below:

$$\bar{G}\left(Y\right) = e^{-\lambda Y} \tag{1}$$

$$\bar{G}\left(Y\right) = P\{Lifetime\ \ of\ \ the\ \ job > Y\} \tag{2}$$

This term is specified in the measured distribution as shown above. From the above-mentioned equation, it is mentioned obviously that the lifetime of job was more than limit. From this, it is summarized that there was a solution intended for computation of active job's lifetime, lifetime of job should be either above/below the limit as in the equation (1) exponential distribution it was not so easy to compute the lifetime of job. $\bar{G}$ signifies the function distribution, P signifies the function of probability, λ signifies to rate of the jobs arrival and *e* denotes to exponential value. The equation above might be revised consistent with both cases: below and greater than limit as:

$$P\left\{Lifetime\ \ of\ \ the\ \ job > Y \mid Lifetime\ \ of\ \ the\ \ job > ET\right\} = \frac{ET}{Y} \tag{3}$$

Here, ET signifies the time estimated. The expected time falls under job's lifetime state at some limit. Beyond the limit, the job's lifetime condition might be considered as the extension generally, however the solution will be recognized in the time of polynomial over the value of the measured Pareto distribution. This pareto distribution in turn offers the flexibility on attaining the solution that ranges from 0 to 2. From this approach, the flexibility varied from 0.7 to 1.1. in this, the lifetime of job might be broken to several portions for attaining the solution over pareto distribution flexibility. The distribution will be equated again consistent with a flexibility by way of specified equation as shown. In this, the flexibility is being revealed as β:

$$\bar{G}\left(Y\right) = \frac{ET}{Y\beta} \tag{4}$$

The Pareto distribution characteristics is essential highly and is employed through several business products for identifying the actual loss and profit in the entire turns of the specified artefact. The pareto distribution was employed once there is the exponential distribution rate chance in the real-world issues.

The pareto distribution is then associated by their flexibility as follows:

$$\bar{G}_y\left(y\right) = P\left\{Y > y\right\} = y^{-\beta}, for\ \ y \geq ET \tag{5}$$

where:

$$0 < \beta < 2\ \ is\ \ called\ \ Pareto\left(\beta\right) flexible\ \ distribution$$

The other significant entity intended for employing Pareto distribution was that their rate of failure was least and also the rate of failure is then reduced on behalf of the imminent iterations. It is revealed by the computation provided below as follows:

$$\bar{G}\left(y\right) = P\left\{Y > y\right\} = y^{-\beta}, y \geq ET \tag{6}$$

where:

$$G\left(y\right) = P\left\{Y < y\right\} = ET - x^{-\beta}, y \geq ET$$

$$g\left(y\right) = \frac{dG\left(y\right)}{dy} = \beta y^{-\beta - ET}, y \geq ET$$

$$r\left(y\right) = \frac{g\left(y\right)}{G\left(y\right)} = \frac{\beta y^{-\beta-ET}}{y^{-\beta}} = \frac{\beta}{y}, y \geq ET$$

As of the above calculations this has been noted clearly that:

$$\int_1^{\omega} g\left(y\right) dy = \int_1^{\omega} \beta y^{-\beta-ET} dy = ET \tag{7}$$

By the way, the claims of $g\left(y\right)$ Pareto distribution is an assessable PD (probability distribution) on relating exponential distribution. The failure rate of the pareto distribution in turn declines gradually for example $r\left(y\right) = \frac{\beta}{y}$ declines with *y*. Even Pareto distribution's variance and mean can too be considered in cases similar to *β£ET*. At this time the function of density is united once $0 < \beta \leq ET$ and equated as shown:

$$Job\left[remaining\ \ Lifetime \middle| size = \beta\right\rangle ET\right] = \infty \tag{8}$$

Over Pareto distribution, it is probable for computing the lifetime of several jobs in case of *β = ET*. For instance, this is probable to recognize the lifetime of Job1 probability to of lifetime of Job2 probability in case of Job2 > Job1, wherever *β = ET*. The result is an intellectual one, at which the job migration is shown in the server:

$$P\left\{Lifetime > Job2 \middle| Life = Job1 \right\rangle ET\right\} = \frac{ET \;/\; Job2}{ET \;/\; Job1} = \frac{Job1}{Job2} \tag{9}$$

By this expression, in case of distribution of Pareto *(β = ET)*, distribution might be interpreted as follows:

- A probability of particular job having ET lifetime and in turn exploits ≥ ET around T, the make span is $\frac{ET}{T}$.
- In 50% of the jobs, for Jobs that has ET lifetime, might have lifetime ≥ ET.
- The particular job probability having T lifetime and in turn exploits ≥ T about Tn, lifetime is $\frac{ET}{Tn}$.

