Efficient Cloudlet Deployment: Local Cooperation and Regional Proxy

Dawei Li*, Jie Wu†, and Wei Chang ‡
*Department of Computer Science, Montclair State University, Montclair, NJ, 07043
†Department of Computer and Information Sciences, Temple University, Philadelphia, PA, 19121
‡Department of Computer Science, Saint Joseph's University, Philadelphia, PA, 19131
Email: dawei.li@montclair.edu, jiewu@temple.edu, wchang@sju.edu

***Abstract*—In this paper, we consider various cloudlet server deployment problems. Our goal is to reduce the cloudlet system's expected response time for mobile application requests submitted to the cloudlet. We propose two novel cloudlet server deployment strategies to achieve this goal. The first one is called local cooperation strategy, where two or more nearby cloudlet servers can help each other process the application requests. The second one is called regional proxy strategy, where service providers choose to deploy a regional proxy server that can provide service to mobile users from a comparatively larger area, while the communication delay from mobile users to the regional proxy server is kept low. We demonstrate the advantages of the two strategies over the traditional flat deployment strategy and the recently proposed hierarchical deployment strategy. Simulation results also verify that two proposed deployment strategies have great potential in reducing system's expected response time.**

***Index Terms*—Cloudlet, edge cloud, mobile computing, local cooperation, regional proxy.**

I. Introduction

In mobile cloud computing [1], mobile devices can utilize cloud computing facilities to process resource-hungry tasks. Recently, cloudlet becomes a promising facility to improve the Quality of Service (QoS) delivered to mobile applications. The key objective of a cloudlet is to deploy application servers closer to mobile devices [2]. In current practice, cloudlet servers are placed in isolated locations; they are independent from each other and are backed directly by remote clouds. In [3], a hierarchical deployment model is presented. When a local cloudlet server is overloaded by its local application requests, the requests will be sent to remote clouds, or a higher level server, where the possible long communication delay will again increase the system's response time perceived by mobile applications.

Intuitively, the way to mitigate this long communication delay problem is to create a shared computing resource pool that is close to the local servers or mobile users. In this paper, we propose two important cloudlet server deployment strategies to implement this idea. The first one is called Local Cooperation Strategy (LCS), which means that, instead of deploying independent servers, we let servers that are close to each other cooperate together to process application requests from the cooperating locations. In LCS, the peak capacity at each of the cooperating locations is increased; additionally, long communication delay is not involved, because local servers are close to each other. The second strategy is called Regional Proxy Strategy (RPS), of which the key idea is to choose a proxy location where it can provide service to users in multiple locations. In RPS, we allow users to send their application requests directly to their regional proxy server; the peak capacity at each location is also increased, and long communication delay can also be avoided, because the proxy location is chosen such that it is close to mobile users.

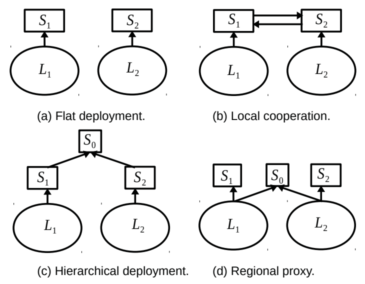


Fig. 1. Cloudlet server deployment strategies. S_0 represents the higher level server and the regional proxy server in (c) and (d), respectively. In (c), local servers send requests to S_0; in (d), some of the users in L_1 and L_2 send requests to S_0 directly.

A. Motivational Example

We provide a simple example to describe the motivation of our work. Assume that the cloud service providers need to provide service for users from two locations/areas, L_1 and L_2. Using state-of-the-art flat deployment method, we are to deploy two independent servers, S_1 and S_2, at L_1 and L_2, respectively, as shown in Fig. 1(a).

A main drawback of the flat deployment strategy is that, the capacity of S_1 and S_2 cannot be shared by their users. We propose to let servers S_1 and S_2 cooperate with each other such that one can send application requests to the other when itself is overloaded, as shown in Fig. 1(b).

