International Journal of Cognitive Informatics and Natural Intelligence
Volume 15 • Issue 3 • July-September 2021

A Multi-Objective Genetic Algorithm-Based Resource Scheduling in Mobile Cloud Computing

Somula Ramasubbareddy, Department of Information Technology, VNRVJIET, Hyderabad, India
Evakattu Swetha, SV College of Engineering, Tirupati, India
Ashish Kumar Luhach, The PNG University of Technology, Papua New Guinea
T. Aditya Sai Srinivas, G. Pullaiah College of Engineering and Technology, Kurnool, India

ABSTRACT

Mobile cloud computing is an emerging technology in recent years. This technology reduces battery consumption and execution time by executing mobile applications in remote cloud server. The virtual machine (VM) load balancing among cloudlets in MCC improves the performance of application in terms of response time. Genetic algorithm (GA) is popular for providing optimal solution for load balancing problems. GA can perform well in both homogeneous and heterogeneous environments. In this paper, the authors consider multi-objective genetic algorithm for load balancing in MCC (MOGALMCC) environment. In MOGALMCC, they consider distance, bandwidth, memory, and cloudlet server load to find optimal cloudlet before scheduling VM in another cloudlet. The framework MOGALMCC aims to improve response time as well as minimizes VM failure rate. The experiment result shows that proposed model performed well by reducing execution time and task waiting time at server.

KEYWORDS

Cloudlet, Genetic Algorithm, Mobile Cloud Computing, VM Load Balancing

INTRODUCTION

Cloud Computing is popular resource platform where user can offload their applications for executing and get result back in order to overcome the limitations of mobile device. Mobile Cloud Computing is emerged by a combination of two popular technologies such as Cloud computing and Network communications. MCC has been gained a lot of attention from researchers. The most advantage of MCC is that it reduces the complexity of the application and improves mobile device performance in terms of power. Mobile Devices (MD) is becoming more powerful for running complex applications such as resource intensive applications. The limitations of mobile devices in terms of battery, CPU speed, and limited memory are making developers unable to run the complex applications (Devare et al., 2010).

In order to improve the performance of the Mobile device, MCC has introduced a Novel concept called Offloading, which can offload resource intensive application into the Cloud. There are various Cloud service providers such as icloud and EC2. Mobile users can use resources of elastic Cloud in order to optimize performance of mobile applications. In (Kumar & Lu, 2010), the author

DOI: 10.4018/IJCINI.20210701.oa5

This article, published as an Open Access article on April 23rd, 2021 in the gold Open Access journal, the International Journal of Cognitive Informatics and Natural Intelligence (converted to gold Open Access January 1st, 2021), is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

has focused on energy utilization of mobile device, by offloading tasks into Cloud environment to improve the performance of the mobile device. In (Z. Li et al., 2001)(Rong & Pedram, 2003), the author offloading computation task to remote Cloud to reduce energy utilization. The networking cost between mobile devices and remote servers was addressed in (Gu et al., 2003) . The application is portioned and offloaded to the nearby remote server for processing(Krishna et al., 2016). The response time between mobile device and Remote Cloud is significant challenge. However, the algorithm has not addressed response time in both wireless and remote environment(Raju & Saritha, 2016). The MCC is used in various areas such as image processing, Speech recognition, Translator etc.(Dinh et al., 2013)(Gkatzikis & Koutsopoulos, 2014)(Y. Wang et al., 2015) (Rahimi et al., 2014) (Sheikhalishahi et al., 2011).

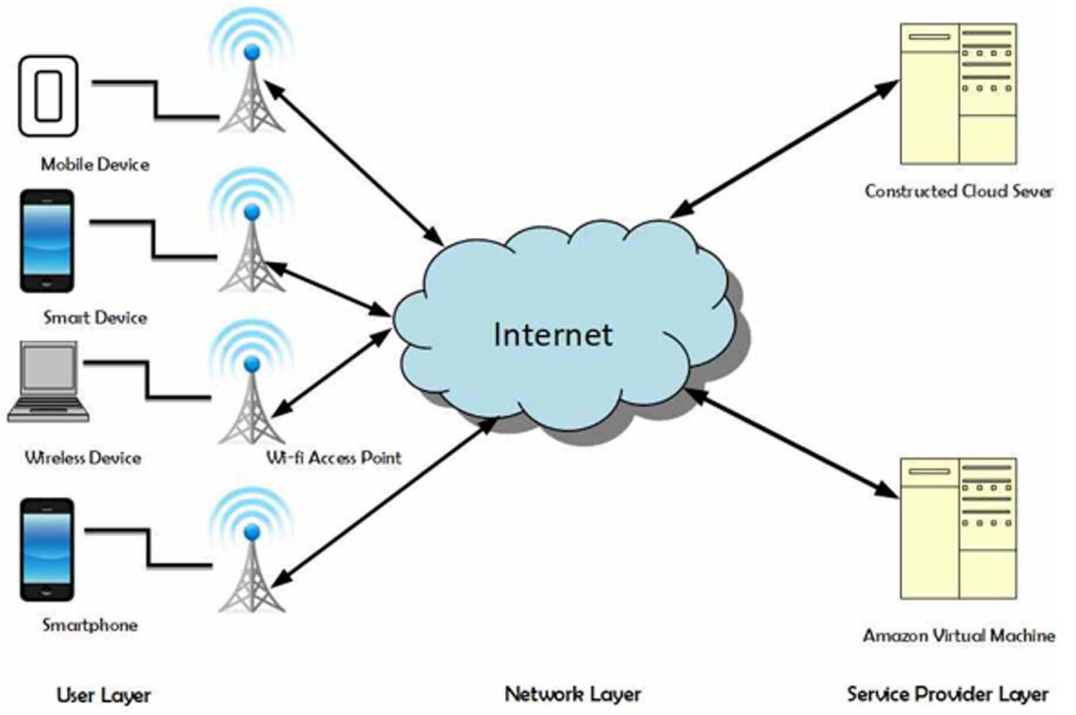
The mobile device can run high end applications which require huge computation power and Storage. The requirement of resources for each application may vary. In result, Resource allocation to mobile devices should be dynamic. In MCC, mobile devices can offload intense application to the remote Cloud for faster execution(Chun et al., 2011)(Cuervo et al., 2010)(Gkatzikis & Koutsopoulos, 2013)(R. Somula et al., 2019). The distance between mobile device and Cloud remote server increases the response time. In result, the overall execution time of intensive application also increases the new concept called Cloudlet has been introduced to address latency related challenge (Satyanarayanan et al., 2009). Cloudlet is small scale data center which is available around the user. By using nearby Cloudlet for execution, the user can decrease the overall response time of application. Cloudlet is a three-tier architecture in Fig.1, which is introduced between mobile devices and remote Cloud. The virtual technology available in Cloudlet to share hardware resources with incoming requests from users. The resources of Cloudlet (available bandwidth, CPU, Memory, etc.) are shared with VMs. In Cloud computing resource processing is a popular area (Bobroff et al., 2007)(Das et al., 2013) (Das et al., 2014)(Jiwei Li et al., 2013)(Van et al., 2009). Resource scheduling among Cloudlets is a significant issue(Gkatzikis & Koutsopoulos, 2014). The mobile user always moves from one location to another location. Therefore, the distance also increases between mobile device and Cloudlet; it causes a delay in execution time. In order to address this issue, the task is moved to nearest Cloudlet by measuring user distance. The distance between mobile device and Cloudlet is not the only reason for task migration to another Cloudlet. The load of the target Cloudlet also one important factor for task migration, when the server handles a greater number of tasks than actual capacity then execution time of task increases gradually. Some works have focused on VM scheduling in MCC (Gkatzikis & Koutsopoulos, 2014)(L. Wang et al., 2014)(Islam et al., 2016)(Liu et al., 2015)(Taleb & Ksentini, 2013)(R. Somula & Sasikala, 2019)(Ramasubbareddy & Sasikala, 2019). The previous works focused on static task execution. In this proposed method, we consider bandwidth, load and distance as constraints to select Cloudlet for scheduling among Cloudlets. In this paper, the proposed model addresses the scheduling among Cloudlets. Therefore, resource constrained devices show better performance. The novelty in this paper is considering three objective functions to schedule VM in another Cloudlet (Distance, Bandwidth, Memory and Load of the Cloudlet).

