A Task Scheduling and Resource Allocation Algorithm for Cloud using Live Migration and Priorities

Harpreet Kaur
Student
Department of IT
Lovely Professional University

Maninder Singh
Assistant Professor
Department of CSE
Lovely Professional University

ABSTRACT
A cloud environment is one of the most shareable environments where multiple clients are connected to the common environment to access the services and the products. A cloud environment can be public or the private cloud.In such case, some approach is required to perform the effective scheduling and the resource allocation. The situation becomes critical when the cloud server is overloaded, in such case to provide the effective service to the client, the process migration is done. The migration is the concept to switch the allocated process to some other virtual machine or the cloud to release the load and to perform the effective execution of cloud requests. The obtained results shows the effective allocation of the process to the associated cloud in overload and the under load conditions.

Keywords
Cloud computing, migration, process, resource utilization, scheduling, virtual machine, virtualization

1. INTRODUCTION
Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. It is composed of five essential characteristics**,** three service models, and four deployment models [1] :

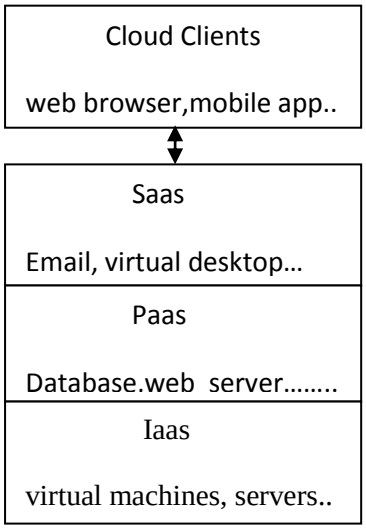

Fig 1 : Three Layers of Cloud

1. Cloud Software as a Service (SaaS) : The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

2. Cloud Platform as a Service (PaaS) : The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations.

3. Cloud Infrastructure as a Service (IaaS) : The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls).

1.1 Characteristics of Cloud Computing :
Cloud computing exhibits various key characteristics like Cost is claimed to be reduced and in a public cloud delivery model capital expenditure is converted to operational expenditure. This is purported to lower barriers to entry, as infrastructure is typically provided by a third-party and does not need to be purchased for one-time or infrequent intensive computing tasks. Pricing on a utility computing basis is fine-grained with usage-based options and fewer IT skills are required for implementation. Device and location independence enable users to access systems using a web browser regardless of their location or what device they are using (e.g., PC, mobile phone). As infrastructure is off-site (typically provided by a third-party) and accessed via the Internet, users can connect from anywhere .Virtualization technology allows servers and storage devices to be shared and utilization be increased. Applications can be easily migrated from one physical server to another. Reliability is improved if multiple redundant sites are used, which makes well-designed cloud computing suitable for business continuity and disaster recovery. Scalability and elasticity via dynamic ("on-demand") provisioning of resources on a fine-grained, self-service basis near real-time , without users having to engineer for peak loads. Performance is monitored, and consistent and loosely coupled architectures are constructed using web services as the system interface. Maintenance of cloud computing applications is easier,

because they do not need to be installed on each user's computer and can be accessed from different places .

Cloud computing possess the following key characteristics [1] :

1.On-demand self-service : A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.

2.Broad network access : Cloud computing provide the users with various capabilities over the network which are accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops etc.)

3.Resource pooling : The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.

4.Rapid elasticity : Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out, and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

5.Measured Service : Cloud systems automatically control and optimize resource use by leveraging a metering capability1 at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

1.2 Types of Cloud

There are four types of clouds [1]

1.2.1 Private cloud: The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

1.2.2 Community cloud: The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise.

1.2.3 Public cloud: The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

1.2.4 Hybrid cloud: The cloud infrastructure is a composition of two or more clouds(private, community or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability.

1.3 Scheduling

Scheduling is the method by which threads, processes or data flows are given access to system resources (e.g. processor time, communications bandwidth). This is usually done to load balance a system effectively or achieve a target quality of service. The need for a scheduling algorithm arises from the requirement for most modern systems to perform multitasking (execute more than one process at a time) and multiplexing (transmit multiple flows simultaneously).The scheduler is concerned mainly with [18] :

1.Throughput : The total number of processes that complete their execution per time unit.

2.Latency, specifically :

- **Turnaround time** - total time between submission of a process and its completion.
- **Response time -** amount of time it takes from when a request was submitted until the first response is produced.

