Int. J. Communications, Network and System Sciences, 2018, 11, 175-185
http://www.scirp.org/journal/ijcns
ISSN Online: 1913-3723
ISSN Print: 1913-3715

Energy Aware Task Assignment with Cost Optimization in Mobile Cloud Computing

Irfan Ali Jamali, Abdullah Lakhan, Abdul Rasheed Mahesar, Dileep Kumar Sajnani

Department of Computer Science in Southeast University, Nanjing, China
Email: jamali.irfan@yahoo.com, abdulrasheed_mahesar@yahoo.com, Abdullah@seu.edu.cn, Sajani.dileep@gmail.com

How to cite this paper: Jamali, I.A., Lakhan, A., Mahesar, A.R. and Sajnani, D.K. (2018) Energy Aware Task Assignment with Cost Optimization in Mobile Cloud Computing. *Int. J. Communications, Network and System Sciences*, **11**, 175-185.
https://doi.org/10.4236/ijcns.2018.118010

Received: August 15, 2018
Accepted: August 28, 2018
Published: August 31, 2018

Abstract

In this paper, we are investigating the power consumption of mobile device while performing offloading system. The offloading system is way in which mobile application can be divided into local and remote execution in order to alleviate the CPU energy consumption. However, existing offloading systems do not consider data transfer communication energy while performing mobile offloading system. They have just focused on mobile CPU energy consumption. In this paper, we are investigating the energy consumption mobile CPU and communication energy collaboratively while performing mobile offloading for complex application. To cope up with the above problem, we have proposed Energy Efficient Task Scheduler (EETS) algorithm, whose aim is to determine optimal tasks execution in offloading system in order to minimize mobile CPU and communication energy. Simulation results show that EETS outperforms as compared to baseline approaches.

Keywords

Mobile Cloud Computing, Energy-Efficient Task Offloading, EETS, Mobile Offloading

1. Introduction

It can be anticipated that more and more applications are to be converted into mobile cloud computing (MCC) [1]. MCC is a framework which augmented the capabilities of mobile computing with rich re-source cloud computing. On the other hand, offloading is the technique which offloads the compute intensive tasks of application to the remote cloud while keeping the total cost as small as possible [2]. Many efforts have been made on computational offloading in mobile cloud computing environment to achieve their objectives such as energy efficient task assignment, minimize total energy consumption and vice versa [3].

In this paper, we are studying the application partitioning and energy efficient task scheduler in mobile cloud environment. In real practice, a real time application such as face recognition is composed of tasks whether offload or not offload tasks whereas considering, network, device and work standing throughout application execution remain challenged [4].

Energy Efficient Task Assignment has three techniques for the application execution in a mobile device; for example full offloading where energy hungry tasks computationally offloaded to the surrogate server it could be local or remote cloud. In order to minimize total energy another approach non-offloading retained all tasks locally; extensive tasks consume much more energy of the device. However, we have proposed EETA (energy efficient task assignment) algorithm and that is a partial offloading technique so that it minimizes and prolongs the device energy as well as communication energy. EETA is the Minimization problem and well known as the NP-hard problem; it is not trivial to solve. However, the energy efficient task scheduler still facing a number of challenges as listed follows Application weighting: every application has different weight with respect to tasks. The application is composed of multi-size tasks. Energy efficient application partitioning and task assignment play an important role in performance. Previously, proposed scheme such as static analysis could not guarantee the optimal and energy efficient task assignment due to adaptively change in mobile device status and workload. A real time inference algorithm must be proposed to be able to tackle run time situation of mobile cloud application.

Network adoption: Application offloading to the cloud server introduces extra communication burden. The behavior of the network changes from time to time. While, during peak hours due to high traffic most of network nodes are affected by congestion. A network adaptive type real time algorithm must be proposed which copes with proceeding challenging.

Dynamic and Effective partitioning: fine grained, has multiple processing costs such as local execution and remote execution. Dynamic and effective assignment of tasks required real time algorithm.

Paper sections organized as follows Section 2 tells about related work and motivation. Section 3 explains the system description. Section 4 tells about system model and problem formulation. Section 5 is about simulation and results.