Authors in [3] propose hierarchical deployment to reduce the communication delay that occurs when local cloudlet server has to send application requests to a remote cloud, as shown in Fig. 1(c). In this model, when local servers are overloaded, they first send the requests to a higher level server, S_0. This model provides low-delay shared resource for mobile users in multiple locations, compared to remote clouds. However, the communication delay to the higher level server may still be too high for delay-sensitive tasks.

To address this issue, we propose to deploy a regional proxy server which is close to some mobile users in both L_1 and L_2, if such a regional proxy location can be found.

This strategy is shown in Fig. 1(d). Like the hierarchical deployment method, RPS can provide shared resource for users at multiple locations. Since the proxy location is chosen such that the communication delay between mobile users and this proxy location is comparable to the delay between mobile users and the local servers, RPS enables resource sharing for users in multiple locations at no additional delay.

B. Main Contributions and Paper Organization

Our main contributions in this paper are as follows:

- We propose two novel cloudlet server deployment strategies that can help to reduce cloudlet system's response time for mobile applications.
- We consider important server capacity allocation problems for all of the four cloudlet deployment strategies using the queueing system model; we provide closed form and/or optimal solutions for most of the problems.
- Theoretical analysis and simulations verify that the proposed LCS and RPS have great potential in reducing system's response time for mobile applications.

The rest of the paper is organized as follows. Section II presents the system model and basic performance measurement. In Section III, we consider the server capacity allocation problems for both flat deployment and LCS. In Section IV, we consider the server capacity allocation problems for both hierarchical deployment and RPS. Related simulation results are provided in Section V. We conclude our paper in Section VI.

II. Preliminaries

We consider the scenario of a wireless metropolitan area network. The network has a set of access points that are deployed at n locations/areas, $L_1, L_2, \cdots, L_n$. The service provider deploys a cloudlet server S_i at L_i to process mobile application requests. We assume that the inter-arrival time of application requests at L_i follows exponential distribution with a mean value of $1/\lambda_i$. The execution time of all requests follows exponential distribution with a mean value of $1/\mu$ when they are executed at a reference speed. We say the processing speed is 1 when it equals the reference speed. Server S_i has a processing speed of c_i, which represents its capacity in processing mobile application requests. Then, the execution time of requests processed at S_i follows exponential distribution with a scaled mean value of $1/(c_i\mu)$.

An important argument that we will use frequently is that, requests that have exponentially distributed inter-arrival time with mean value of $1/\lambda$ is equivalent to requests that have Poisson distributed arrival rate with mean value of λ [4]. Given that the execution times of the the requests are exponentially distributed with a mean value of $1/\mu$, we can also say that the service rate is Poisson distributed with a mean value of μ. Consider a single location with a unit-speed server, the system can be modeled as an $M/M/1$ queueing system [4] with an arrival rate of λ and a service rate of μ. We define the *response time* for an application request submitted to the cloudlet as the time that it spent on waiting for the service plus the service time. Notice that, this response time includes the communication delay between the local cloudlet servers and other servers, but does not include the communication delay between mobile users and cloudlet servers. Essentially, the response time is the total amount of time that the request spends in the cloudlet system.

We have the following results for the queueing system. 1.) The system is stable, i.e., has bounded *expected response time*, if and only if $\mu > \lambda$. 2.) When the system is stable, the expected response time for all requests is $\bar{T} = \frac{1}{\mu-\lambda}$.

III. Optimal Server Capacity Allocation for Flat Deployment and LCS

A. Flat Deployment

We first consider the server capacity allocation problem using flat deployment. Assume that we only have the budget to provide a total computing capacity of C for all the servers. Location L_i has a request arrival rate of λ_i. Our server capacity allocation problem is to find the capacity allocation for all the servers such that the expected response time for all requests is minimized. In flat deployment, the expected response time for requests at S_i can be calculated as: $\bar{T}_{f,i} = \frac{1}{c_i\mu-\lambda_i}$.

