SLA MANAGEMENT FOR COMPREHENSIVE VIRTUAL MACHINE MIGRATION CONSIDERING SCHEDULING AND LOAD BALANCING ALGORITHM IN CLOUD DATA CENTERS

Merita Kasa Halili[1] *and Betim Cico*[2]

[1] South East European University – Contemporary Sciences and Technology
[2] Epoka University, Computer Engineering Department
e-mails: mk26135@seeu.edu.mk, bcico@epoka.edu.al
[1] Republic of North Macedonia, [2] Albania

Abstract: Cloud Computing conveys infrastructure, platform or application services to enormous users with alternatives and persistent changing requirements. The proposed algorithm in this article tries to avoid/reduce SLA violations through optimal cloudlet scheduling and power optimization by reducing the number of VM migration. Its SLA reduction framework consists of three parts: (a) Scheduling algorithm to efficiently allocate cloudlets to the virtual machines based on the processing time of host, (b) MinVm Scheduling algorithm: Allocates cloudlets to virtual machines based on cloudlet allocation counts to each VM, and (c) Credit-Based VM Migration algorithm, which uses the credit of the VMs to take VM migration.
Keywords: Cloud Computing, CloudSim, SLA, Scheduling algorithm

1. INTRODUCTION

Cloud computing is an architecture that provides various computing resources as service over the internet. The provision of such computing resources over the internet must be scalable and should be provisioned rapidly on-demand. This service provisioning in the cloud computing system is based on Service Level Agreements (SLAs). The SLA represents service contracts signed between service consumer and cloud service provider [17]. Therefore, any SLA violation should be avoided to stay away from the costly penalties. On another hand, service providers must need to ensure efficient utilization of available resources to minimize the cost of resource provisioning. Thus, designing an efficient scheduling policy is the need for any cloud computing system [5].

Generally, cloud users select cloud services after considering certain factors such as performance, price, reputation, security, availability, etc. Cloud computing is a technology that provides online services to many critical business applications, which means that it is necessary to have some reliable mechanism to manage the online contract between service consumer and service provider.

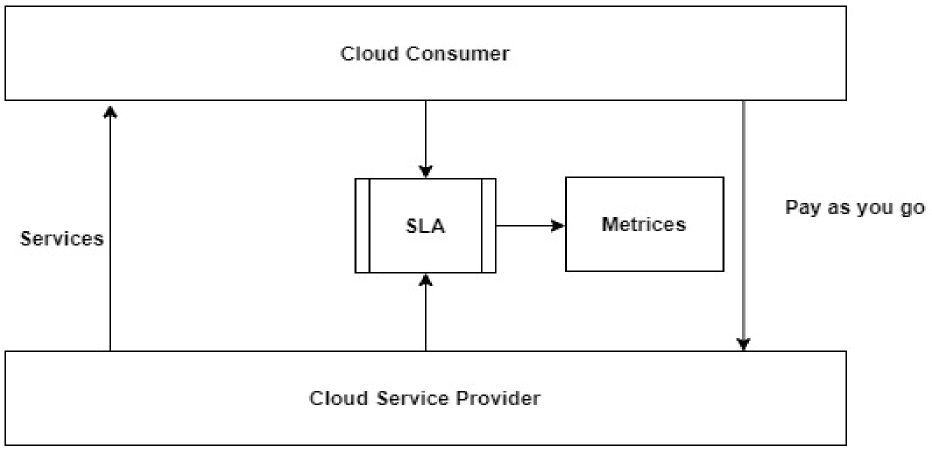


Figure 1. SLA implementation

Figure 1 describes the process of implementing the SLA entity between the Cloud Consumer and Cloud Service Provider. Furthermore, it conveys the Metrics included in our provided algorithm to reduce SLA violation, and indicates the best results from its implementation and maintenance. SLA parameters are defined using cloud computing metrics. These metrics define how service parameters function and how can be measured. Metrics can be classified into two types [10]:

1) Resource metrics: Resource metrics are metrics which are retrieved directly from resource provider and can be used without further processing (e.g., transaction count).

2) Composite metrics: Combination of resource metrics. For example, transaction count per hour requires measurement of transaction count and time.

Table 1 shows common SLA metrics and description [10, 16].

