Sādhanā (2020)45:177
https://doi.org/10.1007/s12046-020-01414-1

CASCM^2: Capability-Aware Supply Chain Management Model for QoS-driven offload-participator selection in Fog environments

A J SHALINI LAKSHMI* and M VIJAYALAKSHMI

Department of Information Science and Technology, College of Engineering, Anna University, Chennai, India
e-mail: ajshalini2020@gmail.com; vijim@auist.net

MS received 24 November 2019; revised 29 March 2020; accepted 31 March 2020

Abstract. Computational offloading happens to be a prominent solution for leveraging the performance of handheld devices. It raises the feasibility of executing computation-intensive and latency-conscious tasks with the help of task migration to proximate cloud servers. However, longer Wide Area Network latencies of cloud and greater mobile data consumption paved the way to adopting opportunistic offloading instead of remote offloading. This proposed work uses an edge-based solution of Fog computing to handle such tasks and to provide the users with a high Quality of Experience. This paper presents a Capability-Aware Supply Chain Management Model (CASCM^2) as an extension of traditional Supply Chain Management (SCM) model. CASCM^2 dynamically selects a crowd of competent mobile devices within a Foglet that is in proximity to the users and offloads the complex computational tasks to them. The proposed model aims at optimizing two parameters, such as communication overhead and conductance cost, as they possess a remarkable impact on offloading delay-sensitive tasks. Hence an overall objective optimization is achieved using a dual Lagrangian decomposition method, which subdivides and solves the optimization of parameters in parallel. Experimental analysis of the participator selection is performed for a single period as well as multiple periods. The performance results yield a considerable contribution that alleviates the issues in delay-sensitive applications deployed in the Fog framework.

Keywords. Capability-Aware Supply Chain Management Model (CASCM^2); Fog computing; communication overhead; conductance cost; dual Lagrangian decomposition.

1. Introduction

The developments in designing applications for smartphones and tablets are highly increasing. In recent times, the usage of smartphones has become inevitable amongst the people of all professions. With the need to adopting recent smartphone technologies, these mobile devices face serious processing and resource limitations [1] on top of the residual battery drain [2]. Hence, to cope up with these issues, mobile cloud computing (MCC) evolved. However, WAN latencies between cloud resource and users [3] become a challenge for delay-sensitive applications. It has led to the transformation of remote offloading to opportunistic offloading [4].

Opportunistic offloading involves an edge cloud formed from an amalgamation of heterogeneous mobile devices instead of predefined virtual machines (VMs). Edge cloud undergoes a prediction process to act as surrogates to the cloudlets deployed in a static cloud. This edge-based cloud requires a negligible cost for implementing it as it can be deployed with a minimum configuration in university campuses, houses and even in smaller workplaces [5]. They also provide a warranty for high availability and satisfies the elasticity property of the cloud [4]. However, an exclusive selection of these proximate devices is required to ensure a higher success rate of offloading.

Offloading is initialized based on a set of decisions taken by the source device dynamically. Such an initialization includes continuous observation of devices and network parameters over different time periods. This watchdog tasks to search for a suitable receiver add up to the overhead, which in turn might affect the execution time of the application. Also, the performance gain from the offloading is accomplished only when the offload execution time stays below the local execution time [6]. Taking all these factors into account, a Capability-Aware Supply Chain Management Model (CASCM^2) is proposed, which reduces the additional overhead and affords a profitable offloading by shortening the application execution time.

Supply Chain Management (SCM) is a popular inventory management solution used in fulfilling the customer

*For correspondence
Published online: 10 July 2020

demands in a dynamic environment considering the resource limitations in mind [7]. It has been applied in cloud computing to identify the right cloud resource for various user's expectations like cost reduction and energy considerations [8]. SCM considers the collaborative sharing of the task as a strategic response to the problems arising from the organizational side [9]. In cloud computing, the SCM model helps in understanding interactions between the nodes and their environment behaviours in the network [10].

This work focuses on modelling a decisive Fog framework by integrating SCM model with the discovery of resource-rich Fog nodes and offloading the tasks of an application to them within a Foglet as per the user's demand. A static cloudlet available at one hop distance may help overcome network latency problem of clouds [11]. Even though the reduction of latency is achieved, a much better solution with minimum latency possible to this issue is the migration of the application task to its next level Fog server. Fog server can be an access point, switch or router located in the next hop to the source node, whereas the cloudlet will be in more than one hop distance to the source connected through an access point or router [12].

There are various techniques like AHP, ANP, MILP [13] and their extended versions that solve the multi-objective optimization model. In CASCM^2, the Lagrangian technique is used as a distinguished solution to solve the dual objective model. It optimizes two QoS parameters such as conductance cost and communication overhead using the dual Lagrangian decomposition method. The parameter "conductance" implies the processing power of the VMs [14]. In this proposed work, conductance cost refers to the combination of the task processing cost and the data transportation cost incurred while offloading the computation-intensive tasks to the Fog nodes in the Fog framework. Communication overhead comes as the second contributor to the additional cost due to offloading. The offload decision-making cost comprises an efficient Fog-node discovery, offload-threshold adjustment, optimization and sorting costs.

For delay-sensitive applications like video streaming and AR/VR applications, the main objective is to minimize its processing cost as much as possible to provide the customer with a high Quality of Experience (QoE). Generally, computational offloading is opted to achieve QoE for such applications. It involves local offload decision to be taken at the mobile device to predict the need for offloading. Moreover, the overhead to arrive at the offload decision and the resource allocation process have a significant impact on the task completion time. Thus, CASCM^2 aims to minimize the conductance cost and the communication overhead to achieve a profitable offloading. Minimization of these two parameters is the ultimate goal of this proposed work.

1.1 *Components of CASCM^2*

An overview of the proposed CASCM^2 is depicted in figure 1. This Fog-based cluster includes three components: (1) the device that initiates the offloading called "owner", which resembles supplier in traditional SCM; (2) the devices that join the owner's network and form a mobile cloud called "participators", complementary to facilitator in SCM and (3) the device that is chosen to help in finding offload-fit participators based on the demand of the owners and their network parameters across several time periods called "advisors", similar to retailers in SCM.

1.2 *Layered architecture of CASCM^2*

The proposed CASCM^2 model comprises four layers: cloud layer, advisor layer, participator layer and application-owner layer, as shown in figure 2. A node in the advisor layer acts as a Fog server. It is responsible for handling the discovery of available participators, selection of offload-fit participator and scheduling of the tasks. It senses and maintains the heartbeat and profiles of the local heterogeneous networked devices across time periods. The participator layer is responsible for supervising scheduled tasks from the owner and sending back appropriate results to the owner. When an advisor itself moves out of the network before the end of an offload session, the second efficient device will be chosen as the advisor.

The remainder of this paper is organized as follows. Section 2 reviews the related works and defines the problem statement. A dual-objective SCM model is formulated, and a solution approach is given in section 3. Section 4

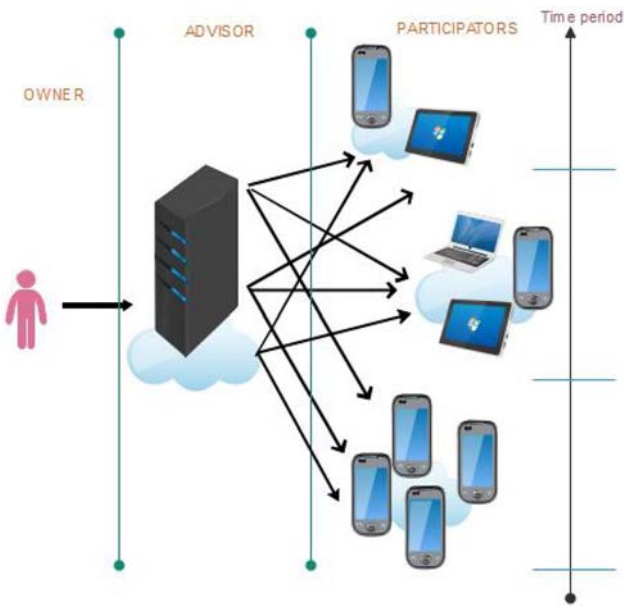

Figure 1. SCM representation of CASCM^2.