Since a fraction of $\lambda_i/\sum_{i=1}^n \lambda_i$ of all the requests are processed at server S_i, the expected response time for all the requests in the system can be calculated as

$$\bar{T}_f = \sum_{i=1}^n \frac{\lambda_i}{\sum_{i=1}^n \lambda_i} \frac{1}{c_i\mu - \lambda_i}. \tag{1}$$

The server capacity allocation problem using flat deployment strategy can be formulated as follows (denoted as *Problem 1*):

$$\min \quad \bar{T}_f \tag{2}$$
$$s.t. \quad \lambda_i - c_i\mu < 0, \forall 1 \le i \le n. \tag{3}$$
$$\sum_{i=1}^n c_i - C = 0. \tag{4}$$

In this problem, c_i's are the optimization variables.

Theorem 1: *Problem 1* has feasible solutions if and only if $C > \sum_{i=1}^n \lambda_i/\mu$. Given that, The minimum value for $\bar{T}_f$ is:

$$\bar{T}_f^{min} = \frac{(\sum_{i=1}^n \sqrt{\lambda_i})^2}{(\mu C - \sum_{i=1}^n \lambda_i)\sum_{i=1}^n \lambda_i} \tag{5}$$

Proof: We apply KKT conditions to solve the problem [5]. Let x_i and y be the Lagrange multipliers associated with constraints in Equation (3) and Equation (4), respectively. For ease of presentation, let $\mathbf{c}$ and $\mathbf{x}$ be the vectors of c_i's and x_i's, respectively. Define the Lagrangian function as follows:

$$\mathcal{L}(\mathbf{c}, \mathbf{x}, y) = \bar{T}_f + \sum_{i=1}^n x_i(\lambda_i - c_i\mu) + y(\sum_{i=0}^n c_i - C). \tag{6}$$

Applying the KKT conditions, we have:

$$\frac{\partial \mathcal{L}(\mathbf{c}, \mathbf{x}, y)}{\partial c_i} = 0, \forall 1 \le i \le n. \tag{7}$$
$$x_i(\lambda_i - c_i\mu) = 0, \forall 1 \le i \le n. \tag{8}$$

Since $\lambda_i - c_i\mu \ne 0$, according to Equation (8), we have $x_i = 0$, $\forall 1 \le i \le n$. According to Equations in (7), we have

$$c_i = \sqrt{\frac{\lambda_i}{y\mu \sum_{i=1}^n \lambda_i}} + \frac{\lambda_i}{\mu}, \forall 1 \le i \le n. \tag{9}$$

According to Equation (4), we have

$$\sum_{i=1}^{n} c_i = \sum_{i=1}^{n} \sqrt{\frac{\lambda_i}{y\mu \sum_{i=1}^{n} \lambda_i}} + \frac{\sum_{i=1}^{n} \lambda_i}{\mu} = C. \tag{10}$$

For y to have real values, we must have $C > \sum_{i=1}^{n} \lambda_i/\mu$. Thus $C > \sum_{i=1}^{n} \lambda_i/\mu$ is the necessary condition for *Problem 1* to have feasible solutions. Given this, y can be solved as follows:

$$y = \frac{1}{\mu \sum_{i=1}^{n} \lambda_i} \frac{(\sum_{i=1}^{n} \sqrt{\lambda_i})^2}{(C - \sum_{i=1}^{n} \lambda_i/\mu)^2}. \tag{11}$$

Then,

$$c_i = \frac{(C - \sum_{i=1}^{n} \lambda_i/\mu)\sqrt{\lambda_i}}{\sum_{i=1}^{n} \sqrt{\lambda_i}} + \frac{\lambda_i}{\mu}. \tag{12}$$

Thus, the minimum value of $\bar{T}_f$ is:

$$\bar{T}_f^{min} = \frac{(\sum_{i=1}^{n} \sqrt{\lambda_i})^2}{(\mu C - \sum_{i=1}^{n} \lambda_i) \sum_{i=1}^{n} \lambda_i} \tag{13}$$

■

Next, we consider the server capacity allocation problem using LCS. Assume that all the n local servers form a cooperation group where any server can cooperate with any other server. Denote $p_{i,j}$ as the fraction of requests, that are generated from L_i and executed at server j. When $i = j$, it means that the requests are processed at the local server; when $i \neq j$, it means that this fraction of requests are sent to the jth server for processing.

