Energy based Efficient Resource Scheduling: A Step Towards Green Computing

Energy based Efficient Resource Scheduling: A Step Towards Green Computing

1. Introduction

Each Cloud host is a computational node that executes a task or a subtask in the Cloud environment in which applications can deploy based on user requirements. Cloud customers pay the providers only when they access the computing resources. Therefore, Cloud consumers access services or resources based on their requests without regard to where the services are hosted. This model has been called as utility computing, or Cloud computing. Hence, Cloud computing is a new standard for the dynamic provisioning of computing services supported by state of art data centers that typically employ Virtual Machine (VM) technologies for consolidation and environment separation purposes. Cloud providers want to reduce the energy consumption of Cloud infrastructure. Intensifying energy cost increases the Total Cost of Ownership (TCO) and cuts the Return on Investment (ROI) of Cloud infrastructure. Though, the current Cloud infrastructure has no or restricted attention for supporting energy-aware service distribution that encounters QoS requirements of customers and reduces energy costs to increase ROI.

VM Migration permitted by virtualization can aid in load balancing, allowing high provisioning and avoiding hot-spots in data centers which decreases energy consumption. Server

consolidation helps in improving resource utilization by consolidating many VMs residing on multiple under-utilized servers onto a single server, hereafter decreasing energy consumption. Load balancing can help in decreasing energy consumption by regularly dispensing the load and decreasing the resource consumption.

This paper focuses on the predominant energy aware techniques in Cloud environment. The central objective of this work is to initiate research and development of energy-aware task consolidation mechanisms and strategies for data centers in such a way that brings Cloud computing as a maintainable ecological conventional technology for achieving cost-effective, organized, and technical enhancement for future generations.

The rest of paper is organized as follows: Section II discusses related work. Section III describes energy based efficient resource scheduling framework. The verification of proposed framework is presented in Section IV. The experimental setup and results have been described in Section V. The contributions of this research are presented in Section VI. Section VII describes conclusions and future scope.


GREEN
COMPUTING

2. Related Work

Qiang Li *et al.*, proposed an architecture using response control theory for adaptive management of virtualized resources is based on VM. Hosted applications access the mandatory resources as per their requirement to meet Service Level Objectives (SLO) if all hardware resources are combined into common shared space in VM-based architecture. Multi Input Multi-Output (MIMO) resource manager is used as adaptive manager in this architecture which includes three controllers: CPU controller, memory controller and I/O controller to control the energy consumption.

Jiayin Li *et al.*, proposed an adaptive resource allocation algorithm for the Cloud system with pre-emptible tasks which regulate the resource allocation adaptively based on the updated actual task executions. Static task scheduling for static resource allocation uses Adaptive List Scheduling (ALS) and Adaptive Min-Min Scheduling (AMMS) algorithms for task scheduling. The online adaptive process is used for re-evaluating the lasting static resource allocation frequently with predefined frequency. In each re-evaluation process, the schedulers are re-calculating the power consumption of their particular submitted tasks.

Two-layer architecture that uses utility functions, adopted in the context of dynamic and autonomous resource allocation contains

of local agents and global arbitrator. The local agents are used to calculate utilities, for given current or

predicted workload and their energy consumption and range of resources, for each AE and results are transferred to global arbitrator.

Yazir Y. O. *et al.*, contributes in two-fold, in which first

scattered architecture is adopted and resource management is divided into independent jobs, each of which is accomplished by Self-governing Node Agents (NA) in a cycle of three activities:

(1) VM Placement, where VM is allocated to by discover appropriate Physical Machine (PM) which is capable of running given VM;

(2) Total resources used by hosted VM is monitored;

(3) If local accommodation is not possible, a VM need to migrate at another PM and process loops back to assignment during VM selection. This method is hypothetically more viable in large data centers than centralized approaches to evaluate energy consumption.

Goudarzi H. *et al.*, considered the issue of resource allocation to improve the total profit enlarged from the multi-dimensional Service Level Agreement (SLA) contracts for multi-tier application. Now the upper bound of total profit is provided with the help of Force- directed Resource Assignment (FRA) heuristic algorithm and the initial solution is based on solution for profit upper bound problem. Next, distribution rates are fixed and local optimization step is used for improving energy consumption of shared resources. Shi J. Y. *et al.*, proposed steady state timing

models for the study of Cloud HPC resource planning, where quantitative application dependent instrumentation method is used to inspect numerous important factors of energy. Aoun R. *et al.*, suggested concept of Mixed Integer Liner

Program (MILP) formulation for resource provisioning and enriched services in Cloud environment. Computing, storage, point-to-point data transfer, and point-to multipoint data transfer are the four types of end-Cloud consumer requests deliberated by the researchers.

Newly, Internet-based distributed, multi-tenant applications connective to internal business applications, known as Software as a Service, are gaining popularity. However, in earlier years, the goals of distributed system have been centered on the decoupling of interfaces from Service Oriented Architectures (SOA), implementation, subscription model, hosting models and social collaboration. The earlier work on web application scalability instigated for static load balancing solution with server clusters but the dynamic scaling of web applications and energy consumption of resources in virtualized Cloud computing has not been impressively considered. A resolution for ynamic scaling of web application delivered in by describing an architecture to scale web application in enthusiastic method, based on threshold in a virtualized Cloud computing environment.