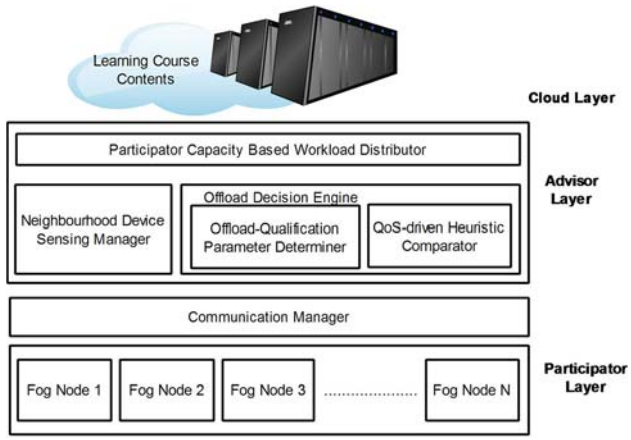

Figure 2. Layered Architecture of CASCM^2 for e-learning application.

elaborates the problem description for solving the proposed model. Section 5 gives an analysis of the experimental results. Closing remarks and conclusions are then outlined in section 6.

2. Related works

Mobile applications such as video streaming, gaming, AR/VR applications, traffic security, video surveillance, applications for vision and others are highly latency-sensitive. They require a cloud resource at the edge of the network rather than a distant central cloud for faster processing. Fog computing or "cloud at the edge" is definitely a significant solution to the delay-sensitive applications. It also provides mobile devices with ubiquitous and agile services [15, 16]. However, the computation and storage services of Fog computing are limited to a few numbers of users only. The emergence of Fog from MCC is the technology advancement of deploying cloud services in next hop to task owners to reduce the latency issue that prevails in MCC. A Fog can be deployed at the router or an access point, while the MCC can be deployed in the machines that are only in the next level to Fog [12].

Cloud Probing Service and Cloud Ranking [17] become a challenging task when the cloud resource to be chosen for offloading is dynamically done during runtime. Therefore, potential optimization is needed for finer and delay-less offloading. Lyapunov optimization is used by [18] to handle the randomness and unpredictable wireless connectivity between cloud and mobile devices. Resource scheduling is an NP-hard optimization problem that may be solved using a heuristic algorithm [19] or an optimization technique like Load balancing Ant Colony Optimization (ACO).

Queuing theory with three queuing models has been applied in [15], each for the mobile device, Fog and cloud centres to address the trade-off among energy consumption, delay and cost of offloading processes in a Fog framework. Parameters such as data rate and power consumption of the wireless link are taken into account. [20] optimizes the offloading decision and resource allocation by considering the parameters such as transmission power, network bandwidth, user fairness and transmission delay. The nature of the application and its requirements has an impact on the optimization of resource allocation while taking quality as a major constraint. Three different scheduling strategies have been proposed by [16] to improve the quality of service. Their deciding parameters are CPU load and execution delay at the cloudlets for different applications.

Game-theoretic approach for resource allocation problem with multiple parameters involved has polynomial time complexity. Its complexity increases with the increase in the number of participators. Thus, game theory would be unsatisfactory for the multi-objective optimization problems. The multi-attribute decision-making methods like TOPSIS, AHP and their variants possess high computational complexity. Their complexity increases with the size of the decision matrix as the number of participators in the Foglet increases [21]. The complexity of Lagrangian optimization is lesser than that of the algorithms as mentioned earlier, and it makes a better option for the environment with dynamic network characteristics. Standard models such as Round-Robin and FCFS have minimum complexity since the offload decision process does not associate an extensive logic behind it. Though their complexities are lower than that of CASCM^2 the time of remote execution carried out in the chosen participator may be high, providing the user with low QoE.

Lagrangian optimizations focus on the trade-off between conductance cost and communication overhead in CASCM^2. Generally, the optimizations are time-consuming and thus impact the execution time of an application. However, the results of optimization techniques are comparatively reliable compared with the other approaches. Since the Lagrangian optimization is local the time consumption is low to arrive at the solution, which is an essential condition to keep the overhead in the offload decision process to be minimum. Lagrangian optimization is an iterative process that solves several sub-problems in parallel within a particular time period [22]. Due to its parallel nature, the overall complexity of the proposed model will be reduced to a great extent. When optimization is constrained, the technique of Lagrange multipliers is one of the right choices among multi-objective optimizations [23].

2.1 *Contributions of the proposed work*

In this proposed model, multi-period refers to "Everlasting decision" in each period, which may fluctuate. This work adds to the following points of view.

- A participator CASCM^2 derived from the regular supply chain model has been proposed with an elemental presumption of a single period to a multi-period planning horizon in a dynamic Foglet environment.
- Offload-fit participators are chosen using a proactive QoS-driven selection among neighbourhood sensing devices in the Fog considering conductance cost and communication overhead as two decision parameters across multiple time periods using device heuristics.
- A dual objective Lagrangian decomposition method for minimizing QoS parameters plays an important role in offload decision. It results in overall objective optimization and yields a successful and profitable offloading.

2.2 *Overview of CASCM^2*

Consider a three-stage supply chain including owner, advisor and set of participators in a Foglet, as shown in figure 3. The dynamicity of participators has been taken into account for choosing them as surrogates to the owners. The higher the dynamism in network circumstances, the higher the cost of constructing an offloading framework and lower the processing time of the entire application. Therefore, a dual objective model should be built to (1) decrease the cost of processing the application in a changing network environment across several time periods and (2) reduce the communication overhead that arises due to offload-fit participator selection.

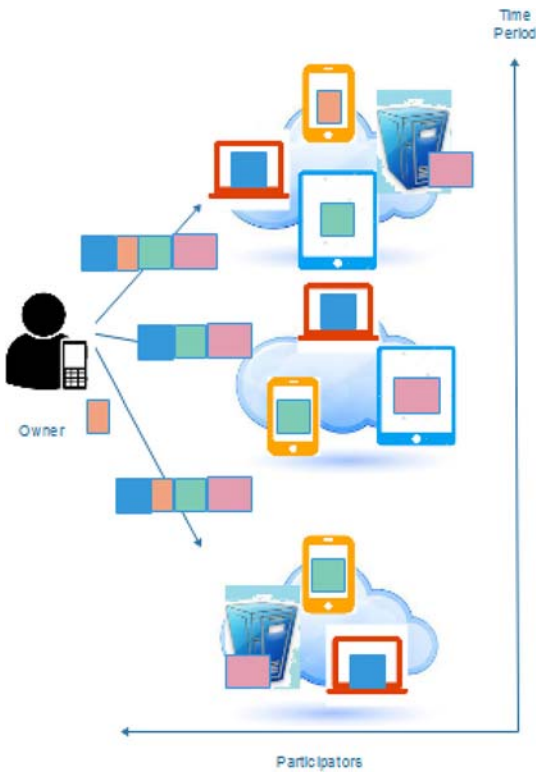

Figure 3. Supply Chain Management Model across time periods.

This work concentrates only on the selection of rich participators for our interest of convenience. It assumes that the computation-intensive tasks of application are known prior to offloading. The heuristic determination of QoS parameters such as communication overhead and conductance cost for every participator has been performed. The change of threshold in consecutive time periods is set in accordance with the dynamic environment.

When network parameters differ between two consecutive periods, an extra cost is incurred for choosing an alternative offload solution. The dynamically varying device and network factors can be treated as discrete values. Precisely, decision-making horizon is partitioned into multiple time periods. Different periods have different jobs of the application and different participators and therefore proposed CASCM^2 model varies in decision making.

3. Problem formulation and solution methodology

The proposed work focuses on offload-fit participator selection and makes the following assumptions: (1) the participator itself sends its availability signal to the Fog server and updates if it moves out of range; (2) the offloading follows non-preemptive job execution in each device; (3) the data transportation cost is calculated from the input size and the network bandwidth and (4) the jobs of an application once categorized as heavy computation job remain the same throughout the time period.

3.1 *Defining the dual objectives for offload-qualified participator selection*

Objective 1: To minimize the conductance cost of supply chain network. It is a combination of the task processing cost of an application in a network fluctuation scenario and the data transportation cost during offloading. These costs are calculated as follows.