3.Fairness / Waiting Time : Equal CPU time to each process (or more generally appropriate times according to each process' priority). It is the time for which the process remains in the ready queue.

2. PRESENT WORK

2.1 Introduction

There are an increasing number of Cloud Services available in the Internet. Cloud services can be a component of a system and different Cloud Servers that would provide different services. In this present work we have defined a multiple cloud environment. Each cloud server is defined with certain limits in terms of memory and the cpu specifications. Now as the users enter to the system, the user request is performed in terms of processes. To represent the parallel user requests, n number of requests are been generated by the users. All these requests are to be handled by the Cloud servers in parallel by using the multiple Cloud concept. A middle layer is defined between the Cloud servers and the client requests that will perform the allocation of the processes to different Clouds in underload and over load conditions. As user requests are performed, some parameters are also defined with each request. These parameters are the process time, deadline, input output specifications etc. In the general case, the allocation of the processes are performed in a sequential order. Each process must be executed within the deadline limit. But if more than one processes occur at same time and not get executed before the deadline, in such case the processes is switched from one Cloud server to other called the process migration. In this present work, a parametric analysis is performed to identify the requirement of process migration and based on this analysis the migration will be performed on these processes. The effectiveness of the work is identified in terms of successful execution of the processes within the time limits.

2.2 Problem Definition

One of the major challenge in a Cloud computing environment is the Service Level Agreement (SLA). SLA basically deals with the effective allocation of the resource so that it will be affected by any overloading and underloading conditions.The overloading condition generate the congestion over the resource that results the bad service or no service

situation. The over load condition is defined as the number of parallel handling of the user requests by a particular Cloud server or the virtual machine. As the load on a server increases, the more complications the server has to face. These complications include the management of extra hardware, queues etc. Even then, many of the processes cannot be executed before the deadline. Because of this, there is the requirement of such algorithm that can handle the overload conditions and execute the maximum number of processes before the deadline.

2.3 Motivation

There are an increasing number of Cloud services available in the Internet. Cloud services can be a component of a system and different Cloud Services would provide different services. To fit different requirements from different clients, different cloud servers are available to provide the relative Cloud services to the users. To achieve this, an efficient process allocation algorithm is required. Several Cloud are combined to a common environment to provide the effective allocation without delay. The presented work is capable to handle the overload condition. To handle the overload condition, the load distribution scheme is presented in this work.

2.4 Formulation of Hypothesis

There are different service providers that provide Cloud Services to different users for different business services. These services are used by the user as well as the programmer in their applications. But now there is also the integration of different services as the single unit. For this interfacing and composition we are providing an approach where we are defining an intermediate state between the clouds servers and the users. We are proposing a middle layer Architecture called Intermediate Layer. In this concept we are placing a layer between the users and the web services. The middle layer will accept the user requests and also monitor the cloud servers for the available load over the services. The middle layer will perform the cloud allocation sequentially and if the service allocation is not possible for a specific cloud, it will perform the migration of process from one cloud to other.

2.5 Objectives

The proposed System will achieve the following objectives:

1. Create An Intermediate Architecture that will accept the user request and monitor the Cloud servers for their capabilities.

2. Scheduling of the users requests is performed to identify the order of allocation of the processes.

3. Performing the effective resource allocation under defined parameters and the Cloud server capabilities.

4. Define a dynamic approach to perform the process migration from one Cloud to other.

5.Analysis of the work using graph under different parameters.

2.6 Research Design

The proposed system is a middle layer architecture to perform the cloud allocation in case of underload and overload conditions. The over load conditions will be handled by using the concepts of process migration. The middle layer will exist between the clouds and the clients. As the request will be performed by the user this request will be accepted by the middle layer and the analysis of the cloud servers is performed by this middle layer. The middle layer is responsible for three main tasks.

1. Scheduling the user requests.

2. Monitor the cloud servers for its capabilities and to perform the process allocation.

3. Process Migration in overload conditions.

3. RESULT AND DISCUSSIONS

3.1 Algorithm

1. Input M number of Clouds with L number of Virtual Machines associated with Each cloud.

2. Define the available memory and load for each virtual machine.

3. Assign the priority to each Cloud.

4.Input N number of user process request with some parameters specifications like arrival time, process time, required memory etc.

