International Journal of Grid Distribution Computing
Vol. 8, No.4, (2015), pp. 47-58
http://dx.doi.org/10.14257/ijgdc.2015.8.4.05

An Energy-efficient Approach based on Learning Automata in Mobile Cloud Computing

Mostafa Ghobaei Arani and Najmeh Moghadasi

Department of Computer Engineering, Islamic Azad University of Parand, Tehran, Iran
Department of Computer Engineering, Islamic Azad University of Mahallat, Mahallat, Iran
Email: mostafaghobaei@piau.ac.ir, najmeh_moghadasi@yahoo.com

Abstract

In recent years, using mobile devices, in daily life, has found a special place and because of the applicability of these devices are increasing the number of users day by day. Business companies have integrated them with cloud computing technology and have provided mobile cloud in order to improve using mobile devices and overcome the energy consumption of mobile devices. Therefore, energy efficiency is a fundamental factor for mobile devices and cloud computing has the potential to save energy and power of mobile devices. In mobile cloud computing, computations and storages of mobile devices applications are transferred to cloud data centers and mobile devices are used merely as user interface to access services. The cloud computing will help to reduce energy consumption of mobile devices. In this paper, a new approach is given to reduce energy consumption of based on Learning Automata in mobile cloud computing. Simulation results show that our proposed approach significantly saves energy consumption through determining the appropriate location for application.

Keywords: *Mobile Cloud Computing, Energy-efficient, Mobile device, Learning Automata*

1. Introduction

Today, due to many benefits and possibilities that are provided by cloud computing to users including unlimited storage capacity, easy and prompt access of users for personal information and data at any time and place, access of multiple users simultaneously on documents and projects has increased the use of this technology [1, 2]. On the other hand, mobile devices (such as, smart phones, tablets, etc.) have become increasingly essential part of human life and are considered the most effective and appropriate form of communication tools. Users of these products get rich experiences of different services through mobile applications (For example, applications of iPhone, Google, etc.). These applications are run on a mobile device or on external servers that operate through a wireless network. The rapid development of mobile cloud computing (MCC) [3, 4] has created strong trend in development of information technology and in advertising industry. However, mobile devices face with many challenges on their own resources (e.g. energy consumption, storage, and bandwidth) and communications (for example, portability, bandwidth and security). Resource constraints prevent significantly the development of tools quality and cloud computing is calculated widely as the next-generation infrastructure [5, 6].

Cloud computing offers advantage by allowing users to use infrastructures (for example, servers, networks and storage facilities),operating systems (for example, middleware services and operating systems) and software (for example, applications) that are provided by cloud providers (*e.g.*, Google, Amazon and Sales force) with low cost

ISSN: 2005-4262 IJGDC

[7,8,9]. In addition, cloud computing enables users to use required resources flexibly. As a result, mobile applications can be quickly prepared and can be published with minimal management effort or interaction of providers. With the explosion of mobile applications and cloud computing support for a variety of services to mobile users, mobile cloud computing was introduced as an integration of cloud computing and mobile environment. Mobile cloud computing provides new tool services for users of mobile devices in order to take advantage of cloud computing.

In cloud computing, energy efficiency has become a challenge for technology capabilities that provides numerous benefits for mobile devices with battery power through saving battery energy of mobile devices. Now, mobile characteristic has decided on cloud technology future; energy management is important for longevity of mobile devices. Therefore, energy efficiency of mobile devices is important since cloud computing does not allow to run existing applications with low energy. So, one of the challenges of mobile cloud computing is to supply and save energy. Energy is consumed in mobile cloud computing, communications or computation. There are solutions for energy optimization which can be effective in reducing power consumption to a great extent [5, 6]. In this paper, we develop an approach in order to reduce the energy consumption of mobile cloud computing using learning automata. The proposed approach reduces energy consumption of mobile cloud computing considering the location of application which can be in cloud or mobile device. The paper has been organized as follows: In Section II, Related works will be reviewed. The proposed approach is described in Section III and the fourth section is devoted to performance evaluation proposed approach and finally, in fifth section, conclusions and future works are discussed.

2. Related Works

Variety of research has been done in the field of reducing energy consumption in mobile cloud computing environment. Generally, energy consumption of mobile devices is related to both computation and communication sectors. Some research has concentrated on calculation and others on communication sector in order to reduce energy consumption. From another perspective, the approaches can be divided into two broad static and dynamic categories. In static approaches, action conditions and environment are predefined and fixed while in dynamic approaches, operating conditions and environment are variable and dynamic. Finally, we study on a number of static [10-13] and dynamic approaches [14-19] in order to reduce energy consumption:

Li et al [10], propose a program partitioning based on energy consumption estimate and before application execution. Several solutions are proposed in order to find the optimal decision for partitioning applications before data transmission. One solution is to provide a schema partition for data transfer computing tasks on mobile devices. This model provides a Figure of cost on computation time and data share during connection and then search space is simplified using branch and bound algorithm so that estimate solution is found. Experimental results show that saved energy is achieved through the pattern and is significant for some programs like Media bench [11], but authors have not reviewed the test results in a dynamic environment such as network miscommunication and changes in bandwidth.

