# Energy efficiency in Clouds
## Cloud Energy efficiency at the VM level

Boris Dabrowski[1] and Harald Steinlechner[2]

[1] e0827131@student.tuwien.ac.at
[2] e0825851@student.tuwien.ac.at

**Abstract.** As clod computing becomes more important continuously, data centers continuously gain additional importance. Cloud computing, by definition a highly dynamic discipline, poses complex challenges on service providers. On one hand quality of service should be maintained, on the other hand the energy consumption must be kept as low as possible to be economically competitive. Data centers highly build on virtualized resources - dynamic scaling of physical resources is therefore essential. In this article we will review important resource management approaches at virtual machine level as well as its implementation techniques. Furthermore we give prospect to these aspects and its additional requirements in the context of real time requirements.

## 1 Introduction

Data centers became an essential player in todays economy and lately also home computing. Due to the emerging market of cloud computing (like Amazon EC2 [1], Microsoft Azure [2]) also non commercial customers transfer data and computation onto the cloud. This centralization of course puts huge responsibility to service providers. Usually Service Level Agreements (SLAs) precisely specify Quality of Service (QoS) parameters of the service. SLAs provide some form of contract between service provider and service consumer. However in cloud computing environments achieving given SLAs is a challenging task. This is due to the high degree of dynamism within the distributed system.
Aside from SLAs reducing the actual energy consumption of systems is essential for service providers. On the one hand there is the global increase in electrical energy consumption and therefore high prices thatof on the market. On the other hand energy-inefficient systems in consequence require better cooling systems which again yields additional cost. Last but not least the aspect of green computing should be taken into account. According to Kaplan et al. [3] data centers consumed about 0.2% of the worldwide energy production. Furthermore energy consumed by data centers doubled from 2000 to 2006 up to 25 thousands households.
Aside from economical aspects carbon emissions additionally exacerbate the need for more efficient computing. Given all that we feel encouraging motivation in the implementation of energy efficient data centers. Surprisingly much energy is

wasted in idle times of servers. According to Meisner et al. [4] server utilization is below 30% in typical deployments. Simple measures like switching off idle nodes seem highly beneficial. Unfortunately reducing energy consumption while maintaining the SLA objectives is challenging. Due to high degree of dynamism in cloud environments the system has to scale dynamically at runtime while keeping power consumption as low as possible.

Regarding energy efficient computing and dynamic utilization of computing resources several fundamentally different approaches have been identified:

- Improvement of the applications algorithms. Minimizing the runtime or CPU and memory utilization of the application itself leads to a better energy consumption in the whole system. Instead of optimizing in a top-down manner this approach can be seen as a bottum-up solution, where the smallest entity in the system gets optimized before looking at the system from a higher abstraction.
- Energy efficient hardware. Even if applications and the system managing them is highly optimised there may be still room for improvement from a hardware point of view. Using more expensive hardware or maybe even engineering new hardware systems that consume less power can easily pay off in the long run when for example cooling costs are minimised as a result.
- Terminal servers and thin clients. Delegating high and power consuming workloads on systems that are able to cope with them in an efficient manner may reduce overall power consumption in the system.
- Dynamic Voltage and Frequency Scaling (DVFS). Adjusting the frequency of processors according to current workloads results in conserving power and a reduction of heat generated by the chip.
- Virtualization of computer resources. Using virtual machines as containers for applications has proven to be an ideal approach for approaching energy efficiency in cloud systems. With the possibility of migrating running virtual machines physical nodes can either be turned off or unloaded in order to save energy.

In this paper we will focus on the last approach to energy efficiency in cloud data centers as it is the state of the art on this matter.

The remainder of this paper is structured as followed:
Chapter 2 presents a brief overview of the current research around the topic of energy efficiency in cloud data centers. Chapter 3 contains a more in-depth overview of the current approaches to the relocation of virtual machines followed by an introduction to implementation techniques for real time cloud service in chapter 4. Chapter 5 presents some of the relevant systems coping with virtual machine migration and chapter 6 eventually wraps up what has been presented in this paper.

## 2 Related Work

Dynamic Voltage and Frequency Scaling (DVFS) which must be supported at hardware level has been employed in the field of super computing [5]. Chen et al. [6] propose a heuristical model for power provisioning using dynamic frequency scaling which gives approximation factor guarantees for large clusters.