5. Arrange the process requests in order of memory requirement

6. For i=1 to N

7. {

8. Identify the priority Cloud and Associated VM having AvailableMemory > RequiredMemory(i)

9. Perform the initial allocation of process to that particular VM and the Cloud

10.}

11. For i=1 to N

12. {

13. Identify the Free Time slot on priority cloud to perform the allocation. As the free slot identify, record the starttime, process time, turnaround time and the deadline of the process.

14. }

15. For i=1 to N

16. {

17. If finishtime(process(i))> Deadline(Process(i))

18. {

19. Print “Migration Required”

20. Identify the next high priority Cloud, that having the free memory and the time slot and perform the migration of the process to that particular cloud and the virtual machine.

21. }

22. }

Table 3.1 : List of Parameters

User ID	arrival time	process time	deadline	I/O req.	Mem req.	type of service
1	9	3	18	4	30876	0
2	97	10	112	4	4540	0
3	91	8	109	3	1142	1
4	93	7	108	3	12551	1
5	17	8	26	1	1477	0
6	82	7	93	4	1102	0
7	38	8	54	0	15672	0
8	64	8	80	1	21750	1
9	16	2	23	4	10892	1
10	22	8	33	2	22370	1

Values of Arrival time, Process time, Deadline, I/O request, memrequest and Typeofservice are computed to schedule requests of users.

Table 3.2 : List of Parameters (sort according to memory)

user ID	Arrival time	process time	Deadline	I/O req	Mem req	Type Of service
6	82	7	93	4	1102	0
3	91	8	109	3	1142	1
5	17	8	26	1	1477	0
2	97	10	112	4	4540	0
9	16	2	23	4	10892	1
4	93	7	108	3	12551	1
7	38	8	54	0	15672	0
8	64	8	80	1	21750	1
10	22	8	33	2	22370	1
1	9	3	18	4	30876	0

User process requests are sorted according to memory.

Table 3.3 : Memory

Cloud No.	Virtual machine no. associated with cloud							
	1	2	3	4	5	6	7	8
1	12847	3777	10250	9630	1124	32000	32000	320
2	32000	32000	32000	32000	32000	32000	32000	320
3	32000	32000	32000	32000	32000	32000	32000	320
4	32000	32000	32000	32000	32000	32000	32000	320
5	32000	32000	32000	32000	32000	32000	32000	320

Values of remaining memory after allocation of processes to particular VM's and clouds.

Table 3.4 : List of Parameters (calculate wait time)

user ID	cloud	virtual mach-ine	Arrival time	Start time	Turnaround time	Finish time	deadline	Wait time
6	1	1	82	82	7	89	93	0
3	1	1	91	91	8	99	109	0
5	1	1	17	20	8	28	26	3
2	1	1	97	100	10	110	112	3
9	1	1	16	16	2	18	23	0
4	1	2	93	111	7	118	108	18
7	1	2	38	39	8	47	54	1
8	1	3	64	64	8	72	80	0
10	1	4	22	29	8	37	33	7
1	1	5	9	9	3	12	18	0

Values of user ID, Cloud, VM, arrival time ,start time , turnaround or process time ,finish time , deadline and wait time of various processes .

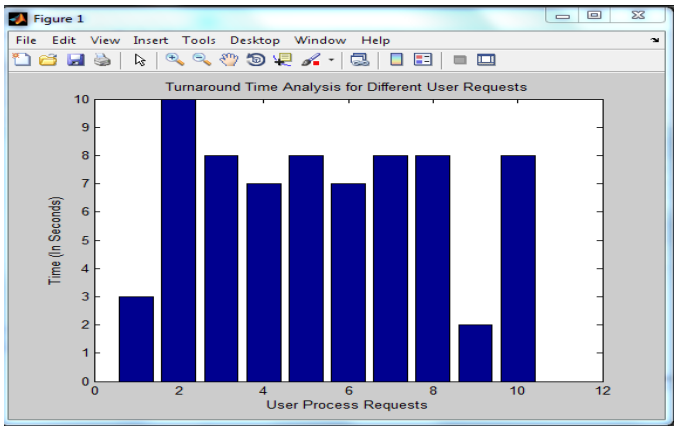

Fig 4.1 : Turn Around Time Analysis

Here figure 1 is showing the turnaround analysis for 10 processes. Here x -axis represents the number of user requests and y- axis represents the time taken by these processes in seconds. The figure is showing the process time required by each process.

