Journal of Information & Computational Science 8: 4 (2011) 716–724
Available at http://www.joics.com

An Energy-efficient Scheduling Approach Based on Private Clouds ⋆

Jiandun Li [a], Junjie Peng [a,b,*], Zhou Lei [a], Wu Zhang [a]

[a] *School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China*

[b] *Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Science, Beijing 100190, China*

Abstract

With further development and wide acceptance of cloud computing, lots of companies and colleges decide to take advantage of it in their own data centers, which is known as private clouds. Since private clouds have some unique characteristics and special requirements, it is still a challenging problem to effectively schedule virtual machine requests onto compute nodes, especially with multiple objectives to meet. In this paper, we explore couples of characteristics related to workflow scheduling in the scenario of private clouds and propose a hybrid energy-efficient scheduling approach. The experiments show that it can save more time for users, conserve more energy and achieve higher level of load balancing.

Keywords: Energy-efficient; Load Balancing; Cloud Computing; Private Clouds

1 Introduction

Cloud computing is a new promising paradigm in distributed and parallel computing. It can offer utility-oriented IT services based on a pay-as-you-go model. It has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased" [1]. It can make good use of economies of scale and dynamically deliver/configure almost any IT related services on demand [2]. Moreover, it can conserve more energy, making the earth greener [3].

Due to so many attractive characteristics, recently, besides those big companies of influential, lots of companies and colleges have made their minds to make good use of it. Before rushing to

⋆Project supported by Shanghai Leading Academic Discipline Project (No. J50103), Ph.D Programs Fund of Ministry of Education of China (No. 200802800007), Key Laboratory of Computer System and Architecture (Institute of Computing Technology
Chinese Academy of Sciences), Natural Science Foundation of Shanghai Municipal (No. 10ZR1411600) and Innovation fund of Shanghai University (No. B16010809007).

*Corresponding author.
Email address: jjie.peng@shu.edu.cn (Junjie Peng).

1548–7741/ Copyright © 2011 Binary Information Press
April 2011

transplant the whole system to a third-party commercial cloud, most of them choose to apply this new paradigm to their own data centers first, which is known as private clouds.

In private clouds, virtual machine (VM) based applications for daily uses are getting popular, such as virtual desktop, virtual classroom and virtual laboratory. However, since private clouds have some unique characteristics and special requirements, it is still a challenging problem to effectively schedule VM requests onto compute nodes, especially with multiple objectives to meet. This problem cannot be overlooked, for it could directly affect the scaling capacity of the entire system, which is one of the key features of clouds. The problem of VM scheduling has been widely studied in previous researches; however it is always limited in the scenario of cluster computing, grid computing or commercial clouds, with one or two objectives considered simultaneously.

In this paper, after exploring couples of characteristics related to VM workflow scheduling in private clouds, we outline three metrics for scheduling and propose a hybrid energy-efficient scheduling approach, which is based on pre-power technique and least-load-first algorithm. The experiments show that this approach can save more time for users, conserve more energy and achieve higher level of load balancing.

The remainder of the paper is organized as follows. Section 2 explores couples of characteristics related to VM workflow scheduling in private clouds and presents the target for scheduling; section 3 describes the energy-efficient scheduling approach in detail; section 4 gives some experiments and comparisons; section 5 discusses some related work and section 6 concludes this paper.

2 VM Workflow Scheduling in Private Clouds

As the scenario varied from cluster computing, grid computing and commercial clouds to private clouds, something different rises up. As related to VM workflow scheduling, there are some new features, such as the energy-performance tradeoff, time-wise curve, scaling characteristic. However, one of the key concerns remained should be the tradeoff problem between energy consumption and performance. Other than addressing it on processor basis (e.g. DVS [4]), we are more interested in handling it from operating system level, or from scheduling point of view. Nonetheless, it is still a challenging problem to effectively schedule VM requests onto compute nodes, especially with multiple objectives to meet.

