World Applied Sciences Journal 33 (12): 1798-1805, 2015
ISSN 1818-4952

DOI: 10.5829/idosi.wasj.2015.33.12.310

Taxonomy of Computational Offloading in Mobile Devices

Mushtaq Ali, Jasni Mohamed Zain, Mohammad Fadli Zolkipli, Gran Badshah

Faculty of Computer Systems and Software Engineering,
University Malaysia Pahang, Lebuhraya Tun, Razak, 23600 Kuantan Pahang, Malaysia

Abstract: The benefits of mobility, convenience and convergence powered mobile devices the first choice for computation, communication and entertaining. However, these devices lack in resources for computation, networking, memory and battery. An effective approach to alleviate the constraints of mobile devices is to offload resource-intensive/ compute intensive components of mobile applications to resource abundant/resourceful servers for processing; called Cyber Foraging / Computational Offloading.We explain theaugmentation approaches of mobile's resources, cyber foraging system's matrices and the taxonomy of the existing cyber foraging approaches considering their type, granularity, surrogate types, parameters of decision,location of the decision maker units, remoteness of execution, migration support and their overheads. At the end, issues are discussed on the basis of comparative studies.

Key words: Cyber Foraging · Mobile Computing · Surrogates · Mobile Devices

INTRODUCTION

The dependency of new generation on mobile devices is increasing each coming day. In the whole planet of around 6.8 billion people, 4.6 billion people reached to cell phone subscription at the end of 2009 and is likely reach to five billion by the end of 2010 [1]. People are tied up in doing tasks all around them through their mobile devices, such as Internet banking, using routing applications, surfing internet and communications. Moreover the mobile devices are blossomed with new features in terms of graphics, speed, memory and sensing which are interesting for users. Also the new mobile applications make them attractive and acceptable for each age group.

On the other hand, the local resources of mobile devices are restricted by the limitation of size and weight which are the primary market demands. Mahadev Satyanarayanan in 1993 stated in a key note that "Mobile elements are resource-poor relative to static elements. Regardless of future technological advances, a mobile unit's weight, power, size and ergonomics will always render it less computationally capable than its static counterpart. While mobile elements will undoubtedly improve in absolute ability, they will always be at a relative disadvantage" [2]. By the Report of NRC (National Researched Council) held in 1997 on low resources issues of mobile devices stated that [3] "Using new materials and chemistries, batteries are approaching explosives in terms of energy density". Mobile batteries are in such case the stick of dynamites which are in our pockets. The fortunate thing is the energy released by the batteries is in intervals and over many hours rather than few microseconds. But an effect of the quote is, the basic improvement of battery technology is very low and the density of batteries is already very high. The additional improvement of batteries is probable but that improvement is drained by the energy demand of new mobile's applications. It is therefore essential to seek some alternative solutions of combating low resources (battery) issues.

To overcome the issues related to the resource scarceness of mobile devices, parallel research working on to reduce the local computation at mobile devices and that will leads us to some successful results [4]. Computational offloading is one of the practical technique which can be used to reduce the burden over handheld devices and effectively leverage the resources to work speedy and smoothly. Computational Offloading defined to offload the complete or a portion of a mobile app to a resourceful

7th May, 2015. This work is sponsored and supported by the University Malaysia Pahang.
Corresponding Author: Mushtaq Ali, Faculty of Computer Systems and Software Engineering University Malaysia Pahang, Lebuhraya Tun, Razak, 23600 Kuantan Pahang, Malaysia.

servers for processing [5]. The net advantage of computational offloading is the proliferation of runtime efficiency and to curtail the consumption of local resources on the mobile device.

In this paper, we emphasize on the cyber foraging technique. Section II presents an overall view of different approaches for augmenting the resources shortage of mobile devices. Section III describes Cyber Foraging by calculation an energy saving technique. Section IV defines the most effective metrics in cyber foraging techniques. Section V presents the taxonomy of cyber foraging techniques. Section VI discusses issues and opinions and Section VII concludes the paper.

Approaches for Augmenting Mobile's Resources: Generally there are four basics approaches used to augment the mobiles resources [4].

- Generate New Hardware Resources
- Execute Program Slowly
- New Application for Resource-Constrained Devices
- Hardware & Software Management Techniques

Generate New Hardware Resources

- Need of a new generation semiconductor technology [6].