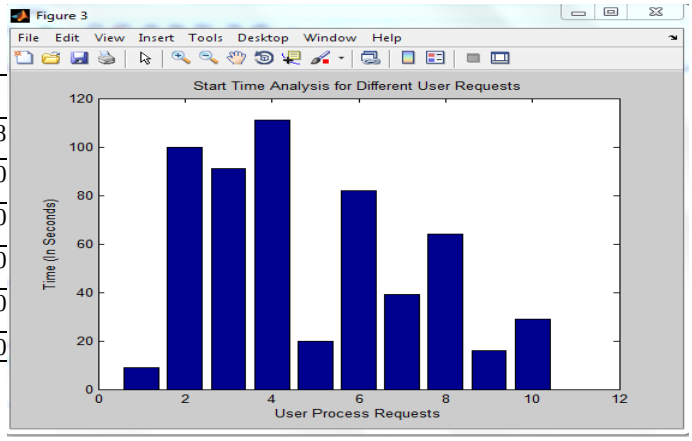

Fig 4.2 : Start Time Analysis

Here figure 2 is showing the Start Time analysis for 10 processes. Here x-axis represents the number of user requests

and y-axis represents the time taken by these processes in seconds. The figure is showing the Start time of each process.

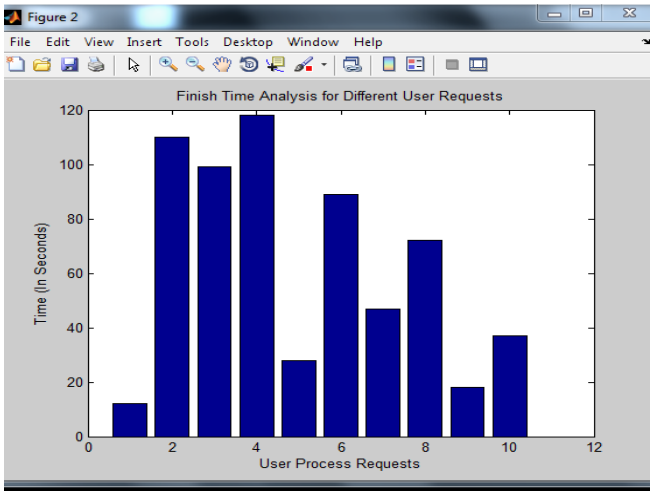

Fig 4.3 : Finish Time Analysis

Here figure 3 is showing the Finish Time analysis for 10 processes. Here x-axis represents the number of user requests and y-axis represents the time taken by these processes in seconds. The figure is showing the finish time of each process.

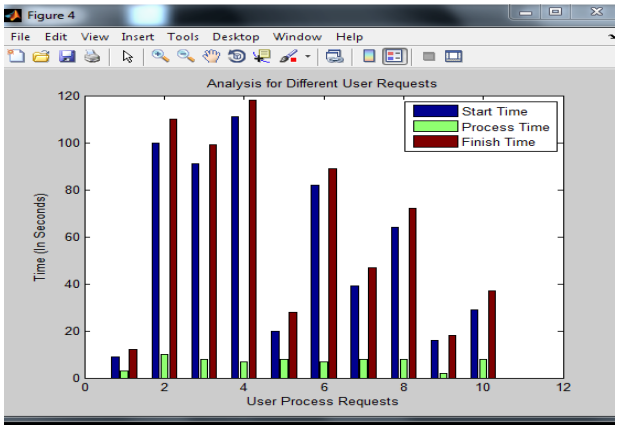

Fig 4.4 : Process Analysis

Here figure 4 is showing the Process analysis for all 10 processes. Here x-axis represents the number of user requests and y-axis represents the time taken by these processes in seconds for all three parameters called process time, finish time and the start time.. The figure is showing these three vectors collectively.