B. LCS with Negligible Communication Delay

We consider the case where the cooperation communication delay is negligible first. With $p_{i,j}$'s, the effective request arrival rate at server S_i can be determined as: $\lambda_i^{'} = \sum_{j=1}^{n} p_{j,i}\lambda_j$, which is the sum of requests that are sent to S_i from all locations. Notice that we have $\sum_{j=1}^{n} p_{i,j} = 1$ and $\sum_{i=1}^{n} \lambda_i^{'} = \sum_{i=1}^{n} \lambda_i$. The expected response time for all requests processed at location L_i can be calculated as $\bar{T}_{c,i} = \frac{1}{c_i\mu - \lambda_i^{'}}$. Since a fraction of $\lambda_i^{'}/\sum_{i=1}^{n} \lambda_i$ of all the requests are processed at L_i, the expected response time for all requests can be calculated as:

$$\bar{T}_c = \sum_{i=1}^{n} \frac{\lambda_i^{'}\bar{T}_{c,i}}{\sum_{i=1}^{n} \lambda_i^{'}} = \sum_{i=1}^{n} \frac{\sum_{j=1}^{n} p_{j,i}\lambda_j}{\sum_{i=1}^{n} \lambda_i} \frac{1}{c_i\mu - \sum_{j=1}^{n} p_{j,i}\lambda_j}. \tag{14}$$

The server capacity allocation problem can be formulated as follows (denoted as *Problem 2*):

$$\min \quad \bar{T}_c \tag{15}$$

$$s.t. \quad \sum_{j=1}^{n} p_{j,i}\lambda_j - c_i\mu < 0, \forall 1 \leq i \leq n. \tag{16}$$

$$0 \leq p_{j,i} \leq 1, \forall 1 \leq i,j \leq n. \tag{17}$$

$$\sum_{i=1}^{n} p_{j,i} = 1, \forall 1 \leq j \leq n. \tag{18}$$

$$\sum_{i=0}^{n} c_i - C = 0. \tag{19}$$

Theorem 2: *Problem 2* is a convex optimization problem, which can be solved in polynomial time.

Proof: The optimization variables for *Problem 2* are c_i's ($\forall 1 \leq i \leq n$) and $p_{i,j}$'s ($\forall 1 \leq i,j \leq n$). We first prove that $\bar{T}_c$ is a convex function of c_i. Taking the derivative of $\bar{T}_c$ over c_i:

$$\begin{aligned}\frac{\partial \bar{T}_c}{\partial c_i} &= \frac{\sum_{j=1}^{n} p_{j,i}\lambda_j}{\sum_{i=1}^{n} \lambda_i} \frac{\partial \left(\frac{1}{c_i\mu - \sum_{j=1}^{n} p_{j,i}\lambda_j}\right)}{\partial c_i} \\ &= \frac{\sum_{j=1}^{n} p_{j,i}\lambda_j}{\sum_{i=1}^{n} \lambda_i} \frac{-\mu}{(c_i\mu - \sum_{j=1}^{n} p_{j,i}\lambda_j)^2}.\end{aligned} \tag{20}$$

The second order derivative of $\bar{T}_c$ over c_i is

$$\frac{\partial^2 \bar{T}_c}{\partial c_i^2} = \frac{\sum_{j=1}^{n} p_{j,i}\lambda_j}{\sum_{i=1}^{n} \lambda_i} \frac{2\mu^2}{(c_i\mu - \sum_{j=1}^{n} p_{j,i}\lambda_j)^3}. \tag{21}$$

Since $c_i\mu - \sum_{j=1}^{n} p_{j,i}\lambda_j > 0$, we have $(\partial^2 \bar{T}_c)/(\partial c_i^2) > 0$ within the feasible domain; thus, $\bar{T}_c$ is a convex function of c_i.

