Journal of Intelligent & Fuzzy Systems 30 (2016) 1581–1600
DOI:10.3233/IFS-151866
IOS Press

EARTH: Energy-aware autonomic resource scheduling in cloud computing

Sukhpal Singh* and Inderveer Chana
Department of Computer Science and Engineering, Thapar University, Patiala, Punjab, India

Abstract. In Cloud computing, data centers gain popularity as an effective platform for scheduling of resources and hosting cloud applications. However, tremendous amount of energy is consumed by these data centers which leads to high operational costs and contributes towards carbon footprints to the environment. Therefore, there is need of energy aware cloud based framework which schedules computing resources automatically by considering energy consumption as a QoS parameter itself. In this paper, we present fuzzy logic based energy-aware autonomic resource scheduling framework for cloud for energy efficient scheduling of cloud computing resources in data centers. We have evaluated the proposed framework in CloudSim based simulation environment and real cloud environment. The experimental results show that the proposed framework performs better in terms of resource utilization and energy consumption along with other QoS parameters.

Keywords: Autonomic cloud computing, energy, resource scheduling, fuzzy logic, self-optimization, green computing

1. Introduction

Cloud computing provides three types of services: Infrastructure, Platform and Software using pay per use model. Due to the cloud data centers, cloud provides effective and reliable infrastructure services to the end user [1]. Presently, customer satisfaction and performance is increased by deploying data centers without taking care of energy consumption in those datacenters. Great amount of energy consumption leads to high operational cost and reduces return on investment. Many governments have also imposed constraints to reduce the carbon footprints which effects environment. To solve this problem there is a need to focus on energy efficiency along with resource management in cloud. Larger IT companies (Microsoft, Google and IBM) are increasing their data centers every year to provide services to the cloud user in a better way. Due to large energy inefficiency, temperature increases gradually which leads to the failure of system and violates the Service Level Agreement (SLA). Literature reported that data center infrastructure generates over 70% of total heat generated [6].

Other reason of wastage of energy is resources are running in idle or underutilization state. Energy efficient resource scheduling in cloud is a challenging job and scheduling of appropriate resources to cloud workloads depends on the QoS requirements of cloud applications and energy consumption of computing resources. In cloud environment, heterogeneity, uncertainty and dispersion of resources encounters problems of energy consumption of resources, which cannot be addressed with existing resource allocation policies [3]. Energy saving and resource utilization in case of heterogeneous cloud workloads is very difficult to improve. Therefore, there is need of cloud based framework which schedules computing resources automatically by considering energy consumption as a QoS parameter. Autonomic resource scheduling is an ability to improve utilization of resources and user satisfaction in autonomic systems which are self-optimizing. Self-Optimizing is the capability to efficiently maximize resource allocation

*Corresponding author. Sukhpal Singh, Department of Computer Science and Engineering, Thapar University, Patiala, Punjab 147004, India. Tel.: +91 9855581444; E-mail: ssgill@thapar.edu.

and utilization for satisfying requirements of different users.

This research work focuses on one of the important aspects of self-optimization i.e. energy consumption. The motivation of this paper is to design a cloud based autonomic framework called EARTH (***E**nergy-aware **A**utonomic **R**esource management **T**ec**H**nique*) for effective scheduling of resources which considers energy consumption as an important QoS parameter and maintains Service Level Agreement (SLA). The main aim of this research work is: i) to propose an autonomic resource scheduling framework through fuzzy logic for execution of heterogeneous workloads, ii) to optimize the QoS parameters such as energy and resource utilization and iii) to implement and perform evaluation with existing work. Paper is structured as follows: Section 2 presents related work and contributions. Proposed framework with problem statement and objectives is presented in Section 3. Section 4 describes the experimental setup used for performance evaluation and results. Section 5 presents conclusions and future directions.

2. Related work

This section is presenting the related work of energy based resource scheduling and autonomic resource scheduling in brief. Scheduling of resources in cloud has been done through different techniques existing in literature but energy saving is an important factor that is difficult to optimize automatically [22]. In past, many techniques like parallel processing, multiprocessing and vector processing was developed to improve the resource utilization and energy consumption. After this hardware virtualization and server virtualization was developed [1]. Recent techniques like live migration of VMs and server virtualization is proposed to improve workload consolidation and reliability [2]. Cloud computing is now using the concept of virtualization to reduce energy consumption as reported from literature. Beloglazov et al. [4] presented energy based resource scheduling technique for management of data centers through virtualization by using virtual network topologies to reduce energy consumption but resource utilization is not considered. Anandharajan et al. [5] proposed energy based load balancing technique to reduce energy consumption through dynamic voltage scaling only for homogeneous workloads. Nguyen Quang-Hung et al. [6] presented energy based genetic algorithm for static VM allocation to reduce energy consumption. TsepoMofolo et al. [7] proposed Modified Best Fit Data (MBFD) algorithm based technique to improve resource utilization without achieving energy improvement.

Suraj Pandey et al. [8] presented a Particle Swarm Optimization (PSO) based heuristic to schedule the applications to cloud resources that proceeds both computation and data transmission cost. It is used for workflow applications by changing its computation and communication costs. Topcuoglu et al. [9] presented the HEFT (Heterogeneous Earliest Time First) algorithm to discover the average execution time of each workload and also the average communication time among the resources of two workloads. Then workloads in the workflow are well-ordered on a rank function. Then the workload with higher rank value is given higher priority. In the resource selection stage workloads are scheduled in priorities and each workload is allocated to the resource that complete the workload at the earliest time. Zhangjun Wu et al. [10] suggested a Market Oriented Hierarchical Scheduling (MOSH) approach which contains of both service level scheduling and workload level scheduling. The service level scheduling deals with the Task to Service assignment and the workload level scheduling deals with the optimization of the Task to Virtual Machine assignment in local cloud data centers. Jia Yu et al. [11] proposed a Cost Based Workflow Scheduling (CBWS) algorithm reduces the execution cost however meeting the deadline for delivering results. It can also adjust to the delays of service accomplishments by rescheduling unexecuted workloads. Varalakshmi et al. [19] described an OWS (Optimal Workflow based Scheduling) framework to discover a solution that tries to meet the user-desired QoS constraints i.e. execution time. This paper shows slight improvement in resource utilization is attained. But it does not consider cost and energy as QoS parameters. Wang et al. [20] presented an ACO (Ant Colony Optimization) based job scheduling framework, which adapts to dynamic characteristics of Cloud computing and incorporates particular benefits of ACO in NP-hard problems. This approach reduced only job completion time based on pheromone. Our Energy-aware Autonomic Resource Scheduling Framework (EARTH) has been compared with existing resource scheduling framework as described in Table 1.

All the above research works have presented energy, autonomic and resource scheduling in cloud computing without considering the energy consumption and resource utilization simultaneously and as a

Table 1
Comparison of energy-aware autonomic resource scheduling framework with existing frameworks

Framework	Mechanism	Workload Type	QoS Parameters
MBFD [7]	Non-Autonomic	Homogenous	Resource Utilization
PSO [8]	Non-Autonomic	Homogenous	Communication Cost
OWS [19]	Non-Autonomic	Homogenous	Execution Time
ACO [20]	Non-Autonomic	Homogenous	Completion Time
HEFT [9]	Non-Autonomic	Homogenous	Communication Time
MOSH [10]	Non-Autonomic	Homogenous	Execution Time
CBWS [11]	Non-Autonomic	Homogenous	Execution Cost
EARTH	Autonomic	Homogenous and Heterogeneous	Energy, Resource Utilization, Energy Efficiency, Computing Capacity and Latency

QoS parameter in autonomic system and concept of reallocation to reduce energy consumption.

None of the existing work considers heterogeneous cloud workloads. In addition; our novel fuzzy logic based energy aware autonomic resource scheduling framework needs to consider the basic features of cloud computing in order to execute the heterogeneous cloud workloads automatically with minimum energy consumption and maximum resource utilization along with other QoS parameters, which is not considered in other existing work.

2.1. Our contributions

We have presented fuzzy logic based energy aware resource scheduling framework for both homogenous and heterogeneous cloud workloads. This is an extension of our previous work [3]. The proposed framework focuses on how to map the cloud workload in order to improve resource utilization and energy consumption and other QoS parameters like latency and computing capacity. Finally, we have validated our proposed framework using CloudSim [14] and real cloud environment and measured the variations. This research work focuses on one important aspect of self-optimization i.e. energy consumption. The main contribution of this paper is: 1) Presents autonomic resource scheduling framework which considers both homogenous and heterogeneous Cloud workloads, and 2) Uses Fuzzy logic to make the decisions based on rules, 3) Optimizes the energy consumption, resource utilization and other QoS parameters. The proposed technique has been verified with the help of case study in Cloud simulated environment.

3. Resource scheduling framework

Scheduling of resources in cloud is an important part of resource management system. Mapping of cloud workloads to appropriate resources is mandatory to improve QoS parameters like energy consumption, resource utilization etc. Based on QoS requirements, scheduling finds and maps the resources and workloads. This section presents the resource scheduling framework as shown in Fig. 1.

Resource scheduling in previous work has been done in following steps [3]: i) understand the expectations and requirements of Cloud user, ii) analyzed and clustered the workloads through machine learning algorithm i.e. *k*-means based clustering algorithm, iii) finds the required number of resources, iv) maps the resources and workloads and iv) schedule and execute the workload on appropriate resources with minimum time and cost. In our previous work [3], we proposed four resource scheduling polices for optimization of cost and time without considering energy consumption, resource utilization as an important QoS parameter. This scheduling framework executes the workloads without self-optimization. But in present scenario, there is need of Cloud based framework which schedules computing resources automatically by considering energy consumption as a QoS parameter. In this research paper, we focused on these two parameters and automated the existing framework (Section 3.4). We presented fuzzy logic based energy-aware autonomic resource scheduling framework in Cloud for energy efficient scheduling of Cloud computing resources in data centers.