The current semi-conductor technology made the transistor smaller, in other words they consume less power, yet due to the smaller size of transistor lots of transistors need for achieving functionalities and better performances. Thus, increasing the number of transistor actually burdenon power source of mobile device and consumes more energy.

Replenish Resources (Battery) by External Action: Human movements, solar light (Nano-Technology) are some of the possible ways which could be used to enhance mobile's battery.

Execute Program Slowly: With the increase of CPU speeds the consumption of battery significantly increasing i.e. if the processor Clock Speed doubles, the power consumption speed nearly octuples.

New Application for Resource-Constrained Devices: One such approach is to rewrite new applications for resource-constrained mobile devices. This is looking somehow impractical, costly and can push towards ad-hoc applications [7].

Hardware & Software Management Techniques

Avoid Wasting Energy [8, 9]: Wasting of energy can be reduced either by avoiding unnecessary processing, better management of resources and to put the components in standby or sleep mode.

Reduce Resources Requirement: To develop the context-aware mobile applications, that could make the device sensible enough when/what to process and when/what not to process to reduce the unnecessary need of mobile resources.

Fidelity Adaptation: Fidelity adaptation manages the compromise between resources consumption and application quality. Althoughfidelity adaptation technique drops the quality of results, it allows the execution of applications when there is no other solution to run the application in standard mode.

Cyber Foraging / Computational Offloading: Here the execution of programs is eliminated altogether and send the heavy computations to remote servers. If the mobile devices utilize its own local resources solely to perform execution of complex applications it will drain battery faster and in some cases the execution will not be possible for some kind of applications due to limited processing speed. By computational offloading the load is eliminated to augment the device's own resources.

Any of the above approach can be adopted to reduce power consumption of handheld devices. Numerous researchers tried and achieved success up to some extent to save mobile's battery. Our review focuses on the last option i.e. to utilize powerful resource of cloud (Cloutdlet) computing instead of local limited resources of mobile device. This way the burden not solely carries by the mobile device. Rudenko [10] first time introduced the term remote execution which is different from the traditional Client-Server architecture, where a thin client always migrates computational task to a server in the same local network while in remote execution the offloading process accomplished with the devices which are outside of the immediate computing environment. Satyanarayanan [10] presented the concept of remote execution by accessing nearby available machines (surrogates) to execute complicated computation on behalf of handheld device. He termed such type of computing "Cyber Foraging" or Surrogate Computing". This way they eliminated execution of the entire complex task locally. Fig. 1 shows the possible ways of augmenting mobile's resources.

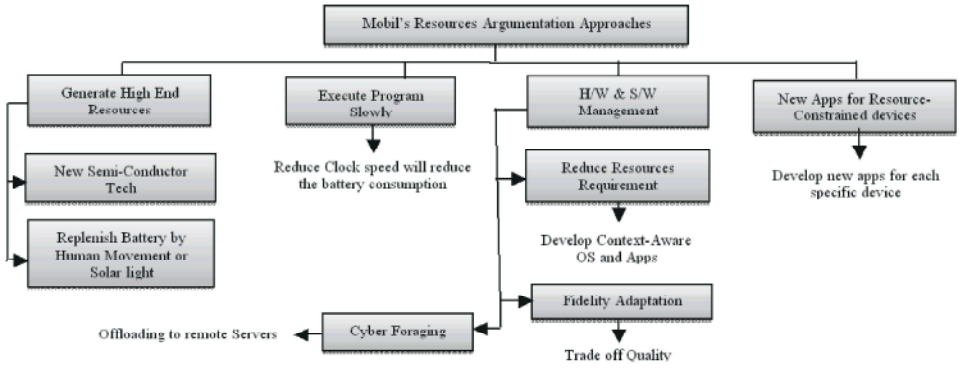


Fig. 1: Mobile's Resources Augmentation Approaches

Is Computational Offloading Save Energy?: The cloud computing is different from the existing models due to a fundamental feature "virtualization". [11] Virtualization allows cloud vendors to provide services of running arbitrary applications of various customers on virtual machines. Accordingly, a cloud vendor provides computingcycles to reduce computation of mobile devices. Computational offloading approach facing two challenges listed below.

What will be the optimum condition for computational offloading from client device to cloud?