Then, we prove that $\bar{T}_c$ is a convex function of $p_{j,i}$. The partial derivative of $\bar{T}_c$ over $p_{j,i}$ is

$$\begin{aligned}\frac{\partial \bar{T}_c}{\partial p_{j,i}} &= \frac{\partial \left(\frac{\sum_{j=1}^{n} p_{j,i}\lambda_j}{\sum_{i=1}^{n} \lambda_i} \frac{1}{c_i\mu - \sum_{j=1}^{n} p_{j,i}\lambda_j}\right)}{\partial p_{j,i}} \\ &= \frac{1}{\sum_{i=1}^{n} \lambda_i} \frac{\partial \frac{1}{\frac{c_i\mu}{\sum_{j=1}^{n} p_{j,i}\lambda_j} - 1}}{\partial p_{j,i}} \\ &= \frac{1}{\sum_{i=1}^{n} \lambda_i} \frac{1}{(\frac{c_i\mu}{\sum_{j=1}^{n} p_{j,i}\lambda_j} - 1)^2} \frac{c_i\mu\lambda_j}{(\sum_{j=1}^{n} p_{j,i}\lambda_j)^2} \\ &= \frac{c_i\mu\lambda_j}{\sum_{i=1}^{n} \lambda_i} \frac{1}{(\sum_{j=1}^{n} p_{j,i}\lambda_j - c_i\mu)^2}.\end{aligned} \tag{22}$$

Then,

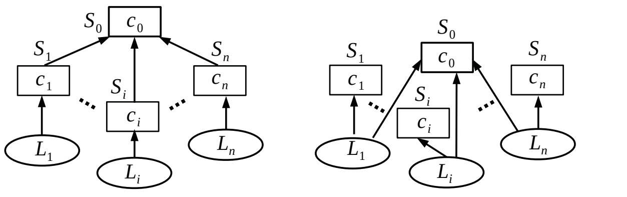
$$\frac{\partial^2 \bar{T}_c}{\partial p_{j,i}^2} = \frac{c_i\mu\lambda_j}{\sum_{i=1}^{n} \lambda_i} \frac{-2}{(\sum_{j=1}^{n} p_{j,i}\lambda_j - c_i\mu)^3}. \tag{23}$$

Since $\sum_{j=1}^{n} p_{j,i}\lambda_j - c_i\mu < 0$, we have $\partial^2 \bar{T}_c/\partial p_{j,i}^2 > 0$ within the feasible domain; thus, $\bar{T}_c$ is a convex function of $p_{j,i}$. So, the original optimization problem is a convex optimization problem that can be solved within polynomial time. ■

C. LCS with Non-negligible Communication Delay

Assume that the bandwidth between each pair of two servers is B, and that the communication volume for a request to be sent to a cooperation server is proportional to its execution time at unit speed, i.e., α/μ. The time it takes to send a request to a cooperation server is $t_c = \alpha/(B\mu)$. Requests that are processed locally are not affected by the communication delay; thus, the expected response time in each location as $\bar{T}_{c,i} = 1/(c_i\mu - \lambda_i^{'})$.

We consider the requests that originate from location L_i. Recall that $p_{i,j}$ is the fraction of λ_i that is executed on server S_j. If a request is sent to another server, then, its expected response time will be the expected response time of the other server plus the time it takes to send the request to the other server. Thus the expected response time for all requests that originate from location L_i can be calculated as follows:



(a) Hierarchical Deployment (b) RPS

Fig. 2. Hierarchical deployment and RPS. S_0 represents the higher level server and the regional proxy server in (a) and (b), respectively. In (a), local servers send requests to S_0; in (b), some users send requests to S_0 directly.

