ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 2, pp: (30-42), Month: April - June 2016, Available at: www.researchpublish.com

Effective Usage of Unutilized Resource for Distributed System in Closed Network

[1]MR.M.DILLI BABU, [2]S.K.PRAVEEN, [3]R.KAMALESH, [4]M.ARVIND, [5]R.PRAVEEN

[1](Asst.Prof), [2,3,4,5] IT Department, Panimalar Engineering College

Abstract: **In high energy consumption of cloud data centers is a great concern of data accessing of dynamic consolidation of virtual machines in this present method if used to significant opportunity to save energy in large data centers. In this method the VM consolidation approach the use of live migration of VM that time the under loaded physical machines can be switch off or put into low power mode. At same time to achieving the exact level of quality of services provide between the cloud provider and the users is very critical, in this paper id main challenge is reduce the energy consumption of data centers for providing require Quality of Services. In this paper we providing distributed virtual machines consolidation to reduce the energy consumption of cloud data center. Finally we done the nearest optimal virtual machines method and this paper automatically allocate the virtual machines machine without buy new virtual machines in cloud computing.**

Keywords: **Dynamic VM consolidation, ant colony system, cloud computing, green computing, energy-efficiency, SLAF.**

1. INTRODUCTION

Cloud computing is a relatively new computing paradigm. It leverages several existing concepts and technologies, such as data centers and hardware virtualization, and gives them a new perspective. Cloud computing provides three service models and four deployment models. The three service models are Infrastructure as a Service, Platform as a Service (PaaS), and Software as a Service (SaaS). Similarly, the four deployment models are private cloud, community cloud, public cloud, and hybrid cloud. With its pay-per-use business model for the customers, cloud computing shifts the capital investment risk for under or over provisioning to the cloud providers. Therefore, several public IaaS, PaaS, and SaaS cloud providers, such as Amazon, Google, and Microsoft, operate large-scale cloud data centers around the world. Moreover, due to the ever increasing cloud infrastructure demand, there has been a significant increase in the size and energy consumption of the cloud data centers. High energy consumption not only translates to a high operating cost, but also leads to higher carbon emissions. Therefore, energy-related costs and environmental impacts of data centers have become major concerns and research communities are being challenged to find efficient energy-aware resource management strategies. On the other hand, achieving the desired level of Quality of Service (QoS) between cloud providers and their users is critical for satisfying customers' expectations concerning performance. The QoS requirements are formalized via Service Level Agreements (SLAs) that describe the required performance levels, such as minimal throughput and maximal response time or latency of the system. Therefore, the main challenge is to reduce energy consumption of data centers while satisfying QoS requirements. The QoS requirements are formalized via Service Level Agreements (SLAs) that describe the required performance levels, such as minimal throughput and maximal response time or latency of the system. Therefore, the main challenge is to reduce energy consumption of data centers while satisfying QoS requirements. Over the past few years, there have been several attempts to reduce energy consumption of data centers. Currently, the two widely-used techniques are dynamic server provisioning and Virtual Machine (VM) consolidation. Dynamic server provisioning approaches save energy by using a reduced amount of resources needed to satisfy the workload requirements. Therefore, unnecessary servers are switched-off or put into a low-power mode when the work-load demand decreases. Similarly, when the demand increases, additional servers are switched-on or put back into a high-power mode. Dynamic VM consolidation is another effective way to improve the utilization of resources and their energy-efficacy It leverages the hardware virtualization technology, which shares a Physical Machine (PM) among multiple performance-isolated platforms called VMs, where each VM runs one or more application tasks. The sharing of the PM resources among multiple VMs

Is handled by the Virtual Machine Monitor (VMM). Therefore, virtualization takes dynamic server pro-visioning one step further and allows different applications to be allocated on the same PM to im-prove the resource utilizations. Moreover, it allows live VM migration and consolidation to pack VMs on a reduced number of PMs, reducing the energy consumption [10]. However, in order to maximize resource utility, it is essential to manage PM resources in an adequate manner. Therefore, one of the most important optimization problems concerning VM consolidation is energy-efficient placement of VMs on PMs. Furthermore, to be able to cope with the work-load variability of different types of applications, the VM consolidation should be performed in an online manner.

In this paper, we address the VM consolidation problem with the objective to reduce energy consumption of data centers while satisfying QoS require-ments. We present a distributed system architecture to perform dynamic VM consolidation to improve resource utilizations of PMs and to reduce their energy consumption. We also propose a dynamic VM consolidation approach that uses a highly adaptive online optimization **metaheuristic algorithm called Ant Colony System** (ACS) [11], [12] to optimize VM placement. The proposed ACS-based VM Consolidation (ACS-VMC) approach uses artificial ants to consolidate VMs into a reduced number of active PMs according to the current resource requirements. These ants work in parallel to build VM migration plans based on a specified objective function. The performance of the proposed ACS-VMC approach is evaluated by using CloudSim [13] simulations on real workload traces, which were obtained from more than a thousand VMs running on servers located at more than 500 places around the world. The simulation results show that ACS-VMC maintains the desired QoS while reducing energy consumption in a cloud data center. It outperforms existing VM consolidation approaches in terms of energy consumption, number of VM migrations, and number of SLA violations.