The rest of the paper as follows: section 2 describes about the Cloudlet and previous load balancing algorithms. Section 3 explains MCC environment. Section 4 frames the problem formulation for resource scheduling algorithm. Section 5 describes simulation results with existing work. Finally, section 6 concludes the topic and section 7 explain the idea of feature work.

BACKGROUND

There are many scheduling algorithms to address relatively routine problems in MCC. The resource scheduling algorithm has a great impact on the performance of the application in terms of execution time and service cost. Therefore, the optimization algorithm minimizes execution time of the task. In result, the solution will be more effective(Dinh et al., 2013)(Gkatzikis & Koutsopoulos, 2013). In (S. Wang et al., 2014), the author proposed load balancing algorithm with objective of maximum resource

Figure 1. Cloudlet Architecture



utilization and maximization of CPU capacity. But the algorithm fails to save energy consumption for resources. In (Song et al., 2010), the author proposed Fuzzy-GA optimization approach for task scheduling process by considering scheduling decision. But it failed to minimize overall execution time of the jobs. In (Jian-feng Li et al., 2011), the author describes ant colony optimization for scheduling tasks by considering make span and mean task completion time objectives. But it failed to address queue length of each Virtual Machine (VM). JuhnkeEtal (Juhnke et al., 2011), proposed a multi-objective algorithm with two main objectives such as execution time and cost by using Pareto model. Increase in the data transferring time is the drawback of this model. In (Guo et al., 2012), proposed multi-objective resource scheduling in Cloud computing(Sheikhalishahi et al., 2011). This algorithm mainly deals with cost and execution time which neglects resource utilization aspect.

There are many works which have been already studied in resource scheduling. But they are not up to the mark, some works are implemented by considering energy consumption as a primary objective using optimization algorithm. Shieh Etal (Shieh & Pong, 2013), for scheduling task in multi core system by focusing on energy consumption. This algorithm did not concern about resource utilization in multi-cores. In (Shieh & Pong, 2013), the author proposed energy aware task scheduling in the Cloud it calculates energy consumption for Cloud and then changes data based on network conditions. This algorithm used bi-level model to schedule tasks in the Cloud. But, this bi-level model fails to address resource utilization issue. In MCC, many methodologies used to find when a user should offload task to Cloudlet (Cuervo et al., 2010)(Gkatzikis & Koutsopoulos, 2013)(Das et al., 2014)(Jiwei Li et al., 2013)(Van et al., 2009)(Islam et al., 2016)(R. Somula & Sasikala, 2018)(R. S. Somula & Sasikala, 2018)(R. Somula & Sasikala, 2019b)(Devare et al., 2010). In(Taleb & Ksentini, 2013), the author proposed service migration based on user mobility in MCC.

In the proposed model, the Cloudlet manager makes a decision on Cloudlets based on distance and execution time of the task. When VM is scheduling to another Cloudlet server the size of the data

to be transferred is also considered in the proposed model. Most of the scheduling approaches do not consider the distance, bandwidth and load of the server in mcc which increases the response time of the applications. When we schedule task or VM among Cloudlets we must consider load, distance and bandwidth in order to handover task to another VM in another Cloudlet. If we don't consider bandwidth, the transferring time of the task increases. In this approach we consider GA methodology for balancing the load among Cloudlets and predicts optimization Cloudlet.

MCC ENVIRONMENT

When mobile user offload task to Cloud environment the task first will be received by Cloudlet Manager (CM). The CM maintains the status of all available Cloudlets. Based on available information, the CM makes a decision to which Cloudlet the user task is to transfer by considering distance and load of the Cloudlet. Each Cloudlet is a collection of multiple VM's. Each VM in Cloudlet is resource provisioned (Bandwidth, Storage and CPU). The user task is allocated to VM's in Cloudlet based on execution time, transferring time and load of the server. If the user moves from one Cloudlet to another Cloudlet in a Cloud environment, then the corresponding VM is scheduled in another Cloudlet. In MCC environment, a set of AP's establishes network communication among Cloudlets. The set of available Cloudlets are denoted as C= $\{C_1, C_2 \ldots C_N\}$. Each of these Cloudlets communicates through CM. the bandwidths between Cloudlet a, b are represented as $B_{a,b}$. E ach Cloudlet has fixed memory Ma and processing power P_a.

