Workload Allocation in Hierarchical Cloudlet Networks

Citation:
Q. Fan and N. Ansari, "Workload Allocation in Hierarchical Cloudlet Networks," *IEEE Communications Letters*, vol. 22, no. 4, pp. 820-823, April 2018.

URL:
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8280558

Workload Allocation in Hierarchical Cloudlet Networks

Qiang Fan, *Student Member, IEEE,* and Nirwan Ansari, *Fellow, IEEE*

***Abstract*—Edge cloudlets are promising to mitigate the high network delay incurred by the remote cloud in executing workloads offloaded from a user equipment (UE). However, the response time of a task request consists of both the network delay and computing delay. Considering the spatial and temporal dynamics of workloads among cloudlets, if the workload of an edge cloudlet is heavy, the computing delay in the cloudlet may be unbearable. In this letter, we design a hierarchical cloudlet network and propose a Workload ALLocation (WALL) scheme to minimize the average response time of UEs' requests by deciding which cloudlet a UE is assigned to and how much computing resource is provisioned to serve it. The performance of the proposed scheme is validated by extensive simulations.**

***Index Terms*—Cloudlet, workload allocation, edge computing, resource allocation.**

I. Introduction

RECENT mobile applications are increasingly computation-intensive while resources of User Equipments (UEs) remain limited. Thus, mobile cloud computing has been introduced to offload UEs' task requests to a centralized cloud in Internet. However, the long distance between a UE and the centralized cloud inherently incurs relatively high latency. The response time may satisfy the requirement of some applications such as web browsing, but is unbearable for many delay-sensitive applications, such as augmented reality, on-line gaming, and image processing. The concept of cloudlets has thus been employed to reduce the network delay by moving the remote cloud resources to the network edge. Since cloudlets are generally placed at access points that are close to UEs, UEs can access the computing resources with a lower network delay. Recent works have already shown that distributed cloudlets can remarkably reduce the response time of task requests. Sun and Ansari [1] proposed a LEAD algorithm to allocate UEs' dedicated virtual machines (VMs) among cloudlets to minimize the network delay by having a number of disk replicas of each UE's virtual VM placed among cloudlets. Kiani *et al.* [2] proposed a hierarchical cloudlet architecture based on the traditional mobile network and designed a mechanism to allocate dedicated virtual machines of hierarchical cloudlets and communications resources to UEs in order to maximize the profit of a service provider (note that the amount of computing resources for UEs are given). In addition, Tong *et al.* [3] proposed a workload placement algorithm in a hierarchical edge network, which selects the cloudlet and allocates the computing resources for each task. However, for delay-sensitive applications, the number of task requests in the network is huge and the size of each request is small; thus, it is hard to run the algorithm in real-time. Rodrigues *et al.* [4], [5] proposed a method to improve the transmission delay and processing delay, respectively, via transmission power control and VM migration among cloudlets co-located with their BSs instead of hierarchical cloudlets.

In a hierarchical cloudlet network where different tiers of cloudlets form a tree topology (i.e., tier-1 cloudlets are attached with access points while each tier-2 cloudlet acts as a high-level cloudlet covering a number of tier-1 cloudlets), the workload allocation among different tiers of cloudlets has a crucial impact on the response time for UEs. As tier-1 cloudlets are closer to UEs, offloading a UE's requests to a tier-1 cloudlet yields a lower network delay than a tier-2 cloudlet. On the other hand, since the computing resource of a tier-2 cloudlet is much richer than a tier-1 cloudlet, offloading a UE's requests to a tier-2 cloudlet incurs a lower computing delay than a tier-1 cloudlet with limited resources. Furthermore, owing to the spatial and temporal dynamics of user distribution, when a tier-1 cloudlet is overloaded, the computing delay tends to be the dominating factor for the response time of requests. Thus, some UEs' workloads should be offloaded to tier-2 cloudlet to decrease their computing delay, although the network delays are relatively deteriorated.

