Int J Parallel Prog (2020) 48:399–415
https://doi.org/10.1007/s10766-018-0563-0

DyTO: Dynamic Task Offloading Strategy for Mobile Cloud Computing Using Surrogate Object Model

A. N. Gnana Jeevan[1] · M. A. Maluk Mohamed[1]

Received: 27 December 2017 / Accepted: 1 March 2018 / Published online: 13 March 2018

Abstract In Distributed Mobile Computing paradigm, mobile devices are offer unlimited services over mobile cloud computing. However, the limitation of the mobile devices and mobile cloud lead to several issues and challenges such as inadequate bandwidth, time delay, limited energy, poor quality of services and frequent disconnected network. To deal with these obstacles, we propose strategy dynamic task optimization in mobile cloud with offloading algorithm. In this paper the offloading algorithm we designed a model for coordinated management and optimized utilization of resource in mobile cloud for offloading process. The experimental evaluation are performed through the simulation results proves the feasibility and reliability, which shows its better mobile transactions with effective utilization of energy, resources, bandwidth, reducing the overhead time and improved quality of services for disconnected network between mobile devices and cloud servers.

Keywords Mobile cloud computing · Surrogate · Cloud · Dynamic task scheduling · Offloading

✉ A. N. Gnana Jeevan
ssg_jeevan@mamce.org

M. A. Maluk Mohamed
ssg_malukmd@mamce.org

1 Software System Group Lab, Department of Computer Science and Engineering, M.A.M. College of Engineering (Affiliated to Anna University, Chennai), Tiruchirappalli, Tamil Nadu, India

1 Introduction

The challenges in Distributed Mobile System raised by Mobile Cloud Computing and put the light on new challenging problems, the management of dynamic and scaling aspects in application offloading, the application are choose the storage location to host their data at global locations with cost effectively and time of task scheduling to minimize total execution time, minimize total cost to execute offload data meets the constraints of the application and the management of time, energy, network bandwidth and other components are effectively and efficiently schedule the application need to utilize the resource in mobile cloud environment. To improve the resource scarceness of mobile host, here the task of offloading computing is introduced. Few efforts are made in offloading to overcome the dependency on a single provider through cloud task. Task strategy in simple terms can be defined as two service providers sharing some data. The task could be user's identity, resources, files, resource information or any other information. We achieve all these challenges by integration dynamic offloading to minimize the data transaction cost and can save more energy by executing the application with minimal time to improve the average response time and average processing time of the system, maximum process utilization with assign task it to appropriate resource to produce better process management to provide Quality of Service, effectively and efficiently with Surrogate object model between mobile devices and cloud environment while the network communication have low bandwidth, mobile device had limited energy and frequent disconnect of network in wireless transaction.

In this paper we propose dynamic task offloading in a network of connected mobile host with surrogate object, which we call a SOMC (surrogate object in mobile cloud). We define the mobile host as the *hostoffloader* which is offloading computation, and a mobile host which is carrying out computation on behalf of another host as the *hostoffloadee*. We define a task as a combination of the data and resources that it takes in as input, and the computation of that task needs to perform in the cloud resource on this data in order to yield a result. An application is comprised with lot of such tasks, and the more data are require more energy to perform computation task wired or wireless network. Figure 1 shows three options for computational offloading.

