Journal of Ambient Intelligence and Humanized Computing (2020) 11:5477–5486
https://doi.org/10.1007/s12652-020-01903-5

ORIGINAL RESEARCH

A novel energy estimation model for constraint based task offloading in mobile cloud computing

S. Erana Veerappa Dinesh[1] · **K. Valarmathi**[1]

Received: 7 August 2019 / Accepted: 19 March 2020 / Published online: 8 May 2020
© Springer-Verlag GmbH Germany, part of Springer Nature 2020

Abstract

The utility of mobile applications has been increased enormously due to the advancements in science and technology to assist the users for various purposes. The main intention of integrating cloud services and resources with the mobile application is to reduce battery usage and to improve the efficiency of mobile devices. Thus the process of shifting a task that can be run on the cloud resources for assistance is referred to as task offloading. The process of task offloading is critical in the field of Mobile Cloud Computing. The major issue that pairs with task offloading are the communication cost estimation of the devices. To overcome the above-mentioned issues and to create an effective task offloading model, A Novel Mobile Cloud Computing framework called Rule Generation based Energy Estimation Model (RG-EEM) is designed. The energy required for executing the task in the local mobile device and cloud is estimated by implementing an energy estimation algorithm. Then a novel constraints specific rule generation algorithm is used to estimate the task execution time and memory utilization of the task, from which the decision has to be taken for offloading or local execution by considering all the possible affecting characteristics. Further, a novel task clustering and scheduling algorithm are implemented to execute the task in the cloud server effectively which will help in allocating similar tasks to a particular virtual machine in the cloud. The RG-EEM algorithm will also help in partitioning the task and parallel execution. The effectiveness of the RG-EEM technique is evaluated using the parametric measures and compared with the existing techniques.

Keywords Mobile cloud computing · Task offloading · Energy estimation · Constraint-based rule generation · Task scheduling · Prioritization

1 Introduction

The rapid advancements in mobile computing and wireless technology increase the mobile contributions to the resource utilization for network accessibility significantly. The mixture of mobile computing, cloud computing, and the mobile internet is termed as the Mobile Cloud Computing (MCC). It incorporates the benefits of all these technologies (Akherfi et al. 2018). The recent growth in cloud computing affords significant benefits to mobile users with higher resource sharing and usage. Because of this advantage, mobile cloud computing becomes abundant with applications in several categories such as health, entertainment, game, business, travel, news and social networking (Cardellini et al. 2016). The most primitive computing platform for several users is the mobile system such as smartphones. Various applications are excessively computation demanding to perform on a mobile system. These mobile applications provide the prospect of interlinking and sharing their data and task with cloud resources. The latest developed mobile applications support the usage of online resources to improve efficiency. Some of the freely available cloud services like online converters, calculators can be quoted as the standing examples of the cloud services which are available and prevalently known. But these computations require a large amount of energy, memory and time (Jararweh et al. 2016). Generally, mobile devices such as smartphones have some constraints like limited memory capacity, battery consumption, and processing capability. Every application in the mobile devices performs a series of a task which executes particular computations. But they require a large amount of energy consumption.

✉ S. Erana Veerappa Dinesh
evdresearch@gmail.com

[1] Department of Computer Science and Engineering, Anna University, Chennai, Tamil Nadu, India

Several existing methodologies such as power scheduling, smart batteries, energy-aware communication protocols, and efficient operating systems and applications are introduced for reducing the energy consumption of mobile devices (Altamimi et al. 2015). But still, they lack some constraints while computing the tasks. Hence task offloading is introduced for reducing the requirement of these constrains (Erana Veerappa Dinesh 2017). Task offloading from smartphones to the cloud hugely helps the cloud computing capability of the devices for reducing battery usage and persisting battery life. The major issue that pairs with task offloading are the communication cost estimation of the engaged devices. Computing and estimating the communicational cost will help in the evaluation of energy utilization and power wastage. Thus the tasks which has to be offloaded from s smartphone to a cloud server has to be decided based on some of the major factors like, task execution time, memory usage, processor utilization, network bandwidths, task upload time, task result download time, task allocation time, virtual machine scheduling time, local execution parameters, etc. Thus task offloading is a critical task in the field of mobile cloud computing and hence considered as the area of research. This process of task offloading is represented in Fig. 1.

Some of the traditional techniques perform task offloading to the cloud and helps in less energy consumption. A mobile terminal that performs task offloading will reach an optimal resource allocation solution for energy minimization by using the FDM for sub-band division for parallel execution of multiple mobile terminals, but this method is not suitable in the multi-base station and multi-cloud scenarios (Zhao et al. 2015). Another method computes the energy utilization and battery capability with the help of the radio transmission, power transmission and delay of uplink time. Similarly, a method uses the IaaS model along with the SaaS model to reduce the runtime component migration reduction to minimize energy consumption (Shiraz et al. 2015).