PowerNap [4] reduces idle times of servers. The system continuously switches between high-performance active state a near zero idle state. The algorithm aims for minimizing idle power and transition times instead of focusing on fine grained power profiles.

Cloud computing in its nature of virtualizing resources led approaches based on virtual machines and dynamic migration thereof. *pMapper* proposed by Verma et al. [7] utilizes migration cost as cost model for virtual machine migration. Beloglazov et al. [8] propose a resource management scheme aware of given QoS. Resource utilization drives a dynamic relocation scheme for virtual machines. They show energy savings about of 84% compared to non energy aware systems and 66% compared to pure DVFS systems.

Guimaraes et al. [9] performs scheduling of virtual machines utilizing heuristics. Furthermore it utilizes DVFS additionally to dynamic migration and introduces power aware active cooling to reduce cost per MIPS.

Duy et al. [10] try to minimize running nodes by employing load prediction implemented using a neuronal network which operates on historical data.

Real-Time cloud services pose tightened challenges when it comes to energy efficiency. Kim et al. [11] propose a power aware scheduling mechanism for soft and hard-time cloud services.

## 3 Dynamic relocation of Virtual Machines

One major approach in reducing energy consumption and maximizing cost effectiveness in data centers is the application of virtualization technologies allowing for consolidation of servers to one physical node as Virtual Machines and thus reducing the amount of hardware in use.

Generally speaking there are three approaches when leveraging live migration of Virtual Machines:

**Optimization over various system resources** The relocation mechanism takes system parameters such as current CPU, RAM and network bandwidth utilization under consideration when deciding on VM replacements.

**Network optimization** Virtual Machines may often communicate with each other over a virtual network topology. Oftentimes the communicating VMs may end up on distant physical nodes due to not optimized node allocation or VM migration. Thus the network communication may involve networking hardware that consumes additional power. Observation of the network communication and replacement of Virtual Machines so that communication partners are located on nearby or ideally the same physical nodes results in a reduction of data transfer overhead and ultimately in a minimization of power consumption.

**Thermal optimization** Electrical energy consumed by computing resources is partly transformed into heat and high temperature can lead to many problems such as a decreased lifetime of hardware and a reduced availability and reliability of the system. In order to increase the longevity of hardware and minimize the risk of crashes and failures the heat emission must be reduced. Cooling is in general a very expensive matter and should be not only dealt with from the hardware perspective but also optimized from the software's side. In this regard the relocation mechanism should take the current temperature of nodes into consideration when relocating Virtual Machines. The aim is to avoid hot spots by reducing workload of the overheated nodes and thus decrease error-proneness and cooling efforts.

### 3.1 Heuristics for migration

The problem of maintaining Virtual Machines on the least amount of physical nodes possible can be intuitively viewed as the classical bin packing problem, one of the original NP-complete problems, where the number of bins of a specific capacity are physical nodes and the objects of different volumes to be filled in bins are Virtual Machines. Classic algorithms such as First-Fit-Decreasing, Best-Fit and Worst-Fit are applicable in the context of VM migration. However these algorithms may bring lots of resource gaps because once an element has been placed into a bin it has no chance to move to another for optimum bin usage. Thus other variations which take the relocation of elements into other bins under consideration have been presented. The following part shows an overview of different approaches. On a *Workload Departure Event* the remaining workloads of a node that finished a workload will be reassigned as mentioned above, so that the current node may go idle. Eventually the *Workload Resizing Event* can be seen as a combination of a departure event and an arrival event.

Li et al. [12] have presented an event-driven variation as the main component of their system called EnaCloud. Their algorithm considers three events when deciding on VM migrations. Workload Arrival Events are triggered on new incoming workloads and cause the system to insert them into already used nodes instead of starting new ones. Differing from the First-Fit-Decreasing algorithm this heuristic does not simply put new workload into the first node that has resources left, but rather tries to replace smaller already assigned workloads with the new workload. Those replaced smaller workloads are than reassigned in the same fashion.