2. Related Work

Offloading is a technical way to offload the energy-hungry tasks to the external surrogate servers; nevertheless, many efforts have been made in offloading Scheme to achieve multiple objectives under different cost models. Some of the applications including face-recognition, 3D Gaming and Augmented Reality, could take a significant amount of time due to their computationally intensive nature. Processors for mobile devices are gradually getting faster year by year; however, with-out aid from special purpose hardware, they may not be fast enough for those computationally intensive applications. In this section, we de-

scribe the most prominent solutions of mobile cloud computing (MCC) and high-light their main shortcomings. MCC is composed of the capabilities of mobile computing and cloud computing. Cloud computing brings mobile devices a great number of computing resources through virtualization technology. Computation offloading has great attention nowadays in regard academic as well as industry. Research literature has been made many efforts on the computation offloading with their respective objective (*i.e.*, energy consumption, execution time minimization).

Could computing can offer computing source, and users can use these to decrease the amounts of computation on mobile systems and save energy. Thus, cloud computing can save energy for mobile users through computation offloading [5] [6]. However, migrate a large amount of data in a low-bandwidth network will cost much more energy than local computation. Offloading [7] [8] [9] is a method to offload part of the computation from the mobile device to another resource-rich platform A number of modern works intend different methodologies to offload computation for specific applications [10] [11] [12]. In an additional part, a lot of works have a discussion about on how to pre-estimate the real boost in terms of energy [13]. For a code complication, offloading might consume more energy than that of local processing when the size of codes is small [14]. The study [15] suggests a program partitioning based on the assessment of the energy consumption (communication energy and computation energy) before the program execution. The optimal program partitioning for offloading is calculated based on the trade-off between the communication and computation costs to do offloading in the dynamic environments, Tang *et al.* [16] consider three common environmental changes: power level, connection status, and bandwidth. But they just ex-plain the suitable solutions for offloading for different environments separately Chun *et al.* [17] present a system to partition an application in dynamic environments. The proposed system follows three steps with different requirements related to the application structuring, the partitioning choice, and the security.

3. Problem Description

In this paper, we are investigating the power consumption of mobile device which has linear relationship with process energy and communication while performing offloading system. The proposed model will be discussed in the following subsections.

3.1. Application Scenario

Elastic application clutches with several tasks in the course of data dependency. In Algorithm 1 EETA (Energy Efficient Task Assignment) the input is call graph G (V; E). However, two independent Disjoint set respectively. Whereas, E is the communication between caller and caller that could be Ptr between two tasks or two disjoint circles. EETA has a lot of variables; W is the disjoint set weight, R,

mobile device available resources, D device current workload, F mobile service rate (*i.e.*, processing power, velocity) and *TP*, *RP* is required power and total power of the current device. Each task instruction can be measured execution per second.

3.2. System Model and Problem Description

We analyze the application tasks as call graph as shown in **Figure 1(a)**. Furthermore, application partitioned into consumption graph as shown in **Figure 1(b)**. We show the application system as call graph G (V; E). However, V has a two disjoint subset of tasks such as V_l m and V_{rc} local disjoint set and cloud disjoint set respectively. Whereas, E is the communication between callers and called that could be *Ptr* between two tasks or two disjoint circles. EETA has a lot of variables, W is the disjoint set weight, R, mobile device available re-sources, D device current workload, F mobile service rate (*i.e.*, processing power, velocity) and T P, RP are required power and total power of the current device.

3.3. Energy Model

Energy consumption of mobile devices depends on the computation and communication loads. To explore the energy consumption of each task, we suppose the task computation requires I instructions. The task needs to deal with D bytes of data and will generate D' bytes result we use B to stand for current network bandwidth. It will task D/B seconds to transmit and receive data. We define our task as follows: Definition 1. [Application Task] $T(I, D, D')$.

A mobile application task T has I instructions to be executed. The task uses D bytes input data and generates D bytes output data. The mobile system consumes P_c (watt per instruction) for computing and P_{tr} (watt per second) for sending and receiving data. If we choose to execute our task using offloading, we need to send our code and data to server. So the total energy consumption is:

$$E_{off} = PC * I + P_{tr} * \frac{D}{B} \tag{1}$$

Equation (1) shows the non offloading power consumption while performing the mobile application execution locally.