2.1 Characteristics of VM Workflow Scheduling in Private Clouds

In this paper, we refer workflows as VM request workflows for daily use, such as virtual desktop, virtual classroom and virtual laboratory. They differ greatly from high performance computing (HPC) job workflows (e.g. [5]), which require more than one VM at a time and could last for couples of days or even longer. In this manner, attributes of the workflow (e.g. scale, sparseness) would present some normal patterns cyclically. Here we assume that they obey Gaussian-distribution within a cycle. This assumption is reasonable because it is the most regular distribution in real world, which can depict most of the group behaviors. Additionally, since private clouds serve people for daily use, no one's instance is more important than others (i.e. no priority); thus, one instance should not be suspended for another one, i.e. intrusion is not accepted. It is known that one VM would experiences pending, extant and teardown three main

phases in general. In this paper we are more concerned on the switch from pending to extant, which may take a while if the target host is running on low-power mode. Thus we argue that, if the response time is too long (e.g. couples of minutes), it is not acceptable for our particular scenario. From this point of view, the scheduling algorithm should be light-weighted (e.g. less than $O(N)$). Additionally, request queue is also beyond consideration, so as to deliver a rapid answer and provision the access to users in seconds. Moreover, it is obvious that, without further optimization or consolidation (e.g. migration) one-time or first-time scheduling can barely balance the workloads across all hosts; however, applying migration could in the same time incur extra overhead, introduce more migrations, or even interrupt user's current use, especially for daily use most of which is interactive-centric. Therefore, it is not considered in current work. To sum up, we argue that workflow scheduling in private clouds could have the following features.

- Intrusion is not accepted.
- Workflow attributes (e.g. scale, sparseness) regularly changes cyclically.
- Conserving energy, whereas keeping response time too long is beyond consideration.
- The scheduling algorithm should be light-weighted.
- Further optimization or consolidation (e.g. migration) should be cautiously applied.

2.2 Target for VM Workflow Scheduling in Private Clouds

Considering the special feature and particular requirement discussed above, we outline three metrics for scheduling (i.e. response time, conserved energy and load balancing). Response time represents the time elapsed from VM request to ready to access. Conserved energy is the total amount of seconds running on low-power mode among all nodes. Load balancing used to be one of the key metrics in job scheduling, which simply requires workloads to be equally shared across all compute nodes; nonetheless, in this paper it means much more. To be specific, it is as much as a combined metric, in which both load balancing and energy conservation should be considered.

Assume there are N nodes and each node can host at most M VMs. Complete set $\mathbf{U}$ consists of all compute nodes. Subset $\mathbf{S}$ of $\mathbf{U}$ is composed of idle nodes that have no workload at a particular time slot. And we introduce two integer functions to illustrate the state of compute nodes, where $F(i)$ represents the scale of workloads on node i, and $Z(i)$ represents the awake/asleep state of node i. The value of $F(i)$ is integer, which would range from 0 to M, whereas the value of function $Z(i)$ is binary (i.e. "0" (awake), "1" (asleep)). Assume that $Z(i) = 0$ when $F(i) > 0$.

Paradigm 1 *Each node in* $\mathbf{U}$ *satisfy* $F(i) > 0$*, or* $Z(i) = 1$*, i.e.*

$$\prod_{i \in \mathbf{U}} (F(i) + Z(i)) > 0. \tag{1}$$

Paradigm 2 *On the basis of Eq. (1), scale of* $\bar{\mathbf{S}}$ *cannot further shrink, i.e.*

$$M(\left|\bar{\mathbf{S}}\right| - 1) < \sum_{i \in \bar{\mathbf{S}}} F(i) \leq M(\left|\bar{\mathbf{S}}\right|) \tag{2}$$

Paradigm 3 *On the basis of Eq. (1) and Eq. (2), workloads have been equally distributed across* $\bar{\mathbf{S}}$*, i.e.*

$$\sum_{i\in\bar{\mathbf{S}},j\in\bar{\mathbf{S}},i\neq j}(|F(i)-F(j)|)+\sum_{k\in\mathbf{S}}(|Z(k)-1|)=0 \tag{3}$$

Since there is a chance where workloads are not equally distributed technically, but cannot be further balanced because of the integrity of VM, we relax Paradigm 3 as Paradigm 4.

Paradigm 4 *On the basis of Eq. (1) and Eq. (2), workloads cannot be further balanced, i.e.*

$$\sum_{i\in\bar{\mathbf{S}},j\in\bar{\mathbf{S}},i\neq j}(|F(i)-F(j)|)(|F(i)-F(j)|-1)+\sum_{k\in\mathbf{S}}(|Z(k)-1|)=0 \tag{4}$$

3 An Energy-efficient Scheduling Approach Based on Private Clouds

Specifically, considering previous studies and current platforms, there are at least two challenging problems in VM scheduling: a) how to reduce the coming request's response time, and b) how to balance the workloads, when the datacenter is running on low-power mode. These two problems happen because: a) powering up a sleeping node via remote access is kind of time-consuming; b) traditional energy-efficient algorithms always fall short to sharing workloads across hosts. Aiming at these two problems, we propose a hybrid energy-efficient scheduling approach, which is comprised of pre-power technique and least-load-first algorithm.