3.1. Objectives and commitments

The main objectives of this work are: i) To extend the existing framework and propose an energy aware autonomic resource scheduling framework using fuzzy logic, ii) To reduce energy consumption and improve resource utilization by considering energy consumption as a QoS parameter and iii) To verify the proposed framework.

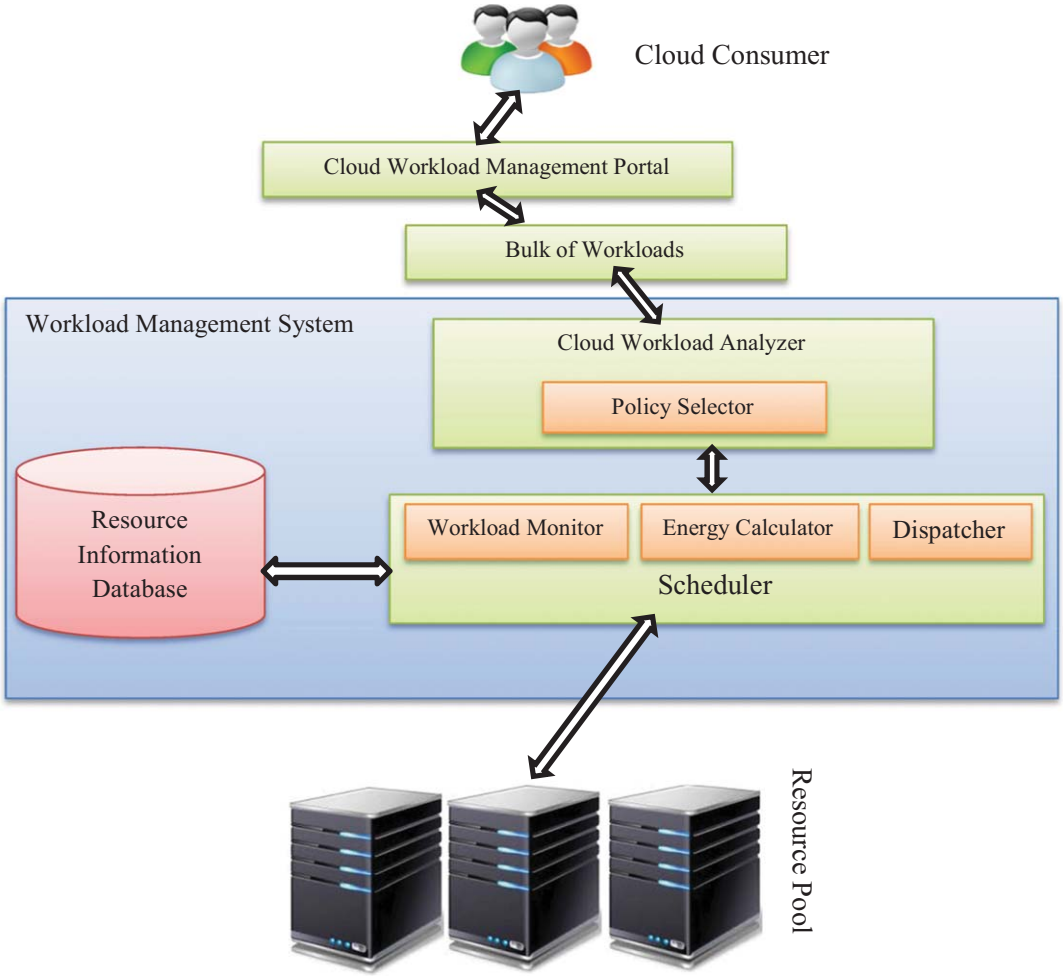

Fig. 1. Resource scheduling framework [3].

3.2. Problem statement

Cloud provider wants to improve resource utilization and energy consumption [12]. Thus smaller values of energy consumption would indicate that the scheduler is planning the cloud workloads in an efficient manner. Resource utilization is another optimization criterion, which refers to the resource usage of the cloud workload execution on a particular resource. The problem has been derived to get an optimal solution. The problem can be expressed as: To consider this problem, a set of independent cloud workloads $\{w_1, w_2, w_3, \ldots, w_m\}$ to map on a set of heterogeneous and dynamic resources $\{r_1, r_2, r_3, \ldots, r_n\}$ has been taken. $R = \{r_k 1 \leq k \leq n\}$ is the collection of resources and n is the total number of resources. $W = \{w_i | 1 \leq i \leq m\}$ is the collection of cloud workloads and m is the total number of cloud workloads [3].

3.3. Objective function

The goal of Cloud provider is to maximize the resource utilization and minimize the actual energy consumption. The Cloud workload will be executed only when the Actual Energy Consumption denoted as $Energy_{\min}$ is less than the threshold value of energy consumption (E_t). The energy model is devised on the basis that processor utilization has a linear relationship with energy consumption. For a particular Cloud workload, the information on its energy consumption and processor utilization is sufficient to measure the energy consumption for that Cloud workload [3]. The overall energy consumption of *EARTH*

(***E**nergy-aware **A**utonomic **R**esource management **T**ec**H**nique*) can be expressed as the following formula (Equation 1) [16]:

$$PCP = PCP_{Datacenter} + PCP_{Transceivers} + PCP_{Memory} + PCP_{Extra} \quad (1)$$

$PCP_{Datacenter}$ represents the datacenter's energy consumption, $PCP_{Transceivers}$ represents the energy consumption of all the switching equipment. PCP_{Memory} represents the energy consumption of the storage device. PCP_{Extra} represents the energy consumption of other parts, including the fans, the current conversion loss and others [12]. The above formula can be further disintegrated; a Cloud computing environment with d datacenters, t transceivers equipment and a centralized memory device, its energy consumption can be expressed as (Equation 2):

$$\begin{aligned} PCP = d(&PCP_{Processor} + PCP_{PrimaryStorage} \\ &+ PCP_{SecondaryStorage} + PCP_{Motherboards} \\ &+ PCP_{NetworkCards}) + \mathrm{t}\,(PCP_{Hardware} \\ &+ PCP_{LANcards} + \sum_{f=0}^{F} d_{connectors,f} + PCP_f) \\ &+ (P_{NetworkAnalysisServer} + P_{MemoryManager} \\ &+ P_{NetworkAttachedStorageArrays}) + P_{Extra} \end{aligned} \quad (2)$$

The energy consumed by a transceiver and all its ports can be defined as: where $PCP_{Hardware}$ is related to the power consumed by the transceiver, $PCP_{LANcards}$ is the power consumed by any active network LAN card, P_f corresponds to the power consumed by a connector (port) running at the frequency f. In above equation, only the last component appears to be dependent on the link frequency while other components, such as $PCP_{Hardware}$ and $PCP_{LANcards}$ remain fixed for all the duration of transceiver operation. Therefore, $PCP_{Hardware}$ and $PCP_{LANcards}$ can be avoided by turning the transceiver off or putting it into sleep mode. For a particular Cloud workload, the information on its energy consumption and processor utilization is sufficient to measure the energy consumption for that Cloud workload [3]. $P_{t',b}$ is the power consumption (Equation 3) of a cloud workload w_t is defined as:

$$P_{t',b}(r) = z \times PCP_{max} + (1 - z) \times PCP_{max} \times r \quad (3)$$

Where PCP_{max} is maximum power consumption while resource is fully utilized, z is fraction of power consumed by idle resource and r is CPU utilization. CPU utilization is change over time and it is function of time and presented as $r(t)$. For a resource r_t at given time period (t_1 to t_2), the server utilization U_t is defined as (Equation 4):

$$U_t = \int_{t_1}^{t_2} \sum_{b=1}^{c} P_{t',b}(r(t)) \quad dt \quad (4)$$

where c is the number of cloud workloads running at time t. The actual energy consumption $Energy_{min}$ of a resource r_t at given time t is defined as (Equation 5):

$$Energy_{min} = (PCP_{max} - PCP_{min}) \times U_t + PCP_{min} \quad (5)$$

where PC_{max} is the power consumption at the peak load (or 100% utilization) and PC_{min} is the minimum power consumption in the active/idle mode (or as low as 1% utilization).

3.4. Mode of operation: Energy-aware autonomic resource scheduling framework

In cloud computing, autonomic resource scheduling is a complex task [1]. It is very difficult to maintain SLA by fulfilling all the QoS requirements [2]. Based on these existing issues, we have designed an energy based framework for resource scheduling [12]. In this paper, we have extended the existing work [3] by considering two important QoS parameters through automation called EARTH (**E**nergy-aware **A**utonomic **R**esource management **T**ec**H**nique). The types of workload that have been identified during workload analysis, are Websites, Technological Computing, Endeavour Software, Performance Testing, Online Transaction Processing, E-Com, Central Financial Services, Storage and Backup Services, Production Applications, Software/Project Development and Testing, Graphics Oriented, Critical Internet Applications and Mobile Computing Services [13, 21]. Energy-aware autonomic resource scheduling framework is shown in Fig. 2.

