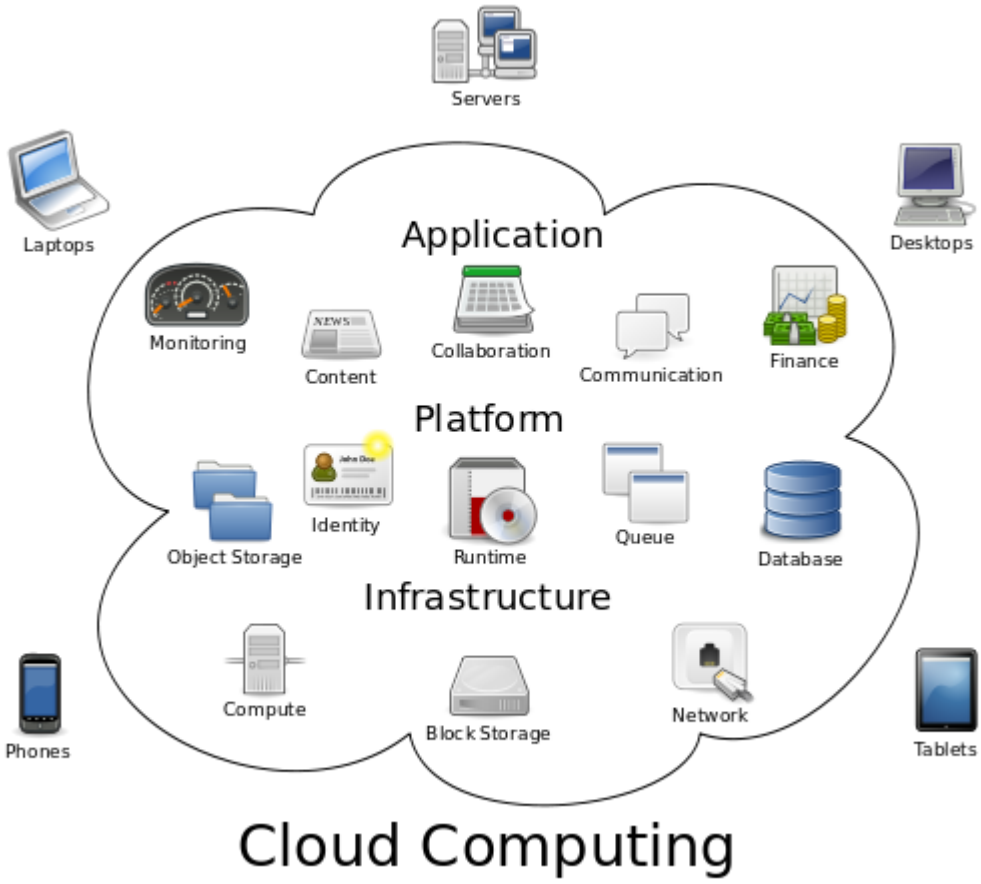
Introduction

The term Cloud Computing implies that you receive IT processing as a service rather than as a product or software. The simple method to visualize this is to compare to electricity: local computing is comparable to everyone owing a mechanical generator to produce their own electricity [1]. Cloud computing is about centralizing the computing activity, similar to producing electricity in power plants and distributing it via grids. Cloud computing is broken down into three segments: “Applications”, ”Platforms”, and “Infrastructure”. Each segment serves a different product for businesses and individuals around the world [1].A central server administers the system, monitoring traffic and client demands to ensure everything runs smoothly. Cloud computing is an emerging computing technology that uses the internet and central remote servers to maintain data and applications [1]. This technology allows for much more efficient computing by centralizing storage, memory, processing and bandwidth. Cloud computing is an outgrowth of the ease of access to remote computing sites provided by the internet. Cloud computing relies on sharing of resources to achieve coherence and economies of scale similar to a utility (like the electricity grid) over a network [1], [2].

Characteristics

The characteristics of cloud computing include on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service. On-demand self -service means that customers (usually organizations) can request and manage their own computing resources. Broad network access allows services to be offered over the Internet or private networks. Pooled resources means that customers draw from a pool of computing resources, usually in remote data centers. Services can be scaled and use of a service is measured and customers are billed accordingly.