Traditionally, idle hosts are sent to asleep, when the idle duration passed the given idle threshold. Accordingly, the whole system is ramped down and energy is saved. When current capacity is running out, the system has to grow in order to fit user's requests. However, the growing process would take couple of minutes, which is not acceptable.

To reduce the response time, we apply a pre-power technique. In our approach, we first set a desired spectrum for the left capacity (i.e. available capacity of awake hosts) and what we would do next is by all means to keep the left capacity in this spectrum. In this manner, powering up or down a node is not passively controlled by the idle threshold, which is difficult to set, but fully controlled by cloud scheduler. When the left capacity of current private cloud is running low (not running out), a power-up command would be send to one or more asleep nodes to wake up. Therefore, the left capacity scales up and the coming request would not have to wait that much any more. When the left capacity overflows the desired spectrum, a power-down command would be send to one or more awake nodes to asleep. Thus, the left capacity scales down and more energy would be saved than controlled by the idle threshold. In a word, we do not use the idle threshold to save energy, but make the left capacity a little more than the current workloads, and keep this margin by forcibly powering up or down.

It is always a tradeoff problem between energy conservation and load balancing. Let's examine two classic algorithms first. One of the traditional ways to balance workloads across compute nodes is Round-robin (RR) [6], which assigns workloads to each host in equal portions and in circular order, handling all nodes without priority. It is simple and easy to apply to application, but not energy-efficient. Another traditional way to conserve energy would be Greedy [7], which

schedule workloads on the first node and sticks to it until it is saturated. It can save much energy by turning the idle host down when it passed the given threshold; however, the sharing of workloads across nodes is neglected.

Through the desired spectrum of left capacity and the least-load-first algorithm, we propose a hybrid energy-efficient scheduling approach. Procedure of this approach is show in Fig. 1. And the time complexity has been limited to $O(N)$, which is light-weighted.

```
//Preprocessing
Set default value to N, M, F(i), Z(i), S, LC,
UL, LL;
Loop
//Resource Selection
        tempCapacity = M;
        tempR = 0;
        Loop i in U-S
                If ( F(i) < tempCapacity )
                        tempCapacity = F(i);
                        tempR = i;
                        i++;
                End If
        End Loop
```

```
//Resource provision
        Run VM instance on i;
        F(i) ++;
//Post-processing
        If ( LC < LL )
                Power up nodes in S;
        Else If ( LC > UL )
                        Power down nodes in
U-S;
                End If
        End If
        Update S;
End Loop
```

Fig. 1: Procedure of Request Scheduling

4 Experiments

We present series of experiments that illustrate the effect of using the scheduling approach. The testbed is composed of 4 personal computers (HP Compad dc 7900), each of which has 4 cores (Intel Core 2 Quad Q8400 2.66 GHz 2.67 GHz), $4GB$ memory and $300GB$ Hard disk, connected with $100Mbps$ switched Ethernet. We select one computer to be the cloud scheduler and make the remainder as compute nodes, which can host at most 4 VMs. And we use Eucalyptus [8] to setup the base environment for private clouds. We generate a simulation workflow, which can simulate VM requests within a cycle. The workflow consists of 16 requests distributed as $N(2400, 1550)$. And we set the desired spectrum for left capacity to $[2, M]$. We take our approach, RR and Greedy respectively to schedule the workflow (See Fig. 2, Fig. 3 and Table 1).

Table 1: Comparisons on different metrics

Metrics	Greedy (s)	RR (s)	Our approach (s)
Conserved energy	5064	0	5432
Average response time	54.25	14.81	15.44
Paradigm 4	1508	660	2802
Paradigm 2	2938	660	3153
Paradigm 1	4512	4626	5512

Response time Greedy is the worst on this metric. In order to save more energy, it sends all the idle nodes to asleep and wakes a new one when it has to; so the coming request has to wait a long time (about 4–6 minutes). Fig. 2. d highlights this phenomenon. Additionally, RR is slightly better than our approach and this is reasonable because $O(1)$ is better than $O(N)$.

