Islamic University of Technology (IUT)

QoS enabled Green Cloud: Architecture and Policy

Submitted by

Md. Fakhrul Alam Onik (**084416**)
Md. Asif Ahmad Oni (**084420**)

Supervisor

Md. Ali Al Mamun
Assistant Professor
Department of Computer Science and Engineering
Islamic University of Technology

Co-Supervisor

Dr. Kamrul Hasan, PhD
Assistant Professor
Department of Computer Science and Engineering
Islamic University of Technology

CERTIFICATE OF RESEARCH

This is to certify that the work represented in this thesis paper is the outcome of the investigation carried out by the candidates under the supervision of Md. Ali Al Mamun, Assistant Professor, in the Department of Computer Science and Engineering, Islamic University of Technology, Gazipur. It is also declared that neither of this thesis work or any part thereof has been submitted anywhere else for award of any degree or diploma.

Signature of Authors:

_____ _____

Md. Fakhrul Alam Onik
Student No: 084416

Md. Asif Ahmad Oni
Student No: 084420

Date: _____

Signature of Supervisor:

_____

Md. Ali Al Mamun
Assistant Professor, Department of Computer Science and Engineering
Islamic University of Technology

Date: _____

Signature of Head of Department:

_____

Prof. Dr. M. A. Mottalib
Head, Department of Computer Science and Engineering
Islamic University of Technology

Date: _____

ACKNOWLEDGEMENT

First of all , we thank Almighty for making everything possible. Secondly, we are extremely grateful to our supervisor, Assistant Professor Md. Ali Al Mamun and co-supervisor Dr. Kamrul Hasan, PhD. Their consistence guidance providing technical information, thought, suggestions, encouragement and continuous observation make the whole matter successful one.

Again, t is our pleasure to get the cooperation and coordination from Head of the Department, Prof. M.A. Mottalib during phases of work. We are grateful to him for his constant and energetic supervision, constructive criticism and valuable advice.

We also like the opportunity to express our sincerest gratitude and heartiest thanks to Mr. Hasan Mahmud, Assistant Professor, Islamic University of Technology, for his all out cooperation and providing valuable advice about Cloud Computing.

Finally, we like to offer thanks to all who contributed us in any way during the thesis.

INDEX

Abstract

Cloud computing a variant of utility computing offers the users a large pool of computational and storage resources on demand through internet service. Maintaining QoS (Quality of Service) and SLAs (Service Level Agreement) are the most important issues in cloud computing. However, it is also important to reduce the power consumption (i.e. increase the energy efficiency), CO2 emission rate and bandwidth usage and thereby make the cloud Green. Maintaining QoS and at the same time making the cloud green is conflicting objectives. In this paper, we have proposed a SQ-Green cloud framework that ensures SLA and QoS to the users requested application. We have proposed three near-optimal scheduling policies that consolidate heterogeneity across multiple data centers for a Cloud provider. We have considered a number of energy efficiency factors such as-bandwidth, energy consumption, CO_2 emission rate, and total profit which change depending on the location, architectural design and management system.

1. Introduction

The average view of businesses regarding IT infrastructure provision is that IT service providers are too inflexible; IT is too complex and too expensive. IT does not follow the increasing pace of business dynamics, is slow in delivery and resistant to change. IT financial lifecycles are not consistent with reality. In summary: IT infrastructure needs to become a commodity that is significantly easier to acquire and use, at lower total costs of ownership. So, from this IT perspective and the advent of computing a new concept of Utility Computing comes as "**Cloud Computing**". Actually utility computing is not a new concept, but rather has quite a long history. Among the earliest references is:

"If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility... The computer utility could become the basis of a new and important industry." **- —*John McCarthy,*** speaking at the MIT Centennial in 1961.

1.1 Utility Computing

Utility computing is a service provisioning model in which a service provider makes computing resources and infrastructure management available to the customer as needed, and charges them for specific usage rather than a flat rate. Like other types of on-demand computing (such as grid computing), the utility model seeks to maximize the efficient use of resources and/or minimize associated costs.

The word ***utility*** is used to make an analogy to other services, such as electrical power, that seek to meet fluctuating customer needs, and charge for the resources based on usage rather than on a flat-rate basis. This approach, sometimes known as ***pay-per-use*** or metered services is becoming increasingly common in enterprise computing and is sometimes used for the consumer market as well, for Internet service, Web site access, file sharing, and other applications.

IBM, HP and Microsoft were early leaders in the new field of Utility Computing with their business units and researchers working on the architecture, payment and development challenges of the new computing model. Google, Amazon and others started to take the lead in 2008, as they established their own utility services for computing, storage and applications.

"***Utility computing***" has usually envisioned some form of virtualization so that the amount of storage or computing power available is considerably larger than that of a single time-sharing computer. Multiple servers are used on the "back end" to make this possible. These might be a dedicated computer cluster specifically built for the purpose of being rented out, or even an under-utilized supercomputer. The technique of running a single calculation on multiple computers is known as distributed computing.

1.2 Cloud Computing

The "***cloud***" has always been a metaphor for the Internet; in fact, cloud symbols are often used to portray the Internet on diagrams. As a virtual space that connects users from all over the globe, the Internet is like a cloud, sharing information by way of satellite networks.

Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a utility (like the electricity grid) over a network (typically the Internet).

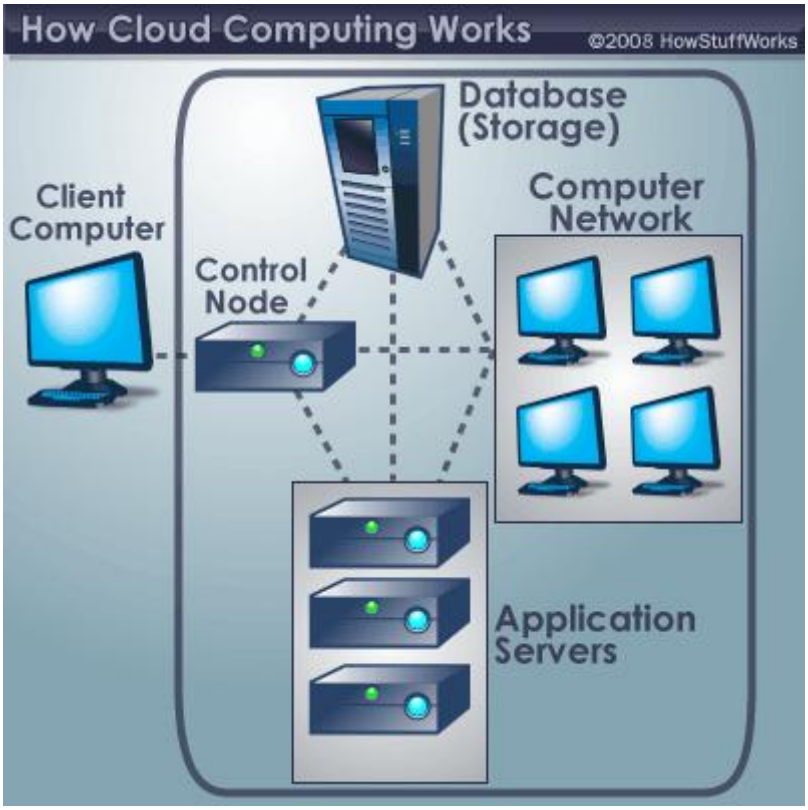

Fig 1: ***Cloud Computing***

A cloud service has three distinct characteristics that differentiate it from traditional hosting. It is sold on demand, typically by the minute or the hour; it is elastic -- a user can have as much or as little of a service as they want at any given time; and the service is fully managed by the provider (the consumer needs nothing but a personal computer and Internet access). Significant innovations in virtualization and distributed computing, as well as improved access to high-speed Internet and a weak economy, have accelerated interest in cloud computing. Cloud computing entrusts services (typically centralized) with a user's data, software and computation on a published application programming interface (API) over a network. It has considerable overlap with software as a service (SaaS).

End users access cloud based applications through a web browser or a light weight desktop or mobile app while the business software and data are stored on

servers at a remote location. Cloud application providers strive to give the same or better service and performance than if the software programs were installed locally on end-user computers.