The remainder of this paper is organized as follows. Section 2 discusses some of the most important related works and briefly reviews the Ant Colony Optimization (ACO) **metaheuristic.** Section 3 and Section 4 present the system architecture and the proposed dynamic VM consolidation approach, respectively. Sec-tion 5 describes the experimental design and setup. Finally, we present the experimental results in Section 6 and our conclusions in Section 7.

2. BACKGROUND AND RELATED WORK

The existing VM consolidation approaches, such as [14], [15], [16], [17] are used in data centers to reduce under-utilization of PMs and to optimize their energy-efficiency. The main idea in these approaches

In existing all the new needs the cloud or service provider buy new VMs from the server. It will increase the cost and VM usage and it also increase the power energy conception. Here no dynamic VM creation is not possible. Is to use live VM migration [10] to periodically consolidate VMs so that some of the under-loaded PMs can be released for termination. Determining when it is best to reallocate VMs from an overloaded PM is an important aspect of dynamic VM consolidation that directly influences the resource utilization and QoS. In [18], two static thresholds were used to indicate the time of VM reallocation. This approach keeps the total Central Processing Unit (CPU) utilization of a PM between these thresholds. However, setting static thresholds is not efficient for an environment with dynamic workloads. Therefore, Beloglazov and Buyya [19] presented adaptive thresholds that can be derived based on the statistical analysis of the historical data.

Another important aspect of dynamic VM consolidation concerns load prediction on a PM. Using a prediction of the future load enables proactive consolidation of VMs on the overloaded and under-loaded PMs. Therefore, in our previous works [20], [21], we proposed two regression methods to predict CPU utilization of a PM. These methods use the linear regression and the K-nearest neighbor regression algorithms, respectively, to approximate a function based on the data collected during the lifetimes of the VMs. Therefore, we used the function to predict an overloaded or an under-loaded PM for reducing the SLA violations and energy consumption.

The VM consolidation problem is known to be NP-hard [15], [22]. Therefore, it is expensive to find an optimal solution with a large number of PMs and VMs. In some of the existing approaches, VM consolidation has been formulated as an optimization problem with the objective to find a near optimal solution by using a greedy approach [22], [23], [24]. Since an optimization problem is associated with constraints, such as data center capacity and SLA, these works use a heuristic to consolidate workload in a multi-dimensional bin packing problem. The PMs are assumed to be the bins and the VMs are considered as the objects. The objective of bin packing is to minimize the number of bins while packing all the objects. Wood et al. [22] used a greedy algorithm to determine a sequence of moves to migrate overloaded VMs to under-loaded PMs. Ajiro and Tanaka [23] used a load balancing algorithm called least-loaded (LL) to balance the load among PMs. Wang et al [24] formulated the VM consolidation problem as a stochastic bin packing problem and used an online packing algorithm to consolidate VMs with dynamic bandwidth demands.

In our proposed system dynamic VM concerns load prediction on the PM. Using the prediction method the feature load enables

Proactive of consolidation of VMs on the overload and under-loaded PMs. We are propose K nearest neighbor to approximate a function based on the data collected during the lifetimes of the VMs. Therefore, we used the function to predict an overloaded or an under-loaded PM for reducing the violations and energy consumption.

In this paper, we formulate energy-efficient VM consolidation as a multi-objective combinatorial optimization problem and apply a highly adaptive online optimization [25] **metaheuristic called Ant Colony Optimization (ACO)** [11] to find a near-optimal solution. ACO is a multi-agent approach to difficult combination optimization problems, such as traveling salesman problem (TSP) and network routing [12].

3. EXPERIMENTAL DESIGN AND SETUP

To evaluate the efficiency of our proposed approach, we set up experimental environment using the CloudSim toolkit [13]. CloudSim is a discrete event simulator for implementation and evaluation of resource provisioning and VM consolidation tech-niques for different applications. We simulated a data center comprising 800 heterogeneous PMs and selected two server configurations in CloudSim: HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores 1860 MHz, 4 GB), and HP ProLiant ML110 G5 (Intel Xeon 3075, 2 cores 2660 MHz, 4 GB). Dual-core CPUs are sufficient to evaluate resource management methods that are designed for multi-core CPU architectures. Moreover, it is important to simulate a large number of servers for performance evaluation of VM consolidation methods. To evaluate the efficiency of our proposed approach, we measured four metrics: SLA violations, energy consumption, number of migrations, and energy-SLA violations. The results are based on two different workloads: a random work-load and a real workload. In the random workload, the users submit requests for the provisioning of 800 heterogeneous VMs. In the real workload, the number of VMs on each day is specified in Table 2. The ACS parameters that were used in the proposed approach are tabulated in Table 3. These parameter values were obtained in a series of preliminary experiments.

