International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2019 IJSRCSEIT | Volume 5 | Issue 2 | ISSN : 2456-3307
DOI : https://doi.org/10.32628/CSEIT195253

Adaptive Resource Allocation and Provisioning in Multi-Service Cloud Environments

Anitha Nithya R*, Saran A , Vinoth R
Department of Computer Science and Engineering, Sri Krishna College of Technology, Coimbatore, Tamil Nadu, India

ABSTRACT

Minimizing the energy consumption and resource usage in cloud computing environment is one of the key research issues. Energy aware resource allocation is used to optimize the power consuming by computer resources and storage in cloud. The proposed system is to improve the utilization of computing resources and reduce energy consumption under workload independent quality of service constraints. Using migration for minimizing the number of active physical nodes the dynamic single threshold VM consolidation leverages fine-grained fluctuations in the workloads and continuously reallocates VMs . A genetic algorithm based power-aware scheduling of resource allocation (G-PARS) has been proposed to solve the dynamic virtual machine allocation policy problem. The experiment results show that strategy that has been proposed has a better performance than other strategies, not only in high Quality Of Service(QoS) but also in less energy consumption.

Keywords : Cloud Computing, Resource Utilization, Genetic Algorithm.

I. INTRODUCTION

Cloud computing has become one of the fastest growing paradigms in computer science. Cloud Computing is a model for providing IT resources as a service in a cost efficient manner. It follows pay-per-use technique. By adopting Cloud services, companies and simple users are enabled to externalize their hardware resources, services, applications and their IT functions. The key features of cloud computing are On-demand self-service, Broad network access, Resource pooling, Rapid elasticity and Measured Service. An important aspect to consider with the Cloud is the ownership and use of the Cloud infrastructure. There are three main actors involved in cloud computing that have distinct roles and interactions inside the Cloud environment viz cloud providers, brokers and users.

Cloud Provider: The cloud provider possesses the Cloud infrastructure. The cloud services are deployed on the cloud infrastructure. It is the cloud provider, who controls the cloud resources and handles the user requests.

Cloud user: A Cloud user is a person or an organization that avails the Cloud services.

Cloud Broker: The intermediate player between Cloud users and cloud provider is cloud broker. Based on users' requirements the broker distributes incoming requests between the different providers.

II. CLOUD SERVICE MODELS

Generally, the cloud model is composed of five essential characteristics, three service models, and four deployment models. Cloud service models describe how services are made available to users. We

distinguish between two different types of models: classic Cloud service models and new hybrid ones. Classic Cloud service models can be categorized into three types. They are (IaaS)Infrastructure as a Service , Platform as a Service (PaaS) and Software as a Service (SaaS).

IaaS is the most straightforward model for delivering cloud services. It refers to the provisioning and the delivery of basic resources such as virtual machines, physical servers, network and storage. The multiple applications of various types can coexist in an IaaS environment and share physical resources, it is necessary to derive a workload independent QoS metric that can be used to define system-wide QoS constraints. Migrating VMs from overloaded servers to avoid performance degradation. Migrating VMs from underloaded servers to improve the utilization of resources and minimize energy consumption. It is necessary to make a crucial decision that must be made in both situations as determining the best time to migrate VMs to minimize energy consumption, also the defined QoS constraints must be satisfied.

The proposed project aims in Determining the best placement of new VMs or the VMs selected for migration to other servers is another essential aspect that influences the quality of VM consolidation and energy consumption by the system.

III. RELATED WORKS

The previous work considers the cloud market where various resources such as CPUs, memory, and storage in the form of Virtual Machine (VM) instances can be provisioned and then leased to clients with QoS guarantees. The proposed system uses a novel Service Level Agreement (SLA) framework for cloud computing, in which a price control parameter is used to meet QoS demands for all classes in the market. The framework uses reinforcement learning(RL) to derive a VM hiring policy that can adapt to changes in the system to guarantee the QoS for all client classes. [1]. The problem which is not addressed in the above paper is, the power consumption is not considered. The proposed system addresses the power consumption thereby overcoming the problem. . Other research works includes, developing a novel auction-based scheme for trading free processors in cloud computing environment. [2], SLA-based resource provisioning for hosted software as a service applications in cloud computing environments[3], Resource trading in cloud environments for profit maximisation using an auction model[4].

