Constraints Based Heuristic Approach for Task Offloading In Mobile Cloud Computing

Raj Kumari and Sakshi Kaushal

***Abstract*— Mobile devices are supporting a wide range of applications irrespective of their configuration. There is a need to make the mobile applications executable on mobile devices without concern of battery life. For optimizing mobile applications computational offloading is highly preferred. It helps to overcome the severity of scarce resources constraint mobile devices. In offloading, which part of the application to be offloaded, on which processor and what is available bandwidth rate are the main crucial issues. As subtasks of mobile applications are interdependent, efficient execution of application requires research of favorable wireless network conditions before to take the offloading decision. Broadly in mobile cloud computing the applications is either delay sensitive or delay tolerant. For delay sensitive applications completion time has the highest priority whereas for delay tolerant type of applications depending on the network conditions decision of offloading can be taken. Sometimes, computation time on a cloud server is less but it consumes high communication time which ultimately gives inefficient offloading results. To address this issue, we have proposed a heuristic based level wise task offloading (HTLO). It includes computation time, communication time and maximum energy available on the mobile device to take the decision of offloading. For simulation study, a mobile application is considered as a directed graph and all the tasks are executed on the basis of their levels. The overall results of the proposed heuristic approach are compared with state-of-the-art K-M LARAC algorithm and results show the improvement in execution time, communication time, mobile device energy consumption and total energy consumption.**

***Keywords*— mobile cloud computing, offloading, heuristic, optimization, K-M LARAC.**

I. INTRODUCTION

Nowadays mobile devices have become the necessity of modern life. People prefer to execute numerous applications like web browsing, face recognition, watching videos, online games and many others[1] [2] on mobile devices. These applications are computation intensive and require high processing speed and storage capacity. As mobile devices are resource constraint and battery operated, the concept of mobile cloud computing (MCC) is used to execute these mobile applications[3][4]. All these applications are executed on mobile devices on the stake of battery life. Longer battery life is the priority of all mobile users. MCC integrates the capabilities of cloud computing [5] into the mobile environment and overcomes the problem of resource constraints of the mobile devices. MCC technique executes the mobile application on the cloud server and extends the battery life of the mobile devices.

Now, to execute the mobile applications efficiently with respect to time, energy and cost computation offloading is used [6][7]. Computation offloading improves the performance and reduces the local execution cost through migrating heavy computation tasks to the cloud servers[8]. The use of computation offloading requires high research regarding characteristics of the application i.e. whether the application is computation intensive or communication intensive[9][10]. In some research papers, applications are refer to as delay sensitive or delay tolerant [11][12][13]. For delay, sensitive applications completion time has the highest priority whereas for delay tolerant type of applications depending on the network conditions decision of offloading can be taken. Therefore, it is very important to make the balance between communication and computation during offloading. There are many factors which affect the efficiency of offloading in the MCC environment. Availability of bandwidth rate between the cloud server and mobile device is the main factor which affects the offloading [14]. The internal behavior of the application must be checked before to do the computation offloading. If the application consists of high dependency among the tasks then communication cost may lead towards the inefficient offloading. In such cases, communication overhead is high than the computation overhead. In this paper, we proposed a heuristic approach for level-wise task offloading algorithm. The proposed algorithm aims to reduce the communication overhead of delay sensitive applications. The objective of the proposed work is to minimize the completion time of the application such that overall completion time is within the deadline and maximum available energy. In the K-M LARAC algorithm [15], the

Raj Kumari is with the University Institute of Engineering and Technology,Panjab University, Chandigarh, India
Dr. Sakshi, is with the University Institute of Engineering and Technology,Panjab University, Chandigarh, India

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

tasks are offloaded with the main objective function to minimize the financial cost. In this algorithm, the shortest path of the task graph is selected using the Dijkstra algorithm. If the path satisfies the financial cost constraint then the solution is cost optimized and if it satisfies the energy or deadline constraint then with respect to that constraint solution is optimized. But in our approach, the solution of offloading satisfies the deadline and energy constraint. The performance of the proposed HLTO algorithm is compared with the K-M LARAC algorithm with respect to execution time, communication time, energy consumption on the mobile device and overall energy consumption of the application. The results show high performance as compared to the K-M LARAC algorithm. The existing algorithm concentrates on the shortest path of the application which sometimes does not fulfil the constraints of deadline and energy consumption. To overcome this problem, we have proposed a heuristic approach based level-wise task offloading. This results in reducing the communication overhead and completes the application within the deadline and energy constraint. The contributions of the proposed work can be summarised as follows:

- The objective function is formulated as a cost minimization function by considering the overall execution time and energy consumption within the deadline and maximum available energy respectively.
- To reduce the communication time for the delay sensitive applications, the level wise task offloading technique is used.
- We proposed a heuristic approach for taking the offloading decision between the mobile device and the cloud server.
- Experimental results show the better performance of the proposed work than the existing algorithm for the delay sensitive applications.