2. STATE OF THE ART

Nowadays, the enormous generation of data reflects in tentative to establish adequate infrastructure for management, migration, scalability, and versatility of the resources in accordance with the adoption to this data. There are 2.5 quintillion bytes of data [11] produced by humans every day, leading to the necessity of creating flexible cloud computing data centers or public clouds. Figure 2 specifies the process of communication between different SLAs in different public clouds, and it conveys an SLA algorithm including certain QoS parameters [6] to meet the user requirements. The communication between public clouds is possible through VM (Virtual Machine) migration techniques that depict as a core standard for managing computing resources, minimizing VM performance overhead, reaching energy efficiency and load balancing in cloud computing [9, 12].

Table 1: SLA metrics description

No.	**Metrics**	**Description**
1	Integration	Integration with other services and platforms
2	Scalability	Increase or decrease in users of the system
3	Pay as you go	Pay for services which you want to access
4	Reliability	Performing or operating as desired in most cases
5	Usability	Easy to use interface
6	Availability	Data and services available throughout the time
7	Storage space	Amount of storage space available for user
8	Security	Security and cryptographic services provided during data transfer and authentication
9	Bandwidth	Transfer capacity of communication channel
10	Billing/cost	Cost of services used
11	Response time	Time taken by cloud service providers to respond to the user request
12	Makespan	Time taken between the start of execution to end

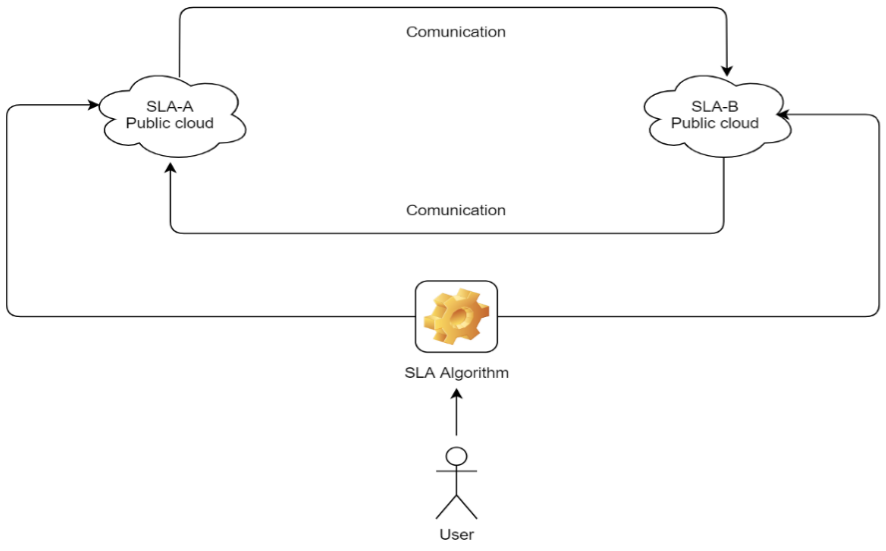


Figure 2. Communication between different Public Clouds

- **Applying scheduling algorithms with QoS in cloud computing [1]**

In this paper, the authors propose a novel SLA-aware scheduling algorithm for CloudSim [15] simulation tool. The author proposes priority-aware scheduling framework for CloudSim. The proposed framework allocates cloudlets to available VMs based on the priorities of the cloudlets. Cloudlets with higher priorities are allowed to execute first. The authors also demonstrate how CloudSim simulator can be used to implement the SLA-aware scheduling algorithm. The authors also state that CloudSim framework does not support priority allocation for cloudlets. However, they show how it could be implemented a priority-aware scheduling algorithm using CloudSim, in the proposed scheme section. Figure 3 shows the working of the SLA-aware priority scheduling algorithm. Cloudlets are scheduled based on their priority. Cloudlets with higher priority get executed first [1].