IV. ENERGY CONSUMPTION CHALLENGE IN CLOUDS

The numerous advantages of cloud computing environments, including cost effectiveness, on-demand scalability, and ease of management, encourage service providers to adopt them and offer solutions via cloud models. This in turn encourages platform providers to increase the underlying capacity of their data centers to accommodate the increasing demand of new customers. The need for more energy to power these large-scale infrastructures is one of the main drawbacks of the growth in capacity of cloud data centers . Such a drastic growth in energy consumption of cloud data centers is a major concern of cloud providers.

Energy wastage in data centers are driven by various reasons such as inefficiency in data center cooling systems, network equipments, and server utilization. Nevertheless, in a data center, the servers are the main power consumers. Both the amount of work and the efficiency with which the work is performed affect the power consumption of servers. Therefore, for improving the power efficiency of data centers, the energy consumption of servers should be made more proportional to the workload.

Power proportionality is defined as the proportion of the amount of power consumed comparing to the actual workload and it can be achieved by either

decreasing servers' idle power utilization at hardware level or efficient provisioning of servers through power-aware resource management policies at software level. Although there is a large body of research on energy efficient resource management of IaaS, not enough attention has been given to PaaS environments with containers. Hence, this thesis focuses on software-level energy management techniques that are applicable to containerized cloud environments. The main objective is improving data center energy consumption while maintaining the required Quality of Service (QoS) through decreasing SLA violations. This thesis contributes to the literature by considering both containerized and enterprise cloud environments while addressing their new challenges. One of the aspects that distinguishes this thesis from the related work is that this thesis tackles the problem of data center energy consumption through the study of real enterprise cloud backend data. It also explores the potential benefits, for enterprise and containerized cloud environments, from a comprehensive cloud workload study and how it can decrease the amount of energy consumption in the data center.

V. RESOURCE OPTIMIZATION ALGORITHMS

In-order to provide the best services to the cloud users, various resource optimization algorithms are used in cloud computing to optimize the resources. In this paper existing four algorithms were analyzed and compared with the proposed Tri-objective resource optimization algorithm. Genetic Algorithm (GA), Ant colony optimization Algorithm (ACO), Particle Swarm Optimization Algorithm (PSO) and Bacterial Foraging Optimization (BFO) Algorithm.

VI. POWER-AWARE RESOURCE MANAGEMENT

There is a large body of literature investigating energy management techniques for PaaS cloud service model that provides a platform for cloud customers to develop, run, and manage their applications without worrying about the underlying infrastructure and the required software. Both kinds of virtualization namely, OS level and System level virtualization, are considered and the newly introduced CaaS model can be viewed as a form of OS level virtualization service. Since CaaS cloud model has been newly introduced, we grouped all the research with the focus on containerized (OS-level virtualized) cloud environments under the PaaS category.

Servers are one of the most power-hungry elements in data centers, with CPU and memory as their main power consumers. The average power consumption of CPU and memory is reported to be 33% and 23% of the server's total power consumption respectively. Therefore, any improvement on processor and memory-level power consumption would definitely reduce the total power consumption of the server, which also improves the energy efficiency of data center. Dynamic voltage and frequency scaling (DVFS) are an effective system level technique utilized both for memory and CPU in bare metal environments and it is demonstrated to improve the power consumption of these two elements considerably. DVFS enables dynamic power management through varying the supply voltage or the operating frequencies of the processor and/or memory.

A. Dynamic Voltage and Frequency Scaling of CPU

The technologies present in the market are AMD Turbo Core, Intel Turbo Boost, and Intel Enhanced Speed Stepping Technology, which dynamically adjust the CPU frequency and voltage according to the workload. The harnessed the DVFS capability of CPU in the proposed scheduling algorithm. DVS scheduling scheme considers the deadline of the Bag-of-Tasks applications as a constraint and the CPU

frequency is adjusted so that the sub-tasks are finished by the deadline. An application made of a group of independent and identical tasks is an example of Bag-of-Task applications. DVS scheduling algorithms are provided for both time-shared and space-shared resource sharing policies. Proposed algorithm is validated through simulation and is shown to be more energy efficient when compared to the static voltage schemes.