The rest of the paper is structured as follows. Section 2 explores the factors that influence application performance and offloading. It also introduces the existing work on application offloading. Section 3 describes system and computation model. Section 4 illustrates the heuristic approach for level-wise task offloading. Section 5 presents the experimental setup, evaluation, and analysis of the results. Discussion on the proposed work is presented in section 6. Finally, section 7 concludes the work and highlights future directions.

I. PROCEDURE FOR PAPER SUBMISSION

Partitioning of mobile application into computation intensive and non-computation is a big research area in the field of mobile cloud computing. After partitioning, computation offloading of an application is also a complex task[16][17][18][19]. Offloading efficiency depends on many parameters like dynamic network conditions, characteristics of the mobile application, processing efficiency of mobile devices and services of the cloud server. A heuristic algorithm that can find a solution for the offloading problem is discussed in [15]. In the algorithm, the best solution for offloading is equivalent to finding the constrained shortest path in the task graph. In [20] two algorithms have been proposed for the offloading problem between mobile device and cloud data center. The results show the minimum offloading time for mobile device and the minimum execution time for the cloud data center. Simulation study shows the minimization in total execution time as well as reduction in energy consumption.

The genetic algorithm is proposed in [21] for service workflows, mobility-enabled and fault-tolerance offloading system. In this paper, the issue of connectivity of mobile networks with portable devices is considered for offloading. The near-optimal solutions have been witnessed with regard to the problem size. A Mobile Application's Offloading (MAO) algorithm is proposed in [18] which considered CPU load and battery life. They considered various interactive and delay tolerant mobile applications for experimental work. Results show the significant energy gain by offloading some applications to remote server. A novel offloading algorithm called dynamic programming with hamming distance termination is presented in [22]. The algorithm aims to good solutions with low computational overhead. When the network transmission bandwidth is high, the algorithm offloads the as many as possible tasks to the cloud. This improves the total execution time and extends the energy of the mobile device.
The authors in [19] proposed a stable method to effectively and dynamically partition a given application into local and remote parts called l min-cost offloading partitioning algorithm. It significantly reduces execution time and energy consumption by optimally distributing tasks between mobile devices and cloud servers.

Cost and time constraint task partitioning and offloading algorithm, multi-site task scheduling algorithm based on teaching, learning-based optimization and the energy saving on multi-sites using DVS technique is proposed in [23]. The algorithms deal with the trade-off between time and cost for the task partitioning, offloading and time efficient scheduling on multi-sites.

A health care application model is developed in [24]. This model categorises the data into normal, critical and super critical. This model helps to process the patient's data efficiently in terms of aggregation efficiency, end-to-end delay, and total transmission time. An offloading technique for contextual network conditions are used in

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

deciding whether to offload to the cloud or not is proposed in [25]. Depending on the the current network conditions, the proposed model takes the decision for offloading. The concept of delayed offloading is also implemented for energy saving when network conditions are not good.

Factors affecting the energy consumption of mobile clients is discussed in [26].They discussed the the balance between local and remote computing for mobile devices. Also showed the impact of trade-offs among the data communication patterns, technologies used and workload.A computation offloading scheme on handheld devices is presented in [17]. They used clustering to handle the program partitioning and execution. Results showed the improvement in energy consumption and performance through computation offloading. To extend the battery life for portable computers through offloading is presented in [16]. They considered the execution time of the program. If the program can be executed within the time limit than there is no offloading, otherwise program is offloaded to the server. Results showed the significant improvement in energy saving. An android and Hadoop based prototype 'Phone Cloud' is developed in [27] for energy saving on smartphones. It offloads the computation of an application running on smartphones to the cloud. The focus is to improve the energy efficiency of smartphones as well as improves the application's performance through reducing its execution time. A queueing model is used to minimize a weighted sum of energy consumption in [28]. An analytical model in MCIoT environment is discussed in [29]. They focused on the suitable partition rate for the application as well as considered network parameters that affect the performance of the application. The appropriate partition rate effects execution time and energy consumption on smart mobile devices and on the cloud.

In [30], focus on the real-time video applications that have stringent delay and bandwidth constraints. Paper discussed the performance and energy efficiency of representative mobile cloud applications under dynamic wireless network channels. Developed a generic model for energy-efficient computation offloading for real-time video applications. A comparative experimental study has been presented in [31] to compare the rate of energy consumption and total execution time in mobile cloud computing and local devices.

The previous work on energy efficient mobile cloud offloading, no application is partitioned and offloaded on the basis of levels. The motive of this paper is to reduce the communication overhead (due to dependency among the tasks in an application) as well as completion of execution of the application within the deadline and available maximum energy.

III. SYSTEM AND COMPUTATION MODEL

To make the system energy and time efficient the communication and computation models play a key role in mobile cloud computing, therefore this section gives the introduction about the communication model, computation models and proposed methodology in detail.