Task Completion Time

The task completion time depends on available resources to VM in Cloudlet. Let us denote mathematically, $A_{n,k}^{a}$ illustrates minimum task completion time of VM_k for each n^{th} individual, when scheduling VM from one Cloudlet 'a' to another Cloudlet 'b'. If VM is not scheduled from Cloudlet 'a', it is same as 'b'. Then following equation (1) calculates minimum task completion time.

$$f_1 = \min_{\forall_{b \in A_{k,a}}} \left\{ \frac{A_{n,k}^{a,b}}{max_{\forall_{b \in A_{k,a}, t \in I_p A_{n,k}^{a,b}}}} \right\} \tag{1}$$

VM Transfer Time

The time required for scheduling VM from Cloudlet 'a' to Cloudlet 'b' is denoted as $T_{n,k}^{a,b}$. When Cloudlet a, b is equal to zero, that is $T_k^{a,b} = 0$. The following formula (2) is used to predict VM transferring time.

$$f_2 = \min_{\forall_{b \in A_{k,a}}} \left\{ \frac{T_{n,k}^{a,b}}{max_{\forall_{b \in A_{k,a}, t \in I_p T_{n,k}^{a,b}}}} \right\} \tag{2}$$

VM Current Load

When the user moves from one Cloudlet to another Cloudlet, the execution time increases, so that the Cloudlet schedule VM in another Cloudlet. The following equation (3) is used to predict a load of the Cloudlet.

$$f_3 = \underset{\forall_{b\in A_{k,a}}}{min}\left\{\frac{L_{n,k}^{a,b}}{max_{\forall_{b\in A_{k,a},t\in I_p L_{n,k}^{a,b}}}}\right\} \tag{3}$$

Available Bandwidth

The bandwidth of each VM in network differs from other VM. The task transfer to another VM when a task takes more time for processing. The following equation (4) is used to predict bandwidth of the VM.

$$f_4 = \underset{\forall_{b\in A_{k,a}}}{max}\left\{BW_{n,l}^{a}\left(VM_a, VM_b\right)^{-1}\right\} \tag{4}$$

Available Memory

The task length of Cloudlet is considered as one of the objectives to optimize scheduling in MCC. When task arriving rate increases, then the length of task queue increases. In order to achieve optimize scheduling, then the length of task queue has to be reduced. The length of task queue VM_K can be controlled by QL (VM_K) by calculating remaining memory R (VM_K^{mem}) and no. of available CPUs R (VM_K^{CPU}.) the following equation (8) is used to predict available memory of the VM.

$$QL\left(VM_k\right) = R\left(VM_K^{mem}\right) * R\left(VM_K^{CPU}\right) \tag{5}$$

Where

$$R\left(VM_K^{mem}\right) = O\left(VM_K^{mem}\right) - A\left(T_j^{mem}\right) \tag{6}$$

$$R\left(VM_K^{CPU}\right) = O\left(VM_K^{CPU}\right) - A\left(T_j^{CPU}\right) \tag{7}$$

O (VM_K^{mem}) represents allocated memory for (VM_K) during initialization, $A(T_j^{mem})$ represents assigned memory for task from $\left(VM_K\right)$, O (VM_K^{CPU}) represents the allocated no. of CPUs during VM initialization, $A(T_j^{CPU})$ represents assigned CPUs to task T_j from $\left(VM_K\right)$.

$$f_5 = \sum_{k=1}^{n} QL\left(VM_k\right) \tag{8}$$

Problem Formulation

In this section, the problem statement describes about multi-object VM scheduling among Cloudlets. These objective functions focus on execution time, transfer time, load, available bandwidth and memory of the server and available bandwidth among VM's.

Problem:

$$F_1 = \left\{f_1, f_2, f_3, f_5\right\}$$
$$F_2 = \left\{f_4\right\}$$

Solution:

$$\forall_{a,b} = 1,2,3,4,\ldots,x$$
$$\forall_{k} = 1,2,3,4,\ldots,n$$

MULTI-OBJECTIVE GENETIC ALGORITHM BASED LOAD BALANCING IN MCC (MOGALMCC)

In this model, when the user moves towards another Cloudlet the VM's scheduled to another Cloudlet which is load free and near to mobile user. There are many existing load balancing algorithms but they don't provide an optimal solution in polynomial time. Existing algorithms scheduled VM's based on load and size of the task. In our model, we used GA for scheduling VM's among Cloudlets by taking load, distance and bandwidth as constraints for selecting Cloudlet. GA is known for solving NP-hard problems. In order to achieve good performance, we must modify operators to get better performance. The GA performs following operations: population generation, selection, crossover, mutation. In the first step, GA selects N number of populations randomly. Then remaining operations follow one by one to generate child population. Then the first child population and parent generations are merged to generate the second generation.

Chromosome Representation

In genetic algorithm, the population can be represented by set of chromosomes $C_1 = \left\{C_1, C_2, \ldots, C_N\right\}$ of length N. each of these chromosomes contains set of bits. Each bit of chromosome considered as one VM in our model and value of the VM can be used to find target Cloudlet. The number of tasks process by Cloudlet is equal to number of VM's available in Cloudlet.

Initialization of Population

Population initialization is a major step in the genetic algorithm. If the population is good then it leads to good selection in search space. Otherwise bad optimization selection. In our model, we first assume population as I_p which changes for every generation. We consider set of access points as population A_k for each VM in Cloudlet 'a'. Each Cloudlet calculates the probability of accessibility $P_{k,b}^{A}$ for each VM 'k' in A Cloudlet 'b'. The following equation (9) $P_{k,b}^{a}$ evaluates probability of accessibility for each VM:

$$P_{k,b}^{A} = \frac{A_{k,b}^{a}}{max_{\forall_{b \in A_k^{a,A_{k,b}^{a}}}}} \tag{9}$$

Fitness Function

In this section, the cross over operation will perform between two individuals. Each individual section probability (9) the following equation (1), (2), (3), (4) and (8) evaluate n value.

$$F = \left\{F_1, F_2\right\} \tag{10}$$

```
Algorithm 1   Population Initialization
1.        The Probability of Each VM K in AP b is calculated
2.        P_B → (Previous Access Point)
3.        If (P_b avilable) then
4.        Use P_B with a probability of P_{k,b}^A
5.        If user does not belong to current then P_B does not
belong to current then P_B is not used for that user.
6.        The AP which is available in P_B will have higher
Probability.
7.        This process continues using Roulette Wheel Selection,
AP for each VM.
8.        In this selection process, The AP with Higher
Probability of having more chances to be selected.
9.        Else
10.        Generate Population using Roulette Wheel selection
strategy.
11.        End if
```

The individual who have high probability has a higher chance of getting selected.

