International Journal on Emerging Technologies 11(4): 269-273(2020)

ISSN No. (Print): 0975-8364
ISSN No. (Online): 2249-3255

Economic-Driven Model for Virtual Machine Allocation in Cloud Data Center

Avneesh Vashistha[1, 2] and Pushpneel Verma[1]
[1]Department of Computer Science & Engineering, Bhagwant University, Ajmer (Rajasthan), India.
[2]Department of Information Technology, IMS Ghaziabad (Uttar Pradesh), India.

(Corresponding author: Avneesh Vashistha)
(Received 22 May 2020, Revised 27 June 2020, Accepted 07 July 2020)
(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: Elasticity is the nature of Cloud and thus a virtual machine would inevitably be operated under a dynamic workload that results in a VM encounter the problem of over-provisioning or under-provisioning. Both these cases could be good or bad: the former implies over-provisioning but the future pay-off would be more significant; while the latter refers to over/under-utilization leads to trivial pay-off or nothing. This kind of notion matches the concept of technical debt. Technical debt has been used in software engineering, software architecture, and web service composition. We are introducing technical debt in the context of virtual machine allocation and migration. VM migration decisions may come with Technical Debt- an operational liability that may incur interest in terms of cost, if not identified, managed properly and transformed from liability to future value. Developing an economically driven model for virtual machine allocation is the need of the current competitive environment. Such kind of model itself includes the identification and accompanying estimation of technical debt. we motivate the need for considering the technical debt for VM migration. In this paper, we propose a technical debt-aware approach as an economic driven model, that combines technical debt with ARMA, a time forecasting techniques, for identification and estimation of technical debt in virtual machine allocation. Through experiment results, we demonstrate that our approach can identify technical debt on virtual machines and provides accurate insights for virtual machine allocation or migration.

Keywords: Virtual Machine, SLA, Technical Debt, Forecasting, Utility.

I. INTRODUCTION

Cloud Computing is an internet-based computing platform that provides on-demand scalable computing resources such as Virtual Machine, Networks, Storage and Software, etc. The growing popularity of Infrastructure as a Service (IaaS) model in the cloud market, more and more business applications are deployed on the world-wide cloud data center. Resource management is one of the most challenging tasks in the cloud data center. Cloud computing has been considered as the fifth generation computing in which various services are offered as on-demand utility services like water, gas, electricity, etc. [1]. Elasticity and Scalability are two essential characteristics of cloud computing that supports on-demand provisioning of a shared pool of resources to meet the desired quality of service parameters like availability, reliability, pricing, throughput, etc. Computing resources offered as virtual machine instances to end-users. The virtual machine is the least tangible unit deployed over the cloud infrastructure. Scalability and elasticity characteristics are the enablers for taking economic driven decisions and leverage to drop the average running cost of VMs. These two characteristics deal with dynamic resource provisioning and releasing of resources but may not prevent VMs to be under-utilized or over-utilized that leads to sub-optimal. Taking this dynamism into account, we need to allocate cloud resources to running applications in such a way that the minimum number of VMs required for running applications. Under-provisioning VM violates service level agreements (SLAs), which often associated with a financial penalty. Under-provisioning VM creates resource scarcity between users. The resource requirements for applications are rarely static, varying, as a result, depends upon the current workload. The workload on the virtual machines changes dynamically. The increased running cost of the data center is a major economic concern. Several initiatives are found which deals with the running cost of a VM. Recently the work called "skewness" measures the unevenness in resource utilization and improves the overall utilization of server resources [2]. Similar work has been done for the dynamic mapping between VMs and PMs [3]. Moreover, a study is concerned with a two-tiered on-demand VM allocation with a feedback mechanism [4]. To better support the VM migration decision in IaaS Cloud, in this paper, we introduce an economically driven model that combines technical debt approachwith Autoregressive Moving Average (ARMA), a time forecasting techniques, for identification and estimation of technical debt in virtual machine allocation. Notably, we have made the following contributions:

— We tailor a time series forecasting model, ARMA into the economic debt-aware.
— The proposed technical debt-aware approach as an economic driven model maps the concept of technical debt into the context of VM allocation and migration in the IaaS cloud. This model also provides more insights for the decision-making process of VM allocation and migration
— Experiments on the real datasets of Materna Trace-1.