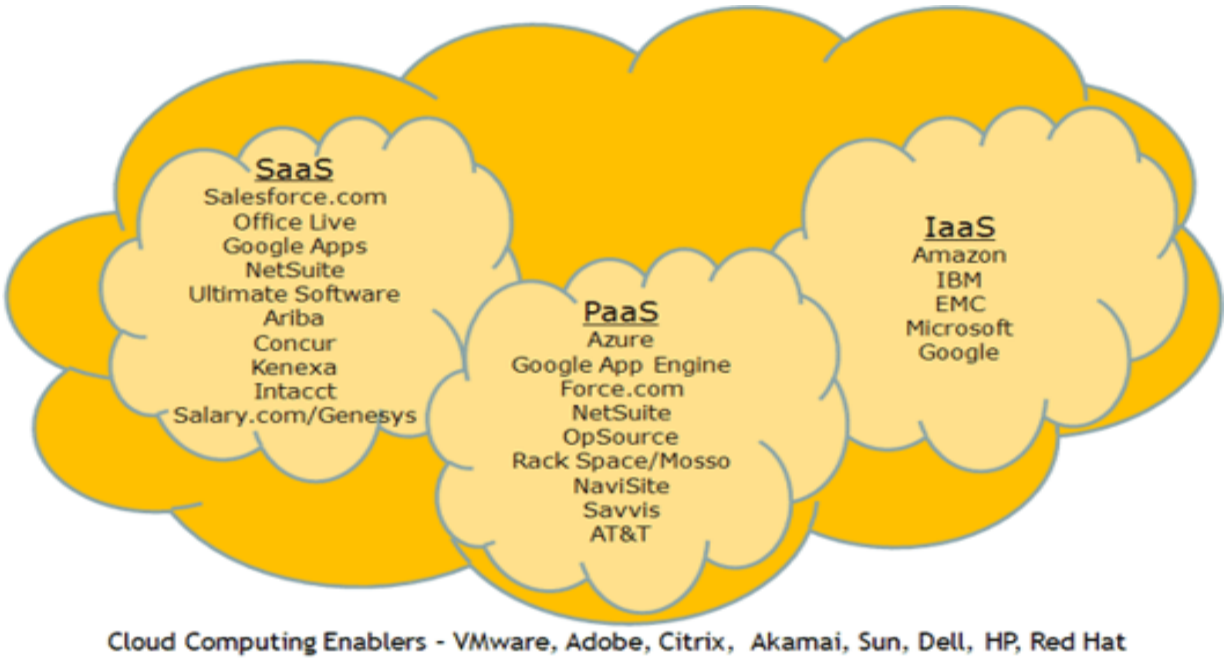


Cloud Services:

Software as a service (SaaS) is a software distribution model in which applications are hosted by a vendor or service provider and made available to customers over a network, typically the internet [1][2].1.1.2Platform as service (PaaS):

Platform as a service (PaaS) encapsulates a layer of software and provides it as service that can be used to build higher level services. There are at least two perspectives on PaaS depending on the perspective of the producer or consumer of the services.

Infrastructure as a service (IaaS) delivers basic storage and compute capabilities as standardized services over the network. Servers, storage systems, switches, routers and other systems are pooled and made available to handle workloads that range from application components to high performance computing applications.



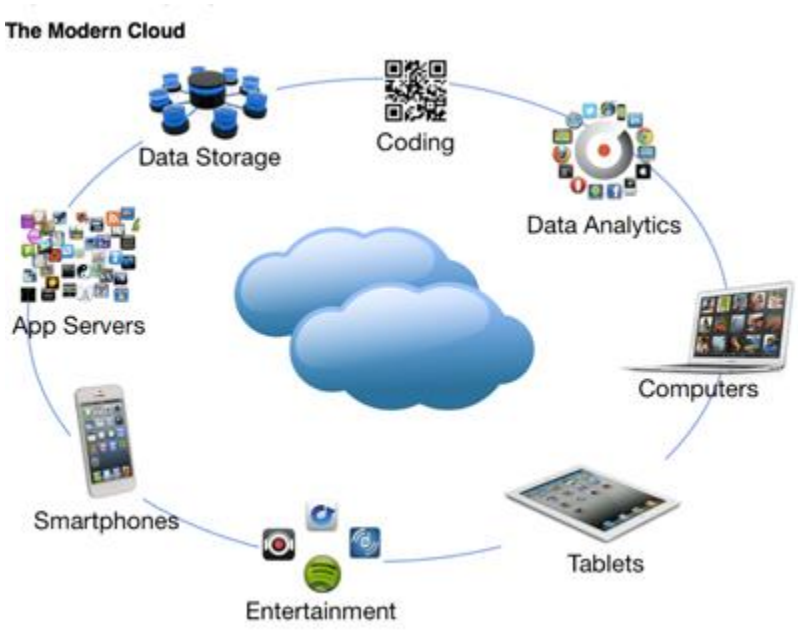
Cloud computing Models

Public Clouds: Public clouds are run by third parties and applications from different customers are likely to be mixed together on the cloud's servers, storage systems, and networks. Public clouds are most often hosted away from customer premises.