3.1 SLA Violations:

Maintaining the desired QoS is an important requirement for cloud data centers. QoS requirements are commonly formalized in the form of SLAs, which specify enterprise service-level requirements for data center in terms of minimum latency or maximum response time. Beloglazov and Buyya [19] proposed a workload independent metric called SLAV (SLA Violations) to evaluate the SLA delivered by a VM in an IaaS cloud. It represents both the SLA Violations due to Over-utilization (SLAVO) and SLA Violations due to Migrations (SLAVM). The SLAVO

and SLAVM metrics independently and with equal importance characterize the level of SLA violations by the infrastructure. Therefore, a combined metric (SLAV) describes performance degradations due to the overloading of the host PMs as well as those caused by VM migrations, as

$$SLAV = SLAV\,O \;\; SLAV\,M \qquad (13)$$

SLAVO indicates the percentage of time, during which active PMs have experienced the CPU utilization of 100%. It is defined as

$$SLAV\,O = \frac{1}{M}\sum_{i=1}^{M}\frac{T_{s_i}}{T_{a_i}} \qquad (14)$$

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 2, pp: (30-42), Month: April - June 2016, Available at: www.researchpublish.com

Table 2: Number of VMs in the real workload

Date	Number of VMs
3 March	1052
6 March	898
9 March	1061
22 March	1516
25 March	1078
3 April	1463
9 April	1358
11 April	1233
12 April	1054
20 April	1033

Table 3: ACS parameters in the proposed approach

				q_0	nA	nI	w
0:1	0:9	5	0:1	0:9	10	2	$_2$ 7

Where M is the number of PMs; T_{si} is the total time that the PM i has experienced the utilization of 100% leading to an SLA violation. T_{ai} is the total duration of the PM i being in the active state. SLAVM shows the overall performance degradation by VMs due to migrations. It is computed as

$$SLAVM = \frac{1}{N}\sum_{j=1}^{N}\frac{C_{dj}}{C_{rj}} \qquad (15)$$

Where N is the number of VMs; C_{dj} is the estimate of the performance degradation of the VM j caused by migrations; C_{rj} is the total CPU capacity requested by the VM j during its lifetime. Based on our preliminary experiments, we estimated C_{dj} as 10% of the CPU utilization in MIPS during all migrations of the VM j.

3.2 Energy Consumption:

We consider the total energy consumption of the physical resources in a data center that is required to handle the application workloads. The energy consumption of a PM depends on the utilization of its CPU, memory, disk, and network card. Most studies [19], [31] show that CPU consumes more power than memory, disk storage, and network interface. Therefore, the resource utilization of a PM is usually represented by its CPU utilization. Instead of using an analytical model of energy consumption, we used the real data in the SPECpower benchmark[1]. Table 4 illustrates the amount of energy consumption of HP G4 and G5 servers at different load levels.

3.3 Number of VM Migrations:

Live VM migration is a costly operation that includes some amount of CPU processing on the source PM, the link bandwidth between the source and destination PMs, the downtime of the services on the migrating VM, and the total migration time [16]. Therefore, one of our objectives was to minimize the number of migrations. The length of a VM migration in CloudSim takes as long as it needs to migrate the memory assigned to the VM over the network bandwidth link between source and destination PMs. In our simulations, we used 1Gbps network links.

3.4 Energy and SLA Violations:

The objective of the proposed VM consolidation approach is to minimize both energy and SLA violations. Since there is a trade-off between performance and energy consumption, we measured a combined metric called ESV (Energy and SLA Violations) that captures both the Energy Consumption (EC) and the SLA Violations (SLAV) as

$$ESV = EC\ \ SLAV \qquad (16)$$

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 2, pp: (30-42), Month: April - June 2016, Available at: www.researchpublish.com

4. EXPERIMENTAL RESULTS

In this section, we compare the proposed ACS-VMC approach with an ACS based VM consolidation algorithm in [27] and four heuristic algorithms for dynamic reallocation of VMs in [19]. The AVVMC consolidation scheme [27] proposes the ant colony optimization with balanced usage of computing re-sources based on vector algebra. Moreover, the main idea of these algorithms [19] is to set upper and lower utilization thresholds and keep the total CPU utilization of a node between them. When the upper threshold is exceeded, VMs are reallocated for load balancing and when the utilization of a PM drops below the lower threshold, VMs are reallocated for consolidation. The algorithms adapt the utilization threshold dynamically based on the Median Absolute Deviation (MAD), the Interquartile Range (IQR), and Local Regression (LR) approach to estimate the CPU utilization. Moreover, a static threshold method (THR) is proposed in [19] that monitors the CPU utilization and migrates a VM when the current utilization exceeds 80% of the total amount of available CPU capacity on the PM. In our experiments, we consider two type of workloads:

1. http : ==www:spec:org=power ssj2008=

4.1 Random Workload:

In the random workload, each VM runs an application with a variable utilization of CPU, which is generated with a uniform distribution. Figure 2(a) presents the SLA violation levels caused by the ACS-VMC, AVVMC, THR, MAD, IQR, and LR methods in the random workload. The results indicate that ACS-VMC reduced the SLA violations more efficiently than the other approaches. This is due to the fact that ACS-VMC prevents SLA violations by using a pre-diction of the overloaded PMs and that the heuristic value in (8) ensures that the destination PM does not become overloaded when a VM migrates on it. Figure 2(b)shows that the proposed dynamic VM consolidation approach, ACS-VMC, brought higher energy savings in comparison to the other approaches in the random workload. In ACS-VMC, a signicant reduction of the energy consumption of 7:3%, 16:4%, 45:2%, 34:5%, and 47:7% was achieved when com-pared to AVVMC, LR, MAD, THR, and IQR, respectively. In addition, the trade-off between maximizing the QoS and minimizing the energy consumption of data center is demonstrated in Figure 2(c). Figure 2(d) depicts the total number of VMs migration during the VM consolidation in the random workload. The ACS-VMC outperforms the AVVMC and adaptive-threshold based algorithms due to predictions of utilization, and therefore decreased the number of VM migrations.

4.2 Real Workload:

Real workload data is provided as a part of the CoMon project, a monitoring infrastructure for PlanetLab [32]. In this project, the CPU usage data is collected every five minutes from more than a thou-sand VMs and is stored in different files. The VMs are allocated on servers that are located at more than 500 places around the world. In fact, the workload is representative of an IaaS cloud environment such as Amazon EC2, which several independent users create and manage VMs. Figure 3a shows that ACS-VMC led to significantly less SLA violations than the other four benchmark algorithms. The main reason is that ACS-VMC employs measures to prevent VM migrations that would result in the overloading of the destination PM. Moreover, it preemptively reallocates VMs from a predicted overloaded PM. Figure 3b shows that ACS-VMC consumed less power than the other benchmark algorithms in the real workload traces. our proposed VM consolidation approach reduces energy consumption by up to 53:4% with desirable system performance in March 2011 load traces. This is because, the defined objective function tries to maximize the number of dormant PMs by packing VMs into the PMs that have enough capacity. The number of VM migrations in the real workload are shown in Figure 4a. As observed from the results, ACS-VMC has minimum number of migrations compared with the other benchmark methods. Because it creates a migration plan that has require the minimum number of migrations. In addition, Figure 4b illustrates the ACS-VMC consumes less ESV than other benchmarks algorithms in the real workload traces.

TABLE 4: Energy consumption at different load levels in watts

Server	sleep mode	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP ProLiant G4	10	86	89:4	92:6	96	99:5	102	106	108	112	114	117
HP ProLiant G5	10	93:7	97	101	105	110	116	121	125	129	133	135

Feasibility Study:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

Economical Feasibility:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social Feasibility:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

5. SYSTEM ARCHITECTURE

A cloud data center consists of m heterogeneous PMs that have different resource capacities. Each PM contains a CPU, which is often a multi-core. The CPU performance can be defined in terms of Millions of Instructions Per Second (MIPS). In addition, a PM is also characterized by the amount of memory, network I/O, and storage capacity. At any given time, a cloud data center usually serves many simultaneous users. Users submit their requests for provisioning of n VMs, which are allocated to the PMs. The length of each request is specified in millions of instructions (MI). In our proposed approach, the VMs are initially allo-cated to PMs based on the Best Fit Decreasing (BFD) algorithm, which is the best known heuristic for the bin-packing problem [16]. BFD first sorts all VMs by their utilization weights in the decreasing order. Then, it starts with the VMs that require the largest amount of resources. The BFD algorithm allocates VMs in such a way that the unused capacity in the destination PMs is minimized. Thus, it selects a PM for which the amount of available resources is closest to the requested amount of resources by the VM. However, due to dynamic workloads, the resource utilizations of VMs continue to vary over time. Therefore, an initial efficient allocation approach needs to be aug-mented with a VM consolidation algorithm that can be applied periodically. In our proposed approach, the ACS-VMC algorithm is applied periodically in order to adapt and optimize the VM placement according to the workload.