This framework is based on IBM's autonomic model [18] that considers four steps of autonomic system: 1) Monitor, 2) Analyze, 3) Plan and 4) Execute. The assumptions of proposed framework are: a) Multi user accessing the cloud based system simultaneously, b) Workloads have different execution time and c) Workloads have different deadlines. The units of proposed framework are described below:

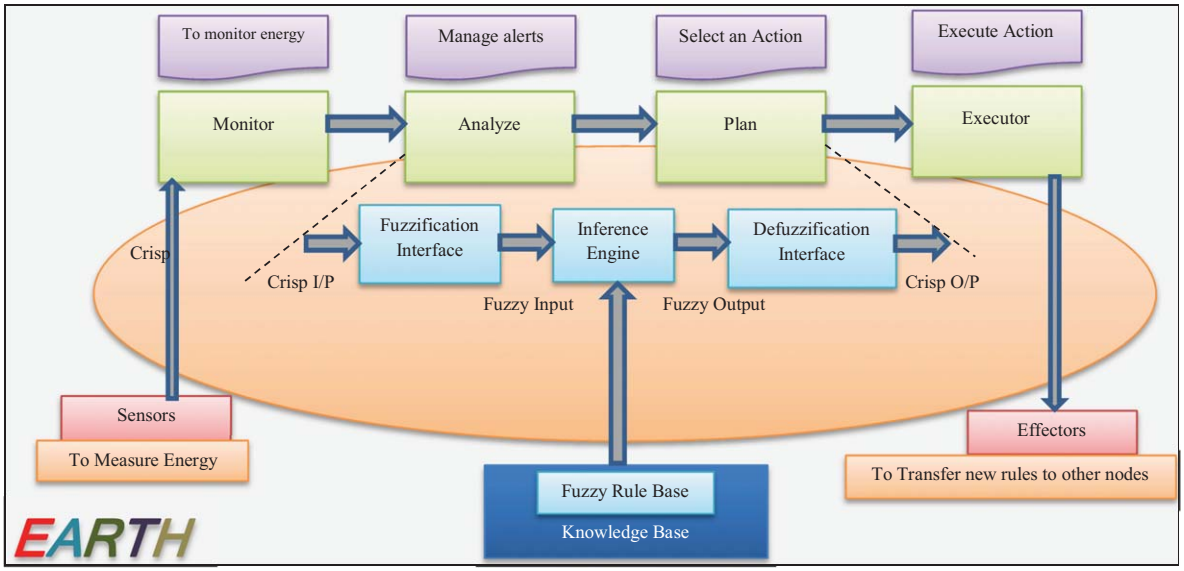

Fig. 2. Energy-aware Autonomic Resource Scheduling Framework.

Table 2
Actions and alerts

Alert	Action
If the node is not working after as required.	Restarting the failed node and use it, otherwise add new resources/node/VM according to [Algorithm 3]
If there are insufficient resources available to execute workload.	Add new resources/node/VM according to [Algorithm 3]
If the actual energy consumption is more than threshold value of energy consumption [Algorithm 4].	Use [Algorithm 3] to perform following actions: 1. Current node is declared as dead node. 2. Remove dead node. 3. Add new resource (s). 4. Reallocate resources.

3.4.1. Monitor [M]

Initially, Monitors are used to collect the information from sensors (Sensors get the information about energy consumption and resource utilization of all the systems working under Cloud and update the information time to time) for monitoring continuously the value of energy consumption and resource utilization and transfer this information to next module for further analysis.

3.4.2. Analysis and Plan [AP]

Analyze and Plan module start analyzing the information received from monitoring module and make a plan for adequate actions for corresponding alert as described in Table 2.

Once data has been analyzed then this framework executes the actions corresponding to the alerts automatically. We have used fuzzy logic [16, 17] to execute the workloads based on requirements (explained in next Section 3.4.4). The working of proposed framework is shown in Fig. 3. Bulk of workloads is an input for the framework as shown in Fig. 3.

Execution Time: We have used following formula to calculate Execution Time (Equation 6):

$$Execution\ \ Time_i = \sum_{i=1}^{n} \left(\frac{WC_i - WS_i}{n} \right) \qquad (6)$$

Where WC_i is workload completion time and WS_i is workload submission time and n is the number of workloads.

Deadline Urgency: We have used following formula to calculate Deadline Urgency [3] (Equation 7):

$$Deadline\ Urgency_i = \sum_{i=1}^{n} \left(\frac{Dt_i}{Et_i} - 1 \right) \qquad (7)$$

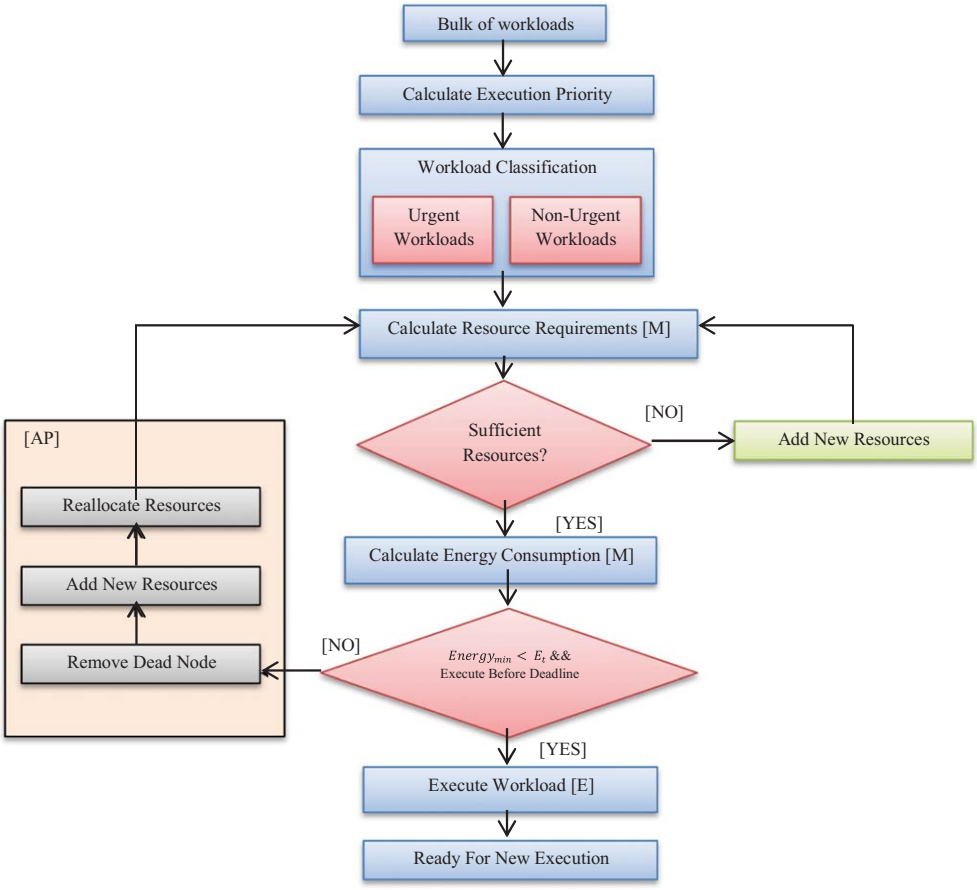

Fig. 3. Flowchart of Energy aware Autonomic Resource Scheduling Framework.

Algorithm 1: The algorithm to find the execution priority of arriving workload is as the following:

Loop
{
(there are workloads waiting for being executed)
1. For every arriving workload W_i, its $Execution\ Time_i$ and $Waiting\ Time_i$ is sent to inference engine. Priority of workload is output of inference module.
2. Execute the workload with highest execution priority.
3. System states update execution priority and deadline of remaining workloads.

Execute Workload
}
End Loop

Fig. 4. Algorithm to find the execution priority of arriving workload.

Algorithm 2: The algorithm to find the execution priority of arriving workload by considering deadline is as the following:

Loop
{
(there are workloads waiting for being executed)
1. For every arriving workload W_i, its $Execution\ Time_i$ and deadline is sent to inference engine. Priority of workload is output of inference module.
2. Execute the workload with highest execution priority based on deadline.
3. System states update execution priority and deadline of remaining workloads.

Execute Workload
}
End Loop

Fig. 5. Algorithm to find the execution priority of arriving workload by considering deadline.

Algorithm 3: The algorithm to add VM/Resource with minimum energy consumption is as the following:

Loop
{
if ($Energy_{min} \geq E_t$) [VM is consuming energy more than threshold value of energy] **then**
(declared VM as dead node and removed)
}
End Loop
Loop
{
if (VM is required to execute the workloads without degradation in resource utilization) **then**
(add new VM node from resource pool with minimum energy consumption and ($Energy_{min} \leq E_t$)
}
End Loop
Loop
{
if (VM is required to execute the workloads without degradation in resource utilization but not available in resource pool) **then**
(add new VM node from reserve resource pool with minimum energy consumption and ($Energy_{min} \leq E_t$)
}
End Loop

Fig. 6. An algorithm to add VM/Resource with minimum energy consumption.

Algorithm 4: The algorithm to allocate the VMs to workloads with energy consumption less than threshold value of energy consumption is as the following:

1. Sorting of VM based on Resource Utilization in decreasing order
2. **for** every VM find
3. $power_{min} \leftarrow Energy_{min}$
4. allocatedhost ← VMlist[]
5. **if (** $power_{min} \leq E_t$) **then**
6. allocatehost ← host
7. $power_{min} \leftarrow Energy_{min}$
8. **Elseif** allocatehost= NULL **then**
9. Generate Alert
10. Call [Algorithm 3]
11. **Goto step 2**
12. **end if**
13. **end for**

Fig. 7. Algorithm to compare the energy consumption.

Where Dt_i is deadline time and Et_i is execution time.

Deadline Time: We have used following formula to calculate Deadline Time [3] (Equation 8):

$$Deadline\ Time_i = \sum_{i=1}^{n} (Wd_i - Ct_i) \qquad (8)$$

Where Wd_i is deadline of workload and Ct_i is current time.

Waiting Time: We have used following formula to calculate Waiting Time (Equation 9):

$$Waiting\ Time_i = \sum_{i-1}^{n} \left(\frac{WE_i - WS_i}{n} \right) \qquad (9)$$

Where WE_i is workload execution start time and WS_i is workload submission time and n is the number of workloads.

Further, [Algorithm 4] is used to compare the actual energy consumption ($Energy_{\min}$) with threshold value of energy consumption (E_t). If energy consumption $Energy_{\min}$ is less than threshold value of energy then execution of workloads continues otherwise it will generate alerts for analysis. [Algorithm 4] is used in Energy-aware autonomic resource scheduling and is shown in Fig. 7. E_t is threshold value of energy consumption and $Energy_{\min}$ is actual energy consumption. Input stage consists of three linguistic variables: Workload Waiting Time (WWT), Workload Execution Time (WET) and Resource Energy Consumption (REC). Output is to execute the workload first with highest priority [Workload Processing Priority (WPP)] with minimum energy consumption and maximum resource utilization. We have classified workloads into two categories based on Workload Processing Priority: urgent workloads and non-urgent workloads. We used [Algorithm 1] and [Algorithm 2] to calculate the Workload Processing Priority (WPP). The algorithm to find the execution priority of arriving workload is shown in

Fig. 4. Output of [Algorithm 1] and workload deadline (Wd_i) is used as input for [Algorithm 2] to calculate the final priority of workload. The algorithm used to find the execution priority of arriving workload based on deadline is shown in Fig. 5. Workload with highest priority is put into the categories of urgent workloads and remaining will be considered as non-urgent workloads. Fuzzy rules are used to schedule the workloads according to their priorities.