The Pareto distribution's measured distribution offers both lifetimes' data measured below the limit and over limit. Through this, the Pareto distribution property, values will be computed altogether and the results were achieved. For formulating jobs lifetime in a structure, Bounded Pareto distribution might be employed. The Bounded pareto distribution $\left(f, q, \beta\right)$ is specified as:

$$g\left(y\right) = \beta y^{-\beta - ET} \cdot \frac{f^{\beta}}{1 - \left(\frac{f}{q}\right)^{\beta}} \tag{10}$$

Here, $f \le y \le q \;\; and \;\; 0 < \beta < 2$.
The expression:

$$\frac{f^{\beta}}{1 - \left(\frac{f}{q}\right)^{\beta}}$$

that integrates the function of density among *f* and *q* might move in *ET* range since this has been normalized already.

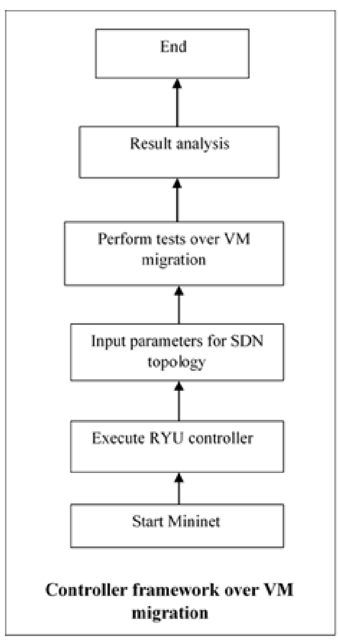
For handling the jobs that exceeds the lifetime limit, weighted End property might be employed. As per the job's assets, jobs having lifetime under limit might be united over the jobs combinations that has lifetime limit higher. On doing this, the jobs serving has the demand that has lifetime beyond the limit that might come down. The weighted end property is being employed through the Bounded and the Normal Pareto distributions. Because of their asset, the presented approach employs the Pareto distribution intended for the entire kinds of lifetime computation of the job. The job's lifetime calculation shows a dynamic part in recognizing the rate of arrival, waiting time, time of response and the time of job service at the stated framework. A novel attempt was carried out for migrating the active jobs in spite of older job in the environment. In the point of economic view, the Pareto distribution is being employed in several circumstances for identifying particular solutions to specific issue in an entire dimension like specified through problem parameters. This is one such purpose for selecting Pareto distribution in present approach for identifying thorough lifetime of job since this is mentioned as an exponential distribution factor previously.

2.2 RYU-SDN Controller Using Mininet Protype Network

Ryu is one such popular controllers among industry having NTT lab support having an active community documentation in depth. Ryu is written entirely in Python and it employs green threads for implementing the multithreading. This in turn offers provides software components that allows software developers to extend network management and control applications for utilizing the features of SDN controller. It currently supports variety of southbound protocols such as OpenFlow, OF-Config, NETCONF, etc, whereas the proposed system uses Mininet prototype network. This supports a huge range of third party libraries which includes Open vSwitch Python buildings and aids in parse VLAN, MPLS, and so on of various protocol packets. Also, it supports sFlow and Netflow protocols that were specific to the management of network traffic. Ryu in turn supports till the version of OpenFlow 1.4. this employs encoder and decoder library for the OpenFlow packets and in turn comprises OpenFlow controller in its heart. This comprises of major executables termed Ryu manager where they listen to the specified IP address and on the 6633 ports, a typical OpenFlow port. In the Ryu structure, the major components include event management, messaging service and so on.

Mininet employs process-based virtualization for running the switches and the hosts in the operating system. This in turn requires each process having their individual routing and the ARP table, and the network interfaces. The Hosts are emulated as the bash process at which each code runs like a normal Linux shell. The virtual Ethernet (Veth) is employed for connecting all network devices. The application of Mininet can be installed either on the traditional Linux system or this could be downloaded in the prepackaged VM that runs on VirtualBox.

Figure 1. Controller framework over VM migration



The Open Flow Protocol Specification of the SDN controllers allows the controllers for informing switches at which place the data packets are to be delivered. Open Flow is regarded as the one which permits the researchers for performing the protocols that are experimental on the networks that was used daily in addition it specifies the messages formats that migrate from one controller to the others. Open Flow is applied to the commercial device network like routers, switches, and wireless access points as a capability that offers structured connection for allowing researchers to conduct tests deprived of manufacturers that discloses its devices' network internal operations. OpenFlow is regarded as the one that is now adopted by major suppliers, besides the switches underlying were available in the marketplace nowadays.