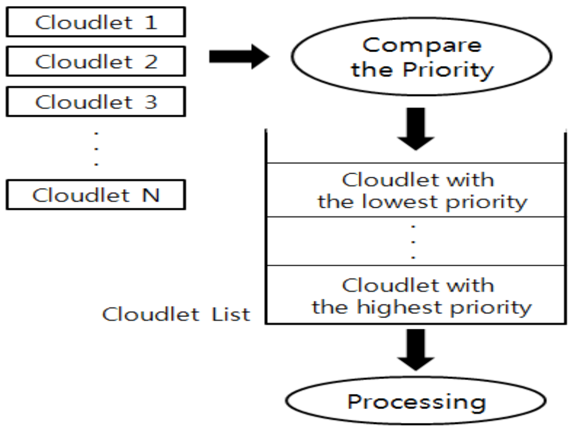


Figure 3. Cloudlet processing priority

- **Selection-based scheduling algorithms under SLA constraints [2]**

In this paper, the authors propose four different scheduling algorithms to address SLA constraints through resource scheduling:

1.Generic first-fit increasing or decreasing allocation of tasks.

This algorithm orders incoming tasks in either increasing or decreasing order of task length. After sorting cloudlets are allocated to VMs through First fit policy i.e., first cloudlet will be allocated to first VM if VM status is IDEAL and capable of handling incoming tasks.

2.Generic weighted allocation of heterogeneous independent tasks.

This algorithm considers the weight of VM during cloudlet allocations. The weight of VM is computed using the following equation, where VMs will be sorted based on the weight of VM:

$$\textit{Weight of VM}=\frac{MIPS\ of\ selected\ VM}{Total\ MIPS\ of\ all\ VMs\ in\ DC} \qquad (1)$$

The algorithm uses the same first fit policy to allocate cloudlets to VM.

3.Generic first-fi allocation of heterogeneous independent tasks with broker centralized SLA constraints, based on a load-aware scheduling strategy.

The Algorithm maintains the SLA contract for the list of the VMs and allocates cloudlets to the dVMs if VM is capable of fulfilling the SLA contract.

4.Generic first-fit allocation of heterogeneous independent tasks by using broker centralized resource weights and SLA constraints.

This algorithm is a combination of 3rd and 4th algorithms. It uses the SLA and weight of VMs to schedule cloudlets to the VMs.

- **SLA-aware application deployment and resource allocation in clouds [3, 8].**

In this paper, the authors present a heuristic scheduling algorithm with an integrated load balancer. The heuristic scheduling algorithm deploys VMs on the host based on the SLA requirements of VMs. The load balancer tries to balance the load on VMs so that there will be no underloading and overloading of resources in the system. The proposed scheduling algorithm and load balancer were implemented using the CloudSim tool.

- **Scheduling service with SLA assurance for private cloud systems [4].**

This algorithm allocates VM to host based on received SLA contracts. The algorithm is implemented as Haizea add-on. The main objective of this algorithm is not only to find a host that fullfills SLA requirements, but also to minimize resource allocation cost. The algorithm takes the resource allocation decision [8] based on the penalty function that will be computed by observing many SLA parameters such as available CPU and memory, minimum and maximum amount of memory and provided CPU.

The above evaluated papers do not express the heterogeneity of services and communication between geo-distant clouds by avoiding SLA-violation. In our contribution, we provide a case study, in which we represent a user who needs to combine 2 public clouds. Each cloud comes with its own SLA, both in terms of speed and other computer resources. A method and algorithm is adapted to both public clouds, so that the cloud with the weakest resources can communicate with the strongest cloud, in terms of speed, energy efficiency [9] and scheduling, and not to miss any important feature, or have any eventual failure.

3. PROPOSED METHODOLOGY

To reduce/avoid SLA violations in the cloud computing system, certain steps are taken using the proposed algorithm. After referring certain papers [1-4], certain parameters are identified which help to reduce SLA violations.

1. Migration Control: By controlling the number of VM migrations, we can achieve better response time in the cloud computing system, because with the minimum number of migrations we can reduce migration time.
2. Energy Efficiency [9]: Reducing power consumption is also helpful in reducing SLA violations. It may happen that due to more power consumption, a host gets hit up and shut down.

3. VmScheduling: Scheduling cloudlets to proper VM is necessary for reducing SLA and helps in achieving better response time. In the proposed SLA reduction framework, we are using the minVmSelection scheduling algorithm to schedule cloudlets to VM that allocate cloudlets to minimum loaded VM. This scheduling approach also helps in minimizing VM migration.