Framework automatically checks the total workloads in the workload queue after each new workload is added. Priorities of workloads are changing adaptively. The reason for changing priorities might be priority of new added workload is higher. For this workload deadline is mandatory to consider. Otherwise, new workload with higher priority waits for long time which leads to starvation and reduce user satisfaction. Therefore, we used [Algorithm 1] and [Algorithm 2] for this purpose. After finding the Workload Processing Priority (WPP), framework calculates the resource requirements. Whether resources are sufficient for execution of workload (s) are provided or not. If the sufficient resources are provided then start execution of workload otherwise add new resources by using [Algorithm 3] as shown in Fig. 6. This algorithm can also be used to remove the dead node. In first step, sorting of VM based on resource utilization is performed in decreasing order in [Algorithm 4]. With the help of (Equations 1–5), energy consumption is calculated and allocates the host from VM list and then compares the energy consumption with threshold value of energy. If energy consumption $Energy_{\min}$ is less than threshold value of energy and workload is executing before deadline then execution of workloads continues otherwise no host is allocated and process of reallocation is started using [Algorithm 3].

After the minimum energy consumption, VMs again allocated for further execution.

3.4.3. Executer [E]

Executor implements the plan after analyzing completely. To reduce the latency and energy consumption and improve resource utilization and energy efficiency is a main objective of executor. Based on the output (crisp output) given by analysis and executor tracks the new workload submission and resource addition, and take the action according to rules described in Section 3.4.2. Effector is used to transfer the new policies, rules and alerts to other nodes with updated information. Next section describes the fuzzy rule based framework execution.

3.4.4. Fuzzy rule based autonomic framework

Energy-aware Autonomic Resource Scheduling Framework always executes the workloads with highest priority (which has earliest deadline) as shown in Fig. 2. The components of fuzzy logic system [16, 17] used in this framework is described below:

a) *Fuzzy input and output variables:* It is necessary to find the input and output parameters for fuzzy system. Basic information includes workload name, workload type, resource name and resource type, but these parameters are constant, does not effect on final output. Fuzzy inputs include Workload waiting Time (WWT), Workload Execution Time (WET) and Resource Energy Consumption (REC) and fuzzy output is Workload Processing Priority (WPP). All the three parameters are categorized into three levels: Low (L), Medium (M) and High (H). All the four variables are changing continuously due to this, and hence these variables are considered. Fuzzy outputs variable workload processing priority. Further, all the input and output variables are classified into various groups. Every group represents the fuzzy set on given input or output. Based on this, fuzzy rule set is created to define the behavior of fuzzy system and setting the relationship among inputs and outputs. Maximum waiting time fixed in this framework is 50 seconds; otherwise it indicates shortage of resources. Fuzzy input and output variables are shown in Table 3.

b) *Fuzzy membership functions (Inferences):* In this framework, three membership functions are considered for every input and out variables: Low, Medium and High. Based on these input and output variable, inference engine is making decisions. Block diagram of inference engine is shown in Fig. 8.

Table 3
Fuzzy input and output variables

Workload waiting Time (WWT)	Workload Execution Time (WET)	Resource Energy Consumption (REC)	Workload Processing Priority (WPP)
L	L	L	L
M	M	M	M
H	H	H	H

Fuzzy Inputs: Workload waiting Time (WWT): (from 1 to 50 seconds) Workload Execution Time (WET): (from 1 to 10 seconds) Resource Energy Consumption (REC): (from 1 to 10 kwh) *Fuzzy Outputs:* Workload Processing Priority (WPP): (from 1 to 100)

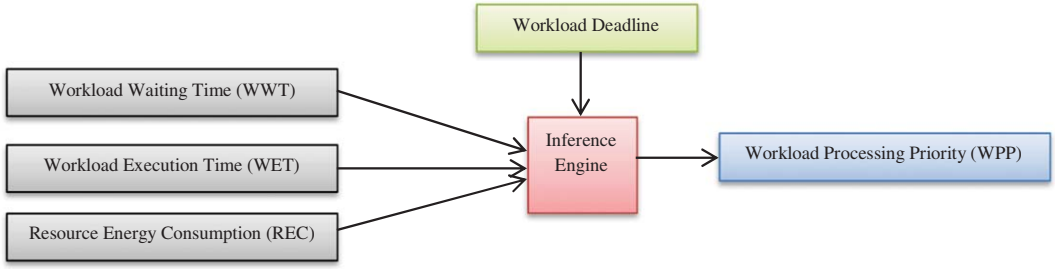

Fig. 8. Block diagram of inference engine.

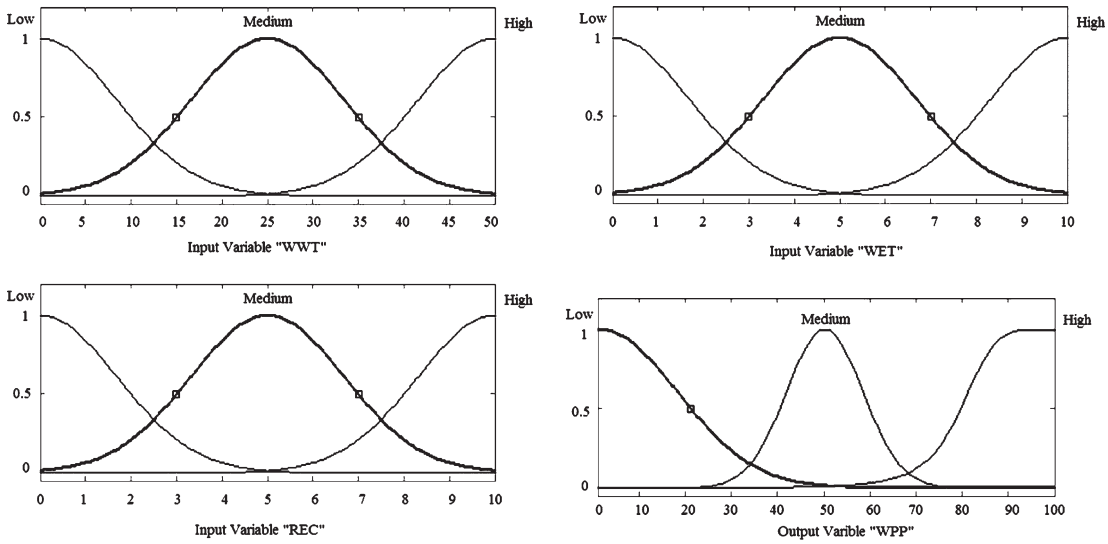

Fig. 9. Fuzzy subsets of fuzzy inputs and fuzzy output.

Value of membership functions can be changed based on the requirements and conditions of every workload. After the inputs and outputs of a fuzzy system are selected, they must be partitioned into appropriate conceptual categories. Based on selected inputs and outputs of the fuzzy system, member functions are created for better representation of relationship among input and output variables. Each of these categories actually represents a fuzzy set on a given input or output domain. The conceptual partitions developed for the input and output dimensions are used to create a fuzzy rule set which determines the behavior of the fuzzy system being constructed. This fuzzy rule set is called the fuzzy algorithm for the system being developed. The fuzzy rule set codifies the relationships that exist among the various partitions of the inputs and outputs dimensions in the form of member functions. Fuzzy subsets of fuzzy inputs and fuzzy output are shown in Fig. 9.

From Fig. 9, it can be observed that the y-axis is the degree of the membership of each of the fuzzy variables. For the input fuzzy variables the universe of discourse (the x-axis) is the quantized sensed values for the Workload Waiting Time (WWT), Workload Execution Time (WET) and Resource Energy Consumption (REC), respectively. For the output fuzzy variable the universe of discourse is Workload Processing Priority (WPP). However, the width and center of the membership functions of these fuzzy subsets can be easily changed and configured according to exterior factors and conditions of workload and cloud environment.

c) *Fuzzy rule base:*In the fuzzy logic system, the working of inference engine is similar to reasoning process of human. Workload Waiting Time (WWT), Workload Execution Time (WET) and Resource Energy Consumption (REC) are antecedents and Workload

Processing Priority (WPP) is consequent. Three membership functions consider for every three inputs. The number of rules for this system is 81 based on 3 inputs (low, medium and high) for every four variables (WWT, WET, REC and WPP) [Total number of possible rules = $(\text{Number\_of\_inputs})^{\text{number\_of\_variables}} = 3^4 = 81$ rules] as shown in Table 3). Several rules are using simultaneously in inference process. Following are the examples of rules:

If (WWT is High) and (WET is High) and (REC is High) then (WPP is Low)

If (*WWT is Low*) *and* (*WET is Low*) *and* (*REC is Low*) *then* (*WPP is High*)

d) *Fuzzification:* To find the degree of truth for every rule, membership function is defined on every input variable is applied to their actual value. We used most popular operator "AND" operator for fuzzy implementation. This function returns the lowest value of among these values entered.

AND (*WWT, WET, REC*) = *MINIMUM* (*truth* (*WWT*), *truth* (*WET*), *truth* (*REC*))

We used fuzzy "AND" operator to evaluate the rules and produce another variable. Rule is said to be fire if value is non-zero. For every rule, resultant value is used to represent the degree of truth. Apply the truth value of every rule to the output value (WPP) is called inference process. We used *MINIMUM* as an inference rules to calculate the degree of truth through the use of fuzzy logic "AND". In MINIMUM inferencing (used in this research), the output membership function is clipped off at a height corresponding to the rule premise's computed degree of truth (fuzzy logic AND).Composition process produces the fuzzy subset, which can be further used to convert to single crisp output through Defuzzification.

e) *Defuzzification:* It is used to convert the value of fuzzy output into crisp output value. We used MAXIMUM method in this work for Defuzzification. In the MAXIMUM method, one of the variable values at which the fuzzy subset has its maximum truth value is chosen as the crisp value for the output variable. We select the crisp output for output variable at which fuzzy subset has maximum.