Private Clouds: Private clouds are built for the exclusive use of one client, providing the utmost control over data, security, and quality of service.

Hybrid Clouds: Hybrid clouds combine both public and private cloud models. They can help to provide on-demand, externally provisional scale.

Reasons For Complexity



Cloud services are popular because they can reduce the cost and complexity of owning and operating computers and networks. Since cloud users do not have to invest in information technology infrastructure, purchase hardware, or buy software licenses, the benefits are low up-front costs, rapid return on investment, rapid deployment, customization, flexible use, and solutions that can make use of new innovations. In addition, cloud providers that have specialized in a particular area (such as e-mail) can bring advanced services that a single company might not be able to afford or develop. Some other benefits to users include scalability, reliability, and efficiency. Scalability means that cloud computing offers unlimited processing and storage capacity. The cloud is reliable in that it enables access to applications and documents anywhere in the world via the Internet. Cloud computing is often considered efficient because it allows organizations to free up resources to focus on innovation and product development. Another potential benefit is that personal information may be better protected in the cloud. Specifically, cloud computing may improve efforts to build privacy protection into technology from the start and the use of better security mechanisms.

Cloud computing will enable more flexible IT acquisition and improvements, which may permit adjustments to procedures based on the sensitivity of the data. Widespread use of the cloud may also encourage open standards for

cloud computing that will establish baseline data security features common across different services and providers. Cloud computing may also allow for better audit trails. In addition, information in the cloud is not as easily lost (when compared to the paper documents or hard drives, for example).

Also cloud has the potential to change the way IT hardware is purchased, designed and used. With its promise of infinite scalability and a pay-as-you-go pricing model, the primary benefit that cloud services extends to the large enterprise is greater business effectiveness at lower IT costs. For the small and medium business (SMB) segment, cloud services lower barriers to market growth by lowering technology costs and upfront investments and Cloud computing brings utility computing closer to reality.

The benefits and challenges of cloud computing

In recent years, cloud computing has emerged as an important solution offering enterprises a potentially cost effective model to ease their computing needs and accomplish business objectives. Wilson Law, a manager at member firm, Moore Stephens LLP Singapore, provides some key benefits below worth considering:

a) **Optimized server utilisation -** as most enterprises typically underutilize their server computing resources, cloud computing will manage the server utilization to the optimum level.

b) **Cost saving -** IT infrastructure costs are almost always substantial and are treated as a capital expense (CAPEX). However, if the IT infrastructure usually becomes an operating expense (OPEX). In some countries, this results in a tax advantage regarding income taxes. Also, cloud computing cost saving can be realised via resource pooling.

c) Dynamic scalability - many enterprises include a reasonably large buffer from their average computing requirement, just to ensure that capacity is in place to satisfy peak demand. Cloud computing provides an extra processing buffer as needed at a low cost and without the capital investment or contingency fees to users.

d) Shortened development life cycle - cloud computing adopts the service-orientates architecture (SOA) development approach which has significantly

shorter development life cycle that that required by the traditional development approach. Any new business application can be developed online, connecting proven functional application building blocks together.

e) Reduced time for implementation - cloud computing provides the processing power and data storage as needed at the capacity required. This can be obtained in near-real time instead of weeks or months that occur when a new business initiative is brought online in a traditional way.

For all the above benefits of cloud computing, it also incorporates some unique and notable technical or business risk as follows:
a) Data location - cloud computing technology allows cloud servers to reside anywhere, thus the enterprise may not know the physical location of the server used to store and process their data and applications. Although from the technology point of view, location is least relevant, this has become a critical issue for data governance requirements. It is essential to understand that many Cloud Service Providers (CSPs) can also specifically define where data is to be located.

b) Commingled data - application sharing and multi-tenancy of data is one of the characteristics associated with cloud computing. Although many CSPs have multi-tenant applications that are secure, scalable and customisable, security and privacy issues are still often concerns among enterprises. Data encryption is another control that can assist data confidentiality.

c) Cloud security policy / procedures transparency - some CSPs may have less transparency than others about their information security policy. The rationalisation for such difference is the policies may be proprietary. As a result, it may create conflict with the enterprise's information compliance requirement. The enterprise needs to have detailed understanding of the service level agreements (SLAs) that stipulated the desired level of security provided by the CSPs.