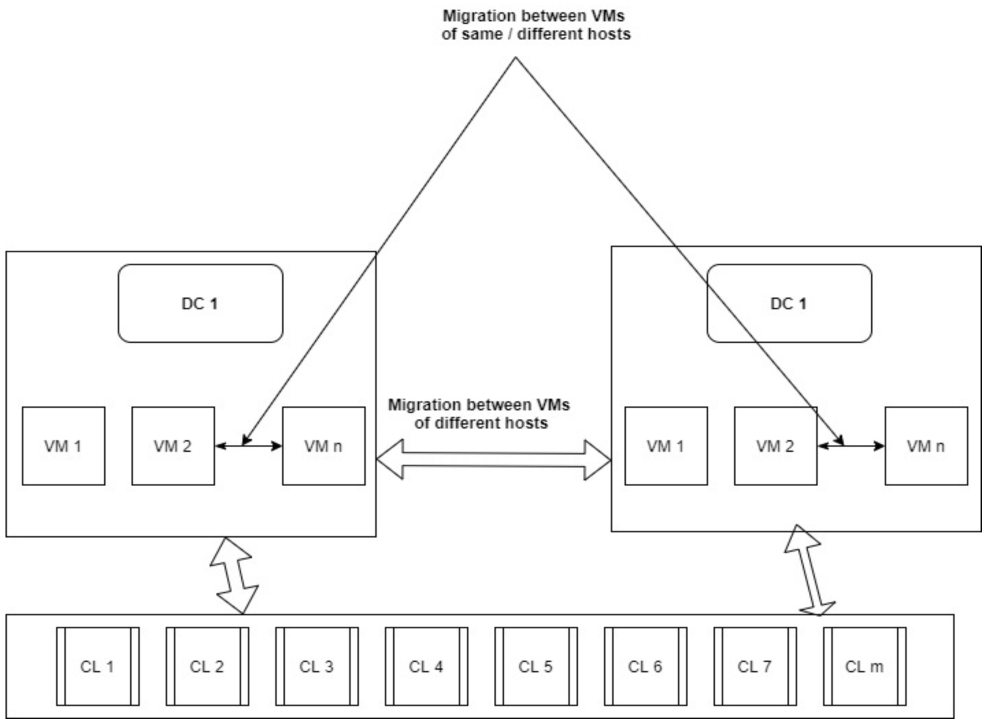

Migration between VMs of same / different hosts
DC 1
VM 1
VM 2
VM n
Migration between VMs of different hosts
DC 1
VM 1
VM 2
VM n
CL 1
CL 2
CL 3
CL 4
CL 5
CL 6
CL 7
CL m

Figure 4. SLA reduction framework using Cloudsim

The above proposed framework, represented in Figure 4 indicates the execution of our algorithm for the migration of VMs from the same and different hosts or data servers. The communication line between DC1 and DC2 specifies the migration of VM1, VM2, …, VMn of different hosts, whilst CL1, CL2, ..., CLn represents the cloudlet processes to be executed in CloudSim during the migration process of VMs.

The proposed algorithm tries to avoid/reduce SLA violations through optimal cloudlet scheduling and power optimization by reducing the number of VM migration. The proposed SLA reduction framework consists of three parts:

1.Scheduling algorithm to efficiently allocate cloudlets to the virtual machines based on the processing time of host.

Pseudo code of scheduling algorithm is presented below.

```
Input:
List of host
List of VM
List of cloudlets
Algorithm
// return the host with minimum Processing time
// Processing time of host = $\frac{Length\ of\ all\ task\ executing\ on\ VM}{Number\ of\ PES\ X\ \ MIPS\ of\ PES}$     (2)
i ← -1
 mineProcessingTime← Integer.MAX_VALUE
For each host(i) in hostList
 If host(i) available then
 If (ProcessingTimeHost(i) < minProcessingTime) then
minProcessingTime = ProcessingTimeHost(i)
i ← host.getId()
 End if
End if
 End for
// returns the VM of minimumProcessingTime host with minimum allocation
 j ← -1
 mincount← Integer.MAX_VALUE
 For each VM(j) in getVmList(minProcessingTimeHost)
 If VM available then
currCount = current allocation count of VM
 If (currCount < mincount) then
 mincount = currCount
 j ← VM.getId()
 End if
 End if
 End for
Allocate cloudlet to VM
 Mark VM as Busy
 update task length in host
update VmAllocationCount
```

The proposed scheduling algorithm selects a host with minimum processing time. The processing time of the host can be computed with Equation 1. After selecting host with minimum processing time, the algorithm extracts VMs from a given host. From the extracted VMs, the algorithm selects a VM with the minimum number of request allocation and if the VM is ideal then it allocates cloudlet to the VM. The algorithm updates status of the VM to busy and also increments vmAllocation count by 1.

