Delay-Optimized Offloading for Mobile Cloud Computing Services in Heterogenous Networks

Kan Zheng*, Hanlin Meng*, Hua Zhu*, Kai Lin†, Wei Cai°, and Hang Long*

*Wireless Signal Processing & Network Lab
Key laboratory of Universal Wireless Communication, Ministry of Education,
Beijing University of Posts&Telecommunications, Beijing 100088, China
†School of Computer Science and Engineering, Dalian University of Technology,
Dalian, Liaoning, China
°Electrical and Computer University of British Columbia,
Vancouver Canada, V6T 1Z4

Abstract. Offloading is an efficient method for extending the lifetime and speeding up the execution rate of mobile devices by executing remotely in mobile cloud computing (MCC) networks. Meanwhile, heterogeneous network (HetNet), which has multiple types of radio access nodes in the network, is widely accepted as a promising way to satisfy the increased traffic demand. In this paper, we first propose two delay-optimized offloading control schemes in LTE-Advanced heterogeneous networks. One of them is to control the number of offloading users by setting a threshold. Another is to determine whether the users are suited to offload by estimating the execution delay. Both of them take the traffic load of serving cell and neighbouring cells into account. For comparison purpose, the delay performance of different schemes are evaluated by simulations in not only heterogeneous network but also macro-only network.

Key words: Mobile cloud computing (MCC), Heterogeneous Network (HetNet), Offloading

1 Introduction

In recent years, mobile cloud computing (MCC) is introduced as an integration of cloud computing into the mobile environment to overcome the limitations of mobile devices (MDs). However, it has to face many technical challenges because of the integration of two different fields [1]. One of the challenges is computing offloading technique. It is proposed to migrate the large computations and complex processing from resource-limited MDs to remote servers with the objective of extending the battery life and speeding up the operating rate, which avoids requiring changes in the structure or hardware equipment [2]. Experiments in [3] show that offloading is not always the effective way to improve performance. So how to offload the application to the remote servers becomes very important at present.

There are generally two kinds of offloading techniques, called full offloading and partial offloading. Full offloading is to move all computation tasks of mobile applications from the MD to the remote cloud. However, different computation tasks of an application may have different characteristics that make them more or less suitable for offloading. Application can offload only the sub-parts that benefit from the remote execution for more energy saving. So partial offloading has gained more attention, where the application is divides into unoffloadble tasks and offloadble parts. In [4], a simple but basis model is proposed to analyze whether the energy can be saved by computation offloading. It is shown that moving image processing to the remote server reduces 41% for energy consumption [5]. System is modeled within a Markovian dynamic control framework [6], in which the associated energy versus delay trade-off is studied. The power consumption is minimized in [7] by solving two constrained optimization problems, which are setting the optimal clock frequency of local processor and data transmission rate of each time slot for cloud execution within time delay. Then the offloading decision is made with smaller energy consumption. A dynamic offloading algorithm based on Lyapunov optimization is proposed in [8] to make the offloading decision for all the computation tasks while satisfying given application execution time requirement.

Although existing works have improved the basic model from various aspects, there are still some open problems, especially when the offloading data is transmitted via heterogeneous network (Hetnet), in which the base stations with lower transmit power (e.g., pico, femto, realy eNodeB) are deployed besides the macro eNodeB with high transmit power [9]. The offloading decision of an application is made based on the estimation of execution time or energy consumption. However, almost no literature considers traffic load of MD's serving cell and neighboring cells which may affects the offloading decision so far. Traffic load at MD's serving cell impacts on wireless channel gains at the scheduling instants of MDs. Meanwhile, the increased traffic load in the neighboring cells may results in heavy interference situation. Both of them may affect power consumption and execution delay. Therefore, we propose the delay-optimized offloading control schemes in Hetnet environment with the consideration of traffic load, which reduces the average delay of all the mobile users. Simulation results demonstrate the effectiveness of the proposed scheme in terms of average delay.

This paper is organized as follows. Section 2 gives a brief overview of the system and problem formulation. The delay-optimized offloading control schemes are proposed in Section 3. Then, Section 4 gives the simulation results. Finally, Section 5 concludes this paper.