d) Cloud date ownership - in the contract agreements it may state that the CP owns the data stored in the cloud computing environment. The CSP may demand for significant service fees for data to be returned to the enterprise when the cloud computing SLAs terminates.

e) Lock-in with CSP's proprietary application programming interfaces (APIs) - currently many CSPs implement their application by adopting the proprietary APIs. As a result, cloud services transition from one CSP to another CSP, has become extremely complicated, time-consuming and labour-intensive.

f) Compliance requirements - today's cloud computing services, can challenge various compliance audit requirements currently in place. Data location; cloud computing security policy transparency; and IAM, are all challenging issues in compliance auditing efforts. Examples of the compliance requirement including privacy and PII laws; Payment Card Industry (PCI) requirements; and financial reporting laws.

g) Disaster recovery - it is a concern of enterprises about the resiliency of cloud computing, since data may be commingled and scattered around multiple servers and geographical areas. It may be possible that the data for a specific point of time cannot be identified. Unlike traditional hosting, the enterprise knows exactly where the location is of their data, to be rapidly retrieved in the event of disaster recovery. In the cloud computing model, the primary CSP may outsource capabilities to third parties, who may also outsource the recovery process. This will become more complex when the primary CSP does not ultimately hold the data.

Businesses are under increasing pressure to sharpen their business practices. Too few people are aware of the security threats that are emerging. Nevertheless, they are responsible for ensuring that sensitive data will remain authentic, accurate, available, and will satisfy specific compliance requirements. Thus, it is essential for an organization to understand their current IT risks profile in order for them to determine the company's levels of IT risk tolerance and IT risk policies, and oversee management in the design, implementation and monitoring of the risk management and internal controls system.

Abstract

Cloud computing offers 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 electrical energy, contributing to high

operational costs and carbon footprints to the environment. Therefore, we need Green Cloud computing solutions that can not only minimize operational costs but also reduce the environmental impact. In this paper, we define an architectural framework and principles for energy-efficient Cloud computing.

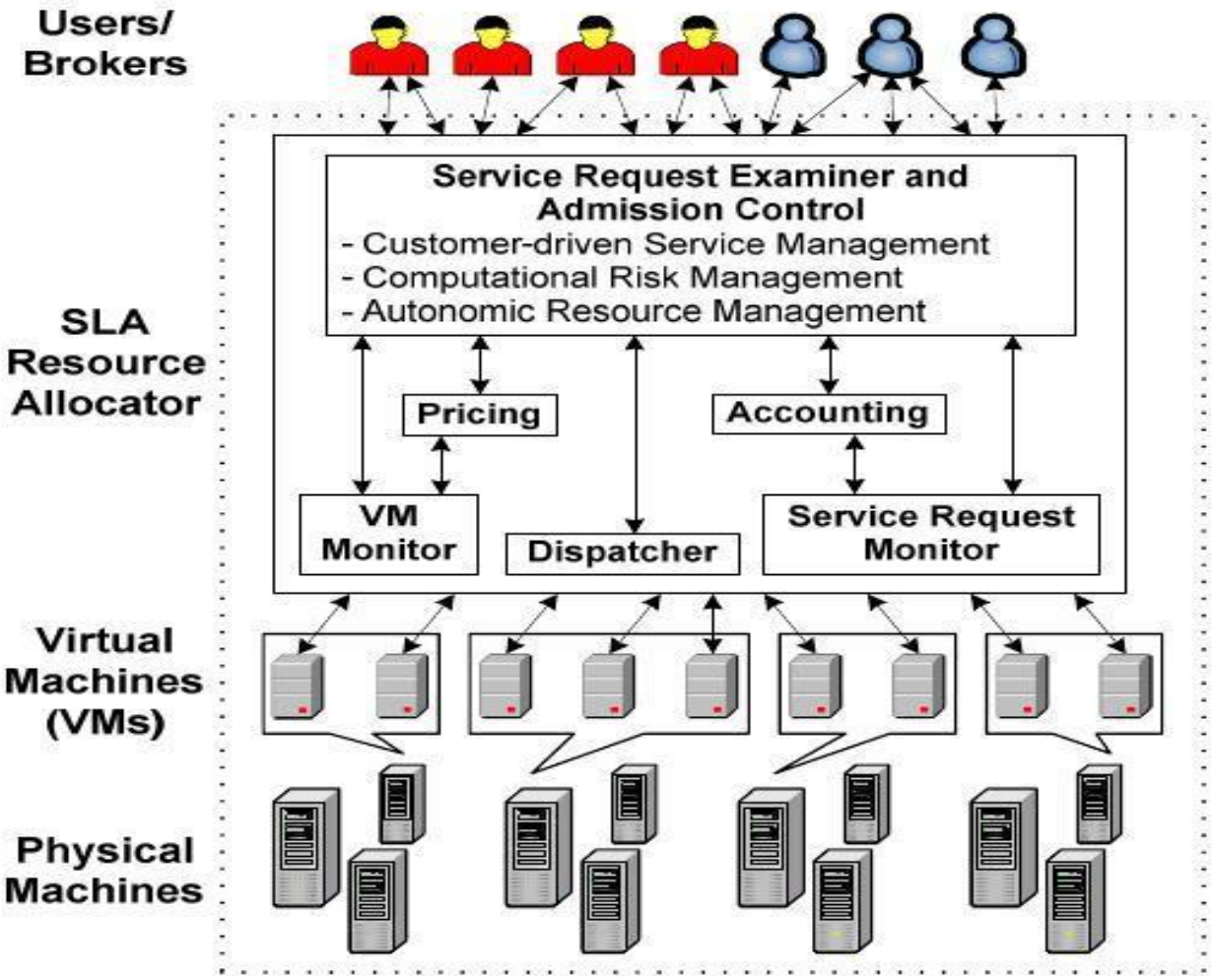
Based on this architecture, we present our vision, open research challenges, and resource provisioning and allocation algorithms for energy-efficient management of Cloud computing environments. The proposed energy-aware allocation heuristics provision data center resources to client applications in a way that improves energy efficiency of the data center, while delivering the negotiated Quality of Service (QoS). In particular, in this paper we conduct a survey of research in energy-efficient computing and propose: (a) architectural principles for energy-efficient management of Clouds; (b) energy-efficient resource allocation policies and scheduling algorithms considering QoS expectations and power usage characteristics of the devices; and (c) a number of open research challenges, addressing which can bring substantial benefits to both resource providers and consumers.