Figure 1 depicts the proposed system model that consists of two types of agents: local and global agent. A Local Agent (LA) resides in a PM to solve the PM status detection sub-problem by observing the current resource utilizations of the PM. The Global Agent (GA) acts as a supervisor and optimizes the

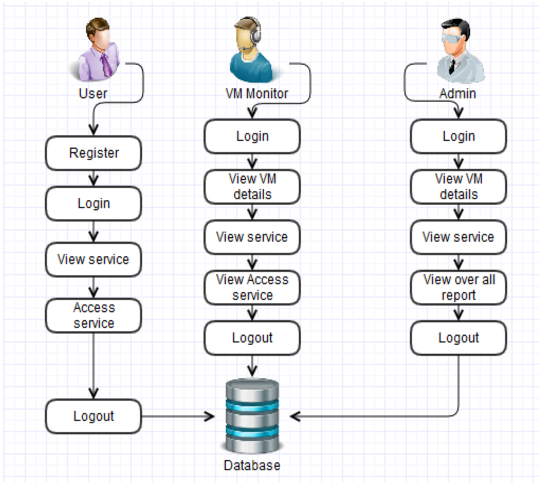


Fig. 1: The system architecture

VM placement by using the proposed ACS-VMC algorithm. The task sequence of these agents is described as follows:

1) Each LA monitors the CPU utilization and

categorizes the PM into one of the four sets

P_{normal}, P_{over}, $\hat{P}_{over}$, and P_{under}. Respectively,

these sets represent the normal, overloaded, predicted overloaded, and under-loaded PMs based on the following conditions:

If the current CPU utilization exceeds PM capacity, the PM is considered as a member

of P_{over}.

If the predicted utilization value is larger than the available CPU capacity, the PM

is considered as a member of $\hat{P}_{over}$. We

use LiRCUP [20] to forecast the short-term CPU utilization of a PM based on the linear regression technique. In LiRCUP, the linear regression approximates the utilization function according to the past utilization values in a PM.

If the current CPU utilization is less than a threshold of the total CPU utilization, the PM is categorized as a member of P_{under}. We performed a series of preliminary experiments to estimate the threshold. Based on our analysis, in general, the best results are obtained when the threshold is set to 50%.

All remaining PMs belong to P_{normal}.

2) The GA collects the status of individual PMs from the LAs and builds a global best migration plan by using the proposed ACS-VMC algorithm, which is described in the next section.

3) The GA sends commands to VMMs for per-forming VM consolidation task. The commands determine which VMs on a source PM should be migrated to which destination PMs.

4) The VMMs perform actual migration of VMs after receiving the commands from the GA.

One way to estimate optimal jMj is to use the nearest neighborhood heuristic [11]. We use the K-nearest neighbor (KNN) heuristic [30] to estimate the optimal jMj by using a training data set. The data set has m samples, where each sample x_i is described by three input variables (x_{i1}; x_{i2}; x_{i3}) and an output variable y_i, that is, x_i = fx_{i1}; x_{i2}; x_{i3}; y_ig. The goal is to find a relationship between the input variables and the output variable. Therefore, we choose the number of under-loaded PMs, the number of overloaded PMs, and the number of VMs as the three input variables (x_{i1}; x_{i2}; x_{i3}) and the migration plan size as the output variable (y_i). The KNN heuristic estimates the output by taking a local average of the training data set.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 2, pp: (30-42), Month: April - June 2016, Available at: www.researchpublish.com

Moreover, the locality is defined in terms of the K samples nearest to the estimation sample. We use Euclidean distance to measure the distance metric between new sample and other samples.

The pseudo-random-proportional-rule in ACS and the global pheromone trail update rule are intended to make the search more directed. The pseudo-random-proportional-rule prefers tuples with a higher pheromone level and a higher heuristic value. There-fore, the ants try to search other high quality solutions in a close proximity of the thus far global best so-lution. On the other hand, the local pheromone trail update rule complements exploration of other high

Step 1: Obtain the subset of users within the same AS numbered *a* as user *u*, denoting them with *a i U* . Use Eq. (6) to compute the similarity between user *u* and each other users *a i v*□*U* . Search *a i U* for the top K most similar users to *u* with similarity greater than 0, and use them to construct *N*(*u*). If fewer than K users are found, proceed to Step 2; else, return *N*(*u*) as the result.

□ Step 2: Obtain the subset of users within the same country numbered *c* as *u*, denoting them with *c i U* . Use Eq. (6) to compute the similarity degrees between user *u* and each other users *c i v*□*U* . Search *c i U* for the top K most similar users to *u* with similarity greater than 0, and use them to construct *N*(*u*). If fewer than K users are found, proceed to Step 3; else, return *N*(*u*) as the result.

□ Step 3: Find the subset of users in *U* that have invoked Web service *i*, denoting them with *i U* . Use Eq. (6) to compute the QoS similarity degrees between *u* and *v* for every user (*v*) in *i U* . Search *i U* for the top K most similar users to *u* with similarity greater than 0, and use them to construct *N*(*u*). If fewer than K users are obtained, use the actual number of similar users (with similarity greater than 0) found to construct *N*(*u*).

function in (4) is applied on M_k^m, and if it yields a score higher than the ant's best score Scr_k (line 16), t is added to the ant-specific migration plan M_k (line 17). Otherwise, the tuple t is removed from the temporary migration plan M_k^m (line 19). Then, towards the end of an iteration when all ants complete their migration plans, all ant-specific migration plans are added to the set of migration plans M S (line 22). Each migra-tion plan M_k 2 M S is evaluated by applying the objective function in (4), the global best application migration plan M^+ is selected (line 24), and the global pheromone trail update rule in (9) and (10) is applied on all tuples (line 25). Finally, when all iterations of the main outer loop complete, the algorithm outputs the global best migration plan M^+.