2 System Model

The LTE-Advanced heterogenous network with macrocell and picocell is considered in this paper. An picocell eNodeB is deployed in each sector of macrocell [10]. There are M sectors in a single macrocell.

2.1 Flow Model

A dynamic flow model with elastic traffic for mobile applications is assumed, where a new flow arrives at the network with a finite-length file request, and leaves the system after the file is transmitted. The flow of user n arriving at cell m in the network follows a Poisson distribution with an average arrival rate of $\lambda_n^{(m)}$. There are N_m active users served by the eNodeB in cell $m, 0 \leq m \leq M$. Note that cell 0 represents the macrocell while others indicate picocells.

The main purposes of computation offloading are to save the power consumption of MDs and speed up the application processing. However, the above two goals may not always be achieved depending on the particulars of the computation task, the server load, and the connectivity to the network. For example, a task with high computation and low communication requirements is more likely to benefit from offloading than a task with low computation and high communication requirements. Therefore, a judicious decision has to be made on whether to offload a computation task or not. For simplicity, a mobile application is assumed to be executed either locally in the mobile device or remotely in the remote cloud servers by full offloading. Let the binary vector $\mathbf{B}^{(m)} = \{b_n^{(m)} | b_n^{(m)} \in \{0,1\}\}_{1\times N_m}$ describe the offloading decision, where $b_n^{(m)} = 1$ denotes that the application of user n in cell m is executed locally, otherwise $b_n^{(m)} = 0$.

2.2 Network Model

For offloading cases, the total execution time for an application consists of the time spent sending the task and data to the cloud servers, idly waiting for the cloud to complete the task, and receiving the result of the task. However, due to the large computation ability of cloud servers, the delay caused by the wireless transmission between the MD and cloud servers especially in the uplink may dominate the total execution time. In this case, the data transmission rate between the mobile device and cloud servers has significant impact on offloading decisions.

The basic radio resource unit for OFDM transmission can be described as a two-dimensional (2-D) time-frequency grid that corresponds to a set of OFDM symbols and subcarriers in the time and frequency domains. In LTE-Advanced networks, the basic unit for data transmission is a pair of resource blocks (RBs) that correspond to a 180-kHz bandwidth during a 1 ms subframe. In this paper, all the radio resources, i.e., K RBs, are assumed to be fully reused between picocells and macrocell. Let the binary matrix $\mathbf{A}^{(m)} = \{a_{k,n}^{(m)} | a_{k,n}^{(m)} \in \{0,1\}\}_{K\times N_m}$ describe the resource allocation among the users, where $a_{k,n}^{(m)} = 1$ denotes that RB k in cell m is assigned to user n, otherwise $a_{k,n}^{(m)} = 0$. Then, the achievable rate of user n on RB k in cell m is given by

$$\eta_{k,n}^{(m)} = W \log_2 \left(1 + \frac{\beta_n^{(m)} h_{k,n}^{(m)} P_{k,n}^{(m)}}{\zeta_{k,n}^{(m)} + \sigma_N^2}\right) , \quad (1)$$
$$1 \leq k \leq K \, , 1 \leq n \leq N_m \, , 0 \leq m \leq M \, ,$$

where W is the bandwidth per RB, $\beta_n^{(m)}$ is the pathloss (PL) from the eNodeB in cell m to user n, $P_{k,n}^{(m)}$ is the transmit power of user n at RB k in cell m, $h_{k,n}^{(m)}$ is the independent, identically distributed (i.i.d.) Rayleigh fading channel gain between the eNodeB in cell m to user n at RB k, i.e., $h_{k,n}^{(m)} \sim \mathcal{CN}(0, \sigma^2)$[1], with $\sigma^2 = \mathbb{E}[|h_{k,n}^{(m)}|^2]$, σ_N^2 is the noise power of the additive white Gaussian noise (AWGN), and $\zeta_{k,n}^{(m)}$ is the interference between cells, i.e.