A. Application model

An application is assumed to be consists of dependent tasks and implemented by the directed acyclic graph (DAG) [32]. The DAG,G = (T,E,P) , where T is the set of tasks to be processed, is the set of edges indicating the precedence constraints between the tasks and P is the set of P processors. $CT_{i,j}$ is the computation time of task i on processor j. It is assumed that T_i is the parent task of T_j, and T_j cannot be executed until its parent task (T_i) is not executed completely. A task with no predecessors is called an entry task (T_{entry}), while the task with no successors is known to be exit task (T_{exit}) as shown in Fig. 2.

B. Computation Model

The computation time (CP_i) of a task t_i on a processor p_j is calculated using the length of the task (Len_i, Millions of instructions per second) and capacity of the processor (Cap_j) to execute the number of instructions per unit time[33]. Computation time for task i is expressed as

$$CP_i = \frac{Leni}{Capj} \quad (1)$$

The computation time of all the tasks on the level l is derived as

$$TT_l = \sum_{i=1}^{l} CP_i \quad (2)$$

where i is the number of tasks on the lth level.

C. Communication Model

In communication time, output data produced by the parent level (outputfilesize) and bandwidth (BW) between the two processors Pi and Pj on which the tasks are scheduled can be expressed as

$$CT_l = \frac{outputfilesize}{BW} \quad (3)$$

Communication time (CTl) is the time of transferring the data from parent level i to child level j is considered only if both the levels are scheduled on different processors. Further, ifparent and child level are processed on the same processor then, CTi,j = 0. In Fig. 2, the value showed on the edges is the communication time.

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

The overall execution time of lth level is the summation of computation time of lth level and communication time can be expressed as

$$ET_l = CT_l + TT_l \quad (4)$$

D. Energy Model

The task execution can have two types of energy consumption. One is dynamic energy dissipation ($Energy_{dynamic}$) while the other one is static energy dissipation ($Energy_{static}$). The dynamic power dissipation ($Power_{dynamic}$) is considered to be one of the main factors of energy consumption. The is related to the voltage (v) and frequency (f) as calculated as

$$Power_{dynamic} = k * v^2 * f \quad (5)$$

where k is the effective switched capacitance depending on the chip architecture. We set $k = 10^{-11}$ that energy consumption is consistent with the measurements in [26]. The energy consumption for the mobile device can be determined as

$$EC_{l,m} = Power_{dynamic} * TT_l \quad (6)$$

The energy consumption for cloud server can be determined as

$$EC_{l,c} = Power_{dynamic} * ET_l \quad (7)$$

where i is the number of tasks on the lth level.

Using execution time and energy consumption, the cost for a mobile device can be defined as

$$cost_{l,m} = a_1 TT_l + a_2 EC_{l,m} \quad (8)$$

where a_1 and a_2 are weights of total execution time and energy consumption. The value of a_1 and a_2 should be greater than or equal to zero and less than or equal to 1 ($0 \le a_1 \le 1$ and $0 \le a_2 \le 1$) We assume that $a_1 + a_2 = 1$.

Similarly, the cost for cloud server execution is expressed as

$$cost_{l,c} = b_1 ET_l + b_2 EC_{l,c} \quad (9)$$

where b1 and b2 are weights of total execution time and energy consumption. The value of b1 and b2 should be greater than or equal to zero and less than or equal to 1 ($0 \le b1 \le 1$ and $0 \le b2 \le 1$) . We assume that $b1 + b2 = 1$. The value of a1 and a2 can be adjusted as per the availability of resources on the mobile device. The value of b1 and b2 can be adjusted as per the communication time and computation time on the cloud server.

E. Proposed Methodology

As discussed in literature work, the offloading is mainly exploited on task basis. Consequently, there is an increase in communication overhead during the offloading. To address this issue, in this work, the offloading scheme is based on level wise offloading of the application instead of task basis. A novel algorithm namely, HLTO is proposed and developed for improving the energy efficiency of the mobile application. The motive behind the proposed HTLO algorithm is to reduce the communication overhead, which in turn reduces the overall execution time and energy consumption of the application. The flowchart of the proposed algorithm is shown in Fig. 1. Initially, information regarding the number of levels and tasks per level is collected. Then, the value of deadline (D) and maximum available energy (E) is set according to the available resources on the mobile device and network conditions in MCC environment. Next, the value of execution time, communication time and energy consumption on the mobile device and cloud server are calculated using Eq. (1) to Eq. (7) as discussed below. Now, these values are checked against the objective function in Eq. (11) and the decision of offloading is taken.

