Soft Computing (2018) 22:6751–6764
https://doi.org/10.1007/s00500-018-3264-0

FOCUS

Energy conscious multi-site computation offloading for mobile cloud computing

Raj Kumari[1] · Sakshi Kaushal[1] · Naveen Chilamkurti[2]

Published online: 28 May 2018

Abstract
In mobile devices, one of the major reasons for battery consumption is due to computation of complex applications. To provide high performance of execution on the mobile devices, the concept of mobile cloud computing (MCC) is used. MCC allows the computation complex modules to offload onto a cloud from a mobile device, which helps to remove the resource constraint condition of the mobile devices. Computations can also be offloaded to nearby clouds called multi-sites, which may have different resources, access delays, computation capability or service charges. The mobile users can specify their priority such as total completion time, cost or energy saving of the application execution in MCC environment. But most of the existing research is focused to optimize only one objective, i.e., either total completion time or cost or energy. But when offloading computation modules onto cloud-based multi-sites, a tradeoff solution is required to strike a balance between the total completion time and energy savings. Further, the entire computational execution on the cloud is to be served efficiently with optimal power utilization. Various algorithms are developed to reduce power consumption, and one such algorithm is dynamic voltage and frequency scaling (DVFS) algorithm. In this paper, new algorithms known as cost and time constraint task partitioning and offloading algorithm (CTTPO), multi-site task scheduling algorithm (MTS) based on teaching, learning-based optimization and the energy saving on multi-sites (ESM) using DVS technique are proposed. CTTPO deals with trade-off between time and cost for the task partitioning and offloading. The MTS algorithm deals with time efficient scheduling on multi-sites, and ESM algorithm saves the energy on the multi-sites by switching the sites from high voltage to low voltage during ideal time. The simulation study demonstrates that the proposed algorithms outperformed the existing techniques based on time, cost and energy parameters.

Keywords Mobile cloud computing · Energy aware · Smart mobile · Multi-tasking

1 Introduction

Due to recent advances in smart phone technologies, complex and interactive applications like face recognition, mobile healthcare monitoring, augmented reality and mobile gaming are being executed on mobile devices. These applications require high processing resources to generate fast response time. Due to this high requirement, the concept of computation offloading has been proposed by the researchers. The basic idea of computation offloading is to offload the computational complex module of the application from the mobile device to resource full cloud (Kumar and Lu 2010). The high performance processors on a cloud can execute the computation complex module with fast response time. But the performance of data offloading on the cloud is highly affected by the available bandwidth (Shu et al. 2013). In regard to computational performance for complex applications and resources on the mobile device, a new technology known as mobile cloud computing (MCC) technology was recently introduced. The MCC (Bheda and Lakhani 2013; Shiraz et al. 2014; Dinh et al. 2013; Khanghani and Ravanmehr 2013; Gajbhe and Sakhare 2015; Kumar et al. 2012)

Communicated by G. Yi.

✉ Naveen Chilamkurti
n.chilamkurti@latrobe.edu.au

Raj Kumari
rajkumari@pu.ac.in

Sakshi Kaushal
sakshi@pu.ac.in

[1] University Institute of Engineering and Technology, Panjab University, Chandigarh, India

[2] La Trobe University, Melbourne, Australia

is an emerging paradigm that encompasses mobile computing, cloud computing, networking and virtualization. The services and infrastructure of the cloud can be used by the users based upon the service level agreement (SLA) and charged on a pay as you use basis. Shiraz et al. (2012) proposed a lightweight distributed computational offloading framework (DCOF) for computational offloading in MCC. DCOF employs distributed approach for the configuration of intensive mobile application between mobile device and cloud server node. Sinha and Kulkarni (2011) describe efficient partitioning and offloading approaches for performing fine-grained, multi-site offloading. The state-of-the-art computational offloading framework shows that the partitioning of the application can be done statically or dynamically on a single site or multi-sites. In static partitioning, the affordable components are decided prior to the execution of the application. In dynamic partitioning approach, the off loadable components are decided during the execution of the application. These frameworks have also discussed the different granularity levels of the application partitioning like class level, method level, component level, module level, etc. (Shiraz et al. 2014). In general, mobile applications are broadly divided into two categories, i.e., computation intensive and communication intensive. Depending upon the nature of the application, application partitioning is done in such a way that overall balance between time, cost and energy can be maintained. This is also one of our objectives in this work. In MCC, as application partitioning is performed to enhance the battery life of the mobile device, the execution of the offloaded module on cloud is required to fulfill certain criteria's such as strict execution time deadlines and cost as per the specified QoS constraints. As discussed above, cloud environment provides resources on charge on the basis of as you use policy. High performance resources charge more as compared to slow performance resources. Thus, different scheduling plans for the application execution in MCC with preference to the cost, time and energy are possible with different policies for the task partitioning. Therefore, the optimized scheduling of the application in MCC requires partitioning of the application with time and cost constraint as per user satisfaction. The time constraint ensures the total execution time, which included the offloaded module execution time and transfer time within the given deadline and the overall cost of the application execution to be within the specified user requirement limit.

In recent years, the researcher focus is on algorithms like simulated annealing (Pandit et al. 2014; Ying and Lei 2014), genetic algorithm (Islam and Islam 2016; Deng et al. 2014) and particle swarm optimization (PSO) (Nguyen et al. 2011) to achieve the best solution for application offloading and scheduling. Some authors have considered the concept of dynamic voltage and frequency scaling (DVFS) (Wang et al. 2010; Park et al. 2013; Kahng and Kang 2013) to make the system more energy efficient. In this paper, we propose various new algorithms known as cost and time constraint task partitioning and offloading algorithm (CTTPO), multi-site task scheduling based on teaching, learning-based optimization (MTS) and the energy saving on multi-sites (ESM) using DVS technique. CTTPO deals with a trade-off between time and cost for the task partitioning and offloading. The MTS algorithm deals with execution on multi-sites, and ESM algorithm saves the energy on the multi-sites by switching the sites from high voltage to low voltage during the ideal time. The main contributions of this paper are summarized below.

- Using mobile cloud computing (MCC), tasks on mobile devices are offloaded to the cloud for speeding up execution time and saving battery life of the mobile devices.
- To make task partitioning decisions, cost or time parameters are considered according to the user's preferences.
- The energy efficient scheduling of the tasks on multi-site is done by implementing an ESM algorithm to obtain the maximum energy savings under a tight deadline and budget constraints.
- The parameters such as the execution time, energy consumption and energy saving are considered to analyze the performance of the proposed work.

The rest of this paper is organized as follows: Sect. 2 presents the related work on time, cost and energy efficient applications in MCC. The problem description is presented in Sect. 3. Section 4 introduces the approach of optimization and describes the proposed time, cost and energy efficient algorithms based on multi-objective teaching Learning-based optimization algorithm. Section 5 discusses the simulation and analysis of the results. Finally, Sect. 6 concludes the paper with a summary of our findings.