2. MinVm scheduling algorithm: Allocates cloudlets to virtual machines based on cloudlet allocation counts to each VM.

```
Input:
cloudletList = List of cloudlets
vmList = List of VMs
Hashmap<Integer, Integer> vmAllocation
Algorithm
selectedVm = null;
vmSize = number of VM
for all vm in vmList do:
        vmAllocation.put(vm.getId(),0);
for all cloudlet in cloudletList do:
        selectedVm = select vm with minimum vmAllocation count
allocate cloudlet to selectedVm.
Mark selectedVm a BUSY
Update vmAllocation count for given VM by 1
```

3. Credit-based VM migration algorithm: This algorithm uses the credit of the VMs to take VM migration decisions. This algorithm tries to minimize the number of VM migration and number of host shutdown which helps in minimizing/ avoiding SLA violation of the cloud system.

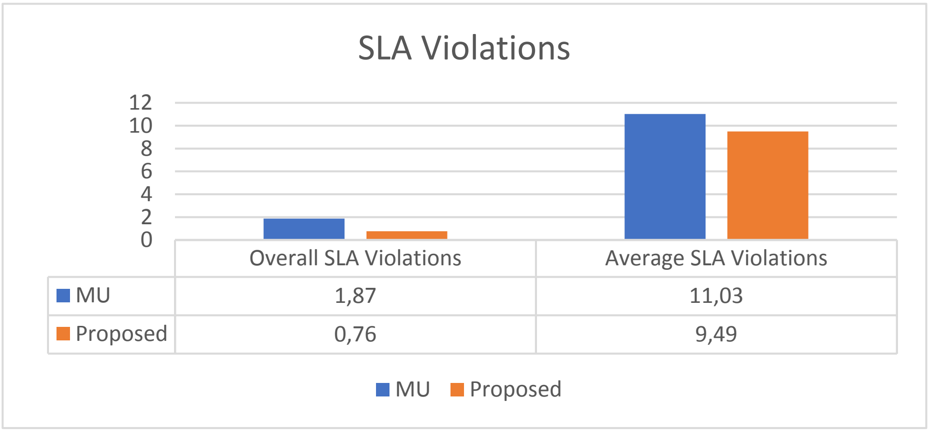
Algorithm used for Vm migration:

```
Algorithm getVmToMigrate():
    migratableVms = list of migratable VMs from host
    if no migratableVms:
        return null
    for all VMs in migratableVms:
        if VM is in migration:
                Continue
        double sizeofVm = vm.getRam()
              double vmUtilization =
    vm.getTotalUtilizationOfCpuMips(CloudSim.clock())/vm.getMips();
              double credit = vmUtilization / sizeofVm;
              vmToMigrate = select VM with minimum credit
return VmToMigrate
```

4. EXPERIMENTAL RESULTS

Figure 5 shows the difference and interaction of SLA violation of the existing algorithm and the proposed algorithm. Referring to the figure, it is clear that the proposed algorithm's overall SLA violation and average SLA violation are 0.76