Beloglazov and Buyya [13] on the other hand have presented a modification of the Best Fit Decreasing algorithm. The modified version sorts all Virtual

Machines in decreasing order of current utilization and then allocates them to the nodes that produce the least increase in power consumption after the allocation. When optimizing already assigned VMs there are two steps involved. First the migration candidates have to be chosen and then allocated to new nodes via the afore mentioned BFD modification. Four different heuristics for choosing the VMs in the first step were proposed. The first heuristic is based on the idea of a Single Threshold, an upper utilization value that must not be exceeded when replacing VMs with the aim of preserving free resources to prevent violation of SLAs in case of a VM consolidation and increasing resource requirement of VMs. The three remaining heuristics are based on lower and upper thresholds with different policies for choosing VMs that must be migrated. In case of a value under the lower threshold the nodes VMs will be migrated and the node turned off. If the upper threshold is exceeded VMs must be migrated so that the value sits between lower and upper threshold again. Each of the remaining heuristics is based on another policy: Random Choice picks the relocating VMs at random. Highest Potential Growth aims to minimize total potential increase of the utilization to prevent SLA violation. Minimization of Migrations on the other hand tries to minimize migration overhead which turned out to create the best results in energy savings.

## 4 Implementation techniques for Real time cloud services

As the *Anything as a Service (XaaS)* paradigm evolves, real time services become apparent in cloud computing. In this section we will discuss challenges and trends in real-time cloud services. The approaches we discussed so far have some shortcomings when it comes to real time requirement. As previously discussed provisioning scheme has to be aware of QoS. This holds as well for deadline constraints - like given in applications with real-time aspects. Of course for these kind of services additional measures have to be taken to meet QoS effectively while maintaining low energy consumption.

Real-time services can be categorized into two types of deadline models:

**soft** if the deadline is not met the service provider receives some penalty. However there may still be value in execution this service. To capture this a penalty function like decreasing function describes the reduced value of the service result

**hard** if the deadline is not met the service provider receives a penalty - independently from if the result is eventually computed.

Usually real time services tasks are composed out of smaller sub-tasks. More precisely such tasks are characterised with following parameters:

- Release time, i.e. the time the task may start, is issued
- Worst-case execution time: the maximal time which is required to complete
- Period: duration of one period in case of periodic tasks

Of course of these different models the scheduling system must be aware of the penalty environment.

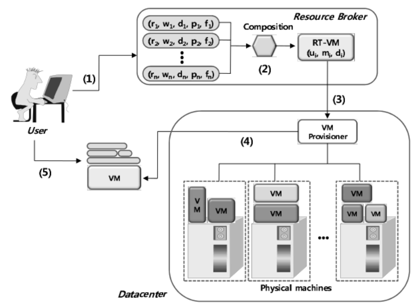


**Fig. 1.** Overall overview of the framework (Kim et al. [11])

Soft Real-time services are parametrised with a penalty function. If the deadline is not met a function describes the reduced value of the service. Example penalty functions are depicted in 4.

$$value = \begin{cases} 1, & \text{if } delay <= 0 \\ \phi_i \ (delay), & \text{if } delay > 0 \end{cases}$$

Since we focus on the characteristics of real-time services respective to VM provisioning, we presume the existence of a real-time VM [14] and a hierarchical real time framework we can build on [15]. The overall architecture of the framework is depicted in Figure 4.

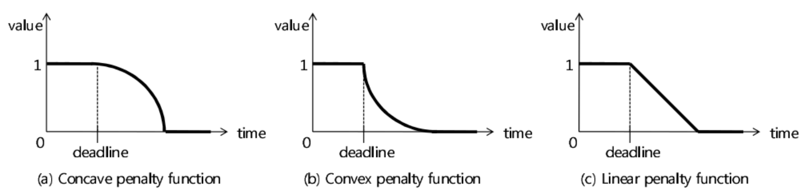


**Fig. 2.** Example penalty functions (Kim et al. [11])

**The power model**

In data centers processing units constitute the main energy consumer. Power consumption may be separated into dynamic and static power. Static power consumption is independent from actual workload on the machine. [11] et al. define the power $P$ to be consumed as follows.

$$P = C * f^3$$

The supply voltage is usually proportional to the processor frequency which is modelled by the coefficient $C$. Let $f$ be the actual processor frequency. Using DVFS the processor frequency may be reduced resulting in lower power consumption but longer running time.

**VM Provisioning**

The provisioning for hard real-time services as proposed by [11] basically works as follows.

- The data center receives a request for a hard Real-time VM from the resource broker.
- The broker assigns the VM to the machine with minimum price on which the VM can actually be scheduled. Soft Real-time service in contrast use a profit function taking penalty functions into account.
- The processors frequency determined by the actual DVFS scheme.