2 Related work

Using MCC, mobile devices are able to meet the requirements of various applications. Instead of being fully dependent on the cloud, different types of platforms like cloudlets and mobile ad hoc cloud (MAC) (Yaqoob et al. 2016) can be used. Ahmed et al. (2013) have discussed the long or permanent network disconnections due to user mobility which increase the execution time. They have proposed the use of process state synchronization (PSS) mechanism to mitigate the impact of network disconnections on the service continuity of cloud-based interactive mobile applications. They develop a mathematical model that incorporates the disconnection and synchronization intervals, and mobile device capabilities along with cloud. The results show the PSS reduces the execution time as compared to COMET and VM-based offloading. Multi-factor multi-site risk-based offloading model is considered in Wu and Huang (2014) for offloading benefits and offloading risk. They used fuzzy inference to aggregate the overall offloading benefits and risks based on the mobile cloud users preference, and used ant-based algorithm to calculate the assignment from application components to surrogate sites. A multi-site offloading execution using energy-efficient multi-site offloading policy (EMOP) algorithm for reduction on the energy consumption of mobiles when compared to a single site offloading execution is discussed in paper (Terefe et al. 2016). They presented a novel model to describe the energy consumption of a multi-site application execution and use a discrete time Markov chain (DTMC) to model fading wireless mobile channels. They adopt a Markov decision process (MDP) framework to formulate the multi-site partitioning problem as a delay-constrained, least-cost shortest path problem on a state transition graph. For service workflows in a mobile cloud computing environment, Deng et al. (2014) proposed mobility enabled, and fault-tolerance offloading system for making computation offloading strategies. They used offloading system based on the genetic algorithm (GACO). GACO randomly defines the initial population on which the fitness function is applied to elect chromosomes with minimal energy consumption and execution time. In the crossover step, standard single-point crossover operator has been used. In the mutation step, a gene is muted with a probability proportional to its fault occurrence and mobility sensibility. Xiang et al. (2014) have addressed the issue of energy-efficient link selection and data transmission scheduling for delay-tolerant and data-intensive applications in MCC. They formulate the problem as a discrete-time stochastic dynamic program (SDP) that aims to optimize both system throughput and energy consumption. Wu and Wolter (2014) addressed the issue of energy-efficient offloading that migrates data-intensive, but delay-tolerant applications for the mobile devices to a remote cloud. Dynamic scheduling and link selection based on Lyapunov optimization for data transmission between the mobile devices and the cloud reduced the battery consumption of the mobile devices for transferring large volumes of data. To minimize the delay, a control algorithm is derived which determines when and on which network to transmit data so that energy-cost is minimized by leveraging delay tolerance. The focus of the paper (Zhang et al. 2016) is on the design of an energy-efficient computation offloading mechanism for mobile edge computing (MEC) in 5G heterogeneous networks. They designed an energy-efficient computation offloading (EECO) scheme, which jointly optimizes offloading and radio resource allocation to obtain the minimal energy consumption under the latency constraints. Zhang et al. (2017) have performed offloading of real-time video applications. For offloading real-time video applications, a generic energy-efficient offloading scheduling problem and adaptive scheduling algorithm are proposed that make fine-grained offloading decisions according to the dynamic wireless network conditions. In Wu et al. (2016), the authors have addressed the wireless network conditions for application partitioning. In this paper, authors have proposed a novel min-cost offloading partitioning (MCOP) algorithm that aims at finding the optimal partitioning plan under different cost models and mobile environments. They overtake the disadvantages of static partition with dynamic partitioning with dynamic network conditions. Wang et al. (2010) have used the slack time for noncritical jobs, to extend their execution time and reduces the energy consumption without increasing the overall task's execution time. They have proposed a framework for run time application repartitioning in the dynamic mobile cloud environment. The objective of Goudarzi et al. (2017) is to minimize the energy consumption of mobile device and execution time of the application. They have proposed the fast hybrid multi-site computation offloading (FHMCO) framework which consists of two different decision algorithms for optimal and near optimal offloading partitioning based on the size of the mobile application. The framework is proposed (Shaukat et al. 2016) in which a mobile user exploits virtual machine (VM) technology to rapidly instantiate customized service software on a nearby cloudlet and then uses that service over a wireless LAN. In wireless networks, a common energy-saving technique is to turn on the mobile device's transceiver only for short periods of time to receive and acknowledge packets that have been buffered at a base station, which increases average end-to-end packet latency as well as jitter. Two approaches that deliver VM state to cloudlet are VM migration in which an executing VM is paused, all its states are transferred to the cloudlet, and then VM execution is again resumed at the cloudlet and dynamic VM synthesis in which mobile device sends a small VM overlay to the cloudlet that already has base VM from which overlay VM was derived.

From the review of the literature, it has been found that scheduling of task in multi-site has an impact on overall system performance. Many researchers have worked on an efficient task partitioning and task scheduling. The objective of the work done by paper (Goudarzi et al. 2017) was to minimize the energy consumption of mobile device and execution time of the application. They have proposed the fast hybrid multi-site computation offloading (FHMCO) framework with optimized multi-site particle swarm optimization (OMPSO) algorithm which evaluates the near optimal solution. The approach of task partitioning, offloading and scheduling used in our paper shows better results as compared to work done in the paper (Goudarzi et al. 2017). Therefore, to show the effectiveness of our work, the multi-objective optimization approach is used under time and cost constraint for task partitioning, offloading and scheduling. Also to make the proposed work energy efficient, DVS technique is applied.

3 Proposed work

In this paper, we propose a novel process of task offloading to reduce the workload and energy consumption of the mobile device, where some tasks are offloaded to the cloud for execution. While doing so, an energy efficient task scheduling is necessary to decide whether one or more task shall be offloaded to the cloud while the time and cost constraints of the application are satisfied. In this section, time, cost and energy model for application partitioning and offloading in MCC is discussed.

3.1 Application model

Suppose an application contains a sequence of n modules. The application is modeled by a graph, defined by a tuple $G\,(N, E)$, where N is the set of k modules $\{n_1, n_{2,\ldots} n_k\}$ and E is a set of e edges, which represents the dependencies. Each $n_i \in N$ represents the module in the application and each edge $(n_i\_n_j) \in E$ represents a precedence constraints, and the execution of $n_j \in$ N cannot be started before $n_i \in$ N finishes its execution. Each module n_i can be executed on the mobile side or on the cloud side. The execution time of n_j is $\mathrm{ET}_{(i,s)}$ if it is offloaded onto the cloud otherwise, it is $\mathrm{ET}_{(i.m)}$, where $\mathrm{ET}_{(i,s)} > \mathrm{ET}_{(i.m)}$. If the two adjacent modules n_j and n_{j+1} execute on two different sides, the data transmission time is considered; otherwise the data transmission time between n_j. and n_{j+1} becomes zero when they execute on the same side. As the dependent tasks and multiple sites are taken, the transmission time between the sites during the data transmission is also considered. The size (tz_i) of module n_i is defined as millions of instructions (MI). $\mathrm{DS}_{(i,s)}$ is the data transfer of module i over the site s. The processing power of the cloud server is millions of instructions per second and denoted as CP. Similarly, MP is the mobile device processing power. The speedup factor of the sites s depicts the speedup of each site compared to the mobile device processing speed as mentioned in Table 1. To transfer the offloaded data from the mobile device to the cloud, the bandwidth rate is assumed in Kb/s.

3.2 Application partitioning and offloading

In this section, the application partitioning and offloading decision based on a cost model that considers the total execution time of an application and energy consumption are presented.

3.2.1 Execution model

Each module of an application can be executed either on the mobile environment or cloud site. The execution time of module i on site s is calculated as in Eq. (1) which is the summation of execution time and data transmission time.

Table 1 Parameters

Parameters	Definition	Values
Bandwidth (Kbytes/s)	Transmission bandwidth	[10–100]
SF^{x}	Speed factor	[2–4]
P^{transfer}(W)	Transmission power	0.7
Cost (price per unit)	Site processing unit cost	[0.4, 0.6, 0.8]

$$\mathrm{ET}_{(i,s)} = \frac{\mathrm{tz}_i}{\mathrm{CP}} + \frac{\mathrm{DS}_{(i,s)}}{B_{(i,s)}} \tag{1}$$

Total execution time for all the offloaded z modules on cloud side is calculated as in Eq. (2)

$$\mathrm{TET}_{(c)} = \sum_{i=1}^{z} \mathrm{ET}_{(i,s)} \tag{2}$$

The complete execution time of an application on the different sites is the summation of execution time on the different sites.

