Engineering Science and Technology, an International Journal 19 (2016) 260–270

HOSTED BY

ELSEVIER

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: http://www.elsevier.com/locate/jestch

Full Length Article

Low power offloading strategy for femto-cloud mobile network

Anwesha Mukherjee, Debashis De *

Department of Computer Science and Engineering, West Bengal University of Technology, B.F.-142, Salt Lake, Sector-1, Kolkata 700064, India

A R T I C L E I N F O

Article history:
Received 4 May 2015
Received in revised form
28 July 2015
Accepted 9 August 2015
Available online 1 September 2015

Keywords:
Low power
Femto-cloud
Offloading
Security

A B S T R A C T

Nowadays offloading is a popular method of mobile cloud computing where the required computation takes place remotely inside the cloud. But whether to process an application inside the mobile device or to the cloud is a challenging issue because communication with the cloud involves latency and power consumption. This paper has proposed a method of decision making regarding whether to offload or not-to-offload an application to the cloud. According to the proposed strategy, application is offloaded only if it results in lower power consumption than local execution within the mobile device itself. If this condition is satisfied, computation time and deadline of the job are considered as the basic parameters to decide whether to offload or not. Experimental results demonstrate that the proposed offloading algorithm reduces the power consumption to approximately 3–32%. To achieve power-efficiency and security both, femto-cloud architecture is used in the proposed work. In this case offloading from the mobile device to the cloud takes place through the low power and secure femtocell base station. Simulation results present that using femto-cloud architecture 70–83% and 52–66% power savings are achieved than using macrocell and microcell base stations respectively while offloading an application to the cloud.

1. Introduction

Green mobile network indicates a low power cellular network without compromising with the system performance. Nowadays mobile devices are not only used for voice call or message services but also for internet service. The number of mobile web users has increased explosively. Various applications like mobile gaming, mobile health care, mobile commerce require internet connectivity and their execution reduces the battery life of the device. Mobile devices have limited processing power, limited battery life, limited resource and limited storage capacity. To deal with these problems, mobile cloud computing comes into the scenario. Mobile cloud computing (MCC) controls unified elastic resources of different clouds and network technologies toward unlimited functionality, storage, and mobility in order to serve a large number of mobile devices anywhere, anytime through the internet [1]. By introducing offloading, MCC has removed the constraint of limited storage, limited processing power, resource constraint and limited battery life of the mobile device. In this case the execution of application is performed inside the cloud and then the result is sent back to the mobile device [2–4]. Hence the processing power of the resource constrained mobile device is not wasted and the battery life is also conserved.

Base stations (BSs) in a mobile network consume huge power. To reduce the power consumption by the BSs as well as to reduce the pressure of large cells while serving huge number of mobile users, the concept of femtocell came about [5,6]. Femtocell is a low power, low cost and highly secured Home Node Base Station (HNB) having coverage of 10–20 m approximately. Femtocell connects to the service provider's network through cable broadband or DSL connection. In this paper we have proposed an offloading scheme for femto-cloud network. Femtocell is selected for its power efficiency and high security. Moreover, using femtocell, high signal strength is achieved which is essential for good internet connectivity. The major contributions of this paper are:

1. A decision making method for offload or not-to-offload an application is proposed where depending on the computation time and deadline of job offloading takes place. The performance of the proposed algorithm is evaluated through experimental analysis performed using cloud servers of our university.
2. Femto-cloud architecture with bio-metric authentication is used for secure data transmission and secured cloud access.
3. Power consumption in offloading in case of femto-cloud architecture is proposed to show that using femtocell power saving is achieved than macrocell and microcell base stations.

This article is organized as: section 2 presents the review works on power efficiency and offloading along with the motivations of the proposed work, section 3 presents the proposed offloading method for

* Corresponding author. Tel.: +919830363215; fax: 033 2321 8776..
E-mail address: dr.debashis.de@gmail.com (D. De).
Peer review under responsibility of Karabuk University.

http://dx.doi.org/10.1016/j.jestch.2015.08.001
2215-0986/

femto-cloud network with the power consumption model, performance of the proposed offloading algorithm and femto-cloud architecture is evaluated in section 4 and finally we conclude in section 5.

2. Related works

Energy efficiency in mobile network is an emerging concern for the operators to maintain profitability and reduce pollution due to power transmission [6]. As BSs are the largest energy consumers in a cellular system, their deployment depending on network traffic and coverage is an important issue. In References 6 and 7 different types of cells like microcell, picocell and femtocells are deployed to develop a “green” i.e. energy efficient mobile network. The power consumption models for macrocell, microcell, picocell and femtocell networks are discussed in References 6–10. Femtocell is considered as an emerging technology for next generation wireless network as it can increase capacity and offload the overlay macrocell traffic [11–15]. Another attractive feature of femtocell is high security. In femtocell, security gateway exists which protects data during transmission as well as acts as firewall to prevent unauthorized data from entering into the system [16–18].

Not only the BSs but also the mobile devices are required to be energy-efficient. Hence energy-efficiency is also an emerging issue for the end-device manufacturer. To reduce the power consumption by a mobile device to increase its battery life, offloading is a good solution [19]. MCC is a combination of mobile computing and cloud computing that allows the execution of heavy weight mobile applications outside the mobile device and inside the cloud [20–24] through offloading. Here the computation is offloaded from a mobile device to the cloud. Cloud is used as a resourceful server where the computations are performed remotely. If the time and power consumed in offloading the task including the communication time and communication power are less than the time and power consumed by the mobile device to execute the task within itself, then offloading improves performance. For users of battery-life constrained devices, the most important criterion is the energy consumed by the application execution [25]. The advantages of offloading applications from mobile devices to remote servers are discussed in Reference 26. In References 21 and 27, it is discussed whether offloading is power saving or not. A path selection strategy for offloading mobile application is proposed in Reference 28. To calculate the power consumption in offloading, communication and computation powers are considered [21,27–34]. To calculate the communication power, bandwidth and data transmission are considered whereas computation power is calculated based on the power consumed by the mobile device at unit time and the time required for the computation. This computation time is obtained by dividing the number of instructions to be executed for computation [21,27] by the speed of the computing device.

In our proposed work, offloading is performed only if power is saved. A mobile application is an executable program that is designed to run inside the mobile device or cloud. It can be a code for file creation and update, mobile games, travel applications etc. Some of the mobile applications are heavy weight and their execution requires high processing power, memory capacity, good battery life etc. Executing such applications inside the mobile device affects the battery life of the device and sometimes introduces delay. It may also happen that due to the resource constraints and limited processing power of the mobile device, it is unable to execute the application. In such a case, offloading to the cloud is required. But offloading all the applications to the cloud may cause congestion or may increase the power and delay than executing the program within the mobile device itself especially if it is a simple application requiring small processing power and memory. Our motivation is to propose an offloading decision making algorithm regarding remote or local execution of an application. To deal with this problem, we have proposed a novel offloading scheme in this paper. The motivations of the proposed work are:

1. To achieve a low power i.e. green offloading scheme with high level of security. To provide an energy-efficient mobile network has a high priority. As small cell network is energy-efficient, an offloading strategy for femtocell network is proposed in this paper. Application size is an important parameter to be considered while offloading with respect to power efficiency. These two criteria i.e. small cell network and application size are merged in the proposed work.
2. Security is another crucial point to be considered as remote execution takes place. Due to the in-build security features, femtocell plays an important role to provide secure data transmission. Retinal image recognition based bio-metric authentication is used to achieve secured access to the cloud. Thus we can summarize that the motivation of the proposed work is to provide a power-efficient and secure offloading strategy for mobile cloud network.