Modules:

1. Authentication Module
2. Access service
3. Creating new VM using VMM
4. Updating VM details to local agent
5. Schedule the VM from local agent
6. View All Reports from PM
7. Buy new VM

Module Explanation:

Authentication Module: The user authentication module is to check whether the authorized user is logged in.This authentication process is verify the user name and password is valid. Before logged in one time registration is mandatory. In this module has been followed the Register page all fields fill into the mandatory. And us well enter into user name and password servers as regular log in confidential.

Access service: This module the user going to access the services from the service provider. Here the register user only access the service. User access service will be updated to the local agent the agent only getting the from accessing service user. And updated to the PM at the time of access service

Creating new VM using VMM: This module will use ti creating new VM for the user access the services it created by the VM Monitor. VM does not create VM with out use of all VM here any VM will free means the VMM allocate the service the free VM. Here all the VM monitor by the VMM.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 2, pp: (30-42), Month: April - June 2016, Available at: www.researchpublish.com

Updating VM details to local agent: This module all the VM details like (Access service, free VM, load of the VM etc.,) updated to the local agent. At the same time the local agent update the VM details to the VM Monitor, here the VMM easily find the free VM and free services.

Schedule the VM from local agent: Here this module local agent will allocate the services to the VM, VM always will get the details and services from the local agent, here the local agent will get the the service information from the VM Monitor, and local agent also update the service details to the VM Monitor.

View All Reports from PM: In this module the PM(VM Monitor) will collect all the details from the local agent and also view over all the VM load and allocated service details, here the VM Monitor will get the pie chart report about the VM and service details.

Buy new VM: In this module the VM monitor need any new VM they buy the new VM system at the time of all VM will busy, here the VM Monitor only buy the new VM. Before that the VM Monitor will check all the VM details.

Verification And Validation:

Once the program exists, we must test it to see if it is free of bugs. High quality products must meet user’s needs and expectations. Further more the product should attain this with minimal or no defects, the focus being on improving products prior to delivery rather than correcting them after delivery. The ultimate goal of building high quality software is user’s satisfaction.

There are two basic approaches to system testing.

Validation is the task of predicting correspondence, which cannot be determined until this system is in place.

Verification is the exercise of determining correctness.

Testing strategies:

The extent of testing a system is controlled by many factors, such as the risk involved, the limitations of the resources and deadlines. We deploy a testing strategy that does the best job of finding the defects in the product within the given constraints. The different testing strategies are:

- **Black Box Testing:**

The concept of black box testing is used to represent the system whose inside workings are not available for inspection. In black box testing, we try various inputs and examine the resulting outputs. Black box testing works very nicely in testing objects in object oriented environment. For inspection the input and output are defined through use cases or other analysis information.

- **White Box Testing:**

White box testing assumes that the specific logic is important and must be tested to guarantee the systems proper functioning. The main use of the white box id the error based testing. In a white box testing, the bugs are looked for that have a low probability of execution that have been overlooked previously. It is also known as path testing.

There are two types of path testing:

Statement testing coverage: where every statement in the objects method is covered by executing it at least once.

Branch testing coverage: it is to perform enough tests to ensure that every branch alternative is executed at least once.

Top down testing:

A top-down strategy supports the user interface and event driven system. This serves two purposes; first the top down approach can test navigation through screens and verify that it matches the requirement. Second, users at the early stage can see how the final application will look and feel.

Bottom up testing:

Bottom up testing starts with the details of the system and proceeds to higher levels by a progressive aggregation of details until they collectively fit requirements of the system. In this testing the methods and classes which are independent are tested.

System Security:

Introduction:

The protection of computer based resources that includes hardware, software, data, procedures and people against unauthorized use or natural.

Disaster is known as System Security.

System Security can be divided into four related issues:

- **System Security**

It refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

- **Data Security**

It is the protection of data from loss, disclosure, modification and destruction.

- **System Integrity**

It refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

- **Privacy**

It defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair or excessive dissemination of information