$$\bar{T}^{'}_{c,i} = p_{i,i}\bar{T}_{c,i} + \sum_{j=1,j\neq i}^{n} p_{i,j}(\bar{T}_{c,j} + t_c). \tag{24}$$

The server capacity allocation problem can be formulated as follows (denoted as *Problem 3*):

$$\begin{aligned} \min \quad & \bar{T}^{'}_{c} = \sum_{i=1}^{n} \frac{\lambda_i}{\sum_{i=1}^{n}\lambda_i}\bar{T}^{'}_{c,i} \qquad (25)\\ s.t. \quad & \text{Equations. (16)-(19).} \end{aligned}$$

Theorem 3: *Problem 3* is a convex optimization problem, which can be solved in polynomial time.

Proof: Problem 3 has the same constraints as that of *Problem 2*. We only need to prove the convexity of $\bar{T}^{'}_{c}$ using similar approaches as in the proof for Theorem 2. Due to space limit and its redundancy, we omit the proof here. ■

We can observe that, if we let $p_{i,i} = 1$ and let $p_{i,j} = 0$ when $i \neq j$ in *Problem 3*, then we have $\bar{T}^{'}_{c,i} = \bar{T}_{c,i}$, and $\lambda^{'}_{i} = \lambda_i$, which further suggests that $\bar{T}^{'}_{c,i} = \bar{T}_{c,i} = \bar{T}_{f,i}$. It indicates that *Problem 3* reduces to *Problem 1* given the $p_{i,j}$ values. In other words, *Problem 1* is a special case of *Problem 3*. Similarly, *Problem 1* is a special case of *Problem 2*. Thus, the minimum objective function values of *Problem 3* and *Problem 2* are less than or equal to that of *Problem 1*.

IV. Optimal Server Capacity Allocation for Hierarchical Deployment and RPS

A. Hierarchical Deployment

We still assume that we only have the budget to provide a total computing capacity of C. For hierarchical deployment, the higher level server, S_0 as shown in Fig. 2(a), has a capacity of c_0. The time to send a request to S_0 from a local server is: $t_h = \alpha/(b\mu)$, where b is the bandwidth between local servers to the higher level server. Denote that a fraction of p_i of all the requests from location L_i are sent to S_0 from S_i.

The local server S_i has an effective arrival rate of $(1 - p_i)\lambda_i$ and an effective service rate of $c_i\mu$. Thus, the expected response time for requests processed at S_i is: $\bar{T}_{h,i} = \frac{1}{c_i\mu-(1-p_i)\lambda_i}$. S_0 has an effective arrival rate of $\sum_{i=1}^{n} p_i\lambda_i$, and an effective service rate of $c_0\mu$. Thus, the expected response time for S_0 is: $\bar{T}_{h,0} = \frac{1}{c_0\mu-\sum_{i=1}^{n} p_i\lambda_i} + t_h$.

The expected response time for all requests in the system can be calculated as:

$$\begin{aligned} \bar{T}_h = &\sum_{i=1}^{n}\left\{\frac{(1-p_i)\lambda_i}{\sum_{i=1}^{n}\lambda_i}\frac{1}{c_i\mu-(1-p_i)\lambda_i}\right\}\\ &+\frac{\sum_{i=1}^{n}p_i\lambda_i}{\sum_{i=1}^{n}\lambda_i}\left(\frac{1}{c_0\mu-\sum_{i=1}^{n}p_i\lambda_i}+t_h\right). \end{aligned} \tag{26}$$

In hierarchical deployment, from the service provider's point of view, we have control over all the local servers as well as the higher level server. We can control p_i's to achieve the best performance. The server capacity allocation problem can be formulated as follows (denoted as *Problem 4*):

$$\begin{aligned} \min \quad & \bar{T}_h && (27)\\ s.t. \quad & 0 \le p_i \le 1. \forall 1 \le i \le n. && (28)\\ & \sum_{i=1}^{n} p_i\lambda_i - c_0\mu < 0. && (29)\\ & (1-p_i)\lambda_i - c_i\mu < 0, \forall 1 \le i \le n. && (30)\\ & \sum_{i=0}^{n} c_i - C = 0. && (31) \end{aligned}$$

Theorem 4: *Problem 4* is a convex optimization problem that can be solved in polynomial time.