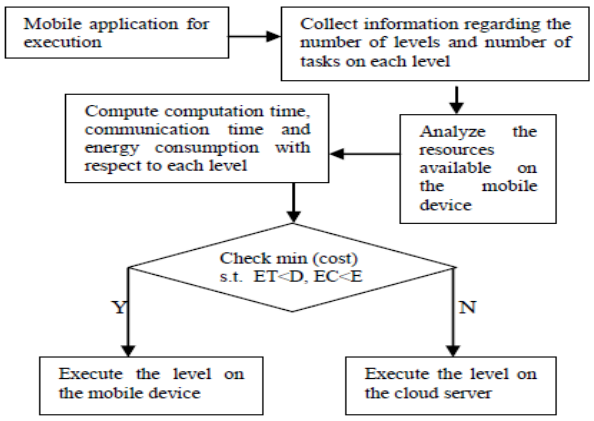

Fig. 1. The flowchart of the proposed offloading scheme

IV. PROBLEM FORMULATION

The total energy-efficiency cost with respect to level l at mobile device and cloud server execution can be given by

$$cost_l = w_l cost_{l,c} + (1 - w_l) cost_{l,m} \quad (10)$$

where w_l donates the offloading selection factor making a decision on a mobile device or cloud server. For an application with the set of dependent tasks with L levels as {1, 2…l}, we aim to provide an optimal solution w= { $w_1, w_2..w_l$ } to achieve the minimum energy efficiency

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

cost. Thus, the objective of the proposed work is the mapping of the application's subtasks to the available processors i.e., between the mobile device processor and cloud server with minimum energy efficiency cost. The energy efficiency cost minimization problem can be formulated as:

$$min \sum_{i=1}^{L} cost_i \quad (11)$$

Such that the following constraints should be satisfied:

(1) Time constraint:

$$\sum_{i=1}^{L} wi(ET_C) + (1 - wi)(ET_m) \leq D \quad (12)$$

(2) Energy constraint:

$$\sum_{i=1}^{L} wi(EC_C) + (1 - wi)(EC_m) \leq E \quad (13)$$

(3) Selection constraint:

$$w_i \in \{0, 1\} \quad (14)$$

The time constraint (12) implied that the total completion time of all the levels should be less than or equal to the deadline D. The deadline of an application is the maximum time required to complete its execution. The energy constraint (13) reflects that the energy consumption of all the levels should be less than or equal to the maximum available energy E. The selection constraint w_i specifies whether $level_i$ is processed on the cloud or on the mobile device.

The objective function in (11) with respect to selection strategy w_l can be expressed into the following form:

$$\min_{wi} \quad w_1 cost_{l.c} + (1 - w_l) cost_{l,m} \quad (15)$$

If $cost_{l.c} > cost_{l,m}$ then set the value of $w_1 = 0$ as a result minimum of (15) will be achieved. Otherwise reverse is true with $w_1=1$. The offloading strategy means when the cost of processing on the cloud is less than the mobile device then the level is offloaded to the cloud server. In (15) the value of w_1 is selected according to the characteristics of the mobile application.

In this paper, we have considered two types of mobile applications i.e., delay sensitive and delay tolerant applications. For delay sensitive applications the value of deadline is hard. The overall completion time of the application should be within the deadline whereas in delay tolerant application the completion time is kept soft. On the basis of these assumptions, the value of w_1 is decided. In Table 1, some examples of some delay sensitive applications are represented[34].

In Fig. 2, applications graph are represented with less inter task dependency to high inter task dependency. As the inter dependency among the tasks increases the delay sensitive of the application increases.

Table I. Delay Sensitive Applications

Application name	Delay sensitivity
File transfer	Low
Web traffic	Moderate
Transaction based	Moderate
Voice over IP	High
Video conferencing	High
Gaming	High

A. The heuristic approach based level-wise task offloading

In this section, the proposed heuristic based level wise task offloading (HLTO) approach is presented. It focuses on efficient execution of an application on cloud server or on mobile device such that overall execution time and energy consumption lies within the deadline and maximum available energy. The processing capabilities of the mobile device are slower than the processing capabilities of the server on the cloud and hence the computation time on mobile device is higher than the computation time on the cloud server. In Table 2, processor 1 and processor 2 are assumed to be on mobile device and cloud respectively. As the availability of resources on the mobile device is high, it can execute more levels in a graph. In Algorithm 1, the pseudo code of HLTO is explained. Initially, for each level, the number of tasks is entered (line 1- line 3). Then, the level wise execution time of the tasks and energy consumption is calculated on mobile device and cloud server (line 7). The overall cost on the cloud server and on the mobile device is calculated (line 8). To minimize the objective function in Eq. (11), the cost of the cloud server and the mobile device is compared (line 9). If the cost of computation on the cloud is less than the mobile device than the level is offloaded to the cloud server and weighting factor w_1 is set to 0 otherwise reverse is true (line 9 –line 12). The time complexity of the algorithm is 0 (n * log n)where log n is the time taken to process the levels and n is the time taken to process the number of tasks on each level.