What can be the factors that are needed to address before starting computational offloading?

Now the question arises, Is Computation Offloading save

Trade off Quality: Energy? Karthik Kumar[12] has addressed these issue by analyzing the energy consumption using computational offloading. They proposed a formula (1) and by using this formula they derived the amount of energy which can be saved during offloading process.

$$PC \times \frac{C}{M} - Pi \times \frac{C}{S} - Ptr \times \frac{D}{B} \quad (1)$$

where,

C - *Is the number of instruction to be offloaded,*
S&M - *Are the speeds instruction /second of Server and mobile device respectively.*
PC - *Mobile power consumption (watts),*
Pi - *Mobile idle power consumption (watts)*
Ptr - *Power Consumption of Mobile's During Transmission.*
D - *Data in bytes to be exchanged.*
B - *Network bandwidth.*

If, the server speed considered F times faster than mobile speed then

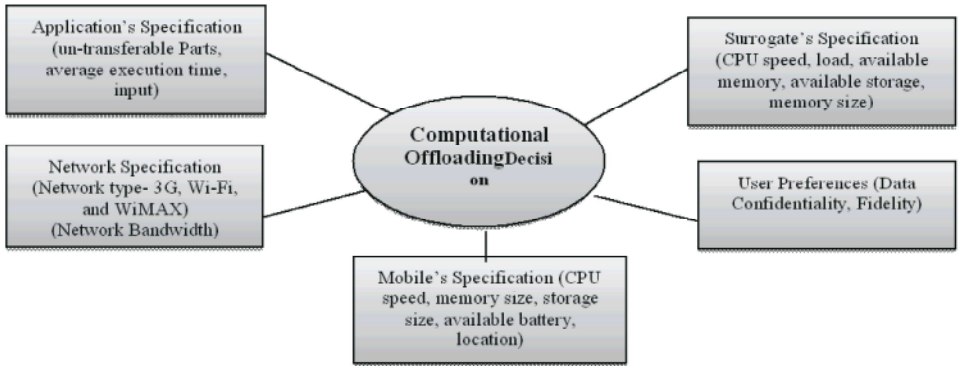
$$S = F \times M \quad (2)$$

And by substituting eq. (2) in (1), the formula can be rewrite as,

$$\frac{C}{M} \times (PC - \frac{Pi}{F}) - Ptr \times \frac{D}{B} \quad (3)$$

Here, the values M, Pi, Pc and Ptr are constant and if eq. (3) provides a positive number then offloading will reduce power consumption of mobile device. The formula will provide positive number if is sufficiently small (i.e.B sufficiently large) and F is sufficiently large. In other words, if bandwidth and server speed are sufficiently large then offloading will reduce power consumption. The relationship between B, D and C is important to predict whether to offload task or not i.e. for large Computation C if communication data D is smaller and bandwidth B is enough large then offloading will be beneficial otherwise for small C and low bandwidth B, it is useful to avoid offloading and process data locally. The relationship of B, D and C can be seen in the figure below.

Metrics of Cyber Foraging: Few matrices could be taking into account which could possibly influence the process of computational offloading. Context Specification, Mobile & Surrogate Specification, Network Specification & Application Specification are few of those matrices. Fig. 2. describes the computational offloading decision influenced by these matrices.