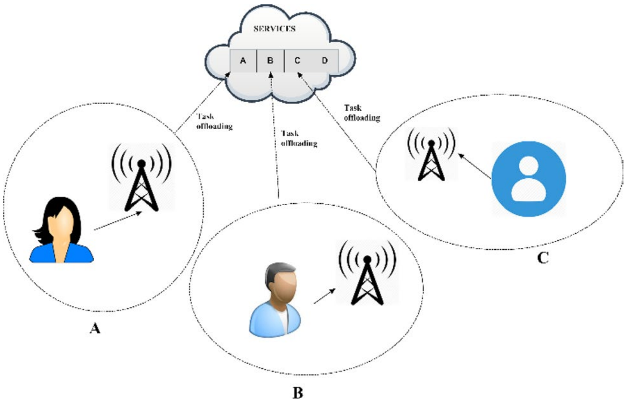

Fig. 1 Task offloading

These existing techniques have given considerable solution for task offloading, but there is no standard model which can assure the efficiency of the model that maintains a trade-off between the energy minimization and best cloud resource allocation of the task.

To overcome the above-mentioned issues and to create an effective task offloading model, A Novel Mobile cloud computing framework called RG-EEM has been designed. Initially, the energy-dependent parameters of the mobile devices are estimated like, current battery power, percentage of RAM utilization, available network bandwidth and the background applications which are working in the devices. Based on these parameters a novel energy estimation algorithm is implemented to find the energy required for executing the task in the local mobile device and cloud. Then a novel constraints specific rule generation algorithm is implemented to estimate the time taken for task execution and memory utilization of the task, from which the decision has to be taken for offloading or local execution on considering all the possible affecting characteristics. Further, a novel task clustering and scheduling algorithm are implemented to execute the task in the cloud server effectively which will help in allocating similar tasks to a particular virtual machine in the cloud. This algorithm will also help in partitioning the task and parallel execution.

The key objectives of this proposed work are given below:

- To create an effective task offloading model to improve the efficiency of the mobile devices by minimizing the battery usage
- To find the energy required for executing the task in the local mobile device and cloud
- To estimate the task execution time and memory utilization of the task, for decision making of offloading or local execution
- To execute the task in the cloud server effectively for allocating similar tasks to a particular virtual machine in the cloud.

The remaining of this paper structured as follows: The detailed descriptions of the related works about the task offloading in mobile cloud computing are studied in Sect. 2. The execution of the proposed technique is explained in Sect. 3. The comparative results of the proposed technique with traditional approaches are depicted in Sect. 4. Lastly, Sect. 5 concluded the proficiency of the proposed approach.

2 Related work

This section describes the traditional techniques that are used for task offloading in mobile computing. The advantages and disadvantages of the traditional approaches are also discussed as follows. Mao et al. (2017) optimized the scheduling for task offloading and allocation of transmit power jointly for Mobile Edge Computing (MEC) systems with multiple independent tasks. The main intention of this work was to minimize the energy consumption of the device and execution latency. In this technique, a low complexity suboptimal algorithm based on the alternating minimization was introduced for minimizing the weighted sum of the energy consumption of the device as well as execution latency. The major drawback of this system was its critical task offloading scheduling. Liu et al. (2017) studied a trade-off between power and delay in the context of task offloading under a multi-user MEC scenario. In this work, a new design was suggested in contrast with conventional system design depending upon the average metrics. This design considered some constraints such as reliability and latency. This network design achieved higher task arrival rates. But the higher consumption of energy was the major issue in this work. Zhou et al. (2018) proposed a context-aware mixed-integer programming model to offer offline optimal solutions for creating the offloading decisions and scheduling the tasks that were offloaded in heterogeneous mobile clouds. The main intention of this model was to reduce global task completion time. Also, a deterministic online algorithm was proposed in this work for solving the real-time problem. But this model offered poor scheduling performance due to the presence of malicious activities in the network.

Pu et al. (2016) introduced a novel mobile task offloading framework model called D2D Fogging based on the network-assisted Device to Device (D2D) collaboration. This framework helped to share the communication and computation resources by the mobile users more beneficially and dynamically through the control assistance by the network operators. The key determination of this D2D Fogging was to execute the tasks efficiently with minimal energy for the users. The advantage of this framework was that it adapted various circumstances like several users, task type, etc. But they could not handle the time-varying computation demands and wireless fading channels. Gu et al. (2019) presented an energy-efficient task scheduling strategy (EETS) for determining the task to be offloaded under various environments. This strategy introduced a cost estimation for determining whether the work was needed to be offloaded or not. This mechanism offered accurate results for efficient choosing of task offloading with reduced energy consumption. But it did not deliberate the power consumption on time waiting for results. Tran and Pompili (2017) considered a Mobile Edge Computing (MEC) enabled multi-cell wireless network in which every base station (BS) was fortified with an MEC server. It also helped the mobile users for implementing the computation tasks through task offloading. The task offloading gains of the users were improved by examining the problems of the Joint Task Offloading and Resource Allocation (JTORA) and this could be evaluated by reducing the energy consumption and task completion time. This approach offered better task offloading utility but the problem was that the suggested approach had the complexity of decision making and optimization.