At the foundation of cloud computing is the broader concept of infrastructure convergence (or Converged Infrastructure) and shared services. This type of data centre environment allows enterprises to get their applications up and running faster, with easier manageability and less maintenance, and enables IT to more rapidly adjust IT resources (such as servers, storage, and networking) to meet fluctuating and unpredictable business demand.

1.3 Cloud Service Models

Cloud computing providers offer their services according to three fundamental models: Infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) where IaaS is the most basic and each higher model abstracts from the details of the lower models.

Infrastructure as a Service (IaaS)

Infrastructure-as-a-Service like Amazon Web Services provides virtual server instance API) to start, stop, access and configure their virtual servers and storage. In the enterprise, cloud computing allows a company to pay for only as much capacity as is needed, and bring more online as soon as required. Because this pay-for-what-you-use model resembles the way electricity, fuel and water are consumed; it's sometimes referred to as utility computing.

Platform as a Service (PaaS)

Platform-as-a-service in the cloud is defined as a set of software and product development tools hosted on the provider's infrastructure. Developers create applications on the provider's platform over the Internet. PaaS providers may use APIs,

website portals or gateway software installed on the customer's computer. Force.com, (an outgrowth of Salesforce.com) and GoogleApps are examples of PaaS. Developers need to know that currently, there are not standards for interoperability or data portability in the cloud. Some providers will not allow software created by their customers to be moved off the provider's platform.

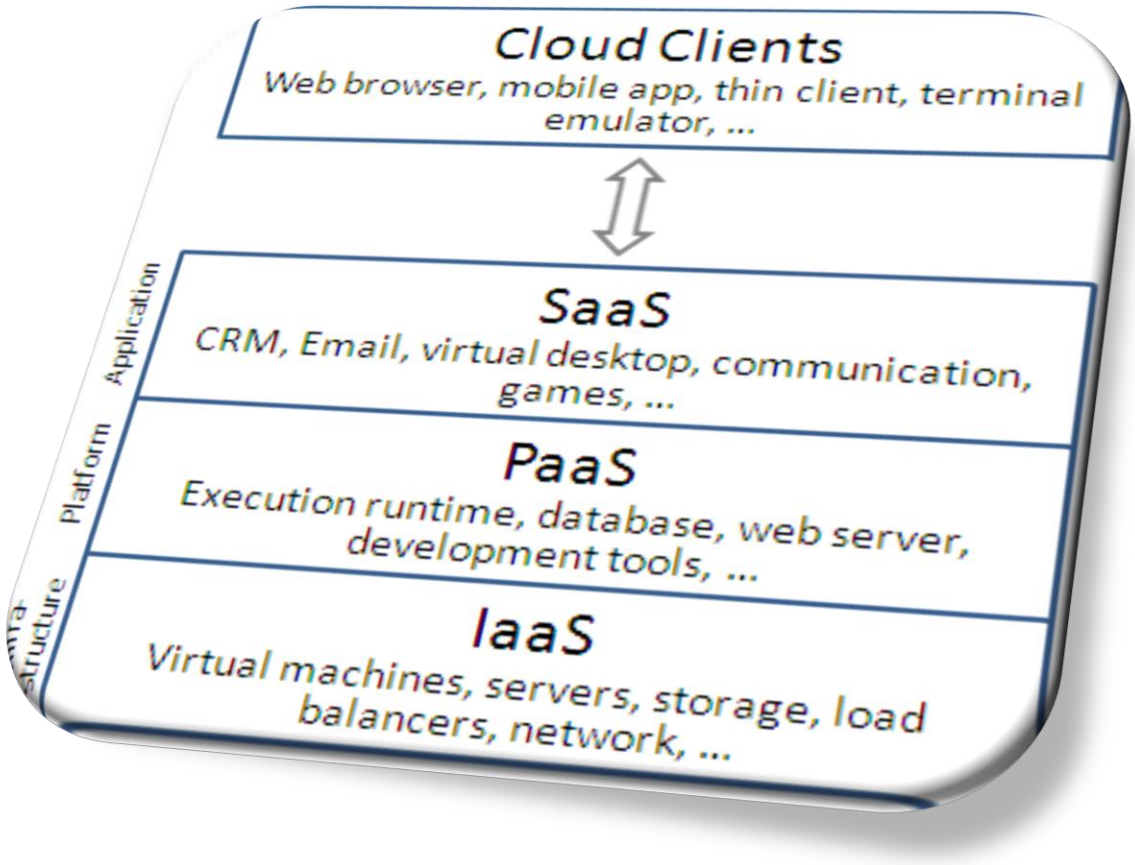

Fig 2: ***Cloud Service Models***

Software as a Service (SaaS)

In this model, cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients. The cloud users do not

manage the cloud infrastructure and platform on which the application is running. This eliminates the need to install and run the application on the cloud user's own computers simplifying maintenance and support. What makes a cloud application different from other applications is its elasticity. This can be achieved by cloning tasks onto multiple virtual machines at run-time to meet the changing work demand. Load balancers distribute the work over the set of virtual machines. This process is transparent to the cloud user who sees only a single access point. To accommodate a large number of cloud users, cloud applications can be multitenant, that is, any machine serves more than one cloud user organization. It is common to refer to special types of cloud based application software with a similar naming convention: desktop as a service, business process as a service, Test Environment as a Service, communication as a service.

1.4 Cloud Deployment Models

A cloud can have different deployment models. Cloud can be private , public or hybrid.

Public cloud:

Public cloud applications, storage, and other resources are made available to the general public by a service provider. These services are free or offered on a pay-per-use model. Generally, public cloud service providers like Microsoft and Google own and operate the infrastructure and offer access only via Internet (direct connectivity is not offered).

Private cloud:

Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third-party and hosted internally or externally. They have attracted criticism because users "still have to buy, build, and manage them"

and thus do not benefit from less hands-on management essentially "[lacking] the economic model that makes cloud computing such an intriguing concept"

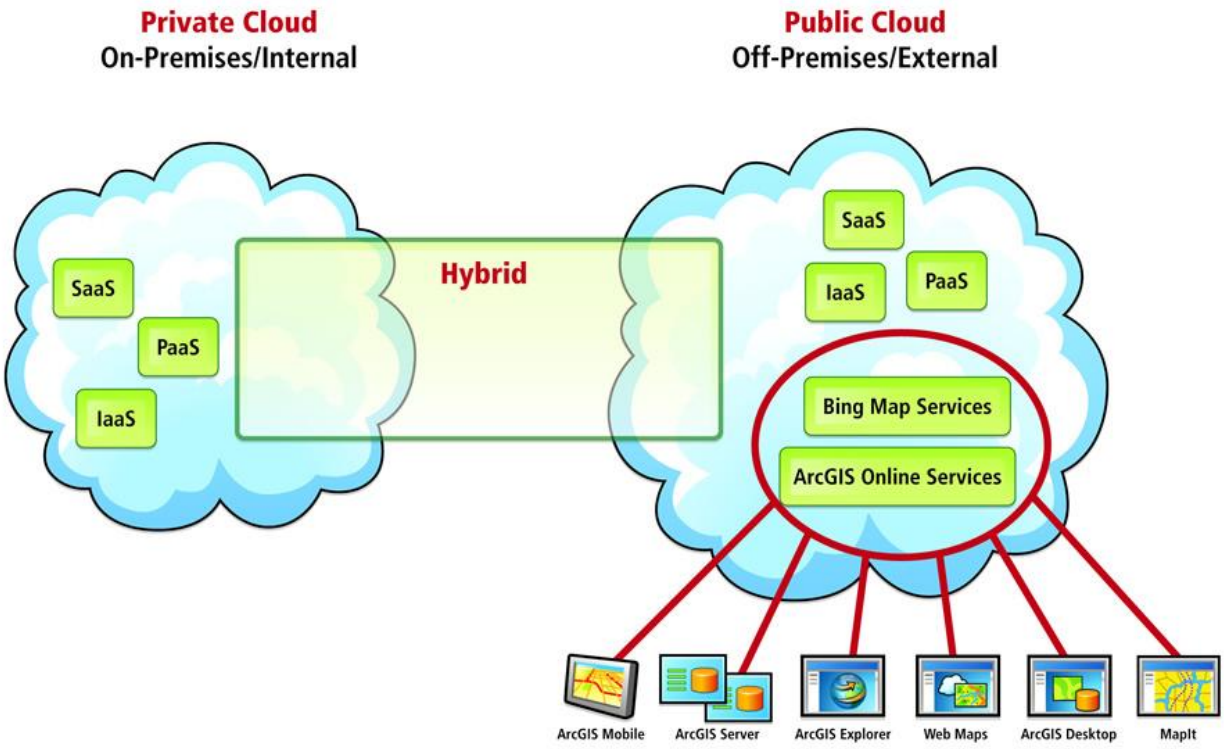

Fig 3 : Cloud deployment models

Hybrid cloud:

Hybrid cloud is a composition of two or more clouds (private, community or public) that remain unique entities but are bound together, offering the benefits of multiple deployment models.