Fi.g. 2: Matrices of Cyber Foraging

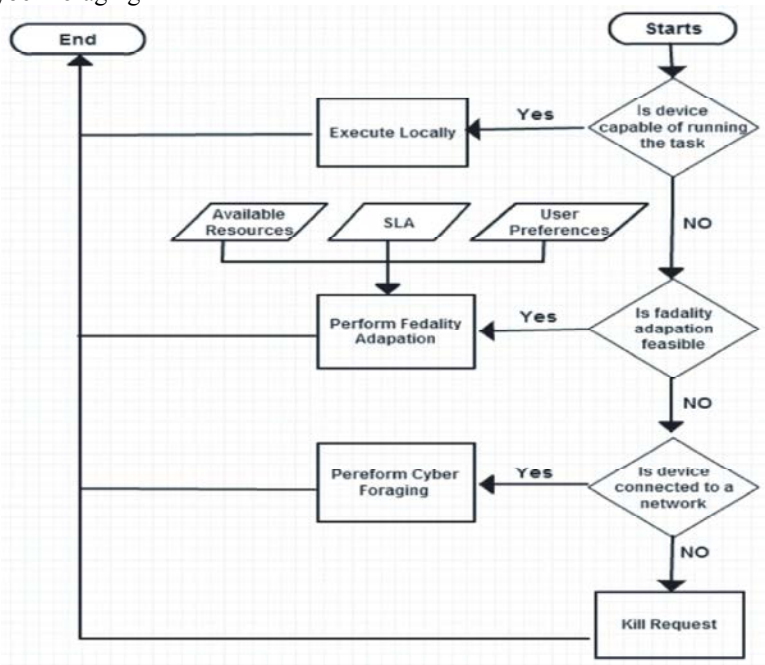


Fig. 3: Execution flow of Mobile's apps

If there are no surrogates available in the surrounding to offload the complex computation then Fidelity adaption is the process of trade-off between quality and speed/power consumption. User can manually specify the low quality for better speed and to reduce battery drainage.

Wireless networks have different type of features and bandwidths. Mobile device can connect and communicate through any kind of available network such as Wifi- 3G, WiMIAX. Therefore computational offloading decision is strongly influenced by the different bandwidth of different networks.

Mobile and surrogate devices have different type of CPUs, speed, available memory, storage capacity. If the mobile devices have not enough speed or storage memory, then cyber foraging/ offloading is the best option to utilize.

Application type is a key metric which need to be checked before offloading. If the application is processor intensive / complex to compute locally, then offloading will be more useful.

On the basis of few other elementary metrics i.e. user QoS requirement, availability of local resources, SLA and network availability S. Abolfazli [13] described a sample flow of mobile application execution as shown in Fig. 3. The flow chart depicts four process of mobile application to be executed. If the device is capable of running the task it will be executed locally otherwise will be offloaded to remote servers. If all the available option go false then application execution request will be killed. Fig-7 below illustrates the flow chart of mobile application execution.

The issues related to offloading are the efficient and dynamic offloading [14] under changing environment. I.e. user moving will affect the bandwidth, then what strategy

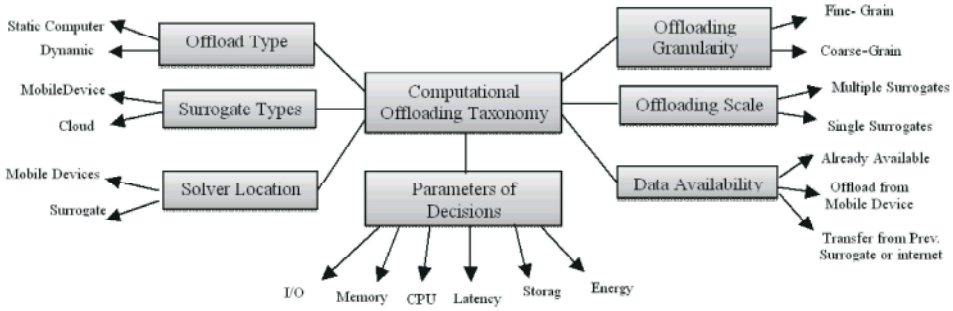


Fig. 4: Taxonomy of Cyber Foraging Approaches

should adopt to offload applications? In case of static offloading the application will be offloaded before runtime to server regardless of environmental changes and user context. By Rudenko [15] static offloading is not always energy efficient approach i.e., if the size of compiling code is small enough then offloading will consume more energy than that of energy consumed in local processing. For instance if the size of compilation codes is 500KB, then offloading use about 5% of mobile's battery for its offloading to server whilelocal processing for the same size of code consumes approx. 10% of the battery for computation. In this case, offloading can save a significant amount of energy (50%). Conversely, if the size of codes is 250KB, then the efficiency reduces up to 30%. Thus, if the size of codes to be executed is small, the offloading will consume more battery than that of local execution.

It is seriously tricky for mobile devices to decide whether to offload or not and which portions of the application's codes needed to offload for improving energy efficiency. Moreover, diverse wireless access technologies require different amount of energy and also support dissimilar data transfer rates. These factors are needed to be taken into account before offload.

