A Predictive Virtual Machine Placement in Decentralized Cloud using Blockchain

Suresh B.Rathod [1]

Symbiosis International University India

Summary
Host's data during transmission. Data tempering results in loss of host's sensitive information, which includes number of VM, storage availability, and other information. In the distributed cloud environment, each server (computing server (CS)) configured with Local Resource Monitors (LRMs) which runs independently and performs Virtual Machine (VM) migrations to nearby servers. Approaches like predictive VM migration [21] [22] by each server considering nearby server's CPU usage, roatative decision making capacity [21] among the servers in distributed cloud environment has been proposed. This approaches usage underlying server's computing power for predicting own server's future resource utilization and nearby server's resource usage computation. It results in running VM and its running application to remain in waiting state for computing power. In order to reduce this, a decentralized decision making hybrid model for VM migration need to be proposed where servers in decentralized cloud receives, future resource usage by analytical computing system and takes decision for migrating VM to its neighbor servers. Host's in the decentralized cloud shares, their detail with peer servers after fixed interval, this results in chance to tempering messages that would be exchanged in between HC and CH. At the same time, it reduces chance of over utilization of peer servers, caused due to compromised host. This paper discusses, an roatative decisive (RD) approach for VM migration among peer computing servers (CS) in decentralized cloud environment, preserving confidentiality and integrity of the host's data. Experimental result shows that, the proposed predictive VM migration approach reduces extra VM migration caused due over utilization of identified servers and reduces number of active servers in greater extent, and ensures confidentiality and integrity of peer host's data.

Keywords:
Virtual Machine (VM), Computing Host (CH), Decisive Host (DH).

1. Introduction

In recent years, cloud computing gaining popularity because of virtualization. Virtualized resources deployed, provisioned and released with minimal management effort [2]. Virtualization addresses varying resource requirement by incorporating partitioning, isolation, and encapsulation [1]. Virtual Machine (VM) is a core element in a cloud environment. It runs on the top of the hypervisor and utilizes underlying host's resource. Each VM differs from other VM by resource, CPU architecture [20], operating system, storage type, network utilizations and the job [20] it has. As a result, hosts in DC have multiple VM's running parallel with different job completion time. Static threshold limit on underlying hosts resources degrades the host's performance. Migration helps to improve hosts performance.

The migration in cloud categorized as, task migration or VM migration. In task migration, tasks from one VM migrated to other VM's in same or different host in the data center. The VM migration, in the data centers involve migrating VM, and its associated memory pages to the other hosts in same or different data centers.

Cloud computing on the structure of organizations categorized as, centralized or a decentralized architecture. Several cloud providers like Google, Amazon, HP, and IBM provides services by adopting either centralize or decentralized cloud architecture. Various authors have discussed approaches to maintain utilization to under normal threshold. These VM migration decision framework considers host's current CPU utilization. As the workload on host's in datacenters varies as per the running applications in the VM. In decentralized cloud environment, individual host takes decision for reducing its CPU utilization. To do so, it selects neighbor host's considering peer host's CPU utilization which it has received from peer hosts. This leads to the problem of selecting same host for VM placement by the multiple host's, if more than one host gets over utilized at the same time.

This leads a requirement to consider host's future CPU utilization in decision making, and a hybrid framework which could avoid the same host selection by multiple host's, and also preservers host's data shared with peer host.

1.1 Authors Contribution

Most of the authors considered current CPU utilization of the host, future utilization of the VM, or two threshold resource limits as the parameter for VM migration. Here, this paper proposes decentralized predictive. This work contributes following attributes.

- Host categorization as per the role
- Proposing hybrid P2P based predictive VM placement considering dynamic threshold usage of source and destination CH.

Manuscript received April 5, 2024
Manuscript revised April 20, 2024
https://doi.org/10.22937/IJCSNS.2024.24.4.7

- Finding CH's future CPU utilization by applying Double Exponential Smoothing (DES).
  - Resolving selection of same destination host by any CH with proposed hybrid model.
  - Using Blockchain to share host information

1.2 A Paper Structure

This paper, discuss the predictive decentralized Peer to Peer(P2P) VM Placement (DPPVP), that initiates VM migration decision by considering dynamic threshold and hosts future utilization. The remaining portion of this paper organized as follows: section 2 describes the related work, section 3 the proposed system, section 4 discusses the results of the proposed system and at the last conclusion.