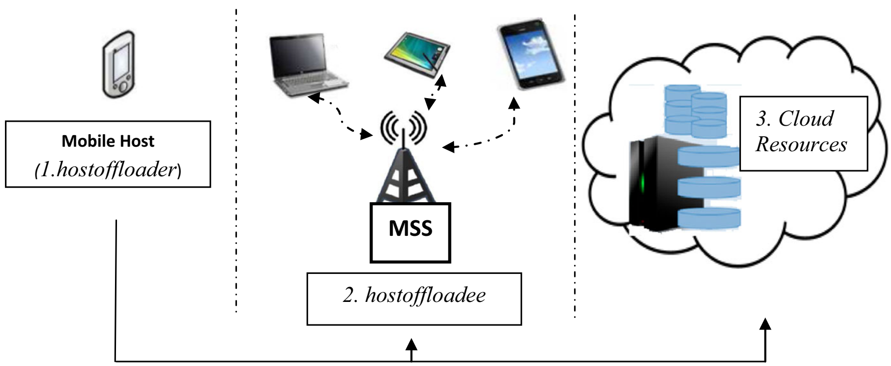


Fig. 1 Strategy for offloading computation

Specifically, a host can offload to a cloud which is far away, to a Mobile Support Station (MSS) which is a smaller set of servers located inside a building, nearby shops, etc., Each of these different choices offers different tradeoffs that need to be kept in mind when making the decision of which platform to offload to cloud resource management. Current work shows that the surrogate object based mobile cloud is a mechanism where resource-intensive computations are migrated from mobile host to the highly rich cloud server or nearby infrastructure. This work highlights the potential gain in energy and execution time which can be achieved by offloading computation among host in a SOMC. We study, if it is possible to gain energy and time by offloading computation by employing offloading in SOMC. We perform a set of experiments that emulate environments with the wired and wireless network characteristics as those provided by these communication techniques so as to be able to determine whether it makes sense to (1) offload computation at all and (2) if so, for what combinations of data and computation should these tasks be dynamic offloaded so as to conserve execution time and energy. Using the emulation test, we show possible gain in both execution time and energy, up to 42% and 21% respectively, which can be achieved by offloading to other mobile host in an SOMC. Through all these considerations our goal of work is to design an efficient dynamic task offloading using Surrogate Object strategy for heterogeneity mobile cloud. The objective of this paper is to derive “task offloading decisions” under on demand priority based task of the mobile device and to provide better performance of cloud resources, less energy consumption, time latency, and financial cost. The main contributions reported in this paper are as follows:

1. First, we propose a system framework model for dynamic task offloading, for energy consumption, latency, federated, time and cost.
2. Dynamic task is “offloading” based on priority task scheduling under the wireless communication.
3. Design a dynamic task offloading algorithm which includes a distributed task scheduling algorithm, based on this algorithm; the mobile devices can make a proper decision about task scheduling. Then, the dynamic offloading of this algorithm is discussed.
4. At last, simulations are conducted to evaluate the proposed strategy.

The rest of this paper is organized as follows:

First, in Sect. 2 the related works are discussed. Then, in Sect. 3 a Dynamic task offloading for the mobile cloud using Surrogate Object. In Sect. 4, we analysis the performance of offloading problem with experimental results. At last, simulations are conducted to perform the proposed strategy. Finally, Sect. 5 concludes this paper.

2 Related Work

In Mobile Cloud environment the user can access the cloud services anytime and anywhere. The services are usually provided by the service providers. In task offloading the providers learn to share their resources to other service providers. If the services are available or not available in the service provider end, (i.e) the service provider do not have enough limited resources to meet the client requirement then it should start denying the acceptance of new client or cancel the lower priority services which are already running on the system. This problem can be overcome via dynamic task offloading in surrogate object model.

2.1 Surrogate Object Model

In a mobile environment, the surrogate object defines an architecture that allows mobile devices to participate seamlessly in computing and communication. This model brings the mobile devices and its supportive environment together with the help of the surrogate object. Moreover, the surrogate objects in the static network (usually hosted in MSS) act on behalf of the mobile host (MH) in a more convenient way [1]. Thus surrogate plays an active role in mobile data access by maintaining the current state of information about the mobile host and ensures proper delivery of data to the host. The following components, environments and configuration need to be considered for an effective object creation.

MH—Mobile host mobile devices such as smart phones, laptop, tablet, PDAs or any other smart devices can act as a mobile host or node. It starts data transaction with the other mobile host in the cloud via mobile support station.

MSS—Mobile support station mobile host connect to the static network through mobile support stations which have the capability to communicate through wireless interfaces. The MSSs are also interconnected to each other MSS through wired connections. This leads to a basic asymmetry in network connectivity and computing power.

SO—Surrogate object surrogate object is a software entity that is hosted on mobile support station and acts on behalf of a mobile device during disconnected communication. It contains data structure relevant to the mobile host and set of methods such as *task_offload()* and *process_service()* to act upon mobile host.

In DTO model, whenever a mobile host newly joins the cloud environment, it registers itself with an MSS. The MSS assigns unique identification for the host (MID) and passes the information to the underlying middleware running on its own cloud and later passes this to other clouds if necessary about the entry of new node to the cloud system. The middleware creates an object corresponding to the node and assigns a unique surrogate object identifier (SOID).

Now both mobile host and surrogate object are considered to be a part of a cloud. Surrogate objects provide seamless data transaction when mobile node is located in and out of the network coverage boundary of the MSS and also provides support for data access and delivers the result of transactions during mobility in wireless communication.

2.2 Computation Offloading for MCC

Computation Offloading, as one of the main advantages of MCC, is a paradigm [2–5] to improve the capability of mobile services through dynamic for migrating heavy computation tasks to powerful servers in clouds. Computation of offloading yields to dynamic offloading saving energy for mobile host, while running intensive computational services, which typically drain a device's battery when being run locally [6]. Nowadays, because of virtualization techniques enable cloud computing environments to run services for mobile devices remotely, there has been a significant amount of research focusing on computation offloading. These research themes are mostly related to explore ideas to make computation offloading feasible, to draw optimal offloading decisions, and to develop offloading infrastructures [7–10]. There are many factors that can adversely affect the efficiency of offloading techniques, especially bandwidth limitation between mobile devices and servers in the cloud and the amounts of data that must be exchanged among them. Many algorithms have already been proposed to optimize offloading strategies to improve [11] dynamic offloading algorithm computational performance to save energy [12–15].