Crossover Operation

In this section, the cross over operation will perform between two individuals. Each individual section probability p_s^n. The following equation (11) evaluates n value.

$$P_n^c = \frac{F_n}{\sum_{i=1}^{n} F_n} \tag{11}$$

Mutation Operation

Mutation operation will perform between two chromosomes to produce diverse population for next generation. The mutation is done by replacing bit positions randomly between chromosomes. The Mutation Probability (MP) must always be less; otherwise it leads to re-initialization of the population. Algorithm.2 describes complete process of an improved multi-objective VM scheduling among Cloudlets in MCC.

Table 1. Simulation setup

Parameters	Value
Max Iteration	800
Crossover Probability	0.8
Mutation Probability	0.1
Size of the Population	300
Simulation Time	1000 Seconds

```
Algorithm 2 Multi-Objective Genetic Algorithm based Load Balancing
in MCC (MOGALMCC)
1.        Algorithm 1 generate initial population
2.        Calculate the Fitness function of Each individual by
using eq.(3.1), eq.(3.2), eq.3.3), eq.(3.4) and eq.(3.8) perform
crossover and mutation for generating offspring population.
3.        Parent chromosome Replace with newly generated child
population until finding suitable Cloudlet for VM scheduling.
```

SIMULATION SETUP

We have evaluated proposed MOGALMCC algorithm in NS-3 Simulation Environment. We observed resource scheduling among Cloudlets by sending a random number of request VMs by considering completion time, transfer time, load, available bandwidth and memory. We conducted extensive simulation by generating a random number of Cloudlets, requests and distance values for observing the performance of the proposed algorithm with a comparison of conventional existing algorithms. The required parameters for simulation are listed in Table 1.

EXPERIMENTED RESULTS

In this section, we have compared our model with existing Greedy, Round Robin, FIFO algorithms (X. Wang et al., 2014)(Yagoubi & Slimani, 2007).

Fig.2 illustrates task execution time before and after scheduling using MOGALMCC. The x-axis represents no of VM's and y-axis represents execution time in seconds. Dynamic VM scheduling using a genetic algorithm, the execution time of each VM is reduced considerably.

Fig.3 represents the VM's response time in seconds for GA, RR, and FIFO. The x-axis shows number of VM's and y-axis represents execution time in seconds. It is clearly noticeable that the MOGALMCC consumes less time compared with other existing algorithms.

Fig.4 shows VM's Makespan for MOGALMCC, RR, FIFO and Greedy. the x-axis represents no. of VM's or tasks and y-axis represent makespan in seconds. This graph clearly shows that our proposed model is more efficient compared with other existing algorithms. we have done simulation by using nearly 500 VMs.

Fig.5, the graph shows the imbalance among Cloudlets with and without proposed GA algorithm. X-axis for no. of tasks and y-axis for the degree of imbalance.

Fig.6 represents degree of imbalance among VM's by using different algorithms (RR, FIFO, and DLB, GA). We compared VM migration balancing among Cloudlets. VM migration means the no. of VM's re-assigned to other Cloudlets in the network area. Our proposed model show better performance compare with greedy and no migration methods.

Figure 2. Comparison of Execution Time before and after MOGALMCC

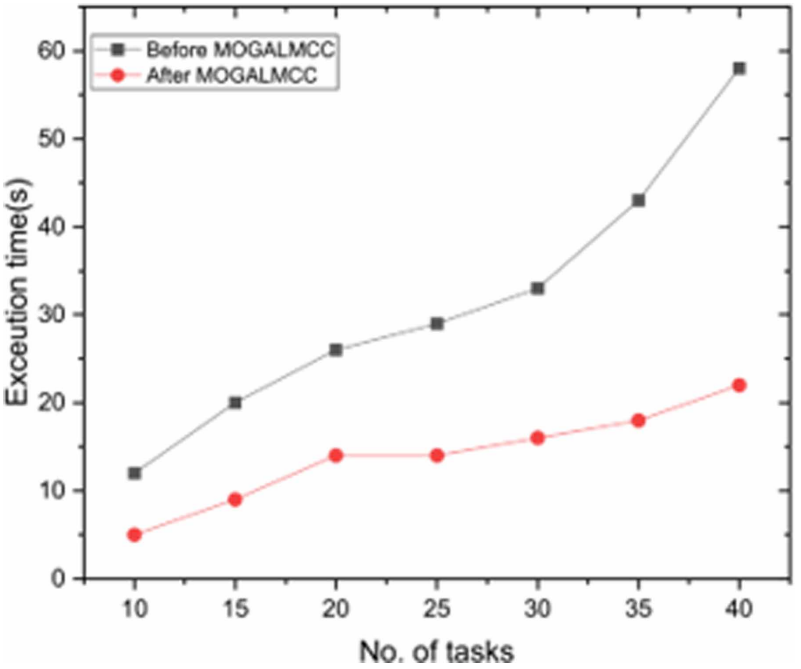


Fig.7, Fig.8, Fig.9 describes VM migration when a number of Cloudlets vary from 3-8 for GA and Greedy algorithm. In all the cases proposed method out performed compared with other algorithm. X-axis for no. of VMs and y-axis for number of VMs migration.

$$\text{Degree of Load Imbalance (DI)} = \frac{\max\left(j_{n,k}^{i}\right) - \min\left(j_{n,k}^{i}\right)}{avg\left(J_{n,k}^{i}\right)}$$

CONCLUSION

In this paper, we have implanted GA based VM load balancing on Cloudlets in MCC. The performance of the proposed method can be achieved considering different factors such as bandwidth, Memory, a load of Cloudlet server and distance to find optimal Cloudlet. This proposed technique works well for heterogeneous environment, which tries to bring Cloud resources close to mobile users. We have compared our proposed with other popular load balancing algorithm in MCC, which is evident that proposed GA based VM load balancing algorithm achieved better performance. In future, we plan to extend our work by considering other factor such as "task size" to improve the performance of the algorithm by reducing response time, VM failure rate.