2. Related Work

Energy based VM placement approach proposed by the author in [6] discussed VM migration by considering penalty cost and energy consumption as the parameters for the VM selection. The solution proposed by the authors [6] suffers from the limitation that if the energy cost and penalty cost increased the overall performance degrades. CPU utilization based distributed load balancing proposed by the author in [7] considered hypercube based VM placement and migration. In the work proposed by author [7], individual host takes decisions for VM migration without considering the destination hosts future CPU utilization. Authors In [8] have proposed optimum dynamic VM Placement policy considering hosts CPU consumption; they have discussed the maximum processing power (MPP) and random host's selection (RS) as an approach for VM migration. VM migrated to destination host by preserving VM's firewall rule. In [9] the author have proposed Hierarchical Decentralized Dynamic VM Consolidation Framework for VM migration, wherein they discussed how the global controller takes decision for VM migration by considering hosts future CPU utilization. The author in their work proposed the solution for VM placement using Ant Colony Optimization (ACO) technique. Distributed load balancing using CPU utilization proposed by the author in [10] considered hypercube based VM migration. Randomized probabilistic technique for distributed live VM migration proposed by [10] , discussed hosts pair formation and initiating VM migration in the selected host pair. Correlation based VM placement on centralized cloud architecture proposed by [12]. Cluster based VM consolidation proposed by the author [13], where they have discussed batch oriented VM consolidation and on demand VM placement. Muti target based VM placement using genetics algorithm proposed by the author in [14] considered SLA violation and CPU utilization as the parameter for VM migration decision making on centralized cloud architecture. In [15], authors have proposed Reinforcement Learning based VM placement wherein the authors discussed how centralized host learn VM deployment and puts host in sleep mode or in active mode considering the past traces.

3. Proposed work

This section discusses the problem formulation followed by the proposed work in the current research.

3.1 Problems Formulation

The mapping of VM to the physical host gives the solution to the VM placement. Let C be the set of physical host represented as $C = \{CH_1, CH_1, \ldots\ldots CH_m\}$and V be the set of virtual hosts deployed on the physical server denoted as $V = \{VM_1, VM_2, VM_3 \ldots., VM_n\}$. $V_{i,j}$ be the virtual machine i deployed on the physical host j, such that (1<i<n) and(1<j<m). X_(i,j) be the binary decision variable representing whether the VM_i selected from the hostC_j to be placed on one of the host from C hosts. The mapping of V_i to the host C_j such that the C_jhas minimum CPU utilization at time t.

$$\forall \sum_{j=1}^{m} X_{i,j} \quad (1)$$

$$\forall_j \sum_{i=1}^{m} \mathrm{VMcpu}, \mathrm{i}X_{i,j} \leq C_{cpu,j} \quad (2)$$

$$\forall_j \sum_{i=1}^{m} VMmem, iX_{i,j} \leq C_{mem,j} \quad (3)$$

Where i is the virtual server and j is the physical host. The above equation (2) and (3) discusses the virtual server should not exceed the physical resources in normal VM placement.

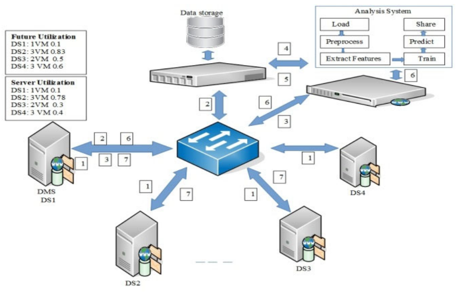

Fig. 1 Generalized decentralized cloud computing environment

3.2 Proposed hybrid decentralized predictive VM placement

The proposed hybrid decentralized cloud architecture formed by considering distributed features like multi-tenant architecture, distributed storage, parallel processing and multithreading. Each host in proposed architecture configured with the agents as shown in Fig.1 and each hosts categorizing as Controller Host (CH) and Host Controller (HC).

Host termed as CH, if it has VM's instances. Host termed as HC, if it does the decisions for VM placement, and has running VM instances. Each host configured with below components.

HC Resource Monitor (HCRM): This is the component available at the CH. It gets activated only when the CH acts as HC and does make decisions for VM migration. It performs tasks like collecting and storing peer hosts detail, providing host information to the Virtual Host Manager (VHM) as and when required.