Similarly, the execution time for the non-offloaded application data processed on the mobile device side is calculated as in Eq. (3)

$$\mathrm{ET}_{(m)} = \frac{\mathrm{tz}_i}{\mathrm{MP}} \tag{3}$$

Total execution time on the mobile side is calculated as in Eq. (4)

$$\mathrm{TET}_{(m)} = \sum_{i=1}^{k-q} \mathrm{ET}_{(i,m)} \tag{4}$$

where k is the total modules of an application, q is the offloaded module and $(k-q)$ modules are for mobile execution. Therefore, total execution time of an application is the summation of Eqs. (2) and (4) as shown in Eq. (5)

$$\mathrm{TET} = \mathrm{TET}_c + \mathrm{TET}_m \tag{5}$$

Using these equations, the value for deadline D is obtained by executing the offloaded module on the site having_minimum values for processing speed, i.e., execution of module on the slowest site.

For an application, suppose the completion time of module n_j is $t_{(j,c)}$ if it is schedule on the cloud side and $t_{(j,m)}$ is the completion time of module n_j if scheduled on the mobile side. c_j is the cloud side computation time, m_j is the mobile side computation time, and ∂ is the data transmission time. If the two adjacent modules j and $j+1$

execute on the different sides, the data transmission time is considered otherwise it becomes zero. Recursive formulation of t_j (http://www.techrepublic.com/resource-library/whitepapers/resource-constrained-multi-user-computation-partitioning-for-interactive-mobile-cloud-applications/) is as:

$$t_{(\mathrm{j,c})} = \min\{t_{(\mathrm{j-1,c})} + c_{\mathrm{j}}\,,\, t_{(\mathrm{j-1,m})} + m_{\mathrm{j}} + \partial_{\mathrm{j-1,j}}\} \tag{6}$$

$$t_{(\mathrm{j,m})} = \min\{t_{(\mathrm{j-1,m})} + m_{\mathrm{j}}\,,\, t_{(\mathrm{j-1,c})} + c_{\mathrm{j}} + \partial_{\mathrm{j-1,j}}\} \tag{7}$$

where $j = 1, 2, 3 \ldots n, n + 1$. The module 0 and module $n+1$ are, respectively, the entry node and exit node added virtually into the application and computation time of these two modules is zero. As multi-sites are considered for offloading, during the application partitioning in Eqs. (6) and (7) cloud c is considered as slowest processing speed site. Using Eqs. (6) and (7), application modules are partitioned into off loadable and non-off loadable modules to be executed on the cloud side or the mobile device side.

Hence, the task partitioning and optimized execution time are calculated using CTTPO and MTS algorithms, presented in Sect. 4.

3.2.2 Cost model

On the cloud side, the multi-site offloading model is considered where each computational site $S = \{s_1, s_2, \ldots s_l\}$ is operating at different processing speed and costs. It is assumed that any module n_i from the application can be executed on the set S. The cost model is based on pay-as-you -use basis.

The execution cost of module n_i on the site s, $\mathrm{EC}_{(i,s)}$, is the product of unit price rate and execution time. It is defined in Eq. (8), where β is the price unit of using the site s for each time interval.

$$\mathrm{EC}_{(i,s)} = \beta * \mathrm{ET}_{(i,s)} \tag{8}$$

Total execution cost is calculated as presented in Eq. (9)

$$\mathrm{TEC} = \sum_{i=1}^{t} \mathrm{EC}_{i,s} \tag{9}$$

Using these equations, the value for Budget B is obtained by executing the offloaded module on the site having maximum values for processing, i.e., execution of module on the fastest site.

3.2.3 Communication model

The interconnection between a mobile device and multi-site is considered homogeneous, which implies that all data transfer between the mobile device and any site will be transmitted at the same rate (Zong et al. 2008). If $\mathrm{PL}_{\mathrm{active}}$ is the power of the link when it is active, then the energy consumption during the transmission of the data is calculated as per Eq. (10).

$$\mathrm{END}_{\mathrm{active}} = \mathrm{PL}_{\mathrm{active}} * \sum_{j=1}^{t} \mathrm{DS}_{(j,s)} \tag{10}$$

where $\mathrm{DS}_{(j,s)}$ is the data size of the module j of an application offloaded on to site s.

Similarly, the interconnection between the multi-sites is considered homogeneous; hence, the energy consumed during the data transfer between the sites is calculated as in Eq. (11)

$$\mathrm{ENS}_{\mathrm{active}} = \mathrm{PS}_{\mathrm{active}} * \sum_{j=1}^{t} \mathrm{DS}_{(j,s,k)} \tag{11}$$

where $\mathrm{DS}_{(\mathrm{j,s,k})}$ is the task size of the module j of an application offloaded on site s from the site k. $\mathrm{PS}_{\mathrm{active}}$ is the power between the sites when it is active

Total energy consumption during the transmission of data is the summation of Eqs. (10) and (11), as presented in Eq. (12)

$$\mathrm{ENC}_{\mathrm{active}} = \mathrm{END}_{\mathrm{active}} + \mathrm{ENS}_{\mathrm{active}} \tag{12}$$

3.2.4 Energy consumption model

The energy model used in this study is derived from the execution of the application on different sites with their processing speed and price rate for a module on site. The energy consumption types such as active energy consumption and ideal energy consumption (Zong et al. 2008) are calculated as shown in following equations. Active energy consumption is calculated when sites are performing computation and set voltage level to high. On the other hand, when the sites are under the ideal condition the voltage is set to a lowest level as shown in Table 2.

Let en_i be the energy consumption caused by the module n_i running on a computational site, and its energy consumption rate is $\mathrm{PE}_{\mathrm{active}}$, then the energy consumption of module n_i can be expressed as using Eq. (13)

$$\mathrm{en}_i = \mathrm{PE}_{(\mathrm{active})} * \mathrm{ET}_{(i,s)} \tag{13}$$

Total energy consumption on single site s is calculated as per Eq. (14)

$$\begin{aligned}\mathrm{EN}_{\mathrm{actice}} &= \sum_{i=1}^{t} \mathrm{en}_i = \sum_{i=1}^{n} \left(\mathrm{PE}_{\mathrm{active}} * \mathrm{ET}_{i,s}\right)\\ &= \mathrm{PE}_{\mathrm{active}} * \sum_{i=1}^{n} \mathrm{ET}_{i,s}\end{aligned} \tag{14}$$

Table 2 Voltage-relative speed pairs

Level	Pair 1		Pair 2		Pair 3	
	Voltage Vi	Relative Speed (%)	Voltage Vi	Relative speed (%)	Voltage Vi	Relative speed (%)
0	1.5	100	2.2	100	1.75	100
1	1.4	90	1.9	85	1.4	80
2	1.3	80	1.6	65	1.2	60
3	1.2	70	1.3	50	0.9	40
4	1.1	60	1.0	35	–	–
5	1.0	50	–	–	–	–
6	0.9	40	–	–	–	–

Similarly, total energy consumption on multiple sites is calculated as per Eq. (15)

$$\mathrm{TEN_{active}} = \sum_{i=1}^{m} \mathrm{EN_{active}} \quad \mathrm{m} \in \text{number of operating sites} \tag{15}$$

Let $\mathrm{PN_{ideal}}$ be the energy consumption rate of a computational site when it is inactive and assume that t_i be the completion time of the offloaded module. The energy consumed by an inactive site is a product of the ideal energy consumption rate $\mathrm{PN_{ideal}}$ and an ideal period. Equation (16) for ideal energy consumption on site s is

$$\mathrm{EN_{(ideal,\mathit{s})}}\mathrm{=PN_{ideal}} * \left(\max_{i=1}^{s}(t_i) - \sum_{i=1}^{s}\mathrm{ET}_{(i,s)}\right) \tag{16}$$

where $\max_{i=1}^{s}(t_i)$ is the maximum completion time of module i among the available sites and $(\max_{i=1}^{s}(t_i) - \sum_{i=1}^{s} \mathrm{ET}_{(i,s)})$ is the ideal time on the site s.