Suppose the output data D' is k times smaller than the original data D, we use compress ratio k to describe the relationship between input data and output data: $\left(D = D \in k\right)$. In order to simplify the compression algorithm impact on the size of output data when the size of input data is different, we just consider the application task with fixed ratio. If the mobile device performs the task, the energy consumption is:

$$E_{Non} = P_c * I \tag{2}$$

Equation (2) shows that non offloading system has no communication cost. The local energy consumption is linearly influence due to process instruction.

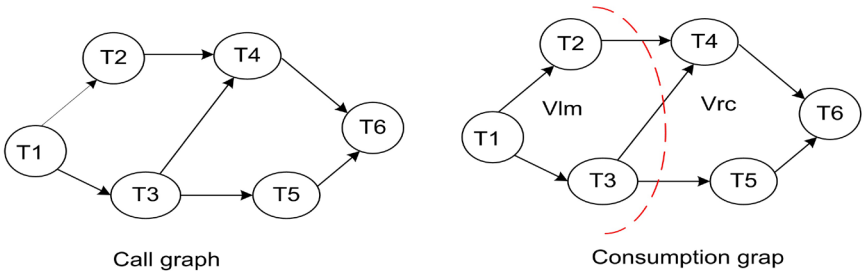

Figure 1. Application scenario.

$$E_{par} = P_c * I + P_{tr} * \frac{D'}{B} \tag{3}$$

Equation (3) shows that offloading decision can be made on either task offload or not, if task offloaded the communication energy will be incurred due to wireless communication cost.

3.4. Problem Formulation

The amount of energy saved is:

$$E_{\text{save}} = \frac{E_{\text{offloading}} - E_{\text{Non}}}{E_{\text{part}}} * 100\% \tag{4}$$

$$E_{\text{save}} = \frac{E_{\text{offloading}} - E_{\text{Non-offloading}}}{E_{\text{offloading}}} * 100 \tag{5}$$

$$P_{tr} * \frac{D}{B} - P_c * I - P_{tr} * \frac{D'}{B} = P_{tr} = (1-k)\frac{D}{B} - P_c * I$$

$$Min_Z = P_{tr} * \frac{D}{B} - P_c * I - P_{tr} * \frac{D'}{B} = P_{tr} = (1-k)\frac{D}{B} - P_c * I \tag{6}$$

In order to minimize mobile energy consumption Equations (3)-(5) must be minimized.

$$Min_Z = PC * I + P_{tr} + \frac{D}{B} \tag{7}$$

Algorithm 1: EETA

Require: G (V,E) input call graph
Ensure: Optimize E={ V_l+V_c+P_{tr}}
1: Declaration W;Net;R;D; Φ ; μ TP, RP
2: **Function** Profiling (E={V_l+V_c+P_{tr}})
3: **if** T < W and W < R **then**
4: **if** RP ≤ TP and W < R **then**
5: E ← R Φ
6: **else**
7: APP Solving {Net,D,T}
8: APP Partitioning {V_l,V_c} via Algorithm 2
9: V_l ← R Φ
10: Partition migration{ V_c+P_{tr}}
11: V_c ←R,μ follow Algorithm 3
12: **Retrun** E

In Algorithm 1, step-1 shows different variables for application partitioning and resource mapping. Step-2 profiles the application before the start via application task characteristics (*i.e.*, weight, and required power). Steps-3-4-5 shows that if the requirement of the application is catered and resources are available, retained the application tasks locally. Step-7 makes offloading decision based available device power status, required power status, program weight, network status and cur-rent device workload before execution of the diligence.

Pace 8 - 11 make application partitioning according to the characteristics and mapping the resource to required disjoint sets (*i.e.*, V_l retain locally and V_c offloaded to the remote cloud). Finally, step-12 return the energy efficient task assignment by looking at both de-vice and communication energy simultaneously. Time complexity is iteratively energy efficient and calculated via $\boldsymbol{log}|V|E$. Algorithm 1 only works on partial offloading against either non-offloading or full offloading.