Lahh et al [12], provide a framework for algorithm- based context-aware mobile services in order to choose a context-aware adapter. Authors consider different subjects such as device environment, user preferences and situational context. Firstly, the algorithm determines types of gaps that occur in a given context. A gap is introduced as a result of context changes. Then, algorithm specifies predetermined gaps reasons before storing service readout in order to improve the disconnect mode. Then, this algorithm selects an appropriate adapter for active user for each of detected gaps. Since the

relationship between a cause and an adapter is predetermined, appropriate action can be selected and executed. The advantages of this method: in the case of user preference context, this relationship can be investigated when mobile user's context is changed. The disadvantage of this method: causes, adapters and gaps of this model are predetermined and may lead to a lack of flexibility in practical use. Strengthened implementation is a technique that is used in order to overcome the limitations of smart phone based on calculation, memory and battery. Chun and colleagues [13], proposed architecture and considered these challenges through off line integrated implementation from telephone to cloud computing structures and implemented a repeated version of smart phone software. Mobile phones host memory client and computing applications. Some or all works are off-line in cloud and are repeated version of running system image. Results of enhanced implementation are re-combined at the completion of work. This approach uses virtualization repeated versions with poor coordination and simulated repeated versions of mobile devices in cloud for broad offline computations. So, it creates an impression that mobile users have stronger device with richer than reality. It also creates the impression that application developer has programmed such a powerful device with no need to manual division of application or proxy provision. Set up a duplicated version of device in cloud is based on cost policy that optimizes the execution time, energy consumption, financial costs and security.

One of the programs that is used in mobile cloud computing include location based applications (LBAs). Xiao Maa et al [14] provided solutions in order to reduce energy consumption in this field. These services have obtained the current position of user through using a localization software like GPS [15] and offer various services related to location. In this way, they discussed on LBAs and factors affecting energy consumption reduction in mobile cloud computing applications.

Zhong Yao *et.al* [16] presented a strategy for scheduling duties with energy efficiency that focuses on reducing amount of data transfer. Their aim is to identify which duty is suitable for cloud computing and which is not appropriate. They have created a model of energy consumption for each duty in order to resolve this issue. Finally, it has been evaluated on the basis of Media bench in order to show the effectiveness of scheduling mechanism. The mentioned technique suggests that a simple duty does not take a long time to run; therefore, there is no need to reduce offloading. Complex uses consume more energy than simple uses. So, it categorizes tasks according to their complexity. This method suggests that a duty can be implemented within cloud using offloading. The advantage of this method is that it obtains dynamically amount of consumed energy in both modes of implementation in cloud and mobile device. Technique uses compression in order to reduce the data volume. Its disadvantage includes task scheduler is limited in some cases and needs the selection of each task profile information in the beginning.

Giurgiu et al [17] provided a middleware application that can distribute automatically different layers of an application between server and device while it optimizes several parameters such as delay, data transfer, cost and etc. A module management is distributed at the center of this approach which automatically and dynamically determines when and what application modules must be offline in order to achieve optimal performance or minimum cost of general application (offline data transfer).

Guriu et al used AlfredO framework [18] in order to distribute application modules between mobile devices and servers. AlfredO framework allows users to analyze and distribute presentation and logic layer of application while data layer always remains on server side. Minimum requirements, user interface UI of application remains on client side. AlfredO is based on R-OSGi [19] that is a conceptual implementation of middleware model OSGi and allows Java applications to be analyzed into software modules. The advantages of this method include analysis and distribution of presentation and logic layer to users and remaining data layer in server and implementation of user interface on

mobile device side. The disadvantage of this method is that it is designed only for Java applications.