Figure 3. VM Response Time for RR, FIFO, Greedy and MOGALMCC

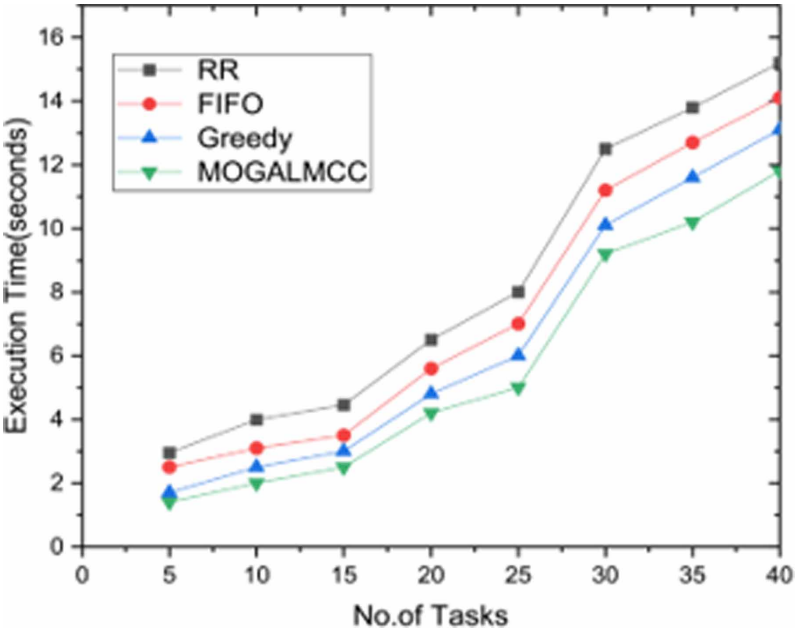


Figure 4. Comparision of Execution Time for RR, FIFO, Greedy and MOGALMCC

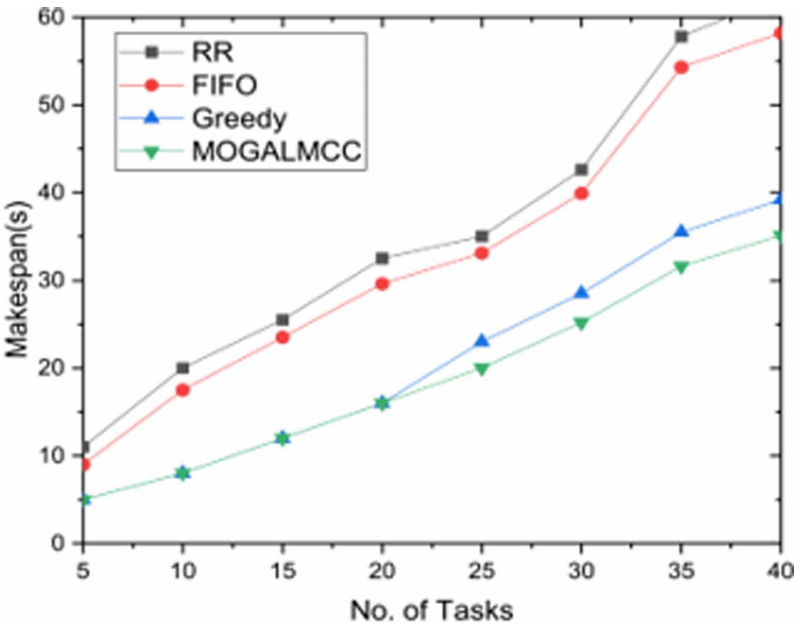


Figure 5. Comparison of Degree of Load Imbalance between VM's for before and after MOGALMCC

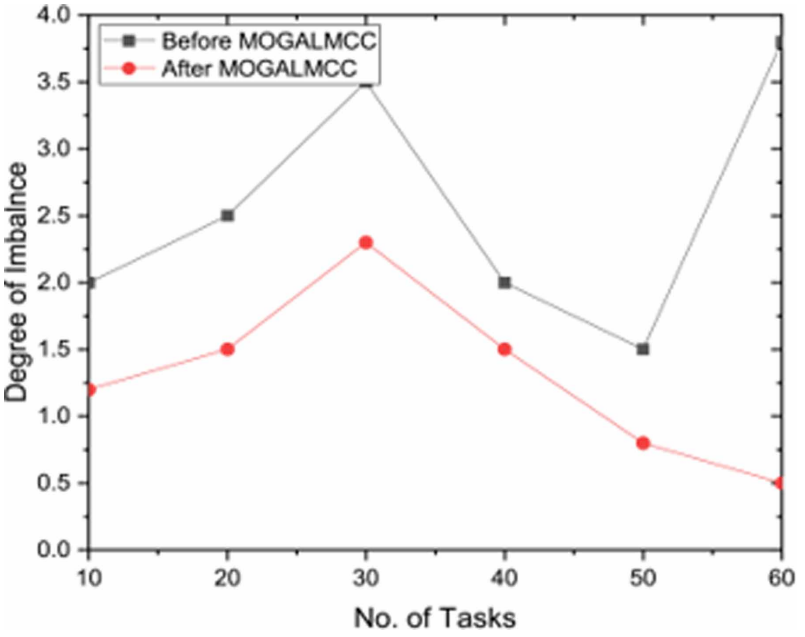


Figure 6. Comparison of Degree of Load Imbalance for RR, FIFO, Greedy and MOGALMCC