To overcome these issues the dynamic offloading techniques used. It will decide at runtime whether to offload and which portion of the application to offload based on energy consumption as suggested by [11]. The optimal partitioning of program takes place on the basis of trade-off between computation and communication costs. Several solutions have been proposed for the optimal application partitioning. By [16] "If a device becomes resource constrained at run time and accept that it can beneficially use nearby resources, it automatically and transparently offloads part of the service to them". They proposed a dynamic shared distributed environment where in case of remote server inaccessibility can share the application portions to another surrogate server (s).

Taxonomy of Cyber Foraging/ Computational Offloading Approaches: In this section based on the available information of the existing cyber foraging/ computational offloading systems we present a proposed cyber foraging taxonomy. The most significant repetitive features such as offload type, surrogate type, offloading scale, solver locations, code availability, offloading granularity, data availability and parameter of decision of computational offloading / cyber foraging systems are used to classify and propose taxonomy. Fig. 4 illustrates cyber foraging taxonomy and is briefly discussed in the following subsections.

Offload Types: Offloading can occur either at start time referred to as static offloading or at runtime called dynamic offloading [17]. During static offloading, a middleware or programmers partition the program before execution. Thus, at runtime, the system identifies which portions of the program should be offloaded. However, due to the expanded uniformity of network environments and surrogates, static offloading cannot ensure the best partitioning for all probable situations which could be beneficial. Chroma and Spectra are the examples of most important works did partitioning before program execution. In contrast, dynamic offloading starts to offload tasks when the required resources for offloading is insufficient and partitions the program according to the availability of at runtime. This approach decides offloading based on existing conditions and is therefore beneficial and more flexible. It however causes more overheads on the system. X. Gu [18] used the dynamic offloading techniques.

Surrogate Types: Cyber foraging can further categorize by the surrogate type. They use either static computers or mobile devices. Generally, most of the cyber foraging approaches use static computers as surrogates [19, 20]. Some works that use mobile surrogates are [21]. Even though powerful stationary computers/surrogates

are suitable for offloading, but in case of no surrogates available or in circumstances such as changing in network topology, user preferences etc. may direct a cyber-foraging system to pick a mobile surrogate for offloading instead.

Offloading Granularity: If the surrogate device does not have the required application then there is need to offload some of the related parts of the application from mobile device to the surrogate. The process of offloading parts or whole application is called offloading granularity. In cyber-foraging approach if some parts of the applications are offload is called fine-grain. The previous works [22] used fine-grain method. Some of the works i.e. [17, 19] offloaded whole program called coarse grain. In fine-grain strategy only the parts which are needed can be offloaded and it leads to enough energy savings. This strategy is suitable for highly mobile environments,because mobile devices move in the environment and probability of network disconnection increases due to load and un-availability of wireless signals.

Offloading Scale: The selection of surrogate devices to offload CPU intensive parts off applications is called offloading scale. Offloading scale differs in different cyber foraging / offloading approaches. The cyber foraging system either select a single surrogate from the pool of available surrogates to offloada task and then get the result back [22], or in some other cases [23], used multiple surrogates to offload. Offloading scale using multiple surrogates to offload is beneficial. The reason is to deal with the mobility nature of mobile devices.Moreover, fault tolerance can also be increased [24] by parallel offloading to multiple surrogatesand it will alsofacilitate the latency control.

Data Availability: To perform an execution of a task, some of the related information, like input data needed to be available in the execution environment. The assumptions and tactics about data availability can be defined in three cases. First case is the one where data is already available on the surrogate and no need to transfer anything from mobile to surrogate [25]. Let suppose two tasks A and B are running in mobile device and theA's output is the requiredB's input. Now if a surrogatehas executed task A, then it has the B's input and it will not need data migration from mobile. In the second case, information is missing with the surrogate and need to transferred from a mobile device to surrogate [26]. In the third case, necessary information fetched from anothersurrogate [20]. This strategy can work more efficient if fetch the required data from the Internet. Also the forecasting and context aware information can help to improve this case to provide prior information before execution such as, user's location, bandwidth / internet availability and to foresee the next available in advance to transfer necessary information before starting to run the next task.

Parameter of Decision: The main goal of cyber foraging/computational offloading is to cope with the resource constraints of mobile devices. Therefore, few of the matrices must keep in mind to consider the cyber foraging a solution for augment mobile's local resources. The most essential factors that could be consider foroffloading are as follows:

Energy Consumption: One of the key constraints of mobile devices is the limited power storage. As the mobile device's energy cannot be replenished by itself that's why many researches [22] considered energy consumption / battery power as a parameter for taking an offloading decision.