$$\zeta_{k,n}^{(m)} = \sum_{j=0, j\neq m}^{M} \sum_{i=1}^{N_j} a_{k,i}^{(j)} b_i^{(j)} \beta_i^{(j)} h_{k,i}^{(j)} P_{k,i}^{(j)} \ . \tag{2}$$

3 Delay-optimized Offloading Control Schemes

3.1 Delay Analysis

If an application of user n in cell m is decided to be offloading to cloud servers for processing, the total execution time can be calculated by

$$\tau_n^{(m)} = \tau_{U,n}^{(m)} + \tau_{C,n}^{(m)} + \tau_{D,n}^{(m)} \ , \tag{3}$$

where $\tau_{U,n}^{(m)}$ and $\tau_{D,n}^{(m)}$ denotes the transmission delay in the uplink and downlink, respectively, and $\tau_{C,n}^{(m)}$ is the idle time waiting for cloud to complete the task. Compared to the transmission delay, the processing time in cloud servers is quite smaller and can be omitted. On the other hand, some kind of typical cloud computing services has much more amount of task data in the uplink than than of the completed result in the downlink. Moreover, the uplink transmission ability is usually the bottleneck of the wireless networks instead of the downlink. Therefore, the total execution time may approximately equal to the delay due to uplink transmission, i.e.

$$\tau_n^{(m)} \approx \tau_{U,n}^{(m)} \ . \tag{4}$$

Otherwise, this application is executed in the mobile device with the local execution time $\tau_{L,n}^{(m)}$. Then, the average delay of all the users in the heterogenous network can be expressed by

$$\overline{\tau} = \frac{1}{\sum_{m=0}^{M} N_m} \sum_{m=0}^{M} \sum_{n=1}^{N_m} \{ b_n^{(m)} \tau_{U,n}^{(m)} + (1 - b_n^{(m)}) \tau_{L,n}^{(m)} \} \ . \tag{5}$$

As we know, the delay due to uplink transmission is highly determined by the achievable data rate $\mu_n^{(m)}$ and the byte number of the transmitted data $D_n^{(m)}$, i.e.

[1] A circularly symmetric complex Gaussian RV x with mean m and covariance R is denoted by $x \sim \mathcal{CN}(m, R)$.

$$\tau_{U,n}^{(m)} = \frac{D_n^{(m)}}{\mu_n^{(m)}} \ . \tag{6}$$

It is noted that the different offloading control scheme and scheduling algorithm may result in the different achievable data rate, i.e.

$$\mu_n^{(m)} = \sum_{k=1}^{K} a_{k,n}^{(m)} b_n^{(m)} \eta_{k,n}^{(m)} \ . \tag{7}$$

Meanwhile, the local execution time $\tau_{L,n}^{(m)}$ can be given by

$$\tau_{L,n}^{(m)} = \frac{C_n^{(m)}}{X_n^{(m)}} \ , \tag{8}$$

where $C_n^{(m)}$ is the computation complexity for application in term of instruction number, and $X_n^{(m)}$ is the speed of the local execution at user n.

3.2 Delay-optimal Offloading Control Scheme

The offloading decision problem is to find $\mathbf{A}$ and $\mathbf{B}$ such that an objective function is optimized. In this paper, the optimization problem with the objective of minimizing the average system delay is needed to be solved for offloading decision, i.e.

$$\begin{aligned} & \arg\min_{\mathbf{A},\mathbf{B}} \overline{\tau} \ , \\ s.t. \quad & a_{k,n}^{(m)} \in \{0,1\} \ , \\ & b_n^{(m)} \in \{0,1\} \ , \\ & 1 \le k \le K \ , 1 \le n \le N_m \ , 0 \le m \le M \ . \end{aligned} \tag{9}$$

By integrating (6), (7) and (8) into (9), we can easily find that the optimization problem is non-concave. To find its optimal solution, exhaustive search over all the possible solution set is needed, which has prohibitively high computational complexity. Therefore, considering the implementation feasibility, it is necessary to deal with the offloading decision problem in the heterogenous network by other methods.