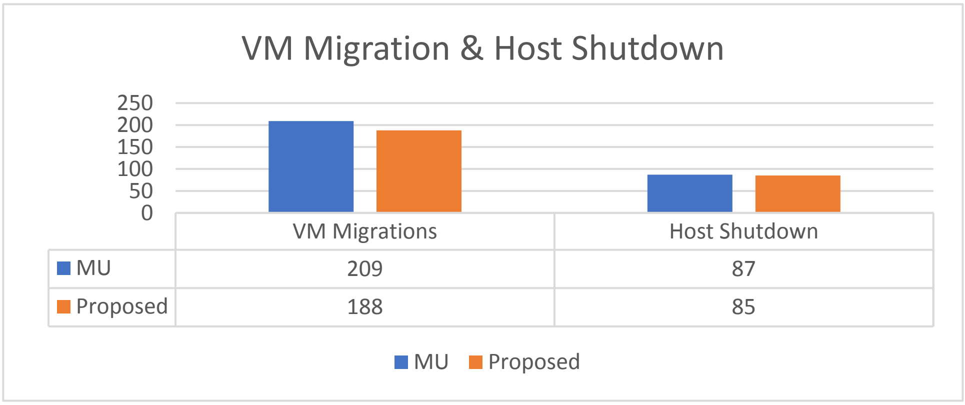
and 9.49, respectively, which is less than the existing minimum utilization algorithm whose overall SLA violation is 1.87 and average SLA violation is 11.03.



	Overall SLA Violations	Average SLA Violations
🟦 MU	1,87	11,03
🟧 Proposed	0,76	9,49

Figure 5. SLA Violations

The following graph in Figure 6 depicts the number of VM migration and host shutdown. The proposed algorithm also achieves better results (VM migration: 188 & host shutdown: 85) compared to the existing algorithm (VM migration: 209 & host shutdown: 87).



	VM Migrations	Host Shutdown
🟦 MU	209	87
🟧 Proposed	188	85

Figure 6. The number of VM (Virtual Machine) migration and host shutdown

Energy consumption, or we can also refer to energy efficiency indicates the complete infrastructure to reduce functional costs while maintaining vital QoS. Whereas energy optimization can be achieved by combining resources as per the ongoing utilization, efficient and effective virtual network roadmap and thermal situation of computing hardware and related nodes. As we can refer to the graph in Figure 7, the proposed algorithm also consumes less energy (1.67 kWh) compared to the existing (MU) algorithm (1.73), meaning that in long distance usage it could be energy consumption efficient.

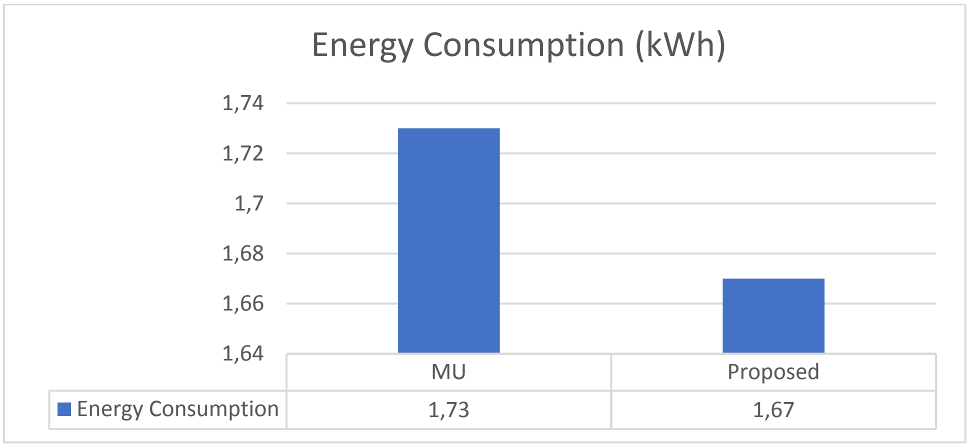


Figure 7. Energy consumption for MU and proposed algorithm

5. CONCLUSION

This paper presents an approach, based on analysis and comparison of minimum utilization policy and novel SLA policy. It indicates the SLA variations and number of Virtual Machine migration. Moreover, we conduct experiments for comparing the effectiveness of the proposed utility-based solution. Through the tailored schemes and composition of our algorithm, these results achieve to depict very less SLA violation. While most of the scheduling algorithms determine the ways to allocate a cloudlet to Virtual Machine, in our proposed case study we have considered scheduling and load balancing, meaning that we are performing scheduling and determination of which VM to migrate in specific occasions.

REFERENCES

[1]. Jung, S. M., Kim, N. U., & Chung, T. M. Applying scheduling algorithms with QoS in the cloud computing. In *2013 IEEE International Conference on Information Science and Applications (ICISA)*, June 2013, pp. 1-2.