Local Resource Monitor (LRM): This component available at each CH. It interacts with the underlying hypervisor, collects underlying host detail and shares this c ollected information with the HCRM.

Virtual Host Manager (VHM): Unlike HCRM, it gets activated whenever the CH acts as HC. It identifies source and destination for VM migration, predicts future CPU utilization of hosts, finds upper threshold, stores hosts detail in tables.

2. Tables, Figures and Equations

2.1 Tables and Figures

To insert "Tables" or "Figures", please paste the data as stated below. All tables and figures must be given sequential numbers (1, 2, 3, etc.) and have a caption placed below the figure ("FigCaption") or above the table("FigTalbe") being described, using 8pt font and please make use of the specified style "caption" from the drop-down menu of style categories

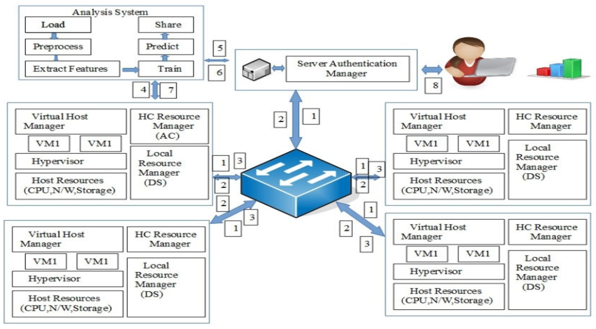

Fig.2 Decentralized Hybrid Host Component Diagram

Initially, each CH establishes secure connection with the central host. The central host acknowledges every CH with the newly elected HC address. Below Algorithm FINDHC is for HC selection algorithm.

```
Algorithm FINDHC.
1: procedure HCSELECTION($HOSTS$)
2:     Address= Random (0, HOSTS.length())
3:     $returnAddress$
4: end procedure
```

The peer hosts here CH, after receiving HC address, establishes secure connection with the HC, and starts sharing underlying server's detail with HC in the form as shown in table 1

Table 1.Host's Information exchange form

Address	No.of VM	CPU utilization	Status	Time

.

```
Algorithm DPVP
1: procedure DOPREDICTIVEMIGRATION($hostList$,$currUtil$)
2:     $med$ ←med[length(currUtil) ]
3:     $median$ ← $median[length(currUtil)]$
4:     for eachUtil host in currUtil do
5:         if hostUtil.getvalue() > threshold then
6:             maxUtilServer.put(hostUtil.getKey(),hostUtil.getValue())
7:         end if
8:     end for
9:     for each host in maxUtilServer do
10:        srcAddress=host
11:        util=findUtilization(currUtil[host])
12:        while util < threshold do
13:            util=findUtilization(hostList,currUtil)
14:            minServer=findMinServer(currUtil)
15:            vmut=findMinVMUtil(srcAddress,currUtil)
16:            setVmUtil(vmut)
17:            putil=doForecast(minServer)
18:            if $oldIndex = index$ then
19:            else
20:                fhost=dofindFutureData()
21:            end if
22:            for each host in currUtil do
23:                if minServer == currUtil[host] then
24:                    for each serv in med do
25:                        if $med[serv] == minServer$ then
26:                            address=med[serv].getKey()
27:                            if $med[serv] == minServer$ then
28:                                medValue=med[serv].getValue()
29:                                if medvalue > 0.9 then
30:                                    medvalue=0.9
31:                                end if
32:                                if medvalue > putil then
33:                                    address=FINDNEXT(srcAddress,address,fhost)
34:                                    vm=findMinVMUtil(srcAddress,currUtil)
35:                                    migrate(srcAddress,address,vm)
36:                                else if putil < medvalue then
37:                                    vm=findMinVMUtil(srcAddress,currUtil)
38:                                    migrate(srcAddress,address,vm)
39:                                end if
40:                            end if
41:                        end if
42:                    end for
43:                end if
44:            end for
45:        end while
46:    end for
47:    for each hostUil in currUtil do
48:        if hostUtil.getvalue() < threshold then
49:            minUtilServer.put(hostUtil.getKey(),hostUtil.getValue())
50:        end if
51:    end for
52:    for each host in minUtilServer do
53:        srcAddress=host
54:        util=findUtilization(currUtil[host])
55:        while util < 0 do                    1
56:            Go to 13
57:        end while
58:    end for
59: end procedure
```

Here, the status flag is to identify HC host, if it is true the CH acts as an HC. Address to identify IP address of the CH. No.of VM, it is integer value specifying number of running VM instances. CPU utilization is floating value, representing CPU utilization of each server, and the time for the time wherein underlying server's utilization has been retrieved.