4. Experimental setup and results

Tools used for setting Cloud environment are Microsoft Visual Studio, NetBeans IDE 7.1.2, CloudSim, IntegratedNETJavaWeb and SQL Server. Microsoft Visual Studio 2010 is an Integrated Development Environment from Microsoft. The integration of multiple environments used to conduct experiments is shown in Fig. 10.

Cloud user interacts with Cloud Workload Management Framework through Cloud Workload Management Portal (CWMP) to submit the workload details. Table 4 shows the characteristics of resources and cloudlets (workloads) that have been used for all the simulations.

User information, workload detail and resource detail are stored in database through SQL Server. Energy-aware Autonomic Resource Scheduling Framework (EARTH) is also tested on Real Cloud Environment [THAPAR CLOUD] has been established at Thapar University, India. For real

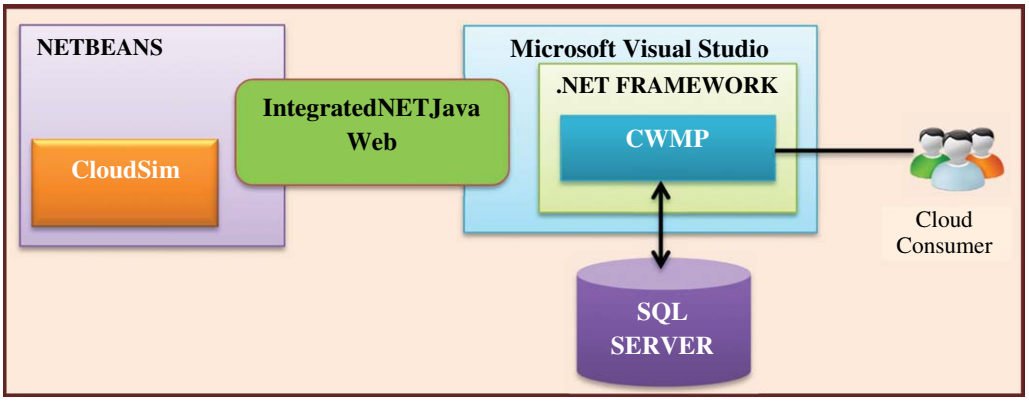

Fig. 10. Cloud Based Testbed [3].

Table 4
Scheduling parameters and their values

Parameter	Value
Number of resources	50–250
Number of Cloudlets (Workloads)	3000
Bandwidth	1000–3000 B/S
Size of Cloud Workload	10000+ (10%–30%) MB
Number of PEs per machine	1
PE ratings	100–4000 MIPS
Cost per Cloud Workload	$3–$5
Memory Size	2048–12576 MB
File size	300 + (15%–40%) MB
Cloud Workload output size	300 + (15%–50%) MB

experiment purposes, we categorized the nodes into further four categories as shown in Table 5. For Energy-aware Autonomic Resource Scheduling Framework (EARTH), we have measured expected performance (Autonomic QoS based RS (E)) in simulated environment and actual performance (Autonomic QoS based RS (A)) in real Cloud Environment. Cloudworkload management portal is implemented in. NET framework and framework is running in Microsoft Visual Studio. NetBeans is used to execute the CloudSim toolkit in which EARTH has been implemented. The workload details gathered from various users are transferred in a specific format from. NET framework to NetBeans through the use of IntegratedNETJavaWeb. We have explained the description of simulation environment in our previous work [3].

4.1. Experimental results

3000 independent Cloud workloads were generated randomly in CloudSim as Cloudlets. All configurations about the resources will remain the same during the experiment. In order to evaluate the effectiveness of the scheduling framework, the simulator namely CloudSim [14] has been used to calculate the energy consumption for each of them. The characteristics of resources and Cloudlets that have been used for all the experiment is described in detail in our previous work [3]. User Cloud workloads are modeled as independent parallel applications are modeled which is computation intensive. Thus the data dependency among the Cloud workloads in the parallel applications is negligible. Each Cloud workload is parallel and is hence considered to be independent of any other Cloud workload. Fuzzy logic based energy aware autonomic resource scheduling framework has been tested experimentally using three different types of test cases and case study along with statistical analysis. *Test Case 1* presents the comparison of average value of energy consumption and average value of resource utilization for both fuzzy logic based energy aware autonomic resource scheduling (RS) (QoS aware) and non-QoS aware resource scheduling technique with same number of workloads. *Test Case 2* presents the experimental results with different number of workloads (500–3000) for resource utilization and energy consumption. In this test case, both expected performance (Autonomic QoS based RS (E)) and actual performance (Autonomic QoS based RS (A)) for fuzzy logic based energy aware autonomic resource scheduling (RS) framework has been measured and compared with non-QoS aware resource scheduling technique. Computing nodes described in Table 5 has been used to measure actual performance by installing different nodes on differ machines while expected performance is performed by using cloud based simulation environment as shown in Table 4. *Test Case 3* presents the experimental resultsdifferent number of clusters (*a*) Compute (C1), *b*) Storage (C2), *c*) Communication (C3) and *d*) Administration (C4)) with different deadline ($0 \leq D_u \leq 1$). Resource utilization and energy consumption for different type of clusters has been tested experimentally because simulation result will be very different for different workloads (different deadline). *Test Case 3* helps to find the effective cluster which provides best results for resource utilization and energy consumption with different number of workloads and different value of their deadline. *Test Case 4* presents convergence curve of integrated algorithm to verify the stability of fuzzy logic based energy-aware autonomic resource scheduling framework with different value of resource utilizations. To validate fuzzy logic based energy-aware autonomic resource scheduling

Table 5
Configuration details of THAPAR CLOUD

Configuration	Specifications	Operating	Node
Intel Core 2 Duo - 2.4 GHz	1 GB RAM and 160 GB HDD	Window	52
Intel Core i5-2310- 2.9GHz	1 GB RAM and 160 GB HDD	Linux	14
Intel XEON E 52407-2.2 GHz	2 GB RAM and 320 GB HDD	Linux	2
Intel XEON E 5620- 2.4 GHz	2 GB RAM and 320 GB HDD	Window	1

framework, performance evaluation through virtualization by considering five important QoS parameters (resource utilization, energy consumption, energy efficiency, computing capacity and latency) in the form of case study of "Banking Datacenters" has been presented to study the effect of over and underutilization of resource at run time. Different experiments has been conducted with the variations in the number of workloads submitted through virtualization by using different nodes as described in Table 5. Statistical significance of the results has been analyzed by coefficient of variation, a statistical method to statistical measure of the distribution of data about the mean value to find the stability of fuzzy logic based energy-aware autonomic resource scheduling framework with small value of coefficient of variation.

Test Case 1: Same number of Workloads

Figure 11, shows that the energy consumption of same number of workloads is lesser with fuzzy logic based energy aware autonomic resource scheduling (RS) (QoS aware) as compared to non-QoS aware resource scheduling technique. Average energy consumption is decreased by 15.15% in QoS aware as compared to non-QoS. Figure 12, shows the average resource utilization also increases 10.2% with fuzzy logic based energy aware autonomic resource scheduling framework (QoS aware) as compared to non-QoS based resource scheduling.

Test Case 2: Different number of Workloads

Figure 13, shows that the energy consumption of different number of workloads (500–3000) is lesser with fuzzy logic based energy aware autonomic resource scheduling (RS) framework (QoS aware) but increases without QoS aware resource scheduling.

For fuzzy logic based energy aware autonomic resource scheduling (RS) framework, we have measured expected performance (Autonomic QoS based RS (E)) and actual performance (Autonomic QoS based RS (A)).

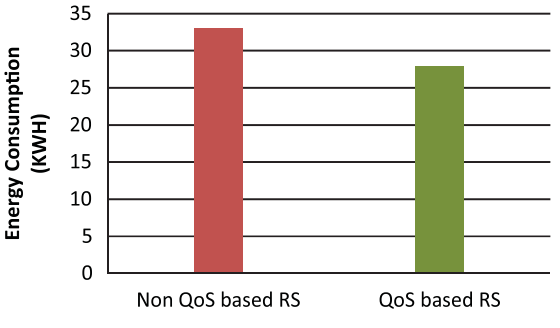

Fig. 11. Comparison of energy consumption with same number of workloads.

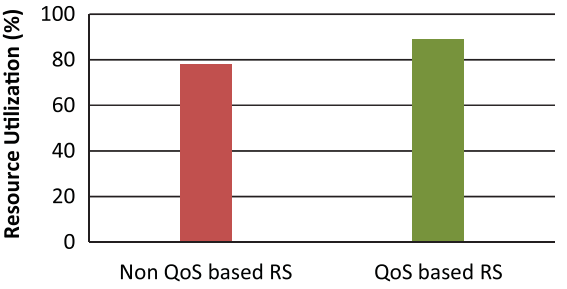

Fig. 12. Comparison of resource utilization with same number of workloads.

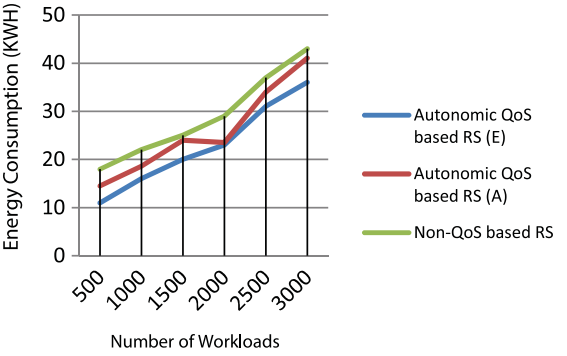

Fig. 13. Comparison of energy consumption with different number of workloads.