Algorithm 1 Heuristic approach for level-wise task offloading (HLTO)

Input:	number of levels, E, D
Output:	optimal application execution policy within the deadline D and energy E
1.	
2.	a[k]= enter the number of tasks on each level
3.	end for
4.	Initialize k=1
5.	for i=1 to L
6.	while j<=a[k]
7.	compute ,, by (1)-(7) respectively
8.	compute by (8)-(9) respectively
9.	if

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

10.	
11.	Else
12.	
13.	end if
14.	a[k]++
15.	end while
16.	end for

B. Numeric Example

To prove the efficiency of HLTO algorithm against K-M LARAC algorithm, we have considered an application having eight subtasks as shown in Fig.2. Processing time with respect to mobile device and cloud server is shown in Table 2. As per the K-M LARAC algorithm, the shortest path of the graph is 1, 4, 6 and 8. The tasks in the shortest path are being processed on the mobile device. While the remaining tasks are processed on the cloud server. As per the K-M LARAC algorithm, tasks on the shortest path (1, 4, 6 and 8) are processed on the mobile device. However, the remaining tasks (2, 3, 5 and 7) are offloaded to the cloud server for processing. The communication time is being added to the tasks if its parent task is processed on different processor. As task 1 is chosen to be processed on the mobile device. So, the communication time is added with the computation time for tasks 2 and 3, as they are children of task1 and are processed on the cloud server. For task 2, communication time (7 units) is added to its computation time (10 units). Therefore, the total time to process the task 2 on the cloud is 17 units. Similarly, total time of the remaining tasks is calculated for the existing algorithm. The total time for this graph is 251. On the other hand, as per the HLTO algorithm, level two is assumed to be executed on the mobile device and the rest of the levels are offloaded to the cloud. As per the availability of the resources on the mobile device, the maximum available energy value (E) of the mobile device can be adjusted. According to our approach, level one is on cloud and communication and computation time is 48, level two is on mobile device, therefore, communication and computation time is 58, level three and four is on cloud and communication and computation time is 58 and 14, respectively. The total communication and computation time is 178 which are better than the K-M LARAC approach.

Table 2. Computation Time (S)

Tasks	Processor 1 (mobile device)	Processor 2 (cloud server)
1	13	11
2	15	10
3	12	9
4	16	11
5	15	11
6	12	9
7	14	10
8	15	14

V. SIMULATION RESULTS

In this section, we have evaluated the performance of the HLTO algorithm in comparison with K-M LARAC algorithm. The experimental environment for the proposed algorithm consists of Intel(R) Core i7-4702MQ CPU 2.20 GHz, 16 GB RAM and implemented in MATLAB R20015. The five random graphs with 8, 16, 20, 24, and 27 tasks are generated in this environment as shown in Fig. 2. The vertex and edges weights are Gaussian distributed with assigned means. The processing speed of the mobile device is lower than the cloud server. The comparison metrics of the K-M LARAC algorithm and HLTO algorithm are: (i) execution time (ii) communication time (iii) mobile device energy consumption(iv) total energy consumption of the application.

A. Execution time

Fig. 3 shows the execution time of the mobile application. As per the K-M LARAC algorithm, the tasks under the shortest path are executed on the mobile device and the rest of the tasks are offloaded on the cloud. As shown in Fig. 2, the shortest path for the 12 node graph is 1,4,8,11,12, for the 16 node graph is 1, 3, 10, 14, 16, for the 20 node graph is 1, 2, 9, 15, 20, for the 24 node graph is 1, 7, 17, 23, 24 and for the 27 node graph is 1, 4, 14, 25, 27. According to this, the execution time, communication time and energy consumption of an application are high in K-M LARAC algorithm as compared to the HLTO algorithm. As per the existing algorithm, the selected path of the graph which satisfies either time, cost or energy constraint remains on the mobile device, while the rest of the tasks are offloaded to the cloud server. But in HLTO algorithm, execution time, communication time and energy consumption of each level of the graph are compared against the D and E. If the condition of the objective function as defined in Eq. (11) is satisfied only then all the tasks of that level are offloaded to cloud server.

B. Communication time

The communication time consumption in HLTO algorithm and K-M LARAC algorithm is shown in Fig. 4. In our approach, the communication time is the time specified for each level instead of for each task. Communication time is added only if the parent task and child tasks are on the different processors. In Fig. 2, for level wise offloading, the maximum communication time is considered for each level among the communication time mentioned on the edges. In the existing approach, a particular path is selected to execute on the mobile device and all other dependent tasks of the selected path are offloaded to the cloud server. Thus, the communication