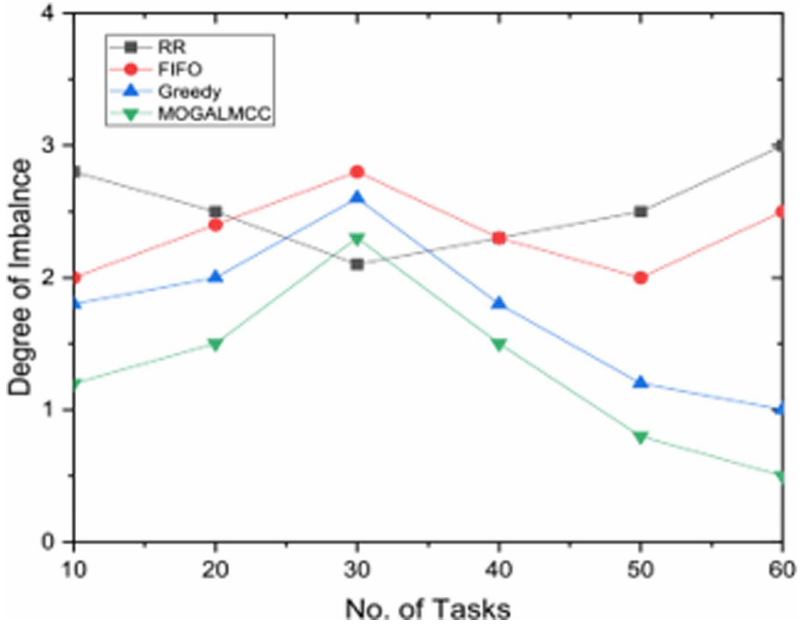


Figure 7. Comparison of VM Migration among 4 Cloudlets

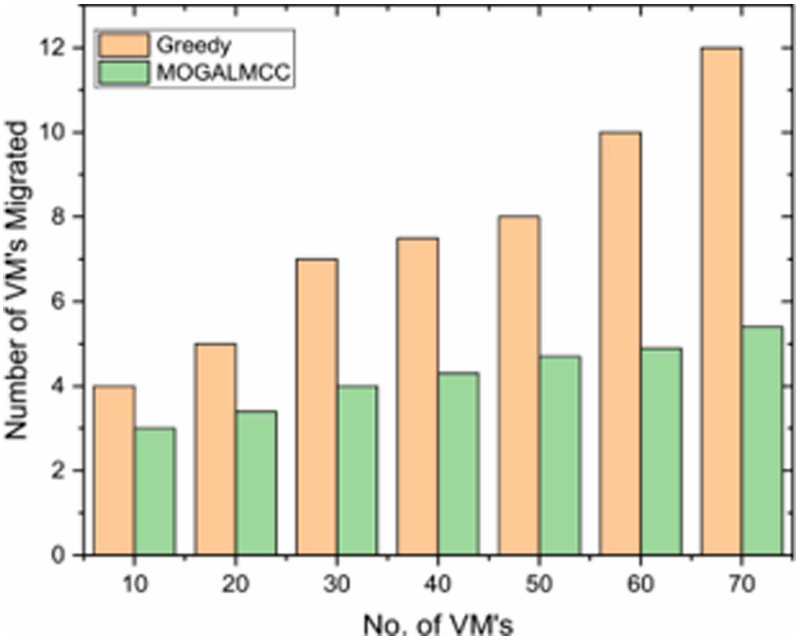


Figure 8. Comparision of VM Migration among 5 Cloudlets

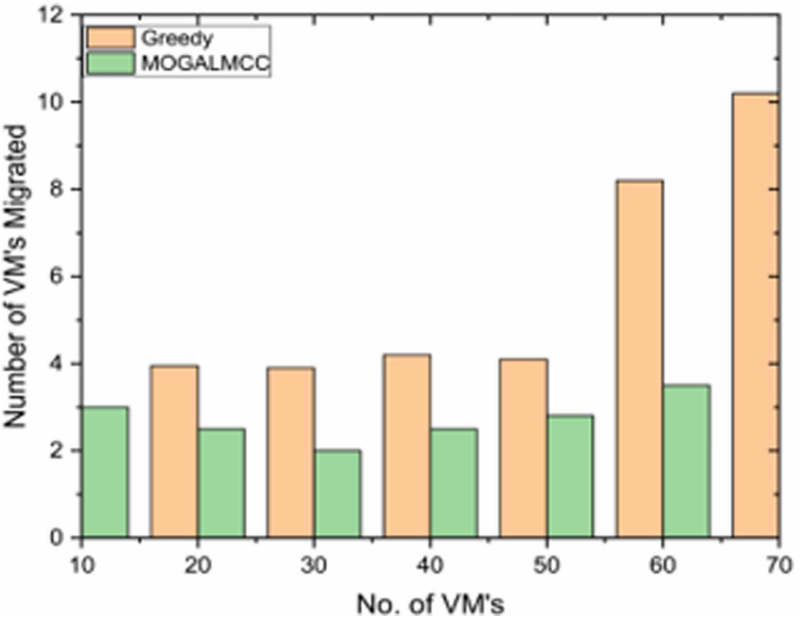
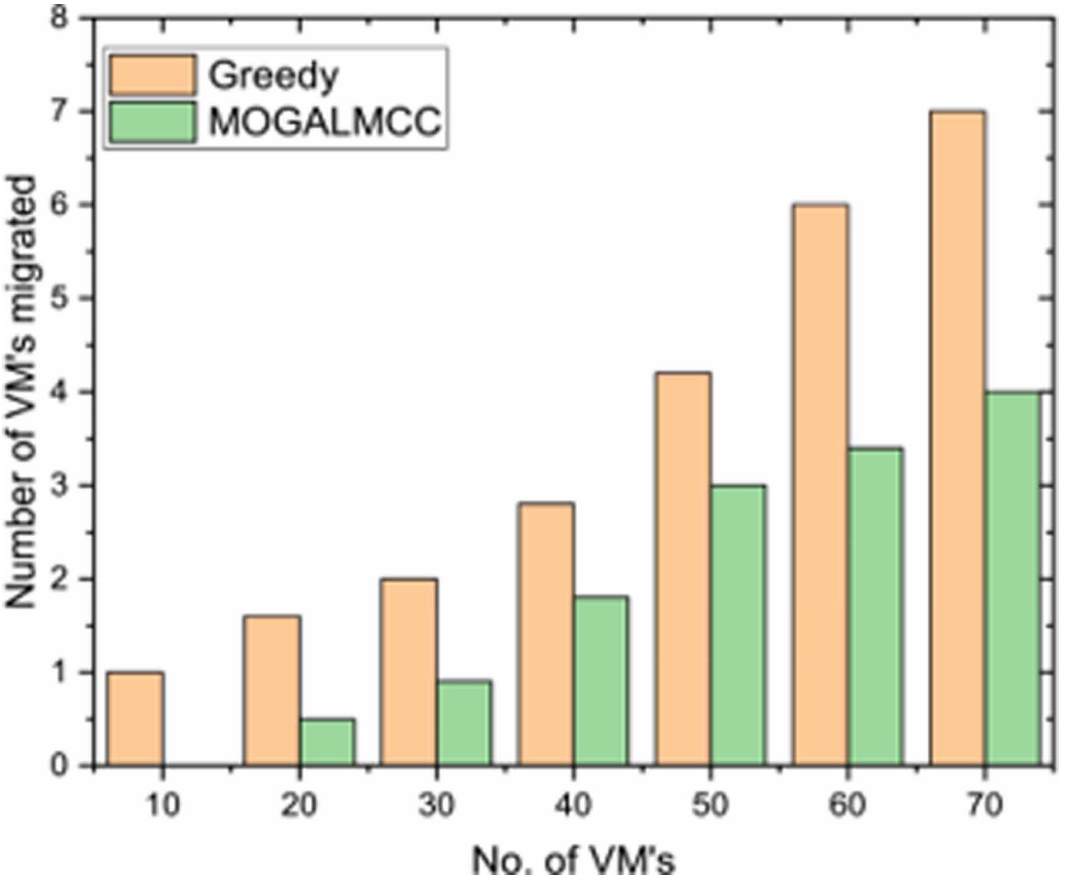