Algorithm 2: local Mapping Resource

Require: Vl ϵV ; Vl UV;
Ensure: Optimize E(Vl ϵV)
1: Declaration W;R;D;W Φ ; RP
2: **Function** Mapping (E={Vl})
3: **if** E(Vl < W and W < D **then**
4: E ←R,RP Φ
5: **else**
6: E (Vc ← R,μ)
7: **retrun** E

Algorithm 2, follow Algorithm 1, mapping the required power and memory (*i.e.*, processor process and storage) for un-offloaded disjoint (e.g., $\boldsymbol{V_l}$) retained locally. Since, Steps 3 - 6, is the assignment process of tasks (*i.e.*, un-offloaded) by perspective of energy and resource utilization. It iteratively chooses the tasks which is following precedence sequence order in the given environment, that could be adaptively change (*i.e.*, device status and workload). The time complexity is polynomial and energy efficient (*i.e.*, $O\left(N+N\right)^2$)

Algorithm 3: Cloud Mapping Resource

Require: Vc ϵV ; Vc UV;
Ensure: Optimize E(Vc ϵV)
1: Declaration W;R;K;W;Th;μ;Bup, Bdown
2: **Function** Mapping (E={Vc+Ptr})
3: **if** E(Vc < W and W < K)**then**
4: E (Vc+Ptr) ← R,μ follow FIFO order
5: **if** Ptr (Bup, Bdown < Th) **then**
6: Re- Partitioning
7: **else**
8: E(Vl ←RorVc ←R,μ)
9: **return E**

Algorithm 3, follow Algorithm 1, mapping the required power and memory (*i.e.*, processor process and storage) for computationally offloaded disjoint (e.g., $\boldsymbol{V_c}$) migrated to the server k. Since, Steps 3 - 6, is the assignment process of tasks (*i.e.*, offloaded) by perspective of energy (*i.e.*, communication and process (queue and execution)) and resource utilization follow FIFO order. It iteratively chooses the tasks which is following precedence sequence order in the given en-

vironment that could be adaptively change (*i.e.*, server status and workload). The time complexity is polynomial and energy efficient *i.e.*, $O(N+N)^2$.

4. Perform Evaluation

The mobile application holds several energy-hungry tasks in order to prolong the battery power and average energy utilization of the system proposed algorithm EETA has significant simulation results. In the meantime, it can adapt to runtime environment changes as compared to existing methods. To evaluate the EETA energy efficient algorithm we need to know three kinds calibration values such as: Unchanging standards: some set of parameters are fixed by the application developer during design analysis such as power conservative elements are required to be fixed for different kinds of devices. Since proposed work has fixed values during simulation.

- **Explicit principles:** Speedup factor of cloud computing and mobile cannot be controlled over the time. Furthermore, bandwidth B = upload and download values are varies due to environmental changes, set of tasks with different data size may suffer due to environmental changes and it could be reasoned for degrading QoS (quality of service) and performance
- **Premeditated standards:** These kinds of standards supposed to be changed by device Characteristics and input parameters of an application tasks Application program is calculated by program pro-file after the application has been started for the period of run time and synchronously adaptive. Corresponding performance is calculated based on catered schemes that could be depicted as an always non offloadable and always off loadable. The consequences can be handled in the following way:
- **Partial offloading:** This scheme offers remote tasks offloaded to the remote cloud for execution and application native tasks computed locally, our proposed EETS algorithm is based on partial offloading.
- **Always Non-Offloading:** This is property and a characteristic of the application tasks since al-l computation happens inside the mobile device. This scheme might be harmful and less efficient since limited constrained device not able to tackle compute and energy-hungry tasks inside device results higher battery consumption and average performance simulated go down.
- **Always Full Offloading:** Cloning the image of tasks moved from mobile device to the remote cloud is the effective way as compared to Non-Offloading. It is directly proportional to the speed factor of the cloud and bandwidth corresponding to the network while adaptively variation in the speed factor and bandwidth the full offloading scheme shows bad performance and less efficient in the case of the result.

This scheme encompasses on both always Non-Offloading and always Full Offloading schemes, it could have better outcomes even though deviation in speedup factor as well as wireless in excess of the point of time. Our proposed system mobilized the saved cost in the following way:

$$OSC = 1 - \frac{EETA}{Nonoff} * 100\% \quad (7)$$

Whereas OSC (Offloading save cost) is relative percentage difference of proposed algorithm EETS and baseline approaches as shown in Equation (8), in order that minimizes the total cost (*i.e.*, mobile and communication energy) through application execution.