Memory and Storage: The applications which are memory intensive usually cannot be run on mobile devices and they are needed to offload. So, many of the researches considered the local memory and storage of mobile device as an effective parameter before offloading [27].

Responsiveness: The execution time of a compute intensive app decreased by offloading if the processing power / CPU Speed of mobile deviceis significantly lower than the static computers. Many of the researchers considered the response/execution time and latency as main parameters which couldaffect the offloading decision.

I/O: Sometimes, I/O devices are considered for theimprovement of quality e.g. when we need to displaying a movie on a larger screen, use bigger speaker for playing musicand using distant printing. Some of the previous works focused on augmenting the I/O as an effective parameter on offloading decision.

Solver Location: The unit which is responsible for taking offloading decision is called a solver. This parameter is considered by many researchers as a location of solver. Normally, every mobile device has its own solver and can play the role of decision makeritself, however, in some works [28] the solver is not located in the mobile device. For example, MAUI generates a call graph of application to execute.

DISCUSSION

Cyber foraging is a worthy solution to augment the resources limitations of mobile devices, but this approach has its own limitations too. Firstly, to accomplish the process of offloading some surrogates should be accessible and willing to share their own resources with others (PDAs, Mobile Devices) via wireless networks. Secondly, through cyber foraging we can't be guaranteed for security of confidential data. Thirdly, cyber foraging is applicable only to the tasks which are transferable while there are some tasks which are not transferable. In addition, offloading of small tasks may not be beneficial due to extra communication overhead or changing of network topology may affect the offloading process too. Fourthly, computation offloading with multiple surrogates and users may cause the issue of load balancing. In addition to improve the efficiency of cyber foraging / remote execution only the required data needed to offload to reduce communication overhead. Fifthly, the dynamic allocation of resources on demand generates the issues of synchronization, resuming/releasing etc., which ultimately causes latency.

A good understanding of all the related issues is crucial to keep in mind before making the cyber foraging practical and beneficial.

CONCLUSION

Mobile devices turned the crucial element of our daily life due to the mobility, convenience and convergence; however these devices are lack in resources for computing, memory and power which makes the devices unreliable in particular circumstances. To effectively combat the resource scarceness of these devices cyber foraging / computational offloading is introduced, where the compute intensive parts of mobile apps are transferred to resources full/ powerful servers located either in the cloud or in a single hop proximity. In this paper we explained the augmentation approaches of mobile's resources, cyber foraging system's matrices and the taxonomy of the existing cyber foraging approaches considering their type, granularity, surrogate types, parameters of decision, location of the decision maker units, remoteness of execution, migration support and their overheads.