The total ideal energy consumption on the multi-site can be calculated as per Eq. (17).

$$\mathrm{TEN_{ideal}} = \sum_{i=1}^{m} \mathrm{EN_{ideal}} \tag{17}$$

Based on the above formulations, total energy consumption of an application can be defined by Eq. (18), which is the summation of the active energy consumption during computation, ideal energy consumption and data transmission energy consumption.

$$E_{\mathrm{total}} = \mathrm{TEN_{active}} + \mathrm{TEN_{ideal}} + \mathrm{ENC_{active}} \tag{18}$$

In order to obtain more energy saving as well as maintain the quality of service by meeting the deadlines, this paper proposed a DVS-enabled technique for energy saving on multi-site. In this work, the sites are DVS enabled. As a result, when the site is in the ideal state, it is switched to the low voltage level using Table 2. Consequently, an energy consumption level further reduces and energy saving increases, as discussed in Sect. 4.3.

4 Application scheduling based on teaching learning-based optimization

This section presents the brief overview of problem definition for task scheduling based on TLBO Optimization.

4.1 Teaching learning-based optimization

TLBO algorithm is a teaching–Learning process inspired algorithm proposed in Rao et al. (2011, 2012), based on the effect of the influence of a Teacher on the output of Learner in class. Teacher and learner are the two key components of the algorithm and describe two basic modes of the learning, through Teacher (known as Teacher phase) and through interaction with the other Learner (known as Learner phase). The output in TLBO algorithm is considered in terms of results of the Learner which depend on the quality of Teaching. Moreover, Learner can learn from the interaction among them which helps in improving their results. TLBO is a population-based method. In this optimization algorithm, a group of Learners is considered as population and different design variables are considered as different subjects offered to the Learner and Learner's result is analogous to the "fitness" value of the optimization problem. In the entire population, the best solution is considered as the Teacher. The working of TLBO is divided into two parts: Teacher phase and Learner phase.

4.1.1 Implementation of TLBO-based task scheduling technique

The teaching learning-based optimization algorithm consists of two Learner modes: Teacher phase and Learner phase. Before the initialization of the algorithm, a group of Learners is considered as a population and different parameters offered to the Learner are considered as different design variables of the optimization problem and a Learner's result is compared to the 'fitness' value of the optimization problem. The best solution in the entire population is considered as the Teacher, and the best solution is the best value of the objective function.

$$\begin{aligned}&\text{Minimize}\, f(X)\\&\text{Subject to } x_i \in x_i = 1,\ 2,\ \ldots,\ D_n\end{aligned}$$

where $f\ (X)$ is the fitness function, (D_n) is number of design variables, and upper limit and lower limit of design variables $(\text{UL},\ \text{LL},\ )$ X is a vector for design variables such that $\text{LL} \ \le x_i\ \le\ \text{UL}$.

The fitness function used in the proposed work is presented in Eq. (19)

$$F\ (X) = b * \text{Time} + (1 - b) * \text{Cost} \tag{19}$$

where time is the total execution time of the scheduled application and is given by Eq. (20), where cost is the total execution cost of the scheduled application and is calculated using Eq. (21) and b is the cost-time balance factor in a range of [0–1] which represents the trade-off between the time and cost savings.

$$\text{Time} =\ \text{TET}_m\ + \text{TET}_c \tag{20}$$

$$\text{Cost} = \text{TEC} \tag{21}$$

Teaching phase During this phase, a Teacher tries to increase the mean result of the given population. At any iteration i, assume that there are 'm' number of parameters (i.e., design variables), 'n' number of Learner (i.e., population size, $p = 1, 2, \ldots, n$) and $M_{j,i}$ be the mean result of the Learner in a particular subject 'j' ($j = 1, 2, \ldots, m$). The best overall result $X_{\text{total}-k\text{best},\, i}$ considering all the parameters together obtained in the entire population of Learner can be considered as the result of best Learnerk_{best}. The difference between the existing mean result of each parameter and the corresponding result of the Teaching for each subject is given by,

$$\text{Difference\_Mean}_{j,k,i} = r_i\left(X_{j,k\text{best},i} - T_{\text{F}} * M_{j,i}\right),$$

where, $X_{j,k\text{best},i}$ is the result of the best Learner (i.e., Teaching) in subject j. T_{F} is the teaching factor which decides the value of mean to be changed, and r_i is the random number in the range [0, 1]. The value of T_{F} is decided randomly with equal probability as,

$$T_{\text{F}} =\ \text{round}\ [1 + \text{rand}\ (0, 1)]\{2 - 1\}] \tag{22}$$

T_{F} is the teaching factor and can have a value of either 1 or 2. It represents the skill value of a Teaching and decides the value of mean to be changed. T_{F} can be randomly decided with the equal probability of having a value of 1 or 2 as shown in Eq. (22). Based on the Difference_Mean$_{j,k,i}$, the existing solution is updated in the Teaching phase according to the following expression.

$$X'_{j,k,i} = X_{j,k,i} + \text{Difference\_Mean}_{j,k,i},$$

where $X'_{j,k,i}$ is the updated value of $X_{j,k,i}$. Accept $X'_{j,k,i}$ if it gives better function value. All the accepted function values at the end of the Teacher phase are maintained, and these values become the input to the Learner phase.

Learner phase Learners increase their knowledge by interaction among themselves. A Learner interacts randomly with other Learner for enhancing his or her knowledge. Considered the result of the Teacher phase, the table is updated by the interaction between the Learners if it generates the better result than the previous values for the objective function. The interaction between the Learners is defined as

$$\begin{aligned}X''_{j,a,i} &= X'_{j,a,i} + \text{ri}\ (X'_{j,a,i} - X'_{j,b,i}), \quad \text{If } X'\text{total} - a, i < X'\text{total} - b, i\\ X''_{j,a,i} &= X'_{j,a,i} + \text{ri}\ (X'_{j,b,i} - X'_{j,a,i}), \quad \text{If } X'\text{total} - b, i < X'\text{total} - a, i\end{aligned}$$

Randomly select two Learners a and b such that $X'\text{total} - a, i\ \neq\ X'\text{total}-b, i$ Accept $X''_{j,a,i}$ if it gives a better function value. The flowchart for the above mentioned procedure is shown in Fig. 1.

4.2 Time efficient multi-site computation offloading

This section discusses the proposed work based on the minimization of the objective function represented in Eq. (23). We considered two conflicting objectives, i.e., minimization of execution time, (time) and minimization of total cost,(cost) of the created schedule. Therefore, scheduling problem is derived as

$$\begin{aligned}&\text{Min (Time, Cost)}\\&\text{Subject to}\\&\text{Time} <\ T_d \text{ and Cost } < B\end{aligned} \tag{23}$$

where T_d is the time constraint required by the users for the application execution and B is the cost constraint (Budget). To show the impact of the objective function as presented in Eq. (23) in the proposed technique, CTTPO algorithm discusses the application partitioning and offloading and MTS algorithm discusses the TLBO-based task scheduling on multi-site environment. CTTPO algorithm deals with the decision making for partitioning of the application modules to be executed on the cloud side or on the mobile device side.