B. Dynamic Voltage and Frequency Scaling of Memory

In addition to CPU, memory of servers also consumes a considerable amount of energy that is not proportional to the load. For memory-intensive workloads, system's memory speed is well tuned and optimized according to the peak computing power. However, there is still a place for improvement for other kinds of workloads that are less sensitive to the memory speed. For this kind of workload, running at lower memory speed would result in less performance degradation and reduce the power consumption via running memory at a lower frequency.

C. Coordinated CPU and Memory DVFS

The CoScale, which jointly applies DVFS on memory and CPU subsystems with the objective of minimizing the systems total power consumption. CoScale is the first work in this area that coordinates DVFS on CPU and memory considering performance constraints. The frequency of each core and the memory bus is selected in a way that energy saving of the whole system is maximized. Therefore, the selected frequencies are not always the lowest ones.

VII. PROPOSED METHODOLOGY

The heuristics for the problem of energy and performance efficient dynamic VM consolidation, which apply statistical analysis of the observed history of system behavior to infer potential future states. The proposed algorithms consolidate and deconsolidate VMs when needed to minimize energy consumption by computing resources under QoS constraints.

The proposed approach to dynamic VM merging consists in splitting the problem into 4 sub-problems:

- The first sub-problem is , determining if a host is considered to be underloaded, so that all VMs must be migrated from it, and the host must be switched to a low-power mode.
- Deciding if a host is considered to be overloaded, to avoid violating the QoS, some VMs should be migrated from it to other active or reactivated hosts requirements.
- Selecting the appropriate VMs to migrate from an overloaded host.
- Placing VMs that are selected for migration on the other active or reactivated hosts.

The splitting problem simplifies the analytic treatment of the sub-problems; and the approach can be implemented in a distributed manner by executing the underload / overload detection and VM placement algorithms on replicated controller hosts, and the VM selection algorithm on compute hosts. Distributed VM consolidation algorithms enable the natural scaling of the system when new compute hosts are added, which is essential for large-scale Cloud providers.

Figure 1 : Architecture Diagram

VIII. CLOUD MODEL

The target system is an IaaS environment, represented by a large-scale data center consisting of N heterogeneous physical nodes. Each node i is characterized by the CPU performance defined in Millions of Instructions Per Second (MIPS), amount of RAM and network bandwidth. The servers do not have direct-attached storage, while the storage is provided by a Network Attached Storage (NAS) or Storage Area Network (SAN) to enable VM live migration. The type of the environment implies no knowledge of application workloads and time for which VMs are provisioned. In other words, the resource management system must be application-agnostic.

Multiple independent users submit requests for provisioning of M heterogeneous VMs characterized by requirements to the processing power defined in MIPS, amount of RAM and network bandwidth. The fact that the VMs are owned and managed by independent users implies that the resulting workload created by consolidating multiple VMs on a single physical node is mixed. The mixed workload is formed by combining various types of applications, such as HPC and web-applications, which utilize the resources simultaneously. The users establish SLAs with the resource provider to formalize the QoS requirements. The provider pays a penalty in cases of SLA violations.

The approach to dynamic VM consolidation proposed in this chapter follows a distributed model, where the problem is divided into 4 sub-problems:

1. Host underload detection.
2. Host overload detection.
3. VM selection.
4. VM placement.

Splitting the problems improves the scalability of the system, as the host underload / overload detection and VM placement algorithms are executed locally by each compute host.

D. Power Model

Generally, the CPU, memory, disk storage, power supplies and cooling systems mostly determines Power consumption by computing nodes in data centers. The power consumed by servers can be accurately described by a linear relationship between CPU utilization and the power consumption, even though the Dynamic Voltage and Frequency Scaling (DVFS) is applied. The main reason lies in the number of states that are limited, that can be set to the frequency and voltage of a CPU and the fact that performance scaling and voltage is not applied to memory and network interfaces. However, due to the proliferation of multi-core CPUs and virtualization, modern servers are typically equipped with large amounts of memory, which begins to dominate the power consumption by a server. This fact combined with the difficulty of modeling power consumption by modern multi-core CPUs makes building precise analytical models a complex research problem.