A task offloading problem can be achieved by suboptimal solution using heuristic algorithm. This improves the average system offloading utility over traditional approaches [16, 17]. With Computational Offloading, a mobile host must decide which tasks should be offloaded to a cloud, to minimize the energy consumption while satisfying a computational time constraint. The offloading approach may reduce energy consumption and improve the performance of the mobile host [18–20].

The offloading approach may reduce energy consumption and improve the performance of the mobile host. But, it depends on the parameters such as the network bandwidth and the data exchange rate are under MSS to assign Surrogate object. The decisions are usually made by analyzing parameters such as network bandwidth, operating speed, load and available memory space of the cloud server, and the amount of data exchanged between the cloud servers and mobile host.

The prime focus of this paper is on developing a Dynamic Offload Algorithm model, for various applications as well as protocols in distributed mobile cloud platform using the surrogate object is presented in this article also provides a new perspective for designing a distributed mobile cloud system.

3 Proposed Model

3.1 Dynamic Task Offloading Strategy (DTO)

In this section, the proposed surrogate object based dynamic task offload model is presented. The Sect. 3.1 describes the concept of dynamic task offloading 3.2 describes the design and structure of our proposed model; Sect. 3.3 provides a detailed description of this model.

3.2 Concept of Dynamic Task Offloading

Offloading is defined as a procedure that migrates resource demanding computations from a mobile host to the cloud or nearby cloud. Mobile Cloud-based computation offloading enhances the application's performance, reduces battery power consumption, and process execution that is unable to execute due to insufficient storage resources on a mobile host. It means that complicated parts of an application run on the cloud and results are communicated back to the mobile host. The three key measures that involve the concept of offloading are as follows:

3.2.1 Mobile Host Classification

The mobile host is classified into three types according to time, energy cost and task computing process. The threshold value can be measured regarding processing time, energy consumption and memory usage. If a particular application, to be offloaded, takes enough time to compute on the mobile host, which consumes much energy and battery lifetime of the client, and need much battery for storage and surpass the minimum threshold value, it is effective to offload the application onto the cloud.

3.2.2 Priority Determination for Data Process

We assign the priorties of the mobile host, which choos offloading their task to the MSS server which determined by the wireless communication state and the task requirement. The dynamic offloading decision depends upon the size of the application to be offloaded.

3.2.3 Critical Section

When system involves in complex computations it requires more time to execute in the critical section, then it will be offloaded. When Complex processing of the application is offloaded in the cloud the resources of the mobile host will be saved.

3.3 Design and Structure

A typical surrogate object based dynamic task offloading model (DTO) architecture structure is shown in Fig. 2. The structure includes various components:

(a) a fixed DTO model with the function of Task Scheduling Algorithm(TSA) for computation offload of data stream in distributed mobile cloud
(b) a mobile host (MH) that can initiate and submit the transaction request and move freely throughout any Mobile Support Station(MSS);
(c) a Mobile Support Station (MSS) that connects the mobile host to the database server through Surrogate object (SO) or another mobile host within a same MSS or nearby MSS.