Open Flow Controller is an object contact system utilizing the Open Flow Protocol requirements with the underlying switches. The controller OpenFlow in most instances is a software that are high-level and manages several switches in OpenFlow logically. An OpenFlow is an internet connection among the Open Flow button in addition to Open Flow controller that is used to handle the switch by the controller.

In this approach, Mininet is used to build the RYU controller SDN network. Mininet network emulator is employed which is employed for running the experiment. This Mininet in turn creates the

Figure 2. Event handling representation of RYU controller

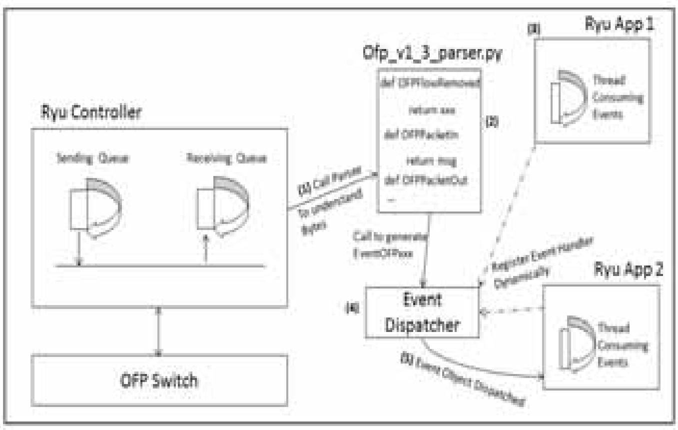
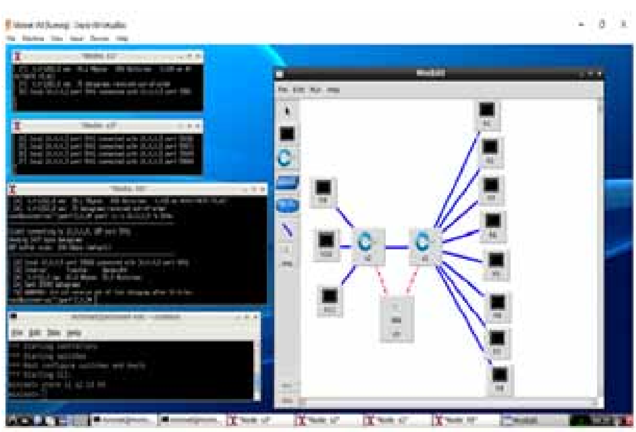


Figure 3. Representation of Mininet VM



virtual realistic network that runs in real kernel, application and switch code on the single machine at minimum time period with an interface of command line.

2.3 Ranking Methods Using Server Distinction Based Equivalent Active Weight (SD-EAW)

The presented SD-EAW algorithm in turn concentrates on the calculation of weight for each and every job of the cloud framework. The weight computation need is to recognize the non-active and active jobs in both continuous and discrete intervals of time. The approach of SDN for the cloud framework is an efficient one on offering higher range of QoS. Once after the active job's identification, these jobs were migrated depending on the algorithms procedure for offering efficient load balancing cloud framework. At first, the jobs weight is checked for identifying if the weight of job is equal. This is carried out for performing the job scheduling among VMs. The VMs might process varied weights of the job.

Once after this computation, the jobs that were higher than the time estimated were forwarded in the instruction Queue, the Queue that is presented for SD-EAW process. The jobs that are having higher time of estimation are regarded as the active jobs by means of Pareto distribution computation. On following this, the cloud framework make span was reduced and the efficient cloud environment is then developed. The minimum VM in the sense loaded, this means not only the VM consist of lesser jobs number, the VMs having minimum time for completion of assigned tasks were supposed to be the virtual machines that are loaded least.

Algorithm 1: Proposed Sd-Eaw Scheme

```
Input: Active Jobs (J1 to Jn), Weights (Wg) of active jobs
Components used: Cloud's Virtual Machine (VM)
Output: Migration Performed (?^8_{n=1} Jn * VMn)
Begin Migration
Compute n * n (VMn1, n2); Estimating VMn1 1 VMn2
i.e VMn1 is distinct with VMn2
/* VMn1 is distinct with VMn2 indicates that both the VMs have
different workload. Here
for easy evaluation we took only two VMs for this algorithm
structure. This algorithm
will work for 'n' number of VMs*/
Declare((VMn1, n2), Wgs) = (J1 to Jn)
/*indicates the weights of the active jobs are given to the VMs in
the cloud environment*/
if(J1 to Jn) = =Y
/*Y indicates the distinct property of the job weights. This
property achieved by the weight calculation of jobs through Pareto
distribution*/
        Then
                Differentiate (J1 to Jn)
                /*Jobs are differentiated. Each job has different
weight Checked among the 'n' jobs*/
                else (J1 to Jn) = =Z
                /*Z indicates equivalent property of the job weights*/
                End if
        While(J1 to Jn) = = Y
```

```
                {Schedule =  (J1;....Jn) * ∑_{n=1}^{∞} VM_n }
        Check (J1 to Jn) = = ∑_{n=1}^{∞} VM_n £ ET
                \*the algorithm checks whether the jobs scheduled
                  to the VMs are below the
                         estimated time (ET)*/
                if(∑_{n=1}^{∞} VM_n ³ ET)
                Then
                        Migrate the jobs to some other VMs
                        \*if a job processing in a VM is superior
to the time estimated at which stall the specific VM, for avoiding
this job migration to another least loaded VM*/
End if
        Migration Process
```