(1) The transportation cost of tasks from owner o to participator p via advisor a is given as

$$\phi_t = \sum_{t\epsilon T, a\epsilon A, o\epsilon O} x_{oa} \sum_{p\epsilon P, j\epsilon J} e_j \gamma_{opajt} + \sum_{t\epsilon T, a\epsilon A, p\epsilon P} x_{ap} \sum_{o\epsilon O, p\epsilon P} e_j \gamma_{opajt}$$

where x_{oa} and x_{ap} are the cost of transporting task j from o to a and from a to p, respectively, with each device capability required by a unit of j, e_j. This cost is calculated for the set of all tasks J initiated at o and

transported to p through a during time period t and is denoted by γ_{opajt}.

(2) The processing cost of the offloaded tasks at p with device capability k_p is given by

$$\sum_{t\epsilon T,p\epsilon P,j\epsilon J,a\epsilon A} k_p \gamma_{opajt} e_j.$$

(3) Fixed time of offloading jobs is determined from the expected cost of task transportation, f_c, with bearable communication overhead if the execution is remote, i.e. $\theta_{act} \neq 0$. Otherwise, the execution is local, which means $\theta_{act} = 0$.

$$\sum_{t\epsilon T,a\epsilon A,c\epsilon C} f_c \theta_{act} = \begin{cases} 0, & \text{if execution is local,} \\ > 0, & \text{if offloading is required.} \end{cases}$$

The local execution cost at o for the remaining non-offloaded tasks depends on the cost required for a unit of task execution at o, h_j.

$$\sum_{t\epsilon T,o\epsilon O} h_j(1-j)$$

Objective 2: To minimize the communication overhead that arises due to the offloading of application tasks to participators in the supply chain network. It comprises the following parts: discovery cost, parameter manipulation such as normalization and weight assignment, efficiency calculation and optimization cost.

(1) The communication overhead from o to p via a is given by

$$\sum_{t\epsilon T,o\epsilon O,a\epsilon A} w_{oa} \sum_{p\epsilon P,j\epsilon J} e_j \gamma_{opajt} + \sum_{t\epsilon T,a\epsilon A,p\epsilon P} w_{ap} \sum_{o\epsilon O,p\epsilon P} e_j \gamma_{opajt}$$

where w_{oa} and w_{ap} are the communication overhead during the transportation of task from o to a and from a to p, respectively.

(2) The expected communication overhead from heuristics can be determined from the actual cost of transportation with communication overhead, v_c, when the heuristic information about p chosen for offloading is already available in advisor a, i.e. $\alpha_{act} = 1$. Otherwise, it is observed newly for the p and thus $\alpha_{act} = 0$. The actual cost of transportation with communication overhead is

$$\sum_{t\epsilon T,a\epsilon A,c\epsilon C} v_c \alpha_{act}.$$

(3) Cost of adjusting the threshold level in advisors, $u_{cc'}$, for each time period in the heuristic database (DB) if needed is

$$\sum_{t\epsilon T,a\epsilon A,c\epsilon C,c'\epsilon C} u_{cc'} \delta_{acc't}$$

whose value can be either 0 if there are no threshold adjustments, c, for a decision parameter p has been made, or any positive value if c changes to new threshold adjustment, c', made between time periods. This adjustment cost depends on the decision factor $\delta_{acc't}$ which is defined as follows:

$$\delta_{acc't} = \begin{cases} 1, & \text{if } a \text{ requires threshold adjustment from } c \text{ to } c', \\ 0, & \text{if no adjustment is required.} \end{cases}$$

The summary of the dual objectives is as follows:

Obj 1: Minimize conductance cost

$$\begin{aligned} \text{Min} & \sum_{t\epsilon T,a\epsilon A,o\epsilon O} x_{oa} \sum_{p\epsilon P,j\epsilon J} e_j \gamma_{opajt} \\ & + \sum_{t\epsilon T,a\epsilon A,p\epsilon P} x_{ap} \sum_{o\epsilon O,p\epsilon P} e_j \gamma_{opajt} \\ & + \sum_{t\epsilon T,a\epsilon A,c\epsilon C} f_c \theta_{act} + \sum_{t\epsilon T,p\epsilon P,j\epsilon J,a\epsilon A} k_p \gamma_{opajt} e_j. \end{aligned}$$

Obj 2: Minimize communication overhead

$$\begin{aligned} & \sum_{t\epsilon T,o\epsilon O,a\epsilon A} w_{oa} \sum_{p\epsilon P,j\epsilon J} e_j \gamma_{opajt} \\ & + \sum_{t\epsilon T,a\epsilon A,p\epsilon P} w_{ap} \sum_{o\epsilon O,p\epsilon P} e_j \gamma_{opajt} \\ & + \sum_{t\epsilon T,a\epsilon A,c\epsilon C} v_c \alpha_{act} + \sum_{t\epsilon T,a\epsilon A,c\epsilon C,c'\epsilon C} u_{cc'} \delta_{acc't}. \end{aligned}$$

The communication overhead within a time period between owner o and the participator p contains a "variable" part, and it is based on the amount of workload transmitted between them.

3.2 *Simplified model*

In order to simplify the above two objective equations, we formulate $A_{opaj} = (x_{oa} + x_{ap})e_j$ and $W_{opaj} = (w_{oa} + w_{ap})e_j$. Then both of the objectives will be reduced as follows.

Obj 1:

$$\begin{aligned} \text{Min} & \sum_{t\epsilon T,o\epsilon O,a\epsilon A,p\epsilon P,j\epsilon J} A_{opaj} \gamma_{opajt} \\ & + \sum_{t\epsilon T,a\epsilon A,c\epsilon C} f_c \theta_{act} \\ & + \sum_{t\epsilon T,p\epsilon P,j\epsilon J,a\epsilon A} k_p \gamma_{opajt} e_j. \end{aligned} \tag{1}$$

Obj 2:

$$\begin{aligned} \text{Min} \sum_{t\epsilon T, o\epsilon O, a\epsilon A, p\epsilon P, j\epsilon J} W_{opaj}\gamma_{opajt} \\ + \sum_{t\epsilon T, a\epsilon A, c\epsilon C} v_c \alpha_{act} \\ + \sum_{t\epsilon T, a\epsilon A, c\epsilon C, c'\epsilon C} u_{cc'}\delta_{acc't} \end{aligned} \tag{2}$$

such that

$$\sum_{t\epsilon T, a\epsilon A, c\epsilon C} \alpha_{act} \leq 1, \ \forall a\epsilon A, t\epsilon T \tag{3}$$

$$\delta_{acc't} \geq \alpha_{ac't} + \alpha_{ac(t-1)} - 1, \ \forall a\epsilon A, \forall c, c'\epsilon C, c \neq c', \forall t\epsilon T \tag{4}$$

$$\sum_{t\epsilon T, o\epsilon O, a\epsilon A, p\epsilon P, j\epsilon J} \gamma_{opajt} \leq P_t, \ \forall p\epsilon P, \forall j\epsilon J, \forall t\epsilon T \tag{5}$$

$$f_c \leq h_j \gamma_{opajt} \tag{6}$$

$$\alpha_{act}\epsilon\{0,1\}, \ \forall a\epsilon A, \forall c\epsilon C, \forall t\epsilon T \tag{7}$$

$$\delta_{acc't}\epsilon\{0,1\}, \ \forall a\epsilon A, \forall c, c'\epsilon C, \forall t\epsilon T \tag{8}$$

$$\gamma_{opajt} \geq 0, \ \forall o\epsilon O, \forall t\epsilon T, \forall p\epsilon P, \forall j\epsilon J, \forall a\epsilon A. \tag{9}$$

Equations (1) and (2) are the objective functions to minimize the dual objectives in the SCM model. Both the objectives need to satisfy the constraints (3)–(9) to arrive at an optimized solution. Condition (3) explains whether the adjustment of the threshold level at the advisor is required or not required. Constraint (4) gives the threshold adjustment level calculated in the consecutive time periods t and $t-1$. The value of (4) becomes one when there is no change between α_{act} and $\alpha_{ac't}$ values. Constraint (5) controls the number of offloads arriving from the owner at each time period with the capability of the participators. Constraint (6) implies that the communication overhead does not exceed local execution cost. Constraints (7)–(9) define the decision factors.