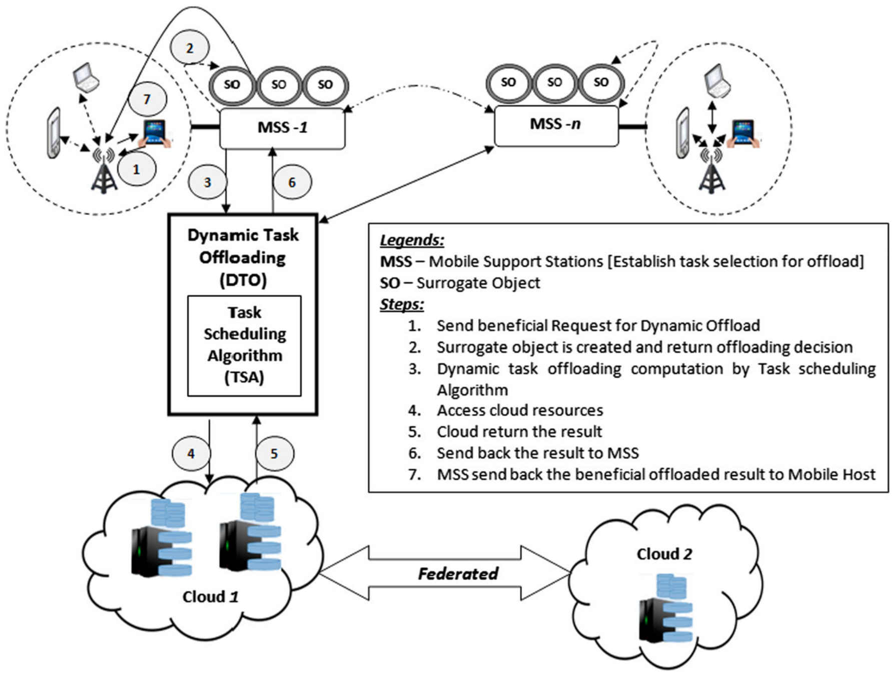


Fig. 2 System architecture of the dynamic task offloading using surrogate object

(d) a surrogate object that is created by MSS to act on behalf of each mobile host during disconnected network.
(e) the Cloud providers, provides the surrogate object of each mobile host and generates a series of reading only transactions requesting some of the data items or frequently used data items stored at the database server; then these data items are retrieved and stored in their object of each mobile host. The object is a non-volatile storage that supports transactions execution in both normal and disconnected mode.

3.4 Detailed Description

The Dynamic Task Offloading using Surrogate object model in which the mobile applications are necessary to make use the resources of the mobile device. Resources of the mobile devices are limited in quantity, so dynamic offloading proves to be useful for such mobile devices using surrogate object of each mobile host. Dynamic Task Offloading is a process comprising of series of steps. Foremost step is to start the system, the three questions arises like to check the threshold value, the weight of the process and critical section; if the weight is smaller than the operation it is typically executed with their local cache and if disconnected of service the surrogate object helps to compute the data. If the weight is larger means, the process is dynamically offloaded using our Task Scheduling Algorithm (TSA) which explains in Algorithm which describes the application is offloaded with the help of DTO.

Algorithm

Algorithm for Task Scheduling (TSA)

```
Input: the set of ordered mobile host (MH) and weight (W) of process of mobile host.
Output: check availability of small process or large process
set of the mobile host S = {1..N}.
for time t = 1 to N do
        check the available surrogate resource of cache on local
    if |S| >W + 1 then
        initialization:
            each mobile host MH → surrogate object (so_n) chooses the computation
            decision so_n(0) = 1 and cache.
        end initialization
            repeat for each mobile host so_n and each decision time slot t in parallel:
             measure the interference of weighted process as μ_n(t).
             compute the best response set Δ_n (t).
        if Δ_n(t) != ∅ then
            choose the decision of task scheduling as so_n(t + 1) ∈ Δ_n(t) for next slot.
            choose appropriate surrogate object for the request message to other users.
            choose nearby MSS or mobile host or federated cloud
        else choose the original decision so_n(t + 1) =so_n(t) for next slot.
        end if
    until all request are offloaded for mobile hosts consecutive slots
    else set S equals empty set.
end if
end for
return S.
```

The different operations involved in surrogate object model for dynamic task offloading are explained below.

1. The System Architecture describes that the mobile user or mobile hosts has their own cache in local and their surrogate object gives request to DTO for the process, the DTO analysis the weight of the data process and assign as High, Medium and Low.
2. If the requested weight of the process is Low it means execute in existing cloud with existing mobile transaction, using surrogate object message transaction with maintaining cache consistency at the surrogate object. If Medium it means the transaction is migrated with nearby mobile support station. Then finally, High means the transaction is dynamic offloaded follows the next step.
3. If the request process data is large and complex message transaction means, the respective surrogate object carried out the process of task scheduling algorithm in our system. The TSA has distributed the requested resource for transaction to increase the throughput, migration cost of cloud from one cloud to another cloud and energy efficiency, and the cost will be reduced.
4. Based on the algorithm the dynamic offloading process is distributed and access with the nearby mobile host or MSS.

5. If all the resources are engaged, and the message transaction is not available in their own cloud then, it will be migrated to another cloud which term as a federated cloud. The message transactions are processed in the federated cloud and sent back the result to the requested resource of the mobile host.
6. The federated cloud is used to reduce the cost of the cloud resource which reduces the migration cost in cloud resource.
7. With the help of surrogate object mobile transaction, all the requested processes will be executed at high speed without any delay and disconnection of the mobile transaction.
8. Each mobile host and MSS has the surrogate object, which results in less data loss in message transaction.
9. The surrogate object model for the dynamic offloading algorithm is executed it allocates the dynamic distributed process of the available resources based on the message transaction.
10. Finally, the request process is executed without any data loss in wired or wireless message transaction on the surrogate object.

4 Performance Analysis

In this section, we propose to figure out the parameters by using a surrogate object for dynamic offloading. The simulation studies was also conducted for several times to compare the DTO model with other offloading models, such as computing offload with and without dynamic.

The parameters we have take for dynamic task offloading model in CloudSim and WorkflowSim simulator, this simulation is performed using CPU core i3, memory 4 GB and 4 MB cache memory.