3. Proposed offloading strategy for femto-cloud network

3.1. System architecture of femto-cloud network

Fig. 1 presents the system architecture of femto-cloud network [20]. As presented in Fig. 1, mobile device is connected with the cloud through the femtocell. The mobile device is first registered under the femtocell to access the network service. The mobile device is connected with the femtocell through Uu interface [16]. The femtocell is connected to the core network through a security gateway (Se-GW) and HNB-Gateway (HNB-GW) and communicates through Iuh interface [35]. Registration and deregistration of the femtocell to the HNB-GW are performed using Home Node B Application Part (HNBAP) protocol [35]. The mobile devices under a femtocell are also registered and deregistered to the HNB-GW using HNBAP protocol. The user requests for a job using his or her mobile device. Then the mobile device either sends the request to the femtocell or performs a checking procedure to decide whether to offload or not to offload the job. If the femtocell receives request from the mobile device, the checking process is performed within the femtocell. If offloading takes place, the application execution is performed remotely inside the cloud. After execution the result is sent back to the mobile device through the femtocell via Se-GW.

3.2. Parameters used

The parameters used in calculating power consumption in offloading are defined in Table 1.

3.3. Power consumption in femtocell based offloading

The power consumption in offloading using femto-cloud architecture is given by,

$$P_{Cf} = P_c \cdot (I/S_C) + (P_{tsf} \cdot D_u/B_u) + (P_{trf} \cdot D_d/B_d) \tag{1}$$

where P_c, D_u, D_d, B_u, and B_d are power consumed by the cloud in executing instruction per unit time, amount of data transmitted in uplink, amount of data transmitted in downlink, uplink bandwidth and downlink bandwidth respectively. The total power consumption in offloading to the cloud using macrocell base station is given by,

$$P_{Cm} = P_c \cdot (I/S_C) + (P_{tsm} \cdot D_u/B_u) + (P_{trm} \cdot D_d/B_d) \tag{2}$$

The total power consumption in offloading to the cloud using microcell base station is given by,

$$P_{Cmi} = P_c \cdot (I/S_C) + (P_{tsmi} \cdot D_u/B_u) + (P_{trmi} \cdot D_d/B_d) \tag{3}$$

Due to smaller coverage than macrocell and microcell, the power required to transmit and receive data using femtocell is lesser, i.e.

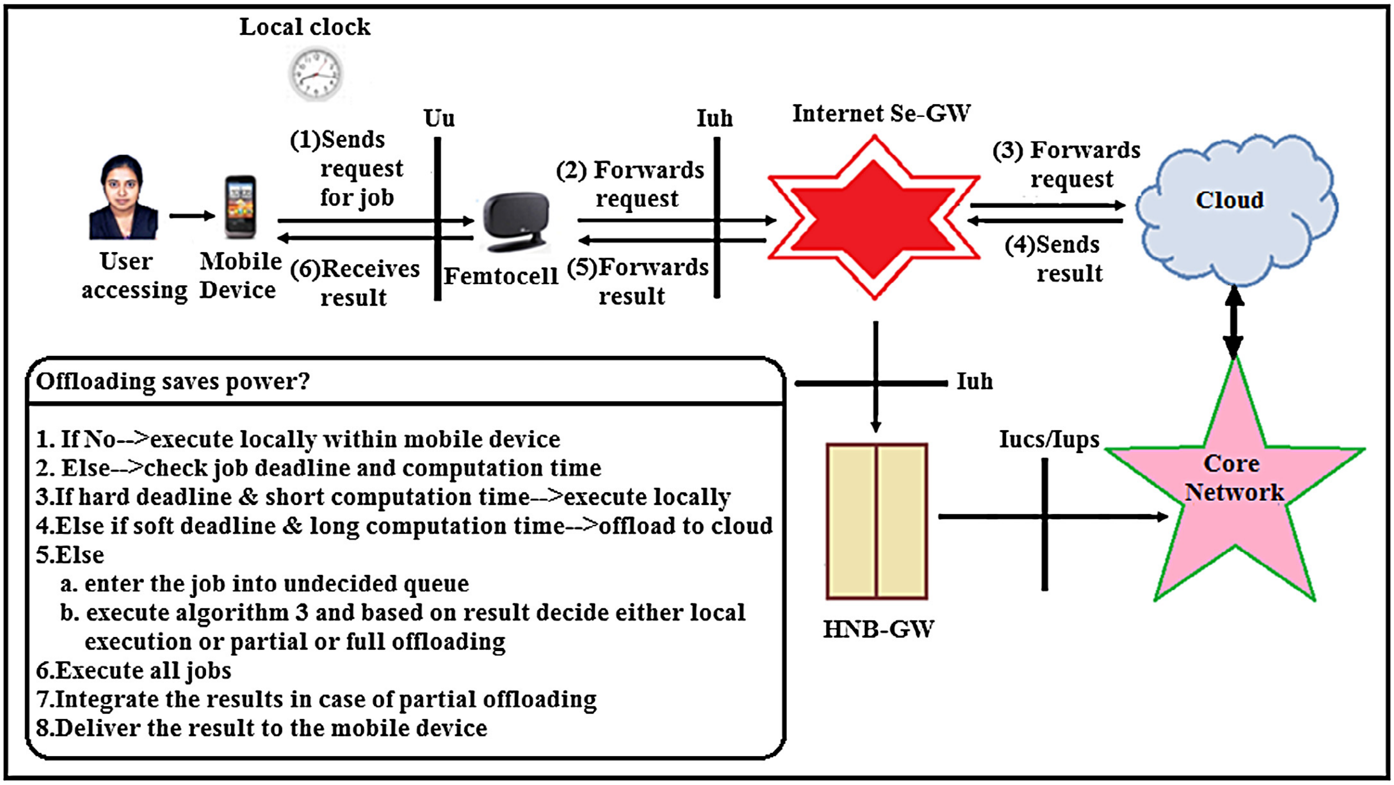

Fig. 1. System architecture of femto-cloud network.

$P_{tsf} \ll P_{tsmi} < P_{tsm}$ and $P_{trf} \ll P_{trmi} < P_{trm}$. Therefore comparing equation (1) with equations (2) and (3), it can be implied that $P_{Cf} \ll P_{Cmi} < P_{Cm}$. Fig. 2 presents the power consumed in offloading using femto-cloud architecture, macrocell and microcell base stations determined using equations (1), (2) and (3) respectively. Fig. 2 presents that offloading through the femtocell as in the femto-cloud architecture achieves approximately 70–83% and 52–66% reduction in power consumption as compared to offloading through the macrocell and microcell base stations respectively based on the assumed parameter values. As femto-cloud architecture reduces power consumption, it is used in the proposed scheme.

3.4. Offload or not-to-offload method

The computations are offloaded to the cloud through the femtocell in the proposed scheme. If the mobile device has required amount of memory capacity, processing power and battery life to execute the decision making process, the proposed offloading algorithm "Offload or Not-to-Offload" given in Table 2 is executed within the mobile device. Otherwise the proposed offloading algorithm is executed within the femtocell. Depending on the result the decision is taken whether to offload the computations to the cloud or not. If offloading saves power, then the application execution takes place remotely inside the cloud i.e. offloading is performed. Otherwise offloading does not occur. To determine the power consumption involved in executing an application, the number of instructions is considered [21,27]. The number of instructions is required to calculate the amount of time required by the mobile device in case of local execution. This amount of time is calculated by dividing the number of instructions to be executed by the speed of the mobile device. The speed of a mobile device refers to the number of instructions executed per unit time by the mobile

Table 1
Parameters used in power calculation.