3.3 *Dual Lagrangian decomposition method*

Solving the dual objective problem is combining both the objectives into a single equation. The combined objective function is given as

$$Overall\, Obj = y \times Obj_1 + z \times Obj_2. \tag{10}$$

The variables $\beta = y$ and $(1-\beta) = z$ vary the optimization towards any one of the two objectives. The value of β is set between 0 and 1. When it is 1, Obj 2 is achieved and vice versa. Thus, β is considered as the relative importance factor between them. The solution methodology has to find the optimal value of β such that the dual objectives are given importance, subject to the constraints (3)–(9) being satisfied.

A dual Lagrangian decomposition method is used to solve this optimization problem. By applying this method, the following answer has to be found:

$$\text{argmax}_{(y,z)\epsilon\omega}(y \times Obj_1 + z \times Obj_2) \tag{11}$$

where ω is the finite set of solutions to the problem and subject to the constraint (5).

Consider any value of $Obj_1\epsilon R^d$; then it is easy to find $\text{argmax}_{y\epsilon Y}(y \times Obj_1)$. Similarly, for any value of $Obj_2\epsilon R_{d'}$, it is easy to find $\text{argmax}_{z\epsilon Z}(z \times Obj_2)$. Then it is easier to find

$$(y^*, z^*) = \text{argmax}_{(y,z)\epsilon\omega'}(y \times Obj_1 + z \times Obj_2). \tag{12}$$

Moreover, the values of y^*, z^* are set as follows:

$$\begin{aligned} y^* &= \text{argmax}_{(y,z)\epsilon\omega'}(y \times Obj_1), \\ z^* &= \text{argmax}_{(y,z)\epsilon\omega'}(z \times Obj_2). \end{aligned}$$

This is in resemblance to Equation (11). The difference is that ω is replaced by ω'. By doing such changes, it becomes easier to solve Equation (12) rather than solving Equation (11). The idea behind this method is to solve the sub-problems instead of the entire problem. A subset of ω is taken to find the optimized solution within a finite set ω. It is denoted as ω'; ω' does not consider the constraints of the original problem. Later the dual Lagrangian decomposition is derived by introducing the Lagrangian multipliers.

Therefore the equation becomes

$$L(\mu_{pjt}, y, z) = y \times Obj_1 + z \times Obj_2 + \mu_{pjt} \sum_{p\epsilon P} \sum_{j\epsilon J} \sum_{t\epsilon T} \sigma_{pjt}$$

where μ_{pjt}, y and z are the Lagrangian multipliers and $\sigma_{pjt} = (P_t - \sum_{o\epsilon O, a\epsilon A} \gamma_{opajt})$. The dual objective is

$$L(\mu_{pjt}) = \max_{(y,z)\epsilon\omega'} L(\mu_{pjt}, y, z). \tag{13}$$

The optimized solution is obtained by min $L(u)$. The multiplier for the next equation is

$$L\left(\mu_{pjt}^{l+1}\right) = \mu_{pjt}^{l} - \alpha^l \sigma_{pjt}^{l}$$

and $(y^l, z^l) = \text{argmax}_{(y,z)\epsilon\omega'} L(\mu_{pjt}^{l-1}, y, z)$; $\alpha^l \geq 0, l = 0, 1, 2, \ldots$

The solutions y^l, z^l are found easily by verifying it with

$$\begin{aligned} \text{argmax}_{(y,z)\epsilon\omega'} L\left(\mu_{pjt}^{l-1}, y, z\right) = \text{argmax}_{(y,z)\epsilon\omega} y \times Obj_1 \\ + \text{argmax}_{(y,z)\epsilon\omega} z \times Obj_2 \end{aligned}$$

where $Obj_1' = Obj_1 + P_t \mu_{pjt}^{l-1}$ and $Obj2' = Obj_2 - \mu_{pjt}^{l-1} \sum_{o\epsilon O, a\epsilon A} \gamma_{opajt}$. The dual objective has been decomposed into two solvable maximization problems. The original problem can be reconstructed now as

$$\min L(u) = \max_{(\pi,\vartheta)\epsilon\varphi} \pi \times Obj_1 + \vartheta \times Obj_2 \quad (14)$$

and the set ϑ is given as

$$\vartheta = \Bigg\{ (\pi, \vartheta) : (\pi, \vartheta) \epsilon Conv(\omega') \text{ and } P_t \mu_{pjt}^{l-1} - \mu_{pjt}^{l-1} \sum_{o\epsilon O, a\epsilon A} \gamma_{opajt} = d \Bigg\}.$$

Finally, the problem

$$\max_{(\pi,\vartheta)\epsilon\varphi} \pi \times Obj_1 + \vartheta \times Obj_2 \quad (15)$$

is a linear problem and therefore $L(u)$ is its dual objective linear problem.

3.4 *Workload distribution based on participator capacity*

If s be the speed of a participator p, then the time to execute the amount of workload w using p is $T_{executeatp} \leftarrow \frac{w}{s}$ [24]. Suppose the workload is of size d and the processing capacity of all participators $p_{1,\ldots,n}$ is denoted by $M(p_1), M(p_2), \ldots, M(p_n)$, the workload will be chosen to be offloaded to the p_i with less processing time among $p_{1,\ldots,n}$.

Algorithm 1 Device sensing in the Foglet

Input : Source network address
Output : Set of participators, P

procedure AVAILABLE PARTICIPATORS(t, o)
//Finding P nearer to o among set of owners O in a time period t
for each o in O **do**
$nw \leftarrow PROXIMATEFOGLET(t, o)$
$p \leftarrow FOGLETNODES(nw, t, o)$

procedure PROXIMATEFOGLET(t, o)
// For a particular t in set of all time periods T within an execution
for t in T **do**
$o_{ip} \leftarrow getNetIP(o)$
$cidr \leftarrow netmask(ip)$
$o_{nid} \leftarrow \frac{o_{ip}}{cidr}$
return o_{nid}

procedure FOGLETNODES(nw, t, o_{nid}, o_{ip})
if $o_{id} isVALID$ **then**
$start \leftarrow o_{id}.startIP()$
$end \leftarrow o_{id}.endIP()$
for $ip \geq start \&\& ip \leq end$ **do**
if $o_{id} isVALID$ **then**
// Search for active nodes
$updateDB(p_{ip}, a)$
$ip++$
else $ip++$

4. Supply-chain-oriented Offload Decision Model

The enhanced supply-chain-based CASCM^2 model has been deployed to find the optimal participators for offloading.

4.1 *Neighbourhood device sensing*

The opportunistic offloading initiated by the owner triggers an offload request to its respective a. The Neighbourhood Device Sensing Manager in the advisor on accepting the offload request from the owner senses the network to identify the dynamic collections of local heterogeneous devices such as smartphones, and mobile and desktop computers. The accessibility of the available nodes within the Foglet range is confirmed, and their network addresses are recorded in the DB, as explained in Algorithm 1.

4.2 *Determination of offload-qualification parameters*

There are two QoS parameters associated with participator selection in the Foglet: conductance cost η and the communication overhead ξ. The values of η, ξ are determined using Equations (1) and (2) by the Environment analyser as given in Algorithm 2. During runtime, these parameters are calculated by the Fog server to find a suitable candidate from the list of neighbours obtained from the Neighbourhood Device Sensing Manager, provided it satisfies Equations (3)–(9).

Algorithm 2 Offload-qualification parameters

Input : Available proximates from available participators method
Output : Participator's qualification parameters
$\eta \leftarrow eqn.(1)$
$\xi \leftarrow eqn.(2)$
return η, ξ

4.3 *Proactive QoS-driven participator selection*

The Fog server acts as the Entry-Point daemon in Foglet, which keeps statistics on resource usage by all the participators within the range. The participators are chosen commensurate with the heuristics recorded during the previous offloading, and the process is given in Algorithm 3. For example, the participator at time t is p_t and the

participator in the previous time period is p_{t-1}. If p_t has efficiency greater than that of p_{t-1}, then DB is updated with the new details of the participator. Else, the data is discarded.

4.4 *Mobility scenario*

Fog computing infrastructure must have the property of elasticity to dynamically satisfy the demands of the users [16]. The joining and dis-joining of resources in a Foglet is a challenging task to be handled by the advisor in CASCM^2 model.