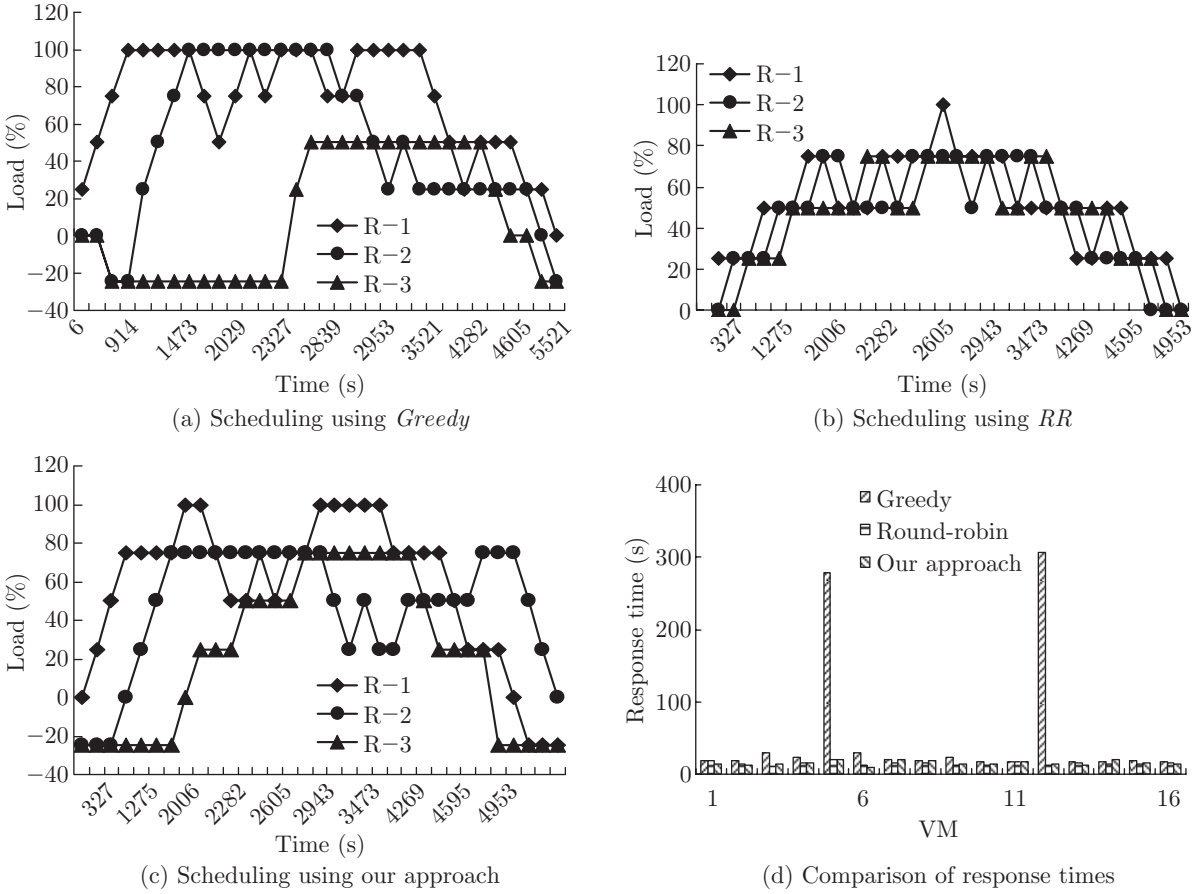

Fig. 2: VM Workload Scheduling

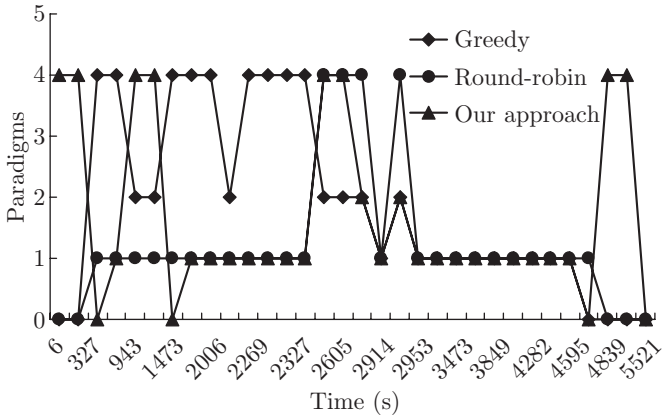

Fig. 3: Dynamic Curves of Paradigm Achievement using Different Approaches

Energy conservation Scheduling using algorithm RR, no energy is saved, whereas Greedy and our approach conserved quite a number of machines. In the very beginning of Greedy, idle nodes ($R-1$ and $R-2$) remained to be awake until the duration passed the given threshold ($300s$), where we can make it better in our approach. At the very start of our approach, nodes beyond the margin ($[2, M]$) are directly set to asleep, leaving only one node ($R-1$) to provision resources. However, after the third request shows up, our approach instantly wakes up an extra node ($R-2$) to maintain the margin. Accordingly, the fourth request would be scheduled to run on $R-2$

according to the least-load-first algorithm. Compared with Greedy at this time slot, one more host is used, but it greatly reduce the response time of the fifth request. Moreover, since our approach does not have to wait until passed the idle threshold, the overall performance on energy conservation is better than Greedy, let alone RR (See Fig. 2, Table 1).

Load balancing As in Fig. 2, our approach is just behind RR, leaving Greedy behind. However, for the combined metric of energy-efficient and load balancing, the situation is completely different. From Fig. 3, we can see that, the curve of RR looks like a pyramid. For most of the time it runs on Paradigm 1 or lower, with just a tiny portion on Paradigm 4 (when the scale of workloads close to total capacity). As related to Greedy, the situation is better, which probably distributed equally on three levels. RR and Greedy appear alike in their beginning and ending, both of which leave idle hosts to be awake, where Paradigm 1 cannot be met. In contrast with them, our approach is more energetic, which experiences from 0 to Paradigm 4 all time long. Moreover, other than concentrated at the axe, achievement of our approach focuses more around the edges. It seems to be even; however, it is deserved to note that this is a workflow of Gaussian-distribution, in which there is a huge gap between scales around the axe (wave crest) and the edges (wave trough). Distance near the troughs means much more time than around the crests. For there is less requests around the edges, it is the key area for energy conservation (See Table 1).

5 Related Work

