# Cloud computing for mobile applications: outsourcing the deployment of virtual machines

**Tauseef Ahmad[a], S. P. Singh[a], Prashant Johri[b], Zafrul Hasan[c]**
[a] Department of Computer Science & Engineering, NIMS university, Jaipur, Rajasthan.
[b] School of Computing Science & Engineering, Galgotia University
[C] College of Nursing, King Saud University, Riyadh, Saudi Arabia
**∗ Corresponding author**
Tauseef Ahmad (ORCID ID- 0000-0001-6769-1617)
Department of Computer Science & Engineering.
NIMS University, Jaipur, Rajasthan.
NH-8, Jaipur Delhi Highway, Jaipur, Rajasthan – 303121
tauseefala@gmail.com

**Abstract:** Application offloading is a software-level technique for enhancing the compute capabilities of smart mobile devices in mobile cloud computing. The use of additional computational resources in the deployment and administration of VM on Smartphone is a problematic component of such frameworks. Virtual Machine (VM) deployment necessitates the use of computational resources for VM construction and configuration. The administration of virtual machines (VMs) involves computer resource usage, VM lifecycle monitoring, and physical resource management for VMs on smartphones. The goal of this project is to ensure that virtual machine deployment and administration do not necessitate additional processing resources on mobile devices for application offloading. VM deployment, application execution times, and the simulation's total execution time are used to evaluate the success of VM deployment and management in application processing. VM deployment and management, according to the research, need additional resources on the computer host. As a result, deploying virtual machines on smart mobile devices has become a heavyweight technique for offloading processes.

**Keywords:**Virtual Machine, Cloud Computing, Mobile Cloud Computing, Network Virtualization, Public Cloud, Private Cloud.

## 1. Introduction

Cloud computing, and mobile cloud computing are some of the advanced computing paradigms that have arisen as a result of recent advancements in computing and communication technologies. Cloud computing has evolved as an intriguing subject of research from both academic and industrial viewpoints, as well as a valuable commercial asset and the best way to access information globally.Cloud computing system has made the concept of computing as a "utility" a reality(Buyya, Yeo, Venugopal, Broberg, & Brandj 2009) by delivering on the hope of computing as a fifth public utility, alongside water, telephone, electricity, and gas.Cloud computing provides centralized computing services on demand by allowing users to access computer resources through centralized servers(Armbrust et al., 2009; Mollah, Azad, &

Vasilakos, 2017). Mobile Cloud Computing (MCC) is an important part of the cloud revolution because it expands the computing capabilities of devices with limited resources(Sánchez Ribes, Mora, Sobecki, & Mora Gimeno, 2020). Researchers are experimenting with various methods to improve the computing capabilities of smart mobile devices(Salkenov & Bagchi, 2019). Application offloading is a popular method for extending the computational capabilities of Smart Mobile Devices (SMDs) by sharing compute workload(Guo, Liu, Yang, Xiao, & Li, 2019). We present a thematic taxonomy for several sorts of application offloading frameworks in this paper. A migration-based application based on virtual machines (VMs). Cloud-based application processing is known as offloading methodology that encapsulates an application in a virtual machine instance and migrates it to a cloud server node(Almutairi & Aldossary, 2021).The additional costs related with deployment, utility and management of VMs on SMD, which demands additional processing resources on SMD, is a problematic feature of VM migration-based offloading mechanisms.VM management involves overhead, deployment of applications in virtual machines, and VM migration, which entails additional hardware resources and is costly to execute(Osanaiye et al., 2017; Sivagami & Easwarakumar, 2019). As a result, computing resources are used in excess over a longer length of time. This study examines several aspects of the virtual machine (VM) lifecycle and deployment for application processing, such as VM formation, application allocation to VM, and VM destruction. In a variety of experiments, we look at how virtual machines affect the execution time of individual programs as well as the total simulation time. VM deployment for application processing needs a high amount of computing power because the computing device and the VM are sharing the same physical resources. From a variety of angles, in this paper we analyze the overhead associated with the deployment of virtual machines in application processing. To collect data, researchers used CloudSim, a simulation toolkit that covers both system and behavior modeling of cloud Infrastructure as a Service, such as datacenters, virtual machines (VMs), applications (cloudlets), and services.

## 2. Background

The principles of cloud computing and mobile cloud computing, as well as the notion of cloud virtualization process, are covered in this section.