T. C. Chieu *et al.*, advanced the task consolidation using the outdated bin-packing problem with two central features:

(a) CPU usage and

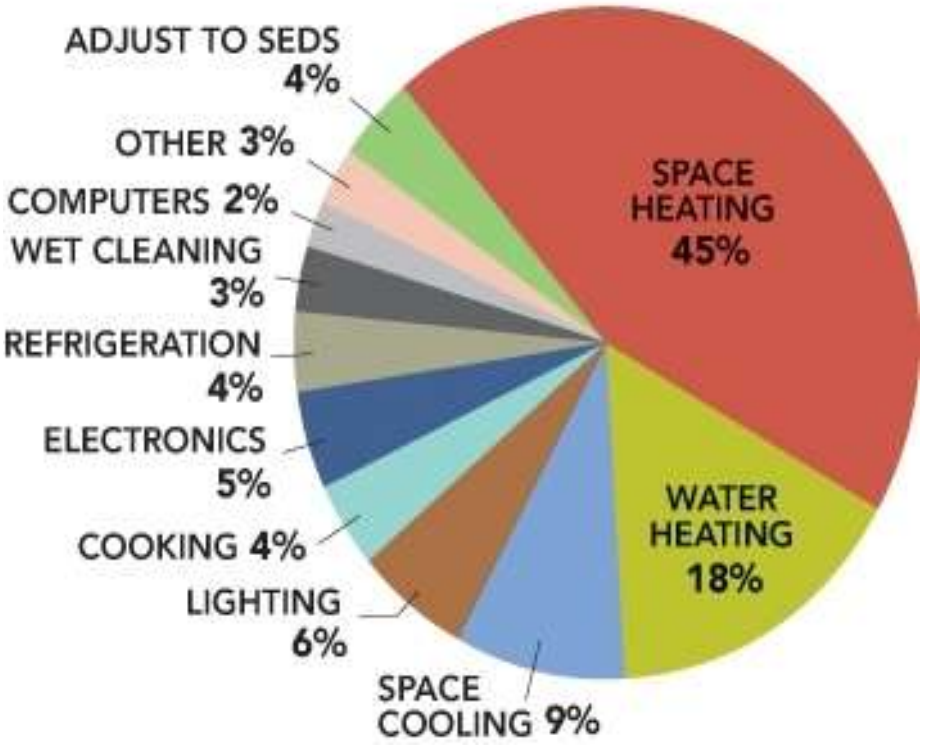
(b) Disk usage.

The recommended algorithm combines the tasks trusting on the Pareto front to stable the energy intake and the performance. V. Ungureanu *et al.*, suggested a utility investigative model for Internet- oriented task association. The model studies task's demand for Web services such as e-books database or e-commerce. The proposed model targets to upturn the resource utilization and to decrease the energy consumption, proposing the similar quality of services appropriate to the dedicated servers. Y. C. Lee *et al.*, proposed energy efficient model only for homogenous cloud workloads.

N. Sadashiv *et al*., presented the task consolidation mechanisms deals with the energy reduction using rare methods. Unlike characteristic task consolidation approaches, the method adopts two interesting techniques:

(1) memory compression and

(2) request percipience.


ADJUST TO SEDS
4%
OTHER 3%
COMPUTERS 2%
WET CLEANING
3%
REFRIGERATION
4%
ELECTRONICS
5%
COOKING 4%
LIGHTING
6%
SPACE
COOLING 9%
WATER
HEATING
18%
SPACE
HEATING
45%

The first permits the transformation of the CPU power into extra memory capacity to allow more (memory intensive) tasks to be consolidated, while the second blocks useless/unfavorable requests to exclude unnecessary resource usage. Key areas of research can be given as making use of computers as energy efficient and designing new algorithms and systems for energy efficiency. There is a need of novel technique which will reduce the energy consumption while considering both homogenous and heterogeneous Cloud workloads.

3. Energy Based Efficient Resource Scheduling Framework

Energy based Efficient Resource Scheduling Framework (EBERSF) schedules the resources on the basis of energy as a QoS parameter as shown in Figure 1.

Essential Guide:

Using green computing for improving energy efficiency

3.1. Type of operation: Resource Scheduling

Resource Scheduling Framework schedules the resources on the basis of energy-based resource scheduling algorithm. First of all, Cloud consumer will try to access the Cloud resources for the execution of the Cloud applications through a Cloud portal. After this a certificate will be produced, and the authentication and authorization would be accomplished through Cloud Security Infrastructure (CSI). CSI Security will communicate with the broker. Broker will gather the information about the resources and Cloud workload status. Cloud computing resources are characteristically functioned under the control of an Energy Consumption Center which implements allocation and prioritization policies while optimizing the execution of all submitted Cloud workloads for efficiency and performance.

An SLA agreement is official document to describe the QoS parameter (energy) in written form.