Advantages

1. **Worldwide Access**. Cloud computing increases mobility, as you can access your documents from any device in any part of the world. For businesses, this means

that employees can work from home or on business trips, without having to carry around documents. This increases productivity and allows faster exchange of information. Employees can also work on the same document without having to be in the same place.

2. **More Storage**. In the past, memory was limited by the particular device in question. If you ran out of memory, you would need a USB drive to backup your current device. Cloud computing provides increased storage, so you won't have to worry about running out of space on your hard drive.
3. **Easy Set-Up**. You can set up a cloud computing service in a matter of minutes. Adjusting your individual settings, such as choosing a password or selecting which devices you want to connect to the network, is similarly simple. After that, you can immediately start using the resources, software, or information in question.
4. **Automatic Updates**. The cloud computing provider is responsible for making sure that updates are available – you just have to download them. This saves you time, and furthermore, you don't need to be an expert to update your device; the cloud computing provider will automatically notify you and provide you with instructions.
5. **Reduced Cost**. Cloud computing is often inexpensive. The software is already installed online, so you won't need to install it yourself. There are numerous cloud computing applications available for free, such as Dropbox, and increasing storage size and memory is affordable. If you need to pay for a cloud computing service, it is paid for incrementally on a monthly or yearly basis. By choosing a plan that has no contract, you can terminate your use of the services at any time; therefore, you only pay for the services when you need them.

Disadvantages

1. **Security**. When using a cloud computing service, you are essentially handing over your data to a third party. The fact that entity, as well as users from all over

the world, are accessing the same server can cause a security issue. Companies handling confidential information might be particularly concerned about using cloud computing, as data could possibly be harmed by viruses and other malware. That said, some servers like Google Cloud Connect come with customizable spam filtering, email encryption, and SSL enforcement for secure HTTPS access, among other security measures.

2. **Privacy**. Cloud computing comes with the risk that unauthorized users might access your information. To protect against this happening, cloud computing services offer password protection and operate on secure servers with data encryption technology.
3. **Loss of Control**. Cloud computing entities control the users. This includes not only how much you have to pay to use the service, but also what information you can store, where you can access it from, and many other factors. You depend on the provider for updates and backups. If for some reason, their server ceases to operate, you run the risk of losing all your information.
4. **Internet Reliance**. While Internet access is increasingly widespread, it is not available everywhere just yet. If the area that you are in doesn't have Internet access, you won't be able to open any of the documents you have stored in the cloud.

1.5 Green Cloud, SLAs, QoS

Green Cloud

Cloud computing is offering utility-oriented IT services to users worldwide. Based on a pay-as-you-go model, it enables hosting of pervasive applications from consumer, scientific, and business domains. However, data centers hosting Cloud applications consume huge amounts of energy, contributing to high operational costs and carbon footprints to the environment. Therefore, Green Cloud computing solutions

are needed that can not only save energy for the environment but also reduce operational costs.

SLA

A ***service-level agreement (SLA)*** is a part of a service contract where the level of service is formally defined. In practice, the term *SLA* is sometimes used to refer to the contracted delivery time (of the service) or performance. As an example, internet service providers will commonly include service level agreements within the terms of their contracts with customers to define the level(s) of service being sold in plain language terms. In this case the SLA will typically have a technical definition in terms of *mean time between failures* (MTBF), *mean time to repair* or *mean time to recovery* (MTTR); various data rates; throughput; jitter; or similar measurable details.

Some metrics that SLAs may specify include:

- What percentage of the time services will be available
- The number of users that can be served simultaneously
- Specific performance benchmarks to which actual performance will be periodically compared
- The schedule for notification in advance of network changes that may affect users
- Help desk response time for various classes of problems

The underlying benefit of cloud computing is shared resources, which is supported by the underlying nature of a shared infrastructure environment. Thus, service level agreements span across the cloud and are offered by service providers as a service based agreement rather than a customer based agreement. Measuring, monitoring and reporting on cloud performance is based upon an end user experience or the end users ability to consume resources. The downside of cloud computing, relative to SLAs, is the difficultly in determining root cause for service interruptions due to the complex nature

of the environment. Actually in Cloud Computing environment, SLA refers to Bandwidth utilization, Resource management, CO_2 emission rate, delay time etc.

QoS

Quality of Service (QoS) for networks is an industry-wide set of standards and mechanisms for ensuring high-quality performance for critical applications. By using QoS mechanisms, network administrators can use existing resources efficiently and ensure the required level of service without reactively expanding or over-provisioning their networks.

Traditionally, the concept of quality in networks meant that all network traffic was treated equally. The result was that all network traffic received the network's best effort, with no guarantees for reliability, delay, variation in delay, or other performance characteristics. With best-effort delivery service, however, a single bandwidth-intensive application can result in poor or unacceptable performance for all applications. The QoS concept of quality is one in which the requirements of some applications and users are more critical than others, which means that some traffic needs preferential treatment.

2. Related Work

Most recent works have been done in energy efficiency of the cloud computing. In [1] they worked on scheduling virtual machines in a compute cluster to reduce power consumption via the technique of Dynamic Voltage Frequency Scaling (DVFS).They have focused on implementing a power-aware scheduling algorithm for high performance cluster computing where virtual machines are dynamically provided for executing cluster jobs. They proposed a new cluster scheduling algorithm to minimize the processor power dissipating by scaling down processor frequencies without drastically increasing the overall virtual machine execution time. This algorithm is implemented in a simulator for DFVS-enabled clusters and an experimental multi-core

cluster. Performance evaluation and discussion are also provided. The aim is for the scheduling algorithm to be deployed in various compute centers such as clusters within Grid Computing deployments. They have provided some models for simulation environment.

Power Aware Virtualization Algorithm

There are a few rules of thumb to build a scheduling algorithm which schedules virtual machines in a cluster while minimizing the power consumption:

1) Minimize the processor supply voltage by scaling down the processor frequency.

2) Schedule virtual machines to PEs with low voltages and try not to scale PE to high voltages.

As shown in Figure 4 incoming virtual machine requests arrive at the cluster and are sorted in a queue. A scheduling algorithm runs as a daemon in a cluster with a predefined schedule interval, INTERVAL. During the period of scheduling interval, incoming virtual machines arrive at the scheduler and will be scheduled at the next schedule round Fj , 1 j ≤ J is a set of PEs that run in the operating point of opj . Firstly the Algorithm sets F1; F2,FJ with empty sets (line 1). The wall clock time, t, is initialized with 0. Algorithm initializes all PEs as follows :

1) set all PEs running to the lowest voltage and processor speed, s_{min};

2) pe_k.sa is defined as the available processor speed if the processor does not change its operating point. Since no virtual machine are initially scheduled, pe_k:sa is initialized with s_{min};

3)pe_k.rs is the available PE processor speed when pe_k is operated to a highest level voltage from current voltage level. pe_k.rs is initialized with s_{max}.