4.4a *Participator joins Foglet*: When a new participator enters the Foglet, it is identified by the advisor using available participators procedure and is denoted as p_{new}. Later the advisor determines the offload qualification parameters of p_{new} from Algorithm 2 and ranks the active participators, including p_{new}, using Algorithm 3. The joining of a participator in the Foglet has been given in joins Foglet procedure of Algorithm 4.

4.4b *Participator disjoins Foglet*: During offloading at a participator (p_{chosen}), if p_{chosen} moves out of the Foglet range, an instant rescheduling of the task has to be carried out to complete the task execution. It requires an offload decision of executing the offloaded task either locally or remotely in one of the active and efficient participators. The disjoins Foglet procedure in Algorithm 4 explains the rescue steps taken during disconnection of p_{chosen}. $T_{elapsed}$ represents the total time elapsed during task execution in the inactive P_{chosen} (T_{rexec}) with the initial offload overhead time, $T_{overhead}$. The strategies of joining and disjoining of participators in a Foglet are presented, respectively, in figures 4 and 5.

4.5 *Complexity analysis*

The computational complexity of the proposed algorithm is briefly analysed here. Using Equations (1) and (2), the QoS parameters such as η and ξ are calculated for the available participators during device discovery. Therefore, for N participators, the device sensing takes $\sum_{i=1}^{N} P_i$ time and therefore its complexity is O(N). The compare participators procedure in Algorithm 3 has O(N^2) time complexity. Since the offload decision by the advisor uses dual Lagrangian decomposition, which divides the original problem into sub-problems, each individual solution takes linear time. The iteration in this algorithm depends on the number of participators P. An increase in P value will increase the participator selection time logarithmically. For the Lagrangian multiplier μ_{pjt} to converge, it takes O($N^2 - 1$) complexity.

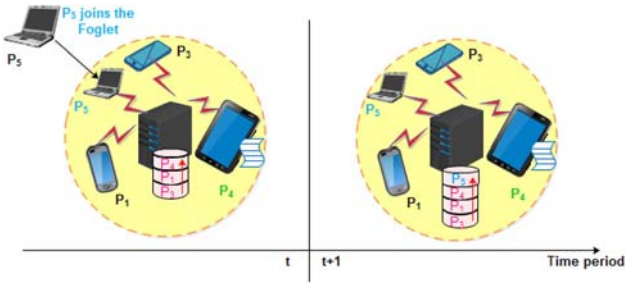

Figure 4. P_1, P_3, P_4 are the active Fog nodes in the Foglet with ranking order $P_4 > P_1 > P_3$. P_5 is a new Fog node that joins the Foglet at a time period t. Participator selection and ranking are done by the advisor as soon as a new node is discovered in the next time period $t+1$. Green colour indicates the selected Offload-participator within the Foglet.

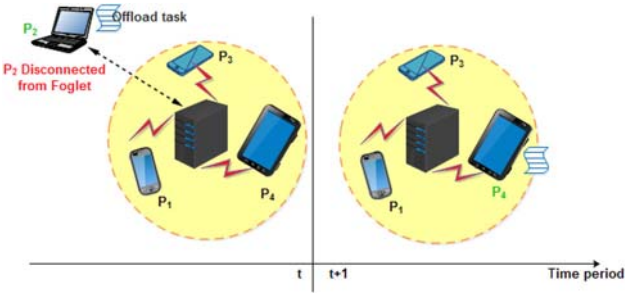

Figure 5. P_1, P_2, P_3, P_4 are the active Fog nodes in Foglet with ranking order $P_2 > P_4 > P_1 > P_3$. P_2 is the chosen offload participator among them. P_2 moves out of range from Foglet (disconnected) at time period t.

Algorithm 3 QoS-driven participator selection

Input : Available participators, owner
Output : Offload-fit participators

procedure FINDEFFICIENCY (p)
  $Efficiency(p_t) \leftarrow \beta\eta + (1-\beta)\xi$
  β is the importance factor
  $storeDB(Efficiency(p_t))$
procedure COMPAREPARTICIPATOR(p_t, p_{t-1}, o)
  $p_{t-1} \leftarrow retrieveDB(p)$
  if $Efficiency(p_t) \geq Efficiency(o)$ **then**
    for $p_t \leftarrow 0 top_t.length$ **do**
      for $p_{t-1} \leftarrow 0 top_{t-1}.length$ **do**
        if $Efficiency(p_t) \geq Efficiency(p_{t-1})$ **then**
          $updateDB(p_t)$
  $p_{offload} = sort(p_t, p_{t-1})$

To obtain a ε-optimal solution for the original problem, Lagrangian algorithms have to perform O($\frac{1}{\varepsilon}$) total projections onto the feasible region. Once the offload-fit participators that fall in the optimal region are found they are

sorted in the order of their η and ξ parameters, which is $O(N \log N)$. Thus, the overall complexity of the proposed model is estimated to be of polynomial order $O((N^2 - 1 + N + N \log N)\log(\frac{1}{\varepsilon}))$. Other models such as game theory have a lower bound time complexity of $O(2^{(2^N)})$ and the ranking algorithms like TOPSIS have $O(N^2 + N + 1)$ complexity. The complexities of AHP and its variants depends on the number of criteria, m. The complexities of AHP and AHP-Fuzzy are $O(mN^2)$, and that of AHP-TOPSIS is $O(N^2)$. The complexities of Round-Robin and FCFS are O(1), which is less than that of CASCM^2; the reason why CASCM^2 outperforms the two models is discussed in section 2.

Algorithm 4 Participator Mobility

Input : Offload-fit Participators, p_{chosen} and sorted Participator list, $p_{offload}$

procedure JOINS FOGLET(p_{new})
  if p_{new} *in nw from available participators*(T, o) **then**
    Step1: Proceed with Algorithm 2
    Step 2: Proceed with Algorithm 3 to find $p_{offload}$
  return $p_{offload}$
procedure DISJOINS FOGLET(p_{chosen})
  if $p_{chosen}.status == inactive$ **then**
    $T_{elapsed} = T_{overhead} + T_{rexec}$
    if $T_{elapsed} < T_{local}$ **then**
      Proceed with Algorithm 3 to find new $P_{offload}$
      return $P_{offload}$
    else return o

5. Results and discussion

To show the efficiency of CASCM^2, Quiz-inline video is taken as a test application. This section compares three priority-based cases under CASCM^2 and also gives a performance overview of CASCM^2 with other models. Conductance cost and communication overhead of the application are investigated for single- and multi-period scenarios.

5.1 *Experimental set-up*

In the test application the quiz overlaying on the video is a computation-intensive task, which demands offloading. This application is made to be executed in mobile devices of the students in a next-gen smart classroom. A lecture video embedded with questions will be displayed on the screen. Video is paused during the question time and once an appropriate answer is chosen, video is continued.

Here the advisor is an Ubuntu server, acting as a Wi-Fi access point. This constitutes the Fog server. Ten Fog nodes are created as VMs using VirtualBox in two physical hosts, each with 8 GB of RAM. During the quiz session, an automatic offload request is generated by the owner to the advisor.

5.2 *Parameter-priority-based results*

The impact of the number of participators and input data size of the application on total offload-task execution time has been observed. The total offload execution time includes both conductance cost η and communication overhead time ξ. This section compares the total offload execution time for different priorities of η and ξ.

5.2a *Case (i). Priority(η) is higher*: When η is considered as the offload-decision parameter, then the participator p will be chosen to conform to the input size d, i.e. $p \propto d$. If d is high, then p with powerful processor configuration is preferred for offloading even though the distance between participator and owner may be large. The number of available participators (P) within the Foglet is taken as 10.

In figure 6, with a constant number of participators (set to 5), it is seen that as d increases, the total offload execution time increases. This is due to the rise in transportation cost ϕ_t. Hence the advisor chooses a potential p with high processing power, which in turn reduces the conductance cost. Alternatively, in figure 7 the input size is kept constant at 100 MB. Here, the total offload execution time increases with the number of participators due to communication overhead.