The VHM at HC, initiates the procedure for VM migration, by giving a call to DPVP algorithm. DPVP algorithm is shown below.

HC after receiving CH details initiates daemon threads to collect all peer CH's detail, manage CH's running VM, and do identify next HC. It is a daemon thread that wakes up's after fixed interval, and performs above mentioned job. Eq. (4) used to reprieve hosts CPU utilization.

$$H_U = \sum_{i=0}^{n} VM_i \qquad (4)$$

Here H_U is the host utilization of server u[20]. It is the sum of all virtual servers [20]VM_irunning on the host u at time interval t.

```
Algorithm    FindMinServer.
procedure FINDMINSERVER($hostUtil$)
    address=""
    $minUtil \leftarrow$hostUtil[0].util
    for each host in hostUtil do
        if $hostUtil[host] \leq minUtil$ then
            $minutil \leftarrow$hostUtil[host].getKey()
            address=hostUtil[host].getValue()
        end if
    end for
    return address
end procedure
```

TheVHM at HC calls store daemon thread to store CH's information in current and past utilization table. HC referres CHs addresses to find source host and destination host during VM migration. As the host's in data centers have heterogeneous configuration, each has varying threshold limit. Each host's upper threshold computed using equation (6), which is computed after finding MAD [9] value using Eq.(5)

$$\text{MAD} = \frac{\sum_t^n y_{t-\hat{y}}}{n} \qquad (5)$$

Here, y_t represent actual CH's utilization and n represent a number of observations and $\hat{y}$ represent fitted value at time t.

UpperThreshold=1-MAD (6)

The VHM at HC, do initiates a thread to find the CH with maximum and minimum CPU utilization findMinServer and FindNext. After this, CH's CPU utilization detail retrieved from the current CPU utilization table. VHM starts searching the VM that has mminimum resource consumption. It is being selected as it requires less time, and less resource migration compared with the VM with maximum CPU utilization. Upon CH with maximum CPU utilization, and minimum CPU utilized server identified, it initiates VM migration from maximum utilized server to minimum utilized [20] server. The VM placement is successful if it satisfies the following conditions.

- The identified destination CH has its future CPU utilization lesser than its upper threshold
- Upper threshold is less than or equal to 0.9.
- The VM placement to the minimum CH discarded, if it satisfies following conditions.
- If the CH's current utilization of destination host is greater than 0.9.
- Future utilization greater than upper threshold.

If VHM, if it finds VM placement conditions then VM migrated from the source CH to the newly identified CH would be initiated. Here, DPVP performs checks, to find whether the selected host's future CPU utilization would be crossing upper threshold, if so then it finds new CH else VM would be migrated to the selected host. The future CPU utilization of each host would be computed using Doubles Exponential Smoothing (DES) [16], and its smoothed value computed using Eq. (9).

$$S_t = \alpha y_t + (1-\alpha)(s_{t-1} + b_{t-1}), 0 \leq \alpha \leq 1 \qquad (7)$$

$$b_t = \gamma(s_t - s_{t-1}) + (1-\gamma)b_{t-1}), 0 \leq \gamma \leq 1 \qquad (8)$$

$$f_{t+m} = s_t + mb_t \qquad (9)$$

Here S_t represents CH's smooth values[20] at time t, the y_t represents observed values over a time period t[20]. b_t represent trend factor over time period t values for the previous [20]periodb_{t-1}. This f_{t+m}called the smoothing function.