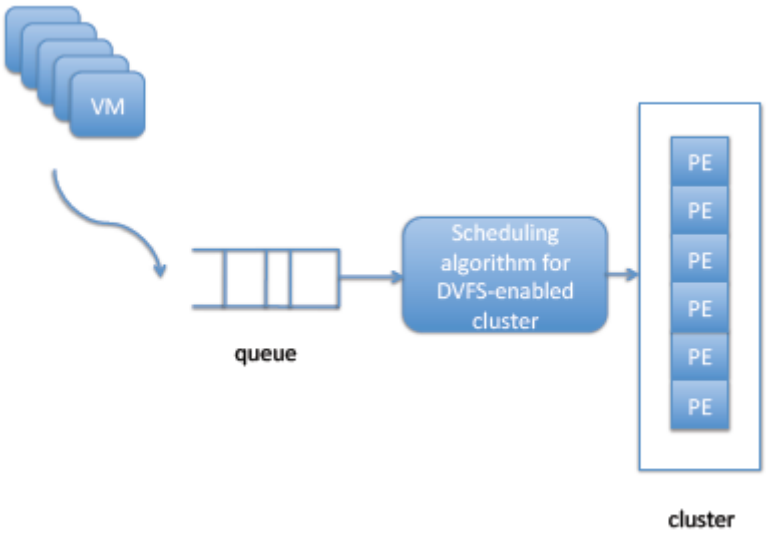

Fig 4: Working scenario of a DVFS enabled cluster

The scheduler iterates with a predefined interval value, INTERVAL, from starting time 0. In each scheduling round, the scheduler firstly levels down power profiles of PEs. The reason is that some virtual machines might finish its execution during the last scheduling round and some PEs are no longer needed to be operated in high voltages. Then the power profiles of some PEs are leveled down. Then the scheduler places incoming virtual machines in the queue to PEs while minimizing the power consumption.

Recent works are done in Energy efficient scheduling of user requested service and applications. In [2] , they proposed near-optimal scheduling policies that exploits heterogeneity across multiple data centers for a Cloud provider. We consider a number of energy efficiency factors such as energy cost, carbon emission rate, workload, and CPU power efficiency which changes across different data center depending on their location, architectural design, and management system. Our carbon/energy based scheduling policies are able to achieve on average up to 30% of energy savings in

comparison to profit based scheduling policies leading to higher profit and less carbon emissions. The key contributions of their work are:
(1) A novel mathematical model for energy efficiency based on various contributing factors such as energy cost, CO2 emission rate, HPC workload, and CPU power efficiency.
(2) The near-optimal energy-efficient scheduling policies which not only minimizes the CO2 emissions and maximizes the profit of the Cloud provider, but also can be readily implemented without much infrastructure changes such as the relocation of existing data centers.

Polices for Mapping Jobs to Resource

We have designed the following meta-scheduling mapping/allocation policies depending on the objective of the cloud provider:

Minimizing Carbon Emission

The following policies optimize the global carbon emission by all cloud sites while keeping number of missed deadlines low.

- **Greedy Minimum Carbon Emission (GMCE)**: In this greedy based policy, all user applications are ordered by their deadline (earliest first),while the data centers at different cloud sites are sorted in decreasing order of their Carbon efficiency Then, the meta-scheduler assigns applications to a cloud site according to this ordering.
- **Minimum Carbon Emission Minimum Carbon Emission (MCE-MCE)**:

MCE-MCE is based on the general concept of the Min-Min idea [3]. The Min-Min type heuristic performed very well in previous studies of different environments (e.g., [4]). In, MCE-MCE, the meta-scheduler finds the best data center or cloud sites for all applications that are considered, and then among these applications/sites pairs, the meta-scheduler selects the best pair to map first. To determine which application/cloud

site pair is the best, we have used the carbon emission resulted due to execution of the application j on the cloud site i i.e. (CO2E)ij as the fitness value. Thus, MCE-MCE includes the following steps:

Step 1: For each application that is present in the meta-scheduler to schedule, find the cloud site for which carbon emission is the minimum, i.e. minimum fitness value (the first MCE) among all cloud sites which can complete the application by its deadline. If there is no cloud site where the application can be completed by its deadline, then the application is dropped or removed from the list of applications which are to be mapped.

Step 2: Among all the application/cloud site pairs found in Step 1, find the pair that resulted in minimum carbon emission, i.e. minimum fitness value (the second MCE). Then, map the application to the resource site, and remove the application from the list of applications which are to be mapped.

Step 3: Update the available slots from the resource sites.

Step 4: Do steps 1 to 3 until all applications are mapped.

Minimizing Carbon Emission and Maximizing Profit (MCE-MP)

In this policy, the objective of the meta-scheduler is to minimize the total carbon emission while maximizing the total profit of the Cloud provider. Thus, this policy handles the tradeoff between profit and carbon emission which may be conflicting. This policy is very similar to MCE-MCE and MP-MP except fitness functions (carbon emission i.e. (CO2E)ij and profit (Prof)ij) for each step of finding best" application/cloud site pair. Thus, the MCE-MP policy include following steps:

Step 1: For each of the applications that are present in meta-scheduler for execution, find the cloud site for which the carbon emission is minimum, i.e. minimum (CO2E)ij (the first MCE) among all the cloud sites that can complete the application by its deadline. If there is no such cloud site where the application can be completed by its

deadline, then the application is removed from the list of applications that are to be mapped.

Step 2: Among all the application/cloud site pairs found in Step 1, find the pair that resulted in maximum profit i.e. maximum (Prof)ij (the second MP). Then, map the application to the cloud site and remove the application from the list of applications that are to be mapped.

Step 3: Update the available slots from the cloud sites.

Step 4: Do steps 1 to 3 until all applications are mapped.

In work [5], researchers focused on power aware scheduling technique using VMM (Virtual Machine Monitor) to migrate the virtual machine from one core to another and thus save a considerable amount of energy. Heterogeneous cluster systems also discussed with power consumption analysis. In [6],in heterogeneous clusters if any request comes, a heuristic bin-packing algorithm is used to provide the service using minimum voltage required. Rajkumar Vuiyan [7] proposed near- optimal scheduling policies that exploits heterogeneity across multiple data centers for a Cloud provider and their carbon/energy based scheduling policies are able to achieve on average up to 30% of energy savings.

Nathuji and Schwan [8] combined the power consumption and virtualization techniques to reduce the power consumption in web workloads.

So, in the above mentioned contributions they have worked on a single server or data center in a single location. Since our proposed scheduling policy improves the energy efficiency across data centers in multiple locations with different carbon emission rates, it can be used in conjunction with these solutions to utilize any energy efficiency already implemented in a single location.

In [9], Rajkumar Buyya proposed MBFD (Modified Best Fit Decreasing) algorithm to sort all VMs in de-creasing order of their current CPU utilizations, and allocate each VM to a host that provides the least increase of power consumption due to this allocation.

Not so many research work studies bandwidth utilization and energy efficiency in an economic cost perspective. R. Ge et al. [10] followed an economic approach where each service 'bid' for the shared server and followed an energy cost approach to maximize the profit in a single data center. None of these research work studies the critical relationship between bandwidth usages per application request in cost effective way. On the other hand, we have proposed a bandwidth utilizer which maps the applications to ensure how to utilize the bandwidth in a cost-effective manner.

2.1 The Need for QoS Green Cloud

Managing application performance and users SLAs is a challenging issue in cloud environment. Application performance is changed due to the presence of other virtual machine in cloud data centers. Besides, other applications cannot be controlled by the users. So, Cloud providers must ensure resource availability, capacity management, bandwidth utilization, energy efficiency, delay time etc.

Performance interference effects impede the normal functionality of the clouds. There are two implications on why there is need for QoS-Clouds-

First, when placing VMs to maximize resource utilization and efficiency it is necessary to realize resource usage and demand for the user submitted application. This information is needed when user submitted VMs and applications do not have the sufficient resources the "Head room" is used to utilize the unused resources to meet the demand before the delay time. So, The first requirement of the QoS-cloud is to reallocate the resources from the "head room" due to interference effects to maintain the SLAs.