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 depending on their QoS requirements [30]. Fig. 1 shows the high-level architecture for supporting energy-efficient service allocation in a Green Cloud computing infrastructure [6]. There are basically four main entities involved:

1. Consumers/Brokers: Cloud consumers or their brokers submit service requests from anywhere in the world to the Cloud. It is important to notice that there can be a difference between Cloud consumers and users of deployed services. For instance, a consumer can be a company deploying a web-application, which presents varying workload according to the number of ‘‘users’’ accessing it.



2. Green Service Allocator: Acts as the interface between the Cloud infrastructure and consumers. It requires the interaction of the following components to support the energy-efficient resource management:

a) Green Negotiator: Negotiates with the consumers/brokers to finalize the SLAs with specified prices and penalties (for violations of the SLAs) between the Cloud provider and consumer depending on the consumer's QoS requirements and energy saving schemes. In case of web-applications, for instance, a QoS metric can be 95% of requests being served in less than 3 s.

b) Service Analyzer: Interprets and analyzes the service requirements of a submitted request before accepting it. Hence, it needs the latest load and energy information from VM Manager and Energy Monitor respectively.

c) Consumer Profiler: Gathers specific characteristics of consumers so that important consumers can be granted special privileges and prioritized over other consumers.
d) Pricing: Decides how service requests are charged to manage the supply and demand of computing resources and facilitate in prioritizing service allocations effectively.
e) 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.
f) Service Scheduler: Assigns requests to VMs and determines resource entitlements for the allocated VMs. If the auto scaling functionality has been requested by a customer, it also decides when VMs are to be added or removed to meet the demand.
g) VM Manager: Keeps track of the availability of VMs and their resource usage. It is in charge of provisioning new VMs as well as reallocating VMs across physical machines to adapt the placement.
h) Accounting: Monitors the actual usage of resources by VMs and accounts for the resource usage costs. Historical data of the resource usage can be used to improve resource allocation decisions.

Virtual Machines: Multiple VMs can be started and stopped on-demand on a single physical machine to meet accepted service requests, hence providing maximum flexibility to configure various partitions of resources on the same physical machine to different specific requirements of service requests. In addition, multiple VMs can concurrently run applications based on different operating system environments on a single physical machine since every VM is completely isolated from one another on the same physical machine.

Physical Machines: The Data Center comprises multiple computing servers that provide the resources to meet the service demands

Power model

Power consumption by computing nodes in data centers is mostly determined by

the CPU, memory, disk storage and network interfaces. In comparison to other system resources, the CPU consumes the main part of energy, and hence in thiswork we focus on managing its power consumption and efficient usage.