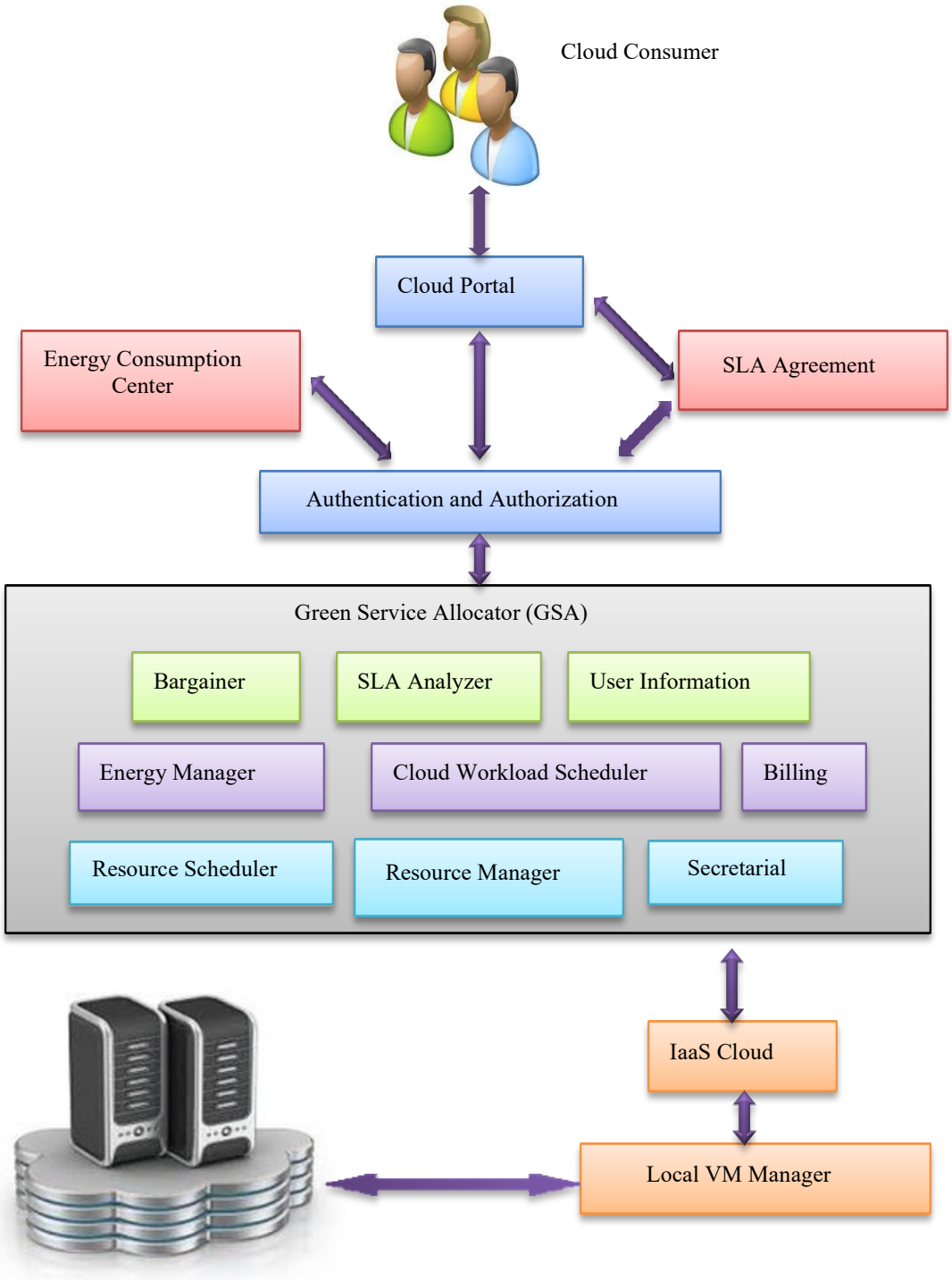


Figure 1. Energy based Efficient Resource Scheduling Framework

Green Service Allocator (GSA) is not a resource scheduler, nonetheless somewhat a procedure application for collaborating with a range of dissimilar local resource schedulers using a standard message format. GSA will communicate with the Resource Manager for resource scheduling. Resource Manager will take the information about algorithm which is stored in the Cloud workload scheduler. The Cloud workload scheduler contains the workload documents and programmatic interface for instantiating scheduling constraints. Resource scheduler is a logical entity that makes scheduling decisions for it or for other network elements that demand such choices. Resource Manager is a logical entity that applies scheduling decisions according to Cloud consumer's requirements. Resource Manager checks for availability of the resources according to scheduling conditions and then schedule the resource to Cloud consu mer's application and after that the scheduler will do scheduling.

Local Resource Manager handles operations such as resource allocation, configuration and advance booking, *etc.* Consequently, getting the result, it is redirected back to the Cloud

consumer. Bargainer bargains with the customers/brokers to confirm the SLA with identified prices and fines (for violations of SLA) between the Cloud provider and customer depending on the consumer's QoS requirements and energy saving patterns. Service Analyzer Interprets and examines the service requirements of an acquiesced request before deciding whether to receive or reject it. Consequently, it wants the most recent Cloud workload and energy information from Resource Manager and Energy Manager. Cloud consumer information gathers particular features of customers so that significant customers can be allowed distinctive rights and prioritized over other customers. Billing decides how service requests are charged to accomplish the supply and demand of computing resources and facilitate in prioritizing service allocations effectively. Energy Manager notices and defines which physical machines to power on/off. Cloud Workload Manager allocates requests to resources and concludes resource rights for allocated resources. It consistently chooses when resources are to be added or removed to meet customer requirement. Resource Manager retains track of the availability of resources and their resource rights. It is respectively in charge of relocating resources across Physical Machines. Secretarial maintains the authentic usage of resources by requests to calculate usage charges. Past usage data can also be used to improve service allocation judgments.

3.2. Objectives and Commitments

The objective of Energy based Efficient Resource Scheduling Framework is to ensure that the framework will provision resources for execution of the Cloud workloads. It facilitates to:

1) Evidently comprehend the present and possible forthcoming desires and outlooks of resource consumers.
2) Measure at a suitable level service performance and reduce scheduling expense by minimizing energy consumption.
3) Improve customer gratification by meeting their desires.

3.3. Methodology

To achieve the above-mentioned objectives, the methodology has been divided into following three steps:

1) Cloud workload scheduler manager will ask the Cloud consumer to submit SLA form to fill their requirements. So that it can provide the facility according to Cloud consumer's requirements.
2) Energy Manager tries to minimize the energy consumption of Cloud consumer's application execution resources.
3) Once energy consumption is minimized, the cost of resources will automatically be reduced.

3.4. Energy based Efficient Resource Scheduling Algorithm

Taking energy as a QoS parameter, a resource is used for the scheduling of Cloud workload execution on consideration of energy consumption of the resources. Energy is a significant feature to be deliberated at the time of resource scheduling. Power consumption is identified as per unit of resources that are consumed by the Cloud consumer for execution of their Cloud workloads. After the minimization of the energy consumption of the resources, resource would be scheduled. Consequently, energy based QoS can be provided to resource

consumer. Below is the proposed algorithm to calculate the power consumption of the data center by the Heterogeneous Cloud Workload Consolidation. The algorithm takes Cloud workload list (Cloud Work load List) as an input and gives migration list (Migration List) of the resources and power consumed by the data center. First the algorithm sorts the list of resources in a decreasing order of CPU utilization and then it continually looks for the best resource to migrate from the host and also calculate the power consumption of the data center. The procedure halts by giving the reposition (migration) list of the resource ‘s and the power consumed by the data center. The Energy based Efficient Resource Scheduling Algorithm is shown in Figure 2. UPPER Threshold is the maximum limit of energy consumption.