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

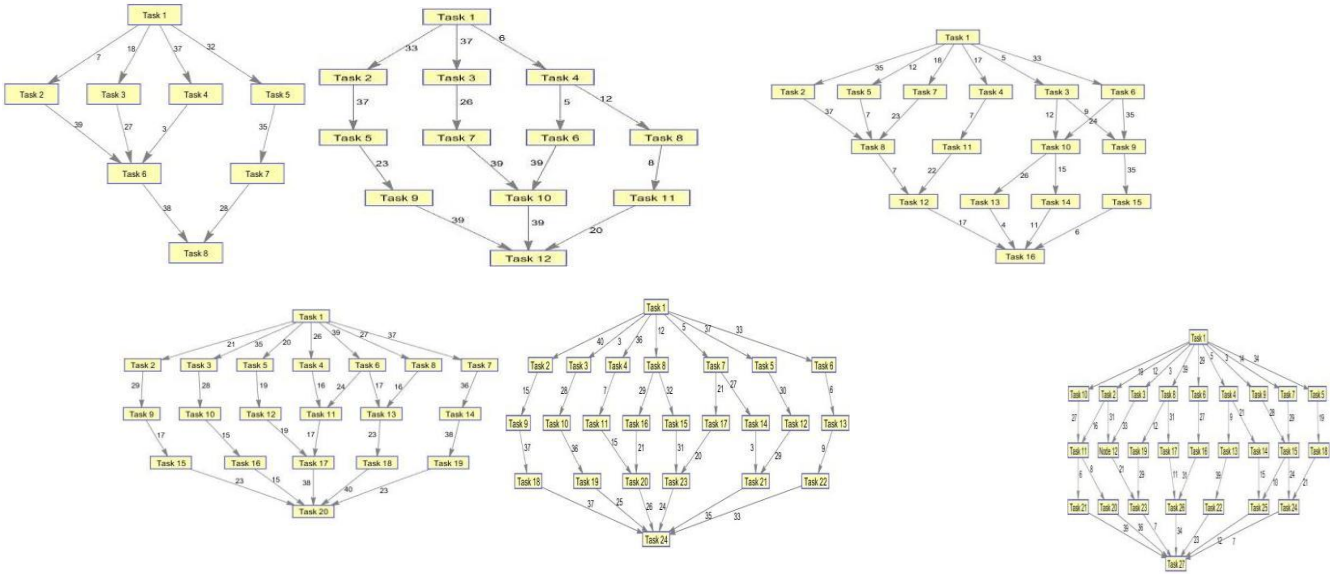

Fig. 2. Different task graphs used for simulation.

overhead between the parent task processing and child task processing increases. But in our algorithm, it helps to reduce the communication overhead by executing the all tasks of a level on the same processor. As shown in the Fig. 4, for small graphs like graphs with 8 and 12 tasks have less communication overhead but as the dependency among the tasks increases, accordingly, communication time increases.

A. Mobile device energy consumption and total energy consumption of the application

Energy consumption on the mobile device and overall energy consumption of the application is presented in Fig. 5 and Fig. 6 respectively. Energy consumption of the application is dependent on the execution time and communication time. The energy consumption on the mobile device and total energy consumption are dependent on the results of section 5.3 and section 5.4. Energy consumption for mobile device and total energy consumption of the application is calculated using the Eq. (6) and Eq. (7). The results show the less energy consumption with graphs of 8 and 12 tasks have less number of tasks and less tasks interdependencies. But it is increasing with increase in tasks and their inter dependencies.

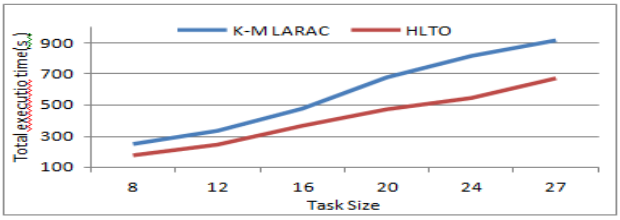

Fig. 3. Total execution time.

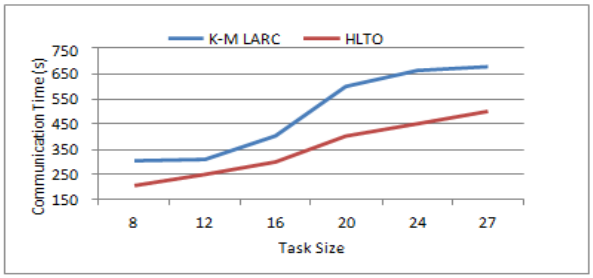

Fig. 4. Total communication time.

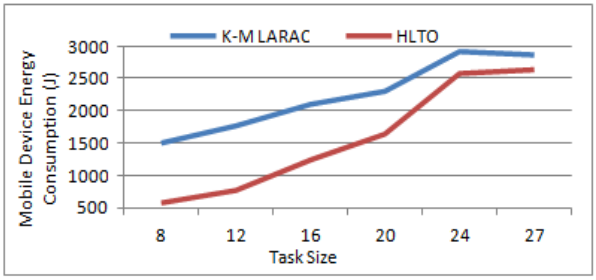

Fig. 5. Mobile device energy consumption.

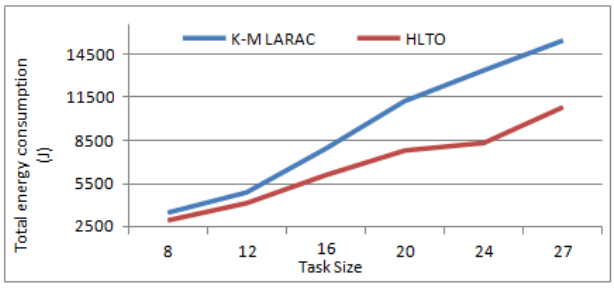

Fig. 6. Total energy consumption.