3. The Proposed Approach

As noted, one of the challenges of mobile cloud computing is to optimize energy consumption of mobile devices. Energy consumption of mobile cloud computing is divided into energy consumption of communication and computing sectors. In cloud computing an application can be run on mobile device or transferred to cloud and implemented there, then, generated output can be transferred to mobile device in order to perform the other functions and tasks. Examining whether it is better to implement program on mobile devices or cloud is a very important issue because it has a considerable effect on energy consumption. When application is transferred to cloud and computation is done in cloud, although, the energy that is used in calculation may be stored, but some energy is consumed for transferring data from cloud to mobile devices. We conclude that cloud computing energy is used in computing and communication sectors. The factors affecting energy consumption in computing sector includes energy consumption for an instruction (there is direct relationship between energy consumption per operation and power of CPU, RAM and mobile devices) and the number of instructions in a program. More the number of instructions, energy consumption will be further in computing sector. Factors affecting energy consumption in communication sector includes the number of input and output data for each application, network bandwidth and energy consumption for data transmission in network. Below is a brief description of learner automata and proposed algorithm.

3.1. Learning Automata

Learning algorithms struggle to improve their performance and features toward a special aim by knowing related environmental situations [20, 21]. Learning automata is kind of such algorithms which try to change its conditions probabilities, based upon responses come from surrounding then it shows special reaction in any conditions. Learning automata [22] is an abstract thing with limited actions. A learning automata's performance is in a manner that it selects one of its actions among set of its actions and applies it to the environment. Then the mentioned action will be evaluated by a random environment and automata select its next action based on response of environment. The method which automata use it for selecting next action will be determined according to the used learning algorithm. Along the process automata learns how to select optimal action. Environment includes all internal and external conditions which affect automata. Generally it is possible to present environment as a set:

$E \equiv \{\alpha, \beta, c\}$ in which α is set of inputs, β is set of outputs and c is environment penalties. Figure 1, shows relationship between learning automata and environment.

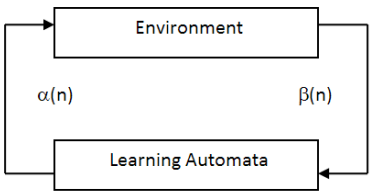

Figure 1. The Relationship Between Learning Automata And Enviroment

α is the number of actions automata is able to do, β is output of environment which differs according to the model and kind of environment Various models defined for probable environments are divided to 3 groups: P-Model, Q-Model and S-Model. In Q-model, environment outputs are discrete values of zero and

one and in S-model output is always a constant value between zero and one. In this paper, we use P-Model. In this model β might be one of the two values of zero or one. Zero means desirable action and the probability of an action increases while probability of other actions decreases. One means undesirable action and probability of current action decreases and probability of other actions increase so that after receiving amount of one from environment, automata change the action. C is the set of probability of penalties for actions of automata which defines as follow:

$$c_i = \Pr ob[\beta(i) = 1 \mid \alpha(i) = \alpha_i], i = 1, 2, ..., r \quad (1)$$

These values change over the time. Values of ci often are not clear and knowing them means thorough understanding of the potential environment that is not possible in most applications. Learning automata tries to know these values. In our sample environment will operate n times in any action:

- New input load (a) enters in environment.
- Includes reward and penalty comes from environment.

Probability vector is as P={0.5, 0.5}.

According to automata operation, higher penalty in non-optimized sample, lower the chance of its selection so it can assure us of selection an optimum sample

If n repetition selects a_i so that we have in (n +1) repetition:

Received a favorable response:

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)] \quad (2)$$

$$p_j(n+1) = (1-a)p_j(n) \quad \forall j \quad j \neq i \quad (3)$$

Received an unfavorable response:

$$p_i(n+1) = (1-b)p_i(n) \quad (4)$$

$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j \quad j \neq i \quad (5)$$

Now, we initialize the automata parameters:

α = a set of actions = {implementing program in cloud, implementing program in mobile device} = { α_i , α_j }

As a result, the number of actions r is equal to 2. Also, i value is equal to i=1,2, ..,n.

According to these values, initial probability is equal to $p_i(n) = 0.5$.

In this method, reward and punishment are used where reward parameter is shown with variable a, and penalty parameter is shown with variable b.

3.2. The Proposed Algorithm

According to computational offloading concept, (data transfer to cloud and program implementation in cloud). Now, we must study the subject when computation offloading is useful and when it is non-useful. Firstly, we must get energy consumption during program implementation in mobile devices (computational non-offloading or E_{nonoff}) and energy consumption during program implementation in cloud (computational offloading or E_{off}).

The amount of energy consumption during program implementation in mobile devices is equal to total energy consumption during computation in mobile device and energy consumption during results transfer from mobile device to cloud. (Results will be moved to cloud because they may be required for subsequent computations and implementation of future programs). Energy consumption during program implementation in cloud is equal to total energy consumption during transfer of program and input data from mobile to cloud and energy consumption during transfer of output data and results from cloud to mobile devices.