There are a few studies which are related to our research. S. K. Garg et al [9] proposed a few scheduling policies that could exploit heterogeneity (e.g. energy cost, carbon emission rate) across multiple datacenters, author A. Verma et al [10] also presented a dynamic power-aware strategy and B. Lawson et al [11] gave a two-level policy to adaptively lower/raise the number of active processors. However all of them are based on HPC applications, which are quite different from daily use discussed in previous sections, so they are not suitable for our research. G. Tesauro et al [12] presented a valuable reinforcement learning approach for real-time management of power consumption. For it is based on a machine learning algorithm, requiring quite a lot of time to learn, so it is too time-consuming for daily use. By taking advantage of the min, max and shares features in virtualization, M. Cardosa et al [13] provided a smooth mechanism for the energy-performance tradeoff problem. They cared much about the grain of VMs, trying to make sure that high priority applications get more resources. In our work, daily VMs are deemed as the same, there is no priority exists. Using workload history and predicting future workload, D. Bradley et al [14] minimized the power utilization in some cases, however this is only reliable in limited scenarios, and it would incur great damage when the pattern derived from the history failed. Compared with our approach, multi-layer based [15] [16] [17] or agent based [18] energy management strategy is too complex, failing to meet the response time in SLA, especially with great number of compute nodes. J. O. Kephart et al [19] particularly handled the tradeoff problem through two schedulers, performance scheduler and power scheduler, reducing the power approximately by 10 %; however the approach, composed of reinforcement learning, is also kind of time-consuming, so it is not appropriate for scenarios in this paper.

A. Beloglazov et al [20] and H. N. Van et al [21] addressed the tradeoff problem using continuous consolidation and migration, which would definitely upgrade the scale of overhead. M. Steinder et al [22] presented a further idea by make good use of SLA by continuous approximation to

achieve an optimal tradeoff. This is a brilliant idea; however, it may consume extra time and the optimal decision could not be achieved without migration.

6 Conclusion

As the scenario varied from cluster computing, grid computing and commercial clouds to private clouds, something different happened with VM workflow scheduling, especially for daily use. In this paper, after exploring several characteristics about VM workflow scheduling in this particular scenario, we propose an energy-efficient scheduling approach and the experiments show that it can save more time for users, conserve more energy and achieve higher level of load balancing.