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

VI. Discussion

In the proposed heuristic approach for level-wise task offloading technique, we have considered a mobile application as a task graph with dependency among the tasks. In the mobile cloud computing environment, the communication time is a critical parameter for task offloading. The overall time of the application execution required to be completed within the deadline of the application as well as the overall energy consumption of the application must be within the maximum available energy on the mobile device. To achieve this objective, we have considered the level-wise task offloading technique for an application. The main benefit of this technique is that it helps to reduce the communication time which ultimately improves the overall execution time of the application. The simulation experiments show the better performance of the proposed work with respect to execution time and energy consumption.

VII. Conclusion

The performance of the mobile applications can be improved by using computational offloading methods. The battery life, as well as other resources of the mobile devices, can be enhanced by using the power of cloud computing. Identifying the computational complex tasks from the mobile application is the key component to take the optimal offloading decision. The K-M LARAC algorithm has not considered the effect of the interdependency of the tasks in a mobile application.

There are number of factors that influence the performance of offloading in the mobile cloud computing environment. Wireless network and application characteristics are the key parameters that affect the execution of the application in MCC. In wireless network, availability of the bandwidth rate plays the crucial role. As the bandwidth rate is high, less communication time will be consumed. But if the interdependency among the tasks in an application is high then the decision about the selection of tasks offloading is highly significant.

In this paper, we proposed an algorithm for time and energy efficient task offloading. The algorithm considers the interdependency among the tasks within an application. As the interdependency increases, delay sensitivity increases. The availability of resources on the mobile device is analyzed and considered as the maximum available energy. If the maximum available energy is enough to execute the level on the mobile device, then no offloading is done. Otherwise, that level is selected for offloading. The results show the improvement in efficient offloading of the application within the deadline and maximum available energy as compared to the state-of-the-art. In future work, more parameters of the mobile device like memory, storage etc. can be considered with battery life to take the offloading decision. Similar kind of work can be performed with the concept of mobile edge computing.

REFERENCES

[1] M. Ayad, M. Taher, and A. Salem, “Real-time mobile cloud computing: A case study in face recognition,” Proc. - 2014 IEEE 28th Int. Conf. Adv. Inf. Netw. Appl. Work. IEEE WAINA 2014, pp. 73–78, 2014.

[2] D. Meil??nder, F. Glinka, S. Gorlatch, L. Lin, W. Zhang, and X. Liao, “Using mobile cloud computing for real-time online applications,” Proc. - 2nd IEEE Int. Conf. Mob. Cloud Comput. Serv. Eng. MobileCloud 2014, pp. 48–56, 2014.

[3] I. Technologies, “Overview of Offloading in Smart Mobile Devices for Mobile Cloud Computing,” vol. 5, no. 6, pp. 7855–7860, 2014.

[4] H. Qian and D. Andresen, “Extending Mobile Device’s Battery Life by Offloading Computation to Cloud,” Proc. - 2nd ACM Int. Conf. Mob. Softw. Eng. Syst. MOBILESoft 2015, pp. 150–151, 2015.

[5] lan L. Bhaskar Prasad Rimal, Eunmi Choi, “2009 Fifth International Joint Conference on INC , IMS and IDC,” Fifth Int. Jt. Conf. INC, IMS IDC, pp. 44–51, 2009.

[6] D. Yao et al., “Energy Efficient Task Scheduling in Mobile Cloud Computing To cite this version : HAL Id : hal-01513756,” pp. 0–12, 2017.

[7] K. Liu, J. Peng, H. Li, X. Zhang, and W. Liu, “Multi-device task offloading with time-constraints for energy efficiency in mobile cloud computing,” Futur. Gener. Comput. Syst., vol. 64, pp. 1–14, 2016.

[8] A. U. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, “A survey of mobile cloud computing application models,” IEEE Commun. Surv. Tutorials, vol. 16, no. 1, pp. 393–413, 2014.

[9] M. V. Barbera, S. Kosta, A. Mei, and J. Stefa, “To offload or not to offload? the bandwidth and energy costs of mobile cloud computing,” in Proceedings - IEEE INFOCOM, 2013.

[10] M. Shiraz, A. Gani, A. Shamim, and S. Khan, “Energy Efficient Computational Offloading Framework for Mobile Cloud Computing,” pp. 1–18, 2015.

[11] O. Chakroun and S. Cherkaoui, “Resource Allocation for Delay Sensitive Applications in Mobile Cloud Computing,” 2016 IEEE 41st Conf. Local Comput. Networks, pp. 615–618, 2016.

[12] M. Abdallah, S. Université, I. De Paris, K. Chen, and A. Sinica, “Delay-Sensitive Video Computing in the Cloud : A Survey,” vol. 14, no. 3, 2018.

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731

[13] E. Ahmed, A. Gani, M. Khurram, R. Buyya, and S. U. Khan, "Journal of Network and Computer Applications Seamless application execution in mobile cloud computing : Motivation , taxonomy , and open challenges," J. Netw. Comput. Appl., vol. 52, pp. 154–172, 2015.