Parameter	Definition	Value
P_{tsf}	Power required in data transmission to the cloud using femtocell	10–20 mW
P_{trf}	Power required in data reception from the cloud using femtocell	5–15 mW
P_{tsm}	Power required in data transmission to the cloud using macrocell	40–50 mW
P_{trm}	Power required in data reception from the cloud using macrocell	35–45 mW
P_{tsmi}	Power required in data transmission to the cloud using microcell	30–40 mW
P_{trmi}	Power required in data reception from the cloud using microcell	25–35 mW
I	Number of instructions in the job	0.5×10^6–1×10^{18}
S_C	Speed of the cloud	3.2 GHz
S_M	Speed of the mobile device	400 MHz

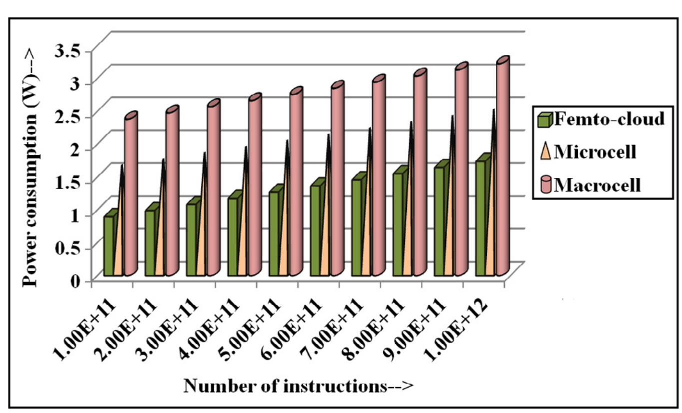

Fig. 2. Number of instructions vs. power consumption in offloading through macrocell, microcell and femtocell base stations.

Table 2
Offload or not-to-offload method.

Algorithm 1: Decision Algorithm Regarding Execution of Algorithm 2 within Mobile Device/Femtocell

1. Start
2. Check the memory capacity, processing power and battery life of the mobile device
3. **If** the mobile device is able to calculate the power consumption required for offloading, execute Algorithm 2 within the mobile device
4. **Else** execute Algorithm 2 within the femtocell under which the mobile device is registered
5. **End If**
6. **End**

Algorithm 2: Queue Assignment Algorithm

1. Start
2. Set t=T, i=1
3. **While**($t<=T+t_s$)
4. Mobile device requests for execution of job J_i
5. Calculate the power consumed in execution of J_i in case of offloading and locally within the mobile device. Let the power consumed in offloading and local execution are P_{Cf} and P_M respectively calculated using equation (1) and (4)
6. **If**($P_{Cf}>P_M$)
7. Insert J_i into the Local queue
8. **Else if**(deadline of J_i==HARD && computation time of J_i==SHORT)
9. Insert the job J_i into the Local queue
10. **Else if** (deadline of J_i==SOFT && computation time of J_i==LONG)
11. Insert the job J_i into the Remote queue
12. **Else** Insert the job J_i into the Undecided queue and Execute Algorithm 3
13. **End If**
14. Set i=i+1, t=t+1
15. **End while**
16. Execute all the jobs as queued
17. Execute Algorithm 4 and deliver output
18. **End**

Algorithm 3: Reassignment of Jobs from Undecided Queue

1. Start
2. Set k=1
3. Consider an empty set Segmented
4. **While** (k<i)
5. **If** (J_k==J_1)
6. Run job J_k locally
7. **Else if** ($J_{k-1} \in$ Remote and $J_{k+1} \in$ Local)
8. Insert job id J_k into the set Segmented
9. Divide J_k into two segments
10. Insert the first segment to the Remote queue and the last segment to the Local queue
11. **Else if** ($J_{k-1} \in$ Local and $J_{k+1} \in$ Local)
12. Insert job id J_k into the set Segmented
13. Divide J_k into three segments
14. Insert the first and last segments to the Local queue and the middle segment to the Remote queue
15. **Else if** ($J_{k-1} \in$ Local and $J_{k+1} \in$ Remote)
16. Insert job id J_k into the set Segmented
17. Divide J_k into two segments
18. Insert the first segment to the Local queue and the last segment to the Remote queue
19. **Else if** ($J_{k-1} \in$ Remote and $J_{k+1} \in$ Remote)
20. Insert job id J_k into the set Segmented
21. Divide J_k into three segments
22. Insert the first and last segments to the Remote queue and the middle segment to the Local queue
23. **Else if** ($J_{k+1} \in$ Undecided||J_k==J_{i-1})
24. Insert J_k into the queue of J_{k-1}
25. **End if**
26. **End while**
27. **End**

(continued on next page)

Table 2 (*continued*)

Algorithm 4: Integration Algorithm
Start
If ($J_i \in$ Local)
Output is delivered to the mobile device directly
Else if ($J_i \in$ Remote)
Output is sent to the mobile device from the cloud through the secure base station femtocell
Else if ($J_i \in$ Segmented)
If number of segments is three,
If (Segment 1 $\in$ Local)
Output of Segment 2 is received by the mobile device from the cloud through femtocell and the final output is obtained by merging the received output with the output of Segment 1 and 3 which have been executed locally
Else if (Segment1 $\in$ Remote)
Outputs of Segments 1 and 3 are received by the mobile device from the cloud through femtocell and combined with the output of Segment 2 which has been executed locally to generate the final output
End if
Else if the number of segments is two,
If (Segment1 $\in$ Local)
Output of Segment 2 received by the mobile device from the cloud through the femtocell is merged with the output of Segment 1 which has been executed locally and the final output is generated
Else if (Segment1 $\in$ Remote)
Output of Segment 1 received by the mobile device from the cloud through the femtocell is merged with the output of Segment 2 which has been executed locally and the final output is generated
End if
End if
End if
End

device. This result is multiplied by the power consumed by the mobile device per unit time. In case of offloading, the number of instructions is used to calculate the amount of time required by the cloud server to execute the application [21,27]. This amount of time is calculated by dividing the number of instructions to be executed by the speed of the cloud server. The speed of a cloud server refers to the number of instructions executed per unit time by the cloud server. This result is multiplied by the power consumed by the cloud server per unit time.

If the job is executed locally, the power consumption is determined as [21,27],

$$P_M = P_m \cdot (I/S_M) \tag{4}$$

where P_m is the power consumed by the mobile device per unit time for instruction execution.

If job execution is performed to the remote cloud using femtocloud network, then the total power consumption is calculated using equation (1) and denoted as P_{Cf}.

If $P_M > P_{Cf}$, offloading to the cloud is performed. Otherwise the job is executed locally within the mobile device. In femto-cloud architecture, service cloud (SC) [36] is introduced for successful completion of jobs. SC integrates the concepts of cloud and service oriented computing (SOC). SOC is a computing paradigm which utilizes services for application development.

Fig. 3 presents the power consumed in case of offloading and without using offloading calculated using equations (1) and (4) respectively. From Fig. 3 it is observed that when number of instruction is less i.e. application size is small and the power involved in communication with the cloud is high, local execution within the mobile device is more power saving than using offloading. On the other hand, for large number of instructions i.e. large size application, offloading to the cloud is better than local execution only if the power involved in communication with the cloud is low. In our proposed scheme, offloading is performed if it is power saving. Thus depending on the application size decision should be taken regarding offloading as power consumption is directly proportional to the number of instructions to be executed i.e. size of the application.

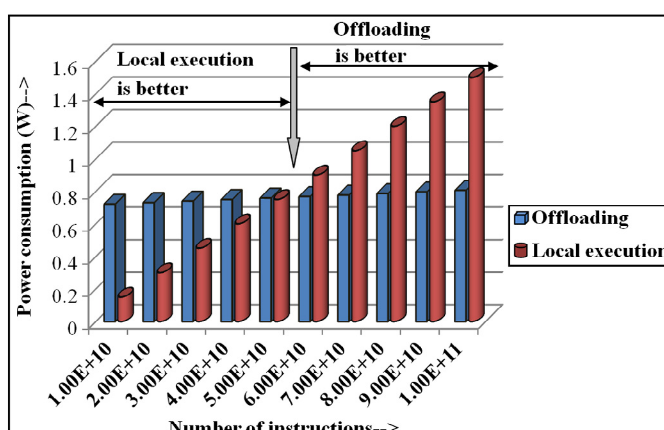

Fig. 3. Number of instructions vs. power consumption in offloading and local execution.

Fig. 3 presents that for large size application offloading gives approximately 7–34% power saving than local execution based on the assumed parameter values.

In the proposed offloading method, three queues are used to put the jobs namely, Local, Remote, and Undecided, depending on their local or remote execution. Very small time period t_s is considered. Let the current time instant on a local clock be T. The proposed method is divided into four parts:

- Algorithm 1: Decision Algorithm Regarding Execution of Algorithm 2 within Mobile Device/Femtocell.
- Algorithm 2: Queue Assignment Algorithm.
- Algorithm 3: Reassignment of Jobs from Undecided Queue.
- Algorithm 4: Integration Algorithm.