[2]. Iordache, G. V., Pop, F., Esposito, C., & Castiglione, A. Selection-based scheduling algorithms under service level agreement constraints. In *21st International Conference on Control Systems and Computer Science (CSCS)*, IEEE, May 2017, pp. 134-140.

[3]. Emeakaroha, V. C., Brandic, I., Maurer, M., & Breskovic, I. SLA-aware application deployment and resource allocation in clouds. In *2011 IEEE 35th Annual Computer Software and Applications Conference Workshops, July 2011*, pp. 298-303.

[4]. Nita, M. C., Pop, F., & Cristea, V. Scheduling service with sla assurance for private cloud systems. In *2012 IEEE 8th International Conference on Intelligent Computer Communication and Processing*, august 2012, pp. 331-334.

[5]. Jennings B, Stadler R. Resource management in clouds: Survey and research challenges. *J Network Syst Management*, **3** (vol. 23), 2015, pp. 567–619. doi:http://dx.doi.org/10.1007/s10922-014-9307-7.

[6]. Beloglazov A, Buyya R. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. *IEEE Trans Parallel Distrib Syst*, 7 (vol. 24), 2017, pp. 1366–1379. doi:http://dx.doi.org/10.1109/TPDS.2012.240.

[7]. Shi L, Butler B, Botvich D, Jennings B. Provisioning of requests for virtual machine sets with placement constraints in Iaas clouds In: *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, Ghent, Belgium, 27-31 May 2013,pp. 499–505.

[8]. Wolke A, Tsend-Ayush B, Pfeiffer C, Bichler M. More than bin packing: Dynamic resource allocation strategies in cloud data centers. *Information Systems*, vol. 52, 2015, pp. 83–95.

[9]. Monil MAH, Qasim R, Rahman R. M. Energy-aware VM consolidation approach using combination of heuristics and migration control In: *Ninth International Conference on Digital Information Management (ICDIM)*, 2014, pp. 74–79, doi:http://dx.doi.org/10.1109/ICDIM.2014.6991413.

[10]. Aljoumah, Eman & Al-Mousawi, Fajer & Ahmad, Imtiaz & Al-Shammri, Maha & Al Jady, Zahraa. SLA in Cloud Computing Architectures: A Comprehensive Study. *International Journal of Grid and Distributed Computing*. 8, 2015, pp. 7-32 (10.14257/ijgdc.2015.8.5.02).

[11]. DOMO Report, *"Data Never Sleeps"*, [Online Accessed 05.09.2020: https://www.domo.com/solution/data-never-sleeps-6]

[12]. M. Ghobaei-Arani, A. Rahmanian, et.al. A learning-based approach for virtual machine placement in cloud data centers, I*nternational Journal of Communication Systems*, February 2018.

[13]. Mishra, Suchintan & Sahoo, Manmath. *On using CloudSim as a Cloud Simulator: The Manual*, 2017, 10.13140/RG.2.2.30215.91041.

[14]. Xiao Z, Song W, Chen Q. Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE Trans Parallel Distributed Systems*, **6** (vol. 24), 2013, pp. 1107–1117.

[15]. Chowdhury MR, Mahmud MR, Rahman RM. Implementation and performance analysis of various vm placement strategies in cloudsim. *J Cloud Computing*, **1** (vol. 4), 2015, p.1.

[16]. F. de Vaulx, E. Simmon,et.al,"Cloud Computing Service Metrics Description", *NIST Special publication 500-007*, US Department of Commerce, April 2018.

[17]. M.K. Halili, B. Cico. Towards Custom Tailored SLA in IaaS environment through negotiation model. *7th Mediterranean Conference on Embedded Computing MECO'2018*, Budva, Montenegro.

Information about the authors:

Merita Kasa Halili – PhD candidate at South East European University, Faculty of Contemporary Sciences and Technology and Teaching Assistant at State University of Tetova at the department of informatics. Merita is awarded a Google scholarship in 2012 and also the President's award "golden engineering ring".

Betim Cico –full professor in Epoka University,Albania; research interest in fields of Cloud computing, Digital System Design, Digital Image Processing, Advance Computer Architecture, Internet of Things, FPGA. He is author of many research papers, projects and superviser of several PhD students.

Manuscript received on 11 September 2020