E. Dynamic Merging

- Host Underload Detection

Although complex underload detection strategies can be applied, for the purpose of simulations in this chapter a simple approach is used. First, all the overloaded hosts are found using the selected overload detection algorithm, and the VMs selected for migration are allocated to the destination hosts. Then, the system finds a compute host with the minimal utilization compared with the other hosts, and attempts to place all the VMs from this host on other hosts, while keeping them not overloaded. If such a placement is feasible, the VMs are set for migration to the determined target hosts. Once the migrations are completed, the source host is switched to the sleep mode to save energy. The host is kept active, if all the VMs from the source host cannot be placed on other hosts, Host Overload Detection.

Each compute host periodically executes an overload detection algorithm to de-consolidate VMs when needed in order to avoid performance degradation and SLA violation. This section describes several heuristics proposed for the host overload detection problem.

- CPU Utilization Threshold

One of the simplest overload detection algorithms which is based on the method of setting a CPU utilization threshold differentiating the non-overload and overload states of the host. As soon as the algorithm is invoked, the algorithm compares the current CPU utilization of the host with the threshold that is defined. The algorithm detects a host overload, if the threshold is exceeded.

The dynamic VM merging considering multiple hosts and multiple VMs. For this problem, it is defined that there are n homogeneous hosts, and the capacity of each host is A_h. The maximum CPU capacity that can be allocated to a VM is A_v, although VMs experience variable workloads. Therefore, the maximum number of VMs allocated to a host when they demand their maximum CPU capacity is $m = \frac{A_h}{A_v}$. Let the total number of VMs be nm. Using live migration with a migration time t_m, the VMs can be migrated between hosts . As for the single VM migration, an SLA violation occurs when the total demand for the CPU performance exceeds the available CPU capacity A_h. $C_p$denotes the cost of the power, and C_v denotes the the cost of SLA violation per unit of time. Without loss of generality, the following relations can be defined: $C_p = 1 \; and \; C_v = s$, where $s \in R^+$. This is equivalent to defining $C_p = 1/s$ and $C_v = 1$. It is assumed that when a host is idle, i. e. there are no allocated VMs, it is switched off and consumes no power, or switched to the sleep mode with negligible power consumption. Non-idle hosts are referred to as active. The total cost C is defined as follows:

$$C = \sum_{t=t_0}^{T} \left(c_p \sum_{i=0}^{n} a_{ti} + c_v \sum_{j=0}^{n} v_{tj} \right) (1)$$

Where t_0 is the initial time; T is the total time; $a_{ti} \in \{0, 1\}$ indicating whether the host i is active at the time $v_{tj} \in \{0, 1\}$ indicating whether the host j is experiencing an SLA violation at the time t. The problem occurs in the determination of what time, which VMs and where it should be migrated in order to minimize the total cost C.

F. Online Deterministic Algorithm

The VM migration problem, an optimal online deterministic algorithm for the dynamic VM consolidation problem migrates a VM from a host when an SLA violation occurs at this host. The algorithm always consolidates VMs to the minimum number of hosts, ensuring that the allocation does not cause an SLA violation. The omnipotent malicious adversary generates the CPU demand by VMs in a way that cause as much SLA violation as possible, while keeping as many hosts active (consuming energy) as possible.

As $mA_v = A_h$, for any $k > m, k \in N, kA_v > A_h$. In other words, an SLA violation occurs at a host when at least m + 1 VMs are allocated to this host, and these VMs demand their maximum CPU capacity Av. Therefore, the maximum number of hosts that experience an SLA violation simultaneously n_v is defined as in

$$n_v = \lfloor \frac{nm}{m+1} \rfloor \text{ (2)}$$

In case a simultaneous SLA violation at n_v hosts occurs, the number of hosts not experiencing an SLA violation is given by$n_r = n - n_v$. The strategy of the adversary is to make the online algorithm keep all the hosts active all the time and make n_v hosts experience an SLA violation half of the time. To show how this is achieved, the time is split into periods of length $2t_m$. Then $T - t_0 = 2t_m\tau$, where $\tau \in R^+$. Each of these periods are or can be splitted into two equal parts of length t_m. The adversary acts as follows:

- During the first t_m, the adversary sets the CPU demand by the VMs in a way to allocate exactly m + 1 VMs to n_v hosts by migrating VMs from n_r hosts. As the VM migration time is t_m, the total cost during this period of time is$t_m nC_p$, as all the hosts are active during migrations, and there is no SLA violation.
- In the next t_m, the adversary sets the CPU demand by the VMs to the maximum giving rise to an SLA violation at n_v hosts. The online algorithm reacts to the SLA violation, and migrates the necessary number of VMs back to nr hosts. The total cost is $t_m(nC_p + n_v C_v)$ during This period of time, as all the hosts are again active, and n_v hosts are experiencing an SLA violation.