This paper is structured as follows: Section II discusses related work. Section III highlights the technical debt in the context of the VM. In section IV, we provide the technical-debt aware computing model for estimating the technical debt. Section V evaluates and discusses the experimental results of the model. Section VI and VII discuss the conclusion and future work.

II. RELATED WORK

Sotomayor *et al.,* [5] proposed an architecture that allows cost-effective on-demand short term virtual machine lease management while continuing to support the existing workload. Voorsluys *et al.,* [6] analyzed the effect and cost of virtual machine live migration in terms of performance evaluation of applications running inside Xen VMs. Beloglazov *et al*., [7] proposed an energy-efficient resource management system for cloud infrastructure. This system reduces operational costs and ensures the required QoS parameters. Lee *et al.,* [8] considered a three-tier cloud structure consisting of infrastructure vendors, service providers, consumers, and then the problem of profit-driven service request scheduling algorithms in the cloud environment has been addressed. Li *et al*., [9] proposed a novel and reliable multicast approach for cloud data center networks that minimize packet loss. Chimakurthi *et al*., [10] proposed a power-efficient resource allocation framework, based on an Ant colony that allocates resources to applications without violating SLAs. Strunk & Dargie [11] investigated major factors that lead to power consumption during VM migration. Qian and Medhi [12] solved two resource management problems first proposed an approach to minimize server operational costs and second resource allocation while considering QoE.

III. TECHNICAL DEBT AT VM LEVEL

A. Technical Debt

Technical Debt metaphor was initially coined by Ward Cunningham for managing software debt. According to Cunningham, "Shipping first-time code is like going into technical debt. A little debt speeds development so long as it is paid back promptly with a rewrite. Objects make the cost of this transaction tolerable. The danger occurs when the debt is not repaid. Every minute spent on not-quite-right counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unconsolidated implementation, object-oriented or otherwise" [13]. Technical debt makes visible when there is a trade-off between actual and optimal decisions. Technical debt can be attributed to sub-optimal decisions, shortcuts on decisions, and/or deferred activities that can incur extra cost/rework if it would be carried in the future as when compared the current time [14]. The major aspect of technical debt is that it must be serviced i.e., once a system incurs a debt then interest charges must be paid off. There are always some deadlines to follow within a given time frame. To meet these deadlines there is always a tacit pressure that encourages shortcuts. Technical debt is analogous to financial debt(Principal) that would be taken to gain short terms benefits in business but need to pay back at earliest for avoiding incurred interest on debt concerning time [15]. In recent years, Technical debt as a metaphor has been applied in the field of software engineering such as software architecture, design, testing, coding, and documentation, etc. Gradually this term is being investigated in the context of VM management also for different perspectives [16-18]. Whenever a VM is sub-optimal, it carries a Technical Debt. Shortcuts contribute to technical debt and compromise quality.

B. Technical Debt in VM Allocation

In the context of VM, the Technical Debt may be used to investigate workload prediction for VM. The workload is usually measured in terms of the number of requests being assigned to or executed by a virtual machine for a given time interval. Undoubtedly the cloud is dynamic that reflects changes in the workload, i.e. a VM may be under-utilized or over-utilized. Based on the predicted workload, it may be calculated in advance how many VMs shall be required in the future to process the workload.

Technical Debt may be categorized as intentional or unintentional [19]. Every VM carries a TD in either of the forms whether it is intentional or unintentional. Intentional debt may be introduced intentionally or strategically for optimizing the current state of the existing virtual machine and not much bother for future consequences. In most of the cases, TD may be linked to ill and/or poorly-justified decisions that may produce short term gains but fails in future value generation and results in unintentional debt. Unintentional debt maybe finds out at the test, architecture, and configuration levels.

C. Technical Debt Indicators in VM Environment

SLA Violation: The foundation of SLA is the trust in the service provider and its purpose is to ensure that the performance and availability of the resources the service provider guarantees to deliver to the customer. SLA violation formed unintentional debt on VM when current response time does not match with pre-defined response time mentioned in SLA. In this case, penalty cost against each service request violation would be count as an interest in technical debt.

— Run Time Decision: An ill or poorly justified run-time decision for VM allocation may lead to technical debt in such a way that the allocated VM may not support scalability requirements in the future according to the dynamic workload.

— VM Utility: When a VM is sub-optimal it carries a technical debt from a utility point of view. For example, a VM may be sub-optimal in both the cases of over-provisioned or under-provisioned.