According to above-mentioned materials, if we present energy consumption during program implementation in mobile devices with E_{nonoff} and energy consumption during program implementation in cloud with E_{off} and consider variables affecting energy consumption as Table 2, then we will have:

$$E_{nonoff} = (\frac{D_{out}}{b'_{send}} * e_{send}) + (e_{comput} * I) \quad (6)$$

$$E_{off} = (\frac{D_{in}}{b_{send}} * e_{send}) + (\frac{D_{out}}{b_{receive}} * e_{receive}) \quad (7)$$

Table 2. Variables Used In Proposed Approach

Variable	Description
D_{in}	The number of input data
D_{out}	The number of output data
b_{send}	Network bandwidth when transmitting input data from mobile to cloud
$b_{receive}$	Network bandwidth when transmitting output data from cloud to mobile
b'_{send}	Network bandwidth when transmitting output data from mobile to cloud
e_{send}	Energy consumption during data transfer from mobile to cloud
$e_{receive}$	Energy consumption during data transfer from cloud to mobile
$e_{compute}$	Energy consumption during calculation of the mobile devices
I	Number of instructions
p_{cloud}	The possibility of performing program in mobile devices
p_{mobile}	The possibility of performing program in the cloud
E_{total}	Energy difference between cloud and mobile
a	Reward parameter in LA
b	Penalty parameter in LA
r	Number of action LA

If we obtain the difference between two values we will understand that if the program is implemented in cloud or mobile device, less energy will be consumed. So, we have:

$$E_{total} = E_{off} - E_{nonoff} \quad (8)$$

With respect to value obtained for E_{total}, two modes will occur; in first case E_{total}<0 and this means that energy consumption during implementation in the cloud is less than during implementation in mobile, then it is better to implement application in cloud. In the latter case E_{total}> 0 and this means that energy consumption during implementation in mobile is less than during implementation in cloud, so, it is better to implement application in mobile.

With regard to equations presented in previous section for calculatingE_{off}, E_{nonoff} there are two possibilities of implementing program in mobile devices and implementing program in cloud due to learning automata [11]. Then, r is equal to 2.

$$E_{nonoff} = \left(\frac{D_{out}}{b'_{send}} * e_{send}\right) + \left(e_{comput} * I\right) \quad (9)$$

$$E_{off} = \left(\frac{D_{in}}{b_{send}} * e_{send}\right) + \left(\frac{D_{out}}{b_{receive}} * e_{receive}\right) \quad (10)$$

$$E_{total} = E_{off} - E_{nonoff} \quad (11)$$

If obtained value for $E_{total}$is less than zero, it means that E_{off} is less than E_{nonoff}, then the possibility of implementation in cloud (p_{cloud}) will be rewarded and possibility of implementation in mobile (p_{mobile}) will be penalized. (According to equations 13 and 14) and if obtained value for $E_{total}$is more than zero, this means that E_{off} is more than E_{nonoff}, then possibility of implementation in cloud (p_{cloud}) will be penalized and possibility of implementation in mobile (p_{mobile}) will be rewarded. (According to equations 15 and 16) this process continues for n times and finally, p_{cloud} and p_{mobile} are compared with each other. If p_{cloud} is more than p_{mobile} the program will be implemented in cloud; otherwise, it will be implemented in mobile device, Also, Diagram of proposed approach is shown in Fig 2.

$$p_{cloud} = p_{cloud} + a * (1 - p_{cloud}) \quad (12)$$

$$p_{mobile} = (1 - a) * p_{mobile} \quad (13)$$

$$p_{cloud} = (1 - b) * p_{cloud} \quad (14)$$

$$p_{mobile} = \frac{b}{r-1} + (1 - b) * p_{mobile} \quad (15)$$

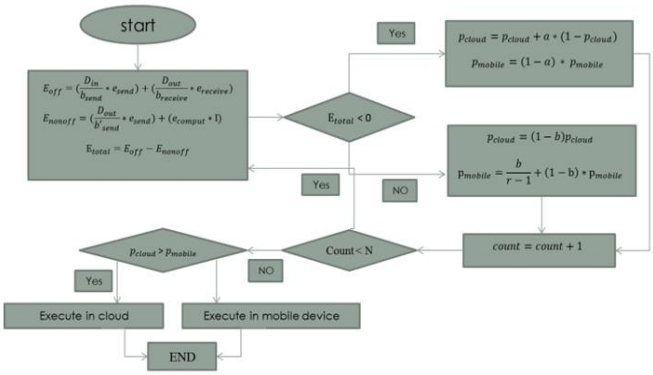

Figure 2. Diagram Of Proposed Algorithm

4. Performance Evaluation

The results of simulation for four mobile devices are shown in Table 3. Mobile devices are 1 to 4, in terms of speed, respectively. That is, mobile device 1 is the fastest and mobile device is the slowest. Hypothetical network bandwidth varies between 100 and 3500 Mbps. We have considered eight different statuses for network bandwidth changes based on Table 4. We have evaluated our proposed approach (based on learning automata) with simple approach (implementing application in mobile device and non-use of computational offloading) in the scenario of$e_{receive} = e_{send} > e_{comput}$.