the total cost during a time period $2t_m$ is defined as follows:

$$C = 2t_m nC_p + t_m n_v c_v \text{ (3)}$$

This gives rise to the following total cost obtained by an optimal online deterministic algorithm
(ALG) for the input I:

$$ALG(I) = \tau t_m\big(2nC_p + n_v C_v\big) \text{ (4)}$$

An optimal offline algorithm for this kind of workload will just keep m VMs at each host all the time without any migrations. Thus, the total cost incurred by an optimal offline algorithm is defined

$$OPT(I) = 2\tau t_m nC_p \text{ (5)}$$

Having determined both costs, the competitive ratio of an optimal online deterministic
algorithm can be derived.

$$\frac{ALG(I)}{OPT(I)} = \frac{\tau t_m\big(2nC_p + n_v C_v\big)}{2\tau t_m nC_p} = \frac{2nC_p + n_v C_v}{2nC_p}$$
$$= 1 + \frac{n_v C_v}{2nC_p} \text{ (6)}$$

Via the substitution of $C_p = \frac{1}{s}$ and $C_v = 1$,

$$\frac{ALG_1(I)}{OPT(I)} = 1 + \frac{n_v s}{2n} \text{ (7)}$$

First, consider the case when mod $\frac{nm}{m+1} = 0$, and thus $n_v = \frac{nm}{m+1}$. For this case $ALG_1(I)$ the competitive ratio is shown

$$\frac{ALG_1(I)}{OPT(I)} = 1 + \frac{nms}{2n(m+1)} = 1 + \frac{ms}{2(m+1)} \text{ (8)}$$

The second case $ALG_2(I)$ is when $\frac{nm}{m+1} \neq 0$. Then, due to the remainder, n_v is less than in the first case. Therefore, the competitive ratio is defined as

$$\frac{ALG_2(I)}{OPT(I)} < 1 + \frac{ms}{2(m+1)} \text{ (9)}$$

If both cases are combined, the competitive ratio can be defined as which is an upper bound on the competitive ratio of an optimal online deterministic algorithm for the dynamic VM merging problem

$$\frac{ALG_2(I)}{OPT(I)} \leq 1 + \frac{ms}{2(m+1)} \text{ (10)}$$

IX. OPTIMAL MARKOV HOST OVERLOAD DETECTION

The Closed-form equations for $L_1(\infty), L_2(\infty), \ldots . L_N(\infty)$ are pre-computed offline

therefore, the run-time computation is not required. The values of transition probabilities are substituted into the equations for $L_1(\infty), L_2(\infty), \ldots . L_N(\infty)$, and the objective and constraint functions of the NLP problem are generated by the algorithm. To solve the NLP problem, a brute-force search algorithm with a step of 0. 1 is applied, as its performance was sufficient for the purposes of simulations. In a decision to migrate a VM is made only if either no feasible solution can be found, or the migration probability corresponding to the current state is 1. The justification for this is the fact that if a feasible solution exists and the migration probability is less than 1, then for the current conditions there is no hard requirement for VM to be migrated immediately.

Algorithm: The OPT algorithm

Input: Transition probabilities

Output: A decision on whether to migrate a VM or not.

1: Build the objective and constraint functions
2: Invoke the brute-force search to find the m vector
3: if a feasible solution exists then
4: Extract the VM migration probability
5: if the probability is < 1 then
6: return false
7: return true

G. The CPU model

The models and algorithms proposed in this chapter are suitable for both single core and multi-core CPU architectures. The capacity of a single core CPU is modeled in terms of its clock frequency F. A VM's CPU utilization ui is relative to the VM's CPU frequency fi and is transformed into a fraction of the host's CPU utilization U. These fractions are summed up over the N VMs allocated to the host to obtain the host's CPU utilization,

$$U = F \sum_{i}^{N} f_i u_i \quad (11)$$

For the purpose of the host overload detection problem, multi-core CPUs are modeled multi-core CPU with n cores each having a frequency f is modeled as a single core CPU with the n f frequency. The replaced by nf . The justification for the simplification is, as applications and VMs are not tied down to a specific core, but can by dynamically assigned to an arbitrary core by a time-shared scheduling algorithm. But the CPU capacity allocated to a VM cannot exceed the capacity of a single core is the only physical constraint. Removal of this constraint requires the VM to be executed on more than one core in parallel. However, automatic parallelization of Virtual Machines and their applications cannot be assumed.