DPVP give a call to the FINDNEXT procedure, if the current CH is not suitable for VM placement, and needs new CH identification. The pseudo code for FINDNEXT algorithms is shown below.

```
Algorithm FINDNEXT.
procedure FINDNEXT(src,dest,fhostUtil)
    address=""
    for each host in fhostUtil do
        address=host.key
        if src == address then
        else if dest==address then
            dest
        else
            if host.value() ≤ 0.90 then
                address=host.key
                break
            end if
        end if
    end for
    return address
end procedure
```

The HC tries to maintain each CH's in normal state by searching the CH with max and min utilization. The CH said to be in normal state, if it CPU utilization is less than 0.7 and greater than 0.1 as shown in fig.4.4. CH said to be over utilized [20] if its current CPU utilization is greater than 0.7[20].If CH is underutilized, in that case HC will initiates process called as MigrateAllVMs. Algorithm for MigrateAllVMs is shown below.

```
Algorithm MigrateAllVMs.
procedure DOMIGRATEALL(hostList,currUtil)
    for each host in currUtil do
        util=findUtilization(currUtil[host])
        while util ⟨ threshold do
            util=findUtilization(hostList,currUtil)
            minServer=findMinServer(currUtil)
            doPredictiveMigration(host,minServer,hostList,currUtil)
        end while
    end for
end procedure
```

It will migrate all VM's from current CH to rest of CH whose current and future CPU utilization is lesser than lower limit as 0.1,0.2,03,0.4. These upper and lower limits would be set by the administrator whoever is managing CHs.

4. Result and Discussion

The proposed framework developed by considering hybrid peer to peer network topology. Here, simulation for decentralized cloud has been created using core java. VM's utilization taken from Azure [23].

When all CHs boot up, CH's starts connecting central host. The central hosts calls HC identification and marks one of the CH as the HC. After initial HC identification done, it shares current HC address with all connected CHs. Here, every CH's receives, name of HC and the public key of the HC from the current HC. Each CH's updates public address, and public key of the new HC. It uses these details while sharing information with the new HC. New HC, uses its private key and reads data shared by the CH. Each CH, after receiving the HC address, starts collecting their underlying server's details and starts sharing server details with the HC.

HC, after receiving each server's details, it initiates the process to store CH's details in its current CPU utilization table, and updates the past utilization table too, after an fixed interval which administrator had set before booting process initiations. Table 2 shows snippet of current CPU utilization table. HC refers this current utilization table to identify the hosts during VM migration, next HC identification, and CH's future CPU utilization.

The past utilization table referred by the HC to predict destination host's future CPU utilization and to find the destination hosts new upper threshold. From Table.4.1the host with address 10.0.0.3 has the minimum CPU utilization at current instant of time and the host with address 10.0.0.1 has maximum CPU utilization. The HC marks 10.0.0.3 as the destination host and 10.0.0.1 as the source host. VHM at HC applies the DPPVP and checks whether the 10.0.0.1 has any VM running on it, if any then marks the VM for migration that has maximum CPU utilization compared with other VM on same CH. If it does not found any VM on the 10.0.0.1, then the new CH searched here 10.0.0.2 one VM selected that has maximum CPU utilization and marked to be placed on the 10.0.0.3. Before VM to be placed the future load of the 10.0.0.3 calculated using Eq. (6).

Table 2. Current utilization table at HC.

SERVER	*CPU utilization*	*NO.VM*	*Status*
10.0.0.1	0.212	4	FALSE
10.0.0.2	0.098	2	FALSE
10.0.0.1	0.13	1	TRUE
10.0.0.3	0.168	3	FALSE
10.0.0.2	0.128	3	FALSE
10.0.0.1	0.133	1	TRUE
10.0.0.3	0.125	2	FALSE
10.0.0.2	0.165	4	FALSE
10.0.0.1	0.135	1	TRUE
10.0.0.3	0.153	3	FALSE

This future CPU utilization compared with the above mentioned conditions, if it satisfies the above condition the VM from the identified CH placed on the 10.0.0.3 else the procedure of new CH initiated. The new CH would be identified such that it has less CPU utilization compared with current CH and the future CPU utilization of newly identified CH would be less compared with current upper threshold. After CH gets identified VM migration from the originating host to the newly identified CH would be initiated.

Fig.3 shows the non-predictive VM placement. From fig. 3 it found that some servers are overloaded whereas some are underutilized, and some servers are shut down. Here we found the server 10.0.0.29 is over utilized whereas server 10.0.0.3 is underutilized.