5.2b *Case (ii). Priority(ξ) is higher*: For the chosen participator p, when ξ is given a higher priority, the distance between p and o determines the offload decision. In other words, if the distance between p and o is low then that particular p is chosen as the offload-fit participator. Although the network distance is optimum in this case, conductance cost increases, leading to a relatively higher offload execution time. This can be seen in figure 6.

5.2c *Case (iii). Priority(η) = Priority(ξ)*: In this case, η and ξ are given equal priority. Advisor appoints p in such a way that conductance cost, as well as the communication overhead, is optimal. This scenario provides the most optimum offload task execution time, as seen in figures 6 and 7. This may vary depending on the test application. If the application has a complex task with small input size then Case (i) is optimal, whereas Case (ii) would be preferred for an application with large input size and minimum task complexity. For the considered test application, which requires both heavy computations and large input size, Case (iii) performs better.

5.2d *Discussion on single-period decision results*: In figure 6, where the number of participators is kept constant, the optimization cost varies roughly. Such optimization cost does not have a greater impact on ξ. Thus, Case (ii)

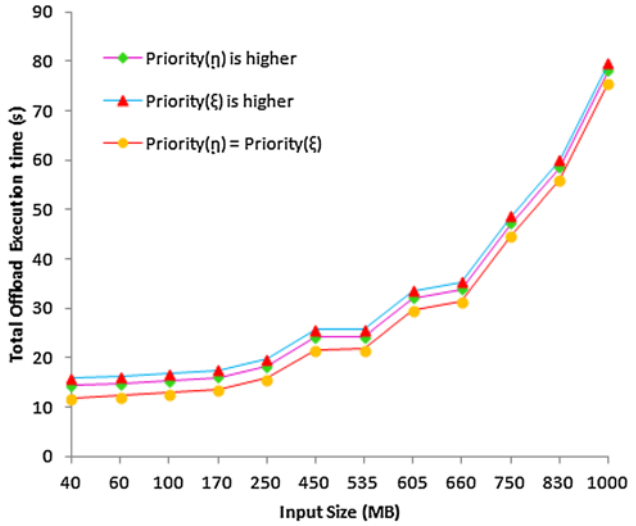

Figure 6. Total offload execution time with respect to input size.

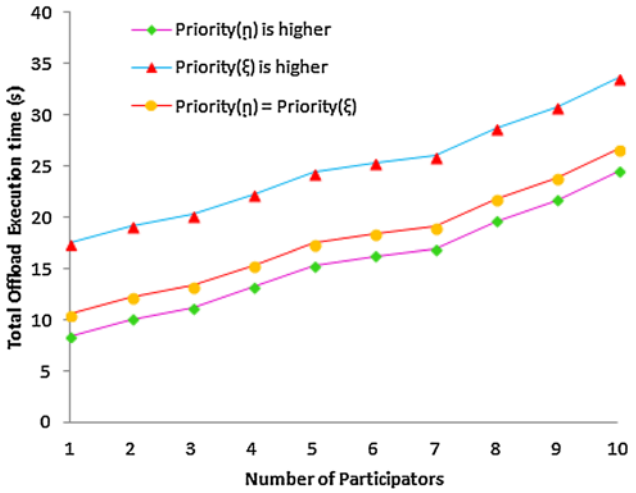

Figure 7. Total offload execution time with respect to number of participators.

behaves similar to Case (i) and Case (iii) provides the optimal offload execution time marginally. Though equal priority is given to η and ξ in Case (iii), considerable variation in total offload execution time is not present provided that the number of participators is kept constant. Figure 6 shows that all three cases depend upon input size, which is directly proportional to the total offloading time.

When the number of participators is incremented from 1 to 10, the variation in total offload execution time among the three cases is significant. This can be observed in figure 7. A raise in participators count makes the efficient device discovery and optimization processes to consume time, which in turn increases the communication overhead to some extent. Thus, Case (ii) shows the highest total offload execution time.

In Case (iii) where equal parametric priorities are given, the advisor has the responsibility to choose participators with both higher processing power and lower communication overhead. This becomes tedious when the number of participators increases. This is shown in figure 7, where Case (i) has an advantage over the constant input size. The conductance cost of CASCM^2 and local execution are compared in figure 8. When the input size increases above 60 MB, the conductance cost of the proposed model becomes lower than that of local execution. This results in a beneficial offloading for the computation-intensive task of the test application.

5.3 *Single-period-based decision with varying* β

The importance factor (β) between η and ξ has been varied from 0.1 to 1 and efficiency has been noted for a 500 MB input size, as shown in figure 9. The efficiency is calculated using Algorithm 3 for each of the active participators. The algorithm conveys that a low conductance cost and a high communication overhead appear for a higher β value. For lower β value, a high conductance cost and low communication overhead are obtained. When β reaches closer to the central point, both η and ξ are given equal priority. Thus the application execution becomes optimum at that point, making efficiency to be the highest at that optimal point.

When β increases from 0.5 to 1 the importance level for ξ, which is $(1-\beta)$, reduces and deviates from its optimal range. Though η value starts becoming lower after the median point, a rise in ξ value degrades the efficiency of offload decision. Hence, it leads to the selection of a less-efficient participator during offloading. A comparison of offloading overhead for mCloud [25], ThinkAir [26] and CASCM^2 is presented in figure 10. It is seen that the overhead of CASCM^2 is lower than that of the other two models.

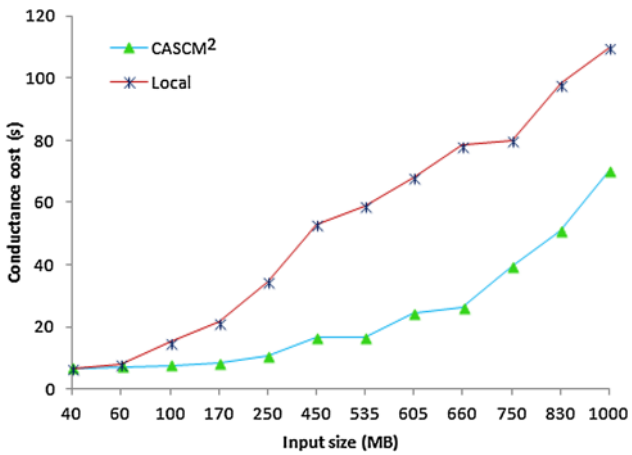

Figure 8. Remote execution time for the offloaded task in the offload-fit participator.

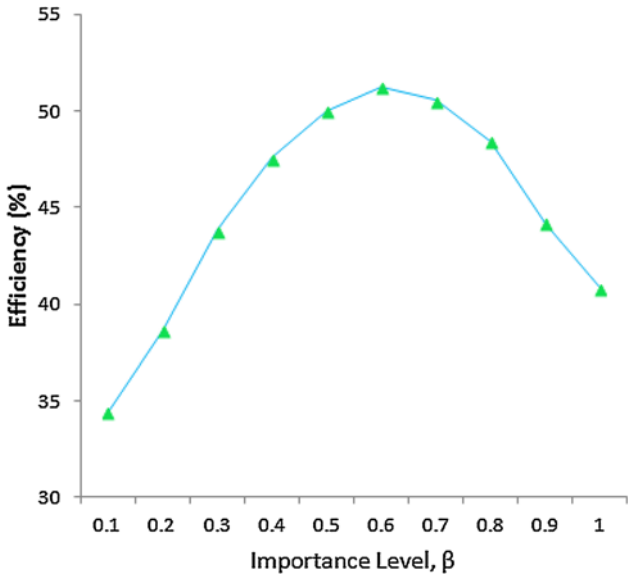

Figure 9. Efficiency for varying β.

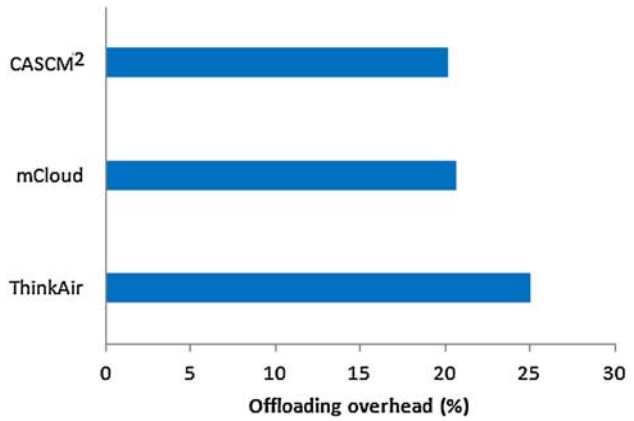

Figure 10. Offload decision overhead for proposed, mCloud and ThinkAir models.