Three different DVFS schemes have been identified:

1. Lowest-DVFS: The processor speed is reduced to the lowest frequency which let all VMs meet their deadlines
2. $\delta$-Advanced-DVFS: Since Lowest-DVFS suffers from low acceptance rate due to the conservative nature of the approach this approach operates the processor $\delta$% faster.
3. Adaptive-DVFS: depending on future arrival times the processor runs faster than required. Note that this only works if the arrival rate is actually known.

As shown by simulation results found in [11] Lowest-DVFS does a surprisingly good job. However $\delta$-Adaptive-DVFS is superior regarding profit per unit of consumed power. For in depth review and analysis we refer to Kim et al. [11].

## 5 Comparison of selected systems

### 5.1 GridEye

GridEye [16] was proposed as a novel service-oriented monitoring system with a flexible and scalable architecture. It does not cope with virtual machine migration and placement strategies per se, though we think that it deserves being mentioned in this paper as it's architecture and general ideas can be seen as a predecessor to the more modern virtual machine migration systems.

The system evolved due to shortages in the current state of the art, such as the lack of interoperability. At the time each project employed its own protocols and schemes whilst providing different interfaces thus making it hard to interoperate with each other. Intrusive Overhead prevented many systems from being scalable. Eventually there was a lack of forecasting mechanisms, that left users with no ideas on how to optimize their applications. These problems have been seriously considered in the design of GridEye.

In a nutshell it was the first system to present the following qualities:

- A high-efficient and autonomic monitoring system that is supported by a flexible and scalable architecture.
- The possibility of monitoring different resources with different operating system platforms including MPP, Clusters, workstations and personal computers.
- A loose-coupled interface based on web services that allows for inter-operating and convenient information retrieval.
- A performance prediction mechanism based on an improved forecasting algorithm that uses mathematical models such as the $MA(k)$ and $ExS(\alpha)$ computation models without introducing too much load overheads.

### 5.2 PowerNap

According to Meisner et al. [4] typical deployments perform surprisingly bad respective to server utilization. Instead of switching of idle nodes like the approaches mentioned earlier, PowerNap introduced by Meisner et al. [4] in 2009 increases server utilization by putting underemployed nodes in a low energy state (a nap). Further on the authors [4] found that idle periods in typical systems are usally below $1s$ with mean lengths of 100's of milliseconds. Unlike hibernate states as provided by current operation systems the system manages very fast transition between high performance state and near-zero power idle state which accounts the requirement to handle such short idle periods efficently.

Processors only account for about 25% of power consumption in data-centers [17, 18]. So pure DVFS does not solve the energy efficiency problem (as mentioned earlier). PowerNap however reduces overall system power consumption to a minimum.

Unfortunately power supplies are inefficient at low loads [19] which renders PowerNap ineffective. Therefore the PowerNap utilizes an array of additional power supplies which may be switched on and off depending on processing nodes in nap mode.

Compared with optimal DVFS as reference power management system PowerNap outperforms the reference in respect to power efficiency.

### 5.3 EnaCloud

Li et al. proposed EnaCloud [12] as a novel approach to solving the current issues in minimizing the energy consumption in cloud data centers. The system uses virtual machines to encapsulate applications, which support application scheduling and live migration to minimize the amount of running physical machines and therefore optimize energy efficiency. Thus applications can move to other physical nodes without interrupting the service.

The application placement in EnaCloud is based on a modification of the classic bin packing problem, where each physical node corresponds to a bin. Additionally an energy-aware heuristic algorithm is proposed to get the application placement schemes regarding to specific workload events such as arrival, departure or resizing. Also a resource provision method has been introduced to avoid the frequent virtual machine migration due to often occurring resource resizing.

EnaCloud's architecture is designed and implemented in the python programming language for the iVIC platform, a virtual computing environment developed for HaaS (Hardware as a Service) and SaaS (Software as a Service) applications.

Physical nodes in EnaCloud can either be an open box, i.e. a running server node that contains assigned virtual machines, or a closed box, i.e. an idle server node without any running virtual machines. The heuristics aim is to aggregate the workloads tightly so as to ensure that the number of open boxes is minimal. This goal is reached via the event-driven heuristic that optimizes virtual machine positioning according to incoming events such as the Workload Arrival Event, Workload Departure Event and Workload Resizing Event as described in chapter 3. In the basic algorithm, handling the resizing events may incur more migration overhead. Resource demands may change frequently in real world applications, which will result in a lot of resizing events. A provisioning method has been introduced to reduce the resizing overhead and its higher energy costs of frequently migrating virtual machines. The method basically allocates more resources given by a specific *over-provision ration* to the workloads than they actually need according to past events.