These algorithms are described in Table 2.

3.4.1. Explanation of proposed offloading method

In algorithm 1 it is decided whether the decision making process regarding offloading will be executed inside the mobile device or femtocell depending on the battery life, processing power and memory capacity of the mobile device. From algorithm 2, it is observed that offloading is performed only if it saves power. Otherwise the application execution is performed locally within the mobile device. Now it may occur that many jobs are required to be offloaded to the cloud; it can cause congestion. Hence to deal with this problem instead of offloading all the jobs, decision is taken whether to offload or not. As mentioned in algorithm 2 if the job has a hard deadline and short execution time, offloading is not performed. Otherwise if the job has a soft dead line and large execution time, the job is offloaded to the cloud. Otherwise algorithm 3 is executed to make a decision whether to offload or not. If the preceding and succeeding jobs are both executed remotely or locally, then the current job is segmented into three parts. The first and last segments are either executed remotely in the cloud or locally in the mobile device. If they are executed remotely, the middle segment is locally executed. Otherwise if they are executed locally, the middle segment is executed remotely. This is a case of partial offloading. If among the preceding and succeeding jobs, one executes locally and another remotely, then the current job is segmented into two parts. One of them is executed locally and the other one is remotely executed in the cloud. This is also a case of partial offloading. After execution of all the segments their outputs are merged to generate the final output as observed from algorithm 4.

3.5. Case study of proposed offloading scheme

In the case study ten jobs are considered and it is assumed that $t_s = 10$ ns.

3.5.1. Allocation to queue

Let's assume ten jobs have arrived in the mobile device within the time duration (t + 10) i.e. a single cycle of jobs. The proposed offloading method is used to decide whether offloading will take place or not. Suppose after executing algorithm 2 the jobs (J_1, J_2, J_3, J_4, J_5, J_6, J_7, J_8, J_9, J_{10}) have been assigned to the queues as shown in Table 3.

Fig. 4 (a) shows the allocation of the jobs in the Local and Remote queues. Jobs J_3, J_4, J_7 and J_9 are placed in the Undecided queue. Therefore algorithm 3 is executed. As J_2 is assigned to the Local queue and J_4 is in the Undecided queue, J_3 is assigned to the Local queue as per algorithm 3. As J_3 is assigned to the Local queue and J_5 to the Remote queue, J_4 is divided into two segments: the first segment is assigned to the Local queue and the second segment is assigned to the Remote queue. As J_6 and J_8 are assigned to the Remote and Local queue respectively, J_7 is divided into two segments: the first segment is assigned to the Remote queue and the second segment is assigned to the Local queue. As observed from Table 3, J_8 and J_{10} are both assigned to the Local queue. As per algorithm 3, J_9 is divided into three segments: the first and last segments are assigned to the Local queue and the middle part is assigned to the Remote queue. It is observed that J_4, J_7 and J_9 are partially offloaded to the cloud.

Table 3
Preliminary job allocation to queue.

Job ID	Is $P_{cf} > P_M$	Deadline	Computation time	Queue
J_1	Yes	Hard	Short	Local
J_2	Yes	Soft	Long	Local
J_3	No	Soft	Short	Undecided
J_4	No	Hard	Long	Undecided
J_5	No	Soft	Long	Remote
J_6	No	Soft	Long	Remote
J_7	No	Hard	Long	Undecided
J_8	Yes	Hard	Short	Local
J_9	No	Hard	Long	Undecided
J_{10}	No	Hard	Short	Local

3.5.2. Power consumption

In this case, J_1, J_2, J_3, J_8 and J_{10} are executed locally within the mobile device. Hence the power consumed for each of J_1, J_2, J_3, J_8 and J_{10} is determined using equation (4). If J_1, J_2, J_3, J_8 and J_{10} have I_1, I_2, I_3, I_8 and I_{10} numbers of instructions respectively, the total power consumed by the locally executed jobs is given by,

$$P_L = \sum_{Local} P_M = P_m \cdot [(I_1 + I_2 + I_3 + I_8 + I_{10})/S_M] \tag{5}$$

J_5 and J_6 are offloaded to the cloud. Hence the power consumed for each of J_5 and J_6 is determined using equation (1). If J_5 and J_6 have I_5 and I_6 numbers of instructions respectively, the total power consumed by these two remotely executed jobs is given by,

$$P_R = \sum_{Remote} P_{cf} = [P_c \cdot (I_5 + I_6)/S_C] + 2 \cdot [(P_{tsf} \cdot D_u/B_u) + (P_{trf} \cdot D_d/B_d)] \tag{6}$$

J_4 and J_7 are divided into two segments. One segment of each of J_4 and J_7 is offloaded to the cloud and another one is locally executed. Thus the number of instructions in case of J_4 and J_7 is divided into two parts. If J_4 and J_7 have I_4 and I_7 numbers of instructions respectively, the total power consumed by these two jobs is given by,

$$P_{S1} = [P_c \cdot ((I_4 + I_7)/2)/S_C] + 2 \cdot [(P_{tsf} \cdot D_u/B_u) + (P_{trf} \cdot D_d/B_d)] + [P_m \cdot ((I_4 + I_7)/2)/S_M] \tag{7}$$

J_9 is divided into three segments among which one is offloaded to the cloud and the other two are locally executed. Thus the number of instructions in case of J_9 is divided into three parts.

If J_9 has I_9 number of instructions, the power consumed by J_9 is given by,

$$P_{S2} = [P_c \cdot (I_9/3)/S_C] + [(P_{tsf} \cdot D_u/B_u) + (P_{trf} \cdot D_d/B_d)] + [P_m \cdot (2 \cdot I_9/3)/S_M] \tag{8}$$

Hence applying the proposed offloading method, the total power consumed for executing these ten jobs is given by,

$$P_T = P_L + P_R + P_{S1} + P_{S2} \tag{9}$$

If all these jobs are executed locally, the total power consumed is given by,

$$P_{TL} = P_m \cdot [(I_1 + I_2 + I_3 + I_4 + I_5 + I_6 + I_7 + I_8 + I_9 + I_{10})/S_M] \tag{10}$$