The proposed algorithm is designed for the simulator in CloudSim with java program *task_offload()* method the step which follows:

Step 1 Initialize Datacenter Broker for Mobile support Station (MSS). The status of the Virtual Machine and the state of the existing are tabulated for surrogate object (SO). At the time of initialization no Virtual Machines were allocated the Cloudlet act as a task.
Step 2 When there is a request to allocate new Virtual Machine come Datacenter Broker, DatacenterBroker analyzes the status. Then, calculate the total execution time of all existing cloudlet in the queue and the expected completion time of the new cloudlet being prepared for processing.
If the VM has the smallest processing time expected that machine is chosen to submit the next cloudlet. If there is more than one, the first VM is selected.
Step 3 Send the selected VMID (SOID) to the Datacenter Broker (MSS) then the Datacenter broker send the cloudlet to the VM allocated by that ID.
Step 4 Datacenter broker (MSS) notifies about new allocation and updates to VM (SO) and cloudlet status.
Step 5 When the VM (SO) completes the processing request and Datacenter Broker (MSS) receives the Cloudlet (task) response, it will update the Cloudlets status as completed and reduce a cloudlet in the status.

Table 1 CloudSim and WorkflowSim Parameters

Entity type Cloudsim	Process parameter	Value of resource
Datacenter	Number of datacenter	1
Process of CloudSim	Length of process in time	1–1000
	Total number of process	100
	Weight of process	High, medium and low
	MIPS of PE	1000–30000 MIPS
	Number of PEs per host	1–4
Cloudlet	Number of cloudlet	10–60
	Number of hosts	2–4
	Length of cloudlet	1024–20480
	VMM & VMs scheduler	Xen & Resource shared, Time shared and multiple threshold
	Operating system	Windows
	Architecture	X86
	PE number	1–3
Virtual machine	Number of VMs	10
	VM memory	512–3072
	Bandwidth	1024 MB
	WorkflowSim scheduler	Resources shared and time shared
	Number of processor equipments	1–4

Table 2 WorkFlowSim Parameters

Arch	OS	Vmm	time_zone	cost	costPerMem	CostPerStorage	costPerBw
x86	Linux	Xen	10.0	0.3	0.05	0.1	0.1

Step 6 Repeat the step 2 until the process is completed.

The purpose of this simulation is to compare, analyze, the response time and execution time of the proposed algorithm are evaluated. The system parameter are mentioned in Table 1, the results have been obtained with a system simulator program setting are shown in Table 2 and the parameter of cloudlets is shown in Table 3.

The proposed algorithm is simulated in a simulation toolkit workflowsim. We have created a data center with the following properties and created 10 virtual machines.

The calculated average execution time and average response time of all the tasks are:

- Avg execution time: 285.64 ms.
- Avg response time: 1670.46 ms.

The above result shows that our proposed algorithm has better response times that the existing without the task offloading (Fig. 3).

Table 3 Parameters of cloudlets

Cloudlet IDs	Length of cloudlet	No. of required PE
0	2000	1
1	3000	2
2	4000	1
3	5000	2
4	3000	1
5	2000	2
6	2000	1
7	1000	2
8	6000	2
9	4000	1

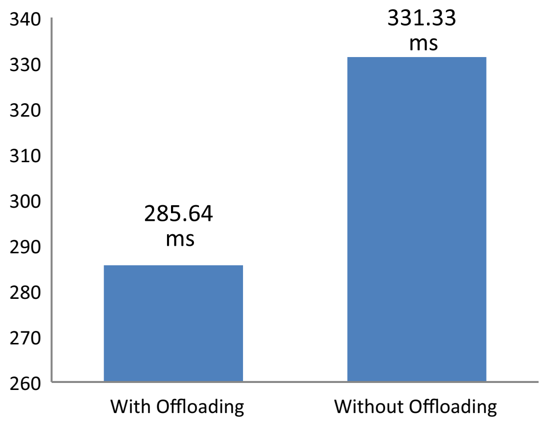


Fig. 3 Comparison of execution time (ms) with and without proposed offloading algorithm

The CloudSim toolkit is used to simulate heterogeneous resource environment and the communication environment. CloudSim simulator is used to verify the results. The experiments are performed with Sequential assignment which is default in CloudSim with the proposed algorithm. The jobs arrival is Uniformly Randomly dynamic Distributed to get generalized development. The scheduler submits these jobs on available resources according to these algorithms.

In DTO model, each transaction initiated by a mobile host is forwarded to respective MSS; the MSS divides the transaction into several sub-transactions and transmits them to the respective surrogate object. From the participating surrogate objects with dynamic offload and validated transaction happen locally. However, in most of the existing models, the transactions were executed either in a remote database server or the respective mobile node and validated globally. This simulation result varies each time depending on the arrival rate of transactions. The process transaction for time and it is executed with available resources (Virtual Machines) and is monitored by Virtual Machine Monitor (VMM) in CloudSim simulator is shown in Fig. 4.