Second, to complete the request on the cloud server users should provide some states or factors that should be achieved for the required SLAs. So, the second requirement is to provide some Q-states to define the required SLAs.

3. Proposed Architecture and Policy

3.1 SQ-Green Cloud Architecture

Architectural framework

Clouds aim to drive the design of the next generation data centers by architecting them as networks of virtual services (hardware, database, user-interface, application logic) so that users can access and deploy applications from anywhere in the world on demand at competitive costs de-pending on their QoS requirements [11]. Fig 4 [12] shows an ideal SQ-green cloud infrastructure which provides the four entities-

Cloud Users:

Cloud users submit their request from anywhere in the world. Different types of service requests are made which cause different workloads in the service providers.

SQ Green Cloud Provider:

Interface between the cloud users and the cloud providers. It consists of the following parts-

1) Green negotiator:

Cloud providers negotiate with the users about the service, user SLAs, Price and cost .

2) Bandwidth utilizer:

Measures bandwidth requirement needed by the users requested application. If the user requested bandwidth exceeds the minimum bandwidth level then server stops the

respective application and provide service to the next request. The stopped request will be given service later.

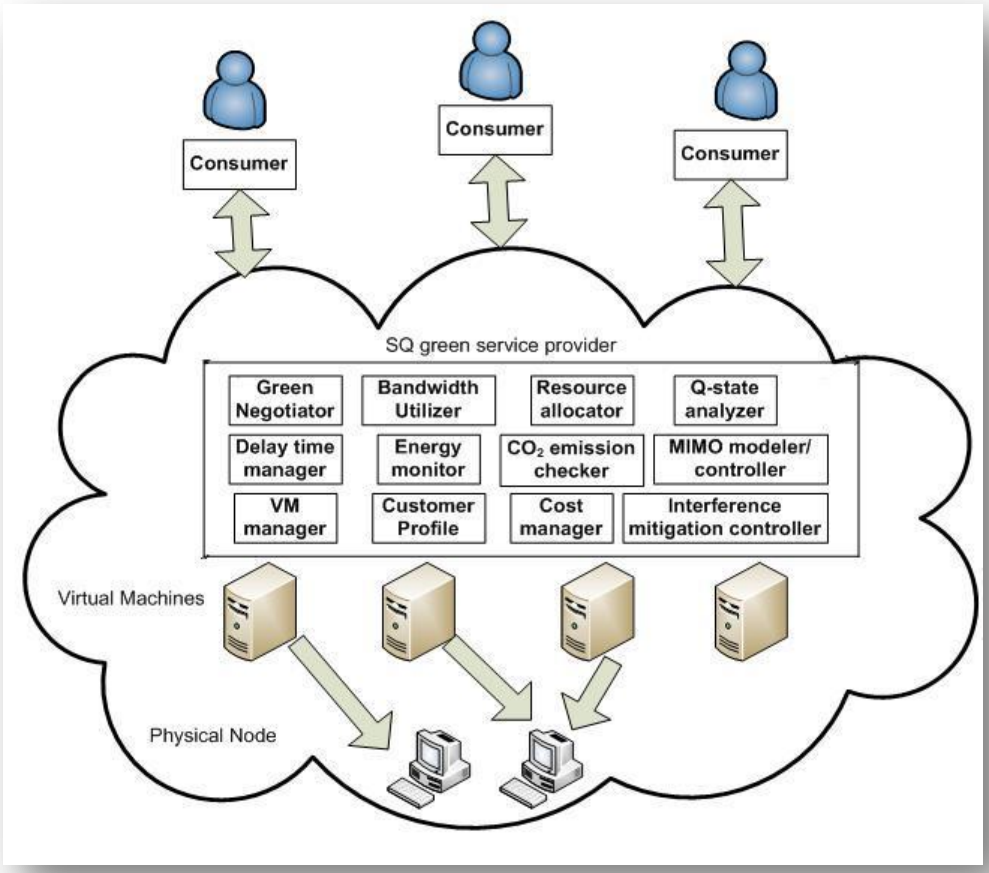

Fig 4: SQ-Green Cloud Architecture

3) Resource allocator:

It allocates resources needed to give the service to the user requests. If the current resources cannot fulfill users desired SLAs then "head room" resources are used to provide the service.

4) Q-state analyzer:

Users define the required SLAs needed in their application request. Q-state analyzer analyzes the User submitted SLAs and match the SLAs when delivering services to a particular request.

5) Delay Time Manager:

Each request of the user has a predefined time within which it should be given service. Delay Time Manager measures the time and let the provider know to complete giving service to the request.

6) Energy Monitor:

Observes energy consumption caused by VMs and physical machines and provides this information to the VM manager to make energy-efficient resource allocation decisions.

7) Interference Mitigation Controller:

As Performance interference happens deliberately in the cloud providers side described earlier, Interference Mitigation Controller lessens the performance degradation realizing Q-states. SQ- Green Cloud provider determines the placement of the user requests based on resource requirements of the workloads and other constraints for security and reliability purposes [13].

8) MIMO controller:

Multiple Input Multiple Output (MIMO) controller learns its input and output from control feedback through online and acts as a prediction tool for optimization. It captures interference relationships between applications. It is a closed loop controller helps resource allocator tuning allocation of resources dynamically to achieve desired level of QoS.

9) VM Manager:

Keep the track of every virtual machine in the server's side. It is also responsible for Provisioning VMs in the physical nodes.

10) Customer profiler:

Gathers information about customers so that important customers can be categorized based on their need.

11) Cost manager:

It computes the requested service charge and informs the customers at the same time.

12) CO_2 emission checker:

It checks the CO2 emission in providing service to the request. If the CO2 emission is higher and also affects the user SLAs then it sends the information to the Green Negotiator to decide what to do.

Virtual Machines:

One or more virtual machines are installed in a single physical node. Each virtual machine will provide service to the corresponding user's request once at a time. Each virtual machine can be provisioned or migrated from one physical node to another by VM manager.

Physical Nodes:

It refers to the server's computers. Each computer has multi-core processor. Each node meet user's submitted request with minimum CPU frequency.

3.2 Power Consumption Algorithm (Case Based)

Algorithm 1: Power Consumption Algorithm

Input: hostList, vmList, jobQueue, dcList, resourceList, appList

Output: allocation of app , allocation of vm

appMigration ()

repeat

foreach dataCenter *in* dcList **do**

calTime ← MAX

CPUF ← *sort* (availableCPUF)

allocatedapp ← NULL

```
        endfor
            if allocatedapp ≠ NULL then
           time ← estimateTime (allocatedapp, dataCenter)
            endif
              if time < calTime then
                 allocate CPUF  to  allocatedapp
              else  migrate app
             resourceInfo ← update ( resourceList )
        until  all applications are mapped
        return allocation of app
       foreach vm in vmList do
            vmPower ← MAX
            allocatedHost ← NULL
   endfor
foreach host in hostList do
              if (hostResource < vmResource) then
                  power←estimatepower(host,vm)
                if power < vmPower then
                   allocatedHost ←  host
                 vmPower ← power
               else migrate job to vm with min(BW,PW)
              end
             endif
               if (hostResource > vmResource) then
                 power ← estimatepower (host,vm)
                 if power > vmPower then
                     vmPower ← power
                   migrate job to vm with min(BW,PW)
```

endif

endif

endfor

return allocation of vm

We have proposed three VM migration policies for the user requested jobs and their SLAs.

Case1: If there are no jobs in queue then no need of Job migration.

Case2: In case of few jobs in two/ more processors check the current power demand and predict the future demand accordingly based on the SLAs and program history profile.

Migrate all the jobs to the low core processor (s) given the migrated jobs will be completed without further migration.