Moreover, the CPU utilization is typically proportional to the overall system load. Recent studies [16,12,11,15] have shown that the application of DVFS on the CPU results in almost linear power-to-frequency.The problem of VM allocation can be divided in two: the first part is the admission of new requests for VM provisioning and placing the VMs on hosts, whereas the second part is the optimization of the current VM allocation. The first part can be seen as a bin packing problem with variable bin sizes and prices relationship for a server. The reason lies in the limited number of states that can be set to the frequency and voltage of the CPU and the fact that DVFS is not applied to other system components apart from the CPU. Moreover, these studies have shown that on average an idle server consumes approximately 70% of the power consumed by the server running at the full CPU speed. This fact justifies the technique of switching idle servers to the sleep mode to reduce the total power consumption. Therefore, in this work we use the power model defined in (1).

$$P(u) = k \cdot P\text{max} + (1 - k) \cdot P\text{max} \cdot u,$$

Pmax is the maximum power consumed when the server is fully utilized; k is the fraction of power consumed by the idle server (i.e. 70%); and u is the CPU utilization. For our experiments Pmax is set to 250 W, which is a usual value for modern servers. For example, according to the SPECpower benchmark, for the fourth quarter of 2010, the average power consumption at 100% utilization for servers consuming less than 1000 W was approximately 259 W.The utilization of the CPU may change over time due to the workload variability. Thus, the CPU utilization is a function of time and is represented as $u(t)$. Therefore, the total energy consumption by a physical node (E) can be defined as an integral of the power consumption function over a period of time

$$E = P(u(t))dt$$

2. Related work

1. One of the first works, in which power management has been applied at the data center level, has been done by Pinheiro et al. [7]. In this work the authors have proposed a technique for minimization of power consumption in a heterogeneous cluster of computing nodes serving multiple web-applications. The main technique applied to minimize power consumption is concentrating the workload to the minimum of physical nodes and switching idle nodes off. This approach requires dealing with the power/performance trade-off, as performance of applications can be degraded due to the workload consolidation. Requirements to the throughput and execution time of applications are defined in SLAs to ensure reliable QoS. The proposed algorithm periodically monitors the load of resources (CPU, disk storage and network interface) and makes decisions on switching nodes on/off to minimize the overall power consumption, while providing the expected performance. The actual load balancing is not handled by the system and has to be managed by the applications. The algorithm runs on a master node, which creates a Single Point of Failure (SPF) and may become a performance bottleneck in a large system. In addition, the authors have pointed out that the reconfiguration operations are time-consuming, and the algorithm adds or removes only one node at a time, which may also be a reason for slow reaction in large-scale environments. The proposed approach can be applied to multi-application mixed-workload environments with fixed SLAs.
2. Another resource that has been recognized by the research community as a significant energy consumer is network infrastructure. Gupta et al. [17] have suggested putting network interfaces,links, switches and outers into sleep modes when they are idle in order to save the energy consumed by the Internet backbone and consumers.
3. Similarly to [11], the approach suits only enterprise environments as it does not support strict SLAs and requires the knowledge of application priorities to define the shares parameter. Other limitations are that the allocation of

VMs is not adapted at run-time (the allocation is static) and no other resources except for the CPU are considered during the VM reallocation.

4. Verma et al. [15] have formulated the problem of power-aware dynamic placement of applications in virtualized heterogeneous systems as continuous optimization: at each time frame the placement of VMs is optimized to minimize power consumption and maximize performance. Like in [13], the authors have applied a heuristic for the bin packing problem with variable bin sizes and costs. Similarly to [10], live migration of VMs is used to achieve a new placement at each time frame. The proposed algorithms, on the contrary to our approach, do not handle strict SLA requirements: SLAs can be violated due to variability of the workload.
5. Cal-heiros et al. [27] have investigated the problem of mapping VMs on physical nodes optimizing network communication between VMs however, the problem has not been explored in the context of the optimization of energy consumption.
6. Recently, a number of research works have been done on the thermal-efficient resource management in data centers [28,29].The studies have shown that the software-driven thermal management and temperature-aware workload placement bring additional energy savings. However, the problem of thermal management in the context of virtualized data centers has not been investigated.

Various policies designed for virtual machine migration

1. Virtual Machine Migration Policies in Clouds by Amritpal Singh and Supriya Kinger

The Minimum Migration Time Policy:

The Minimum Migration Time (MMT) policy migrate a VM that requires the minimum time to complete a migration relatively to the other VMs allocated to the host. The migration time is estimated as the amount of RAM utilized by the VM divided by the spare network bandwidth available for the host j.