### 2.1 Cloud computing

Distributed computing is a model that facilitates the on-demand provision of computer resources and services, as well as pay-per-use business models(Ahmad, Haque, Al-Nafjan, & Ansari, 2013; Buyya et al., 2009).Client devices' capacity and capabilities are increased by computational clouds, rather than owning infrastructure and software applications(Al-Marsy, Chaudhary, & Rodger, 2021), Providing access to software applications and leased infrastructure. It has allowed for new types of communication and collaboration, as well as new information service access techniques. On-demand computing services and resources are provided by computational clouds in a number of service models(Tseng, Tseng, Yang, Liu, & Chou, 2018). Service providers provide different service models such as Software as a Service (SaaS), Infrastructure as a Service (IaaS), and Platform as a Service (PaaS)(Olokunde, Misra, & Adewumi, 2017). Figure 1 depicts a tiered cloud computing architecture at a basic level.


Computational clouds provide physical resources in the virtual machines(Kapil, Tyagi, Kumar, & Tamta, 2017; Yadav, Garg, & Ritika, 2019). The hypervisor is in charge of virtual machine deployment and management, as well as providing access to physical resources(Qin, Wu, Chen, Xue, & Wei, 2019; Salah, Abdulgahni, & Ahmad, 2014). Application hosting platforms consist of cloud development environments and monitoring tools, such as quality of service (QoS) negotiation, admission control, and pricing. Cloud apps run in total isolation on virtual machine instances. Public utility computing is exemplified by Amazon Web Services (AWS)(S3, 2011), Google AppEngine, Microsoft Azure, and Aneka. AWS offers infrastructure as a service (IaaS) and software as a service (SaaS) to enable cloud datacenters to use virtualized resources and services(Aljamal, El-Mousa, & Jubair, 2018). AWS's Simple Storage Service (S3) is used to store personal data, while its elastic cloud compute is used to use application processing capabilities in the cloud computing the Google AppEngine is a computational cloud that provides an application development and deployment platform in Google's data centers using the Platform as a Service architecture(K, Laxmaiah, & Sharma, 2019; T. Ahmad, A.A. Ansari, A. Akhtar, & Parveen, 2014). App Engine is a platform that allows you to create apps using popular programming languages like Java and Python.By assisting with the design, development, and deployment of applications and services in cloud datacenters, Microsoft Windows Azure supports the PaaS service model of computational cloud. Aneka provides a PaaS platform and an application development platform for creating bespoke apps and delivering them on public clouds computing or private clouds computing.

**2.2 Mobile cloud computing**

Mobile cloud computing system is a divided up computing approach,mobile devices with smart capabilities can now access a wide range of cloud datacenter services(Alshehri, Alshahrani, Alzahrani, Alharthi, & Aloufi, 2018). For resource-constrained smart mobile devices, many computational cloud service models are used in this computing paradigm. MCC's goal is to use cloud datacenter storage and processing capacity to boost the computing capabilities of smart mobile devices(Alsenaidy & Tauseef, 2012; Armbrust et al., 2009). A smartphone, an internet connection, and a computational cloud are the three components of the MCC architecture. A smartphone is a compact, mobile computer device that combines the computing capabilities of PDAs with the voice calling capabilities of traditional cellular phones.Application offloading,

which involves generating a dispersed program execution environment during runtime, is the most prevalent solution in MCC for reducing resource restrictions on smart mobile devices(Paranjothi, Khan, & Nijim, 2017). The concept of application offloading stems from the idea of outsourcing computationally intensive applications to datacenters with abundant resources, either partially or altogether. Offloading approaches now in use are primarily concerned with what to offload and where to offload, and do not take into account the overhead of dispersed program execution on smart mobile devices(Alsenaidy& Tauseef, 2012; Chun, Ihm, Maniatis, Naik, & Patti, 2011; Satyanarayanan, Bahl, Caceres, & Davies, 2009).An application running on VM migration-based distributed execution is kept (partially or entirely) in a VM instance residing on SMD, which is then migrated to a cloud server node distant from the application.