$$\frac{J_1}{J_1 \text{ to } J_n} > ET \rightarrow I_Q$$

```
                /*The job which is higher on comparing time
estimated (ET) to forward Instruction Queue*/ From this line the
job will be served to the least
                               loaded VM*/
                Migrate
```

$$\frac{I_Q(J_1)}{J_1 \text{ to } J_n} \rightarrow \sum_{n=1}^{\infty} J_n * VM_n$$

```
                /*From this queue the job will be served to the
least loaded VM*/
```

On promoting the active jobs to the queue instruction, this is done on the initial stage by themselves because of their environment's service time that has been reduced. As the active jobs might take excess time for processing, the residual jobs should have to wait in the pool of scheduling for processing till the active jobs complete their execution process. The presented SD-EAW algorithm functions well over all traditional approaches and in turn offers an optimum solution through minimizing the cloud environment's make span.

4. PERFORMANCE ANALYSIS

The performance analysis of the proposed system is depicted in this section. The comparative analysis of the proposed and existing system (Mekala, 2019) in terms of VMM, SLA violation time, SLA violation of SLATAH (SLAV of each active host), energy consumption is estimated at the utilization rate of resource δ=0.31 0.25, for evaluating the impacts and the outcomes attained are shown in this section.

4.1 Performance Metrics

The performance metrics are described which is employed for estimating the response time and the accuracy of proposed algorithm:

Table 1. Comparative analysis of proposed and existing system

Algorithm	Energy (kWh)	VMM	SLAV (10^{-5})	SLATAH (%)
Initial	285.05	570	0.9210	0.03549
RFAware	260.34	515	0.8321	0.03975
Sercon	245.79	426	0.7456	0.03112
ERVS (δ=0.31)	210.14	369	0.4123	0.02559
ERVS (δ=0.25)	211.50	375	0.4910	0.02971
SD-EAW(δ=0.31) (proposed)	209.38	341	0.4011	0.02122
SD-EAW(δ=0.25) (proposed)	210.04	356	0.4872	0.02463

1. The energy consumption refers to the utilization of total amount of energy of the cloud data centers and is defined by the following equation:

$$EC = \overset{h_m}{\underset{m=1}{?}} \int_t^0 (?_u \cdot Max(?_i) + (1 - ?_u) \cdot Max(?_i) \cdot d_i^{mips}(t))\, dt \tag{11}$$

Here, m signifies the number of VM, $?_u$ =0.7 and it refers to the energy consumption that are unused by the server, $Max(?_i) = 250W$ and is defined as the utilization of energy by host at their running condition and $d_i^{mips}(t)$ represents the CPU utilization at time t.

2. SLAV (service level agreements violation) is referred to the combination of both PDF and SLATAH metrics.
3. SLATAH (SLA violation time of each active host) is defined as the time proportion at which the active hosts consist of 100% utilization of CPU and is represented by the following equation:

$$SLATAH = \frac{1}{M} \overset{M}{\underset{i=1}{?}} \frac{T_{SLA_i}}{T_{ACT_i}} \tag{12}$$

Here, M is represented as the number of PM (physical machine), and T_{SLA_i} signifies the time of SLA violation at ith PM and T_{ACT_i} is signifies as the ith PM active time.

4. The migration process's performance degradation (PDM) is represented by the following equation:

$$PDM = \frac{1}{p} \overset{p}{\underset{j=1}{?}} \frac{PD_j}{CD_j} \tag{13}$$

Here, p signifies the VMs total number, PD_j is a performance deprivation that is determined that could be restrained intended for further 10% CPU utilization of the jth VM that are instigated by migrations and CD_j refers to a total CPU capacity that is required by jth VM.

Hence:

$$SLAV = SLATAH \times PDM$$

4.2 Comparative Analysis of Proposed and Existing Technique

The energy consumption analysis is estimated for the proposed algorithm and is compared with existing techniques and is shown in Table 1.