Algo: Energy based Efficient Resource Scheduling Algorithm

1. **Input:** Cloud Work load List and **Output:** Migration List,
2. Foreach Cloud workload in Cloud Work load List Do
3. Resource List - Cloud work load. Get Resource List
4. Resource List. Sort Reducing Utilization
5. Cloud Work load Utilization - Cloud work load. Get Utilization
6. Optimum Utilization - MAXIMUM
7. MINIMUM *PCP* consumption - MAXIMUM
8. While Cloud Work load Utiliztion > UPPER_Threshold Do
9. Foreach Resource in Resource List Do
10. if Cloud Workload has enough resources Then
11. *PCP* consumption - approx *PCP* consumption (Cloud Workload, Resource)
12. if Resource. Get Utilization( ) > Cloud Workload Utilization – UPPER_Threshold Then
13. a - Resource. Get Utilization( ) – Cloud Workload Utilization + UPPER_Threshold
14. if a < Optimum Utilization Then
15. Optimum Utilization - a
16. Optimum Resource – Resource else
17. if Optimum Utilization = MAXIMUM Then
18. Optimum Resource - Resource
19. Break
20. Cloud Workload Utilization – Cloud Workload Utilization – Optimum Resource. Get Utilization( )

21. Migration List. Add Optimum Resource)
22. Resource List. Remove (Optimum Resource) 24 return Migration List,

Figure 2. Energy based Efficient Resource Scheduling Algorithm

3.5. Energy Model of Cloud Computing

The overall energy consumption of Cloud computing system can be expressed as the following formula:

*PCP C*onsumption = *PCP D*atacenter + *PCP* Transceivers + *PCP Memory* + *PCP Extra*

*PCP D*atacenter 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. The above formula can be further disintegrated; a Cloud computing environment with d datacenters, transceivers

equipment and a centralized memory device, its energy consumption can be expressed as:

$$PCP_{Consumption} = d(P_{CPProcessor} + PCP_{PrimaryStorage} + PCP_{secondarystorage} + PCP_{Motherboards} + PCP_{NetworkCards}) + t\,(PCP_{Hardware} + PCP_{LANcards} + \sum_{f=0}^{F} d_{connectors,f} + PCP_f) + (P_{NetworkAnalysisServer} + P_{MemoryManager} + P_{NetworkAttachedStorageArrays}) + P_{Extra}$$

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 of transceiver operation. Therefore, and can be avoided by turning the transceiver off or putting it into sleep mode.

Our energy model is devised on the basis that Resource Utilization ( ) has a linear relationship with energy ingestion. For a specific Cloud workload, the information on its processing time and resource utilization is sufficient to measure the energy consumption for that task. For a resource at any given time, the utilization is defined as

$$RU_a = \sum_{b=1}^{c} RU_{a,b}$$

where c is the number of Cloud workloads running at that time and RUa,b is the resource usage of Cloud workload w_a. The energy consumption E_{cloud} of a resource r_i at any given time is defined as

$$PCP_{Consumption} = (PCP_{maximum} - PCP_{minimum}) \times RU_a + PCP_{minimum}$$

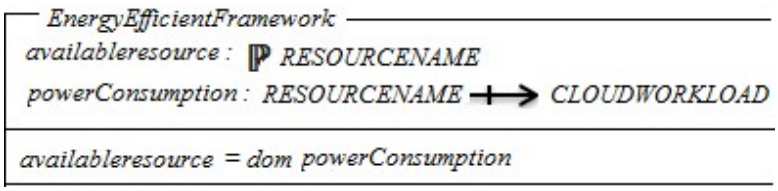
where is the power consumption at the peak load (or 100% utilization) and *PCP*minimum is the minimum power consumption in the active mode (or as low as 1% utilization).

4. Verification of Proposed Framework

The Formal method is used in developing critical computer systems and fault avoidance techniques and help in reduction of errors. Energy based Efficient Resource Scheduling Framework (EBERSF) stated above can be verified using a formal language like Z specification. In Z, schemas are used to describe both static and dynamic aspects of a system. The states it can occupy; the invariant relationships that are maintained as the system moves from one state to another state. In our EBERSF Framework, we need to deal with resources and consumers. In this framework, we

will provision the resources to those consumers whose requirement as energy consumption is fulfilled by energy manager. We introduce the set of all resources and the set of all consumers as basic types of the specification:

The first aspect of the Energy Efficient Framework is its state space.



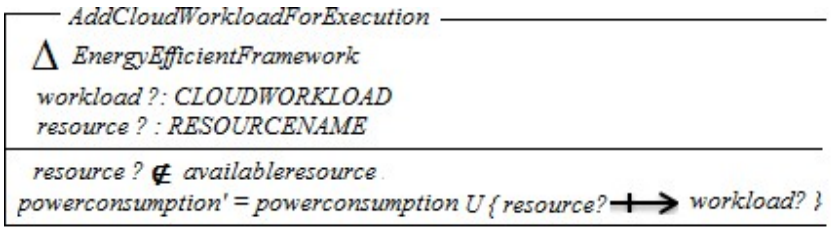
In our work, the space of power consumption has been described and the three variables represent important observations which can make the state.

- *availableresource* is the set of available resources.
- *powerconsumption* is a function that when applied to certain resources (Res), calculate the power consumption of workloads with resources associated with them.
- *set* item is the same as the domain of the function allocate the resources to which it can be validly applied.

availableresource = {Res1, Res2, Res3}

powerconsumption = {Res1 - Cloudworkload3
Res2 - Cloudworkload2
Res3 - Cloudworkload1}

The invariant is satisfied because execution details a CLOUDWORKLOD for three RESOURCENAME in power consumption. There are some operations that can apply on the Energy Efficient Framework: The first of all there is to add a new cloud workload, and we describe it with schema:



The Δ *EnergyEfficientFramework* alerts us to the fact that the schema is describing a state change: it introduces four variable availableresource, powerconsumption, availableresource' and powerconsumption'. The first two are observations of the state earlier the modification, and the most recent two are interpretations of the state after the change. We expect that the set of resource known to *EnergyEfficientFramework* will augmented with new resource.

powerconsumption' = powerconsumption U {resource ?}

We can prove this from the specification of *AddCloudWorkloadForExecution* using the invariants on the state before and after.