Manikandan et al. (2019) investigated the scheduling policy for the collaborative execution in mobile cloud computing. The main intention of this design was to reduce the energy which was consumed by the mobile device during the time limit. This could be formulated as a problem related to the constrained shortest path in Directed Acyclic Graph (DAC) and this problem was solved by adopting the canonical LARAC algorithm. This collaborative execution of task reduced the energy consumption efficiently on mobile devices. But the structural properties for the optimal task scheduling was not established. (Qian et al. 2015) presented a system that included the capabilities of energy-aware computation offloading called Jade in android apps. This Jade helped to monitor the device, status of the application and decided the execution of the task automatically. This Jade reduced the burden for building applications with the ability of computation offloading. As a result, this system reduced the energy consumption of mobile devices and changed their offloading scheme dynamically based on the status of the mobile device. Jade computation offloading for the security concerns was still a challenging issue. Kaur and Mehta (2019) recommended a system called COSMOS which offered the computation offloading as a service for resolving the mismatch amongst the demand of computing resources in the mobile devices and these resources offered by the cloud providers. The two key challenges such as managing and sharing the cloud resources and handling the variable connectivity for decision making in offloading were solved by this COSMOS. But the optimization of energy consumption in mobile devices became a challenging issue in this work.

Guo et al. (2016) and Pan et al. (2019) provided an energy-efficient dynamic offloading and resource scheduling (eDors) scheme for reducing energy consumption and the accomplishment time of the application. Also, a distributed eDors algorithm was proposed for solving the optimization problem. This algorithm was comprised of three sub algorithms such as selecting the computation offloading, controlling the clock frequency and allocating the transmission power. In this scheme, the energy-efficient cost could be decreased by adjusting the CPU clock frequency of Smart Mobile Devices (SMD) optimally based on the dynamic voltage and frequency scaling (DVFS) technique. But this scheme had the effect of the SMD mobility on the computation offloading scheme. Yousafzai et al. (2019) suggested an active service migration (ASM) framework by utilizing the lightweight procedure for the deployment of a runtime distributed platform.

The suggested framework was employed the modest progressive and deployment procedures in MCC for computational offloading. It was observed that this framework reduced the energy efficiency cost and computational time required for the application compared to the conventional techniques. This depicted the lightweight nature of the suggested framework for computational offloading. But it had the overhead of the migration of the components. Kero et al. (2019) investigated the tradeoff among the energy consumption in wireless communication and local computation. Here, a wide-ranging set of measurement experiments were conducted by utilizing smartphones. It was the initial step for the development of accurate energy models to estimate the energy required for task offloading in smartphones. The major benefit of this model was that it did not depend upon the network parameters. But this model lost precision which became the issue.

Ragona et al. (2015) presented a model for the task offloading which was performed by the wearable devices in MCC. The major performance metrics that were measured in this analysis were the energy required for the computation and communications as well as the time required for data transmission and for obtaining the results back. Here the optimization could be achieved by either energy consumption or execution time. But the battery life for the wearable devices was limited only. Chen et al. (2016) considered a multi-user MCC system in which every mobile user had multiple independent tasks. While offloading the tasks in clouds, the multi-users shared the communication resources. The main intention of this work was to optimize the offloading decisions of all the users along with the communication resource allocation jointly. This helped to reduce the energy costs, delay and computation complexities for all users. The separable semidefinite relaxation was used for an efficient approximate solution. This approach offered optimal performance compared to the other techniques.

Hao et al. (2018) explained the task offloading for mobile edge computing. The applications of augmented reality become the most popular in computation based intensive tasks. The effective mobile edge computing provides the solution to follow the low latency demand. This research described the new design of task caching. It defines the caching of overall task applications and related information in the edge cloud. They estimated the join optimization technique for task caching and task offloading in edge cloud computing with the storage resource constraints. Solving the issues of mixed-integer programming was a challenging task. To solve this issue, this research proposed an efficient algorithm named as task caching and offloading (TCO). It mainly based on the alternating iterative method. In the end, the simulation results proved that the proposed TCO method superior performances with the minimum energy cost measurements.

Dinh et al. (2017) described the offloading in mobile edge computing with task allocation and frequency scaling. This research proposed the optimization technique for task offloading from a single mobile device (MD) to multiple edge devices. The main aim of this study is to reduce both energy consumption and task execution latency. The utilized optimization technique was the joint optimization method. It considers two cases for mobile devices such as elastic CPU frequency and fixed CPU frequency. They proposed the linear relaxation-based method with a semidefinite relaxation (SDR) method for an exhaustive search-based, SDR-based approach and fixed CPU frequency case. The evaluation results proved that the proposed SDR-based method achieved the highest optimal performance with the metrics of tasks' execution latency and energy consumption.

Pan et al. (2019) depicted the dependency aware computation offloading in mobile computing. The IT environment enables cloud computing at the edge of mobile devices. To reduce the battery power restrictions, service latency and to develop the service efficiency this research motivated the one-shot offloading with consideration of task dependency. It provides a more adaptive way comprehensive way to take both inherent dependencies of tasks and energy constraints. This paper mainly focused to analyze the issues of dependency aware task offloading in mobile edge computing. And minimizing the execution time of mobile applications on energy consumption. To solve those issues, it proposed a model-free approach based on the reinforcement learning and Q-learning approach for optimizing the offloading decision in the network environment.

Huynh et al. (2020) proposed the enhanced particle swarm optimization for task offloading in mobile computing. They evaluated the two-tier computation-offloading strategy in a heterogeneous network. The results are estimated computation-offloading decision strategy and joint resource-allocation for minimizing the computational time. They divided the optimization problems into two subproblems such as computation offloading and resource allocation decision. This algorithm provides a high-quality solution for mitigating optimization problems.