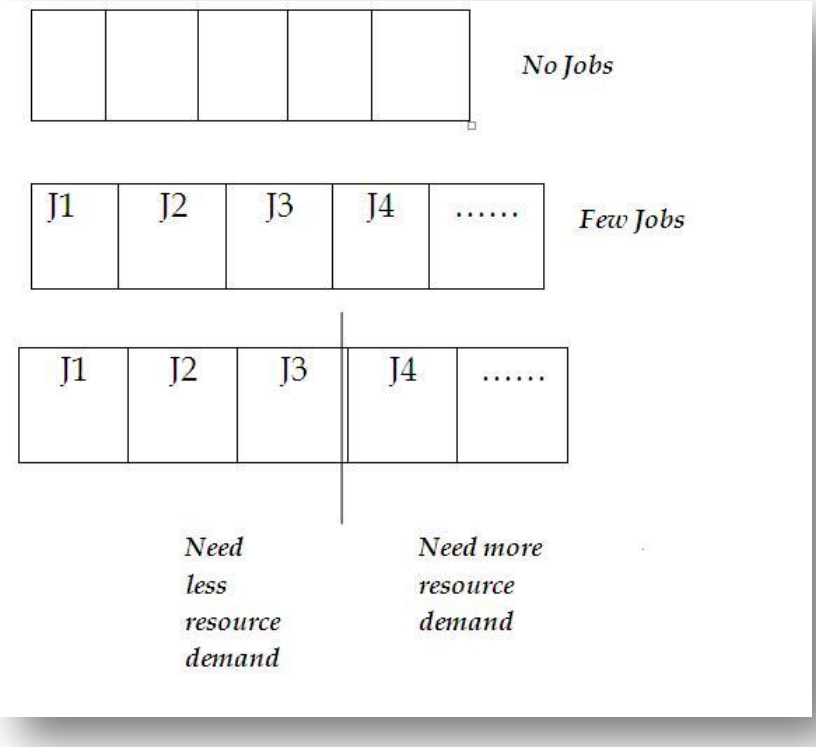

Fig 5 : Power Consumption policy

Case3: *If* some of the jobs have less power demand and some need more then the following policies should be followed:

Policy1 (Jobs having more power demand):

Migrate all the jobs to the low core processor (s) given. If the power demand becomes high then the processor where the jobs are migrated may not fulfill the user's current power demand. Then only one "**Thrashing**" occurs. After thrashing don't migrate the jobs to the low end processor (s) rather we migrate the jobs to the high core processor (s) so that user jobs can fully completed there without further migration.

Policy2 (Jobs having less power demand): Check whether the current VM(s) can fulfill the power demand. If it can then no need of job migration otherwise migrate all the jobs to the low core processor (s) given the migrated jobs will be completed without further migration.

3.3 Bandwidth Utilization Algorithm:

BUP is based on the general concept of the Min-Min idea [15]. The Min-Min type heuristic performed well in the Optimization purpose (e.g., [16]). In our BUP, the bandwidth utilizer finds the best data center for all applications that are considered, and then among these applications/data center pairs, the bandwidth utilizer selects the best pair to map first. To determine which application/data center pair is the best, we have used the bandwidth used by the user application j on the data center i i.e. $(BW)_{ij}$ as the fitness value. Thus,
BUP includes the following steps:

Step 1: For each user requested application the bandwidth utilizer finds the data center for which bandwidth is minimum, i.e. minimum fitness value $(BW)_{ij}$ among all the data centers which can complete the application by its deadline. If there is no data center which can complete the job within the deadline then the application is removed from the list of applications are to be mapped.

Step 2: Among all the application / data center pair found in step 1 find the pair which has minimum bandwidth, i.e. minimum fitness value $(BW)_{ij}$. Then map the application to that data center and remove it from the list of applications are to be mapped.

Step3: Update the available information in cloud data center

Step4: Do steps 1 to 3 until all applications are mapped.

Algorithm 2: Bandwidth Utilization Algorithm

Input: dcList, resourceList, appList

Output: allocation of app

```
Repeat
    foreach dataCenter in dcList do
        calTime ← MAX
        BW ← sort (availableBW)
        allocatedapp ← NULL
        availableBW ← min (BW)
    endfor

        if allocatedapp ≠ NULL then
            time ← estimateTime (allocatedapp, dataCenter)
        endif

         if  time < calTime then
                allocate  availableBW  to  allocatedapp
            else  migrate app
          resourceInfo ← update ( resourceList )
```

until *all applications are mapped*

return allocation of app

4. Simulation

For the simulation environment we have created our own simulator using JAVA. We have done the simulation to evaluate the efficiency of our job migration and Power Consumption algorithm working in the complicated, real-condition liked federated cloud environment. In the simulation, we have defined a cloud consists 50 servers or datacenters , each datacenter has multiple number of Power Elements(PEs) or CPUs which in term hold multiple Virtual Machine (VM) . We have also defined job. For the purpose of simulation we have defined some parameters bandwidth, memory storage, power, delay time which varies with the job and servers.

In this simulation, we set the interval of monitoring as 2 minutes. But we have collected our data based on the time defined in the user request.

We managed to do 50 times experiments based on the random data.

Results obtained by the implemented model have been compared with results obtained in the Dynamic Voltage Frequency Scaling (DVFS) Algorithm which is the most popular algorithm for the optimization of cloud power efficiency. This algorithm computes the best possible optimization case by case, spreading the whole network load to the most energy efficient machines, and using the 100% of their resources. Therefore, it is used as an upper limit to understand the performance of our proposed algorithm.

4.1 Experimental Scenarios

We examine various experimental scenarios to evaluate the performance of our algorithms:

- ***Power consumption***: calculated as the sum of the energy class indicators of each server.
- ***Power saving percentage***: reduction of power consumption with respect to the initial situation.
- ***Delay time:*** calculate the job completion time using our PCA algorithm and compare with the DVFS.

The remainder of this section shows the results obtained according to the following simulation plan. In the first simulation we report the results of some experiments made to understand the behavior of the algorithm when varying the Power Element (PEs) in each server. In the following simulations we evaluated the algorithm's performance using the power optimization policy and shows how it varies with the DVFS algorithm. After several runs we have seen that the results obtained in our experiment improve 9.56% than the DVFS.

4.2 Performance Analysis

Bandwidth Utilization:

We simulate the bandwidth utilization algorithm with 10, 20, 30, 40, and 50 compute nodes. Multiple virtual machines are generated in the job module. We simulate 100, 200, 300, 400 and 500 virtual machines. To satisfy the cluster module described above, the Virtual machine resource requirements are fixed in the range of 1000 MHz – 5000 MHz and required execution times are randomly generated in the range of 10 time unit – 50 time unit.

Table 1: Bandwidth utilization vs. Power element number

PE number	VM=100	VM=200	VM=300	VM=400	VM=500
10	3700ms	3900ms	4150ms	4450ms	4800ms
20	3500ms	3700ms	4050ms	4150ms	4500ms
30	3300ms	3500ms	3900ms	4100ms	4300ms
40	3200ms	3300ms	3700ms	4000ms	4200ms
50	3100ms	3400ms	3600ms	3800ms	4000ms

Figure 6 shows simulation results for Bandwidth utilization of the power element s. The X-axis is the number of PEs and the Y-axis is the normalized bandwidth consumption. The base for normalization is the bandwidth consumption when all PEs are operated in the highest voltage. Looking at this data, we can make the following observations:

- Observation 1: The Bandwidth Utilization algorithm can reduce bandwidth consumption.
- Observation 2: In case that the number of PEs is fixed, the bandwidth consumption increases as the number of incoming virtual machines increases.
- Observation 3: In case that the number of incoming virtual machine is fixed, the bandwidth consumption decreases as the number of PEs increases.