Figure 4 is the representation of number of migrations for varied range of VMs. The analysis is estimated and is compared with existing techniques like Initial, RFAware, Sercon, ERVS, and proposed strategy. The analysis shows that the proposed system is better on comparing existing techniques.

Figure 5 is the illustration of energy consumption analysis for varied range of VMs. The analysis is estimated and is compared with existing techniques like Initial, RFAware, Sercon, ERVS, and proposed strategy. The analysis shows that the proposed system is better on comparing existing techniques.

Figure 6 shows the energy consumption before and after migration process of VMs. From the analysis it was evident that the energy consumption after migration is lower than before migration process.

Figure 7 shows the resource utilization before and after migration process of VMs. From the analysis it was evident that the resource utilization after migration is higher than before migration process.

The comparative analysis of proposed and existing techniques are carried out for the SLAV to prove the effectiveness of proposed system and is shown in figure 8.

The comparative analysis of proposed and existing techniques is carried out for the task completion time in terms of cloudlet ID to prove the effectiveness of proposed system and is shown in figure 9.

Figure 10 is the depiction of makespan representation of proposed SD-EAW approach illustration simulated for number of VMs.

Figure 4. Number of migration analysis

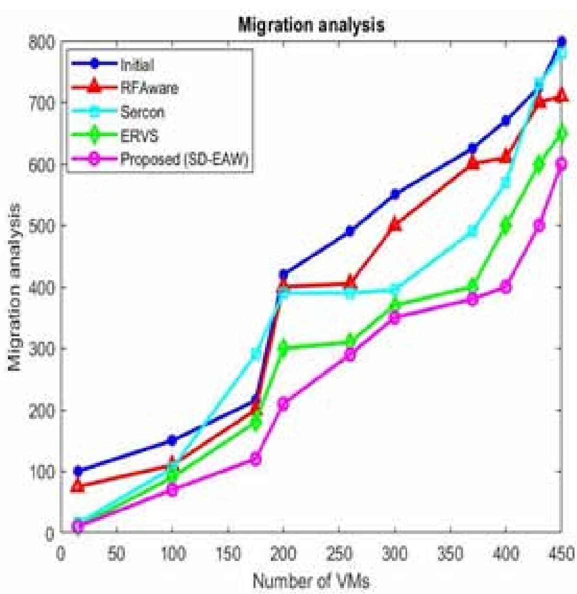


Figure 5. Energy consumption analysis

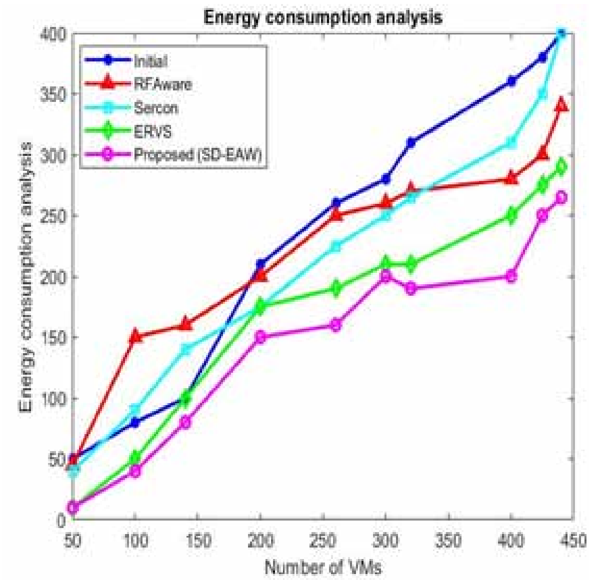


Figure 6. Analysis of energy consumption before and after migration

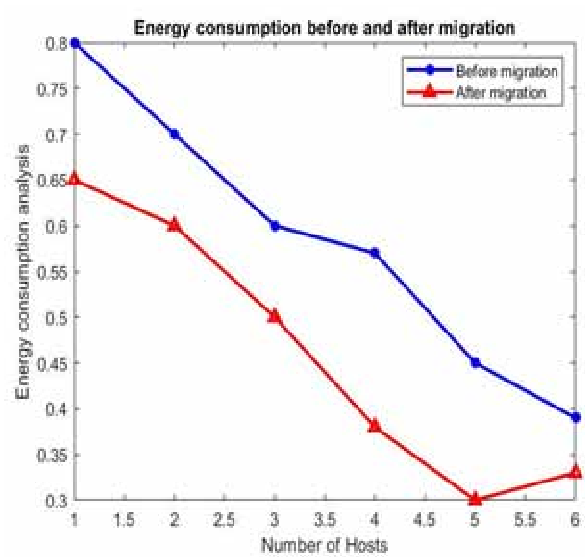


Figure 11 illustrates the resource utilization rate of the host of proposed system at which most of the node consist of good balance and resource utilization extremely. Thus, from the evaluation, it was evident that the migration count is reduced adequately and in turn preserves the consumption of energy. Thus, the proposed system is said to be effective than other traditional techniques.