**2.3 Cloud virtualization**

Virtualization of the cloud enables the separation of devices and resources, such as servers, storage devices, networks, and operating systems, into one or more execution contexts using a framework to create a virtual instance of it(Borangiu, Trentesaux, Thomas, Leitão, & Barata, 2019; Morabito, Cozzolino, Ding, Beijar, & Ott, 2018).Virtualization totally refers to the process of deploying and managing virtual instances. Software or firmware responsible for building a virtual machine on the host hardware is known as a hypervisor. Hypervisor hides the hosting system's intricacies from the guest execution environment.Virtualization in cloud computing relies heavily on the provision of consolidated VM instances to meet the needs of computing users(Helali & Omri, 2021; Kominos, 2017). A physical host is shared by multiple virtual machines (VMs), resulting in dynamic multiplexing of computation resources and high scalability.

## 3. Methodology

Experiments in a simulation environment were conducted using CloudSim, a simulation toolset for cloud system components including datacenters, virtual machines (VMs), and resource provisioning schemes that allows both system and behavior modeling.It employs generic application provisioning techniques that are easy to extend and require minimal effort. It allows for the modeling and simulation of cloud computing architectures that include both single and interconnected clouds(Calheiros, Ranjan, Beloglazov, Rose, & Buyya, 2011). Virtual clouds in CloudSim consist of three primary components: hosts, datacenters, and broker services. CloudSim's resourceproviders are datacenters.We are simulating two datacenters, each of which has the following attributes: VMM, X86 Architecture, Linux Operating System (VM Ware). The Datacenter is made up of 20 hosts, each with 12 GB RAM, 100 GB of storage, and a bandwidth of 10000 MB. Every host is made up of two Processing Elements (PEs), each capable of processing 1000 million instructions per second. PEs inside the host are scheduled using the timeshared scheduling strategy. In each datacenter, a datacenter broker facilitates the agreement between SaaS- and cloud-based service providers, which are based on Quality of Service (QoS) standards. The cloud datacenter broker is an intermediary between SaaS providers and datacenters. It searches the Cloud Information Service (CIS) for eligible Cloud service providers

and conducts online negotiations for resource/service allocation that meets the application's QoS requirements. This class has been extended to assess and test virtual machine and cloudlet performance metrics. In our experiments, we create virtual machines with nearly identical settings.Virtual machines consist of a 10000 MB image, 1GB RAM (VM memory), 250 MIPS processing power, 1000 MB bandwidth, and a CPU/PE.A list of virtual machines is compiled and every virtual machine is assigned to a particular datacenter broker. VMs are allocated to datacenters by default of the simulator's VM Allocation policy (VmAllocationPolicySimple), while the host VMs are scheduled by the timeshared scheduling policy. One or more cloudlets are assigned to each VM. As a cloudlet, you can access cloud-based applications like social networks, content delivery, business workflows, and more. CloudSim classifies an application's complexity according to its computational requirements. Throughout its existence, every application service has its own a pre-determined instruction duration and data transmission overhead. A cloudlet is assigned to a certain VM within a datacenter's host. A virtual machine (VM) can run one or more Cloudlets at the same time. It's also feasible to move a cloudlet between various VMs while it's running. The Cloudlet object stores all of the Cloudlet's metadata as well as the VM's ID. Several cloudlets are launched with some requirements in our experiment.In cloudlets, users are identified by attributes such as their id so that resources can return to them after execution, etc. cloudlet File Size 300 bytes, cloudlet Length 40000 millions of instructions (MI). The output file size 300 bytes of cloudlet and required 1 CPU for processing. The list data structure is updated with all cloudlets. A datacenter broker is assigned to a cloudlet list. For the allocation of cloudlets to virtual machines, the default scheduling policy of the data center broker is used. All of the VMs on a host share the host's computing power. At runtime, VMs are established on the host, and a timeshared scheduling policy is used to schedule PEs within a host. In the CloudSim simulation environment, the datacenter broker receives both the VM and Cloudlet lists. The following methodology is used to experiment and assess VM deployments for application processing.

**3.1 Experimental setting**

There are three basic experimental categories in the taxonomy of experimentation. (1) We look at the costs of deploying virtual machines. As an example, we examine the time taken to build VMs, allocate applications to VMs, and destroy VMs. Tests are conducted to determine an exact average time for creating virtual machines (VMs) with varied resource allocations to VM time, and time for destroying virtual machines (VMs). An average value is used to calculate the overhead in VM deployment for application processing. (2) We assess the overhead involved with running an application in a virtual machine.(2) We assess the overhead involved with running an application in a virtual machine (VMs). (3) The influence of virtual machines on simulation execution time is assessed by comparing total simulation time for the non-shared virtual machines to total simulation time for the shared virtual machines.