Table 3. Profile of Mobile Devices

RAM	H.D.D	VGA	Core	CPU	Main Board	Mobile Device
DDR 3*4	SSD*2	NVIDIA GeForce6200	4	Xeon(3200 MHZ)	Quad socket platform	1
DDR 3*2	SCSI	NVIDIA GeForce6200	4	Celeron(1600 MHZ)	Quad socket platform	2
DDR 3	SSD	NVIDIA GeForce6200	1	Xeon X-5000(3000MHZ)	2x8pin E-ATX	3
DDR 3	SSD	NVIDIA GeForce6200	1	AMD(1400MHZ)	2x8pin E-ATX	4

Table 4. Different States Of Network Bandwidth Changes

b_{send}	$b_{receive}$	b'_{send}	Status
Increase	Increase	Increase	1
Increase	Increase	Decrease	2
Decrease	Increase	Increase	3
Increase	Decrease	Increase	4
Decrease	Decrease	Decrease	5
Increase	Decrease	Decrease	6
Decrease	Increase	Decrease	7
Decrease	Decrease	Increase	8

It is clear from the inequality of $e_{receive} = e_{send} > e_{comput}$ that amount of energy consumption during computations in mobile device is less than energy consumption during sending and receiving data to cloud. Figures 3, 4, 5 and 6 indicate that the more robust and faster mobile device is (in terms of computing power and CPU, RAM speed), number of situations in which energy consumption is equal in both simple and proposed approaches, will be more, as a result, the number of situations where it is better to implement applications in a mobile device, will be more. When the amount of energy consumption in simple and proposed approaches is equal means that computation cost of mobile devices is equal to transfer cost of applications in cloud and the obtained result is that increasing the number of instructions, number of conditions where these two approaches are equivalent in energy consumption is decreased; that is, increasing number of instructions will increase the amount of energy consumption in computing sector of mobile device Also for mobile devices 1 and 2 that are faster than mobile devices 3 and 4, when number of instructions is equal to one, in 5 states the energy consumption is equal in two approaches and in 3 and 4, when the number of instructions is equal to one, energy consumption is equal; then we conclude that for a mobile device of one that is faster, number of situations where it is better for different instruction to implement application in mobile devices, is more and for mobile device 4 which is slower, is less. Comparing Figures 3, 4, 5 and 6 it becomes clear that mobile device 1 has the least and mobile device 4 has the most level of energy consumption, then, faster mobile device will lead to the least energy consumption.

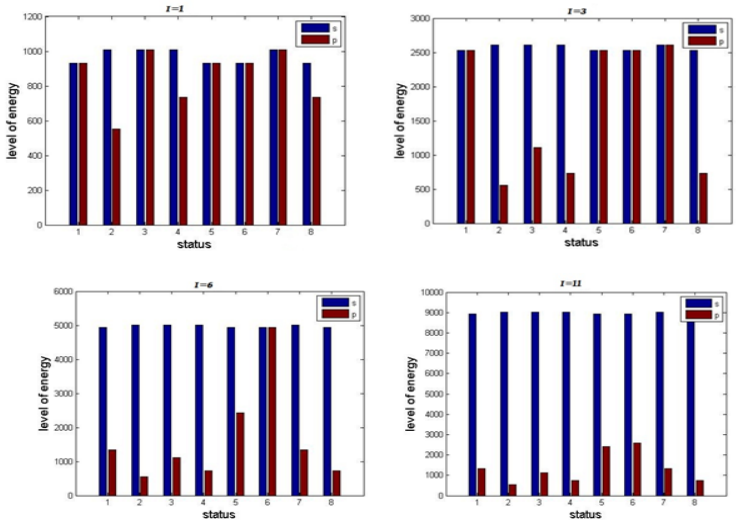

Figure 3. Comparing The Energy Consumption Per A Number Of Instructions For Mobile Device 1