5.4 *Multi-period-based decision*

In comparison to single-period decisions, multi-period decisions take previous optimal solutions into account. This heuristic data supports the offload decision to arrive at an optimal solution faster than single-period decision. It also assists in preferring the best offload-fit participators and establishing a stable connection between users and participators. It reduces the chance of connection breakdowns during offloading and thus, ensures connection reliability.

5.4a *Case (i). Multi-period decision considering* η: This case draws η value by concatenating the decisions over earlier time periods. It gives a perspective about the optimal η and chooses the offload-fit participators accordingly. This heuristic method cuts down the communication overhead, as seen in figure 11.

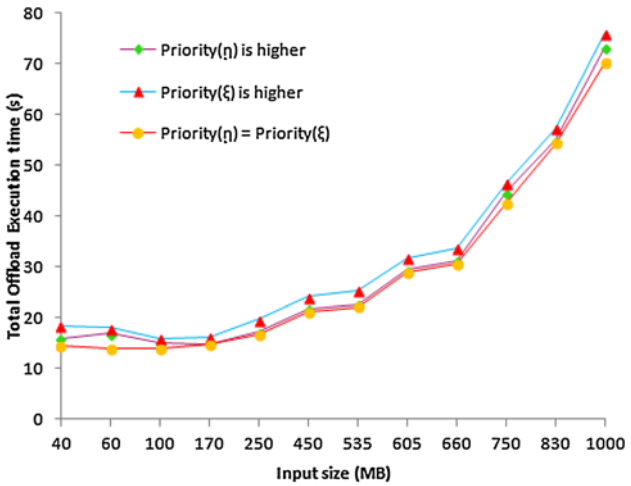

Figure 11. Offload task completion time for multi-period decision when considering η and ξ as decision parameters.

5.4b *Case (ii). Multi-period decision considering* ξ: While taking ξ as the decision criterion across several time periods, participators with less communication overhead are chosen. Its corresponding total offload execution time is plotted in figure 11.

5.4c *Case (iii). Multi-period decision considering both* η *and* ξ: This case considers both η and ξ from the heuristic data to determine the offload-fit nodes within a Foglet. It is seen from figure 11 that Case (iii) has the lowest offload execution time.

5.4d *Discussion on multi-period decision results*: Between single- and multiple-period-based decision results, the latter performs well in all the three priority-based cases. This can be compared from figures 6 and 11. Additional comparisons on the offload execution time for CASCM^2 are made with Round-Robin (resource selected in the order of participators identified and stored in the participator DB) and First Discovered First Chosen (FDFC – the first discovered resource is chosen as participator) models in figure 12.

There are a few instances in figure 12 at which the offload execution time of Round-Robin algorithm overwhelms that of the proposed algorithm. The sequential selection of p by Round-Robin algorithm during an offload run may be efficient by chance wherein CASCM^2 may consume extra time due to decision overhead. The Round-Robin method contains a participator DB with the resources arranged in the order of their arrival time. It assigns participator from the DB during offloading. If the resource assigned at a particular time period is optimal by chance, then Round-Robin may perform better than CASCM^2. In our scenario, it happened for the input size of 450 MB.

Especially with 450 MB input size, CASCM^2 has a delay of 3.619 s over Round-Robin due to communication overhead. However, CASCM^2 always chooses an optimal participator and is more reliable than the Round-Robin

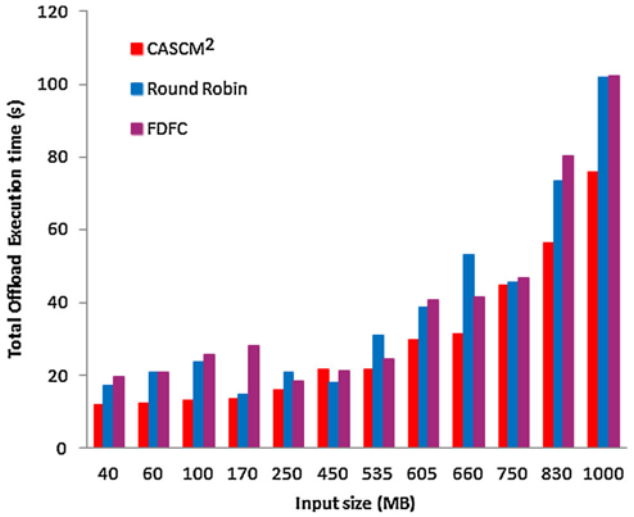

Figure 12. Comparison of offload execution times for proposed, FDFC and Round-Robin models.

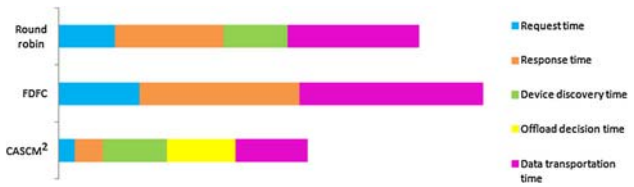

Figure 13. Overhead analysis of CASCM^2 with FDFC and Round-Robin models.

method. On comparing the time taken for request, response, device discovery, offload decision and data transportation using proposed, FDFC and Round-Robin models, it is found that CASCM^2 performs better than the other two models, as shown in figure 13. It is due to the optimization of overhead and conductance cost by CASCM^2.

5.5 *Impact on energy consumption*

Communication overhead plays an important role in energy consumption for most of the offloading models. The number of active Fog nodes tends to raise the communication overhead or execution delay to arrive at an optimal resource selection by the advisor. Figures 14 and 15 show the execution delay and energy consumption due to the number of active nodes, respectively. As an overview, the energy consumption for E&D&P [15] is compared to that of CASCM^2. It is seen that CASCM^2 model consumes lesser energy than E&D&P model up to a certain number of participators. Figure 16 gives an idea about how energy consumption varies with varying input size during remote execution.

From figure 16, it is evident that offloading yields a profitable result when the input size is greater than 100 MB.

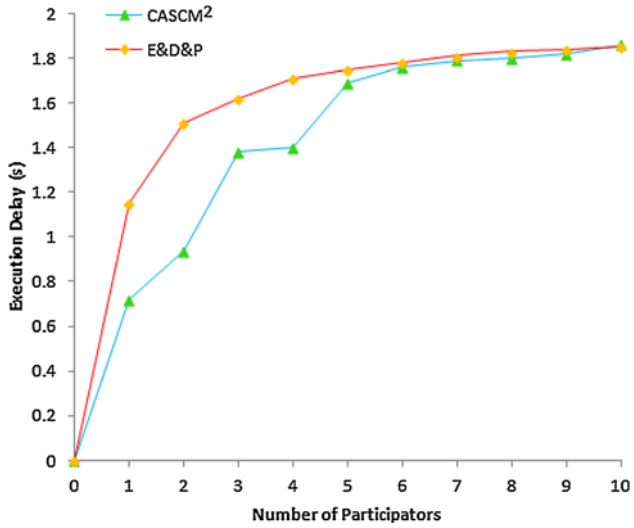

Figure 14. Execution delay for different number of Fog nodes.

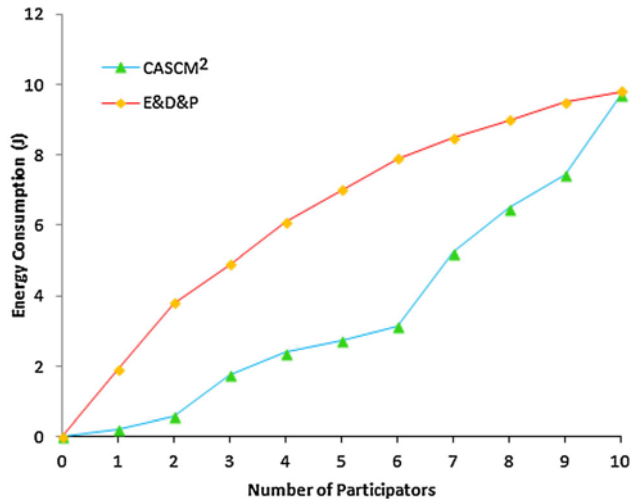

Figure 15. Energy consumed during offload decision process.