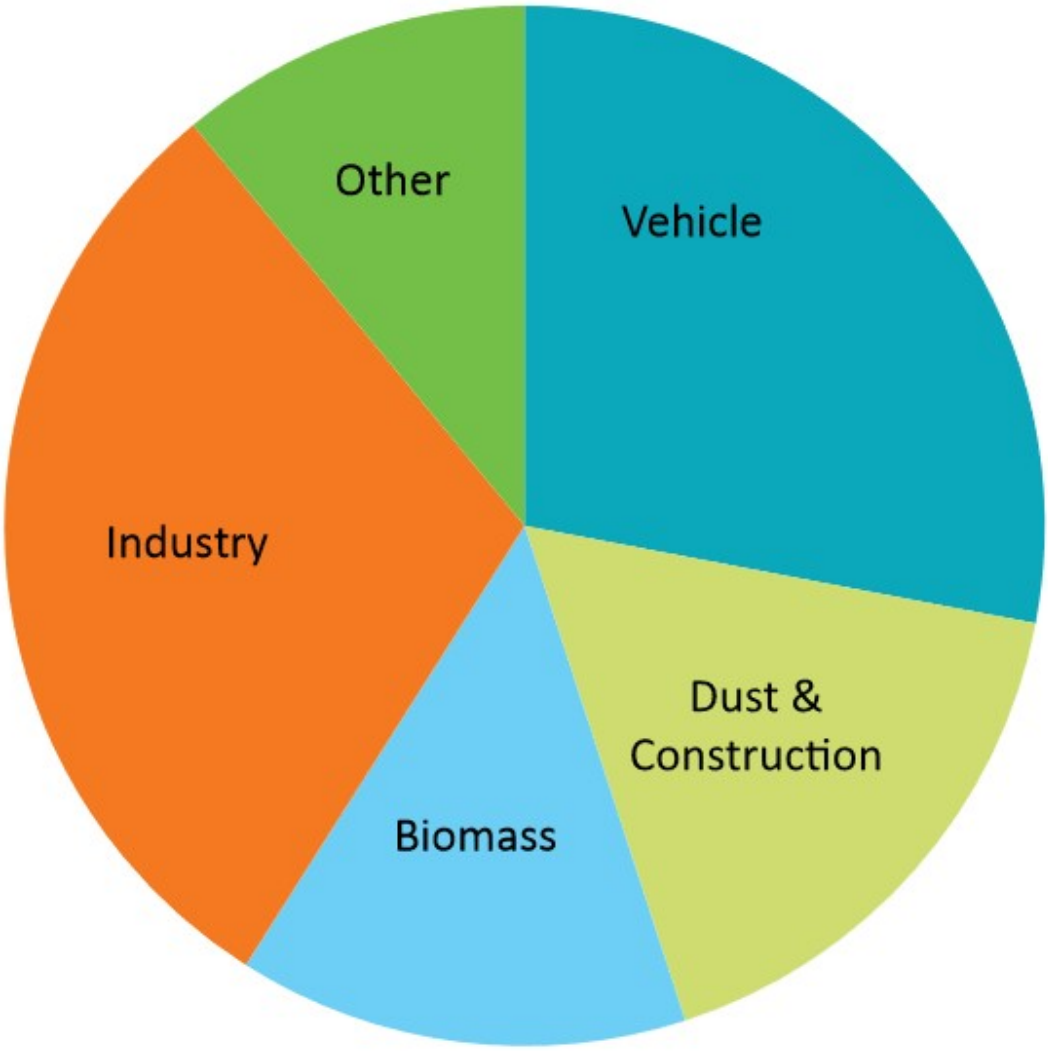
availableresource ' = dom powerconsumption'

= dom (powerconsumption U {resource? □ workload?})

= dom powerconsumption U dom {resource? □ workload?

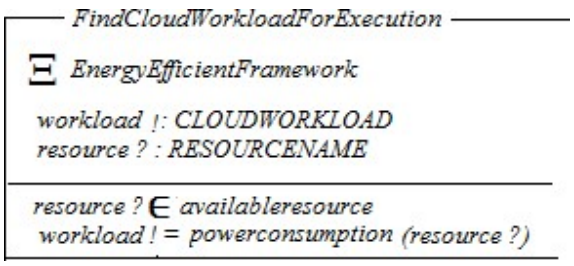
= dom powerconsumption U {resource?}

= powerconsumption U {resource?}



Source: NCAP report

In *FindCloudWorloadForExecution*, find the resources to execute the Cloud workloads based on user requirements.

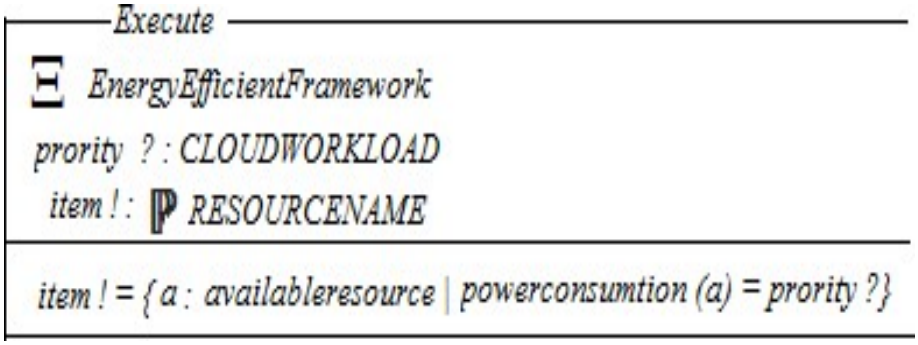


The declaration *ΞEnergyEfficientFramework* indicates that this is an operation in which the state does not change, the value of avilableresource' and powerconsumption' of the observations after the operation are equal to these values availableresource and powerconsumption. Including *ΞEnergyEfficientFramework* above the line has the same effect as including Δ*EnergyEfficientFramework* above the line and two equations below it.

availableresource' = availableresource

powerconsumption' = powerconsumption

The other notation (!) for an output the *FindCloudWorloadForExecution* operations take an avilableresource as input and yield corresponding execution as output. The most useful operation on Energy Efficient Framework is one to find which workload execute with available resource. The operation has an input prority? And one output, item! Which is set of resources for powerconsumption? There may be zero, one or more workloads execute with particular resource, to whom resource item should be sent.