From the survey, it is observed that the traditional approaches have some drawbacks such as

- Inaccurate energy estimation
- High power consumption
- Hard to execute both the local and cloud server simultaneously
- Increased memory utilization
- High time complexity

Historically, there were various models are presented for offloading information with reducing cost, reaction time, vitality and battery life on the cloud. There is no much work done to consequently task offloading based on efficient clustering which can be run on the cloud. The proposed work is different from existing studies. We investigated the clustering and scheduling process in a cloud server.

3 Rule generation based energy estimation model

This section describes the RG-EEM technique for estimating the required bandwidth, power consumption, energy consumption and time in task offloading. The workflow of the RG-EEM technique is shown in Fig. 2.

This mobile computing system includes multiple mobile devices on the cloud. Each mobile device can communicate with the cloud by using a wireless channel. This system consists of M number of mobile devices and computational tasks (T).

$$M = \{1, 2, \ldots n\}$$

$$T = \{1, 2, \ldots t\}$$

We consider that the number of users higher than the number of computational tasks. Because some of the tasks contain high priority, it requested repeatedly and processed multiple times.

Initially, the parameters of the mobile device such as current battery usage, RAM utilization, bandwidth availability are estimated. Using these parameters, the energy requirement is computed. Based on the parameters, a constraint specific rule is created. After that, it checks a condition for task offloading. If the condition is satisfied, the task is forwarded to the cloud for execution. Else the task is executed in the local device using resource allocation and it optimizes the memory. On the other side, the task to be executed in

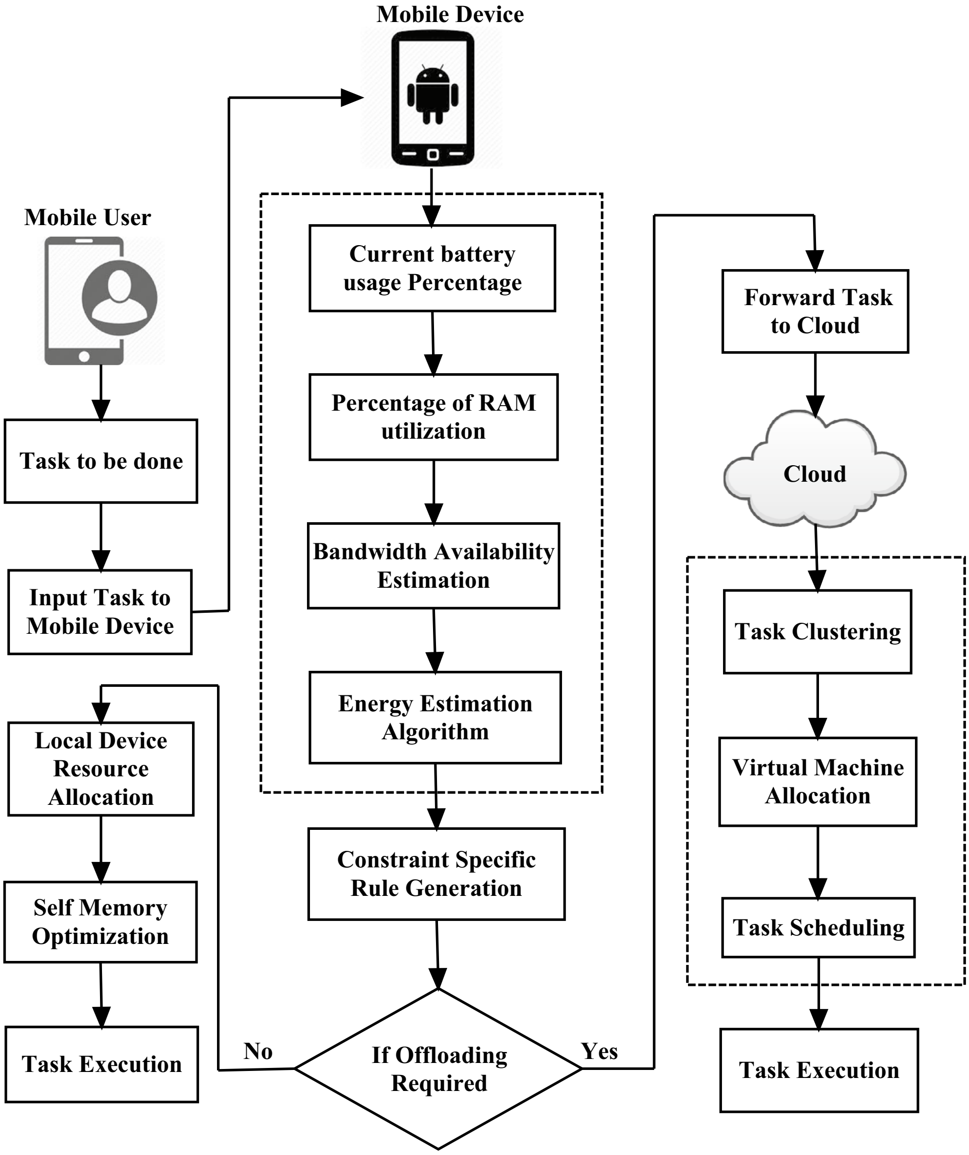
Mobile Device
Mobile User
Task to be done
Input Task to Mobile Device
Current battery usage Percentage
Percentage of RAM utilization
Bandwidth Availability Estimation
Energy Estimation Algorithm
Constraint Specific Rule Generation
If Offloading Required
No
Yes
Local Device Resource Allocation
Self Memory Optimization
Task Execution
Forward Task to Cloud
Cloud
Task Clustering
Virtual Machine Allocation
Task Scheduling
Task Execution

Fig. 2 Flow diagram of RG-EEM

the cloud are clustered. And these tasks are then offloaded to the virtual machine which allocates and schedules the task for execution. Finally, the task is executed based on the prioritization method.