Figure 7. Analysis of resource utilization before and after migration

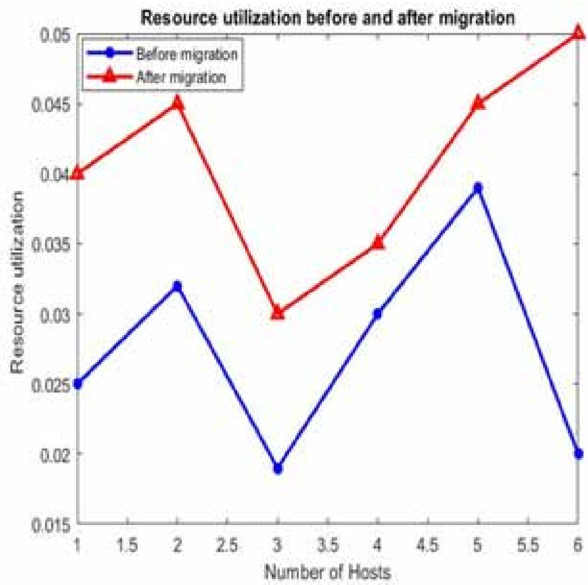
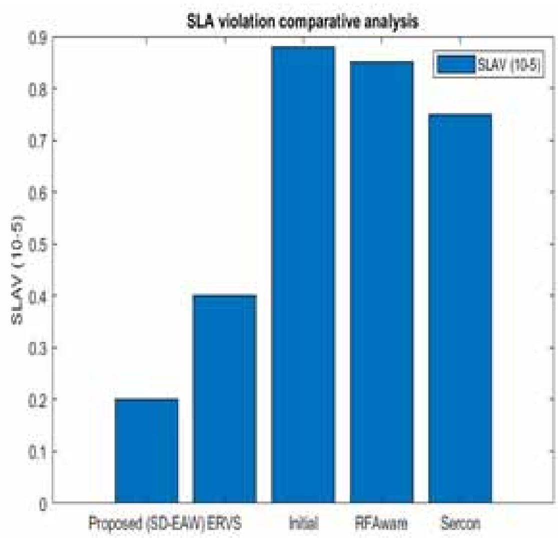


Figure 8. Comparative analysis of SLAV



5. CONCLUSION

In this approach, an efficient cloud environment is built based on SDN for producing the optimal solution and thus to maintain QoS for cloud framework. An efficient load balancing mechanism termed SD-EAW ranking algorithm was presented for the live migration of active jobs over several virtual machines was implemented in a real cloud testbed. In the proposed approach experiment, several heterogeneous resources were considered for processing the jobs. The presented technique employs