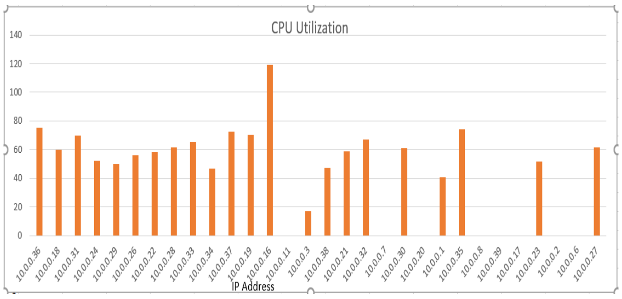

Fig.3 Randomized VM placement in Decentralized cloud environment

Fig.4 shows the results of proposed framework. In Fig.4 VM migration at time t has been specified. From fig.4, it found that, proposed framework has better CPU utilization, and has server's utilization below threshold usage limit.

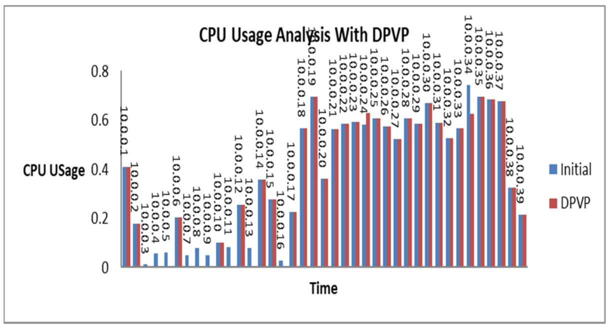

Fig.4. Predictive VM placement using DPPVP.

The comparative analysis of proposed work with randomized peer to peer VM placement is shown in Fig 5. In fig.5, 10 represent 10 percent of CPU utilization whereas 70 as 70% of CPU utilization.

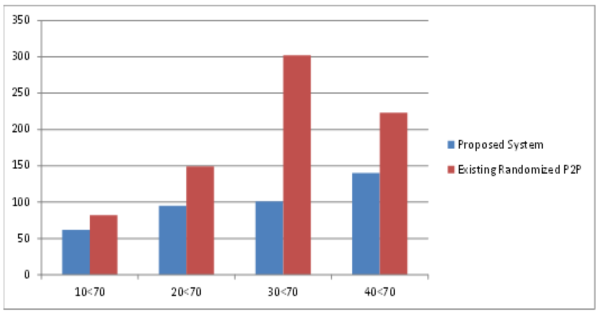

Fig.5 Comparison of proposed DPVP with Randomized peer to peer based VM migration

Table 3 shows comparative analysis of proposed work with othesrs

Table 3. Comparative Analysis.

Author	*Architecture Type*	*Decision Making*	*Predictive Decision Making*	*Destination host information sharing*	*Considered blockchain for Data Sharing*
[1][5]	Central	Central	No	No	No
[19]	Central	Central	No	No	No
.[18]	Central	Central	No	No	No
[17]	Distributed	Peer Host	No	No	No
[4]	Distributed	Peer Host	No	No	No
[5] [7]	Distributed	Peer Host	Yes	No	No
[6] [8]	Distributed	Peer Host	Yes	No	No
[7] [9]	Distributed	Peer Host	Yes	No	No
[8][11]	Distributed	Peer Host	Yes	No	No
[9]	Distributed	Peer Host	Yes	No	No
[10]	Distributed	Peer Host	Yes	No	No
[11]	Distributed	Peer Host	Yes	No	No

Conclusion

Inproposed hybrid decentralized architecture the host categorization into CH and HC helps in reducing network bandwidth consumption and processing power in peer hosts by restring network traffic between CH and HC. The decision for VM placement done by the HC, it helps in avoiding same destination host selection by multiple hosts during VM placement. Considering destination hosts current and future CPU utilization helps to avoid unnecessary VM placement and hosts overloading due to VM placement. Using blockchain in data exchange in between HC and CH reduces chance of reading message, and message format which CH and HC uses to communicate with each other, and at the same time hides next HC details from other hosts.

References

[1] National Institute of Standards and Technology NIST[online].Available https://www.nist.gov/.

[2] Cloud computing dened Characteristics service levels- Cloud computing news[Online].Available https://www.ibm.com/blogs/cloud-computing/2014/01/cloud-computing-defined-characteristics-service-levels/.

[3] Paraiso F, Merle P. A Study of Virtual Machine Placement Optimization in Data Centers, 7th International Conference on Cloud Computing and Services Science, CLOSER 2017, Porto, Portugal, pp.343-350(2017),

[4] Feller E, Morin C, Esnault A. A case for fully decentralized dynamic VM consolidation in clouds,CloudCom 2012 - Proc 2012 4th IEEE Int Conf Cloud Comput Technol Sci., Tai- wan ,pp. 26-33(2012).