X. IMPLEMENTATION AND RESULTS

Figure 2 : Resource Utilization

Figure 3 : Power Consumption

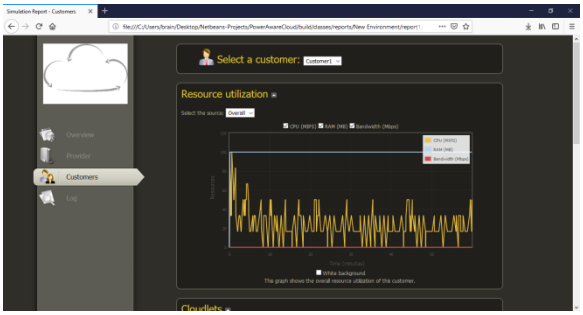

Figure 4 : Customer Resource Utilization

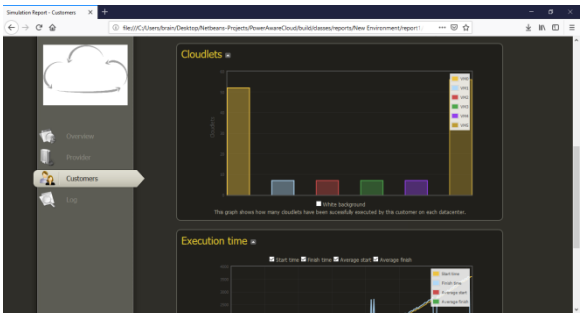

Figure 5 : Cloudlets running

Figure 6 : Execution Time

IX. CONCLUSION

The proposed and investigated a suite of novel techniques for implementing distributed dynamic VM consolidation in IaaS Clouds under workload-independent QoS constraints. The proposed approach improves the utilization of data center resources and reduces energy consumption, while satisfying the defined QoS requirements. The dynamic VM merging algorithms have been analyzed to obtain theoretical performance estimates and insights into designing online algorithms for dynamic VM consolidation. proposed an approach to distributed dynamic VM consolidation consisting in splitting the problem into 4 sub-problems: (1) host underload detection; (2) host overload detection; (3) VM selection; and (4) VM placement. Splitting the problem allows executing algorithms for the first 3 sub-problems in a distributed manner independently on each compute node. VM placement decisions still need to be made by a global manager; however, the load on the global manager is significantly reduced since it only makes placement decisions for the VMs selected for migration.

X. FUTURE ENHANCEMENT

The future work replication of the global managers would lead to multiple instances of the VM placement algorithm being executed concurrently on multiple controller nodes. It is important to develop advanced VM placement algorithms that would efficiently exchange information between the algorithm instances to reduce the impact of the distribution and provide the quality of VM placement close to centralized algorithms.

XI. REFERENCES

[1]. Ayoub Alsarhan,Awni Itradat,Ahmed Y,AI-Dubai,Senior Member,IEEE,Albert Y.Zomaya,IEEE, Fellow and Geyong Min, “Adaptive Resource Allocation and Provisioning in Multi-Service Cloud Environments”,IEEE transactions on parallel and distributed systems.

[2]. A. Alsarhan and A. Al-Khasawneh, “Resource trading in cloud environments for utility maximization using game theoretic modelling approach,” 2016

[3]. L. Wu et., al., “SLA-based resource provisioning for hosted software as a service applications in cloud computing environments,” 2013.

[4]. A. Alsarhan et., al., “Resource trading in cloud environments for profit maximisation using an auction model,” 2014.

[5]. A. S. Prasad and S. Rao., “A Mechanism Design Approach to Resource Procurement in Cloud Computing,” 2014.

Cite this article as :

Anitha Nithya R, Saran A , Vinoth R, "Adaptive Resource Allocation and Provisioning in Multi-Service Cloud Environments ", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 5 Issue 2, pp. 372-381, March-April 2019. Available at doi : https://doi.org/10.32628/CSEIT195253
Journal URL : http://ijsrcseit.com/CSEIT195253