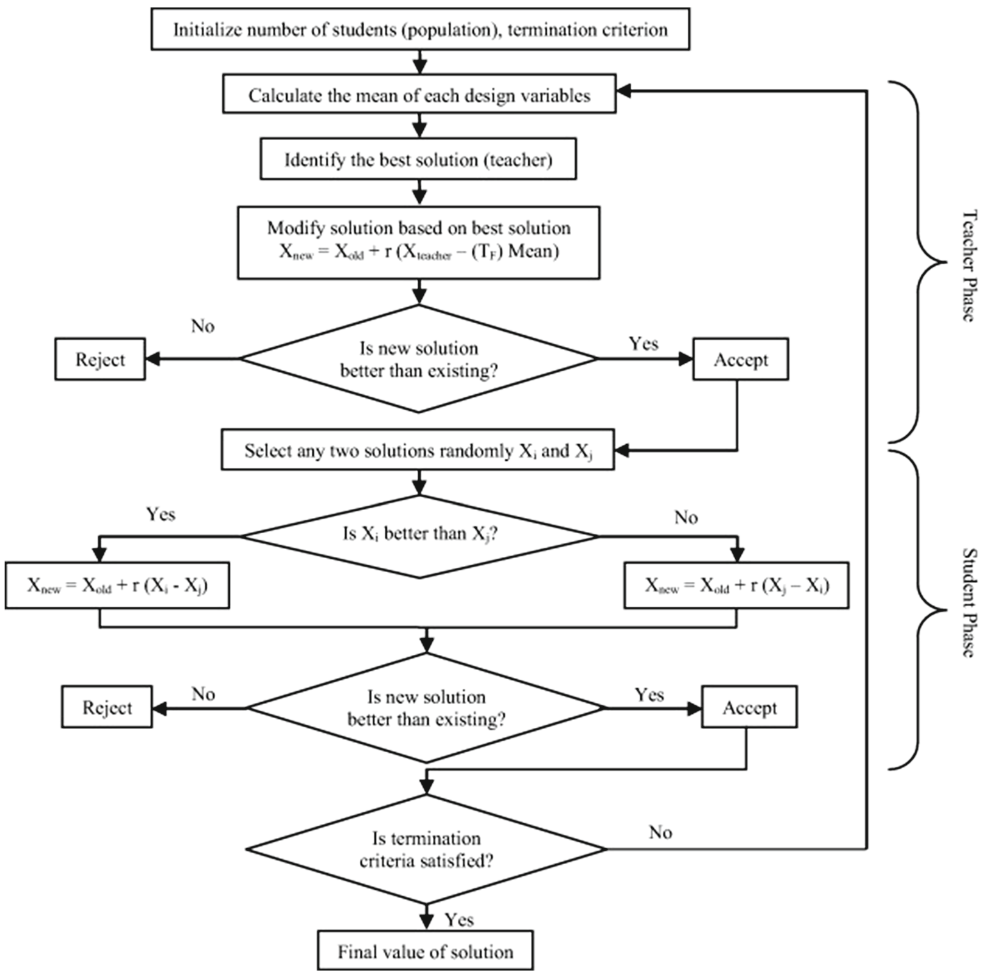

Initialize number of students (population), termination criterion
Calculate the mean of each design variables
Identify the best solution (teacher)
Modify solution based on best solution $X_{new} = X_{old} + r (X_{teacher} - (T_F) Mean)$
Is new solution better than existing?
No
Reject
Yes
Accept
Select any two solutions randomly X_i and X_j
Is X_i better than X_j?
Yes
No
$X_{new} = X_{old} + r (X_i - X_j)$
$X_{new} = X_{old} + r (X_j - X_i)$
Is new solution better than existing?
No
Reject
Yes
Accept
Is termination criteria satisfied?
No
Yes
Final value of solution
Teacher Phase
Student Phase

Fig. 1 Flowchart of proposed TLBO algorithm

After taking the decision, MTS algorithm generates the optimized time efficient schedule for the multi-site environment.

4.3 Energy efficiency of computation offloading in MCC

As discussed in Sect. 3, the energy consumption of offloaded module on their respective sites is calculated using Eq. (15). The total power consumption of the application is the summation of the energy consumed during the transmission of the data and energy consumed by the processing of the offloaded module on the selected site. In order to obtain more energy reduction as well as maintain the quality of service by meeting the deadlines, this paper has used the concept of DVS-enabled sites to achieve the energy saving on multisites. The offloaded modules on cloud sites can operate under the lowest voltage using DVS technique in the idle slots, without violating the dependency constraints.

To make the energy consumed by the modules on the sites, as low as possible, the DVS technique is used as discussed in Sect. 3. The main steps followed in minimization of execution time and cost are described in Figs. 2 and 3, respectively. Figure 4 presents the algorithm for the minimization of energy consumption. The feasibility of the objectives is only if cost $<$ Budget and Time $<$ task deadline, otherwise it is considered as infeasible. ESM algorithm deals with the energy saving on multi-sites in mobile cloud computing environment.

For the implementation of CTTPO and MTS algorithms pseudocode is explained in Fig. 5.

5 Performance evaluation

In this section, the effectiveness of the proposed algorithms, *namely*, cost and time constraint task partitioning and offloading algorithm (CTTPO), multi-site task scheduling algorithm (MTS) and energy saving on multi-sites algorithm (ESM) are evaluated and the comparison is done with a state-of-art algorithm, multi-site particle swarm optimization algorithm

```
CTTPO Algorithm: application partition and offloading
Input: application with $k$ modules, modules for mobile processing, site p rocessing speed, mobile
device processing speed
Output: optimised execution time and cost
Step I: Task graph is partitioned according to the eq. (6) and eq.(7)
Step II: Calculate execution time and cost
        a) For the execution of a partitioned module on the mobile side, eq. (4) is used to calculate
           the execution time
        b) Call MTS algorithm for offloaded modules.
Step III: complete execution time and cost generated.
End
```

Fig. 2 Task partitioning and offloading algorithm

```
MTS Algorithm: TLBO based task scheduling on multiple sites
Input: offloaded modules, site processing speed
Output: minimized execution time
Process:
Initially random task assignment to the site
    i.    With initial assignments, fitness function (eq. (19)) is calculated for the evaluation of
          objective function (eq. (23))
    ii.   For eq. (19) set the value of $b$ based on the tradeoff between time and cost
    iii.  Using eq. (20) and eq. (21) the time and cost is calculate
    iv.   Calculate the mean values column wise with respect to each assignment of the site
    v.    Teaching Phase: Minimum mean value in step (iii) is selected as the best fitness function
          value according to eq. 19 (best solution or Teaching) and this selected value is compared
          next time with the new updated table value
              % Step v is the process of selection of best Teaching %
    vi.   Updated table with optimized objective function (eq. (23)) is further processed to minimize
          the objective function with the interaction of the modules with other modules.
              % passed as input to the Learner phase%
    vii.  Learner phase: Interaction between the updated table is done row-wise
              a. calculation of objective function
              b. new value of the objective function is compared with the previous value
              c. if new fitness function value <= previous fitness function value using eq. (19)
              d. update the site values with respect to the fitness function value
              e. otherwise keep the previous value
    viii. step v and vi continues till the optimized result value of time or cost becomes stable.
    ix.   return optimized execution time
```

Fig. 3 TLBO-based Multi-site Task scheduling algorithm

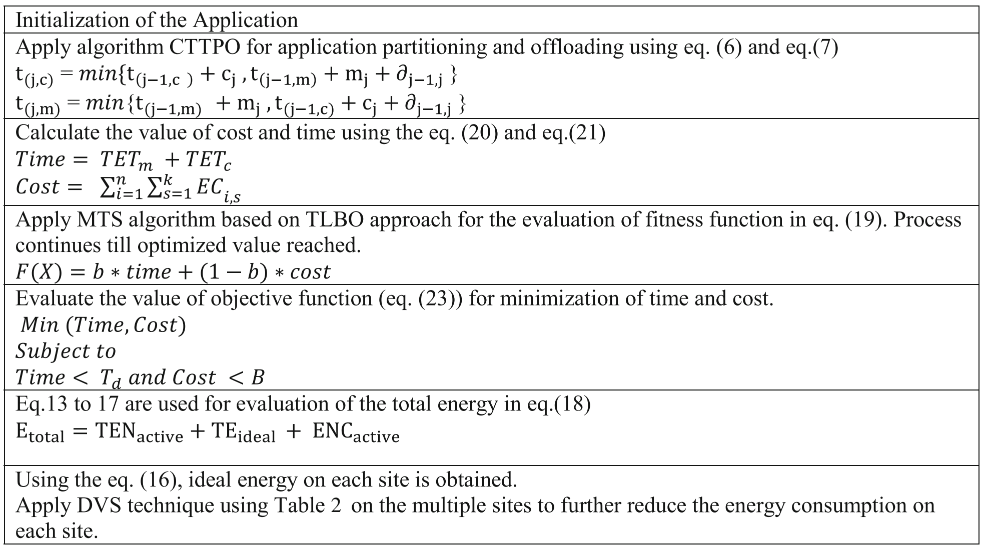
```
ESM: Energy saving in multi-site environment algorithm
Input: output of MTS algorithm
Output: minimum energy consumption
Process:
    i.   Optimized time of the offloaded task on different sites is obtained using MTS algorithm
    ii.  Calculate the total energy consumption of the application using the eq. (18)
    iii. Apply the DVS technique for the further minimization of energy consumption.
    iv.  Exit
```