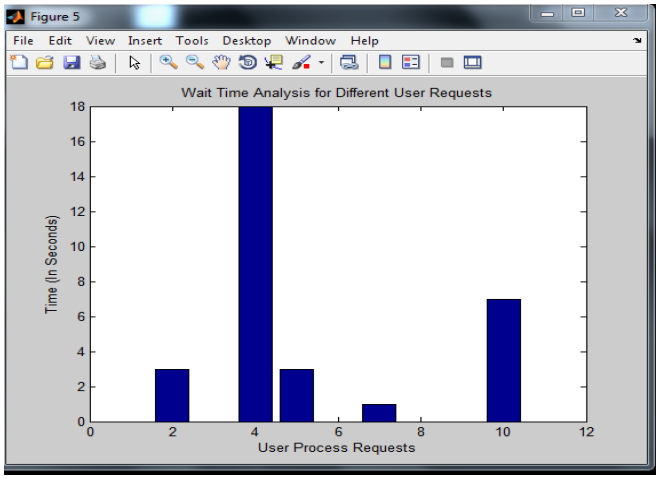

Fig 4.5 : Wait Time Analysis

Here figure 5 is showing the Wait Time analysis for 10 processes. Here x-axis represents the number of user requests and y-axis represents the time taken by these processes in seconds. The figure is showing the wait time of each process. As we can see, most of the processes are executed without any wait. And some are having the nominal wait time.

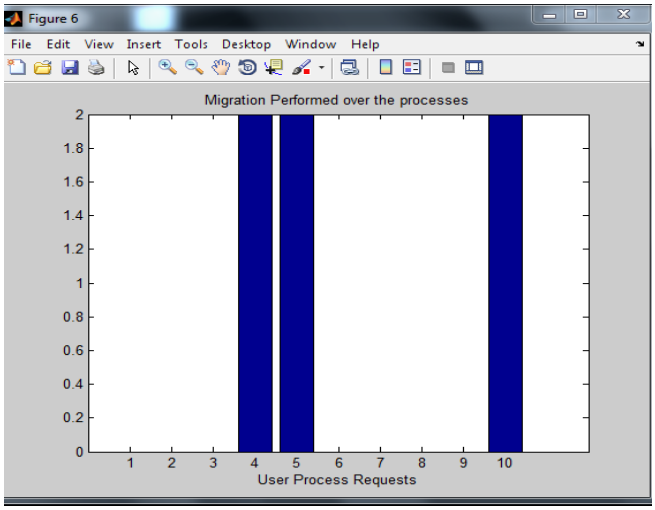

Fig 4.6 : Migrated Processes

Here figure 6 is showing the list of migrated processes . Here x-axis represents the number of user requests and y-axis represents the clouds. As we can see, three processes 4,5 and 10 are migrated to Cloud 2 .

4. CONCLUSION

In this present work, a resource allocation scheme on multiple Clouds in both the under load and the over load conditions. As the request is performed by the user, certain parameters are defined with each user request, these parameters includes the arrival time, process time, deadline and the input output requirement of the processes. The Cloud environment taken in this work is the public cloud environment with multiple clouds. Each Cloud is here defined with some virtual machines.To perform the effective allocation, we have assigned some priority to each cloud. The virtual machines are here to perform the actual allocation. These are defined with certain limits in terms of memory, load etc. As the allocation begins, at first the scheduling of the processes is performed

respective to the memory requirements. And along with it, the allocation of the process is done to the Cloud based on the requirement and the availability analysis. If the allocated process cannot be executed in its required time slot, in such case the migration of the process is required. The migration of the processes is here defined in case of overload conditions. The overload condition is defined in terms of simultaneous processes that are required to execute at particular instance of time. The analysis of the work is done in terms of wait time, process time of the processes. The obtain results shows the successful execution of all the processes within time limit. The work is performed on a generic system that can have n number of Clouds.

The presented work is defined for the public Cloud environment, but in future, the work can be extended to private and the hybrid Cloud environment.

5. ACKNOWLEDGMENT

First and foremost, I would like to express my sincere gratitude to my supervisor Maninder Singh, Associate Professor, Computer Science & Engineering Department for his immense help, guidance, stimulating suggestions and encouragement all the times. He always provided a motivating and enthusiastic atmosphere to work with; it was a great pleasure to do this thesis under his supervision. I am also very thankful to the entire faculty and staff members of Computer Science & Engineering Department for their direct-indirect help, cooperation, which made my stay at L.P.U. memorable. Finally, my special thanks go to my family for their never-ending love and support. Nothing would have ever been possible without my parent's unconditional love and encouragement. Last but not the least; I would like to thank the lord for his blessings and for driving me with faith, hope and courage in the thinnest of the times.