For simplicity, not only the computation complexity of application but also the speed of the local execution for all users are assumed to be same, i.e., $C_n^{(m)} = C$ and $X_n^{(m)} = X$. Then, the local execution time $\tau_{L,n}^{(m)}$ becomes the constant, i.e., $\tau_{L,n}^{(m)} = \tau_L = C/X$. Moreover, the amount of data bytes that needed to be transmitted on the link for all users is assumed to be fixed, i.e., $D_n^{(m)} = D$. Now, the average delay of all the users in the heterogenous network can be simplified as

$$\overline{\tau} = \frac{1}{\sum_{m=0}^{M} N_m} \sum_{m=0}^{M} \sum_{n=1}^{N_m} \{ b_n^{(m)} \tau_{U,n}^{(m)} + (1 - b_n^{(m)}) \tau_L \} \ . \tag{10}$$

It can be further rewritten as

$$\overline{\tau} = \rho \overline{\tau}_O + (1 - \rho) \tau_L \ , \tag{11}$$

where ρ is the ratio of the applications in the network that needs to be offloading, i.e.

$$\rho = \frac{\sum_{m=0}^{M} \sum_{n=1}^{N_m} b_n^{(m)}}{\sum_{m=0}^{M} N_m} \ . \tag{12}$$

The average execution time for all the applications to be offloading is expressed by

$$\overline{\tau}_O = \frac{\sum_{m=0}^{M} \sum_{n=1}^{N_m} b_n^{(m)} \tau_{U,n}^{(m)}}{\sum_{m=0}^{M} \sum_{n=1}^{N_m} b_n^{(m)}} \ . \tag{13}$$

It can be seen that $\overline{\tau}_O$ is not only dependent on the ratio of offloading applications but also on the scheduling algorithm and wireless channel characters of offloading users, i.e.

$$\overline{\tau}_O = f(\rho, \mathcal{S}, \boldsymbol{\gamma}) \ , \tag{14}$$

where $\mathcal{S}$ represents the scheduling algorithm such Round Rubin, max C/I and so on, $\boldsymbol{\gamma}$ represents the wireless channel characters of offloading users. We propose two methods to perform the offloading decision in order to decrease the average delay as much as possible.

Threshold-based Method Given that $\mathcal{S}$ in (14) is fixed, then the τ_O only depends on the the ratio ρ and the channel condition $\boldsymbol{\gamma}$. Furthermore, in order to find the solution with low complexity, $\boldsymbol{\gamma}$ is ignored. So, the τ_O can be expressed by $f(\rho)$. When the following conditions can be satisfied for $f(\rho)$, the optimal solution of $f(\rho)$ exists in the network, i.e., i) $f(\rho)$ increases monotonically in ρ, ii) $f(\rho)$ is strictly convex. Finally, the offloading ratio ρ can be obtained. When an application becomes active in the MD, the random variable with the uniform distribution in $[0, 1]$ is first generated by the offloading controller. Then, the value of this random variable is compared with a certain threshold (i.e., offloading ratio ρ). If it is no more than the threshold, the offloading controller decides that this application is to be offloading. Otherwise, it will be executed locally.

Rate-prediction Method Unlike the above method, we take account of the channel characters for more energy saving at the cost of larger complexity. When an application arrives at the cell m in the time t, the MD n sends the offloading

request to its donor eNodeB. Then, the eNodeB estimates the average transmission rate per RB of MD n according to its channel characters in the time t, i.e.

$$\tilde{\eta}_{RB,n}^{(m)}(t) = \frac{\sum_{k=0}^{K} \eta_{k,n}^{(m)}(t)}{K} \ , 0 \leq m \leq M \ , 0 \leq k \leq K \ . \tag{15}$$

Next, the average achievable data rate of user n is predicted by

$$\tilde{\mu}_{n}^{(m)}(t) = \frac{K \times \tilde{\eta}_{RB,n}^{(m)}(t)}{N_{m}^{(t)}} \ , 0 \leq m \leq M \ . \tag{16}$$

The delay due to uplink transmission is approximately estimated by

$$\tilde{\tau}_{U,n}^{(m)}(t) = \frac{D}{\tilde{\mu}_{n}^{(m)}}(t) \ . \tag{17}$$

If this estimated delay is no more than the local execution time, i.e., $\tilde{\tau}_{U,n}^{(m)}(t) \leq \tau_L$, this applicaiton is decided to be offloading. Otherwise, it has to be processed locally. This offloading decision is sent to the MD from the eNodeB. By this method, the system not only has a good control of the offloading ratio, but also choose the mobile users with better channel state as the offloading users, which can reduce the average delay effectively.