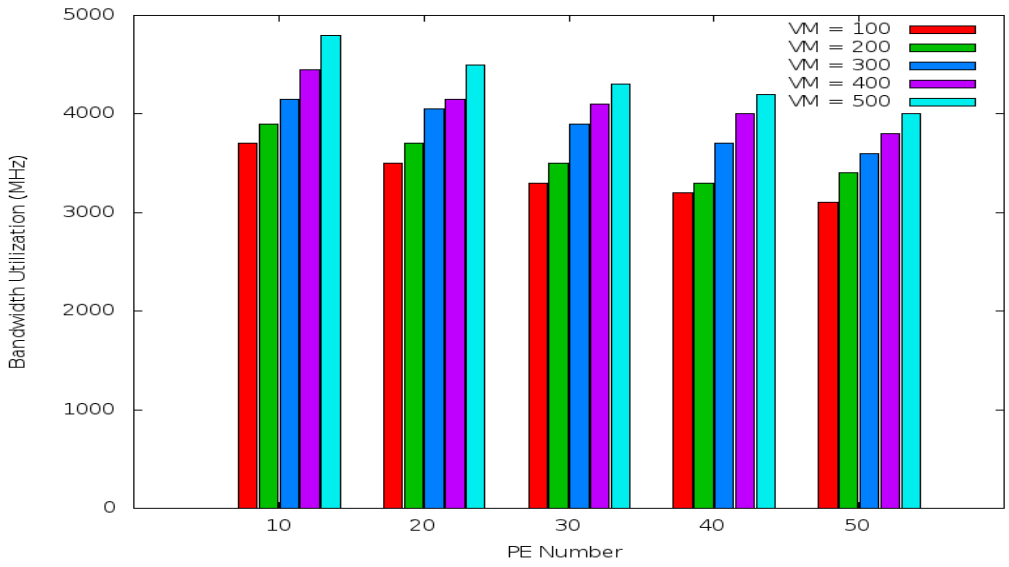

Fig 6: Bandwidth Utilization vs. Power Element Number

We interpret the simulated behavior as follows. For Observation1, in the operation of our Bandwidth Utilization Policy some PEs operate with lower voltages. Thus, compared with a fully utilized PEs using the highest voltages, less power is consumed. Observation 2 can be interpreted as when more virtual machines arrive in a cluster, the PEs of the cluster are forced to consume more bandwidth to provide more processing capacities, thus requiring more bandwidth consumption.

On the contrary, when more PEs are available for a fixed number of incoming virtual machines, the PEs consume lower bandwidth since enough PEs are provided leading to Observation 3 where less bandwidth utilization is observed.

Power Consumption:

Power consumption factor is very much important for maintaining green property in cloud architecture. Power consumption actually means total power consumed by virtual machine while processing jobs. The major goal of our approach is to reduce power consumption in the data center while meeting performance requirements. The power consumption of data centers needs to be minimized to reduce operating costs and avoid system overheating. Various power- efficient performance management

strategies have been proposed based on dynamic voltage and frequency scaling (DVFS). Virtualization technologies have also made it possible to consolidate multiple virtual machines (VMs) on to a smaller number of active physical servers for even greater power savings, but at the cost of a higher overhead. Our strategy is to compare our algorithmic result with DVFS one and evaluate percentage of performance.

At first we have created several data center with distinctive virtual machines. Each virtual machine holds its own resource parameter. We have created several jobs and then allocated to those virtual machines. Every job has own resource requirement. According to that, virtual machines receive their corresponding jobs. In DVFS technique, each virtual machine start to process their job and after completion it stops working. As a result power consumed by virtual machine is less. But our PCA (Power Consumption Algorithm) shows better performance than DVFS. After simulation whatever result we have got is given in Table 2. We run simulation for different number of jobs like 4, 8, 12, 16. Power consumed by virtual machines is also fixed like 0-250 watt having difference of 50 watt.

After generating result we plot both DVFS and PCA value in GNUPlot for making a graph showing on fig. 7. From graph, we can predict that our proposed algorithm provide better result. We put number of jobs in X-axis and power consumption value in Y-axis.

Table 2: Calculating power consumed value for both DVFS and PCA

Number of Jobs	DVFS level (Watt)	PCA level(Watt)
4	45	42
8	70	51.75
12	95	90.25
16	187	176.97

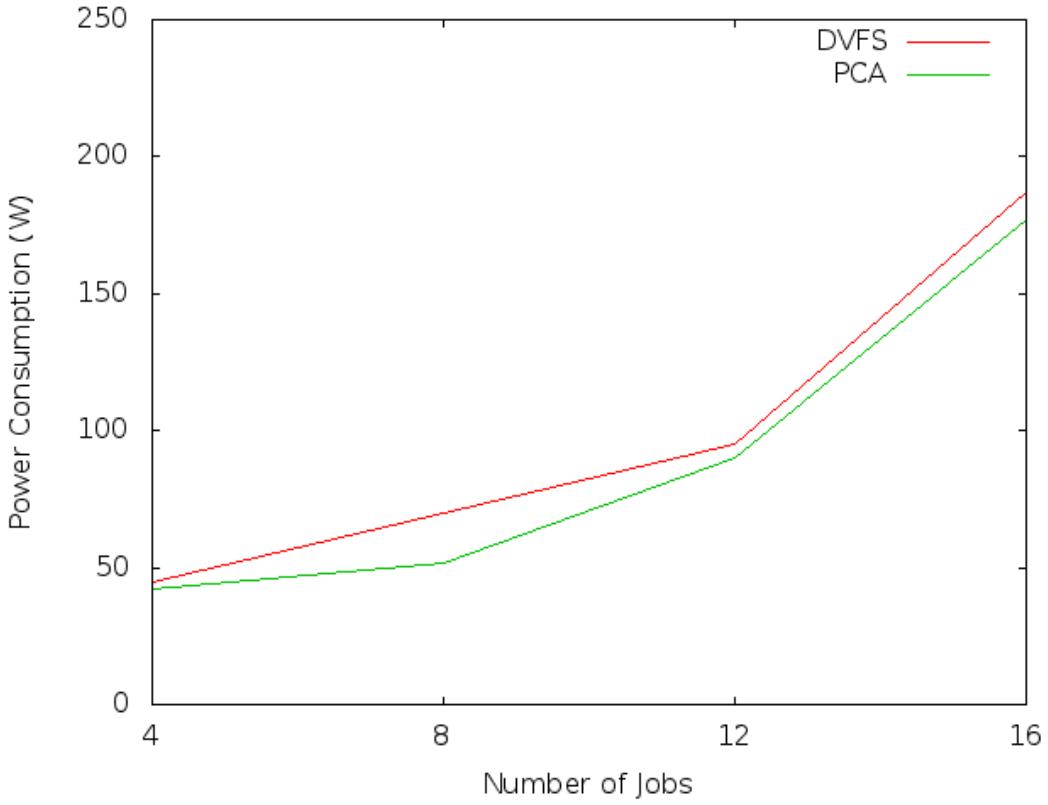

Fig 7: Power consumption with respective different jobs

Total power consumed by DVFS is 397 watt and for PCA it is 360.97 watt. Difference is 36.03 watt. In percentage, PCA value is better than DVFS for 9.08%. We can at least assure slight performance than DVFS. Less power consumption means green property.

Calculating delay time considering various numbers of jobs

For this experiment, we schedule all virtual machines to compute actual delay time. At first we take delay time value for DVFS and second we consider PCA (Power Consumption Algorithm). For both case we considered jobs which are coming from different users, processed in low core processor with minimum delay time. Also those jobs consume less power of data center. Those jobs are processed in different virtual machine having minimum CPU frequency. After that we have taken delay time value for DVFS and PCA both in higher level, means jobs require more time and processed in

high core processor. Actually delay time depicts both required time given by user for processing and also job completion time. This time-value is calculated for checking QoS (Quality of Service). Our proposal is, introducing QoS enabled green cloud. To ensure better quality that means QoS, our objective should to get less delay time. We can assure that some improvement is brought by us. Corresponding approval and simulation process are described later. Now come for term DVFS (Dynamic voltage and frequency scaling), which is an efficient technology to control the processor power consumption. With aide of support technologies such as Intel SpeedStep and AMD PowerNow!,modern processors can be operated in several frequencies with different supply voltages. Whatever value we have gathered from simulation are given below-

Table 3: Different delay time in DVFS and PCA level considering low resource demand for various numbers of jobs

Number of Jobs	Delay Time (ms)	
	DVFS (for low resource demand)	PCA (for low resource demand)
2	12	10
4	14.5	12
6	15	14
8	22.25	22
10	45.40	40

To get above values we take bunch of jobs like 2, 4, 6, 8, 10 numbers. Assign those jobs to low core processor for processing purpose. That processor has minimum CPU frequency. Every job has own time period which is defined earlier. We keep resource demand less in every job like bandwidth requirement, memory usage capacity etc. Now we run simulation and get the result, declared in Table 3 earlier. From Fig 8, we have

generated Table 3. We can see total time for DVFS is 109.15ms and for PCA is 98ms. Time difference is 11.15ms. So from this calculation we can say our produced PCA provide 10.21% betterment than DVFS.