This time there is no pre-condition. The item! is specified to be equal to the set of all values a drawn from the set item such that the value of the powerconsumption function at a is priority?. In

general, b is a member of the set {e: D |……..e……} exactly if b is a member of D and the condition ……b……, obtained by replacing e with b, is satisfied:

$b \in \{e: D|\ldots e\ldots..\} \leftrightarrow b \in D \wedge (\ldots.b\ldots.)$

$c \in \{a:$ item $\wedge$ powerconsumption (c)= priority? $\} \leftrightarrow c \in$ item $\wedge$ powercosumption (c) = priority?

A name c is in output set item! exactly if it is known to the Energy Efficient Framework and the allocation recorded for it is priority?

4.1. Implementing Proposed Framework

The proposed Energy based Efficient Resource Scheduling Framework has been implemented at Thapar University, Patiala, Punjab. A Cloud setup has been established. The details of Cloud setup using Energy based Efficient Resource Scheduling Framework has been presented in the form of case study in the next section.

4.2. Case Study: Thapar Cloud

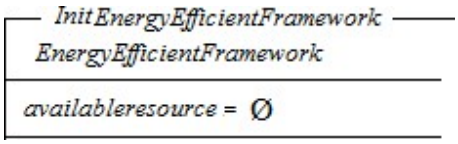
Different Labs of Computer Science and Engineering Department have individual computational resources. Specification of resources has been shown in Table 1. Cloud environment has been setup in the High End Computing Lab (HECL) for implementing Energy based Efficient Resource Scheduling Framework. For setting Cloud environment in labs, Cloud Foundry [26] has been installed on main server in HECL lab and all the other nodes are connected to main server for the verification of proposed framework. KVM (in Linux 4.3) and Xen (Linux 3.5) are used to measure the performance within a VMware by taking three different types of resources in this research.

Table 1. Resource specifications

Name	Lab	Resource Specification
	Grid Lab	PC C-2-D Quad /250GB Sata
	SE Lab	IBM C-2-D 2.33 Ghz
	HECL Lab	IBM PCs P4 3.0 Ghz Dual core 2.8 C-2-D 2.33 Ghz, HP Proliant DL 180 G6(590638- 371): G6 E5620 1P 8 GB- R P410/256 BBWC/ (1) Intel Xeon E5620(2.40 GHZ/ 4 Core/ 12 MB/ 80W)/ 8MB shared L3 cache/ 8 GB (2x4GB) PC3-10600R-9(RDIMM)
	Self-Maintenance Lab.	P-IV 3.0 GHz
	New PG Lab	P-IV (HT),3.0 GHz
	Network Research Lab	Dell optiplex 330 PIV Dual Core 1.6 GHz

Test Cases for Energy based Efficient Resource Scheduling Framework have been considered are:

Test Case 1: Initial State of Energy based Efficient Resource Scheduling Framework The given below schema identify the initial state of the Energy Efficient Framework:



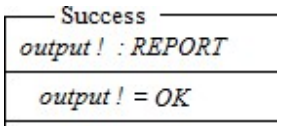
This schema describes an *EnergyEfficientFramework* in which the set known is empty: in consequence, the function powerconsumption empty too.

Test Case 2: Successful Working of Proposed Framework

We shall add an extra output! to each action in Energy Efficient Framework. After successful execution of given process the outcome will be OK, but it may take the other value AlreadyExecuted and NotExecuted when error is detected. REPORT defines the set contains three values.

REPORT::=OK/ AlreadyExecuted/NotExecuted

The result should be OK after proper execution of success schema without saying how the state changes.

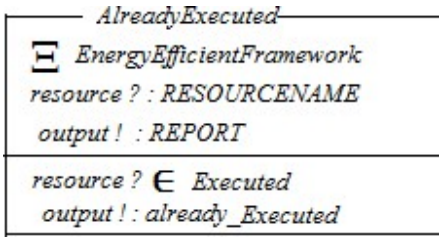


EnergyEfficientFramework $\wedge$ *Success*

The conjunction operator $\wedge$ of the schema calculates allows us to combine this description with our previous description of *EnergyEfficientFramework.* The process for accurate input has defines both acts as described by *EnergyEfficientFramework* and produces the result OK.

Test Case 2.1: Add same resource to available resource list

Schema specified that the report AlreadyExecuted should be produced when input resource? Is already a Cloud Workload is executed?



Ξ *EnergyEfficientFramework* specifies that if the error occurs, the state of the Energy Efficient Framework should not change.

$$R\,EnergyEfficientFramework \triangleq EnergyEfficientFramework \vee AlreadyExecuted$$

R EnergyEfficientFramework

Δ *EnergyEfficientFramework*

workload ?: CLOUDWORKLOAD
resource ? : RESOURCENAME
output ! : REPORT

(resource ? ∈ Executed ∧
powerconsumption' = powerconsumption ∪ { resource? ⟶ workload? } ∧
output ! = OK) ∨
resource ? ∈ Executed ∧ powerconsumption' = powerconsumption ∧
output ! : already_ Executed

Test Case 2.2: Find resource which is not in available resource list

Schema specified that the report NotExecuted should be produced when input resource? Is a Cloud Workload is not executed?

Not Executed

Ξ *EnergyEfficientFramework*

resource ? : RESOURCENAME
output ! : REPORT

---

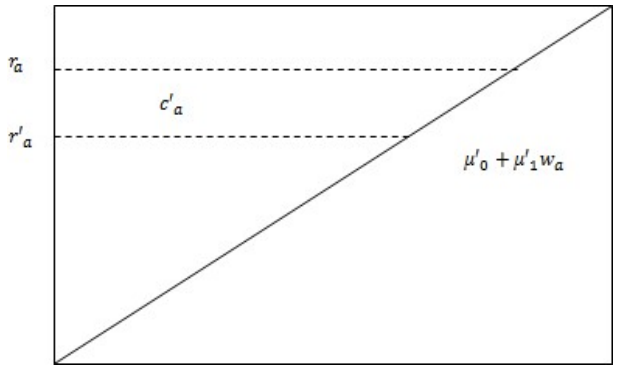
resource ? ∉ Executed
output ! : not_Executed

$$R\,FindCloudWorkloadForExecution \triangleq (FindCloudWorkloadForExecution \wedge \text{Success}) \vee NotExecuted$$

Mapped operation can be called at any time, it never results an error robust version need only add reporting of success.

$$RExecuted \stackrel{\wedge}{=} Executed \ \Lambda \ \text{Success}$$

The idea of formal specification is that the cloud provider gives the facility of execution of cloud workloads for optimum results. The implementation of this framework contributes to the achievement of minimum energy as least as possible.