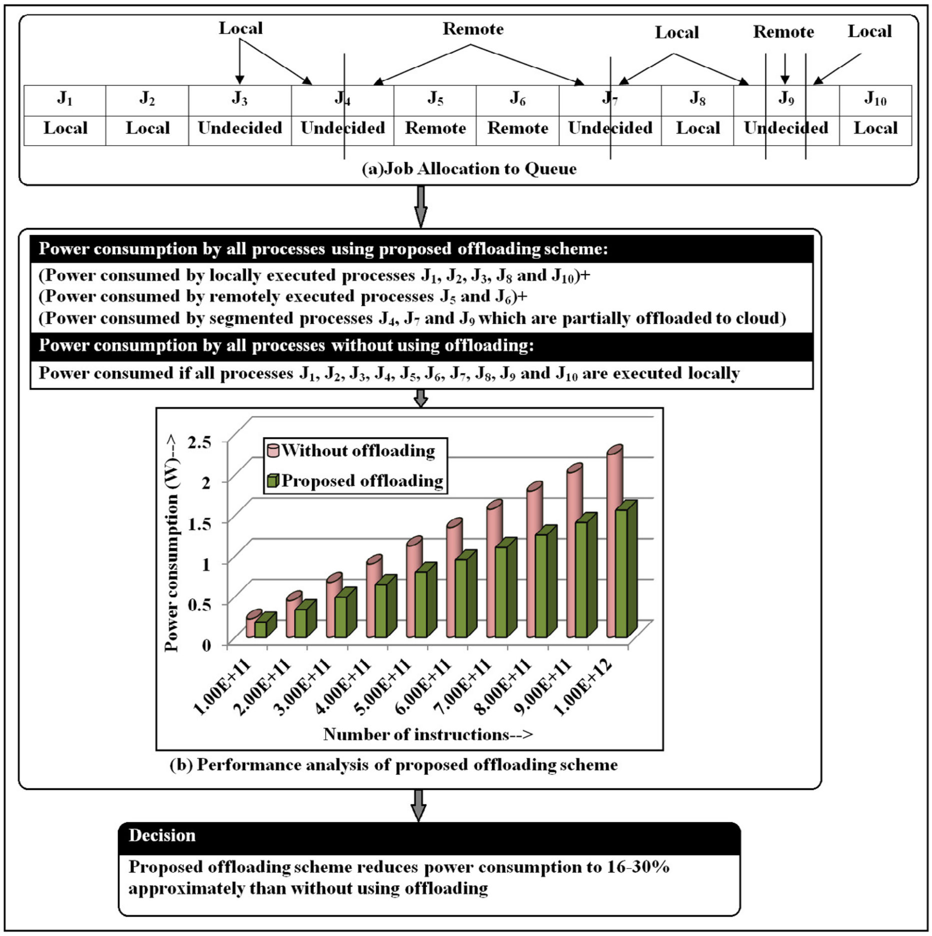

Fig. 4. (a) Job allocation in Local and Remote queue; (b) performance analysis of proposed offloading scheme.

In Fig. 4 (b) the power consumed in job execution using the proposed offloading strategy is presented with respect to the number of job instructions. Fig. 4 (b) also presents the power consumption if all these jobs are executed within the mobile device itself. From Fig. 4 (b) it is observed that using proposed offloading strategy power consumption in job execution can be reduced to approximately 16–30% than executing them locally within the mobile device.

3.5.3. Inference from case study

From the case study, it is observed that by offloading jobs using the proposed scheme, proper allocation of jobs can be done and power consumption can also be reduced than their local execution. Thus we can refer to our strategy as a green offloading scheme.

3.6. Security in proposed scheme

3.6.1. Security in femto-cloud architecture

To provide secure data transmission, offloading is performed through the femtocell in the proposed scheme. The following precautions are taken in femtocell for security purpose [37]:

- Femtocell is connected with the network through the Se-GW as shown in Fig. 1.
- A security tunnel is created between the Se-GW and femto access point in order to protect the transmitted information through the backhaul link to achieve data integrity and confidentiality.
- It is verified by the HNB-GW or femto access point management system whether a femtocell is operating in a licensed spectrum.
- Adequate cryptographic algorithm is used to ensure confidentiality, data integrity and for authentication purpose.
- Identities of the subscribers registered under a particular femtocell are not exposed.
- Any information from an unauthenticated source is discarded by the femto access point.
- To protect from unauthorized access, authentication details and user information cannot be accessed in plain text at the femto access point.
- Whenever offloading takes place through the femtocell, if any vulnerability is detected, the data are discarded.
- The Se-GW prevents any unauthorized user to access the data during transmission.

Hence due to the presence of Se-GW, authentication details are securely transferred in the femto-cloud network. Now there still remains a question relating to the security in data access inside the cloud. For the purpose of protecting personal data or computation offloading inside the cloud, retinal image recognition based biometric authentication is used in the present work.

3.6.2. Retinal image recognition based secure cloud access

Biometric authentication refers to the identification of human beings by their unique physiological or behavioral characteristics e.g. finger prints, retina and face recognitions, keystroke analysis etc. [38]. This physiological or behavioral characteristic does not always match perfectly. Hence a threshold is maintained. In the proposed scheme, we have used retinal image recognition for authentication. This is used

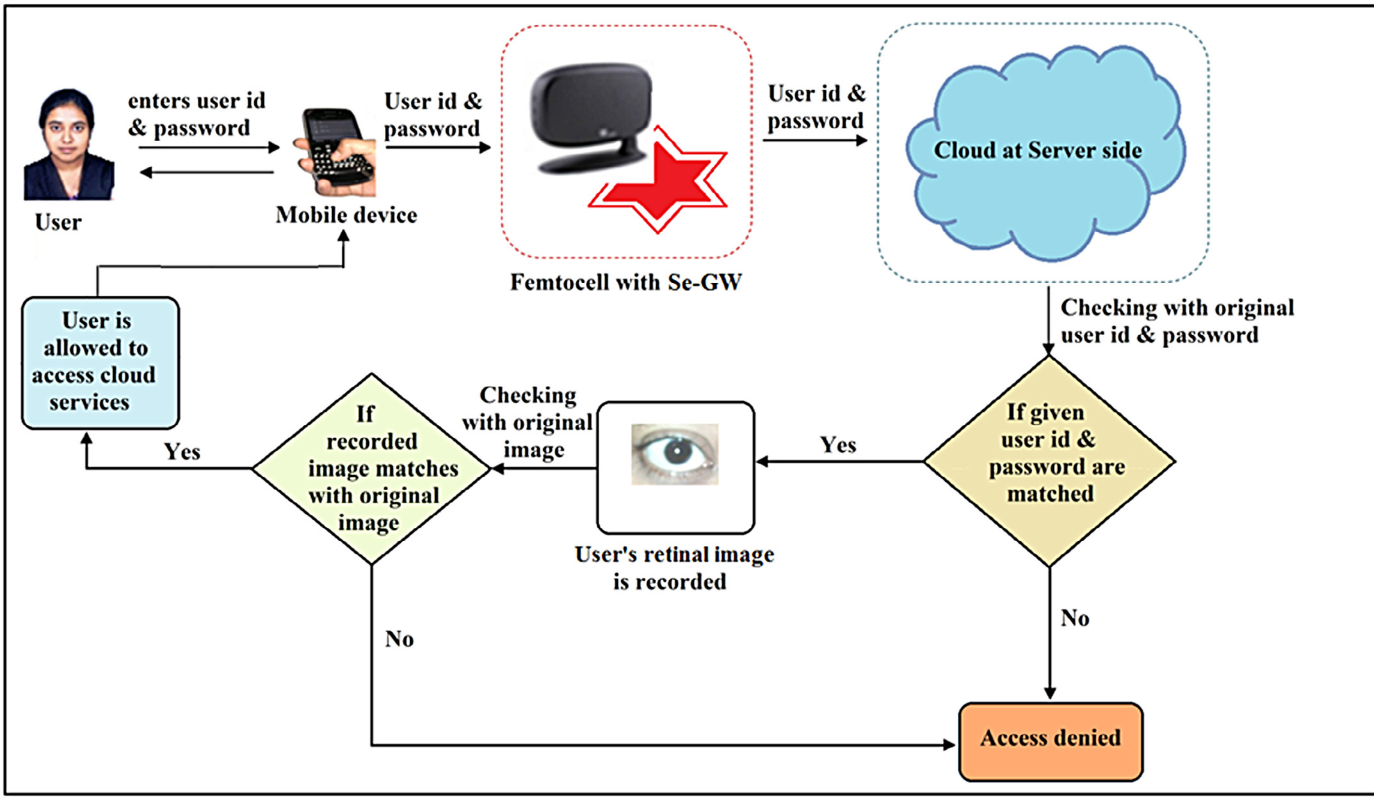

Fig. 5. User authentication through retinal image recognition.

in our approach as nowadays most mobile devices support webcam, and retinal image of each human being is unique.

The steps to authenticate a user through his or her retinal image are presented in Fig. 5 and described as follows:

1. When a mobile user registered under a femtocell tries to access the cloud services for the first time, a user ID and password is generated by the server and sent back to the mobile device of that user. The user ID and password are received securely due to the Se-GW of the femto-cloud network.
2. Then the user has to enter the received user ID and password.
3. After correct user ID and password are provided, the retinal image of the user is recorded through the webcam and stored at the server side.
4. Whenever the user wishes to access the cloud services, the user has to enter the correct user ID and password.
5. If the provided information is correct, the user's retinal image is captured using webcam and checked against the recorded image.
6. If the retinal image is matched above the threshold with the original image stored at server side, the user is permitted to access the cloud services.