REFERENCES

1. Sharifi, M., S. Kafaie and O. Kashefi, 2012. A survey and taxonomy of cyber foraging of mobile devices, IEEE Commun. Surv. Tutorials, 14(4): 1232-1243.
2. Satyanarayanan, M., 2014. A Brief History of Cloud Offload, GetMobile, 18(4): 19-23.
3. Flynn, P., 2004. Meeting the Energy Needs of Future Warriors Committee, National Research Council (US) Committee of Soldier Power/Energy Systems.
4. Ali, M., J.M. Zain, M.F. Zolkipli and G. Badshah, 2014. Mobile Cloud Computing & Mobile Battery Augmentation Techniques?: A Survey, IEEE SCOReD 2014. pp: 1-6.
5. Fernando, N., S.W. Loke and W. Rahayu, 2013. Mobile cloud computing: A survey, Futur. Gener. Comput. Syst., 29(1): 84-106.
6. "New semiconductor research may extend integrated circuit battery life tenfold," *SEMATECH*. [Online]. Available: http://phys.org/news/2013-01-semiconductor-circuit-battery-life-tenfold.html.
7. Batteries, A.D., 2013. From the Editor-in-Chief., Nurs. Leadersh. (Tor. Ont)., 26(1): 1-2.
8. Vallina-rodriguez, N. and J. Crowcroft, 2013. Energy Management Techniques Modern Mobile Handsets, IEEE Communication Surveys & Tutorials pp: 1-20, Vol-15 no.-1 First Qaurter 2013.
9. Balasubramanian, N., 2009. Energy Consumption in Mobile Phones?: A Measurement Study and Implications for Network Applications, IMC09, Nov 4-6 Chicago USA, 2009.
10. Satyanarayanan, M., 2001. Pervasive computing: vision and challenges, IEEE Pers. Commun., 8(4): 10-17.
11. Kumar, K., 2010. Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?, Computer (Long. Beach. Calif)., 43(4): 51-56.
12. Kumar, K., J. Liu, Y.H. Lu and B. Bhargava, 2012. A Survey of Computation Offloading for Mobile Systems, Mob. Networks Appl., 18(1): 129-140.
13. Abolfazli, S., Z. Sanaei and A. Gani, 2012. Mobile Cloud Computing?: A Review on Smartphone Augmentation. 1st International Conference on Computing, Information Systems and Communications (CISCO'12) Singapore, May 2012.
14. Dinh, H.T., C. Lee, D. Niyato and P. Wang, 2013. A Survey of Mobile Cloud Computing?: Architecture, Applications and Approaches, Wireless Commnucations and Mobile Computing, no. Cc, pp: 1-38. 25th Dec 2013.
15. Rudenko, A., P. Reiher, G.J. Popek and G.H. Kuenning, 1998. Saving portable computer battery power through remote process execution, ACM SIGMOBILE Mob. Comput. Commun. Rev., 2(1): 19-26.

16. Messer, A., I. Greenberg, P. Bernadat, D. Milojicic, D. Chen, T.J. Giuli and X. Gu, 2002. Towards a Distributed Platform for Resource-Constrained Devices, Proceedings. 22nd International Conference on Distributed Computing Systems, pp: 43-51.
17. Murarasu, A.F. and T. Magedanz, 2009. Mobile middleware solution for automatic reconfiguration of applications, ITNG 2009 - 6th Int. Conf. Inf. Technol. New Gener., pp: 1049-1055.
18. Gu, X., K. Nahrstedt, A. Messer, I. Greenberg and D. Milojicic, 2004. Adaptive offloading for pervasive computing, IEEE Pervasive Comput., 3(3): 66-73.
19. Terena, J.D., 2009. The Case for, no. January, 2009.
20. Su, Y.Y. and J. Flinn, 2005. Slingshot: Deploying Stateful Services in Wireless Hotspots, Proc. 3rd Int. Conf. Mob. Syst. Appl. Serv. - MobiSys, 05: 79-92.
21. Begum, Y.M. and M.AM. Mohamed, 2010. A DHT-based process migration policy for mobile clusters, ITNG2010 - 7th Int. Conf. Inf. Technol. New Gener, pp: 934-938.
22. Flinn, J. and M. Satyanarayanan, 2002. Balancing performance, energy and quality in pervasive computing, Proc. 22nd Int. Conf. Distrib. Comput. Syst., pp: 217-226.
23. Kristensen, M.D., 2008. Execution plans for cyber foraging, Proc. 1st Work. Mob. Middlew. Embrac. Pers. Commun. device MobMid 08: 1.
24. Zhang, X., S. Jeong, A. Kunjithapatham Gibbs and Simon, 2010. Towards Capabilities, an Elastic Application Model for Augmenting Computing of Mobile Platforms, Third Int. ICST Conf. Mob. Wirel. MiddleWARE, Oper. Syst. Appl., 2010.
25. Kristensen, M.D., 2010. Scavenger: Transparent development of efficient cyber foraging applications, Pervasive Comput. Commun. (PerCom), 2010 IEEE Int. Conf., pp: 217-226.
26. Francisco, S., 2003. Proceedings of MobiSys 2003: The First International Conference on Mobile Systems, Applications and Services, Networks, 2003.
27. Ou, S.O.S., K.Y.K. Yang and Q.Z.Q. Zhang, 2006. An efficient runtime offloading approach for pervasive services, IEEE Wirel. Commun. Netw. Conf. 2006. WCNC 2006., 4(c): 2229-2234.
28. Cuervo, E. and A. Balasubramanian, 2010. MAUI: making *smartphones* last longer with code offload, MobiSys'10, June 15-18, 2010, San Francisco, California, USA, 17: 49-62.