4 Simulation Results

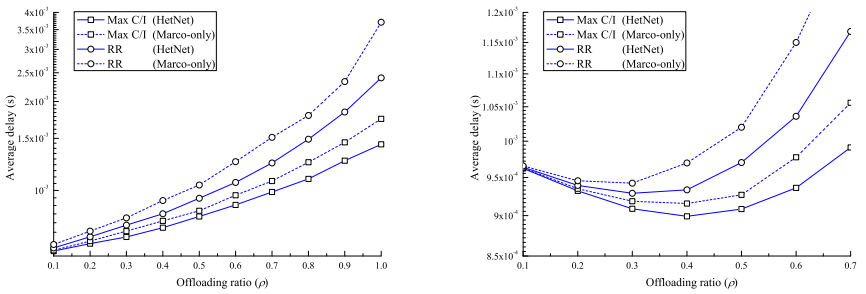
In this section, simulation results are presented to evaluate the delay performances of the proposed offloading control schemes in LTE-Advanced heterogenous networks (HetNet). Here only one macrocell with three picocell is considered. For comparison purpose, the performances of the network with macro-cell only are also given.

Detailed simulation parameters including channel model and system assumptions are summarized in Table 1 [10].

In order to determine the threshold value for offloading decision, the performance of the networks, in which the applications may be offloading according to the given probability, i.e., threshold, are firstly given. The offloading ratio ρ can be varied from 0 to 100%. In these simulations, different scheduling algorithms are adopted under different scenarios. Fig.1(a) shows that the average delay of users whose applications are decided to be offloading. It can be seen that the delay is increased with the larger offloading ratio. When the number of application that needed to be offloading becomes large, the radio resource that can be allocated to each user is decreased. Correspondingly, the achievable data rate of users become less so that the transmission delay is increased. Due to

Table 1. Parameters assumption in LTE-Advanced wireless networks.

Parameters	**Values**
Carrier (GHz)	2
Bandwidth (MHz)	10
Time slot duration (ms)	0.5
Resource block separation (kHz)	180
Number of resource blocks	50
Channel model	VA, Speed=3 km/h
Arrival Rate	2 applications/sub-frame
Offloading file size (Kbytes)	10
Target SNR (dB)	10
Transmit Power in eNodeB (dBm)	46 in Macro / 30 in Pico
UE Power Class (P_{max}) (dBm)	23
Antenna Configuration	Tx × Rx= 1 × 1
Pathloss model in Macro	128.1+37.6log10(R), R in km
Pathloss model in Pico	140.7+36.7log10(R), R in km
Scheduling algorithm	RR, Max C/I
Power Control	open loop with full pathloss compensation



(a) offloading users. (b) all users.

Fig. 1. Delay performances of (a) and (b) with different offloading ratio.

the scheduling gain, the performances of the networks with Max C/I algorithm are better than those with Round Rubin (RR) algorithm. Moreover, the delay in Hetnet is much less than that in macro-only network. It is because that the radio resources are reused between the macro-cell and pico-cell, which leads to more number of RBs to users for transmitting the offloading data. It can be observed that the average delay is increased with offloading ratio ρ and is the convex function of ρ under all simulated scenarios. According to the analysis in

Section 3, there exists a optimal offloading ratio ρ, which can be used as the threshold for offloading decision.

Then, the average delay performances of all users whose application are offloading are shown in Fig.1(b), where the local execution time is set to be fixed, i.e., $\tau_L = 1\ ms$. As expected, the average delay of all the users becomes smaller first and then larger with the increase of the offloading ratio ρ. It is noted that the optimal offloading ratio is highly dependent on the scheduling algorithm and network environment. For example, in HetNet scenarios, the optimal offloading ratio of 40% can be taken as the threshold in case of Max C/I scheduling algorithm while 30% is in case of RR scheduling algorithm. So, we have to find the different threshold values for the threshold-based offloading control scheme by simulations.