The architecture of the EnaCloud framework is based on a central global controller as seen in the figure below. It contains a Concentration Manager and a Job Scheduler. The latter is responsible for managing incoming workloads and and receiving a virtual machine replacement schema from the Contentration Manager, which holds the main algorithms and acts as the brain of the system. The computed schema is then forwarded to the relevant physical nodes, which hold a Hypervisor, that is responsible for communicating with the Global Controller, and its virtual machines.

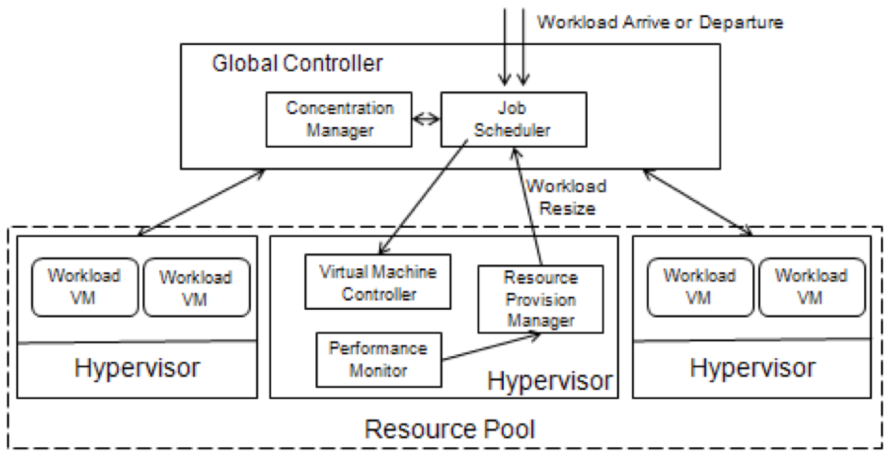


**Fig. 3.** EnaCloud's System Architecture [12]

EnaCloud can be seen as a modern approach to the idea of relocating virtual machines in real time in order to optimize overall energy consumption. However the centralized aspect in its architecture may act as a bottleneck and single point of failure. Also at this time the only reference value of the system for computing relocations is the current energy consumption. Further system data such as CPU or memory usage are planed for future implementations.

### 5.4 Mistral

Mistral [20] is a holistic controller framework that optimizes power consumption, application performance and power costs due to adapting actions and decision making in a single environment. The fundamental idea behind Mistral is its focus on minimizing adaptation costs that can easily lead to an overall worse power consumption in a scenario where workloads change often. In those cases it might be better to suffer from a small performance drop rather than introduce an expensive migration. Even a slight change such as the redistribution of resources amongst physical nodes could be more efficient than switching on a new node. Also the decision delay and power consumption caused by making the migration decision must also be considered. Mistral takes all these thoughts into account and establishes a multiple application and dynamic workload environment implemented as a scalable and multi-level controller that is able to deal with a vast number of applications and hosts with migration actions ranging from a few milliseconds to minutes in order to provide the optimal decision delay.

When deciding on how and when to adapt the system at run-time, Mistral computes an estimate of the possible benefits and the costs of each adaptation action. Those costs are determined by the duration of an action and its impacts on response times and power consumption during the adaptation. The benefits on the other hand takes issues such as the improved response time and power consumption of an action and how long the system may remain in this new adaptation which depends on how long the workload of the system remains approximately the same.