3.1 Initial parameter extraction

The android emulator is created the android mobile which creates the application for executing a task. Initially, the mobile device extracts some factors such as the percentage of current battery usage, transmission, and reception byte rates. Subsequently, it extracts the memory status such as available memory, used memory, and total memory. Also, it extracts the RAM status like available RAM, used RAM and total RAM.

3.2 Energy estimation

After extracting the initial parameters, the energy requirement for executing a task in a mobile device is also estimated. First, the task and size of the task are initialized. The execution time taken for one byte of the task is also initialized. The memory utilization required for one byte of the task is also initialized. After that, the energy required to execute a single byte, memory, bandwidth, and migration rates is fixed. Finally, the energy consumed to execute a task is estimated using these parameters.

Considering a mobile cloud participant (or) user who is using the mobile application to execute a task. Task has to be executed in the local mobile while the resources are available for the task, else it has to be offloaded. Let T be the input task and T_L be the size of the task. The energy of the task in the mobile device can be estimated using

$$Eg_T = (T_L \times Eg_B) + (T_L \times M_B \times Eg_M) + (T_L \times Eg_{Bw}) + (T_L \times Eg_{Mg}) + (T_L \times Et_B) \tag{1}$$

T_L represents task list, Eg_B represents Energy/Bandwidth rate, M_B represents Memory Utilization/Byte of task, Eg_M represents Energy/Memory rate, Et_B represents Execution time/Byte of task, Eg_{Mg} represents Energy/Migration rate.

The energy estimation algorithm is described as follows

```
Energy Estimation Algorithm
Input: Task, Task Size
Output: Estimated Energy
Step 1: Initialize Execution time/Byte ($Et_B$) of the Task
Step 2: Initialize Memory Utilization/Byte ($M_B$) of the Task
Step 3: Set Energy/Byte ($Eg_B$), Energy/Memory ($Eg_M$),
        Energy/Bandwidth ($Eg_{Bw}$), Energy/Migration
        ($Eg_{Mg}$) rates
Step 4: Estimate the energy of the task using equation (1)
```

3.3 Constraint specific rule generation

The certain parameters of the mobile devices such as utilized battery power, battery threshold level, transmission byte rate, received byte rate, available unused ram capacity and available unused memory are extracted. The memory and CPU utilization of a task is estimated. The memory requirement for the task execution is estimated by

$$R_{My} = \left(T_L \times M_B\right) \tag{2}$$

R_{My} represents a memory requirement for the task.

Then the CPU utilization of the task is estimated using the following equation,

$$R_{Cpu} = (T_L \times Et_B \times M_B) \tag{3}$$

With these extracted parameters, the constraint-based rule generation is created. It checks a certain condition of battery consumptions, memory, RAM and time requirement to decide on task execution. If the condition is satisfied, the task is executed in a local machine. Otherwise, the task is offloaded to the cloud for execution.

The algorithm for constraint specific rule generation is explained as follows

```
Constraints Specific Rule Generation Algorithm
Input: Task, Task Size, Estimated Energy
Output: Decision Output ($T_{Dec}$)
Step 1: Extract the parameters of the mobile devices
        $Bt_P$ ← Extract the Battery Power Utilized
        Set the $Bt_{Th}$ ← as the Battery Threshold Level
        $Tr_{By}$ ← Extract the transmission byte rate
        $Rc_{By}$ ← Extract the received byte rate
        $Av_{Ram}$ ← Available unused Ram capacity
        $Av_{My}$ ← Available unused Memory
Step 2: Estimate the required Memory using equation (2)
Step 3: Estimate the Cup utilization of the Task using
        equation (3)
Step 4: If ($Bt_P > Bt_{Th}$ &$T_L < Tr_{By}$)
Step 5:   If ($Av_{My} > R_{My}$ &&$Av_{Ram} > R_{Cpu}$)
              $T_{Dec}$ ← Execute Task in local Machine
          End if
Else if
          $T_{Dec}$ ← Offload Task to cloud
End
```

3.4 Task clustering and scheduling

The memory requirement for a single byte of a task is initialized. After that, the energy required for executing a single byte, memory, bandwidth, transmission and reception rates is fixed. The energy required for executing a task is computed using the following equation.

$$Vm\_Eg_T = (T_L \times VM\_Eg_B)) + (T_L \times VM\_M_B \times VM\_Eg_M) + (T_L \times VM\_Eg_{Bw}) + (T_L \times Et_B) \tag{4}$$

Also, the overall energy required for executing a task can be estimated as

$$Eg_O = \sum_{i=1}^{N} Vm\_Eg_T(i) \tag{5}$$

Based on the estimated energy consumption, the available task list is sorted. After that, a threshold level is fixed for energy consumption. If the estimated energy consumption is less than the threshold level, then they are clustered as one. The clustered tasks are scheduled for execution. The execution of tasks is carried out based on prioritization. Here the tasks are prioritized using Shortest Job First (SJF) method. By using this method, the tasks are executed and send the output to the mobile users.