Proof: The proof is similar to that of Theorem 2. Due to space limit and its redundancy, we omit the proof here. ■

B. RPS

In RPS, S_0 is the regional proxy server, and p_i is still the fraction of requests that originate from L_i and sent to S_0. The average response time for all requests is:

$$\begin{aligned} \bar{T}_p = &\sum_{i=1}^{n}\left\{\frac{(1-p_i)\lambda_i}{\sum_{i=1}^{n}\lambda_i}\frac{1}{c_i\mu-(1-p_i)\lambda_i}\right\}\\ &+\frac{\sum_{i=1}^{n}p_i\lambda_i}{\sum_{i=1}^{n}\lambda_i}\frac{1}{c_0\mu-\sum_{i=1}^{n}p_i\lambda_i}. \end{aligned} \tag{32}$$

We can see that, given the same p_i's, using RPS can result in a smaller expected response than that of hierarchical deployment, due to the lack of the communication delay, t_h. In practical systems, we may not have the same p_i's for the two strategies. It is definitely not true that RPS is always superior to hierarchical deployment. Our goal is to show that RPS has the potential to reduce system's response time.

The server capacity allocation problem using RPS can be formulated as follows (denoted as *Problem 5*):

$$\begin{aligned} \min \quad & \bar{T}_p \qquad (33)\\ s.t. \quad & \text{Equations (29)-(31)} \end{aligned}$$

We consider the problem with fixed p_i's, i.e., only c_i's are optimization variables. The reason is that in practice, though a service provider have full control over the local servers and the proxy server, it may not have control over mobile users' behaviors.

Theorem 5: *Problem 5* has feasible solutions if and only if $C > \sum_{i=1}^{n}\lambda_i/\mu$. Given that, the minimum value of $\bar{T}_p$ is:

$$\bar{T}_p^{min} = \frac{\sqrt{\sum_{i=1}^{n}\lambda_i} + \sum_{i=1}^{n}\sqrt{(1-p_i)\lambda_i}}{\mu C - \sum_{i=1}n\lambda_i}. \tag{34}$$

Proof: We use the same techniques as that in Theorem 1. Details are omitted due to space limit. ■

V. Simulations

A. Simulation Settings

The mean execution time of all requests are set as $1/\mu = 0.4$, with the units as seconds. The inter-arrival times $1/\lambda_i$ of application requests are randomly generated within $[0.5, 1]$. We choose $\alpha = 100$ such that the communication delay is comparable to execution time with practical bandwidth values.

In the first group of simulations, we compare the optimal expected response time of five cloudlet systems. The first system, denoted as "Flat", uses the purely independent flat deployment. The second one, denoted as "Coop w/o delay" represents a system using LCS without cooperation communication delay. The third one, denoted as "Coop with delay", uses LCS with communication delay. The fourth one, denoted as "Coop fixed", uses LCS with fixed server capacity; the fixed server capacity is naively set as C/n. The fifth one, denoted as "No coop fixed", is a system without cooperation and the servers' capacities are naively set to the same values as in "Coop fixed". For each simulation, we change the bandwidth for local cooperation from 20 Mbps to 1000 Mbps. We conduct two sets of simulations, with each simulation targeting systems with $n = 5$ and 10 locations, respectively.

In the second group of simulations, we also compare the optimal expected response time of five cloudlet systems. For comparing these systems, we keep p_i's the same in each system; also, p_i's take the same value for different locations. The first system "Regional proxy" uses RPS. The second ("Hierchical 1"), third ("Hierchical 2"), fourth ("Hierchical 3"), and fifth ("Hierchical 4") use hierarchical deployment with communication bandwidths as 20 Mbps, 50 Mbps, 100 Mbps, and 500 Mbps, respectively. For each simulation, we vary p_i's from 5% to 60%, with a step size of 5%. We conduct two sets of simulations, with each simulation targeting systems with $n = 5$, and 10 locations, respectively.