Fig. 4 Energy saving on multi-sites

(OMPSO). To measure the performance of proposed algorithms, two random graphs R1 and R2 as specified in Table 3 are used. The different random graphs R1 and R2 with various number of vertices and edges are generated. R1 is light computation intensive graph, and R2 is a heavy computation intensive graph.

5.1 Experimental setup

We have assumed a set of four execution sites, including one mobile device and three sites with different computation resources, processing speed and price on the cloud for simulation study and MATLAB (Version 7.11 R2015b), and 64-bit machine is used. For data transmission between the mobile device and the multi-site, bandwidth is varied between 10



```
Initialization of the Application
Apply algorithm CTTPO for application partitioning and offloading using eq. (6) and eq.(7)
    $t_{(j,c)} = min\{t_{(j-1,c)} + c_j, t_{(j-1,m)} + m_j + \partial_{j-1,j}\}$
    $t_{(j,m)} = min\{t_{(j-1,m)} + m_j, t_{(j-1,c)} + c_j + \partial_{j-1,j}\}$
Calculate the value of cost and time using the eq. (20) and eq.(21)
    $Time = TET_m + TET_c$
    $Cost = \sum_{i=1}^{n} \sum_{s=1}^{k} EC_{i,s}$
Apply MTS algorithm based on TLBO approach for the evaluation of fitness function in eq. (19). Process
continues till optimized value reached.
    $F(X) = b * time + (1 - b) * cost$
Evaluate the value of objective function (eq. (23)) for minimization of time and cost.
    $Min\ (Time, Cost)$
    $Subject\ to$
    $Time < T_d\ and\ Cost < B$
Eq.13 to 17 are used for evaluation of the total energy in eq.(18)
    $E_{total} = TEN_{active} + TE_{ideal} + ENC_{active}$
Using the eq. (16), ideal energy on each site is obtained.
Apply DVS technique using Table 2 on the multiple sites to further reduce the energy consumption on
each site.
```

Fig. 5 Pseudocode for the implementation

Table 3 Graph specifications

Random graph	Node	Edges
R1	8	25
R2	20	110

and 100 Kbytes/s to represent the impact of different data rates on offloading decision. Other parameters are listed in Table 1. All cloud resources are dynamic voltage scaling (DVS) enabled, and in other words, it can operate with different voltage scaling Levels (VSLs), at different clock frequencies. For each resource, a set V_j Of v VSLs is random and uniformly distributed among three different sets of VSLs as shown in Table 2, with the assumption that when resources are busy then they are operating at a maximum voltage scaling level and during idle state, their voltage level drops to the minimum scale.

5.2 Performance metrics

The following performance metrics are used for analyzing the performance of the proposed work for different values of deadline and budget:

(a) Execution time: Among the number of sites, using CTTPO and MTS algorithms, execution time on each site is calculated and the summation of mobile side execution time and cloud side execution time is the execution time of the application.
(b) Energy consumption: concerning the execution time of the offloaded module on their respective sites is multiplied by the maximum level of the power supply as per Table 2. Summation of the energy consumption by the modules on their respective sites is the overall energy consumption of the application.
(c) Energy saving: using the DVS enabled sites, during the computation time site are working with high voltage level to give fastest response time. And when the site is in an ideal state, i.e., during ideal time, sites are switched to the lowest voltage level.

5.3 Results

This section presents simulation results and analysis of proposed algorithms and its comparison with an existing technique. All the experiments are executed 30 times so that the best average minimum value can be achieved. Experimental work is performed on the configuration set C1, C2 and C3 where PS is population size, I is iteration numbers and B is bandwidth as described in Table 4.

5.3.1 Impact of time and cost on population size, iteration numbers and balancing factor b

To measure the impact of varying population size and balancing factor, *b* on execution time and cost, configuration set C1 is considered. Figure 6 shows that execution time improves with CTTPO and MTS algorithms as compared to an OMPSO algorithm with increasing in population size. In case of OMPSO, the best fitness value is observed till population size 50, whereas, in case of CTTPO with MTS algorithms, the best fitness value is observed till population size 30 and no significant improvement is observed after achieving best fitness value. In Fig. 7, the impact of total execution cost with respect to population size is depicted.

Table 4 Parameters configuration

Configuration	PS	I	B	Execution sites
C1	[10–100]	30	100	4
C2	50	[10–100]	100	4
C3	50	30	[10–100]	4

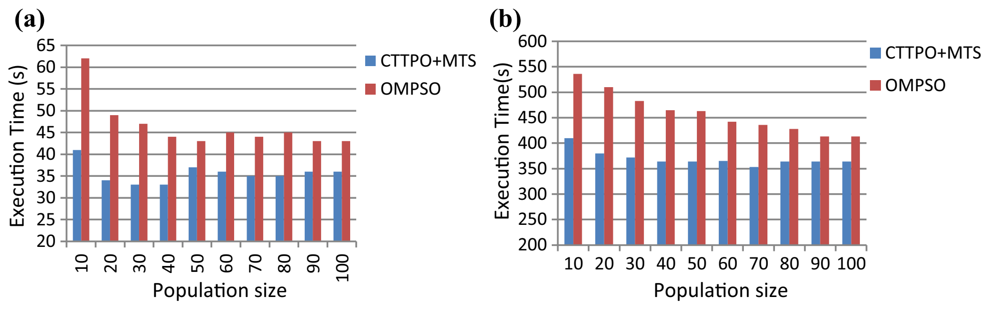


Fig. 6 Impact of Population size on Execution time. **a** Execution time of R1 graph. **b** Execution time of R2 graph

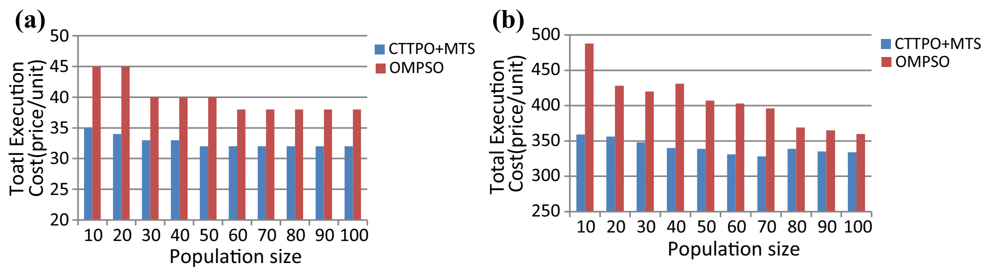


Fig. 7 Impact of Cost on population size. **a** Cost comparison of R1 graph. **b** Cost comparison of R2 graph