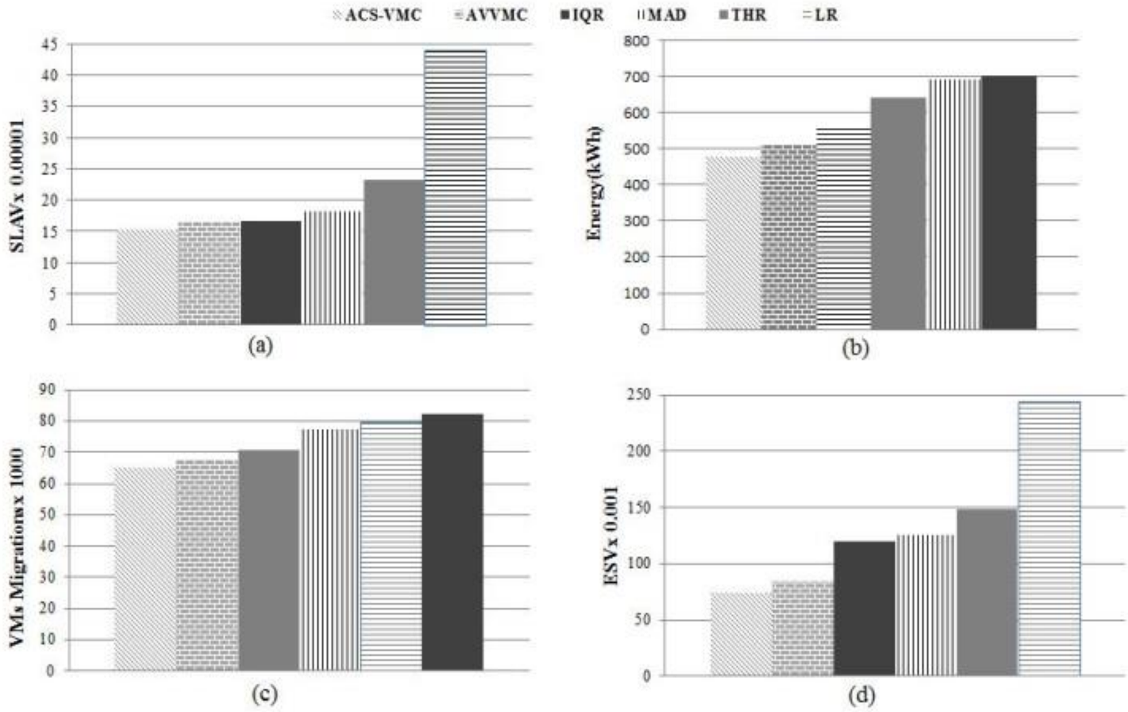
6. CONCLUSION AND FUTURE WORK

In this paper, we presented a novel dynamic Virtual Machine (VM) consolidation approach called Ant Colony System based VM Consolidation (ACS-VMC). It reduces the energy consumption of data centers by consolidating VMs into a reduced number of active Physical Machines (PMs) while preserving Quality of Service (QoS) requirements. Since the VM consolidation problem is strictly NP-hard, we used the Ant Colony System (ACS) to find a near-optimal solution. We defined a multi-objective function that considers both the number of dormant PMs and the number of migrations. When compared to the existing dynamic VM consolidation approaches, ACS-VMC not only reduced the energy consumption, but also minimized SLA violations and the number of migrations. We evaluated the performance of our proposed approach by conducting experiments with ten different real workload traces.

As a future work, we plan to further improve the proposed system model by clustering PMs and assigning them to the respective consolidation managers. We also intend to evaluate the performance of other heuristic methods for VM consolidation. Furthermore, we plan to implement the ACS-VMC algorithm as an extension of the VM manager within the OpenStack Cloud platform[2] to evaluate the proposed VM consolidation algorithm in a real cloud environment.

Server	**sleep mode**	**0%**	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
HP ProLiant G4	10	86	89:4	92:6	96	99:5	102	106	108	112	114	117
HP ProLiant G5	10	93:7	97	101	105	110	116	121	125	129	133	135

Energy Conseption in Different Loads:



REFERENCES

[1] P. Mell and T. Grance, "The NIST definition of cloud comput-ing," Recommendations of the National Institute of Standards and Technology. Special Publication 800-145., September 2011, http://csrc.nist.gov/publications/ nistpubs/800-145/SP800-145.pdf.

[2] G. Motta, N. Sfondrini, and D. Sacco, "Cloud computing: An architectural and technological overview," in Service Sciences (IJCSS), 2012 International Joint Conference on, 2012, pp. 23–27.

[3] I. Sriram and A. Khajeh-Hosseini, "Research agenda in cloud technologies," Large Scale Complex IT Systems (LSCITS), Tech. Rep., 2010.

[4] A. Ashraf, "Cost-efficient virtual machine management: Provi-sioning, admission control, and consolidation," Ph.D. disserta-tion, Turku Centre for Computer Science (TUCS) Dissertations Number 183, October 2014.

[5] A. Ashraf, M. Hartikainen, U. Hassan, K. Heljanko, J. Lilius, T. Mikkonen, I. Porres, M. Syeed, and S. Tarkoma, "Introduc-tion to cloud computing technologies," in Developing Cloud Software: Algorithms, Applications, and Tools, I. Porres, T. Mikko-nen, and A. Ashraf, Eds. Turku Centre for Computer Science (TUCS) General Publication Number 60, October 2013, pp. 1– 41.

[6] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," Advances in Computers, vol. 82, pp. 47–111, 2011.

[7] A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M. A. Kozuch, "AutoScale: Dynamic, robust capacity management for multi-tier data centers," ACM Transactions on Computer Systems, vol. 30, no. 4, pp. 14:1–14:26, 2012.