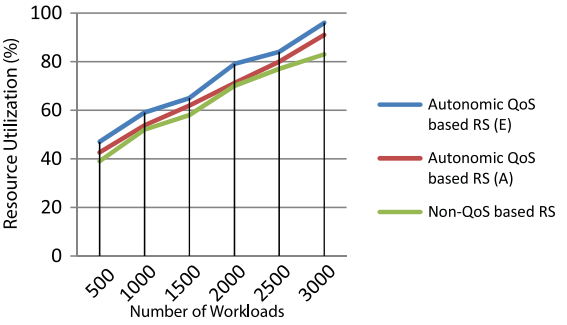

Fig. 14. Comparison of resource utilization with different number of workloads.

In our experiments, actual performance is performed by installing different nodes on differ machines as described in Table 5 while expected performance is performed by using cloud based simulation environment. Energy consumed in Autonomic QoS based RS (A) is more than Autonomic QoS based RS (E) but lesser than Non-QoS based RS. Figure 14, shows the resource utilization increaseswith fuzzy logic based energy aware autonomic resource scheduling framework (QoS aware) as compared to

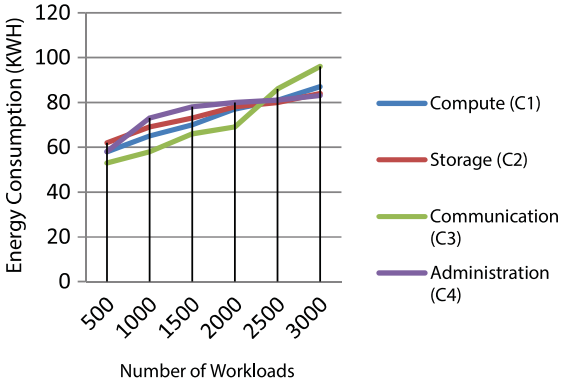

Fig. 15. Comparison of energy consumption with different number of workloads at Deadline ($D_u > 0.75$).

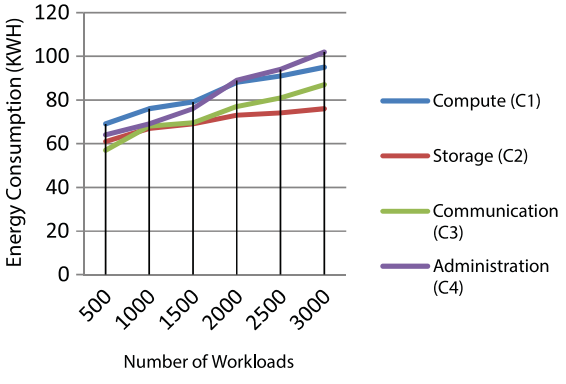

Fig. 16. Comparison of energy consumption with different number of workloads at Deadline ($0.25 \leq D_u \leq 0.75$).

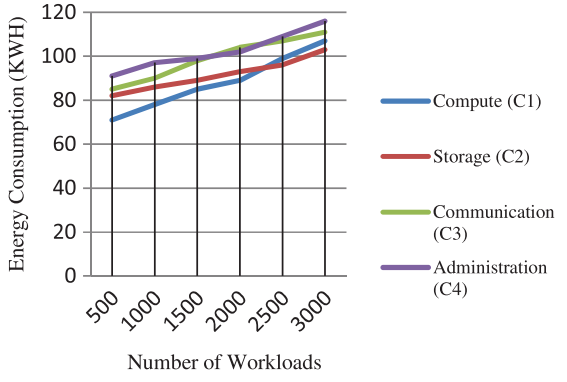

Fig. 17. Comparison of energy consumption with different number of workloads at Deadline ($D_u < 0.25$).

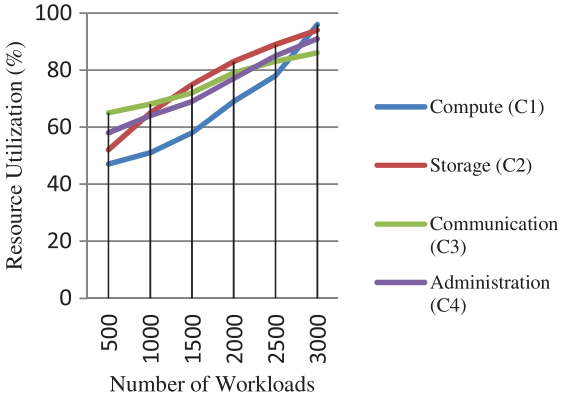

Fig. 18. Comparison of Resource Utilization with different number of workloads at Deadline ($D_u > 0.75$).

non-QoS based resource scheduling but resource utilization in Autonomic QoS based RS (A) is lesser than Autonomic QoS based RS (E).

Test Case 3: Different number of Clusters with different Deadline

We have clustered the workloads through K-Means based clustering algorithm in our previous work [3] namely: *a*) Compute (C1), *b*) Storage (C2), *c*) Communication (C3) and *d*) Administration (C4). We also considered the Deadline Urgency (D_u) and grouped in three categories [3]: i) $D_u > 0.75$, ii) $0.25 \leq D_u \leq 0.75$ and iii) $D_u < 0.25$. Experiment has been performed to know the variation in energy consumption and resource utilization with different number of clusters with different deadline. Figure 15 shows the energy consumption of different number of workloads (500–3000) with fuzzy logic based energy aware autonomic resource scheduling (RS) framework (QoS aware) for different number of clusters with deadline ($D_u > 0.75$). Energy consumption of different number of workloads (500–3000) with fuzzy logic based energy aware autonomic resource scheduling (RS) framework (QoS aware) for different number of clusters with deadline ($0.25 \leq D_u \leq 0.75$), as shown in Fig. 16. Figure 17 shows the energy consumption of different number of workloads (500–3000) with fuzzy logic based energy aware autonomic resource scheduling (RS) framework (QoS aware) for different number of clusters with deadline ($D_u < 0.25$).

Figure 18 shows the resource utilization with different number of workloads (500–3000) with fuzzy logic based energy aware autonomic resource scheduling (RS) framework (QoS aware) for different number of clusters with deadline ($D_u > 0.75$). Resource utilization with different number of workloads (500–3000) with fuzzy logic based energy aware autonomic resource scheduling (RS) framework (QoS aware) for different number of clusters with deadline ($0.25 \leq D_u \leq 0.75$), as shown in

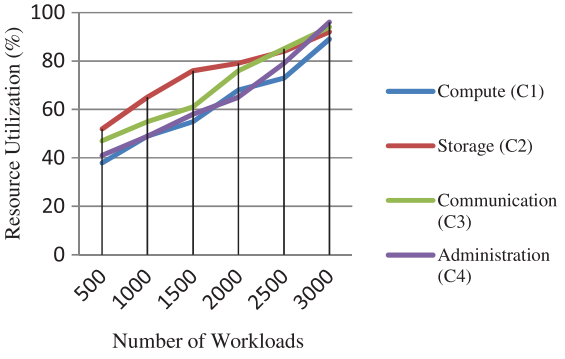

Fig. 19. Comparison of Resource Utilization with different number of workloads at Deadline ($0.25 \leq D_u \leq 0.75$).

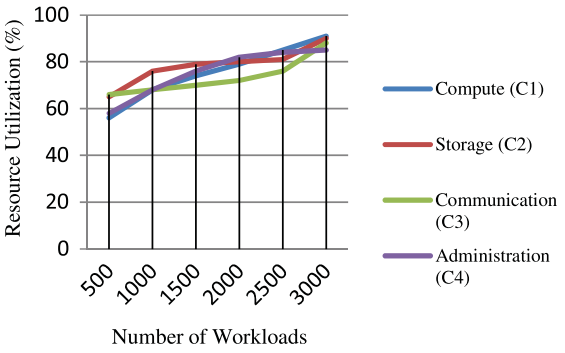

Fig. 20. Comparison of Resource Utilization with different number of workloads at Deadline ($D_u < 0.25$).

Fig. 19. Figure 20 shows the resource utilization with different number of workloads (500–3000) with fuzzy logic based energy aware autonomic resource scheduling (RS) framework (QoS aware) for different number of clusters with deadline ($D_u < 0.25$).

Test Case 4: Convergence of Integrated Algorithm

Figure 21 plots the convergence of total energy consumption computed by integrated algorithm over the number of iterations for different value of Resource Utilization (RU): 55%, 65%, 75% and 85% by executing different number of workloads. Initially the workloads are randomly initialized. Therefore, the total initial energy consumption is very high at 0th iteration. As the algorithm progresses, the convergence is drastic and achieves global minima very quickly. The number of iterations required for the convergence is seen to be 60–70, for our cloud environment.

4.2. Case study: Banking datacenters

To validate our novel fuzzy logic based energy-aware autonomic resource scheduling framework, we have presented performance evaluation through virtualization by considering other QoS parameters in the form of case study of "Banking Datacenters". We have studied case study of "Green Investment Bank: Citibank UK data centre cut its energy usage in first of a kind project" to analyze the energy consumption in banking datacenters [15]. It is very difficult to reduce the energy consumption in datacenters used in banking systems because all the servers are working 24×7 to provide the reliable service to customers. The working of this banking system is studied from centralized website of "http://www.ctrls.in". In the existing banking system, scheduling of resources is done without considering the concept of autonomic and other QoS (energy etc.) parameters, which leads to overutilization of resources and more energy con-

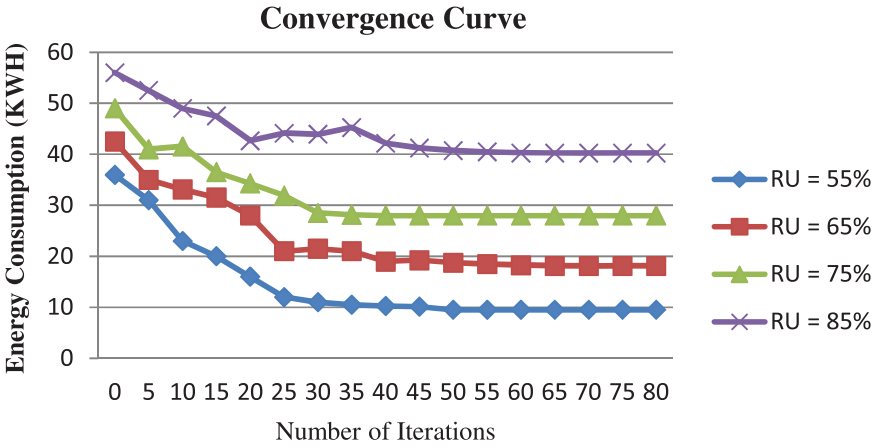

Fig. 21. Convergence curve of total energy consumption.

sumption. To solve this problem, we have considered five QoS parameters [21] (*Resource Utilization*is used to monitor the underutilization and over utilization of resources, *Energy Consumption* is used to calculate the consumption of energy, *Energy Efficiency* is used to calculate the energy consumed to successfully execute the workloads, *Computing Capacity* is used to calculate the usage time of resources and *Latency* is used to calculate the time difference in execution of workload) in this case study to validate our fuzzy logic based energy-aware autonomic resource scheduling framework. Performance of all the five QoS parameters has been evaluated and as compared with non-autonomic based resource scheduling. For verifying proposed framework, we have conducted experiments with the variations in the number of workloads submitted through virtualization by using different nodes as described in Table 5. This case study reflects the scheduling of resources similar to Cloud environment.