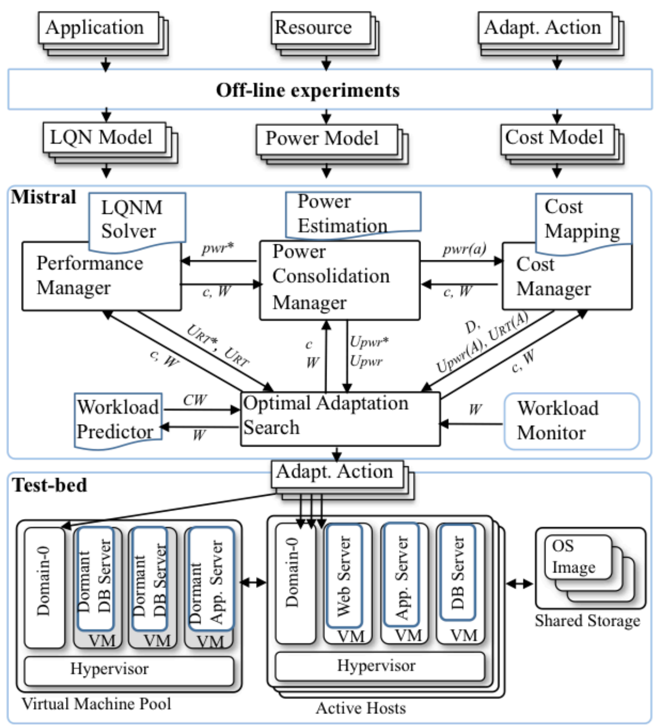


**Fig. 4.** Mistral's System Architecture [20]

Mistrals architecture is shown in figure 4, depicting a single controller unit. It consists of a predictor module, composed of the Performance Manager, the Power Consolidation Manager, the Cost Manager and the Workload Predictor, and an optimizer module, which includes the Optimal Adaption Search engine. As the names suggest the predictor module forecasts the application and power utilities to a given configuration and workload, where the Cost manager estimates application costs and the Workload Predictor computes workload resizing based on previous workload behaviour. On the other hand the optimizer module is responsible for creating an adaptation plan that will maximize utility. Typically

Mistral is deployed as a hierarchical control scheme with multiple layers and multiple controllers managing different groups of hosts and applications.

### 5.5 Comparison

To put those various frameworks coping with virtual machine relocation and other energy-efficiency techniques into perspective we put together a comparison of the main aspects of such systems, as shown table 1. We identified four main criteria that play a role when comparing those frameworks. Relocation Criteria describe the data on which relocation decisions and actions are based on. Depending on a frameworks concept and implementation its supported workloads can either be homogeneous or ideally heterogeneous. Systems can be designed for various environments such as all-purpose platforms which may allow easy integration or specific platforms which on the other hand may benefit from higher performance. Even tually there are different approaches to the heuristics at the core of each system.

| | Relocation Criteria | Workload | Environment | Heuristics |
|---|---|---|---|---|
| PowerNap | none | heterogeneous | standard | CPU utilization |
| EnaCloud | events | homogeneous | iVIC platform | event driven |
| vGreen | different schemes | heterogeneous | standard | VM characteristics |
| Mistral | estimated benefits | heterogeneous | standard | utility function |

**Table 1.** Comparison of Frameworks

In our compared systems the Relocation Criteria vary from event based information such as incoming, departing and resizing workload in EnaCloud to an estimation of energy benefits after a possible system adaptation in Mistral. Workloads and their environment are for the most part heterogeneous and all-purpose. Only EnaCloud stands out by running on the iVIC platform and only supporting homogeneous applications by its design. It seems as if the main difference between all the system lies in their heuristics. Most of them are based on the bin packing algorithm, but variations such as handling workload-events and forecasting workload behaviour have evolved.

## 6 Conclusion

Data centers based on cloud systems are growing rapidly as cloud computing continuously gains additional importance. This goes hand in hand with the growth of power consumption in data centers to unprecedented levels. Therefore energy efficiency now plays an even more important role and data centers work towards green environments. Virtualization of workloads is a key idea in this respect.

In this paper we provided an overview of current approaches to minimizing energy efficiency by migrating virtual machines. We briefly described the main implementations and systems, namely GridEye, PowerNap, EnaCloud, vGreen and Mistral, and created a short comparison based on the most important characteristics of such systems.

Research has shown that virtual machine management is a very approachable and feasible idea when it comes to optimizing energy efficiency in cloud oriented data centers. Though it seems as if this area has not really matured yet. There is still room for improvement as most publications describe various methods on deciding when to migrate but none of the implementations use them. Most of the time those migration decisions are based on CPU utilization. Other approaches such as thermal optimization or the minimization of network bandwidth overhead due to virtual machines communication with each other from different physical nodes have not really been studied in practice yet. Another issue that remains to be dealt with are dependencies between VMs. None of the mentioned systems represents such dependencies thus the handling of applications that run on different VMs but depend on each other is either undefined or not supported.

Apart from that we see this area of research as a promising and important aspect for the future of cloud computing.