Figure 9. Comparision of VM Migration among 7 Cloudlets



REFERENCES

Bobroff, N., Kochut, A., & Beaty, K. (2007). Dynamic placement of virtual machines for managing sla violations. *Integrated Network Management, 2007. IM'07. 10th IFIP/IEEE International Symposium On*, 119–128.

Chun, B.-G., Ihm, S., Maniatis, P., Naik, M., & Patti, A. (2011). Clonecloud: elastic execution between mobile device and cloud. *Proceedings of the Sixth Conference on Computer Systems*, 301–314. doi:10.1145/1966445.1966473

Cuervo, E., Balasubramanian, A., Cho, D., Wolman, A., Saroiu, S., Chandra, R., & Bahl, P. (2010). MAUI: making smartphones last longer with code offload. *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, 49–62. doi:10.1145/1814433.1814441

Das, A. K., Adhikary, T., Razzaque, M. A., Cho, E. J., & Hong, C. S. (2014). A QoS and profit aware cloud confederation model for IaaS service providers. *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication*, 42. doi:10.1145/2557977.2558064

Das, A. K., Adhikary, T., Razzaque, M. A., & Hong, C. S. (2013). An intelligent approach for virtual machine and QoS provisioning in cloud computing. *Information Networking (ICOIN), 2013 International Conference On*, 462–467.

Devare, M., Sheikhalishahi, M., & Grandinetti, L. (2010). A Prototype Implementation of Desktop Clouds. *High Performance Computing Workshop*, 124–137.

Dinh, H. T., Lee, C., Niyato, D., & Wang, P. (2013). A survey of mobile cloud computing: Architecture, applications, and approaches. *Wireless Communications and Mobile Computing*, *13*(18), 1587–1611. doi:10.1002/wcm.1203

Gkatzikis, L., & Koutsopoulos, I. (2013). Migrate or not? Exploiting dynamic task migration in mobile cloud computing systems. *IEEE Wireless Communications*, *20*(3), 24–32. doi:10.1109/MWC.2013.6549280

Gkatzikis, L., & Koutsopoulos, I. (2014). Mobiles on cloud nine: efficient task migration policies for cloud computing systems. *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference On*, 204–210.

Gu, X., Nahrstedt, K., Messer, A., Greenberg, I., & Milojicic, D. (2003). Adaptive offloading inference for delivering applications in pervasive computing environments. *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference On*, 107–114.

Guo, L., Zhao, S., Shen, S., & Jiang, C. (2012). Task scheduling optimization in cloud computing based on heuristic algorithm. *JNW*, *7*(3), 547–553. doi:10.4304/jnw.7.3.547-553

Islam, M., Razzaque, A., & Islam, J. (2016). A genetic algorithm for virtual machine migration in heterogeneous mobile cloud computing. *Networking Systems and Security (NSysS), 2016 International Conference On*, 1–6.

Juhnke, E., Dornemann, T., Bock, D., & Freisleben, B. (2011). Multi-objective scheduling of BPEL workflows in geographically distributed clouds. *Cloud Computing (CLOUD), 2011 IEEE International Conference On*, 412–419.

Krishna, P. V., Misra, S., Nagaraju, D., Saritha, V., & Obaidat, M. S. (2016). Learning automata based decision making algorithm for task offloading in mobile cloud. *Computer, Information and Telecommunication Systems (CITS), 2016 International Conference On*, 1–6.

Kumar, K., & Lu, Y.-H. (2010). Cloud computing for mobile users: Can offloading computation save energy? *Computer*, *43*(4), 51–56. doi:10.1109/MC.2010.98

Li, J., Bu, K., Liu, X., & Xiao, B. (2013). Enda: Embracing network inconsistency for dynamic application offloading in mobile cloud computing. *Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing*, 39–44. doi:10.1145/2491266.2491274

Li, Z., Wang, C., & Xu, R. (2001). Computation offloading to save energy on handheld devices: a partition scheme. *Proceedings of the 2001 International Conference on Compilers, Architecture, and Synthesis for Embedded Systems*, 238–246. doi:10.1145/502217.502257

Li, J-f., Peng, J., Cao, X., & Li, H. (2011). A task scheduling algorithm based on improved ant colony optimization in cloud computing environment. *Energy Procedia, 13*, 6833–6840.

Liu, J., Li, Y., Jin, D., Su, L., & Zeng, L. (2015). Traffic aware cross-site virtual machine migration in future mobile cloud computing. *Mobile Networks and Applications*, *20*(1), 62–71. doi:10.1007/s11036-014-0534-7

Rahimi, M. R., Ren, J., Liu, C. H., Vasilakos, A. V., & Venkatasubramanian, N. (2014). Mobile cloud computing: A survey, state of art and future directions. *Mobile Networks and Applications*, *19*(2), 133–143. doi:10.1007/s11036-013-0477-4

Raju, D. N., & Saritha, V. (2016). Architecture for Fault Tolerance in Mobile Cloud Computing using Disease Resistance Approach. *International Journal of Communication Networks and Information Security*, *8*(2), 112.