## 4. Results

### 4.1Virtual machine creation with time association.

To help the SMD encapsulate mobile application instances for offloading, a virtual machine instance on the device will require processing power. The total time it goes for create a virtual machine in a simulation environment differs from the time it takes to create a virtual machine in a real-world environment because virtual machines in CloudSim start instantly. As a result, there is no wait for obtaining the pictures or launching the VMs: the such kind of process all hosts have access to all of the images file. We compare the average VM creation time across various experiments with changing numbers of VMs. Figure 2 showing that increase average time of VM as the total number of VMs increases. The average VM creation time for 3–36 VM instances in the simulated environment ranges from 10 to 120 milliseconds. It demonstrates that the formation of a VM instance necessitates the use of more resources on the host computer. As a result, VM generation is a resource-intensive and time-consuming process. As a result, the overhead of VM formation on SMD is included in VM migration-based mobile application offloading.

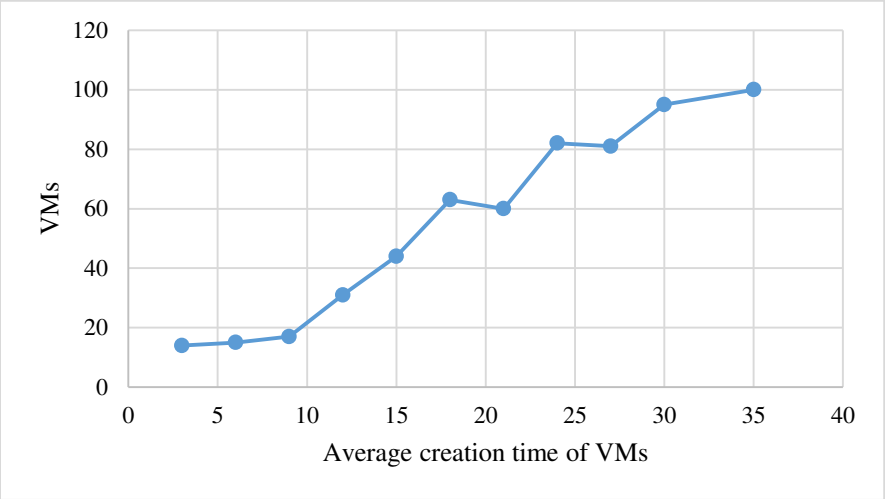



Figure-2 virtual machine creation time result.

### 4.2Analysis of application allocation to VM time

A mobile application offloading technique based on VM migration encapsulates operational instances in VM instances on the distribution monitoring device. The offloaded mobile application is continued by copying its VM instance to a cloud server node, which continues the offloaded mobile application using compute resources. In several studies, we analyze application allocation to VM time. In CloudSim, the datacenter broker assigns apps to VMs based on the scheduling rules. We test the overhead associated with application to VM allocation and monitor the average time took for application to VM allocation while keeping the number of VMs and apps the same. Figure 3 depicts the increasing trend in the average time necessary to allocate applications to VM. The time it takes an application to be allocated to a VM ranges from 10 to 325 milliseconds when distributing 3 to 36 applications among separate virtual machines. We allocate two applications to virtual machines, but rather than allocate two applications to each

VM, we allocate 20 applications to each VM. Allocating 36 different applications with the same computing load to separate VMs consumes 325 milliseconds of computing resources.As the number of apps and VMs increases, the time it takes to assign apps to VMs increases. The higher VM allocation time total value illustrates the application VM allocation overhead.