6. REFERENCES

[1] Aaron J. Elmore (2011) " Zephyr: Live Migration in Shared Nothing Databases for Elastic Cloud Platforms", SIGMOD'11, June 12–16, 2011, Athens, Greece. ACM 978-1-4503-0661-4/11/06

[2] Abirami S.P. and Shalini Ramanathan (2012) "Linear Scheduling Strategy for Resource Allocation in Cloud Environment", International Journal on Cloud Computing: Services and Architecture (IJCCSA),Vol.2.

[3] Anton Beloglazov (2010) " Energy Efficient Allocation of Virtual Machines in Cloud Data Centers", 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing 978-0-7695-4039-9/10 © 2010 IEEE.

[4] Anton Beloglazov (2010) " Energy Efficient Resource Management in Virtualized Cloud Data Centers", 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing 978-0-7695-4039-9/10© 2010 IEEE

[5] Balaji Viswanathan (2012) " Rapid Adjustment and Adoption to MIaaS Clouds", Middleware 2012 Industry Track, December 3-7, 2012, Montreal, Quebec, Canada. ACM 978-1-4503-1613-2/12/12

[6] Baocun Hou (2011) " Research on Independent and Dynamic Fault-Tolerant and Migration Technology for Cloud Simulation Resources", GCMS 2011

[7] Bogdan Nicolae (2012) " A Hybrid Local Storage Transfer Scheme for Live Migration of I/O Intensive Workloads", HPDC'12, June 18–22, 2012, Delft, The Netherlands. ACM 978-1-4503-0805/12/06

[8] Boris Danev (2011) " Enabling Secure VM-vTPM Migration in Private Clouds", ACSAC '11 Dec. 5-9, 2011, Orlando, Florida USA ACM 978-1-4503-0672-0/11/12

[9] Damien Borgetto (2012) " Energy-efficient and SLA-Aware Management of IaaS Clouds", e-Energy 2012, May 9-11 2012, Madrid, Spain. ACM 978-1-4503-1055-0/12/05

[10] Hadi Goudarzi (2012) " SLA-based Optimization of Power and Migration Cost in Cloud Computing", 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing 978-0-7695-4691-9/12© 2012 IEEE

[11] J. Brandt (2010) " Using Cloud Constructs and Predictive Analysis to Enable Pre-Failure Process Migration in HPC Systems", 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing 978-0-7695-4039-9/10© 2010 IEEE

[12] Jie Zheng (2011) " Workload-Aware Live Storage Migration for Clouds", VEE'11, March 9–11, 2011, Newport Beach, California, USA. ACM 978-1-4503-0501-3/11/03

[13] Kejiang Ye (2012) " VC-Migration: Live Migration of Virtual Clusters in the Cloud", 2012 ACM/IEEE 13th International Conference on Grid Computing 1550-5510/12 © 2012 IEEE

[14] Kento Sato (2009) " A Model-Based Algorithm for Optimizing I/O Intensive Applications in Clouds using VM-Based Migration", 9th IEEE/ACM International Symposium on Cluster Computing and the Grid 978-0-7695-3622-4/09© 2009 IEEE

[15] Madhan Kumar Srinivasan (2012) " State-of-the-art Cloud Computing Security Taxonomies A classification of security challenges in the present cloud computing environment", ICACCI '12, August 03 - 05 2012, CHENNAI, India ACM 978-1-4503-1196-0/12/08

[16] Michael Menzel (2012) " CloudGenius: Decision Support for Web Server Cloud Migration", WWW 2012, April 16–20, 2012, Lyon, France. ACM 978-1-4503-1229-5/12/04

[17] Mohammad Hajjat (2010) ''Cloudward Bound: Planning for Beneficial Migration of Enterprise Applications to the Cloud", SIGCOMM'10, August 30–September 3, 2010, New Delhi, India. ACM 978-1-4503-0201-2/10/08

[18] Muhammad Ali Babar (2011) "A Tale of Migration to Cloud Computing for Sharing Experiences and Observations", Waikiki, Honolulu, HI, USA. ACM 978-1-4503-0582-2/11/05

[19] Ruijin Zhou (2013) " Optimizing Virtual Machine Live Storage Migration in Heterogeneous Storage Environment", VEE'13, March 16–17, 2013, Houston, Texas, USA. ACM 978-1-4503-1266-0/13/03

[20] Samer Al-Kiswany (2011) " VMFlock: Virtual Machine Co-Migration for the Cloud", HPDC'11, June 8–11, 2011, San Jose, California, USA. ACM 978-1-4503-0552-5/11/06