The algorithm for task clustering and scheduling is illustrated as follows

```
Task clustering and scheduling algorithm
Input: Vmlist, avail Ram, Avail Memory, Transmit and Receive Bytes, $Eg_T$, TaskList($T_L$)
Output: Executed task
Step 1: Initialize Memory Utilization/Byte ($VM\_M_B$) of the Task
Step 2: Set Energy/Byte ($VM\_Eg_B$), Energy/Memory ($VM\_Eg_M$), Energy/Bandwidth ($VM\_Eg_{Bw}$) rates
Step 3: Estimate the energy of the task using equation (4)
Step 4: Compute Overall Energy Required for executing a task by using equation (5)
Step 5: Set threshold ($Eg_{Th}$) = $Eg_O/T_L$
Step 6: Sort the Available task list based on the estimated energy consumption
Step 7: For j=1: Number ($T_L$)
            If ( $Eg_T(j) \leq Eg_{Th}$ )
                Set $T_L(j)$ to the $C_1$
            Else
                Set $T_L(j)$ to the $C_2$
            End if
Step 8: End for
Step 9: Execute task and send output
```

4 Performance analysis

This section demonstrates the performance analysis of the RG-EEM technique for task offloading in mobile devices. This methodology is tested with the experimental setup which is comprised of five different kinds of smartphones such as Samsung Galaxy S3, Samsung Galaxy Note 3, HTC Nexus One, Blackberry Z10 and LG Nexus 4 (Altamimi et al. 2015). They can be accessed with WLAN, 3G or 4G networks. Using these networks, smartphones can download and upload the files from and to the cloud. The supply power is power the smartphones and notices the power consumption simultaneously. These readings are recorded on the laptop during the experiments. The effectiveness of this proposed technique is analyzed using parameters like total energy consumption, time delay, execution cost, and average power consumption.

Also, the performance of the RG-EEM technique is compared with several existing techniques.

4.1 Evaluation parameters

4.1.1 Total energy consumption

The total energy consumption is the total energy consumed by various system parts. It is represented by Joule. It is also referred to as average energy consumption. This can be explained using the following equation

$$E_{total} = E_{WNI} + E_{OS} \tag{6}$$

where E_{total} is the total energy consumed for task execution, E_{WNI} is the energy consumed by the Wireless Network Interface and E_{OS} is the energy consumed by the Operating System.

4.1.2 Delay time

Delay time defines the overall response time to compute offloading decision update and the proposed system requires the minimum amount of delay time to update the task.

4.1.3 Cost

The cloud environment optimizes the task, it defines the cost of computing the overall tasks in the cloud and is directly proportional to the amount of taken time for computation.

5 Results and discussion

The performance of the proposed and existing techniques are analyzed in this section. They can be analyzed using parametric measures such as total energy consumption, time delay, execution cost, and average power consumption.

Table 1 shows the total energy consumption for the existing and proposed techniques. Here the energy consumption for various system parts is measured with the varying file size.

The graphical representation of the total energy consumption is shown in Fig. 3. Here the energy consumed by the proposed technique is reduced by 33.33% than the Energy Model for the file size of 25 MB. From this, it is observed that the proposed technique achieved less energy consumption compared to the existing technique.

The Fig. 4 describes the comparison of time delay for the proposed method and existing methods such as online learning based task offloading (OTO) and OTO under the condition of γ =0:9995 (Wang et al. 2019a, b). γ has a great influence on the OTO method. In this table, the time delay is evaluated for 1000, 1500, 2000, 2500, 3000, 3500 and 4000 number of tasks. the proposed method had 0.25% of the time delay for 1000 tasks, 0.26% of time delay for 1500 tasks, 0.29% of 2000 tasks. Likely 0.32%, 0.35%, 0.25%, 0.32% of time delay contains 2500, 3000, 3500 and 4000 tasks. This statistical analysis proved that the proposed method takes a minimum amount of time to execute the overall process compared to existing OTO and OTON methods.

The Fig. 5 depicts the proposed method of cost comparison with existing methods such as MoEOD, and Clone cloud (Manukumar and Muthuswamy 2019). Proposed method 0.8 s of cost for 10 tasks, 1.8 s of cost for 20 tasks, 3.9 s for 30 tasks, 4.9 s for 40 tasks, 6.9 s for 50 tasks and 7.9 s for 60 tasks. Form this comparison the proposed method had minimum cost and the existing method had the highest cost for the overall process. This analysis proved that the proposed method takes a minimum amount of cost based on the number of tasks compared to existing methods (Table 2).

The average power consumption for both the existing and proposed techniques are shown in Fig. 6. Every device offers consistent results but the only variation is the power consumption. Hence this figure shows the average power consumed by each smartphone for uploading and downloading the file vover various networks such as WLAN, 3G, and 4G.