5. Experimental Setup and Results

For experimental results, different types of resources have been considered. Energy based Efficient Resource Scheduling Framework (EBERSF) give the facility of resource scheduling to Cloud consumer for optimum results and better services and avoid the violations of service level guarantees by taking energy as a QoS parameter. The implementation of this framework will enable the Cloud consumer to analyze customer requirements and define processes that will contribute to the achievement of a product or service that is acceptable to their resource consumer and executes all the submitted Cloud workloads by using minimum energy consumption ( ) i.e. it should be less than threshold value (UPPER_Threshold) at particular number of Cloud workloads.

5.1. Performance Evaluation Criteria

The performance evaluation is performed through the simulation by using CloudSim. In addition to designing an EBERSF, we have also directed a comprehensive performance analysis. We have verified the Energy based Efficient Resource Scheduling Framework. We have compared our algorithm with existing approach i.e. Energy Efficient Utilization of Resources (EEUR) in Cloud Computing Systems.

5.2. Results

We selected energy as a QoS parameter in all the three cases: Low resource usage, High resource usage and Random resource usage. The energy is measured in terms of Mega Watt Hour (MWH), One MWH is equal to One Millions Watt Hour. Energy saving using three different task consolidations techniques is calculated and compared at different number of Cloud workloads.

5.2.1 Low Resource Usage: To validate our framework, 103 resources and 4000 cloud workloads (Cloudlets) are considered. Figure 3 show that the Low resource usage, we conclude that the energy consumption of Cloud workload reduces by using our framework. The unit used to measure energy consumption in Figure 3, Figure 4 and Figure 5 is Millions Watt Hour, denoted by Millions.

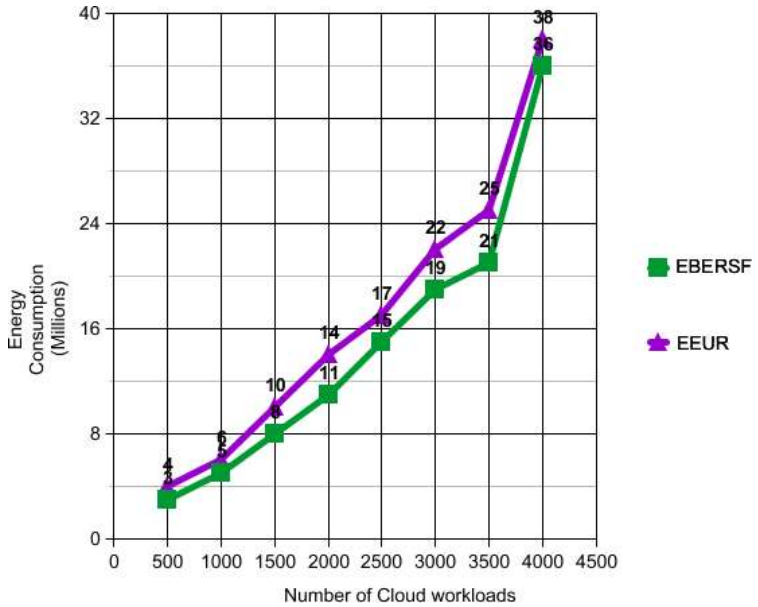


Figure 3. Low Resource Usage

5.2.2 High Resource Usage: Figure 4 shows that the High resource usage, in which the energy consumption is reduced and lesser than low resource usage with same number of cloud workloads.

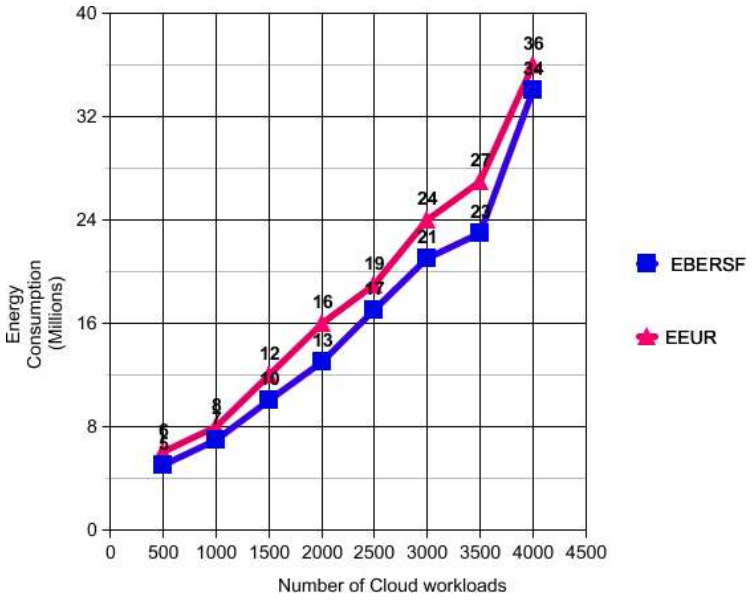


Figure 4. High Resource Usage

5.2.3 Random Resource Usage: Figure 5 show that Random resource usage, in which our framework performs better and energy consumption at 4000 cloud workloads is lesser as compared to low and high resource usage, and EEUR.