Figure 9. Comparative analysis of task completion time

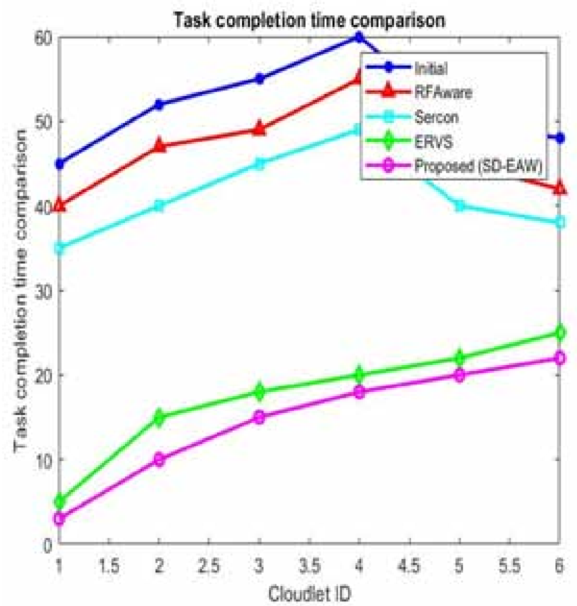


Figure 10. Make span representation of proposed SD-EAW approach

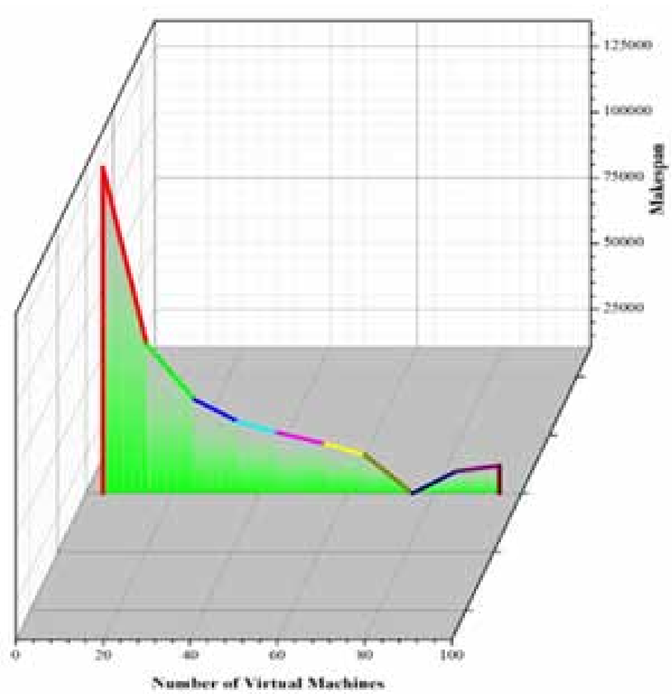
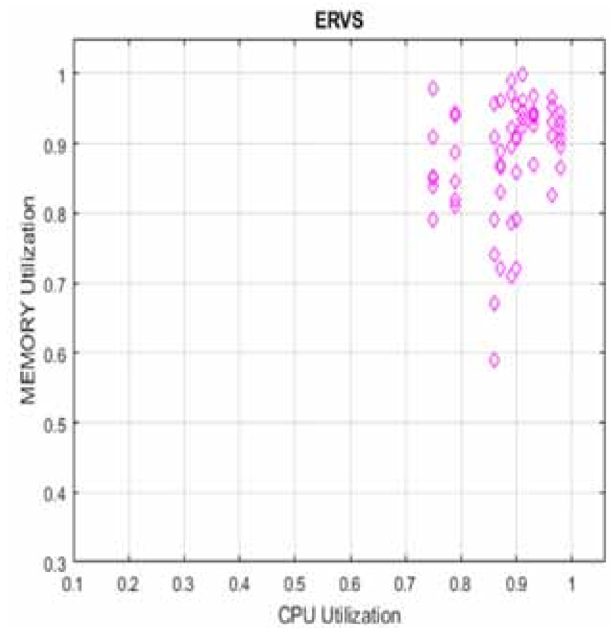


Figure 11. Resource utilization rate using proposed approach



Ryu-SDN controller with Mininet that provides software components that allows software developers to extend network management and control applications for utilizing the features of SDN controller. Also, the presented approach exploits Pareto distribution for computing the exact lifetime of active jobs. The presented technique performs well for producing optimal make span and migration time for 5G cloud framework. The performance is estimated in terms of consumption of energy, utilization of resource, migration analysis, and task completion time analysis. The rate of resource utilization of proposed system host at which most of the node consist of good balance and resource utilization extremely. Thus, from the evaluation, it was evident that the migration count is reduced adequately and in turn preserves the consumption of energy. Thus, the proposed system is said to be effective than other traditional techniques.

CONFLICT OF INTEREST

The authors of this publication declare there is no conflict of interest.

FUNDING INFORMATION

No Funding Information. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

REFERENCES

Abdelmoniem, A. M., & Bensaou, B. J. C. D. (2016). *SDN-based incast congestion control framework for data centers: Implementation and evaluation*.

Amoore, L. (2018). Cloud geographies: Computing, data, sovereignty. *Progress in Human Geography*, *42*(1), 4–24. doi:10.1177/0309132516662147

Deshpande, U., & Keahey, K. (2017). Traffic-sensitive live migration of virtual machines. *Future Generation Computer Systems*, *72*, 118–128. doi:10.1016/j.future.2016.05.003

Devi, D. C., & Uthariaraj, V. R. (2016). Load balancing in cloud computing environment using improved weighted round robin algorithm for nonpreemptive dependent tasks. *TheScientificWorldJournal*, 1–14. doi:10.1155/2016/3896065 PMID:26955656

Forsman, M., Glad, A., Lundberg, L., & Ilie, D. (2015). Algorithms for automated live migration of virtual machines. *Journal of Systems and Software*, *101*, 110–126. doi:10.1016/j.jss.2014.11.044

Ge, X., Cheng, H., Guizani, M., & Han, T. (2014). 5G wireless backhaul networks: Challenges and research advances. *IEEE Network*, *28*(6), 6–11. doi:10.1109/MNET.2014.6963798

Kliazovich, D., Pecero, J. E., Tchernykh, A., Bouvry, P., Khan, S. U., & Zomaya, A. Y. (2016). CA-DAG: Modeling communication-aware applications for scheduling in cloud computing. *Journal of Grid Computing*, *14*(1), 23–39. doi:10.1007/s10723-015-9337-8