Therefore in the proposed scheme security is achieved through Se-GW during data transmission and through retinal image recognition during cloud access.

4. Results and discussions

4.1. Energy consumption of femto-cloud network

The femto-cloud scenario is implemented using Qualnet Simulator version 7. The simulation parameters are presented in Table 4.

In the simulation model, the mobile user sends request for offloading an application to the cloud through the femtocell. The femto-cloud scenario is shown in Fig. 6.

In the scenario, nodes 1, 2, 3, 4 and 5 present the mobile device, femtocell, Se-GW, HNB-GW and cloud respectively. It is assumed that the femtocell consumes power of 10 mW–100 mW. The energy consumption by all the nodes of the system is compared against the macrocell and microcell based mobile cloud networks. In this simulation, total energy consumption is calculated as the energy

Table 4
Parameters used in simulation.

Layer	Parameter	Value
Physical layer	Radio type	802.11b radio
	Packet reception model	PHY 802.11b reception model
	Antenna model	Omni directional
	Energy model	User specified
	Temperature	290.0 K
	Noise Factor	10.0
MAC layer	MAC protocol	802.11
Network layer	Network protocol	IPV4
	Type of router	Predefined
	Routing protocol	Bellman Ford
	Router type	Generic
Transport layer	Sending and receiving buffer size (bytes)	512, 8162, 16,384 (for mobile device, femtocell and cloud respectively)
Battery model	Battery model	Linear model
	Battery change monitoring interval	60 s
	Full battery capacity (mAh)	1200
Scenario properties	Simulation time (s)	300
CBR properties	Item size (bytes)	512

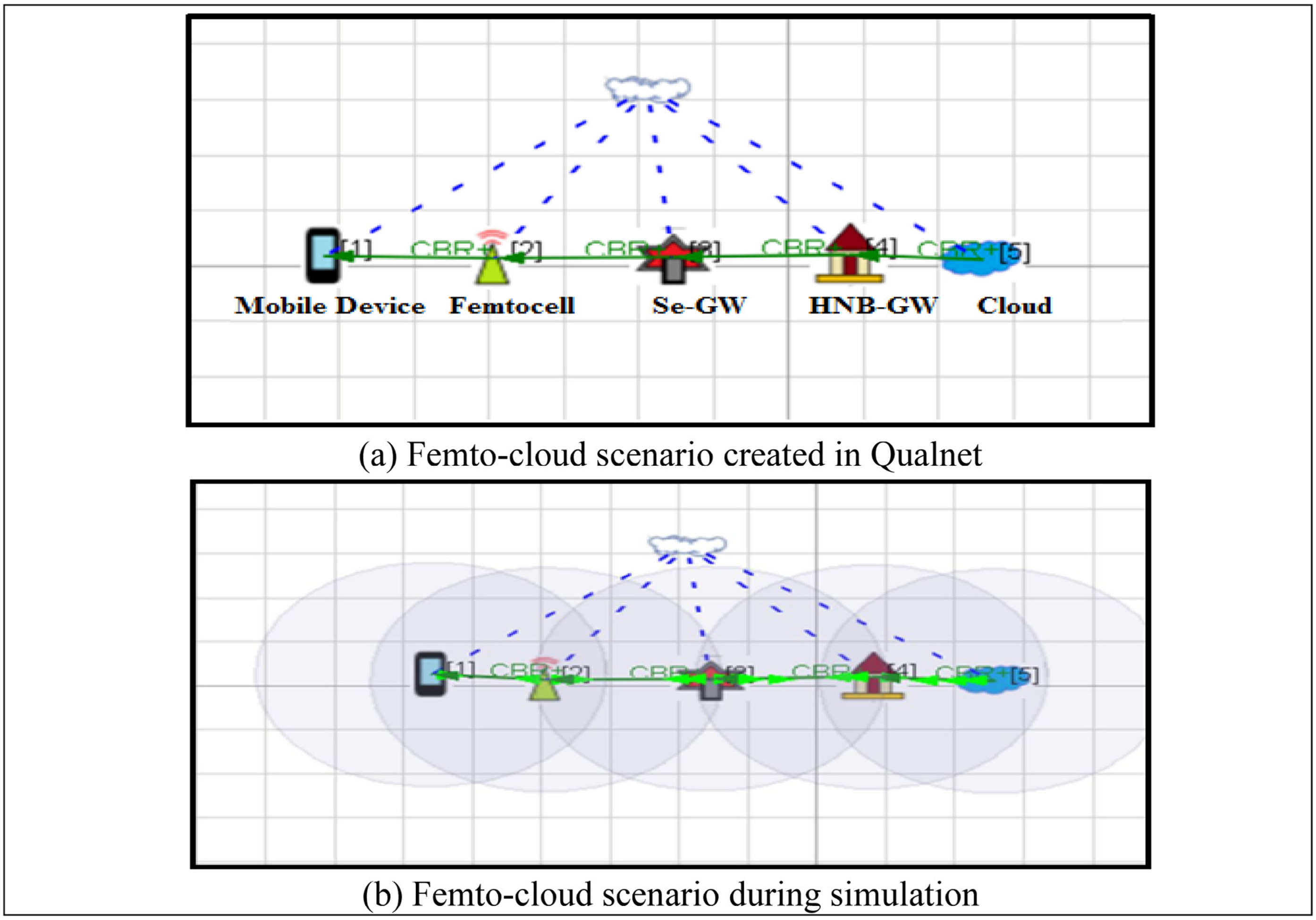

Fig. 6. Femto-cloud scenario simulated in Qualnet.

consumption in transmit and receive mode by all the nodes. Fig. 7 shows that the energy consumed by the femto-cloud system ranges from 10 to 30 mWh approximately for 200–500 kB data transmission.

Fig. 7 shows that using femto-cloud architecture, the energy consumption can be reduced to 25–29% and 19–23% approximately than accessing cloud using macrocell and microcell base stations respectively.

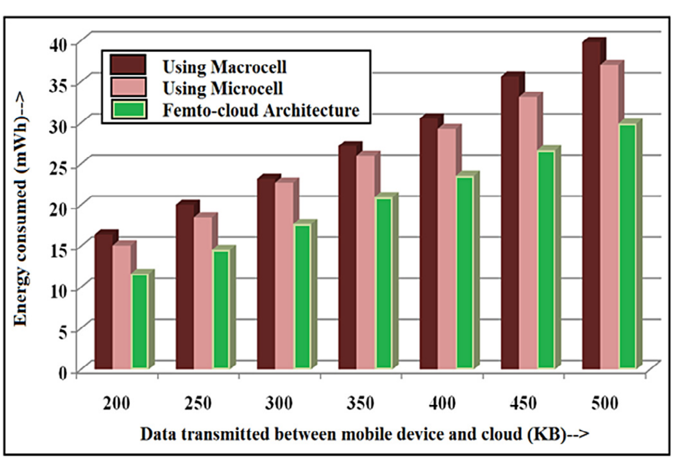

Fig. 7. Energy consumption of the system considering mobile device, femtocell, Se-GW, HNB-GW and cloud.

4.2. Experimental analysis of proposed offloading algorithm using cloud servers

An experimental analysis of the proposed offloading strategy is performed using the WBUT cloud servers located at the West Bengal University of Technology (WBUT) campus. File creation and writing into the file is considered as job in our work. The process of creating files of different sizes (10–40 kB) has been executed locally in mobile devices. Four Lenevo laptops, three Dell tablets and three Samsung smart phones have been used as the mobile devices for local execution. The configurations of these devices are given in Table 5.

The same process is offloaded partially or fully inside the cloud using our proposed offloading scheme described in Table 2. Four

Table 5
Configurations of mobile devices used for local execution of ten jobs.

Device name	RAM	HDD/Storage	Processor
Lenevo laptop	2 GB	320 GB	Intel(R) Pentium(R) CPU B940 @ 2.00 GHz
Dell Latitude 10 tablet	2 GB	64 GB	Intel Z2760 1.8 GHz
Samsung smart phone Asus ZenFone 5	2 GB	16 GB	Intel Atom Z2560 1.6 GHz

Table 6
Configurations of cloud servers located at WBUT campus.