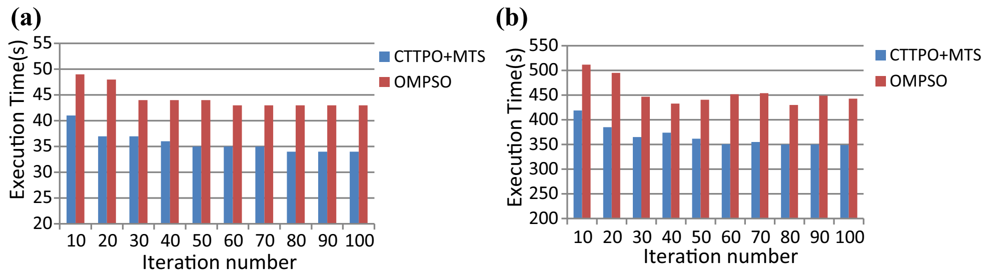


Fig. 8 Impact of iteration numbers on Execution time. **a** Execution time comparison of R1 graph. **b** Execution time comparison of R2 graph

Results show the total execution cost in CTTPO and MTS algorithms is less as compared to the OMPSO algorithm. Similarly, in Fig. 8, CTTPO and MTS algorithms generate better results for execution time as compared to the OMPSO algorithm with parameter iteration numbers, considered configuration set C2. In case of OMPSO algorithm, up to $I = 30$, it generates the best fitness value whereas in CTTPO and MTS algorithm obtains the best fitness value is when I is between 20 and 30. In this work, the fitness function takes the offloading decision on the basis of trade-off between time and cost. The value of balancing factor b is adjusted according to the user requirements. When the user set $b = 0.8$, then the user gives more preference to minimize the total execution time as compared to minimize the total execution cost (according to Eq. (19)). As the value of b decreases, the user preference for minimizing the total cost increases. Figure 9 shows the impact of b value on the fitness value minimization. In OMPSO algorithm, the authors have proposed a solution to achieve a near optimal offloading solution. OMPSO algorithm requires the setting of basic algorithm-specific parameters such as inertia weight, learning coefficients whereas in TLBO algorithm only basic common control parameters are required such as population size, iteration numbers. Therefore, our proposed

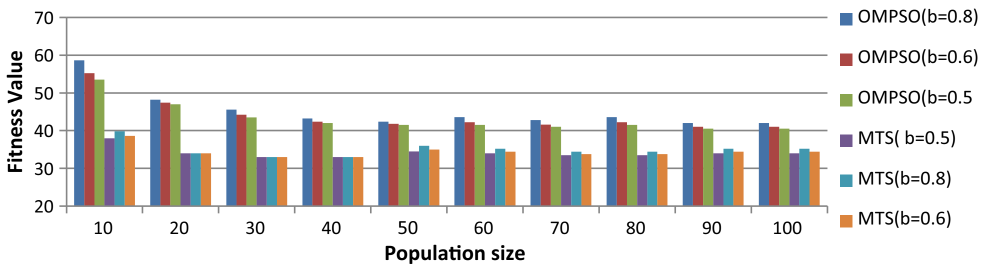


Fig. 9 Impact of *b* value on the fitness value minimization

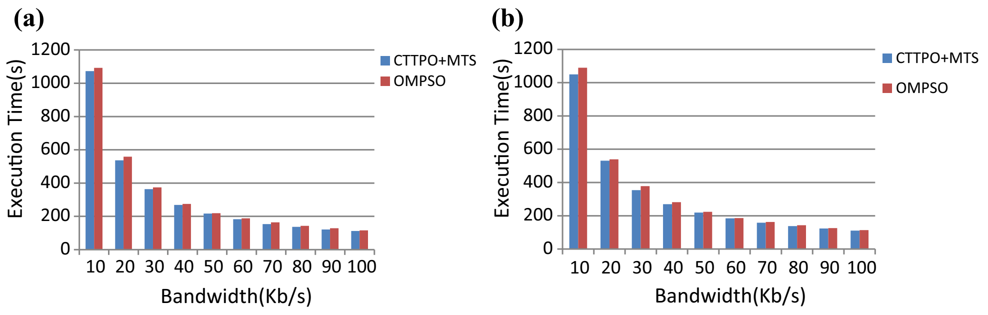


Fig. 10 Impact of different bandwidths on total execution time. **a** Execution time of R1 graph. **b** Execution time of R2 graph

work obtains an effective global solution for scheduling in the multi-site environment by involving less computational efforts in comparison to other state-of-the-art algorithm.

5.3.2 Impact of dynamic network conditions

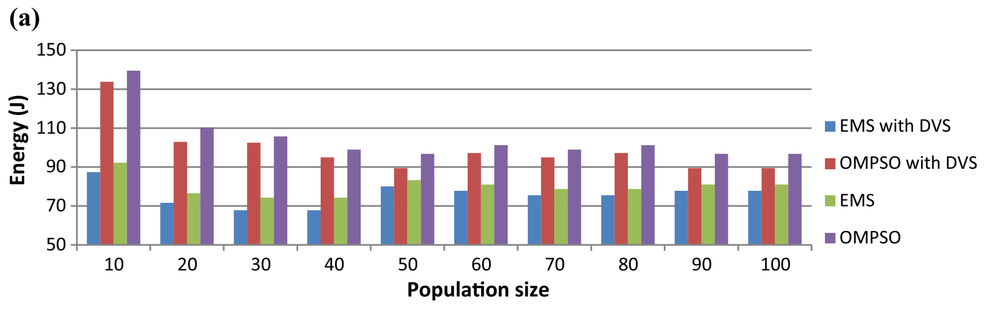
The dynamic behavior of the mobile network plays an important role when determining whether offloading a certain task from the mobile device to the cloud would be beneficiary to the application user. Poor network condition would result in more retransmission, which adds to the application response time and energy consumption. Figure 10 shows the bandwidth impact on graphs R1 and R2 with configuration set C3. The results show fast transmission rate with better bandwidth availability in the CTTPO and MTS algorithm as compared to the OMPSO algorithm. CTTPO algorithm helps to obtain the appropriate partitioning, which helps the MTS algorithm to generate the overall minimized execution time with best fitness value.

5.3.3 Impact of energy consumption

Figure 11 shows the comparison of energy consumption for graphs R1 and R2 by applying configuration set C1. The mobile device consumes maximum energy when it performs computation intensive tasks. Therefore, the offloading of computation intensive tasks over the cloud gives better results. But offloading is not always beneficial. Results show more energy saving in proposed EMS algorithm as compared to the OMPSO algorithm. The result also shows the comparison of energy consumption, saving with DVS technique. To make the system energy conscious not only for the mobile side, but also on the cloud side, the concept of DVS technique is implemented. When sites are under execution mode, they operate at the highest voltage level, whereas when sites are under ideal state, it switches to the low voltage level. With the use of DVS technique, it saves more energy consumption by switching the sites on the lowest power mode when not in use. From all results obtained, it is proven that the proposed scheme is able to reduce the overall execution time, cost and saves energy consumption.