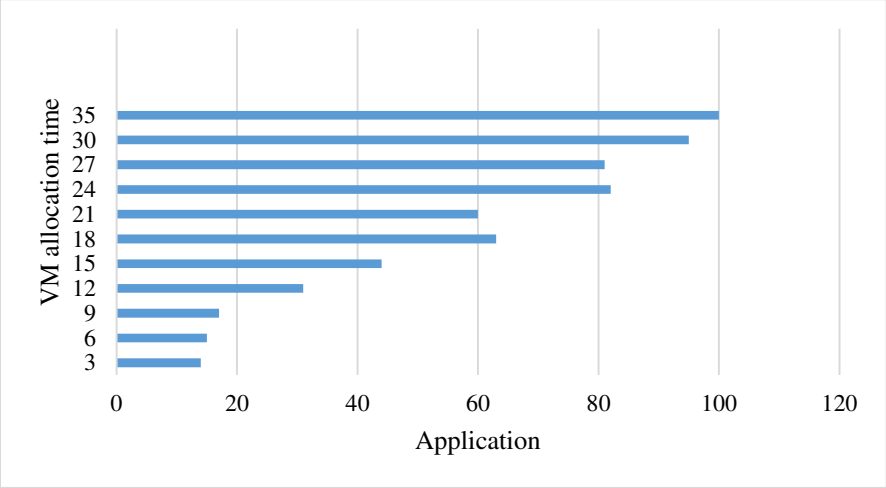
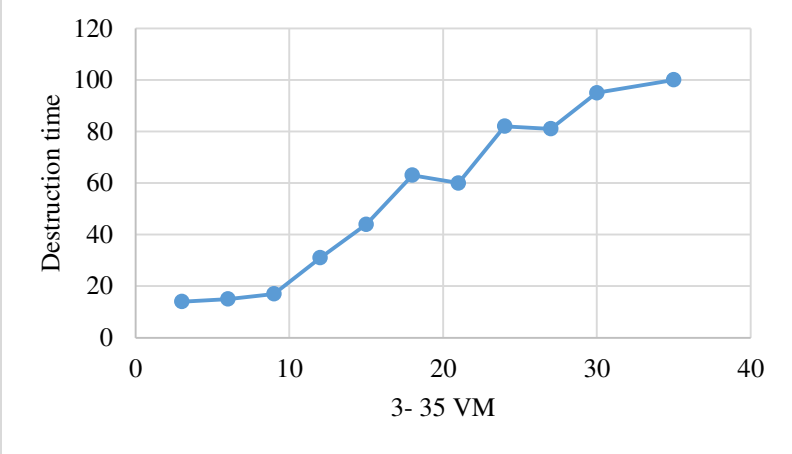



Figure 3- VM application allocation time analysis.

**4.3Analysis of the time it takes for VMs destruction**

Finally, the VM lifecycle ends with the VM being destroyed, which involves returning its physical resources to its host. According to the data, the deletion of VM results in a minimal amount of overhead.For up to 35 virtual machines, the average destruction time is calculated. In a simulation scenario, varying numbers of virtual machines result in the same amount of time required for VM destruction to occur across trials. Figure 4 shows the increase in VM destruction from above 3 to 100. As one aspect of CloudSim, a cloudlet is an application service like social networking, content distribution, and business processing. The following properties are used to model a mobile application (cloudlet). Every application has a User ID, which allows cloud resources to return to it after the execution is completed. 40000 million of instruction (MI) Cloudlet Length, total cloudlet file 300 bytes and output size 300 bytes also. the requested CPU for this process only 1. All of the applications in the experiment are run on the local host, as a result, application migration in datacenters has no overhead at runtime. We run tests with various numbers of applications to determine the average time spent on each application's execution.Two separate instances of the application's execution time in VM are examined. (a) In one scenario, each application is given its own virtual machine instance. In this case, the application assigned to a VM instance consumes all of the resources of the VM instance. (b) Another scenario involves reducing the number of virtual machines, but maintaining the number of applications at twice as many as virtual machines. The result is that two applications are sharing the same VM.




**4.4Total execution time for application for VMs that are not shared**

VM scheduling is not an issue in this scenario, as each application has its own virtual machine instance.A mobile application that runs in two distinct virtual machines takes 380294 ns on average to execute. During the application execution of 36 separate VMs, the total average execution time was 2938711 ns. From Table 1, it can be seen that increasing the number of apps from two to five increases the average execution time for a single application by 28% and from five to ten increases the average execution time by 54%.Similarly, even if each program runs in its own virtual machine, as the number of applications and virtual machines increases, the average execution time of a single program increases.The execution time of 2–45 applications increases by 57 percent on average.This indicates the burden of VM deployment and management for application processing as demonstrated in the increase in execution time value.