B. Simulation Results

The results for the first comparison group are presented in Fig. 3. We can see that LCS without considering the communication delay has the minimum expected response time, which is consistent with our theoretical results. Taking the communication delay into consideration ("Coop with delay"),the system's response time is still no greater than that of "Flat". when bandwidth is low and delay is large, "Coop with delay" chooses not to cooperate; when bandwidth is large, "Coop with delay" chooses to cooperate and thus achieves smaller response time. When the bandwidth for cooperation increases, the system's response time decreases, because the time spent on sending the requests is reduced. Comparing "Coop fixed" against "No coop fixed", we can see that, even when the capacity on the servers are fixed, allowing local cooperation among them still has the potential to reduce the system's response time.

The results for the second comparison group are presented in Fig. 4. If RPS is possible, "Regional proxy" always achieves the minimum response time. When the p_i increases, meaning that more requests are sent to the regional proxy center, the system's expected response time decrease. When p_i is too small, increasing p_i may result in slightly increased response time; this means that when few users can send requests to a regional proxy server, it is better not to adopt RPS. Using hierarchical deployment, due to the communication delay between local servers and the higher level server, the system's response time is much greater than that of RPS. Also, as p_i increases, the system's expected response time may increase or reduce, which is the results of two conflicting factors: when p_i increase, more requests can enjoy the shared resource, and their execution time reduces; however, more requests will experience the communication delay at the same time. If the bandwidth from local servers to the higher level server is small, the system's response time increase as p_i increases. If the bandwidth is large, the system's response time may decrease with p_i at some point.

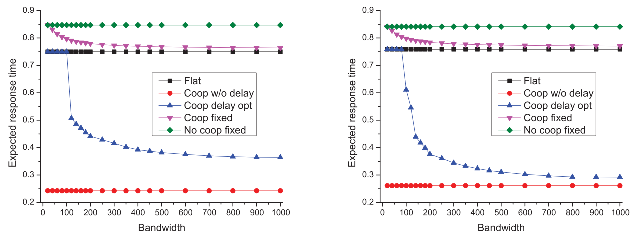


Fig. 3. Comparisons between flat deployment and LCS. $n = 5$ and 10 for the left and right sub-figures, respectively.

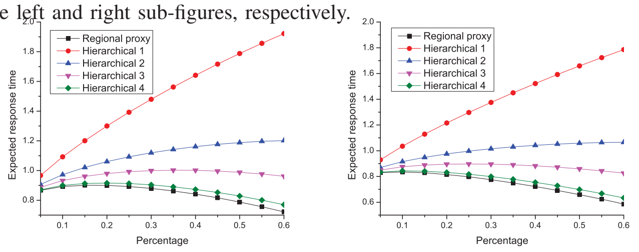


Fig. 4. Comparisons between hierarchical deployment and RPS. $n = 5$ and 10for the left and right sub-figures, respectively.

VI. Conclusions

In this paper, we consider the cloudlet service problem from the service provider's point of view. We propose two cloudlet server deployment strategies that can help to reduce system's response time for mobile applications. We demonstrate the potential advantages of the two strategies over traditional flat deployment and recently proposed hierarchical deployment methods.

References

[1] H. Qi and A. Gani, "Research on mobile cloud computing: Review, trend and perspectives," in *2nd International Conference on Digital Information and Communication Technology and it's Applications*, May 2012, pp. 195–202.

[2] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, Oct. 2009.

[3] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *IEEE INFOCOM*, Apr. 2016.

[4] D. Gross, J. F. Shortle, J. M. Thompson, and C. M. Harris, *Fundamentals of Queueing Theory*, 4th ed. New York, NY, USA: Wiley-Interscience, 2008.

[5] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.