4.1. Task Energy Calculation

An application program such as tasks (granularity could be a method or class or object) can be observed via program profiler. Even though, each task is required to power consumption and memory for the execution. It could be executed either on local device with service rate ϕ_m or remote execution is followed by μ_m. However, a set of cloud servers can be represented by $k \in K$. Nevertheless, cloud servers are homogeneous in nature and have the same speed and storage capability.

4.2. Workload Analysis

We have examined four real-time application workload such as LINPAC math tool, 3D-Game EEG Beam 3D Game, Augmented Reality (gesture application) and Face Recognition application. Each workload is characterized by some properties such as tasks size, memory requirements, and required power consumption execution time. The individual application has benchmark application features, each task is required executed with a given threshold it could be energy or execution time bound. We can be seen that Figure 2 speedup factor F synthetically and synchronously clutches have whipping influence over the energy consumption and application performance. In view of the fact that, proposed algorithm EETA has better consequences over traditional schemes *i.e.*, full offloading and Non-Offloading. A number of tasks generated randomly, and arrival rates follow uniform distribution and it could be a Poisson process.

4.3. Perform Evaluation and Comparison

We have experienced a large number of experiments based on some factor such as speedup factor F and network bandwidth. Ultimately, speedup factor and bandwidth have thumping stimulus over device and network energy consumption. In the first experiment, the legislation variables are fixed such as a number of tasks, speedup factor F, and bandwidth. Whereas, speedup factor 2 means your processing speed is best else B = 1 responds slow ratio regarding network speed as shown in Figure 2.

Figure 3 shows that after being tested the different applications workload and granularity we have supposed to be reached on the final conclusion that every individual is required power consumption and it must be worked under given threshold value. Whereas, proposed method EETA is again leading better performance over traditional schemes.

Whereas, application energy consumption on the mobile devices depends

upon some factors such as bandwidth and speed factors generally, in mobile cloud paradigm speed up usually consider together just as mobile CPU speed and cloud CPU speed while performing offloading. It is can be seen from Figure 4, our proposed algorithm outperforms on different bandwidth and speedup factors as compared to baseline algorithm in terms of power consumption. Even Though, we have tested all algorithms exclude benchmarks application on face recognition application, it can be observed that power consumption of face recognition is improved 30% in the proposed algorithm EETA.

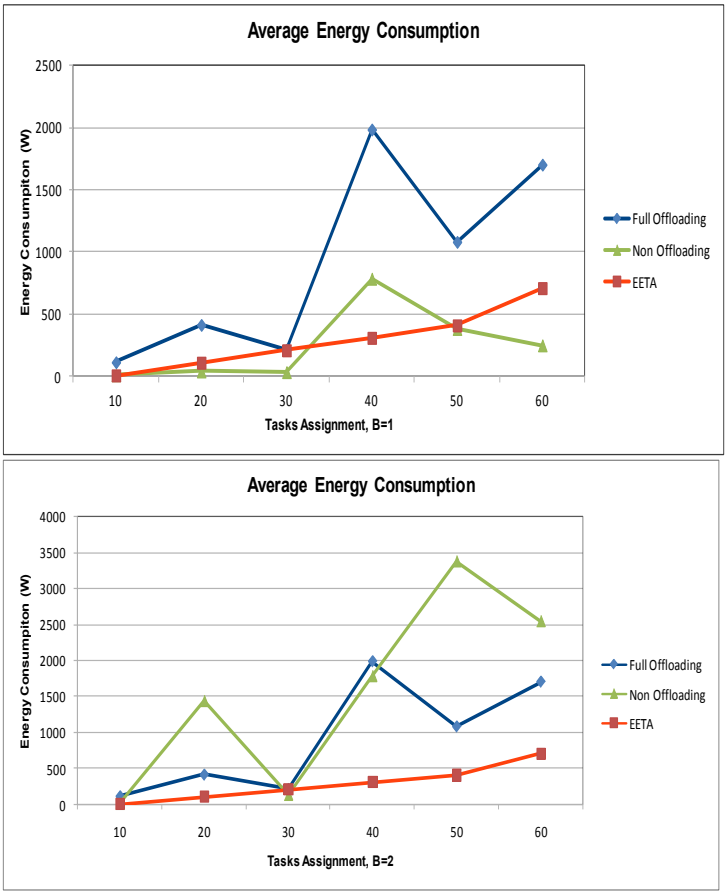

Figure 2. Energy efficient task assignment F = 2, B = 1.