The energy consumption due to offloading overhead adds up to the remote execution and exceeds the energy consumed by local execution. When the input size exceeds 100 MB, the high conductance cost for local execution increases its energy consumption. On the other hand, offloading has the advantage of choosing powerful processors to carry out computation-intensive tasks. This, in turn, keeps the energy consumption lower than that of local execution. Thus it is inferred that remote execution should be chosen for larger input sizes, whereas opting local execution for smaller input sizes will be effective. An average of 20.33% of energy is saved due to offloading when the input size is above 100 MB.

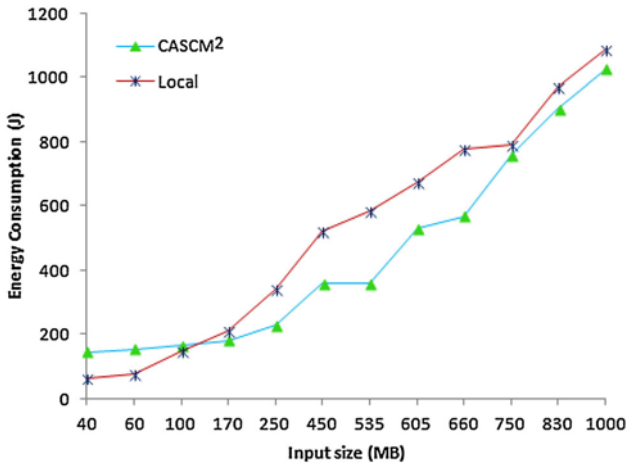

Figure 16. Energy consumption for remote execution.

6. Conclusion

The supply-chain-based CASCM^2 provides an edge-based solution for minimizing QoS parameters such as conductance cost and communication overhead. For a Fog framework with dynamic offloading, optimization is achieved using dual Lagrangian decomposition. The proactive participator selection yields efficient offload-fit participators. These participators handle latency-sensitive and computation-intensive tasks of the test application. Results have been obtained for both single and multi-period decisions by varying the priorities of QoS parameters. Comparison of results shows that CASCM^2 performs better than other models in terms of overhead, energy consumption and execution time. In future work, a co-operative execution of the application tasks has to be carried out with a crowd of participators simultaneously.

Acknowledgement

This research is supported by Visvesvaraya PhD Scheme for Electronics and IT (VISPHD-MEITY-2560), Ministry of Electronics and Information Technology, Government of India. The authors also want to thank the editor and reviewers for their valuable comments and suggestions.

References

[1] Abreu D P, Velasquez K, Assis M R M, Bittencourt L F, Curado M, Monteiro E and Madeira E 2018 A rank scheduling mechanism for fog environments. In: *Proceedings of the 2018 6th IEEE International Conference on Future Internet of Things and Cloud (FiCloud)*, pp. 363–369

[2] Zhang W, Wen Y and Wu D O 2015 Collaborative task execution in mobile cloud computing under a stochastic wireless channel. *IEEE Trans. Wirel. Commun.* 14: 81–93

[3] Orsini G, Bade D and Lamersdorf W 2015 Context-aware computation offloading for mobile cloud computing: requirements analysis, survey and design guideline. *Procedia Comput. Sci.* 56: 10–17

[4] El-Barbary A E H G, El-Sayed L A, Aly H H and El-Derini M N 2015 A cloudlet architecture using mobile devices. In: *Proceedings of the 2015 12th IEEE/ACS International Conference of Computer Systems and Applications (AICCSA)*, pp. 1–8

[5] Kumar J, Malik A, Dhurandher S K and Nicopolitidis P 2017 Demand-based computation offloading framework for mobile devices. *IEEE Syst. J.* 12: 3693–3702

[6] Kumar K, Liu J, Lu Y H and Bhargava B 2013 A survey of computation offloading for mobile systems. *Mob. Netw. Appl.* 18: 129–140

[7] Duan X, Huang M, Yang X and Wan B 2013 A method of partner selection for supply chain based on Grey-ANP in cloud computing. In: *Proceedings of the 2013 10th Web Information System and Application Conference*, pp. 377–382

[8] Fernando N, Loke S W and Rahayu W 2016 Computing with nearby mobile devices: a work sharing algorithm for mobile edge-clouds. *IEEE Trans. Cloud Comput.* 7: 329–343

[9] Herrera A and Janczewski L 2016 Cloud supply chain resilience model: development and validation. In: *Proceedings of the 2016 49th Hawaii International Conference on System Sciences (HICSS)*, pp. 3938–3947

[10] Son S, Kim J and Ahn J 2017 Design structure matrix modeling of a supply chain management system using biperspective group decision. *IEEE Trans. Eng. Manag.* 64: 220–233

[11] Parmar D, Kumar A S, Nivangune A, Joshi P and Rao U P 2016 Discovery and selection mechanism of cloudlets in a decentralized MCC environment. In: *Proceedings of the International Conference on Mobile Software Engineering and Systems*, pp. 15–16

[12] Dolui K and Datta S K 2017 Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing. In: *Proceedings of 2017 Global Internet of Things Summit (GIoTS)*, pp. 1–6

[13] Kiliç H S 2012 Supplier selection application based on a fuzzy multiple criteria decision making methodology. *AJIT-e: Online Acad. J. Inf. Technol.* 3: 7–18

[14] Banerjee S, Adhikari M, Kar S and Biswas U 2015 Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud. *Arab. J. Sci. Eng.* 40: 1409–1425

[15] Liu L, Chang Z, Guo X, Mao S and Ristaniemi T 2017 Multi-objective optimization for computation offloading in fog computing. *IEEE IoT J.* 5: 283–294

[16] Bittencourt L F, Diaz-Montes J, Buyya R, Rana O F and Parashar M 2017 Mobility-aware application scheduling in fog computing. *IEEE Cloud Comput.* 4: 26–35

[17] Panigrahi C R, Sarkar J L and Pati B 2018 Transmission in mobile cloudlet systems with intermittent connectivity in emergency areas. *Digital Commun. Netw.* 4: 69–75

[18] Shu P, Liu F, Jin H, Chen M, Wen F, Qu Y and Li B 2013 eTime: energy-efficient transmission between cloud and mobile devices. In: *Proceedings of 2013 IEEE INFOCOM*, pp. 195–199

[19] Pham X Q and Huh E N 2016 Towards task scheduling in a cloud-fog computing system. In: *Proceedings of the 2016 18th Asia–Pacific Network Operations and Management Symposium (APNOMS)*, pp. 1–4

[20] Du J, Zhao L, Feng J and Chu X 2018 Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee. *IEEE Trans. Commun.* 66: 1594–1608

[21] Dewi R K, Hanggara B T and Pinandito A 2018 A comparison between AHP and hybrid AHP for mobile based culinary recommendation system. *Int. J. Interact. Mob. Technol.* 12: 133–140

[22] Chang Z, Zhou Z, Ristaniemi T and Niu Z 2017 Energy efficient optimization for computation offloading in fog computing system. In: *Proceedings of the GLOBECOM-2017 IEEE Global Communications Conference*, pp. 1–6

[23] Ling J M and Liu P H 2015 Economic analysis of Lagrangian and genetic algorithm for the optimal capacity planning of photovoltaic generation. *Math. Probl. Eng.* 2015: 1–7

[24] Chen X, Chen S, Zeng X, Zheng X, Zhang Y and Rong C 2017 Framework for context-aware computation offloading in mobile cloud computing. *J. Cloud Comput.* 6: 1–17

[25] Zhou B, Dastjerdi A V, Calheiros R N, Srirama S N and Buyya R 2015 mCloud: a context-aware offloading framework for heterogeneous mobile cloud. *IEEE Trans. Serv. Comput.* 10: 797–810

[26] Kosta S, Aucinas A, Hui P, Mortier R and Zhang X 2012 Thinkair: dynamic resource allocation and parallel execution in the cloud for mobile code offloading. In: *Proceedings of 2012 IEEE Infocom*, pp. 945–953