Table 4: Different delay time in DVFS and PCA considering high resource demand for various numbers of jobs

Number of Jobs	Delay Time (ms)	
	DVFS (for high resource demand)	PCA (for high resource demand)
2	32	32
4	40	35
6	43	37
8	28	21
10	21	16

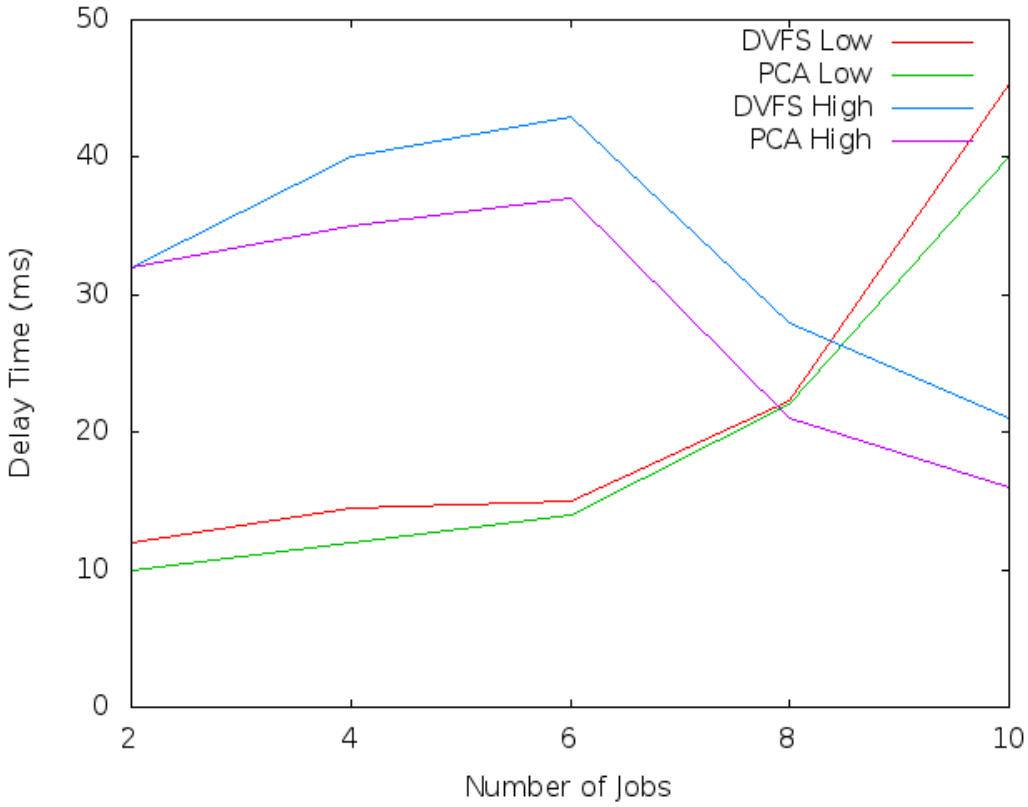

Fig 8: Delay time graph for different jobs maintaining QoS

For low level segment graph is like going upward. That's because we design job allocation in an increment order. Shape can be changed if resource assigning is random.

Now for high level case, we allocate jobs to all high level processor having minimum CPU frequency. Here we are considering random resource allocation like different bandwidth requirement, memory capacity etc. After that, simulation is done and we get above graph of Fig 8. Values are given in Table 4. We can see total time for DVFS is 164ms and for PCA is 141ms. Time difference is 23ms. So from this calculation we can say our PCA provide 14.02% betterment than DVFS. In an average performance is 12.115% better. It's enough for proving that our proposal is satisfying QoS. Because according to the policy, every virtual machine process jobs within its assigned time period. We can ensure 12.115% better quality to the user than DVFS policy.

5. Conclusion and Future Work

The usage of energy has become a major concern since the price of electricity has increased dramatically. Especially, Cloud providers need a high amount of electricity to run and maintain their computer resources in order to provide the best service level for the customer. Although this importance has been emphasized in a lot of research literature, the combined approach of analyzing the profit and energy sustainability in the resource allocation process has not been taken into consideration.

The goal of this paper is to outline how managing resource allocation across multiple locations can have an impact on the energy cost of a provider. The overall meta-scheduling problem is described as an optimization problem with dual objective functions. We have tried to improve our power based and bandwidth based algorithm to improve the overall performance of the cloud as well as ensuring the QoS of the users with the cloud providers. We have achieved a better performance than the existing one. Hope it can be more improved in future with more extensive research in this field.

6. References

[1] Gregor von Laszewskiy, Lizhe Wangz, Andrew J. Youngez, Xi Hez" Power-Aware Scheduling of Virtual Machines in DVFS-enabled Clusters" ,*Intel Technology Journal, vol. 6, no. 1, pp. 4–15, 2002.*

[2,7,15] Saurabh Kumar Garg, Chee Shin Yeo , Arun Anandasivam, Rajkumar Buyya" Energy-Efficient Scheduling of HPC Applications in Cloud Computing Environments" ,*in Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing, Honolulu, USA, 2008.*

[3]O. Ibarra, C. Kim, "Heuristic Algorithms for Scheduling Independent Tasks on Nonidentical Processors," *Journal of the ACM (JACM) 24 (2) (1977) 280-289.*

[4] T. Braun, H. Siegel, N. Beck, L. Boloni, M. Maheswaran, A. Reuther,J. Robertson, M. Theys, B. Yao, D. Hensgen, et al., "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems", *Journal of Parallel and Distributed Computing* 61 (6) (2001) 810837.

[5] S. Srikantaiah, A. Kansal, and F. Zhao. "Energy aware consolidation for cloud computing ", *In Workshop on Power Aware Computing and Systems (HotPower '08). San Diego, USA, December 2008.*

[6,11] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, I. Brandic," Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility ", *Future Generation Computer Systems 25 (6) (2009) 599{616}.*

[8] R. Nathuji, K. Schwan," Virtualpower: coordinated power management in virtualized enterprise systems ", *in: SOSP '07: Proceedings of 21st ACM SIGOPS 31 symposium on Operating systems principles, ACM, New York, NY, USA, 2007, pp. 265-278.*

[9] Anton Beloglazova,_, Jemal Abawajyb, Rajkumar Buyyaa " Energy-Aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing", *submitted to Future Gen- eration Computer Systems,2010.*

[10] R. Ge, X. Feng, and K. Cameron, "Performance-constrained distribute ddvs scheduling for scientific applications on power-aware clusters," *in Proceedings of the 2005 ACM/IEEE conference on Supercomputing. IEEE Computer Society Washington, DC, USA, 2005.*

[12]S. Srikantaiah, A. Kansal, and F. Zhao. " Energy aware consolidation for cloud computing", *In Workshop on Power Aware Computing and Systems (HotPower '08). San Diego, USA, December 2008.*

[13] Ripal Nathuji and Aman Kansal, Alireza Ghaffarkhah " Q-Clouds: Managing Performance Interference Effects for QoS-Aware Clouds", *in Proceedings of the 2010 ACM conference on Ditributed Systems, Paris, France, 2010.*

[14] R. Buyya, A. Beloglazov, J. Abawajy, " Energy-e_cient management of data center resources for cloud computing: A vision, architectural elements, and open challenges ", *in: Proceedings of the 2010 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2010), LasVegas, USA, 2010.*

[16] O.Ibara, C. Kim ," Heuristics Algorithm for Scheduling Independent Task on Nonidentical processor" , *Proceedings of the 21st annual international conference on Supercomputing. New York, NY, USA: ACM, 2007, pp. 23–32.*