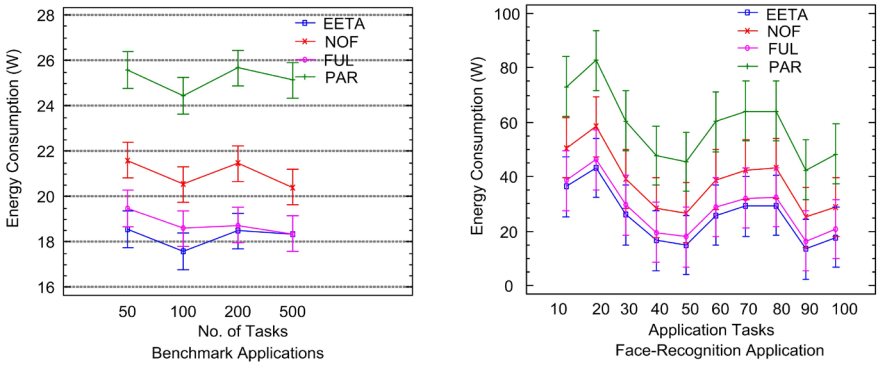

Figure 3. Energy efficient task assignment F = 2, B = 1.

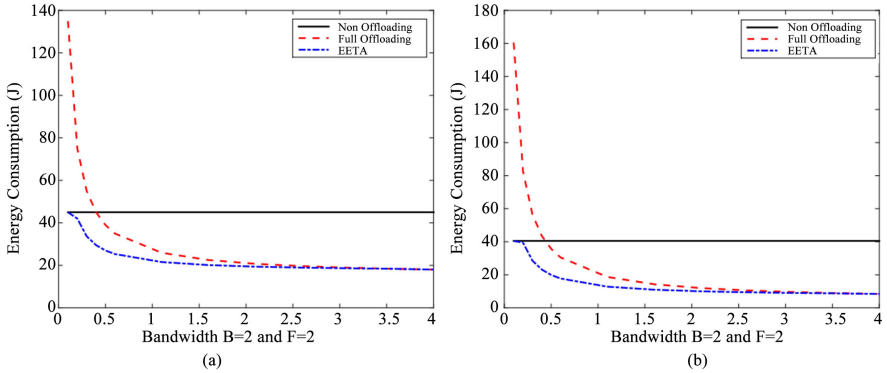

Figure 4. Different types of energy consumptions. (a) Energy consumption; (b) Energy consumption.

5. Conclusion

The EETA framework proposed in this paper does not consider the power consumption on idle and waiting time for task result and disk usage. Dynamic adaption could happen during application execution life cycle *i.e.*, network and device profiling change by over time. The preceding condition produces the inaccurate offloading result. In the future, we will propose dynamic and effective energy efficient algorithm which would be able to cope with environmental changes (e.g., bandwidth and workload change). They consider that the problem will be handled by dynamic optimization scheme.

Acknowledgements

I am really appreciated my authors friends, they have taken crucial role during research and paper designing.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Satyanarayanan, M., Bahl, P., Caceres, R. and Davies, N. (2013) The Case for VM-Based Cloudlets in Mobile Computing. *IEEE Pervasive Computing*, **8**, 14-23. https://doi.org/10.1109/MPRV.2009.82

[2] Dinh, H.T., Lee, C., Niyato, D. and Wang, P. (2014) A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches. *Wireless Communications and Mobile Computing*, **13**, 1587-1611. https://doi.org/10.1002/wcm.1203

[3] Baliga, J., Ayre, R. and Hinton, K. (2011) Green Cloud Computing: Balancing Energy in Processing, Storage, and Transport. *Proceedings of the IEEE*, **99**, 149-167. https://doi.org/10.1109/JPROC.2010.2060451