Test Case 1: Resource Utilization

We have calculated percentage of resource utilization for both fuzzy logic based energy-aware autonomic resource scheduling framework and non-autonomic based resource scheduling with different number of Cloud workloads. *Resource utilization* is a ratio of actual time spent by resource to execute workload to total uptime of resource for single resource. We have used following formula to calculate resource utilization (Equation 10):

$$\textit{Resource Utilization}_i = \sum_{i=1}^{n} \left(\frac{\textit{actual time spent by resource to execute workload}}{\textit{total uptime of resource}} \right) \tag{10}$$

With increasing the number of Cloud workloads, the percentage of resource utilization is increasing. The percentage of resource utilization in fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) is more as compared to non-autonomic based resource scheduling (non-autonomic) at different number of Cloud workloads as shown in Fig. 22. The maximum percentage of resource utilization is 94.1% at 3000 cloud workloads.

Test Case 2: Energy Consumption

We have calculated value of energy consumption in Kilo Watt Hour (kwh) for both fuzzy based energy-aware autonomic resource scheduling framework and non-autonomic based resource scheduling with different number of Cloud workloads using formulas discussed in Section 3.3 (Equations 1–5). With increasing the number of Cloud workloads, the value of energy consumption is increasing. The minimum value of energy consumption is 16 kwh at 500 cloud workloads. Fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) performs better than non-autonomic based resource scheduling (non-autonomic) in terms of energy consumption at different number of Cloud workloads as shown in Fig. 23.

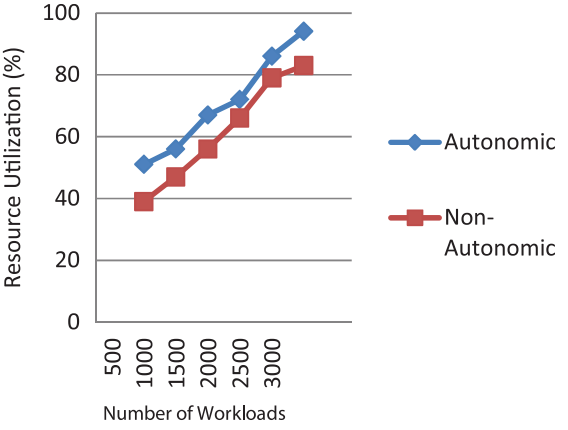

Fig. 22. Influence of change in number of workloads submitted on Resource Utilization.

Test Case 3: Energy Efficiency

We have calculated value of energy efficiency for both fuzzy logic based energy-aware autonomic resource scheduling framework and non-autonomic based resource scheduling with different number of Cloud workloads. Energy efficiency is a ratio of number of workloads successfully executed in a data center to total energy consumed to execute those

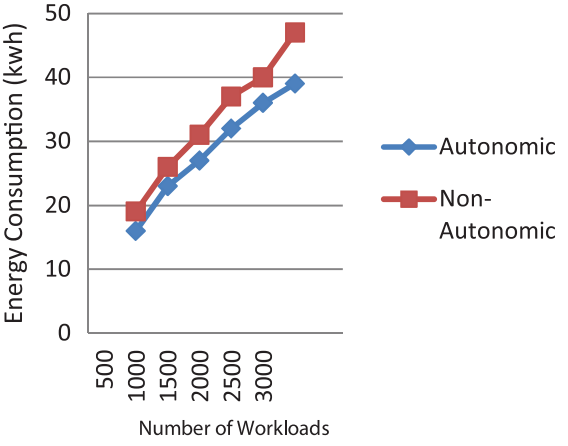

Fig. 23. Influence of change in number of workloads submitted on Energy Consumption.

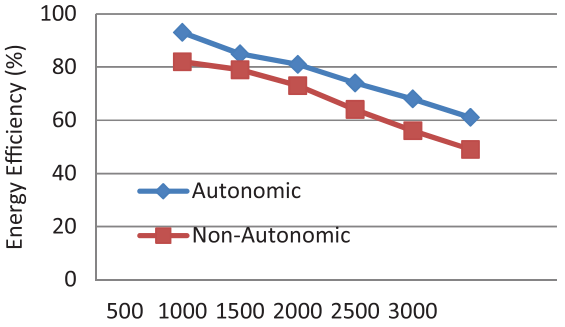

Fig. 24. Influence of change in number of workloads submitted on Energy Efficiency.

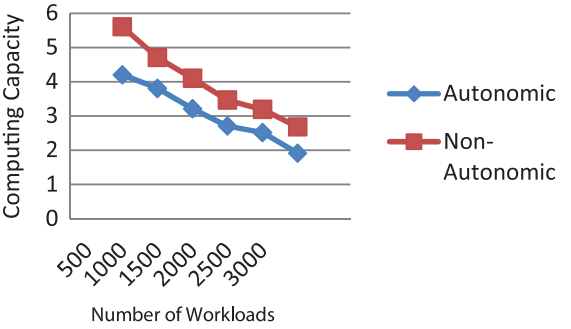

Fig. 25. Influence of change in number of workloads submitted on Computing Capacity.

workloads. We have used following formula to calculate energy efficiency (Equation 11):

$$Energy\ Efficiency_i = \sum_{i=1}^{n} \left(\frac{\text{number of workloads successfully executed in a data center}}{\text{total energy consumed to execute those workloads}} \right) \quad (11)$$

With increasing the number of Cloud workloads, the value of energy efficiency is decreasing. The value of energy efficiency in fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) is more as compared to non-autonomic based resource scheduling (non-autonomic) at different number of Cloud workloads as shown in Fig. 24. The maximum value of energy efficiency is 93% at 500 cloud workloads.

Test Case 4: Computing Capacity

We have calculated value of Computing Capacity for both fuzzy logic based energy-aware autonomic resource scheduling framework and non-autonomic based resource scheduling with different number of Cloud workloads. Computing Capacity is a ratio of actual usage time of resource to expected usage time of resource. We have used following formula to calculate Computing Capacity (Equation 12):

$$Computing\ Capacity_i = \sum_{i=1}^{n} \left(\frac{\text{actual usage time of resource}}{\text{expected usage time of resource}} \right) \quad (12)$$

With increasing the number of Cloud workloads, the value of Computing Capacity is decreasing. Fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) performs better than non-autonomic based resource scheduling (non-autonomic) at different number of Cloud workloads as shown in Fig. 25. The minimum value of computing capacity is 1.91 at 3000 cloud workloads.

Test Case 5: Latency

We have calculated value of latency for both fuzzy logic based energy-aware autonomic resource scheduling framework and non-autonomic based resource scheduling with different number of Cloud workloads. Latency is a defined as difference of time of input cloud workload and time of output produced with respect to that workload. We have used following formula to calculate Latency (Equation 13):

$$Latency_i = \sum_{i=1}^{n} (time\ of\ output\ produced\ after\ execution - time\ of\ input\ of\ cloud\ workload) \quad (13)$$

With increasing the number of Cloud workloads, the value of latency is increasing. The value of latency in fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) is lesser as compared to non-autonomic based resource scheduling (non-autonomic) at different number of Cloud workloads as shown in Fig. 26. The minimum value of latency is 1.24 seconds at 500 cloud workloads.

4.3. Statistical analysis

Statistical significance of the results has been analyzed by Coefficient of Variation (*Coff. of Var.*), a statistical method. *Coff. of Var.* is statistical measure

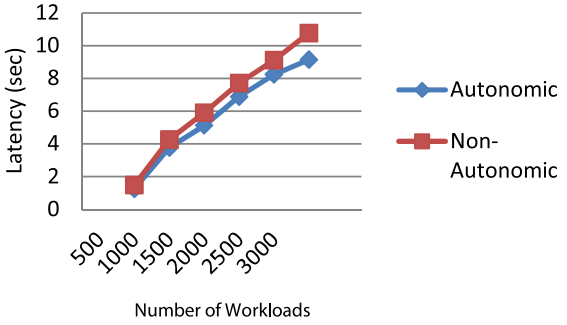

Fig. 26. Influence of change in number of workloads submitted on Latency.

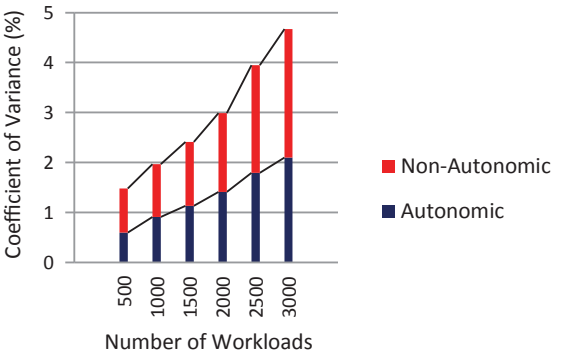

Fig. 27. CoV for Resource Utilization with each scheduling technique.