## References


1. Amazon Corporation: Amazon elastic compute cloud (amazon ec2), 2002. URL: `http://aws.amazon.com/ec2/` (accessed on 14th June 2012)
2. Microsoft Corporation: Windows azure, 2010. URL: `http://www.windowsazure.com/en-us/` (accessed on 14th June 2012)
3. Kaplan, J.M., Forrest, W., Kindle, N.: Revolutionizing Data Center Energy Efficiency. Technical report, McKinsey & Company (July 2008)
4. Meisner, D., Gold, B.T., Wenisch, T.F.: Powernap: eliminating server idle power. In: Proceedings of the 14th international conference on Architectural support for programming languages and operating systems. ASPLOS '09, New York, NY, USA, ACM (2009) 205–216
5. Hsu, C.h., Feng, W.c.: A power-aware run-time system for high-performance computing. In: Proceedings of the 2005 ACM/IEEE conference on Supercomputing. SC '05, Washington, DC, USA, IEEE Computer Society (2005) 1–
6. Ge, R., Feng, X., Cameron, K.W.: Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters. In: Proceedings of the 2005 ACM/IEEE conference on Supercomputing. SC '05, Washington, DC, USA, IEEE Computer Society (2005) 34–
7. Verma, A., Ahuja, P., Neogi, A.: Power-aware dynamic placement of hpc applications. In: Proceedings of the 22nd annual international conference on Supercomputing. ICS '08, New York, NY, USA, ACM (2008) 175–184
8. Beloglazov, A., Buyya, R.: Energy efficient resource management in virtualized cloud data centers. In: Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on. (may 2010) 826 –831
9. Lago, D.G.d., Madeira, E.R.M., Bittencourt, L.F.: Power-aware virtual machine scheduling on clouds using active cooling control and dvfs. In: Proceedings of the 9th International Workshop on Middleware for Grids, Clouds and e-Science. MGC '11, New York, NY, USA, ACM (2011) 2:1–2:6
10. Duy, T.V.T., Sato, Y., Inoguchi, Y.: Performance evaluation of a green scheduling algorithm for energy savings in cloud computing. In: Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on. (april 2010) 1 –8
11. Kim, K.H., Beloglazov, A., Buyya, R.: Power-aware provisioning of virtual machines for real-time cloud services. Concurr. Comput. : Pract. Exper. **23**(13) (September 2011) 1491–1505
12. Li, B., Li, J., Huai, J., Wo, T., Li, Q., Zhong, L.: Enacloud: An energy-saving application live placement approach for cloud computing environments. Cloud Computing, IEEE International Conference on **0** (2009) 17–24
13. Beloglazov, A., Buyya, R.: Energy efficient allocation of virtual machines in cloud data centers. In: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing. CCGRID '10, Washington, DC, USA, IEEE Computer Society (2010) 577–578
14. Yoo, S., Park, M., Yoo, C.: A step to support real-time in virtual machine. In: Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE. (jan. 2009) 1 –7
15. Feng, X.A., Mok, A.K.: A model of hierarchical real-time virtual resources. In: Proceedings of the 23rd IEEE Real-Time Systems Symposium. RTSS '02, Washington, DC, USA, IEEE Computer Society (2002) 26–

16. Fu, W., Huang, Q.: Grideye: A service-oriented grid monitoring system with improved forecasting algorithm. In: Proceedings of the Fifth International Conference on Grid and Cooperative Computing Workshops. GCCW '06, Washington, DC, USA, IEEE Computer Society (2006) 5–12
17. Fan, X., Weber, W.D., Barroso, L.A.: Power provisioning for a warehouse-sized computer. In: Proceedings of the 34th annual international symposium on Computer architecture. ISCA '07, New York, NY, USA, ACM (2007) 13–23
18. Lefurgy, C., Wang, X., Ware, M.: Server-level power control. In: Proceedings of the Fourth International Conference on Autonomic Computing. ICAC '07, Washington, DC, USA, IEEE Computer Society (2007) 4–
19. Molloy, C., Iqbal, M.: Improving data-center efficiency for a smarter planet. IBM J. Res. Dev. **54**(4) (July 2010) 388–395
20. Jung, G., Hiltunen, M.A., Joshi, K.R., Schlichting, R.D., Pu, C.: Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In: Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems. ICDCS '10, Washington, DC, USA, IEEE Computer Society (2010) 62–73