[4] Barbera, M.V., Kosta, S., Mei, A. and Stefa, J. (2013) To Offload or Not to Offload? The Bandwidth and Energy Costs of Mobile Cloud Computing. 2013 *Proceedings IEEE INFOCOM*, Turin, 14-19 April 2013, 1285-1293. https://doi.org/10.1109/INFCOM.2013.6566921

[5] Yang, K., Ou, S. and Chen, H.H. (2008) On Effective Offloading Services for Resource Constrained Mobile Devices Running Heavier Mobile Internet Applications. *IEEE Communications Magazine*, **46**, 56-63. https://doi.org/10.1109/MCOM.2008.4427231

[6] Kumar, K. and Lu, Y.H. (2010) Cloud Computing for Mobile Users: Can Offloading Computation Save Energy? *Computer*, **43**, 51-56. https://doi.org/10.1109/MC.2010.98

[7] Portokalidis, G., Homburg, P., Anagnostakis, K. and Bos, H. (2010) Paranoid Android: Versatile Protection for Smartphones. *Proceedings of the* 26*th Annual Computer Security Applications Conference*, Austin, 6-10 December 2010, 347-356.

[8] Chen, E.Y. and Itoh, M. (2010) Virtual Smartphone over IP. 2010 *IEEE International Symposium on* "*A World of Wireless, Mobile and Multimedia Networks*" (*WoWMoM*), Montrreal, QC, 14-17 June 2010, 1-6. https://doi.org/10.1109/WOWMOM.2010.5534992

[9] Wen, Y., Zhang, W. and Luo, H. (2012) Energy-Optimal Mobile Application Execution: Taming Resource-Poor Mobile Devices with Cloud Clones. 2012 *Proceedings IEEE INFOCOM*, Orlando, FL, 25-30 March 2012, 2716-2720. https://doi.org/10.1109/INFCOM.2012.6195685

[10] Rudenko, A., Reiher, P., Popek, G.J. and Kuenning, G.H. (1998) Saving Portable Computer Battery Power through Remote Process Execution. *ACM SIGMOBILE Mobile Computing and Communications Review*, **2**, 19-26. https://doi.org/10.1145/584007.584008

[11] Mahesar, A.R., Lakhan, A., Sajnani, D.K. and Jamali, I.A. (2018) Hybrid Delay Optimization and Workload Assignment in Mobile Edge Cloud Networks. *Open Access Library Journal*, **5**, e4854. https://doi.org/10.4236/oalib.1104854

[12] Chun, B.G. and Maniatis, P. (2010) Dynamically Partitioning Applications between Weak Devices and Clouds. *Proceedings of the* 1*st ACM Workshop on Mobile Cloud Computing & Services*: *Social Networks and Beyond*, San Francisco, 15 June 2010, Article No. 7. https://doi.org/10.1145/1810931.1810938

[13] Soomro, A.A., Lakhan, M.A.R. and Khan, A. (2011) The Secure Data Storage in Mobile Cloud Computing. *Journal of Information & Communication Technology*, **9**, 142-150.

[14] Waseem, M., Lakhan, A. and Jamali, I.A. (2016) Data Security of Mobile Cloud Computing on Cloud Server. *Open Access Library Journal*, **3**, 1.

[15] Balakrishnan, P. and Tham, C.-K. (2016) Energy-Efficient Mapping and Scheduling of Task Interaction Graphs for Code offloading in Mobile Cloud Computing. *Proceedings of the* 2013 *IEEE/ACM* 6*th International Conference on Utility and Cloud Computing*, Washington DC, 9-12 December 2013, 34-41.

[16] Balakrishnan, P. and Tham, C.K. (2013) Energy-Efficient Mapping and Scheduling of Task Interaction Graphs for Code Offloading in Mobile Cloud Computing. *Proceedings of the* 2013 *IEEE/ACM* 6*th International Conference on Utility and Cloud Computing*, Dresden, 9 December 2013, 34-41. https://doi.org/10.1109/UCC.2013.23

[17] Ali, J.I., Lakhan, A., Kumar, D. and Mahessar, A.R. (2018) Energy Efficient Task Assignment Algorithm Framework in Mobile Cloud Computing. *GSJ*, **6**, 171.