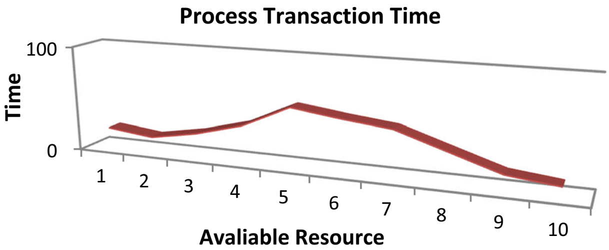
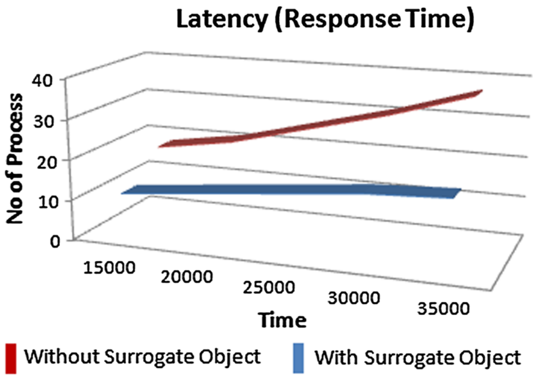


Fig. 4 Process transaction time for virtual machines

Fig. 5 Comparison of latencies (response time)



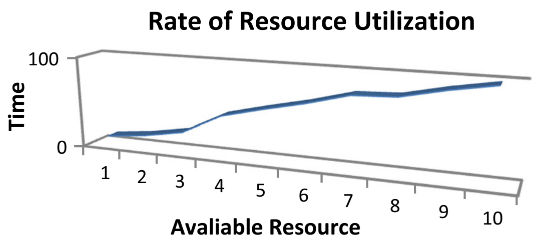
The same simulation prototype also experimented by Aneka cloud simulator which is .Net based service oriented management platform for only the latencies of the task. In this simulator we consider 10 nodes which, configured and monitored properly through the node selection module, and service setting modules. The nodes are grouped into different clouds. The two different clouds are created with advance setting in simulator such as cloud1, and cloud 2. After a successful creation of nodes for two different clouds, the user can create a surrogate objects and its cache in the respective nodes by using management tool and data base setting tools and finally the application was implemented over this setup. Figure 5 shows the comparison of the latencies (in terms of response time) in the caching process with and without of the surrogate object.

4.1 Resource Management

Cloud can offer you the possibility of storing your files and accessing, storing and retrieving them from any web-enabled interface. The web services interfaces are usually simple. At any time and place you can have high availability, speed, scalability and security for your environment. In this scenario, organizations are only paying for the amount of storage they are actually consuming, and do so without the worries of overseeing the daily maintenance of the storage infrastructure.

Resource management and DTO grants the request made by the client to the server (cloud) which increasing the throughput of the application being dynamic offloaded.

Fig. 6 Rate of resource management



This process is build by *process_service()* method in java program and experimented in CloudSim with the following steps:

Step 1 Calculate completion Time for all tasks on each resourcefor all submitted task in MSS. (T_i)
Step 2 For all resources (R_j) calculate the expected Completion Time of Task ($_i$) on resource ($_j$) and find the earliest completion time (E) and the resource (rj)
$C_{ij} = E_{ij} + rj//\quad Cij$ means expected completion time
Step 3 Do until all tasks are mapped
Step 4 For each task in MSS find the earliest completion time and the resources
Step 5 Find the task with the minimum completion time (T_k)
Step 6 Assign T_k to the resource R_j, gives the earliest completion time
Step 7 Delete task T_k from the MSS
Step 8 Update ready time of resource (R_j) and update C_{ij} for all $_i$.
Step 9 End Do
Step 10 Sort the resource according the completion time (rescheduling)
Step 11 Calculate the maximum completion time (mk) =max(CT(R)) *[MCT]* for all the resource (R)
Step 12 Calculate the next minimum completion time of the tasks which need the resources for all the tasks
Step 13 Calculate the maximum estimated completion time (ET) in resource R_j for all the resource (T_i)
Step 14 Compare MCT < mk; if less than and true reschedule the task to the resource
Step 15 Update the ready time of both resource and task, finally end the condition process.

In the simulation result, the rate of resource management increase in throughput with available resource and is fully used efficiently with latency in time is shown in Fig. 6.