[5] Wen, Wei-Tao,Wang, Chang-Dong,Wu, De-Shen, Xie, Ying-Yan. An ACO-based Scheduling Strategy on Load Balancing in Cloud Computing Environment 2015 Ninth Int Conf Front Comput Sci Technol, China,pp. 364- 369(2015).

[6] Grygorenko D, Farokhi S, Brandic. Cost-aware VM placement across distributed DCs using Bayesian networks, Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics),9512,32-48(2016).

[7] Benali R, Teyeb H, Balma A, Tata S, Ben Hadj-Alouane. N. Evaluation of traffic-aware VM placement policies in distributed cloud using Cloud Sim, Proc - 25th IEEE Int Conf Enabling Technol Infrastruct Collab Enterp WETICE 2016, France, pp. 95-100(2016).

[8] Bagheri Z, Zamanifar K.. Enhancing energy efficiency in resource allocation for real-time cloud services. 7th Int Symp Telecommun IST 2014, Iran, 701-706(2014).

[9] Ferdaus MH, Murshed M, Calheiros RN, Buyya R. An algorithm for network and data aware placement of multitier applications in cloud data centers. J Netw Comput Appl.,65-83(2017).

[10] Pantazoglou M, Tzortzakis G, Delis A. Decentralized and Energy-Efficient Workload Management in Enterprise Clouds.IEEE Trans Cloud Computing, 4(2), 196-209(2015),

[11] Nikzad S. An Approach for Energy Efficient Dynamic Virtual Machine Consolidation in Cloud Environment, International Journal of Advanced Computer Science and Applications,7(9), 1-9(2016).

[12] Zhao Y, Huang W.. Adaptive Distributed Load Balancing Algorithm Based on Live Migration of Virtual Machines in Cloud, Fifth Int Jt Conf INC, IMS IDC, Nexus, 170-175(2009).

[13] Fu X, Zhou C.Virtual machine selection and placement for dynamic consolidation in Cloud computing environment. Front Comput Sci., 9(2), 322-330(2015).

[14] Teng F, Yu L, Li T, Deng D, Magouls F.. Energy efficiency of VM consolidation in IaaS clouds, J Supercomput, 73(2), 782-809(2017).

[15] Arianyan E, Taheri H, Sharian S. Multi target dynamic VM consolidation in cloud data centers using genetic algorithm, J Inf Sci Eng.,32(4),1575-1593(2016).

[16] Double exponential smoothing Insight Central [Online].Availablehttps://analysights.wordpress.com/tag/double-exponential-smoothing/(2017).

[17] Mukhtarov M. Cloud Network Security Monitoring and Response System. International Transactions on Systems Science and Applications, 8(3),181-185(2012).

[18] Anala M.R., Kashyap M., Shobha G Application performance analysis during live migration of virtual machines, Advance Computing Conference (IACC), 2013 IEEE 3rd International , Mysore,India, 366-372.(2013).

[19] Tavakoli, Zahra,Meier, Sebastian,Vensmer, Alexander.A framework for security context migration in a firewall secured virtual machine environment,Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),7479 LNCS,41-51(2012).

[20] S.B.Rathod, V.Krishna Reddy, Decentralized Predictive Secure VS Placement In Cloud Environment, Journal of Computer science, Vol.14, Issue 4,pp- 396-407,(2018.)

[21] S.B.Rathod, V.KrishnaReddy,Predictive Virtual Machine Placement in Decentralized Cloud Environment, ICICEL Express Letters, Vol. 12, Issue 12 September 2018.

[22] S.B.Rathod, V.Krishna Reddy, Decision making framework for Decentralized Virtual Machine Placement, International Journal of Engineering Technology (UAE) Vol.7, Issue 2.7, pp-705-709, March, 2018.

[23] https://github.com/Azure/AzurePublicDataset/blob/master/AzurePublicDatasetV1Links.txt.

suresh B.Rathod received the Ph.D from KLEF Deemed to be University, Vijayawada, A.P. India in 2019. He is working as Asst.Professor in symbiosis Institute of Technology, Pune, India.

Manuscript received April 5, 2024
Manuscript revised April 20, 2024
https://doi.org/10.22937/IJCSNS.2024.24.4.7