Table 1- Total application execution time for not shared process with the % increase

| User | % Increase |
|---|---|
| 1 to 3 | 21.00 |
| 3 to 6 | 39.00 |
| 6 to 9 | 36.00 |
| 9 to 12 | 40.00 |
| 12 to 15 | 43.00 |
| 15 to 18 | 45.00 |
| 18 to 21 | 41.00 |
| 21 to 24 | 49.00 |
| 24 to 27 | 54.00 |
| 27 to 30 | 54.00 |
| 30 to 33 | 60.00 |
| 33 to 36 | 62.00 |

### 4.5Total execution time for application for VMs that are shared

In the current case, as compared to the number of applications, the number of virtual machines (VMs) is cut in half. As a result, numerous apps share a single virtual machine. One obvious argument is that limiting the number of VMs reduces the complexity of VM deployment and management, therefore improving the application's average execution time. However, an examination of the experimental findings reveals that there is a cost associated with allocating the VM's CPU across different apps. As a result, the application's average execution time rises in lockstep.As a result of the shared VM scenario, the average application time increased significantly in table 2. According to the results, the execution time for a single application increases by 28% when the number of applications is increased from 1–3 and run on 1–2 virtual machines. Similarly, even if each VM is shared by numerous apps, the average time for execution a single program grows as the number of applications and VMs increases. For 1–36 apps running on 1–22 virtual machines, the application execution time increases by 55.17 percent on average.

Table 2- Total application execution time for shared process with the % increase

| User | % Increase value |
|---|---|
| 1 to 3 | 28.00 |
| 3 to 6 | 62.00 |
| 6 to 9 | 42.00 |
| 9 to 12 | 46.00 |
| 12 to 15 | 52.00 |
| 15 to 18 | 56.00 |
| 18 to 21 | 64.00 |
| 21 to 24 | 57.00 |
| 24 to 27 | 60.00 |
| 27 to 30 | 64.00 |
| 30 to 33 | 65.00 |
| 33 to 36 | 66.00 |

### 4.6Analysis of the simulation time

Using the simulation time parameter, we can gauge the length of time it took to simulate a cloud computing application running in virtual machines.When virtualization is used, a number of parameters pertain to VM deployment, management, and application execution that influence the total simulation time. In a series of tests, we look at the effects of VM deployment and administration on total simulation execution time. (a) For each application, we use a separate virtual machine to calculate the total execution time. (b) We estimate that halving the number of VMs reduces the entire simulation execution time by half the number of applications.Comparing simulation times for VMs shared by many different apps shows that overall simulation time initially increases for VMs shared by several different apps. However, when the number of applications increases, the total simulation time for VMs shared by several apps lowers when

compared to VMs that are not shared. As shown in figure 5, shared and non-shared virtual machines take different amounts of time to complete simulations. When the number of virtual machines is reduced and VMs share by several applications, the average execution time of the application increases. This is because a VM that is shared by multiple applications has to schedule the processing power (CPU) for these applications. In the current result, there is additional overhead associated with VM CPU scheduling for numerous applications. Moreover, despite the added expense of VM scheduling for numerous applications, for shared VMs, the total simulation execution time decreases.By using virtual machines in a shared environment, management and deployment overhead can be reduced. The result is a reduction in the overhead associated with virtual machines, thereby reducing the duration of simulations as a whole. Two major findings emerge from the analysis of the results. (1) As opposed to building a distinct VM for each application, sharing a VM is more useful for application (cloudlet) execution. (2) The deployment of each VM adds overhead to application processing, increasing the demand for computing resources and the application's execution time. The application is run on a virtual machine, and the application's execution time increases as a result of the VM deployment and management. This means that the computational resources are being utilized for longer periods of time rather than having to deploy VMs every time a program is executed.