The Random Choice Policy:
The Random Choice (RC) policy selects a VM to be migrated according to a uniformly distributed discrete random variable whose values index a set of VMs allocated to a host.

The Maximum Correlation Policy:
The Maximum Correlation (MC) policy is based on the idea that the higher the correlation between the resource usage by applications running on an oversubscribed server, the higher the probability of the server overloading. According to this idea, we select those VMs to be migrated that have the highest correlation of the CPU utilization with other VMs. To estimate the correlation between CPU utilizations by VMs, we apply the multiple correlation coefficients. It is used in multiple regression analysis to assess the quality of the prediction of the dependent variable. The multiple correlation coefficients correspond to the squared correlation between the predicted and the actual values of the dependent variable. It can also be interpreted as the proportion of the variance of the dependent variable explained by the independent variables [11]. In general; figure 3 shows the live migration of virtual machines [12].

Discussed Approach

In contrast to the discussed studies, we propose efficient heuristics for dynamic adaption of VM allocation at run-time according to the current utilization of resources applying live migration, switching idle nodes to the sleep mode, and thus minimizing energy consumption. The proposed approach can effectively handle strict SLAs, heterogeneous infrastructure and heterogeneous VMs. The algorithms do not depend on a particular type of workload and do not require any knowledge about applications running in VMs.

Binary Search:

Binary search or half-interval search algorithm finds the position of a specified input value (the search "key") within an array sorted by key value.[1][2] In each step, the algorithm compares the search key value with the key value of the middle element of the array. If the keys match, then a matching element has been found and its index, or position, is returned. Otherwise, if the search key is less than the middle element's key, then the algorithm repeats its action on the sub-

array to the left of the middle element or, if the search key is greater, on the sub-array to the right. If the remaining array to be searched is empty, then the key cannot be found in the array and a special "not found" indication is returned.

A binary search halves the number of items to check with each iteration, so locating an item (or determining its absence) takes logarithmic time. A binary search is a dichotomic divide and conquer search algorithm.

Worst case performance	O(log n)
Best case performance	O(1)
Average case performance	O(log n)
Worst case space complexity	O(1)

VM Selection

The optimization of the current VM allocation is carried out in two steps: at the first step we select VMs that need to be migrated, at the second step the chosen VMs are placed on the hosts using the MBFD algorithm. To determine when and which VMs should be migrated, we introduce three double-threshold VM selection policies. The basic idea is to set upper and lower utilization thresholds for hosts and keep the total utilization of the CPU by all the VMs allocated to the host between these thresholds. If the CPU utilization of a host falls below the lower threshold, all VMs have to be migrated from this host and the host has to be switched to the sleep mode in order to eliminate the idle power consumption. If the utilization exceeds the upper threshold, some VMs have to be migrated from the host to reduce the utilization. The aim is to preserve free resources in order to prevent SLA violations due to the consolidation in cases when the utilization by VMs increases. The difference between the old and new placements forms a set of VMs that have to be reallocated. The new placement is achieved using live migration of VMs [32]. In the following sections we discuss the proposed VM selection policies.

The Minimization of Migrations policy

The Minimization of Migrations (MM) policy selects the minimum number of VMs needed to migrate from a host to lower the CPU utilization below the upper utilization threshold if the upper threshold is violated.

The pseudo-code for the Minimization of migration algorithm for the over-utilization case is presented in here. The algorithm sorts the list of VMs in the decreasing order of the CPU utilization. Then, it repeatedly looks through the list of VMs and finds a VM that is the best to migrate from the host. The best VM is the one that satisfies two conditions. First, the VM should have the utilization higher than the difference between the host's overall utilization and the upper utilization threshold. Second, if the VM is migrated from the host, the difference between the upper threshold and the new utilization is the minimum across the values provided by all the VMs. If there is no such a VM, the algorithm selects the VM with the highest utilization, removes it from the list of VMs, and proceeds to a new iteration. The algorithm stops when the new utilization of the host is below the upper utilization threshold. The complexity of the algorithm is proportional to the product of the number of over-utilized hosts and the logarithm of number of VMs allocated to these hosts. As we will do binary search instead of linear in virtual machine list as the list is sorted in decreasing order in log (vmList).

hostList – List of all hosts in cloud

migrationList – List of virtual machines to be migrated from host

THRESH_UP – Maximum utilization that any host can beer

THRESH_LOW – Minimum utilization that any host can beer

sortDecreasingUtilization() – Sort the list in decreasing order in terms of utilization

getVmList() – Returns the list of virtual machines on a particular host

hUtil – Temporary variable to store the host utilization

Algorithm : Minimization of Migration (MM)