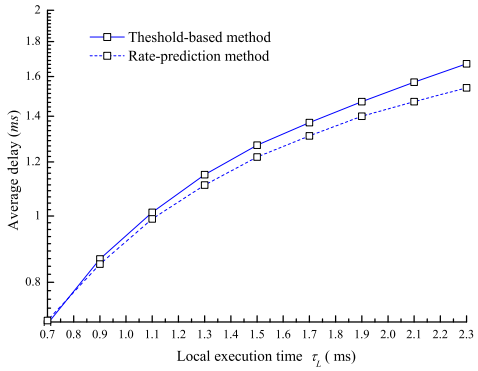


Fig. 2. Performance comparison with different offloading control methods in HetNet.

In Fig.2, we present the average delay performances of the network with our two proposed offloading control schemes while different local execution time is assumed. With the larger value of the local execution time, more applications are likely to be offloading so that the radio resource competition in the network becomes more intense. Then, the delay becomes larger due to the less achievable data rate for each user. Although the threshold-based scheme is simple, it can not always make the best decision. Compared to the threshold-based scheme, the delay performance of the network with the rate-prediction scheme is better no matter which scheduling algorithm is applied. It is because that not only the channel state information but also the number of all users are taken into account in the rate-prediction method.

5 Conclusion

Energy saving and execution speed gains have attracted increasingly attention due to the limitations of the mobile devices. In this article, we propose two offloading control schemes to meet the delay requirements of mobile cloud applications. The performances of the proposed methods with typical scheduling algorithms are studied under various scenarios. Execution delay can be reduced effectively by threshold-based scheme, which is simple but less flexible. Since not only the network environment but also the channel characters are taken into account, the rate-prediction scheme can achieve better delay performances and can be easily extended to various cases.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 61271183), Program for New Century Excellent Talents in University (NCET-11-0600), the National Key Technology R&D Program of China under Grant (2012ZX02001-2), and Chinese Universities Scientific Fund under Grant 2013RC0116.

References

1. H. T. Dinh, C. Lee, D. Niyato, and P. Wang.: A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Communications and Mobile Computing (2011)
2. Mayo R N, Ranganathan P.: Energy consumption in mobile devices: why future systems need requirementsCaware energy scale-down. Power-Aware Computer Systems. Springer Berlin Heidelberg. 3164, 26–40 (2005)
3. Rudenko A, Reiher P, Popek GJ, Kuenning GH.: Saving portable computer battery power through remote process execution. Journal of ACM SIGMOBILE on Mobile Computing and Communications Review. 2, 19–26 (1998)
4. K. Kumar, and Y.-H. Lu.: Cloud computing for mobile users: can offloading computation save energy?. computer. 43, 51–56 (2010)
5. Kremer U, Hicks J, Rehg J.: A compilation framework for power and energy management on mobile computers. Languages and Compilers for Parallel Computing. Springer Berlin Heidelberg. 2624, 115–131 (2003)
6. S. Gitzenis and N. Barnbos.: Joint Task Migration and Power Management in Wireless Computing. IEEE Trans. Mobile Computing. 8, 1189–1204 (2009)
7. Y. Wen, W. Zhang, and H. Luo.: Energy-optimal mobile application execution: taming resource-poor mobile devices with cloud clones. In: INFOCOM, 2012 Proceedings IEEE, pp. 2716–2720. IEEE, Orlando (2012)
8. D. Huang, P. Wang, and D. Niyato.: A dynamic offloading algorithm for mobile computing. IEEE Trans. Wireless Commun. 11, 1991–1995 (2012)
9. Lei L, Zhong Z, Zheng K, et al.: Challenges on wireless heterogeneous networks for mobile cloud computing. IEEE Wireless Communications. 20 (2013)
10. 3GPP TR 36.814, v2.0.0.: Further Advancements for E-UTRA, Physical Layer Aspects (2010)