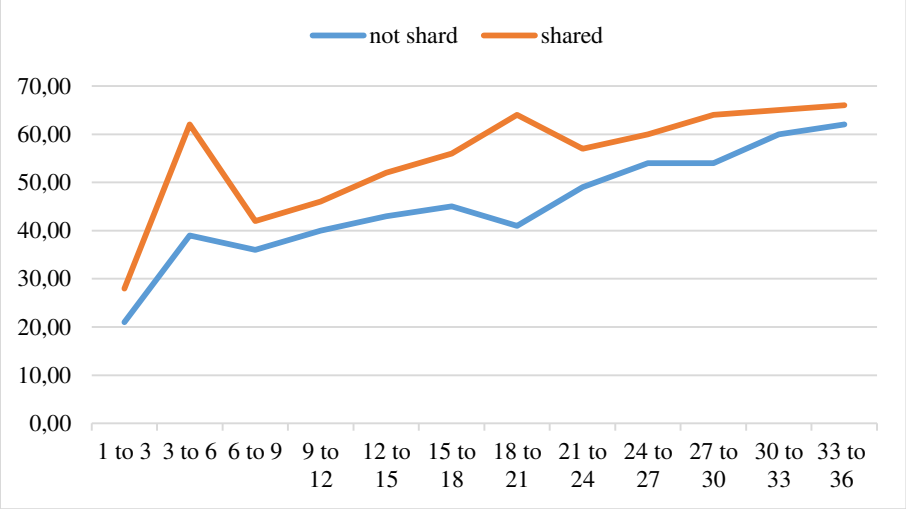



Figure 5- the overall simulation time for shared and non-shared virtual machines

## 5. Conclusion

Application offloading is a way for extending the computation capabilities of SMDs at the application layer. However, VM deployment-based offloading techniques require additional resources to install and operate virtual machines on SMDs, which uses more computing resources and slows the application's execution time. We discovered that the overhead of deploying and managing virtual machines for application processing is substantial. As a result, VM-based process offloading is a heavyweight way for deploying distributed programs in MCC. Because of the small size and inherent limits of SMDs, a lightweight distributed application framework is required to make the most of their computing resources in distant application processing. Future studies will examine new lightweight distributed models and methods for

effective resource consumption on Smartphones for distributed applications in order to achieve the above MCC goals. Instead of relying on runtime platform division for offload processing, we will utilize distributed systems deployment principles in order to explicitly distribute responsibilities among computing entities at design time. The creation of such lightweight procedures will result in significant performance benefits and an overall improvement in the deployment and processing of distributed applications in mobile cloud computing.

**Declaration of Interest Statement:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**References (APA).**

Ahmad, T., Haque, M. A., Al-Nafjan, K., & Ansari, A. A. (2013). Development of Cloud Computing and Security Issues. *Information and Knowledge Management, 1*(3), 34-43.

Al-Marsy, A., Chaudhary, P., & Rodger, J. A. (2021). A Model for Examining Challenges and Opportunities in Use of Cloud Computing for Health Information Systems. *4*(1), 15.

Aljamal, R., El-Mousa, A., & Jubair, F. (2018, 3-5 April 2018). *A comparative review of high-performance computing major cloud service providers.* Paper presented at the 2018 9th International Conference on Information and Communication Systems (ICICS).

Almutairi, J., & Aldossary, M. (2021). Modeling and Analyzing Offloading Strategies of IoT Applications over Edge Computing and Joint Clouds. *13*(3), 402.

Alsenaidy, A., & Tauseef, A. (2012). A review of current state M Government in Saudi Arabia. *GLOBAL ENGINEERS & TECHNOLOGISTS REVIEW, 2*(2), 5-8.

Alshehri, A., Alshahrani, H., Alzahrani, A., Alharthi, R., & Aloufi, E. (2018, 12-14 Dec. 2018). *The Potential of Utilizing Mobile Cloud Computing in Mobile Devices.* Paper presented at the 2018 International Conference on Computational Science and Computational Intelligence (CSCI).

Armbrust, M., Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., . . . Zaharia, M. J. S. (2009). Above the Clouds: A Berkeley View of Cloud Computing. *53*, 07-013.

Borangiu, T., Trentesaux, D., Thomas, A., Leitão, P., & Barata, J. (2019). Digital transformation of manufacturing through cloud services and resource virtualization. *Computers in Industry, 108*, 150-162. doi:https://doi.org/10.1016/j.compind.2019.01.006

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandj , I. J. F. G. C. S. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *25*, 599-616.

Calheiros, R. N., Ranjan, R., Beloglazov, A., Rose, C. A. F. D., & Buyya, R. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *41*(1 %J Softw. Pract. Exper.), 23–50. doi:10.1002/spe.995

Chun, B.-G., Ihm, S., Maniatis, P., Naik, M., & Patti, A. (2011). *CloneCloud: elastic execution between mobile device and cloud*. Paper presented at the Proceedings of the sixth conference on Computer systems, Salzburg, Austria. https://doi.org/10.1145/1966445.1966473

Guo, S., Liu, J., Yang, Y., Xiao, B., & Li, Z. (2019). Energy-Efficient Dynamic Computation Offloading and Cooperative Task Scheduling in Mobile Cloud Computing. *IEEE Transactions on Mobile Computing, 18*(2), 319-333. doi:10.1109/TMC.2018.2831230

Helali, L., & Omri, M. N. (2021). A survey of data center consolidation in cloud computing systems. *Computer Science Review, 39*, 100366. doi:https://doi.org/10.1016/j.cosrev.2021.100366

K, M., Laxmaiah, D. M., & Sharma, D. Y. K. (2019). A COMPARATIVE STUDY ON GOOGLE APP ENGINE AMAZON WEB SERVICES AND MICROSOFT WINDOWS AZURE. *International Journal of Computer Engineering & Technology (IJCET), 10*(1), 54-60.