4.2 Federated Cloud and Migration Cost of Cloud

Federation streamlines the services in cloud technology while meeting back office, plant operation, and day-to-day employee needs. For an affordable cost, the outsourced data standardize and connect in public infrastructure across multiple locations. DTO services include anytime, anywhere, anyhow access to all of your valuable data are dependable by 24 × 7 × 365 support. The challenge is to minimize the migration cost

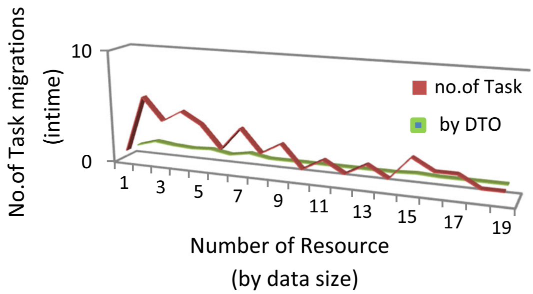


Fig. 7 Transaction of federated and migrated cost

while executing the application on the cloud. The level of migration plays a huge role in minimizing the migration cost. The transaction between the number of messages from the mobile user and available resource on which application is running and the remote server (cloud) also contribute to the latency rate. The near-by devices have low execution time as compared to farther ones. Figure 7 shows the rate of the transaction for cost effiency for task migration or federated by the TSA which assign the weight as Medium and High for dynamic task offloading in the Simulated result.

4.3 Data Analytics

One of the aspects offered by leveraging Mobile Cloud computing is the ability to tap into vast quantities of both structured and unstructured data to harness the benefit of extracting business value. For instance, retailers and suppliers are now extracting information derived from consumers' buying patterns to target their advertising and marketing campaigns to a particular segment of the population. Social networking platforms are now providing the basis for analytics on behavioral patterns that organizations are using to derive meaningful information. Moreover, also the mobile data transaction is available with their cloud or nearby host if not then federated to next cloud the latency of federated cloud. Finally, gives a good outcome result which depends on the availability of the resources and the calculations are performed on the actual processing time which produces effective results with low memory cost of the mobile host and minimum execution time in cloud. The rate of the dynamic offloading is shown in Fig. 8.

4.4 Energy Efficiency

Energy is a primary constraint for mobile host. The surrogate object for dynamic offloading conserves energy and battery lifetime of the mobile host by migrating energy-intensive computations to the cloud. The surrogate object for dynamic offloading makes it possible for the applications to run on the cloud server and the output is sent back to the local mobile host. The complex computations when performed on the limited resources; consumes enough time and energy. For example, playing games on the mobile host consumes enough battery of the device. When playing games with the

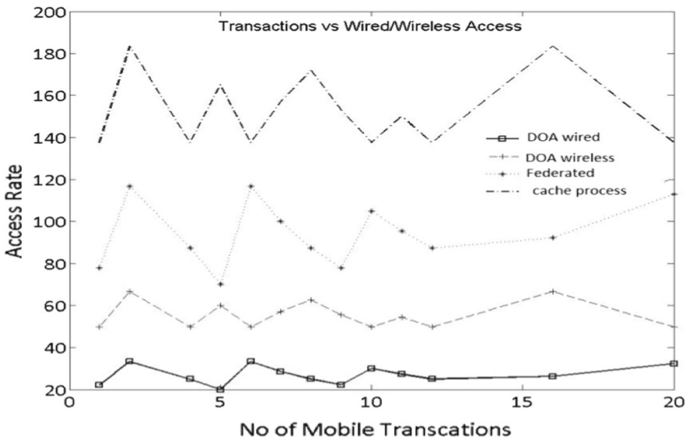
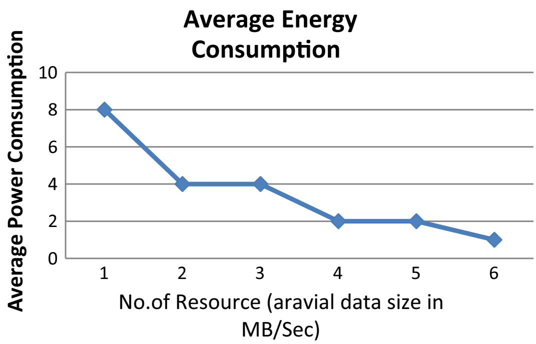


Fig. 8 Data analytics transaction of dynamic offloading

Fig. 9 The average energy consumption based on selection data size



cloud support; the user just interacts with the GUI of the game and all the moves of the game are performed on the cloud. Social networks also share images on the cloud. By using this system, users can save energy and memory of their mobile host to a large extent. Mobile applications do not face storage space as a constraint as their data is stored in the cloud. Figure 9 shows the energy utilization rate according to the different data size and execution time, the cost is lower than the traditional optimization method with growth of data arrival rate.