Sl. no.	RAM	HDD/Storage	Processor
Server 1	16 GB	900 GB	Intel(R) Xeon(R) CPU ES-2667 0 @ 2.90 GHz
Server 2	16 GB	650 GB	Intel(R) Xeon(R) CPU ES-2667 0 @ 2.90 GHz
Server 3	16 GB	450 GB	Intel(R) Xeon(R) CPU ES-2667 0 @ 2.90 GHz
Server 4	16 GB	320 GB	Intel(R) Xeon(R) CPU ES-2667 0 @ 2.90 GHz

cloud servers of WBUT have been used for offloading. These servers have the following configurations presented in Table 6.

The experimental results obtained by executing these jobs using proposed offloading algorithm are presented in Table 7 and compared with their local execution results from the perspective of latency and power consumption.

Based on the data of Table 7, the proposed offloading algorithm is pictorially compared with local execution in Figs. 8 and 9 with respect to latency and power consumption respectively.

From Table 7, Figs. 8 and 9 it is observed that executing these jobs using proposed offloading algorithm can reduce power consumption and latency approximately 3–32% and 4–31% respectively. Hence the proposed offloading algorithm is referred as a green strategy.

Although we have proposed a low power and fast offloading scheme, still energy-optimized job scheduling and resource management inside the cloud are critical matters. The cloud contains a large number of resources. If processor allocation for job execution takes place from the perspective of job size, deadline and utility, power-efficient allocation can be done [39–44]. For a job with medium number of instructions, small utility and soft deadline with longer lifetime, a comparatively low speed processor can be allocated to reduce power consumption. On the other hand if the job contains large number of instructions with small lifetime and high utility, high speed processor has to be allocated to it. For latency minimization, cloudlet [45,46] can be used.

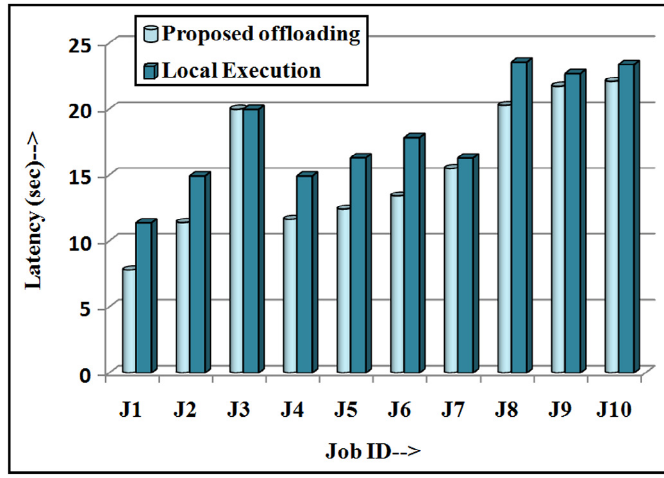

Fig. 8. Latency for job execution using proposed offloading algorithm.

5. Conclusion

In this paper we have proposed a low power method regarding whether to offload or not-to-offload. In the proposed scheme power consumption, computation time and deadline of the jobs are considered along with the preceding and succeeding job allocation to decide whether to offload a job to the cloud or not. Simulation results present that by offloading large size applications to the cloud, approximately 7–34% power saving is achieved. Offloading to the cloud raises another critical issue security. To achieve secure data transmission femto-cloud architecture is used where the application is offloaded from the mobile device to the cloud through the secure and low power base station femtocell. To achieve security during data access inside the cloud, retinal image recognition based bio-metric authentication is used. Theoretical and simulation results present that using femto-cloud architecture approximately 70–83% and 52–66% power savings are achieved than offloading through macrocell and microcell base stations respectively. From the experimental results it is observed that the proposed offloading algorithm reduces the power consumption and latency to approximately 3–32% and 4–31% respectively. There are still various areas in mobile cloud computing where future researches are required such as energy efficient job scheduling and resource management inside the cloud. Although cloudlet reduces latency, still selection of appropriate cloudlet to offload an application is an important issue. The selection of application specific cloudlet with respect to low power and low latency is an open research area.

Acknowledgement

Authors are grateful to the Department of Science and Technology (DST) for sanctioning a research Project entitled “Dynamic Optimization of Green Mobile Networks: Algorithm, Architecture and Applications” under Fast Track Young Scientist scheme reference no.: SERB/F/5044/2012-2013 and No.DST/INSPIRE Fellowship/2013/327 under which this paper has been completed.

Table 7
Comparison of proposed offloading scheme with local execution based on experimental results.

Job ID	Using proposed offloading algorithm					Local execution		Remarks
	Execution			Latency (s)	Power (W)	Latency (s)	Power (W)	
	Local	Partial	Remote					
J1			✓	7.784	0.39	11.357	0.57	Full offloading to cloud saves 31% latency and 32% power
J2		✓		11.374	0.57	14.919	0.75	Partial offloading to cloud saves 24% latency and 24% power
J3	✓			19.96	1	19.96	1	Local execution occurs
J4		✓		11.624	0.58	14.92	0.75	Partial offloading to cloud saves 22% latency and 23% power
J5		✓		12.392	0.62	16.292	0.81	Partial offloading to cloud saves 24% latency and 23% power
J6		✓		13.392	0.67	17.822	0.89	Partial offloading to cloud saves 25% latency and 25% power
J7			✓	15.48	0.77	16.28	0.81	Full offloading to cloud saves 5% latency and 5% power
J8		✓		20.252	1.01	23.552	1.18	Partial offloading to cloud saves 14% latency and 13% power
J9			✓	21.712	1.09	22.69	1.13	Full offloading to cloud saves 4% latency and 3% power
J10			✓	22.080	1.10	23.37	1.17	Full offloading to cloud saves 5% latency and 6% power

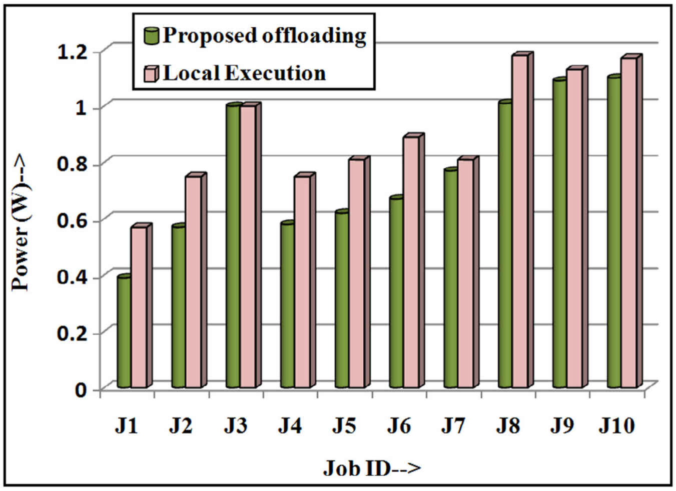

Fig. 9. Power consumption for job execution using proposed offloading algorithm.

References

[1] Z. Sanaei, S. Abolfazli, A. Gani, R. Buyya, Heterogeneity in mobile cloud computing: taxonomy and open challenges, IEEE Commun. Surv. Tutor. 16 (2014) 369–392.

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

[3] N. Fernando, S.W. Loke, W. Rahayu, Mobile cloud computing: a survey, Future Gener. Comput. Syst. 29 (2013) 84–106.

[4] A. Mukherjee, D. De, A novel cost-effective and high-speed location tracking scheme for overlay macrocell-femtocell network, in: URSIGA, pp. 1–4, 2014.

[5] A. Mukherjee, D. De, A cost-effective location tracking strategy for femtocell based mobile network, in: International Conference on Control, Instrumentation, Energy and Communication, pp. 533–537, 2014.

[6] A. Mukherjee, S. Bhattacherjee, S. Pal, D. De, Femtocell based green power consumption methods for mobile network, Comput. Netw. 57 (2013) 162–178.