D. Resources Prediction Method for VM Environment

To predict the values of CPU and memory usage on each timestamp(five minutes interval), we use Autoregressive Moving Average (ARMA), temporal forecasting. We prepared a time series dataset containing CPU and Memory usage at each time interval (e.g., five minutes) and feed this dataset as an input to ARMA(p,q) model for predicted the CPU and Memory usage for each time stamps.Formally the general expression of ARMA is ARMA (p,q) [20], where p is the order of autoregressive polynomial and q is the order of the moving average polynomial. The value of p and q have been identified based on the ACF and PACF functions. ARMA model can be expressed with the following equation -1

$$x_t = \emptyset_1 x_{t-1} + \emptyset_2 x_{t-2} + \ldots$$
$$+ \emptyset_p x_{t-p} + w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} \;..+ \theta_q x_{t-q} \quad (1)$$

where$w_t$is white noise.

In this work, to estimate the values CPU and memory parameters, the ARMA model realized in python.

IV. VM DEBT COMPUTING MODEL

Technical debt has some visible consequences. The impact of technical debt in terms of cost can be seen as composed of principal and interest.

Principal: In the context of VM, the principal is the invested cost of making a sub-optimal VM to optimal that results in VM maximum utilization. Specifically, the principal amount of debt can be calculated using the formula:

$$\text{Principal} \;= (VM_{\text{reworkcost}} * T)$$

Where T indicates the time(in minutes) required for searching and allocating the new VM and $VM_{\text{reworkcost}}$ represents the required VM execution cost for processing [14, 21]. The time for VM reallocation can be easily calculated by averaging the time for previous rounds of VM reallocation.

Suppose that searching and reallocating a VM requires 5 minutes(denoted as T) and the $VM_{reworkcost}$ is the cost for making a sub-optimal VM to optimal is \$0.0025 then it takes a principal as 5* 0.0025=\$0.0125. The time for VM reallocation can be easily calculated by averaging the time for previous rounds of VM reallocation.

Interest: Interest is the cost of fixing a defect when discovered. Interest is the continuing cost and every minute spent on defects counts as interest on that debt which could be calculated with the following formula.

$$\text{Interest(R)} = \begin{cases} \sum_{r=1}^{m}\sum_{t=1}^{n}(R_{\text{provisioning}} - R_{\text{utility}}) * \text{Execution Cost} & \text{if } R_{\text{provisioning}} > R_{\text{utility}} \\ \sum_{t=1}^{n}(SLA_{\text{availability}} - VM_{\text{resource utility}}) * C_p & \text{otherwise} \end{cases}$$

Further, Interest can be calculated in two different cases when VM is either over-provisioned or under-provisioned.

VM Over-Provisioning: In the case of the VM resource over-provisioning (eg., if Rprovisioning>Rutility) i.e, the resource provisioning is greater than the current resource utilization then the interest would be accumulated as the acquired cost of the unused resources of VM and cloud be calculated with the following formula:

$$\text{Interest} = \sum_{r=1}^{m}\sum_{t=1}^{n}(R_{\text{provisioning}} - R_{\text{utility}}) * \text{Execution Cost} \quad \text{if } R_{\text{provisioning}} > R_{\text{utility}}$$

Here R represents two resources: CPU and Memory, $R_{provisioning}$ represents CPU and Memory provisioning, $R_{utility}$ is the current usage of CPU and Memory, Execution cost is the running cost of CPU and Memory for 5 minutes time interval.

To conceptualize the case of VM over-provisioning, let us consider the processing capacity of a VM is 5000 requests/second. Suppose at time interval t_1 the numbers of requests decrease, let's say 4000 requests/second that makes a VM over-provisioned. In this case, revenue generated by the underlying VM is decreased and it carries a technical debt for the service provider.

VM Under-Provisioning: Again, if the VM is under-provisioned (eg., if $R_{provisioning}<R_{utility}$), i.e the resource provisioning is less than or equal to the current resource utilization that results in SLA violation and thus extra cost would be paid off in terms of penalty which is being calculated with the following formula:

$$\text{Interest} = \sum_{t=1}^{n}(SLA_{\text{availability}} - VM_{\text{resource utility}}) * C_p \quad \text{Otherwise}$$

For example, in the case of VM under-provisioning, let us consider again the current workload is 6000 requests/second at time interval t_2. Since VM is processing more requests than its processing capacity, the VM is considered as under-provisioned. At this point, a VM does not meet QoS parameters like response time or throughput as mentioned in SLAs, therefore a penalty cost would be pay off against each request violation. In the long run, this collected penalty cost may be negative or positive debt that depends on time-dependent VM under-provisioning.