6 Conclusion

In MCC environment, the data offloading technique improve the execution time as well as saves energy of the mobile devices in an efficient manner. In this paper, a multi-objective technique for task scheduling considering the time, cost and energy parameters under deadline and budget constraints is proposed. For this purpose, a trade-off between time and cost for offloading in multi-site environment is considered. In the first step as per the user preferences with respect to time or cost parameters, the offloading decision is taken. Using CTTPO algorithm, task partitioning is done, which gives the offloadable and non-offloadable modules of the application. To schedule the offloadable module among the multiple sites, MTS algorithm is used to generate the time optimized efficient scheduling. Depending on the user preference, the value of balancing factor *b* is adjusted for the fitness function. In the



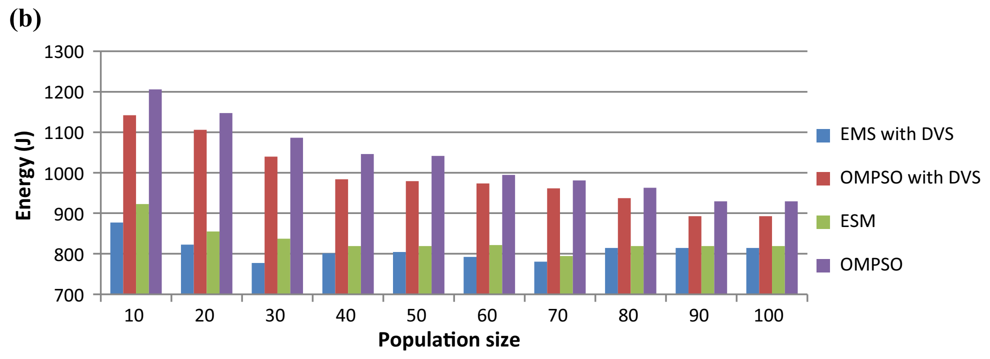


Fig. 11 Impact of energy consumption using population size. **a** Energy consumption of R1 graph. **b** Energy consumption of R2 graph

second step, energy optimization is done for multi-site environment using ESM algorithm. Using DVS technique, the energy reduction is achieved on the different sites by switching the sites on the lowest voltage level during idle times. The experimental work shows that the our proposed CTTPO, MTS and ESM algorithms achieve better performance in saving execution time and energy on the cloud as well as on a mobile device with multiple site execution in comparison to the other state-of-the-art algorithms. In the future, we will extend this work further to consider the other QoS constraints like reliability, fault tolerance, VM migration, etc., for reducing overall energy consumption. The concept of latest telecommunication technologies like 5G can also be considered for data transmission to extend the battery life of the mobile devices.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Ahmed E, Naveed A, Gani A, Hamid SHA, Imran M, Guizani M (2017) Process state synchronization for mobility support in mobile cloud computing. In: IEEE international conference on communications, pp 1–6

Bheda HA, Lakhani J (2013) Application processing approach for smart mobile devices in mobile cloud computing. Int J Adv Res Comput Sci Softw Eng 3(8):1046–1054

Deng S, Huang L, Taheri J, Zomaya AY (2014) Computation offloading for service workflow in mobile cloud computing. IEEE Trans Parallel Distrib Syst 26:3317–3329

Dinh HT, Lee C, Niyato D, Wang P (2013) A survey of mobile cloud computing: architecture, applications, and approaches. Wirel Commun Mob Comput 13(18):1587–1611

Gajbhe M, Sakhare SR (2015) Performance augmentation of mobile devices using cloud computing. Int J Comput Sci Inf Technol, 3581–3587

Goudarzi M, Zamani M, Haghighat AT (2017) A fast hybrid multi-site computation offloading for mobile cloud computing. J Netw Comput Appl 80:219–231

http://www.techrepublic.com/resource-library/whitepapers/765resource-constrained-multi-user-computation-partitioning-for-766interactive-mobile-cloud-applications/ . Accessed 10 Sept 2017 767

Islam M, Islam J (2016) A genetic algorithm for virtual machine migration in heterogeneous mobile cloud computing. In: 2016 International conference on networking systems and security (NSysS), Dhaka, pp 1–6, Jan 7–9

Kahng AB, Kang RKS (2013) Enhancing the efficiency of energy constrained DVFS designs. IEEE Trans Very Large Scale Integr Syst 21(10):1769–1782

Khanghani N, Ravanmehr R (2013) Cloud computing performance evaluation: issues and challenges. Comput 5(1):29–41

Kumar K, Lu YH (2010) Cloud computing for mobile users: can offloading computation save energy? Computer 43(4):51–56

Kumar K, Liu J, Lu Y, Bhargava B (2012) "A survey of computation offloading for mobile systems." In *Mobile Networks and Applications*, pp. 1–12

Nguyen AD, Senac P, Ramiro V (2011) How mobility increases mobile cloud computing processing capacity. In: 1st International symposium on network cloud computing and applications (NCCA), pp 50–55

Pandit D, Chattopadhyay S, Chattopadhyay M, Chaki N (2014) Resource allocation in cloud using simulated annealing. In: 2014

Applications and innovations in mobile computing (AIMoC), IEEE, pp 21–27

Park S, Park J, Shin D, Wang Y, Xie Q, Chang N, Pedram M (2013) Accurate modeling of the delay and energy overhead of dynamic voltage and frequency scaling in modern microprocessors. In: IEEE T.CAD, pp 695–708

Rao RV, Savsani VJ, Vakharia DP (2011) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315

Rao RV, Savsani VJ, Vakharia DP (2012) Teaching-learning-based optimization: an optimization method fo continuous nonlinear large scale problems. Inf Sci 183(1):1–15

Shaukat U, Ahmed E, Anwar Z, Xia F (2016) Cloudlet deployment in local wireless networks: motivation, architectures, applications, and open challenges. Int J Netw Comput Appl 62:18–40

Shiraz Muhammad, Gani Abdullah, Ahmad Raja Wasim, Shah Syed Adeel Ali, Karim Ahmad, Rahman Zulkanain Abdul (2012) "A Lightweight Distributed Framework for Computational Offloading in Mobile Cloud Computing " In Computer Science and Automation Engineering (CSAE). IEEE International Conference. 1:89–93

Shiraz M, Ahmed E, Gani A, Han Q (2014) Investigation on runtime partitioning of elastic mobile applications for mobile cloud computing. J Supercomput 67(1):84–103

Shu P, Liu F, Jin H (2013) eTime: Energy efficient transmission between cloud and mobile devices. In: INFOCOM, IEEE, pp 195–199

Sinha K, Kulkarni M (2011) Techniques for fine-grained, multisite computation offloading. In: Proceedings of the 2011 11th IEEE/ACM international symposium on cluster, cloud and grid computing (CCGrid' 11), pp 184–194, May

Terefe MB, Lee H, Heo N, Fox GC, Oh S (2016) Energy-efficient multisite offloading policy using Markov decision process for mobile cloud computing. Pervasive Mob Comput 27:75–89

Wang L, von Laszewski G, Dayal J, Wang F (2010) Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with DVFS. In: Proceedings of CCGrid'2010, the 10th IEEE/ACM international conference on cluster, cloud and grid computing, pp 368 –377

Wu H, Huang D (2014) Modeling multi-factor multi-site risk-based offloading for mobile cloud computing. In: 2014 10th International conference on network and service management (CNSM), IEEE, pp 230–235

Wu H, Knottenbelt W, Wolter K, Sun Y (2016) An optimal offloading partitioning algorithm in mobile cloud computing. In: International conference on quantitative evaluation of systems,(LNCS, vol 9826), pp 311–328

Wu H, Wolter K (2014) Dynamic transmission scheduling and link selection in mobile cloud computing. In: Analytical and stochastic modeling techniques and applications, Springer, Berlin, pp 61–79

Xiang X, Lin C, Chen X (2014) Energy-efficient link selection and transmission scheduling in mobile cloud computing. IEEE Wirel Commun Lett 3(2):153–156

Yaqoob I, Ahmed E, Gani A, Mokhtar S, Imran M, Guizani S (2016) Mobile ad hoc cloud: a survey. Wireless Commun Mob Comput 16(16):2572–2589

Ying F, Lei G (2014) Optimal scheduling simulation of software for multi-tenant in cloud computing environment. In: 2014 Fifth international conference on intelligent systems design and engineering applications (ISDEA), pp 688–692, June

Zhang K, Mao Y, Leng S, Zhao Q, Li L, Peng X, Pan L, Maharjan S, Zhang Y (2016) Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks. IEEE Access 4:5896–5907

Zhang L, Fu D, Liu J, Ngai ECH, Zhu W (2017) On energy-efficient offloading in mobile cloud for real-time video applications. IEEE Trans Circuits Syst Video Technol 27(1):170–181

Zong Z, Nijim M, Manzanares A, Qin X (2008) Energy efficient scheduling for parallel applications on mobile clusters. Cluster Comput 11(1):91–113