Table 1 Total energy consumption

File size (MB)	Total energy consumption (Joule)				
	Total	E_{OS}	E_{WNI}	Energy model (Altamimi et al. 2015)	RG-EEM
5	30	20	10	10	5
10	60	40	15	15	10
15	85	62	20	20	15
20	115	80	30	30	20
25	140	105	38	38	28

6 Conclusion and future work

In Mobile Cloud computing the most critical task is the task offloading. The major issue that pairs with task offloading are the communication cost estimation of the devices. To overcome the above-mentioned issues and to create an effective task offloading model, a novel mobile cloud computing framework called Rule Generation based Energy Estimation Model (RG-EEM) has been designed. A new energy estimation algorithm is implemented to find the energy required for executing the task in the local mobile device and cloud. Then a novel constraints specific rule generation algorithm is implemented to estimate the task execution time and memory utilization of the task. This helps to decide for offloading or local execution by considering all the possible affecting characteristics. Further, a novel task clustering and scheduling algorithm are implemented to execute the task

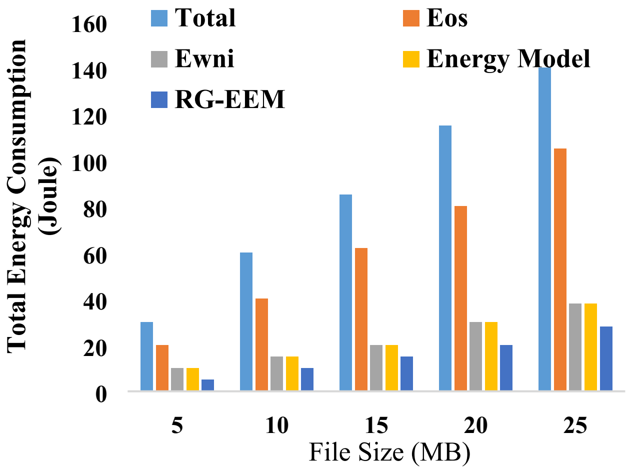

Fig. 3 Total energy consumption

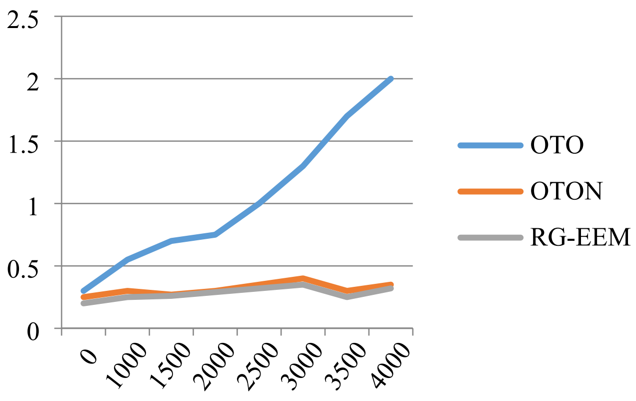

Fig. 4 RG-EEM method of time delay comparison with existing methods

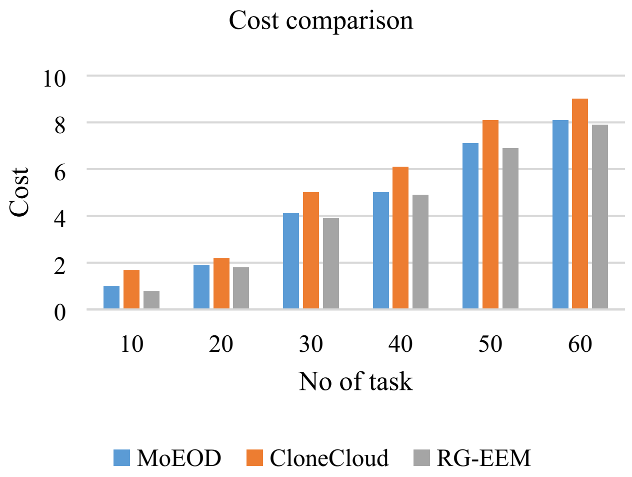

Fig. 5 RG-EEM method of cost comparison with existing methods

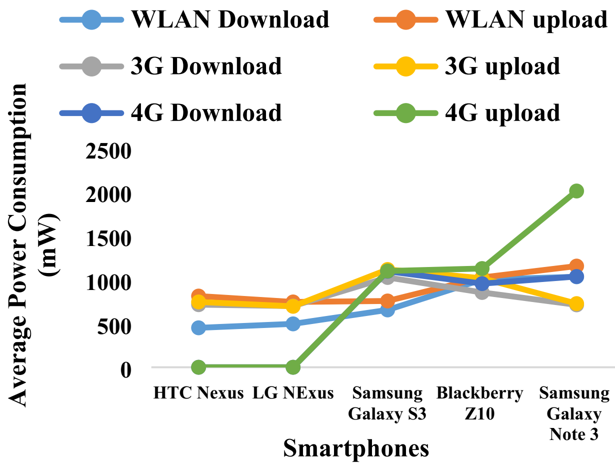

Fig. 6 Average power consumption

Table 2 Average power consumption (mW)