Ramasubbareddy, S., & Sasikala, R. (2019). RTTSMCE: A response time aware task scheduling in multi-cloudlet environment. *International Journal of Computers and Applications*, 1–6. doi:10.1080/1206212X.2019.1629098

Rong, P., & Pedram, M. (2003). Extending the lifetime of a network of battery-powered mobile devices by remote processing: a markovian decision-based approach. *Proceedings of the 40th Annual Design Automation Conference*, 906–911. doi:10.1145/775832.776060

Satyanarayanan, M., Bahl, P., Caceres, R., & Davies, N. (2009). The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, *8*(4), 14–23. doi:10.1109/MPRV.2009.82

Sheikhalishahi, M., Devare, M., Grandinetti, L., & Laganà, D. (2011). A General-purpose and Multi-level Scheduling Approach in Energy Efficient Computing. *Closer*, 37–42.

Shieh, W.-Y., & Pong, C.-C. (2013). Energy and transition-aware runtime task scheduling for multicore processors. *Journal of Parallel and Distributed Computing*, *73*(9), 1225–1238. doi:10.1016/j.jpdc.2013.05.003

Somula, R., Anilkumar, C., Venkatesh, B., Karrothu, A., Kumar, C. S. P., & Sasikala, R. (2019). Cloudlet Services for Healthcare Applications in Mobile Cloud Computing. *Proceedings of the 2nd International Conference on Data Engineering and Communication Technology*, 535–543. doi:10.1007/978-981-13-1610-4_54

Somula, R., & Sasikala, R. (2018). Round Robin with Load Degree: An Algorithm for Optimal Cloudlet Discovery in Mobile Cloud Computing. *Scalable Computing: Practice and Experience*, *19*(1), 39–52. doi:10.12694/scpe.v19i1.1392

Somula, R., & Sasikala, R. (2019). A Honey Bee Inspired Cloudlet Selection for Resource Allocation. In *Smart Intelligent Computing and Applications* (pp. 335–343). Springer. doi:10.1007/978-981-13-1927-3_36

Somula, R., & Sasikala, R. (2019b). A Load and Distance Aware Cloudlet Selection Strategy in Multi-Cloudlet Environment. *International Journal of Grid and High Performance Computing*, *11*(2), 85–102. doi:10.4018/IJGHPC.2019040105

Somula, R. S., & Sasikala, R. (2018). A Survey on Mobile Cloud Computing: Mobile Computing+ Cloud Computing (MCC= MC+ CC). *Scalable Computing: Practice and Experience*, *19*(4), 309–337. doi:10.12694/scpe.v19i4.1411

Song, B., Hassan, M. M., & Huh, E. (2010). A novel heuristic-based task selection and allocation framework in dynamic collaborative cloud service platform. *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference On*, 360–367.

Taleb, T., & Ksentini, A. (2013). An analytical model for follow me cloud. *Global Communications Conference (GLOBECOM)*, 2013 *IEEE*, 1291–1296. doi:10.1109/GLOCOM.2013.6831252

Van, H. N., Tran, F. D., & Menaud, J.-M. (2009). SLA-aware virtual resource management for cloud infrastructures. *Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference On*, *1*, 357–362.

Wang, L., Zhang, F., Vasilakos, A. V., Hou, C., & Liu, Z. (2014). Joint virtual machine assignment and traffic engineering for green data center networks. *Performance Evaluation Review*, *41*(3), 107–112. doi:10.1145/2567529.2567560

Wang, S., Urgaonkar, R., He, T., Zafer, M., Chan, K., & Leung, K. K. (2014). Mobility-induced service migration in mobile micro-clouds. *Military Communications Conference (MILCOM)*, 2014 *IEEE*, 835–840.

Wang, X., Wang, Y., & Cui, Y. (2014). A new multi-objective bi-level programming model for energy and locality aware multi-job scheduling in cloud computing. *Future Generation Computer Systems*, *36*, 91–101. doi:10.1016/j.future.2013.12.004

Wang, Y., Chen, R., & Wang, D.-C. (2015). A survey of mobile cloud computing applications: Perspectives and challenges. *Wireless Personal Communications*, *80*(4), 1607–1623. doi:10.1007/s11277-014-2102-7

Yagoubi, B., & Slimani, Y. (2007). Task load balancing strategy for grid computing. *Journal of Computational Science*, *3*(3), 186–194. doi:10.3844/jcssp.2007.186.194

Somula Ramasubbareddy is pursuing his PhD in Computer Science and Engineering (CSE), from VIT University, Vellore, India. He did his M.Tech from JNTUA, Anantapur, India in 2015. His research areas are Mobile Cloud Computing, Network security, Distributed Computing, Computer Communications (Networks) and Algorithms, IOT.

Swetha E. is currently doing B.Tech in SV College of Engineering, Tirupati. Research interest includes Cloud Computing, Wireless Networks, IoT and Machine Learning . Published various papers in International journals and conferences.

Ashish Kr Luhach received Ph.D degree in department of computer science from Banasthali University, India and post graduated from Latrobe University, Australia. Since, Feb 2018, Dr. Luhach is working as Associate professor and Head for the department of Computer Science and Engineering. He has more than a decade of teaching and research experience. Dr. Luhach also worked with various reputed universities and also holds administrate experience as well. Dr. Luhach has published more 40 research paper in reputed journals and conferences, which are indexed in various international databases. He is Editor/Conference Co-chair for various conferences such ICAICR, IMTC, the proceedings published by Springer and in addition to this he is also editorial board members of various reputed journals. Dr. Luhach, received research excellence award in 2016 at Lovely Professional University for his research contribution during the academic year. He is member of CSI, ACM and IACSIT.

T. Aditya Sai Srinivas is pursuing his PhD in Computer Science and Engineering (CSE), from VIT University, Vellore, India. He did his M.tech from JNTUA, Anantapur, India in 2012. His research areas are Mobile Cloud Computing, Network security, Distributed Computing, Computer Communications (Networks) and Algorithms, IOT.