5 Conclusion and Future Scope

As a result, the surrogate object model is presented as a new model for dynamic offloading and it is built on the top of surrogate object mobile transaction model. The Mobile Cloud Computing is the integration of cloud computing with mobile host which provides rich communication between cloud-mobile users and cloud providers regardless of heterogeneous environments. In such a way the task scheduling algorithm helps in dynamic task offload system of request process for the complex calculations to be performed on cloud and limits energy consumption and battery usage of the mobile host. From the simulation, it clearly shows that the proposed model provides the real benefit of using the surrogate object of dynamic offloading regarding network traffic, utilizing minimum wireless and wired access, achieving the low response time and quality of services with, bandwidth utilization between mobile devices and cloud servers. Further, the proposed model will be enhanced solutions for the issues such as resource management that assigns the tasks in available service providers so that the application is executed on time. Minimized federated and migration cost is another solution to have a check on the latency rate.

References

1. Satyanarayanan, M.: Mobile computing: the next decade. ACM SIGMOBILE Mobile Comput. Commun. Rev. **15**(2), 2–10 (2011)
2. Shi, Z., Gu, R. (2013) A framework for mobile cloud computing selective service system. In: Proceedings of IEEE Wireless Telecommunications Symposium (WTS), pp. 1–5
3. Kumar, K., Liu, J., Lu, Y.-H., Bhargava, B.: A survey of computation offloading for mobile systems. Mobile Netw. Appl. **18**(1), 129–140 (2013)
4. Maluk Mohamed, M.A., Janaki Ram, D., Chakraborty, M. (2005) Surrogate object model: a new paradigm for distributed mobile systems. In: Proceedings of the 4th International Conference on Information Systems Technology and Its Applications (ISTA'2005), New Zealand, 23–25 May 2005, pp 124–138
5. Shiva Narayana Reddy, V., Venkat Reddy, A., Mahesh Kumar, P., Seshagiri, B.: Global transaction models on mobile computing. Int. J. Comput. Sci. Technol. **3**(3), 191–196 (2012)
6. Kesavaraj, G., et al. (2016) An Analytical Study on Mobile Computation Offloading For Cloud Computing. Int. J. Future Innov. Sci. Eng. Res. (IJFISER) **2**(1), 98, ISSN: 2454-1966
7. Mukherjee, S. (2003) A modified Kangaroo model for long lived transactions over mobile networks. Accepted Paper, IEEE region III SOUTHEASTCON, Ocho Rios, Jamaica, 4–6 April 2003
8. Ku, K.-I., Kim, Y.-S. (2000) Moflex transaction model for mobile heterogeneous multidatabase systems. In: Research Issues in Data Engineering, pp. 39–46
9. Le, H.N., Nygård, M. (2005) A mobile affiliation model for supporting mobile collaborative work. In: Workshop on Ubiquitous Mobile Information and Collaboration Systems, pp. 649–660
10. Ravimaran, S., Gnana Jeevan A.N., Maluk Mohamed, M.A. (2015) Information security using surrogate object based encryption in mobile cloud systems. WSEAS Trans. Comput. **14**, 79–87, E-ISSN: 2224-2872
11. Le, H.N., Nygård, M. (2005) Mobile transaction system for supporting mobile work. In: 7th International Database and Expert Systems Applications (DEXA) Workshop on Mobility in Databases and Distributed Systems, pp 1090–1094
12. Kaneda, T., Shiraishi, M., Enokido, T., Takizawa, M. (2004) Mobile agent model for transaction processing on distributed objects. In: International Conference on Advanced Information Networking and Applications, pp. 506–511
13. Chrysanthis, P. (1993) Transaction processing in mobile computing environments. In: Proceedings of IEEE Workshop on Advances in Parallel and Distributed Systems
14. Chrysanthis, P.K. (1999) Transaction processing in mobile computing environment. In: IEEE Workshop on Advances in Parallel and Distributed Systems, pp. 77–83
15. Walborn, G.D., Chrysanthis, P.K. (1999) Transaction processing in PROMOTION. In: ACM Symposium on Applied Computing, pp. 389–398
16. Tran, T.X., Pompili, D. (2017) Joint task offloading and resource allocation formulti-server mobile-edge computing networks. National Science Foundation (NSF) Grant No. CNS-1319945,1705.00704v1
17. Bahl, P., Han, R.Y., Li, L.E., Satyanarayanan, M. (2012) Advancing the state of mobile cloud computing. In: The third ACM Workshop on Mobile Cloud Computing and Services
18. Al-Rousan, M., Al-Shara, E., Jararweh, Y.: Cloudlet-based ad hoc mobile cloud computing: design and evaluation. Int. J. Embed. Syst. **9**(5), 456–463 (2017)
19. Shahzad, H., Szymanski, T.H. (2016) A dynamic programming offloading algorithm using biased randomization. In: IEEE Cloud. IEEE

20. Zhang, F., Ge, J., Li, Z., Li, C., Huang, Z., Kong, L., Luo, B. (2017) Task offloading for scientific workflow application in mobile cloud. In: Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security (IoTBDS 2017), pp. 136–148, ISBN: 978-989-758-245-5, by SCITEPRESS—Science and Technology Publications. https://doi.org/10.5220/0006364501360148