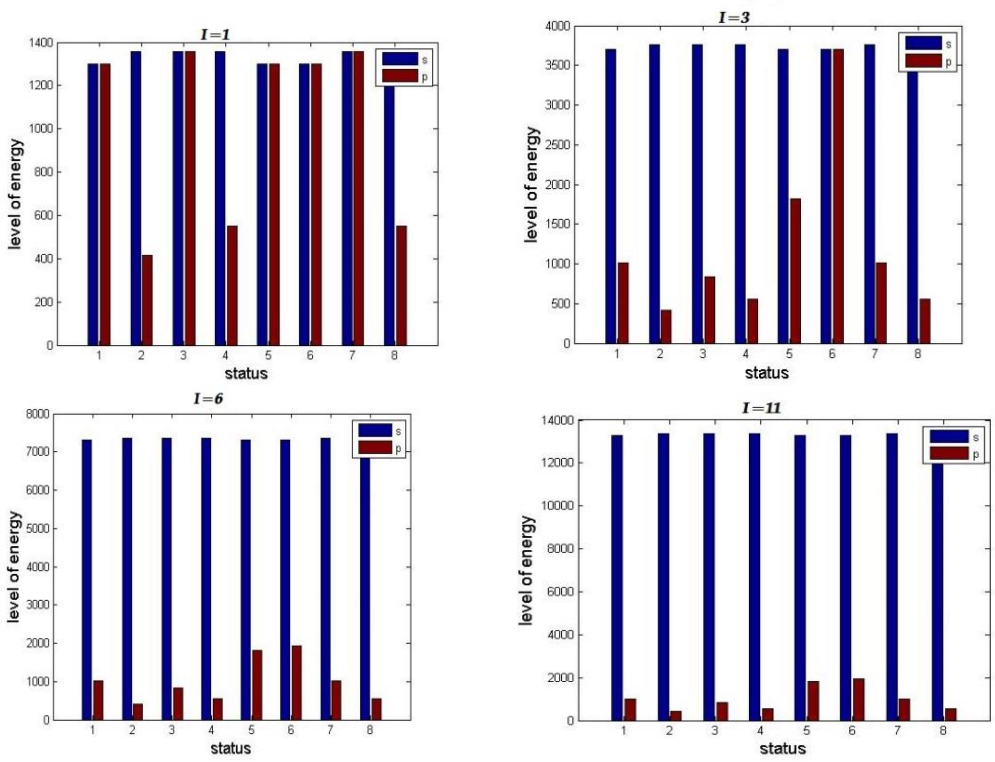

Figure 4. Comparing The Energy Consumption Per A Number Of Instructions For Mobile Device 2

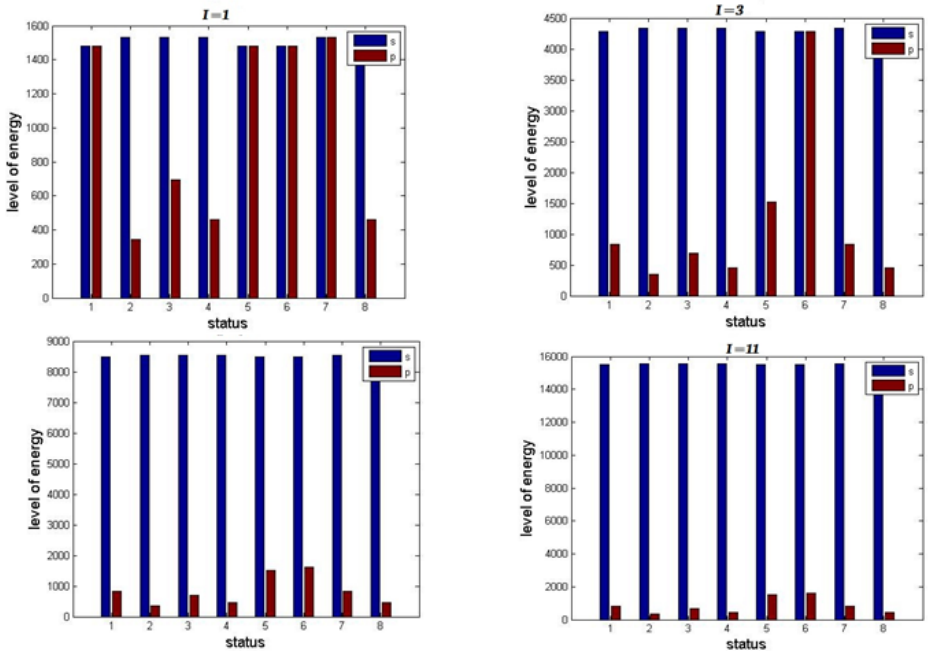

Figure 5. Comparing The Energy Consumption Per A Number Of Instructions For Mobile Device 3