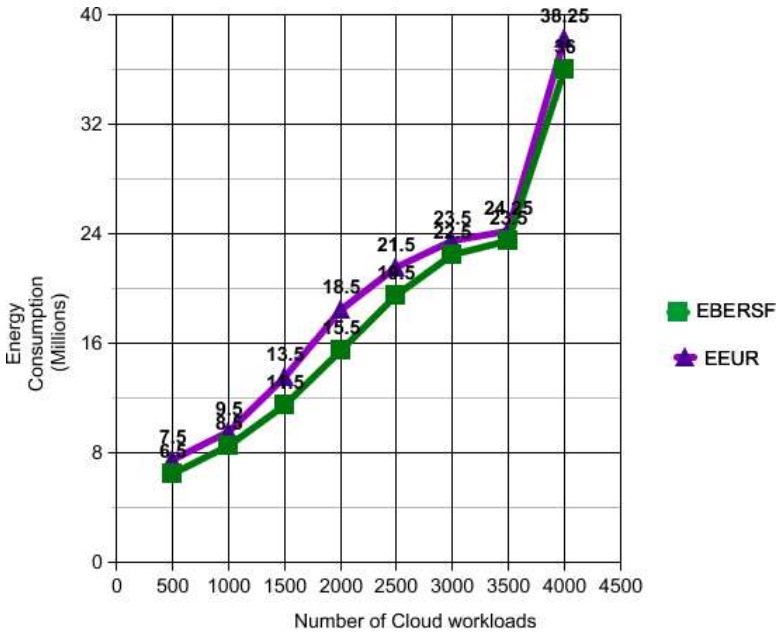


Figure 5. Random Resource Usage

Table 2 shows the comparison of average energy savings of EBERSF and EEUR in all the three above mentioned cases.

Table 2. Average Energy savings comparison

QoS Parameter	EBERSF	EEUR
Energy Savings (Millions)	23.391%	18.7194%

It has been clearly demonstrated that our framework performs better in all the three cases, *i.e.*, Low, High and Random Resource usage. The average amount of energy saved during migrations in our framework is more than EEUR.

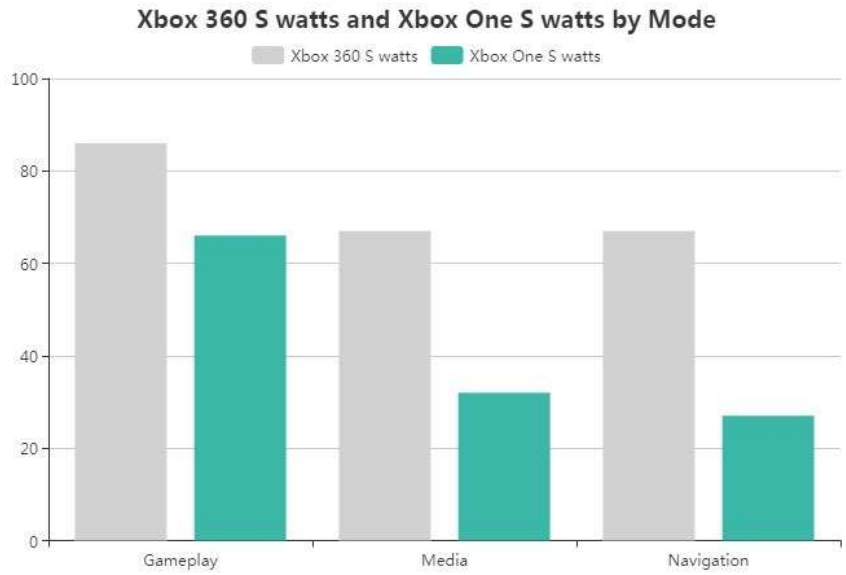
6. Our Contributions

We have proposed a resource scheduling framework for Cloud system in this paper. The resource scheduling framework is responsible for the verification of the QoS parameter i.e. energy. On the basis of the proposed algorithm, we have identified how to execute cloud workloads to available resources in order to reduce energy of the independent Cloudlet. Finally, we have analyzed empirically the performance of the proposed algorithm using the CloudSim simulator in three different cases: Low, High and Random resource usage. Techniques used in the traditional Cloud scheduling have not been very successful in producing efficient and effective results

in terms of energy to manage the resources. The implementation of our proposed algorithm is to minimize the energy and due to energy the cost will be reduced automatically. Formal specifications of resource scheduling framework have ensured that the provider executes resources as per the power consumption. We have developed EBERSF, implemented our proposed resource scheduling algorithm and tested the performance in a simulated Cloud environment.

7. Conclusions and Future Scope

In this paper, we have proposed Energy based Efficient Resource Scheduling Framework for Cloud environment. Energy based Efficient Resource Scheduling Framework can assist organizations in reducing power consumption and contribute directly to the company's growth and institution's progress. Verification and implementation of the proposed policy has been done with the help of Z specification language. Performance evaluation has been done with the help of Cloudsim. In future, more QoS parameters may be incorporated to achieve better performance. CloudSim toolkit has been used to collect the current results, the same results would be verified on actual cloud resource present at Center of Excellence (CoE) in Grid Computing at Thapar University.



REFERENCES:

https://www.researchgate.net/publication/28446750

[1] L. Qiang, Q. Hao, L. Xiao and Z. Li “Adaptive Management of Virtualized Resources in Cloud Computing Using Feedback Control”, in First International Conference on Information Science and Engineering, **(2010)** April, pp. 99-102.

[2] J. Li, M. Qiu, J. -W. Niu, Y. Chen and Z. Ming, “Adaptive resource Allocation for Pre-emptable Jobs in Cloud Systems”, in 10th International Conference on Intelligent System Design and Application, **(2011)** January, pp. 31-36.

[3] W. E. Walsh, G. Tesauro, J. O. Kephart and R. Das, “Utility Functions in Autonomic Systems”, in ICAC “04: Proceedings of the First International Conference on Autonomic Computing. IEEE Computer Society, **(2004)**, pp. 70–77.

[4] Y. O. Yazir, C. Matthews, R. Farahbod, S. Neville, A. Guitouni, S. Ganti, Y. Coady, “Dynamic resource allocation based on distributed multiple criteria decisions in computing Cloud”, in 3rd International Conference on Cloud Computing, **(2010)** August, pp. 91-98.

[5] N. Kim, J. Cho and E. Seo, "Energy-credit scheduler: an energy-aware virtual machine scheduler for cloud systems", Future Generation Computer Systems, vol. 32, **(2014)**, pp. 128-137