[7] A. Mukherjee, D. De, Congestion detection, prevention and avoidance strategies for an intelligent, energy and spectrum efficient green mobile network, J. Comput. Intell. Elec. Syst. 2 (2013) 1–19.

[8] Z. Hasan, H. Boostanimehr, V.K. Bhargava, Green cellular networks: a survey, some research issues and challenges, IEEE Commun. Surv. Tutor. 13 (2011) 524–540.

[9] F. Mhiri, K.S.B. Reguiga, R. Bouallegue, G. Pujolle, A power management algorithm for green femtocell networks, in: Tenth IFIP Annual Mediterranean Ad hoc Networking Workshop, pp. 45–49, 2011.

[10] V. Chandrasekhar, T. Muharemovic, Z. Shen, J.G. Andrews, A. Gatherer, Power control in two-tier femtocell networks, Wireless Communications, IEEE Transactions 8 (2009) 4316–4328.

[11] D. Chee, M.S. Kang, H. Lee, B.C. Jung, A study on the green cellular network with femtocells, in: Third International Conference on Ubiquitous and Future Networks, pp. 235–240, 2011.

[12] Y. Haddad, Y. Mirsky, Power efficient femtocell distribution strategies, in: Nineteenth International Conference on Software, Telecommunications and Computer Networks, pp. 1–5, 2011.

[13] X. Chen, H. Zhang, T. Chen, M. Lasanen, J. Palicot, On improving energy efficiency within green femtocell networks: a hierarchical reinforcement learning approach, 2013.

[14] J. Zhang, G. de la Roche, Femtocells: Technologies and Deployment, John Wiley and Sons, Ltd., 2010.

[15] T. Nihtila, Capacity improvement by employing femto cells in a macro cell HSDPA network, in: IEEE Symposium on Computers and Communications, pp. 838–843, 2008.

[16] D. De, A. Mukherjee, Femtocell based economic health monitoring scheme using mobile cloud computing, in: International Advance Computing Conference, pp. 385–390, 2014.

[17] D. De, A. Mukherjee, Femto-cloud based secure and economic distributed diagnosis and home health care system, J. Med. Imag. Health Informat. 5 (2014) 435–447.

[18] A. Mukherjee, D. De, Femtocell based green health monitoring strategy, in: URSIGA, pp. 1–4, 2014.

[19] K. Yang, S. Ou, H.H. Chen, On effective offloading services for resource-constrained mobile devices running heavier mobile Internet applications, IEEE Commun. Mag. 46 (2008) 56–63.

[20] A. Mukherjee, P. Gupta, D. De, Mobile cloud computing based energy efficient offloading strategies for femtocell network, in: Applications and Innovations in Mobile Computing, pp. 28–35, 2014.

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

[22] S. Singh, H.S. Dhillon, J.G. Andrews, Offloading in heterogeneous networks: modeling, analysis, and design insights, IEEE T. Wirel. Commun. 12 (2013) 2484–2497.

[23] V. March, L.B. Sung, GMoCA: green mobile cloud applications, in: First International Workshop on Green and Sustainable Softwares, pp. 15–20, 2012.

[24] C. Shi, K. Habak, P. Pandurangan, M. Ammar, M. Naik, E. Zegura, COSMOS: computation offloading as a service for mobile devices, in: Proc. of the fifteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 287–296, 2014.

[25] K.I. Koshy, A.M. Juby, V. Namboodiri, M. Overcash, Can cloud computing lead to increased sustainability of mobile devices? in: International Symposium on Sustainable Systems and Technology, pp. 1–4, 2012.

[26] M. Altamimi, R. Palit, K. Naik, A. Nayak, Energy-as-a-service (EaaS): on the efficacy of multimedia cloud computing to save smartphone energy, in: Fifth International Conference on Cloud Computing, pp. 764–771, 2012.

[27] K. Kumar, Y.H. Lu, Cloud computing for mobile users: can offloading computation save energy? Computer 43 (2010) 51–56.

[28] H. Wu, Q. Wang, K. Wolter, Methods of cloud-path selection for offloading in mobile cloud computing systems, in: IEEE CloudCom, pp. 443–448, 2012.

[29] G. Chen, B.T. Kang, M. Kandemir, N. Vijaykrishnan, M.J. Irwin, R. Chandramouli, Studying energy trade offs in offloading computation/compilation in Java-enabled mobile devices, IEEE Trans. Paral. Dist. Syst. 15 (2004) 795–809.

[30] M. Altamimi, A. Abdrabou, S. Naik, A. Nayak, Energy cost models of smartphones for task offloading to the cloud. IEEE Transactions on Emerging Topics in Computing, 2015.

[31] C. Xian, Y.H. Lu, Z. Li, Adaptive computation offloading for energy conservation on battery-powered systems, in: International Conference on Parallel and Distributed Systems, IEEE, pp. 1–8, 2007.

[32] C.X. Mavromoustakis, A. Andreou, G. Mastorakis, A. Bourdena, J.M. Batalla, C. Dobre, On the performance evaluation of a novel offloading-based energy conservation mechanism for wireless devices, in: Mobile Networks and Management, Springer International Publishing, pp. 179–191, 2015.

[33] Z. Li, C. Wang, R. Xu, Computation offloading to save energy on handheld devices: a partition scheme, in: Proc. of the 2001 International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, ACM, pp. 238–246, 2001.

[34] M. Shiraz, A. Gani, A. Shamim, S. Khan, R.W. Ahmad, Energy efficient computational offloading framework for mobile cloud computing, J. Grid Comput. 13 (2015) 1–18.

[35] I. Ashraf, T.W. Ho, H. Claussen, Improving energy efficiency of femtocell base stations via user activity detection, in: WCNC Proceedings on IEEE Communications Society publication, pp. 1–5, 2010.

[36] Y. Ye, N. Jain, L. Xia, S. Joshi, I.L. Yen, F. Bastani, et al., A framework for QoS and power management in a service cloud environment with mobile devices, in: Fifth IEEE International Symposium on Service Oriented System Engineering, pp. 236–243, 2010.

[37] J. Chen, M. Wong, Security implications and considerations for femtocells, J. Cyber Secur. Mobil. 21 (2012) 21–35 River Publishers.

[38] D. De, A. Mukherjee, S. Bhattacherjee, P. Gupta, Trusted Cloud and Femtocell Based Biometric Authentication for Mobile Network. Handbook of Research on Securing Cloud-Based Databases with Biometric Applications, IGI Publications, 2014.

[39] K.H. Kim, R. Buyya, J. Kim, Power aware scheduling of bag-of-tasks applications with deadline constraints on DVS-enabled clusters, in: CCGRID, pp. 541–548, 2007.

[40] K.H. Kim, A. Beloglazov, R. Buyya, Power-aware provisioning of cloud resources for real-time services, in: Proc. of the seventh International Workshop on Middleware for Grids, Clouds and e-Science, pp. 1, 2009.

[41] A. Beloglazov, J. Abawajy, R. Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing, Future Gener. Comput. Syst. 28 (2012) 755–768.

[42] A. Beloglazov, R. Buyya, Y.C. Lee, A. Zomaya, A taxonomy and survey of energy-efficient data centers and cloud computing systems, Adv. Comput. 82 (2011) 47–111.

[43] K.H. Kim, W.Y. Lee, K.I.M. Jong, R. Buyya, SLA-based scheduling of bag-of-tasks applications on power-aware cluster systems, IEICE Trans. Inf. & Syst. 93 (2010) 3194–3201.

[44] S.K. Garg, C.S. Yeo, A. Anandasivam, R. Buyya, Environment-conscious scheduling of HPC applications on distributed cloud-oriented data centers, J. Par. Dist. Comput. 71 (2011) 732–749.

[45] T. Verbelen, S. Pieter, D.T. Filip, D. Bart, Cloudlets: bringing the cloud to the mobile user, in: Proc. of the third ACM Workshop on Mobile Cloud Computing and Services, pp. 29–36, 2012.

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