Kowsigan, M., & Balasubramanie, P. (2019). An efficient performance evaluation model for the resource clusters in cloud environment using continuous time Markov chain and Poisson process. *Cluster Computing*, *22*(S5), 12411–12419. doi:10.1007/s10586-017-1640-7

Langmead, B., & Nellore, A. (2018). Cloud computing for genomic data analysis and collaboration. *Nature Reviews. Genetics*, *19*(4), 208–219. doi:10.1038/nrg.2017.113 PMID:29379135

Lu, C.-T., Chang, C.-W., & Li, J.-S. (2015). Vm scaling based on hurst exponent and markov transition with empirical cloud data. *Journal of Systems and Software*, *99*, 199–207. doi:10.1016/j.jss.2014.10.011

Madni, S. H. H., Abd Latiff, M. S., & Coulibaly, Y. (2016). Resource scheduling for infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities. *Journal of Network and Computer Applications*, *68*, 173–200. doi:10.1016/j.jnca.2016.04.016

Mekala, M. S., & Viswanathan, P. (2019). Energy-efficient virtual machine selection based on resource ranking and utilization factor approach in cloud computing for IoT. *Computers & Electrical Engineering*, *73*, 227–244. doi:10.1016/j.compeleceng.2018.11.021

Mishra, S. K., Sahoo, B., & Parida, P. P. (2020). Load balancing in cloud computing: A big picture. *Journal of King Saud University-Computer and Information Sciences*, *32*(2), 149–158. doi:10.1016/j.jksuci.2018.01.003

Mohan, K., & Palanisamy, B. (2019). A novel resource clustering model to develop an efficient wireless personal cloud environment. *Turkish Journal of Electrical Engineering and Computer Sciences*, *27*(3), 2156–2169. doi:10.3906/elk-1807-187

Noshy, M., Ibrahim, A., & Ali, H. A. (2018). Optimization of live virtual machine migration in cloud computing: A survey and future directions. *Journal of Network and Computer Applications*, *110*, 1–10. doi:10.1016/j.jnca.2018.03.002

Senyo, P. K., Addae, E., & Boateng, R. (2018). Cloud computing research: A review of research themes, frameworks, methods and future research directions. *International Journal of Information Management*, *38*(1), 128–139. doi:10.1016/j.ijinfomgt.2017.07.007

Shirvani, M. H., Rahmani, A. M., & Sahafi, A. (2020). A survey study on virtual machine migration and server consolidation techniques in DVFS-enabled cloud datacenter: Taxonomy and challenges. *Journal of King Saud University-Computer and Information Sciences*, *32*(3), 267–286. doi:10.1016/j.jksuci.2018.07.001

Stergiou, C., Psannis, K. E., Kim, B. G., & Gupta, B. (2018). Secure integration of IoT and cloud computing. *Future Generation Computer Systems*, *78*, 964–975. doi:10.1016/j.future.2016.11.031

Tchernykh, A., Schwiegelsohn, U., Alexandrov, V., & Talbi, E. (2015). Towards understanding uncertainty in cloud computing resource provisioning. *Procedia Computer Science*, *51*, 1772–1781. doi:10.1016/j.procs.2015.05.387

Varghese, B., & Buyya, R. (2018). Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems*, *79*, 849–861. doi:10.1016/j.future.2017.09.020

Wang, Z., Wu, J., Wu, Y., Deng, S., & Huang, H. (2017).QoS aware dynamic pricing and scheduling in wireless cloud computing in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)* (pp. 3702-3707). IEEE.

Zhang, Q., Chen, Z., & Yang, L. T. (2015). A nodes scheduling model based on Markov chain prediction for big streaming data analysis. *International Journal of Communication Systems*, *28*(9), 1610–1619. doi:10.1002/dac.2779

Zhao, Y., Liu, J., Fang, Q., Xu, L., Du, C., & Xiaoqun, Y. (2018). A strategy for improving NetClust server placement for multicloud environments. *Turkish Journal of Electrical Engineering and Computer Sciences*, *26*, 115–124. doi:10.3906/elk-1704-206

Grace Shalini T. completed her PhD from Anna University Chennai, Tamil Nadu and under graduation in Computer Science and Engineering at Magna College of Engineering, Tiruvallur, Tamil Nadu. Later she finished her post-graduation in the same discipline from Velammal College of Engineering and Technology, Madurai. Also she completed M.B.A (System Software) from Madurai Kamaraj University, Madurai. Her primary research interest in Vanets and Wireless Sensor Networks.

Rathnamala S. is working as an associate professor and has 15 years of teaching experience.