In both the cases whether over/under provisioning, we have has two options: First, VM live migration but it results in an increased operational cost that includes the cost for searching new VM within a data center, VM booting time, power consumption, availability, and reliability; Second, wait for more users to join and then that compensates this technical debt. At this point, incurred debt may be good or bad.

— **GOOD DEBT:** A debt in VM allocation is considered as good debt if took intentionally as a strategic decision, for a short period, as having expectations to compensate this debt in the future and generate revenue as well.

— **BAD DEBT:** Bad debt leads to a situation in which a system is continuously under-utilized and the would not be able to pay back accumulated debt in the future.

Finally, for the future timestamps(t), the overall technical debt, for a VM allocation decision can be estimated as the sum of principal and interest

$$\text{Debt} = \text{Principal} + \text{Interest}$$

V. EVALUATION

In this paper, we considered CPU and RAM usage as key parameters concerning the running cost of a virtual machine. The proposed approach finds out the actual resource usage and compared the Technical debt-aware approach with the two existing algorithms: IQRMMT and IQRMC. Technical debt-aware approach outperforms these two existing algorithms in the context of resource utilization (CPU utilization and memory utilization). Moreover, the Technical debt-aware approach shows less incurred debt when compared with IQRMMT and IQRMC.To evaluate Technical debt-aware approach, we design experiments to assess the performance of our approach employing comparing it with the state of the art approaches. In particular, we aim to answer the following research questions (RQs):

RQ1. Can Technical debt-aware approach achieve better resource utilization than the state of the art approaches?

RQ2. Whether the Technical debt-aware approach identifying and estimating the technical debt for VM allocation in the IaaS cloud?

A. Experimental Setup

We conducted our experiments on the CloudSim simulator. We created one data center and five hosts within a data center. Each host contains one virtual machine which has the configuration according to the pricing scheme for n1-machine types for Mumbai (Asia-South) parameters. The pricing would be on-demand pricing for Vcpu and Memory [22]. We realized our experiment on the machine having a configuration withIntel Core i5-6200U 2.8 GHz. Processor, 8 GB DDR4-2133 SDRAM, and Windows 10. In this experiment, the current usage of CPU and memory is collected from Materna-trace-1 consists of 520 VMs. The trace was taken in the distributed Materna Data Centers in Dortmund over a timespan of one month. This trace was taken on a VMware ESX environment using 49 Hosts, 69 CPU cores, and 6780 GB RAM[23]. To evaluate the prediction quality, we pre-process the CPU and memory current usage by using 10 days data for training the forecasting model, while the next 7 days data is used for testing the accuracy.

B. Result & Discussion

To answer all the RQS, we examine the performance of Technical debt-aware approach against the following two approaches:

IQRMMT: IQRMMT is an adaptive threshold algorithm that selects the candidate VM with the minimum migration time relative to other VMs.

IQRMC: IQRMC is an adaptive threshold algorithm that selects the candidate VM with the maximum correlation with the other VMs.

(i) Results discussion for RQ1: Throughout the entire 10080 minutes run of the experiment shows that the total resource utilization of the technical debt-aware approach outperforms the state of the art approaches. The results shreds of evidence that the technical debt-aware model can provide better resource utilization and provide more benefit than simply having a predicted model for VM resource utilization.

The technical debt-aware approach achieves better and more stable results on resource utilization, as shown in the following figures: Fig. 1-3.

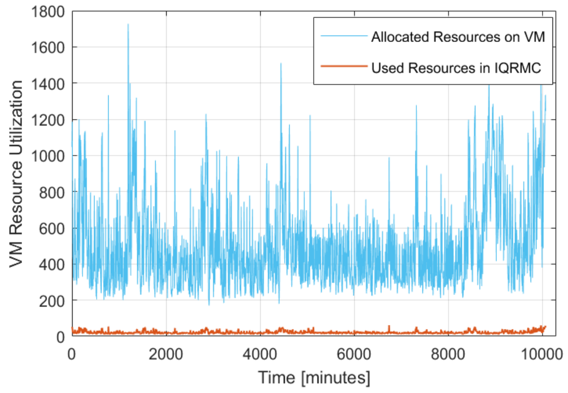

Fig. 1. VM Resource utilization by the IQRMC approach over 10080 Minutes.