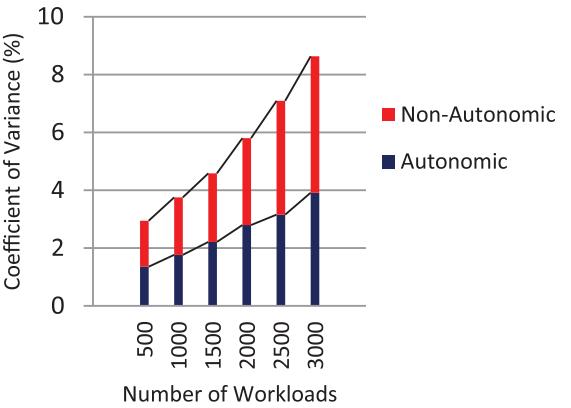

Fig. 28. CoV for Energy Consumption with each scheduling technique.

of the distribution of data about the mean value. *Coff. of Var.* is used to compare to different means and furthermore offer an overall analysis of performance of the framework used for creating the statistics. It states the deviation of the data as a proportion of its average value, and is calculated as follows (Equation 14):

$$Coff.\ of\ Var. = \frac{SD}{\mathrm{M}} \times 100 \qquad (14)$$

Where *SD* is a standard deviation and *M* is mean. *Coff. of Var.* of resource utilization and energy consumption has been studied of Cloud workload of both fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) and non-autonomic based resource scheduling (non-autonomic)as shown in Figs. 27 and 28. *Coff. of Var.* calculated for resource utilization and energy consumption results attained by fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) and non-autonomic based resource scheduling (non-autonomic). Range of *Coff. of Var.* (0.6%–2.10%) for resource utilization and (1.36%–3.91%) for energy consumption approves the stability of fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) as shown in Figs. 27 and 28.

Small value of *Coff. of Var.* signifies fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) is more efficient in resource scheduling in the situations where the number of Cloud workloads has changed. Value of *Coff. of Var.* increases as the number of workloads is increasing. Statistical analysis demonstrates the fuzzy logic based energy-aware autonomic resource scheduling framework (autonomic) outperforms non-autonomic based resource scheduling for large numbers of Cloud workloads. With small value of *Coff. of Var.* system is more stable fuzzy logic based energy-aware autonomic resource scheduling technique (autonomic) attained the best results in the Cloud for resource utilization and energy consumption as QoS parameters.

4.4. Discussions

The performance of fuzzy logic based energy-aware autonomic resource scheduling framework has been compared with existing non-autonomic based resource scheduling. The performance of proposed framework has been analyzed with different number of cloud workloads and resources. The performance of fuzzy logic based energy-aware autonomic resource scheduling framework has been evaluated with respect to resource utilization, energy consumption, energy efficiency, computing capacity and latency through case study. Energy consumption and resource utilization permits the evaluation for selection of resources whereas duration

of workload execution evaluates by latency and computing capacity. Fuzzy logic based energy-aware autonomic resource scheduling framework improves the resource utilization by up to 12.04% compared to existing non-autonomic based resource scheduling. Fuzzy logic based energy-aware autonomic resource scheduling framework reduces the energy consumption by up to 17.02% compared to non-autonomic based resource scheduling. The consequences describe that the computing capacity is decreased by increasing the number of workloads. With increasing the number of workloads, the latency is increasing. The Fuzzy logic based energy-aware autonomic resource scheduling framework executes the same number of Cloud workloads at a minimum energy consumption and maximum resource utilization. The energy consumption is increasing with the increase in number of Cloud workloads and the resource utilization of proposed framework for same number of Cloud workloads is better. The non-autonomic based resource scheduling based workload's execution caused in a schedule which is less efficient in comparison to the Fuzzy based energy-aware autonomic resource scheduling framework. The workload execution using the Fuzzy logic based energy-aware autonomic resource scheduling framework performs better as shown by all the experimental results. The overall energy consumption for Cloud consumer's workload execution is less. With the increase in resource utilization, the more number of resources are utilized efficiently. The Fuzzy logic based energy-aware autonomic resource scheduling framework executes the same number of Cloud workloads at a minimum energy consumption, computing capacity, latency and maximum resource utilization and energy efficiency. Considering all these QoS parameters and simulation outcomes, it is shown that the Fuzzy logic based energy-aware autonomic resource scheduling framework delivers a superior autonomic solution for heterogeneous Cloud workloads and approximate optimum solution for challenges of resource scheduling.

5. Conclusions and future directions

In this paper, fuzzy logic based energy aware resource scheduling framework has been proposed. Proposed framework has been validated in CloudSim based simulation environment, and real cloud environment and the experimental results perform better in terms of resource utilization and energy consumption. The proposed framework considers heterogeneous workload for resource scheduling and uses the concept of reallocation of VMs to improve resource utilization. This framework considers only two QoS parameters of self-optimization. We will develop a QoS aware autonomic resource provisioning and scheduling technique which will consider self-healing (find and react to sudden faults), self-optimization (maximize resource utilization and cost and time efficiency), self-configuration (capability to readjust resources) and self-protecting (detection and protection of cyber-attacks).

Acknowledgments

One of the authors, Sukhpal Singh (SRF-P), acknowledges the Department of Science and Technology (DST), Government of India, for awarding him the INSPIRE (Innovation in Science Pursuit for Inspired Research) Fellowship (Registration/IVR Number: 201400000761 [DST/INSPIRE/03/2014/000359]) to carry out this research work. We would like to thank all the anonymous reviewers for their valuable comments and suggestions for improving the paper. We would also like to thank Dr. Maninder Singh [EC-Council's Certified Ethical Hacker (C-EH)] for his valuable suggestions.

References

[1] S. Singh and I. Chana, Cloud based development issues: A methodical analysis, *International Journal of Cloud Computing and Services Science (IJ-CLOSER)* **2**(1) (2012), 73–84.

[2] I. Chana and S. Singh, *Quality of Service and Service Level Agreements for Cloud Environments: Issues and Challenges*. In Cloud Computing, Springer International Publishing, 2014, pp. 51–72.

[3] S. Singh and I. Chana, QRSF: QoS-aware resource scheduling framework in cloud computing, *The Journal of Supercomputing* **71**(1) (2015), 241–292.

[4] A. Beloglazov and R. Buyya, Energy efficient resource management in virtualized cloud data centers, *IEEE ACM International Conference on Cluster Cloud and Grid Computing (CCGrid)*, 2010, pp. 826–831.

[5] T.R.V. Anandharajan and M.A. Bhagyaveni, Co-operative scheduled energy aware load-balancing technique for an efficient computational cloud, *IJCSI International Journal of Computer Science Issues* **8**(2) (2011).

[6] N. Quang-Hung, P.D. Nienz, N.H. Namz, N.H. Tuong and N. Thoa, *A Genetic Algorithm for Power- Aware Virtual Machine Allocation in Private Cloud*, In Information and Communication Technology, Springer Berlin Heidelberg, 2013, pp. 183–191.

[7] T. Mofolo and R. Suchithra, Heuristic based resource allocation using virtual machine migration: A cloud computing perspective, *International Journal of Computer Application Technology and Research* (2013), 731–736.

[8] S. Pandey, L. Wu, S. Guru and R. Buyya, A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments, *in 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, 2010.

[9] H. Topcuoglu, S. Hariri and M.-Y. Wu, Task scheduling algorithms for heterogeneous processors, *in Heterogeneous Computing Workshop, (HCW'99)*, 1999.

[10] Z. Wu, X. Liu, Z. Ni, D. Yuan and Y. Yang, A market-oriented hierarchical scheduling strategy in cloud workflow systems, *The Journal of Supercomputing* **63**(1) (2013), 256–293.

[11] J. Yu, R. Buyya and C.K. Tham, Cost-based scheduling of scientific workflow applications on utility grids, *in Proceeding of IEEE E-Science and Grid Computing*, 2005.

[12] S. Singh and I. Chana, Energy based efficient resource scheduling: A step towards green computing, *International Journal of Energy, Information & Communications* **5**(2) (2014).

[13] S. Singh and I. Chana, QoS-aware Autonomic Resource Management in Cloud Computing: A Systematic Review, *ACM Comput. Surv.* **48**(3) (2015), 1–39.

[14] R.N. Calheiros, R. Ranjan, C.A.F.D. Rose and R. Buyya, CloudSim: A novel framework for modeling and simulation of Cloud computing infrastructures and services, *Grid Computing and Distributed Systems Laboratory*, The University of Melbourne, Australia, 2009.

[15] Green Investment Bank: Citibank UK data centre cut its energy usage in first of a kind project, [Online]. Available: http://www.clarke-energy.com/2014/green-investment-bank-citibank-uk-data-centre-cut-energy-usage-first-kind-project-2/ [Accessed 10 12 2014].

[16] A. Abraham, Rule-Based Expert Systems. Handbook of measuring system design, 2005.

[17] M.M.M. Fahmy, A fuzzy algorithm for scheduling non-periodic jobs on soft real-time single processor system, *Ain Shams Engineering Journal* (2010), 31–38.

[18] Computing, Autonomic. An architectural blueprint for autonomic computing. IBM White Paper, 2006.

[19] P. Varalakshmi, A. Ramaswamy, A. Balasubramanian and P. Vijaykumar, An optimal workflow based scheduling and resource allocation in Cloud. In Advances in Computing and Communications, Springer Berlin Heidelberg, 2011, pp. 411–420.

[20] K. Li, G. Xu, G. Zhao, Y. Dong and D. Wang, Cloud task scheduling based on load balancing ant colony optimization. *In Chinagrid Conference (ChinaGrid), IEEE*, 2011, pp. 3–9.

[21] S. Singh and I. Chana, Q-aware: Quality of service based cloud resource provisioning, *Computers & Electrical Engineering - Journal - Elsevier*. [http://dx.doi.org/10.1016/j.compeleceng.2015.02.003]

[22] F. DelaPrieta, S. Rodríguez, J.M. Corchado and J. Bajo, Infrastructure to simulate intelligent agents in cloud environments, *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology* **28**(1) (2015), 29–41.