Network	Activity	Smartphones				
		HTC Nexus	LG NExus	Samsung Galaxy S3	Blackberry Z10	Samsung Galaxy Note 3
WLAN	Download	450	495	654	1000	1032
	Upload	812	745	756	1024	1154
3G	Download	714	698	1024	853	712
	Upload	745	695	1115	1015	726
4G	Download	NA	NA	1095	954	1035
	Upload	NA	NA	1098	1126	2010

in the cloud server effectively which will help in allocating similar tasks to a particular virtual machine in the cloud. This algorithm will also help in partitioning the task and parallel execution. The effectiveness of the proposed model is evaluated and compared with the existing models. The proposed model effectively reduced the overall communication cost. Also, the proposed model performs effectively in task offloading with reduced energy consumption and improves the efficiency of the system. It outperforms while comparing with existing techniques on the scale of performance parameters like total energy consumption, time delay, execution cost, and average power consumption. However, the disadvantage of the proposed centralized framework is that it requires the frequent updates of information for optimizing the system performances.

References

Akherfi K et al (2018) Mobile cloud computing for computation offloading: issues and challenges. Appl Comput Inform 14(1):1–16

Altamimi M et al (2015) Energy cost models of smartphones for task offloading to the cloud. IEEE Trans Emerg Top Comput 3(3):384–398

Cardellini V et al (2016) A game-theoretic approach to computation offloading in mobile cloud computing. Math Progr 157(2):421–449

Chen M-H et al (2016) Joint offloading decision and resource allocation for multi-user multi-task mobile cloud. In: Communications (ICC), 2016 IEEE international conference on, IEEE

Dinh TQ et al (2017) Offloading in mobile edge computing: task allocation and computational frequency scaling. IEEE Trans Commun 65(8):3571–3584

Erana Veerappa Dinesh S (2017) EDTOS: energy-aware dynamic mobile task offloading and scheduling model in mobile cloud computing. Int J Adv Res Trends Eng Technol IJARTET 4(1):41–47

Gu L et al (2019) Energy efficient task allocation and energy scheduling in green energy powered edge computing. Future Gener Comput Syst 95:89–99

Guo S et al (2016). Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing. In: INFOCOM 2016-the

35th annual IEEE international conference on computer communications, IEEE, IEEE

Hao Y et al (2018) Energy efficient task caching and offloading for mobile edge computing. IEEE Access 6:11365–11373

Huynh LN et al (2020) Efficient computation offloading in multi-tier multi-access edge computing systems: a particle swarm optimization approach. Appl Sci 10(1):203

Jararweh Y et al (2016) The future of mobile cloud computing: integrating cloudlets and mobile edge computing. In: 2016 23rd international conference on telecommunications (ICT), IEEE

Kaur P, Mehta S (2019) Efficient computation offloading using grey wolf optimization algorithm. In: AIP conference proceedings, AIP Publishing LLC

Kero A et al (2019) An adaptive approach towards computation offloading for mobile cloud computing. Int J Inf Technol Web Eng IJITWE 14(2):52–73

Liu C-F et al (2017) Latency and reliability-aware task offloading and resource allocation for mobile edge computing. arXiv preprint arXiv:1710.00590

Manikandan J et al (2019) Collaborative task execution for application as a general topology in mobile computing under communication cost model. Int Res J Eng Technol (IRJET) 6(10):1335–1340

Manukumar ST, Muthuswamy V (2019) A novel multi-objective efficient offloading decision framework in cloud computing for mobile computing applications. Wirel Pers Commun 107(4):1625–1642

Mao Y et al (2017) Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems. In: Wireless communications and networking conference (WCNC), 2017 IEEE, IEEE

Pan S et al (2019) Dependency-aware computation offloading in mobile edge computing: a reinforcement learning approach. IEEE Access 7:134742–134753

Pu L et al (2016) D2D fogging: an energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration. IEEE J Sel Areas Commun 34(12):3887–3901

Qian H, Andresen D (2015) Reducing mobile device energy consumption with computation offloading. In: Software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD), 2015 16th IEEE/ACIS international conference on, IEEE

Ragona C et al (2015) Energy-efficient computation offloading for wearable devices and smartphones in mobile cloud computing. In: Global communications conference (GLOBECOM), 2015 IEEE, IEEE

Shiraz M et al (2015) Energy efficient computational offloading framework for mobile cloud computing. J Grid Comput 13(1):1–18

Tran TX, Pompili D (2017) Joint task offloading and resource allocation for multi-server mobile-edge computing networks. arXiv preprint arXiv:1705.00704

Wang J et al (2019a) Edge cloud offloading algorithms: issues, methods, and perspectives. ACM Comput Surv CSUR 52(1):1–23

Wang K et al (2019b) Learning-based task offloading for delay-sensitive applications in dynamic fog networks. IEEE Trans Veh Technol 68(11):11399–11403

Yousafzai A et al (2019) Process migration-based computational offloading framework for IoT-supported mobile edge/cloud computing. In: IEEE internet of things journal

Zhao Y et al (2015) Energy-efficient task offloading for multiuser mobile cloud computing. In: Communications in China (ICCC), 2015 IEEE/CIC international conference on, IEEE

Zhou B et al (2018) An online algorithm for task offloading in heterogeneous mobile clouds. ACM Trans Internet Technol TOIT 18(2):23

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.