Next, we would test our approach based on hardware workload, e.g. CPU, memory, and evaluate the effectiveness of VM migration so as conserve more energy.

References

[1] M. Armbrust, A. Fox, R. Griffith et al, Above the clouds: a berkeley view of cloud computing, *Technical Report No. UCB/EECS–2009–28*, University of California at Berkley, 2009, 1-23

[2] I. Foster, Y. Zhao, I. Raicu, S. Lu, Cloud computing and grid computing 360-degree compared, *In Proc. of GCE'08*, 2008 , 1-10

[3] J. Li, W. Zhang, J. Peng, Z. Lei, Y. Lei, A carbon 2.0 framework based on cloud computing, *In Proc. of ICIS 2010*, Kaminoyama, Japan, 2010, 153-158

[4] C. Lefurgy, K. Rajamani, F. Rawson et al, Energy management for commercial servers, *Computer*, 2003, 36(12), 39-48

[5] X. Xu, C. Cheng, J. Xiong, Reliable integrated model of cloud & client computing based on multi-agent, *Journal of Computational Information Systems*, 2010, 4767-4774

[6] Round-robin (RR) on Wikipedia: http://en.wikipedia.org/wiki/Round-robin_scheduling

[7] Greedy on Wikipedia: http://en.wikipedia.org/wiki/Greedy_algorithm

[8] D. Nurmi, R. Wolski, C. Grzegorczyk, et al, The eucalyptus open-source cloud-computing system, *In Proc. of CCGRID'09*, 2009, 124-131

[9] S. K. Garg, C. S. Yeo, A. Anandasivam, R. Buyya, Environment-conscious scheduling of hpc applications on distributed cloud-oriented data centers, *Journal of Parallel and Distributed Computing*, 2010, 1-41

[10] A. Verma, P. Ahuja, A. Neogi, Power-aware dynamic placement of HPC applications, *In Proc. of ICS'08*, Island of Kos, Aegean Sea, Greece, 2008, 175-184

[11] B. Lawson, E. Smirni, Power-aware resource allocation in high-end systems via online simulation, *In Proc. of ICS'05*, Cambridge, USA, 2005, 229-238

[12] G. Tesauro, R. Das, H. Chan, et al, Managing power consumption and performance of computing systems using reinforcement learning, *In Proc. of NIPS'07*, 2007

[13] M. Cardosa, M. R. Korupolu, A. Singh, Shares and utilities based power consolidation in virtualized server environments, *In Proc. of IM 2009*, 2009, 327-334

[14] D. Bradley, R. Harper, S. Hunter, Workload-based power management for parallel computer systems, *IBM Journal of Research and Development*, 2003, 47(5), 703-718

[15] A. Verma, P. Ahuja, A. Neogi, pmapper: Power and migration cost aware application placement in virtualized systems, *In Proc. of MIDDLEWARE 2008*, Leuven, Belgium, 2008, 243-264

[16] I.Whalley, A. Tantawi, M. Steinder et al, Experience with collaborating managers: Node group manager and provisioning manager, *Journal of Cluster Computing*, 2006, 9(4), 401-416

[17] J. Stoess, C. Lang, F. Bellosa, Energy management for hypervisor-based virtual machines, *In Proc. of USENIX'07*, Santa Clara, CA, 2007, 1-14

[18] R. Das, J. O. Kephart, C. Lefurgy et al, Autonomic multi-agent management of power and performance in data centers, *In Proc. of AAMAS'08*, 2008, 107-114

[19] J. O. Kephart, H. Chan, R. Das, Coordinating multiple autonomic managers to achieve specified power-performance tradeoffs, *In Proc. of ICAC'07*, Jacksonville, Florida, USA, 2007, 24-33

[20] A. Beloglazov, R. Buyya, Energy efficient resource management in birtualized vloud fata venters, *In Proc. of CCGRID 2010*, Melbourne, Australia, 2010, 826-831

[21] H. N. Van, F. D. Tran, J. M. Menaud, Performance and power management for cloud infrastructures, *In Proc. of CLOUD 2010*, Miami, Florida, USA, 2010, 329-336

[22] M. Steinder, I. Whalley, J. E. Hanson, J. O. Kephart, Coordinated management of power usage and runtime performance, *In Proc. of NOMS 2008*, 2008, 387-394