Input : hostList **Output** : migrationList

foreach h in hostList **do**

vmList = h.getVmList()

vmList.sortDecreasingUtilization()

hUtil = h.getUtil()

while hUtil > THRESH_UP **do**

bestFitVm = binarySearch (vmList, vm.getUtil() > hUtil – THRESH_UP)

hUtil = hUtil – bestFitVm.getutil()

migrationList.add(bestFitVm)

vmList.remove(bestFitVm)

if hUtil < THRESH_LOW **then**

migrationList.add(h.getVmList())

vmList.remove(h.getVmList())

return migrationList

Here, binarySearch (vmList, vm.getUtil() > hUtil – THRESH_UP) defines binary search on virtual machines utilization list. In this, we will do binary search for the element which is greater than difference of hUtil and THRESH_UP in vmList which is sorted in decreasing order.

Heap:

In optimizing allocation of VM to the hosts when we have set of hosts and set of virtual machines with the decreasing order of their utilization, then it becomes smart to use min heap for host to find minimum current utilizing host that linearly

searching so that when a VM is allocated to it ,it gives rises to minimum increase of power consumption due to allocation of new VM.
Min heap utilized to store set of hosts utilizes properties of heap for efficiently accessing minimum element in O(1).
a **heap** is a specialized tree based data structure that satisfies the *heap property:* If A is a parent node of B then the key of node A is ordered with respect to the key of node B with the same ordering applying across the heap. Either the keys of parent nodes are always greater than or equal to those of the children and the highest key is in the root node (this kind of heap is called *max heap*) or the keys of parent nodes are less than or equal to those of the children and the lowest key is in the root node (*min heap*).

Efficiency of heaps

Assume the heap has N nodes. Then the heap has log2(N+1) levels.

- Since the insert swaps at most once per level, the order of complexity of insert is O(log N)
- Since the remove swaps at most once per level, theorder of complexity of remove is also O(log N)
- Accessing top element O(1)

VM placement:

The problem of VM allocation can be divided in two: the first part is the admission of new requests for VM provisioning and placing the VMs on hosts, whereas the second part is the optimization of the current VM allocation. The first part can be seen as a bin packing problem with variable bin sizes and prices. To solve it we apply a modification of the Best Fit Decreasing 1 bins (where *OPT* is the number of bins given by the optimal solution) .In our modification, the Modified Best Fit Decreasing (MBFD) algorithms, we sort all VMs in decreasing 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. For choosing most power efficient node in O(1), we keep resources with their utilization and maximum capacity in minimum heap. It provides choice of most efficient power consumption node to be at top and can be accessed in constant time and virtual

machine can be allocated to node if it has enough power and node is kept back to heap with new utilization to readjust heap for further allocation . The pseudo-code for the algorithm is presented in Algorithm 1. The complexity of the allocation part of the algorithm is $n \cdot m$, where n is the number of VMs that have to be allocated and m is the number of hosts.

Modified Efficient Algorithm for Virtual Memory Allocation

Input:

1. hostlist=List of available hosts in the cloud.
2. vmlist=List of available virtual machines to be allocated hosts.
3. vm = the utilization of virtual machine for any host.
4. host = power utilization of the hosts in hostlist.
5. sortDecreasingUtilization()= Sort entity(vm or host) in decreasing order of utilization.
6. createMinHeap()=Creates heap of entity(vm or host) as per their utilization.

Algorithm

7. allocatedHost=Allocated for Vm
8. minheapTop()= Returns root value of heap.

A. Sort VM in vmlist in decreasing order of utilization

```
Vmlist. sortDecreasingUtilization();
```

B. Create min Heap of hosts according to their utilization.

```
host.createMinHeap(hostlist);
```

C. **Foreach** vm in vmlist **Do**

```
{ allocatedHost=NULL

   MaxhostUtil=host.minheapTop();
  Temp=host.minheapTop();
 If(MaxhostUtil-Temp>=vm)//host has enough resource for vm
  {  allocatedHost=host;// allocated host for Vm is host
    host.hUtil=Temp+vm;
    Heapify(hostlist,1);
  }
```

}

Return allocatedHost

Performance Analysis

Minimized Migration Benefits:

1. Energy Benefits
2. Load Benefits
3. Online Maintenance
4. Increased lifetime of hosts
5. Reliability of hosts
6. Preventing Performance degradation
7. Fulfilling the criteria of QoS
8. Higher Performance
9. Improved manageability