[14] J. O. F. Information and C. P. Vala, "Improvement of Dynamic Partitioning Technique in Mobile Cloud Computing," pp. 323–325.

[15] V. Haghighi and N. S. Moayedian, "An offloading strategy in mobile cloud computing considering energy and delay constraints," IEEE Access, vol. 6, pp. 11849–11861, 2018.

[16] C. Xian, Y. H. Lu, and Z. Li, "Adaptive computation offloading for energy conservation on battery-powered systems," Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS, vol. 1, 2007.

[17] C. Wang and Z. Li, "A computation offloading scheme on handheld devices," J. Parallel Distrib. Comput., vol. 64, no. 6, pp. 740– 746, 2004.

[18] A. Ellouze, M. Gagnaire, and A. Haddad, "A mobile application offloading algorithm for mobile cloud computing," Proc. - 2015 3rd IEEE Int. Conf. Mob. Cloud Comput. Serv. Eng. MobileCloud 2015, pp. 34–40, 2015.

[19] H. Wu, W. Knottenbelt, K. Wolter, and Y. Sun, "An optimal offloading partitioning algorithm in mobile cloud computing," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9826 LNCS, pp. 311–328, 2016.

[20] F. H. Tseng, H. H. Cho, K. Di Chang, J. C. Li, and T. K. Shih, "Application-oriented offloading in heterogeneous networks for mobile cloud computing," Enterp. Inf. Syst., vol. 12, no. 4, pp. 398–413, 2018.

[21] S. Deng, L. Huang, J. Taheri, and A. Y. Zomaya, "Computation Offloading for Service Workflow in Mobile Cloud Computing," IEEE Trans. Parallel Distrib. Syst., vol. 26, no. 12, pp. 3317–3329, 2015.

[22] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," IEEE Trans. Wirel. Commun., vol. 11, no. 6, pp. 1991–1995, 2012.

[23] R. Kumari, S. Kaushal, and N. Chilamkurti, "Energy conscious multi-site computation offloading for mobile cloud computing," Soft Comput., vol. 22, no. 20, pp. 6751–6764, 2018.

[24] R. Kumari and S. Kaushal, "Application Offloading Using Data Aggregation in Mobile Cloud Computing Environment."

[25] M. Akram and A. Elnahas, "Energy-aware offloading technique for Mobile cloud computing," Proc. - 2015 Int. Conf. Futur. Internet Things Cloud, FiCloud 2015 2015 Int. Conf. Open Big Data, OBD 2015, pp. 349–356, 2015.

[26] A. P. Miettinen, "Energy efficiency of mobile clients in cloud computing," HotCloud'10 Proc. 2nd USENIX Conf. Hot Top. cloud Comput., pp. 4–11, 2010.

[27] F. Xia, F. Ding, J. Li, X. Kong, L. T. Yang, and J. Ma, "Phone2Cloud: Exploiting computation offloading for energy saving on smartphones in mobile cloud computing," Inf. Syst. Front., vol. 16, no. 1, pp. 95–111, 2014.

[28] H. Wu and K. Wolter, "Tradeoff analysis for mobile cloud offloading based on an additive energy-performance metric," Proc. 8th Int. Conf., 2014.

[29] R. Kumari and S. Kaushal, "Energy efficient approach for application execution in mobile cloud IoT environment," Proc. Second Int. Conf. Internet things, Data Cloud Comput. - ICC '17, pp. 1–8, 2017.

[30] L. Zhang, D. Fu, J. Liu, E. C. H. Ngai, and W. Zhu, "On Energy- Efficient Offloading in Mobile Cloud for Real-Time Video Applications," IEEE Trans. Circuits Syst. Video Technol., vol. 27, no. 1, pp. 170–181, 2017.

[31] M. Ahmadi, N. Khanezaei, S. Manavi, F. F. Moghaddam, and T. Khodadadi, "A comparative study of time management and energy consumption in mobile cloud computing," Proc. - 2014 5th IEEE Control Syst. Grad. Res. Colloquium, ICSGRC 2014, pp. 199–203, 2014.

[32] L. Guan, X. Ke, M. Song, and J. Song, "A survey of research on mobile cloud computing," Proc. - 2011 10th IEEE/ACIS Int. Conf. Comput. Inf. Sci. ICIS 2011, pp. 387–392, 2011.

[33] Q. Wang, S. Guo, J. Liu, and Y. Yang, "Sustainable Computing : Informatics and Systems Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing," Sustain. Comput. Informatics Syst., vol. 21, pp. 154–164, 2019.

[34] I. Chantaksinopas, W. Lee, A. Prayote, and P. Oothongsap, "Delay-Sensitive Applications in VANET and Seamless Connectivity : The Limitation of UMTS Network," vol. 12, no. 10, pp. 54–61, 2012.

ISSN (Print): 2204-0595
ISSN (Online): 2203-1731