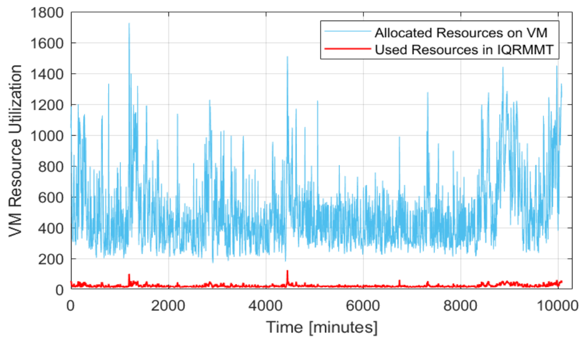

Fig. 2. VM Resource utilization by the IQRMMT approach over 10080 Minutes.

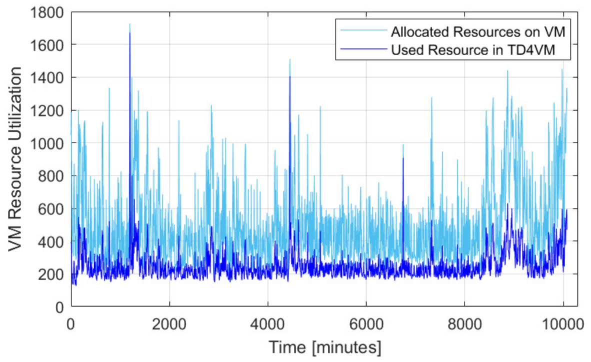

Fig. 3. VM Resource utilization by the Technical debt-aware approach(TD4VM) over 10080 Minutes.

(ii) Results & discussion for RQ2: The experiment ran for 10080 minutes and results shown in Fig. 4, 5 concludes that besides resource utilization the technical debt-aware approach outperformed IQRMC and IQRMMT approaches. The technical debt-aware approach effectively identified and managed the incurred technical debt on running VM and took VM migration decision whenever required.

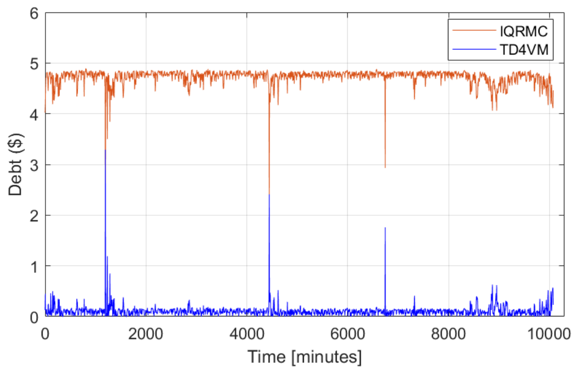

Fig. 4. Total debt accumulated by the Technical debt-aware approach(TD4VM) and IQRMC over 10080 Minutes.

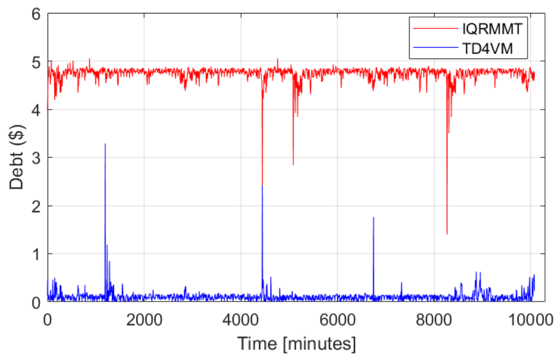

Fig. 5. Total debt accumulated by the Technical debt-aware approach(TD4VM) and IQRMMT over 10080 Minutes.

VI. CONCLUSION

This paper leverages the notion of technical debt-aware approach for the virtual machine allocation and migration. Specifically, we discussed key technical debt indicators for VM allocation. Based on the incurred technical debt, our model can identify, manage, and decide when to migrate a VM to overcome the problem of VM over-provisioning and under-provisioning.