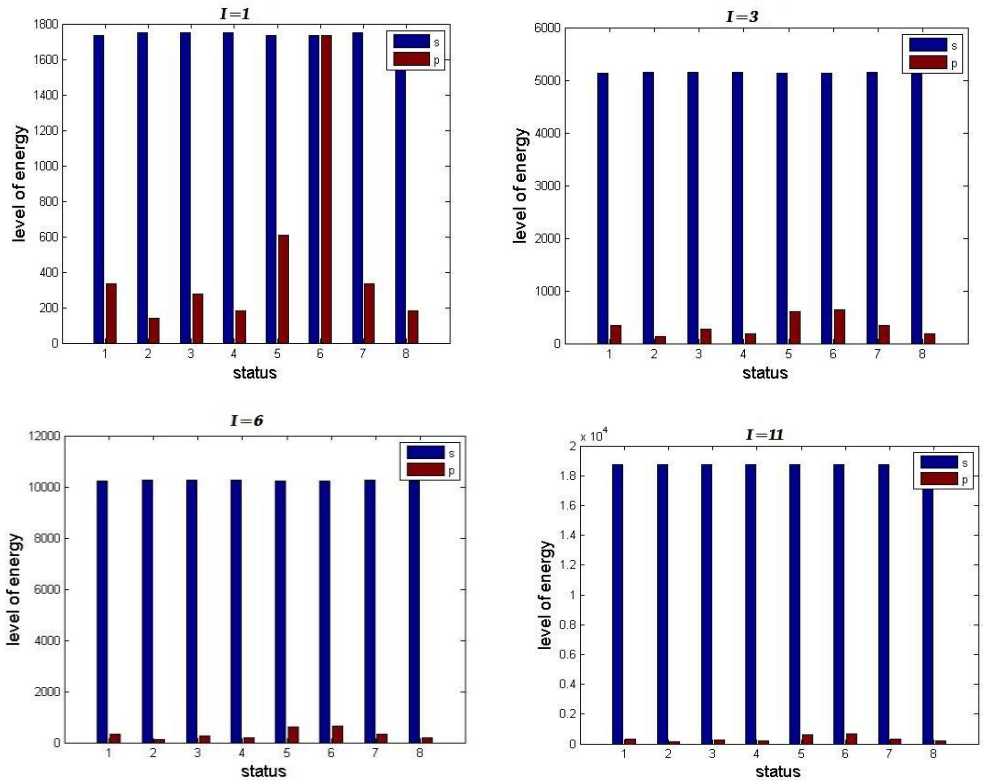

Figure 6. Comparing The Energy Consumption Per A Number Of Instructions For Mobile Device 4

5. Conclusion and Further Work

Given that one of the main challenges in mobile devices is their energy consumption, or in other words battery life, business firms have combined mobile devices with cloud and have created cloud computing. One of the solutions in order to reduce energy consumption and overcome this problem is to locate suitable place for application implementation. The application can be implemented on mobile devices or computational offloading (transfer applications from mobile devices to cloud and implementation in cloud). In this paper simple approach (implementing application on mobile device) is compared with LA based proposed approach. Our proposed approach is a flexible approach because it considers all states of broadband in every moment and considering changes in bandwidth and taking into account the values of e_{comput}, e_{send}, $e_{receive}$ and number of instructions. It determines that application to be implemented in cloud or on mobile devices. The general conclusion is that if mobile device is fast and network bandwidth is low or moderate and the number of instructions is low, it is better to implement application on a mobile device and if the number of instructions is more, it is better to implement application in cloud and if our mobile device is slow and network bandwidth is good, it is better to implement application in cloud.

The results of simulation show that in many cases, the performance of our proposed approach is better than simple approach performance and in a number of cases, both performances are equal. Then, using our proposed approach, better results will be obtained. In both cases, savings in energy consumption is tremendously observed. For future work, number of input and output can be considered variable and accordingly, energy consumption can be calculated. Moreover, the proposed approach does not consider energy consumption (battery) during off state of mobile device. This state must be considered for more accurate computation of energy consumption.