Kapil, D., Tyagi, P., Kumar, S., & Tamta, V. P. (2017, 15-17 Aug. 2017). *Cloud Computing: Overview and Research Issues.* Paper presented at the 2017 International Conference on Green Informatics (ICGI).

Kominos, C. G. (2017). *Performance analysis of different virtualization architectures using OpenStack.* (Independent thesis Advanced level (degree of Master (Two Years)) Student thesis), Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-318099 DiVA database. (17001)

Mollah, M. B., Azad, M. A. K., & Vasilakos, A. (2017). Security and privacy challenges in mobile cloud computing: Survey and way ahead. *Journal of Network and Computer Applications, 84*, 38-54. doi:https://doi.org/10.1016/j.jnca.2017.02.001

Morabito, R., Cozzolino, V., Ding, A. Y., Beijar, N., & Ott, J. (2018). Consolidate IoT Edge Computing with Lightweight Virtualization. *IEEE Network, 32*(1), 102-111. doi:10.1109/MNET.2018.1700175

Olokunde, T., Misra, S., & Adewumi, A. (2017). *Quality Model for Evaluating Platform as a Service in Cloud Computing*, Cham.

Osanaiye, O., Chen, S., Yan, Z., Lu, R., Choo, K. R., & Dlodlo, M. (2017). From Cloud to Fog Computing: A Review and a Conceptual Live VM Migration Framework. *IEEE Access, 5*, 8284-8300. doi:10.1109/ACCESS.2017.2692960

Paranjothi, A., Khan, M. S., & Nijim, M. (2017). Survey on Three Components of Mobile Cloud Computing: Offloading, Distribution and Privacy %J Journal of Computer and Communications. *Vol.05No.06*, 31. doi:10.4236/jcc.2017.56001

Qin, J., Wu, Y., Chen, Y., Xue, K., & Wei, D. S. L. (2019). Online User Distribution-Aware Virtual Machine Re-Deployment and Live Migration in SDN-Based Data Centers. *IEEE Access, 7*, 11152-11164. doi:10.1109/ACCESS.2019.2891115

S3, A. (2011). http://status.aws.amazon.com/s3-20080720.html. *Accessed on 20th July 2011*.

Salah, M., Abdulgahni, H. M., & Ahmad, T. (2014). Current growth of Information and communication technology in Saudi Arabia. *wulfenia journal, 21*(9), 216-223.

Salkenov, A., & Bagchi, S. (2019). Cloud based autonomous monitoring and administration of heterogeneous distributed systems using mobile agents. *Future Generation Computer Systems, 99*, 527-557. doi:https://doi.org/10.1016/j.future.2019.04.047

Sánchez Ribes, V., Mora, H., Sobecki, A., & Mora Gimeno, F. J. (2020). Mobile Cloud computing architecture for massively parallelizable geometric computation. *Computers in Industry, 123*, 103336. doi:https://doi.org/10.1016/j.compind.2020.103336

Satyanarayanan, M., Bahl, P., Caceres, R., & Davies, N. (2009). The Case for VM-Based Cloudlets in Mobile Computing. *8*(4 %J IEEE Pervasive Computing), 14–23. doi:10.1109/mprv.2009.82

Sivagami, V. M., & Easwarakumar, K. S. (2019). An Improved Dynamic Fault Tolerant Management Algorithm during VM migration in Cloud Data Center. *Future Generation Computer Systems, 98*, 35-43. doi:https://doi.org/10.1016/j.future.2018.11.002

T. Ahmad, A.A. Ansari, A. Akhtar, & Parveen, S. (2014). Current review of ICT and m-government services in Saudi Arabia. *International Journal of Computer Engineering and Applications, 7*(2), 71-77.

Tseng, C.-W., Tseng, F.-H., Yang, Y.-T., Liu, C.-C., & Chou, L.-D. (2018). Task Scheduling for Edge Computing with Agile VNFs On-Demand Service Model toward 5G and Beyond. *Wireless Communications and Mobile Computing, 2018*, 7802797. doi:10.1155/2018/7802797

Yadav, A. K., Garg, M. L., & Ritika. (2019). *Docker Containers Versus Virtual Machine-Based Virtualization*, Singapore.