VII. FUTURE WORK

In Future work, we seek to extend the Technical debt-aware approach to explore how time series forecasting method can support for transforming the incurred debt into more specific future values.

REFERNECES

[1]. Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, *25*(6), 599-616.

[2]. Xiao, Z., Song, W., & Chen, Q. (2012). Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE transactions on parallel and distributed systems*, *24*(6), 1107-1117.

[3]. Bobroff, N., Kochut, A., & Beaty, K. (2007). Dynamic placement of virtual machines for managing sla

violations. In *2007 10th IFIP/IEEE International Symposium on Integrated Network Management,* 119-128.
[4]. Song, Y., Sun, Y., & Shi, W. (2011). A two-tiered on-demand resource allocation mechanism for VM-based data centers. *IEEE Transactions on Services Computing*, *6*(1), 116-129.
[5]. Sotomayor, B., Keahey, K., Foster, I., & Freeman, T. (2007). Enabling cost-effective resource leases with virtual machines. In *Hot Topics session in ACM/IEEE International Symposium on High Performance Distributed Computing*, 1-3.
[6]. Voorsluys, W., Broberg, J., Venugopal, S., & Buyya, R. (2009). Cost of virtual machine live migration in clouds: A performance evaluation. In *IEEE International Conference on Cloud Computing* (pp. 254-265). Springer, Berlin, Heidelberg.
[7]. Beloglazov, A., & Buyya, R. (2010). Energy efficient resource management in virtualized cloud data centers. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 826-831.
[8]. Lee, Y. C., Wang, C., Zomaya, A. Y., & Zhou, B. B. (2010). Profit-driven service request scheduling in clouds. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing* (pp. 15-24). IEEE.
[9]. M. X. M. Z. C. G. Y. Z. M. W. D. Li, "RDCM: Reliable Data Center Multicast," in IEEE INFOCOM.
[10]. Chimakurthi, L. (2011). Power efficient resource allocation for clouds using ant colony framework. *arXiv preprint arXiv:1102.2608*.
[11]. Strunk, A., & Dargie, W. (2013). Does live migration of virtual machines cost energy?. In *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, 514-521.
[12]. Qian, H., & Medhi, D. (2013). Data center resource management with temporal dynamic workload. In *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013),* 948-954.
[13]. Cunningham, W. (1992). The WyCash portfolio management system. *ACM SIGPLAN OOPS Messenger*, *4*(2), 29-30.
[14]. Kumar, S., Bahsoon, R., Chen, T., & Buyya, R. (2019). Identifying and estimating technical debt for service composition in SaaS cloud. In *2019 IEEE International Conference on Web Services (ICWS)*, 121-125.
[15]. Ampatzoglou, A., Ampatzoglou, A., Chatzigeorgiou, A., & Avgeriou, P. (2015). The financial aspect of managing technical debt: A systematic literature review. *Information and Software Technology*, *64*, 52-73.
[16]. Vashsitha, P. V. A. (2018). Economic-Driven Strategies for Virtual Machine Allocation in Cloud Data Center. *IJET*, 10-15.
[17]. Kumar, S., Chen, T., Bahsoon, R., & Buyya, R. (2020). DATESSO: Self-Adapting Service Composition with Debt-Aware Two Levels Constraint Reasoning. *arXiv preprint arXiv:2003.14377*.
[18]. Kumar, S., Bahsoon, R., Chen, T., Li, K., & Buyya, R. (2018). Multi-tenant cloud service composition using evolutionary optimization. In *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, 972-979.
[19]. McConnell, S. (2008). Managing technical debt. *Construx Software Builders, Inc.*, 1-14.
[20]. https://nwfsc-timeseries.github.io/atsa-labs/sec-tslab-autoregressive-moving-average-arma-models.html.
[21]. Curtis, B., Sappidi, J., & Szynkarski, A. (2012). Estimating the size, cost, and types of technical debt. In *2012 Third International Workshop on Managing Technical Debt (MTD)*, 49-53.
[22]. "https://cloud.google.com/compute/vm-instance-pricing," GCP, 2020.
[23]. "http://gwa.ewi.tudelft.nl/datasets/gwa-t-13-materna".

How to cite this article: Vashistha, A. and Verma, P. (2020). Economic-Driven Model for Virtual Machine Allocation in Cloud Data Center. *International Journal on Emerging Technologies*, *11*(4): 269–273.