References

[1] I. Foster, Y. Zhao, I. Raicu and S. Lu, "Cloud computing and grid computing 360-degree compared", In Grid Computing Environments Workshop, 2008. GCE'08, **(2008)**, pp. 1-10.
[2] M. Fallah, M. G. Arani and M. Maeen, "NASLA: Novel Auto Scaling Approach based on Learning Automata for Web Application in Cloud Computing Environment", International Journal of Computer Applications, vol. 113, no. 2, **(2015)**, March, pp. 18-23.
[3] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey, " Future Generation Computer Systems, vol. 29, no. 1, pp. 84–106, Jan. 2013.
[4] P. Bahl, R. Y. Han, L. E. Li, M. Satyanarayanan, "Advancing the state of mobile cloud computing", In Proceedings of the third ACM workshop on Mobile cloud computing and services ACM, **(2012)** June pp. 21-28.
[5] N. Fernando, S. W. Loke and W. Rahayu, "Mobile cloud computing: A survey", Future Generation Computer Systems, vol. 29, no. 1, **(2013)**, pp. 84-106.
[6] H. T. Dinh, C. Lee, D. Niyato and Wang, "A survey of mobile cloud computing, architecture, applications, and approaches", Wireless communications and mobile computing, vol. 13, no. 18, **(2013)**, pp. 1587-1611.
[7] M. Fallah and M. G. Arani, "ASTAW, Auto-Scaling Threshold-based Approach for Web Application in Cloud Computing Environment", International Journal of u- and e- Service, Science and Technology (IJUNESST),vol. 8, no. 3, **(2015)**, pp. 221-230.
[8] A. Fereydooni, M. G. Arani and M. Shamsi, "EDLT, An Extended DLT to Enhance Load Balancing in Cloud Computing", International Journal of Computer Applications, vol. 108, no. 7, pp. 6-11, **(2014)** December.
[9] M. I. Alam, M. Pandey and S. S. Rautaray, "A Comprehensive Survey on Cloud Computing", IJITCS, vol. 7, no. 2, **(2015)**, pp. 68-79.
[10] Z. Li, C. Wang and R. Xu, "Computational offloading to save energy on handheld devices, a partition scheme", In Proceedings of the 2001 international conference on Compilers, architecture, and synthesis for embedded systems ACM, **(2001)** November, pp. 238-246.
[11] C. Lee, M. Potkonjak and W. H. Mangione-Smith, "Media Bench: a tool for evaluating and synthesizing multimedia and communications systems", In Proceedings of the 30th annual ACM/IEEE international symposium on Microarchitecture, IEEE Computer Society, **(1997)** December, pp. 330-335.
[12] H. J. La and S. D. Kim, "A conceptual framework for provisioning context-aware mobile cloud services", In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on IEEE, **(2010)** July, pp. 466-473.
[13] B. G. Chun and P. Maniatis, "Augmented Smartphone Applications through Clone Cloud Execution", In HotOS, vol. 9, **(2009)** May pp. 8-11.
[14] X. Ma, Y. Cui and I. Stojmenovic, "Energy efficiency on location based applications in mobile cloud computing, a survey", Procedia Computer Science, vol. 10, **(2012)**, pp. 577-584.
[15] "Global Positioning System, Standard Positioning System Service, Signal Specification", 2nd Edition, **(1995)**, pp. 18.
[16] D. Yao, C. Yu, H. Jin and J. Zhou, "Energy Efficient Task Scheduling in Mobile Cloud Computing", In Network and Parallel Computing Springer Berlin Heidelberg, **(2013)**, pp. 344-355.
[17] I. Giurgiu, O. Riva, D. Juric, I. Krivulev and G. Alonso, "Calling the cloud: enabling mobile phones as interfaces to cloud applications", In Middleware 2009, Springer Berlin Heidelberg, **(2009)**, pp. 83-102.
[18] J. S. Rellermeyer, O. Riva and G. Alonso, "AlfredO an architecture for flexible interaction with electronic devices", In Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware. Springer-Verlag New York, Inc, **(2008)** December, pp. 22-41.
[19] J. S. Rellermeyer, G. Alonso and T. Roscoe, "R-OSGi: distributed applications through software modularization", In Proceedings of the ACM/IFIP/USENIX 2007 International Conference on Middleware, Springer-Verlag New York, Inc., **(2007)** November, pp. 1-20.
[20] K. Narendra and M. A. L. Thathachar, "Learning Automata, an Introduction", Prentice Hall, Englewood Cliffs, New Jersey, **(1989)**.
[21] K. Najim and A. S. Poznyak, "Learning Automata, Theory and Application", Tarrytown, NY, Elsevier Science Ltd., **(1994)**.
[22] B. S. Taheri, M. G. Arani and M. Maeen, "ACCFLA, Access Control in Cloud Federation using Learning Automata", International Journal of Computer Applications, vol. 107, no. 6, **(2014)** December, pp. 30-40.

Authors

Najmeh Moghadasi, received the B.S.C degree in Software Engineering from University Shiraz, Iran in 2009, and M.S.C degree from Azad University of mahallat, Iran in 2014, respectively. Her research interests include Cloud Computing, mobile Cloud Computing.

Mostafa Ghobaei Arani, received the B.S.C degree in Software Engineering from IAU Kashan, Iran in 2009, and M.S.C degree from Azad University of Tehran, Iran in 2011, respectively. He's currently a PhD Candidate in Islamic Azad University, Science and Research Branch, Tehran, Iran. His research interests include Grid Computing, Cloud Computing, Pervasive Computing, Distributed Systems and Software Development.