[8] L. Deboosere, B. Vankeirsbilck, P. Simoens, F. Turck, B. Dhoedt, and P. Demeester, "Efficient resource management for virtual desktop cloud computing," The Journal of Supercomputing, vol. 62, no. 2, pp. 741–767, 2012.

[9] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," SIGOPS Oper. Syst. Rev., vol. 37, no. 5, pp. 164–177, 2003.

[10] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in Proceedings of the 2Nd Conference on Symposium on Networked Systems Design & Implementation, ser. NSDI'05, vol. 2, 2005, pp. 273–286.

[11] M. Dorigo and L. Gambardella, "Ant colony system: a coop-erative learning approach to the traveling salesman problem," Evolutionary Computation, IEEE Transactions on, vol. 1, no. 1, pp. 53–66, 1997.

[12] M. Dorigo, G. Di Caro, and L. M. Gambardella, "Ant algo-rithms for discrete optimization," Artif. Life, vol. 5, no. 2, pp. 137–172, Apr. 1999.

[13] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, no. 1, pp. 23–50, 2011.

[14] W. Vogels, "Beyond server consolidation," ACM Queue, vol. 6, no. 1, pp. 20–26, 2008.

[15] E. Feller, C. Morin, and A. Esnault, "A case for fully decentral-ized dynamic VM consolidation in clouds," in Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, December 2012, pp. 26–33.

[16] A. Murtazaev and S. Oh, "Sercon: Server consolidation al-gorithm using live migration of virtual machines for green computing," IETE Technical Review, vol. 28, no. 3, pp. 212–231, 2011.

[17] M. Marzolla, O. Babaoglu, and F. Panzieri, "Server consolida-tion in clouds through gossiping," in World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a, 2011, pp. 1–6.

[18] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware re-source allocation heuristics for efficient management of data centers for cloud computing," Grid Computing and eScience, Future Generation Computer Systems (FGCS), vol. 28, pp. 755– 768, 2012.

[19] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and perfor-mance efficient dynamic consolidation of virtual machines in cloud data centers," Concurrency and Computation: Practice and Experience, vol. 24, no. 13, pp. 1397–1420, 2012.

[20] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers," in Software Engineering and Advanced Applications (SEAA), 2013 39th EU-ROMICRO Conference on, 2013, pp. 357–364.

[21] F. Farahnakian, T. Pahikkala, P. Liljeberg, and J. Plosila, "En-ergy aware consolidation algorithm based on K-nearest neigh-bor regression for cloud data centers," in Utility and Cloud Computing (UCC), 2013 IEEE/ACM 6th International Conference on, Dec 2013, pp. 256–259.

[22] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Sand-piper: Black-box and gray-box resource management for vir tual machines," Computer Networks, vol. 53, pp. 2923–2938, 2009.

[23] Y. Ajiro and A. Tanaka, "Improving packing algorithms for server consolidation," in Proceedings of the International Con-ference for the Computer Measurement Group (CMG), 2007, pp. 399–407.

[24] M. Wang, X. Meng, and L. Zhang, "Consolidating virtual machines with dynamic bandwidth demand in data centers," in INFOCOM, 2011 Proceedings IEEE, 2011, pp. 71–75.

[25] M. Harman, K. Lakhotia, J. Singer, D. R. White, and S. Yoo, "Cloud engineering is search based software engineering too," Journal of Systems and Software, vol. 86, no. 9, pp. 2225–2241, 2013.

[26] P.-Y. Yin and J.-Y. Wang, "Ant colony optimization for the nonlinear resource allocation problem," Applied Mathematics and Computation, vol. 174, no. 2, pp. 1438–1453, 2006.

[27] M. Ferdaus, M. Murshed, R. Calheiros, and R. Buyya, "Vir-tual machine consolidation in cloud data centers using ACO metaheuristic," in Euro-Par 2014 Parallel Processing, ser. Lecture Notes in Computer Science, F. Silva, I. Dutra, and V. San-tos Costa, Eds. Springer International Publishing, 2014, vol. 8632, pp. 306–317.

[28] A. Ashraf and I. Porres, "Using ant colony system to consol-idate multiple web applications in a cloud environment," in 22nd Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), 2014, pp. 482–489.

[29] M. Mishra and A. Sahoo, "On theory of VM placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach," in Cloud Computing (CLOUD), 2011 IEEE International Conference on, July 2011, pp. 275–282.

[30] T. Cover and P. Hart, "Nearest neighbor pattern classification," Information Theory, IEEE Transactions on, vol. 13, no. 1, pp. 21– 27, 1967.

[31] D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized com-puting environments via lookahead control," in Autonomic Computing, 2008. ICAC '08. International Conference on, 2008, pp. 3–12.

[32] K. Park and V. Pai, "CoMon: a mostly-scalable monitoring sys-tem for PlanetLab," ACM SIGOPS Operating Systems Review, vol. 40, pp. 65–74, 2006.