To minimize the response time, we propose a Workload ALLocation (WALL) scheme for hierarchical cloudlet networks, where both the network delay and computing delay are taken into account. Below are the main contributions of this letter: 1. We propose a novel hierarchical cloudlet network architecture to enable UEs to be assigned to suitable cloudlets. 2. We formulate the problem of minimizing the average response time by offloading UEs' workloads among different tiers of cloudlets and allocating optimal computing resources for each UE. 3. In the workload allocation, the QoS constraint in terms of the response time threshold is satisfied for each UE. 4. We design the novel WALL algorithm to solve the problem and demonstrate its performance via simulations.

II. System Model

A hierarchical cloudlet network architecture is illustrated in Fig. 1, where the software defined network (SDN) based cellular core, which consists of a SDN controller and openflow switches, is employed to provide flexible routes and communications resources between BSs. Tier-1 cloudlets are co-located with base stations (BSs) and tier-2 cloudlets are placed at openflow switches, each of which connects to several BSs. Meanwhile, mobile providers offer seamless wireless

Q. Fan, N. Ansari are with Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, 07102 USA (Email: {qf4, nirwan.ansari}@njit.edu).

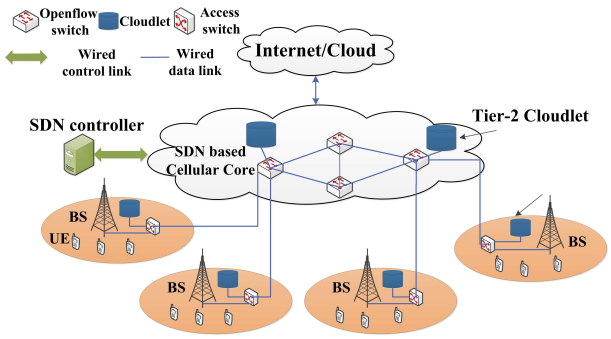

Fig. 1. Cloudlet network architecture.

communications between a UE and its BS, and thus each UE can access its BS and then connect to different tiers of cloudlets. As shown in Fig. 1, the cloudlet network forms a tree topology, i.e., the workload of a tier-1 cloudlet can also be offloaded to its corresponding tier-2 cloudlet. Note that as compared to tier-1 cloudlets, tier-2 cloudlets have more powerful computing resources. Within one cloudlet, each UE is allocated an amount of computing resource according to its workload. Hence, we assume each UE forms a queuing model to process its tasks requests, where the average computing delay (i.e., consisting of the average queuing delay and average processing delay) depends on the task arrival rate of the UE and its allocated computing resources (i.e., the service rate).

TABLE I
THE IMPORTANT NOTATIONS

Symbol	Definition
x_{ij}	Binary indicator of UE j being assigned to cloudlet i or not.
μ_{ij}	Computing resources allocated to UE j in cloudlet i.
I_j	Set of potential cloudlets (i.e., in different tiers) for UE j.
d_{ij}	Network delay between UE j and cloudlet i.
C_i	Computing capacity of cloudlet i.
T_j	Delay threshold for UE j.
Z	Set of tier-2 cloudlets.

Denote $\mathcal{I}$ as the set of cloudlets, $\mathcal{J}$ as the set of UEs, and $\mathcal{K}$ as the set of BSs. Meanwhile, let I_j be the set of potential cloudlets for UE j, i.e., cloudlets in different tiers for UE j. For example, UE j is associated with a BS, which is co-located with tier-1 cloudlet A; cloudlet A connects to a tier-2 cloudlet B. Thus, set I_j of UE j is composed of cloudlet A and B. The key notations of this letter are summarized in Table I.

A. Average Computing Delay

Assume that task requests of each UE j are generated according to a Poisson Process with the average arrival rate λ_j, and the size of tasks in terms of the CPU cycles follows an exponential distribution with the average value of l_j. Let μ_{ij} be the computing resources (i.e., CPU cycles per second) allocated to UE j in cloudlet i in a time slot. Then, for a given μ_{ij} in the time slot, the service time for UE j's requests also follows an exponential distribution, where the average service time can be expressed as l_j/μ_{ij}. Consequently, each UE can realize an M/M/1 queuing model to process its task requests in its cloudlet. To keep the queue system stable, we need to guarantee that λ_j is smaller than the service rate (μ_{ij}/l_j), i.e., $\mu_{ij}/l_j - \lambda_j > 0$. We define the average computing delay of UE j in cloudlet i as the average system delay of the UE (consisting of the queuing delay and processing delay), and thus we have:

$$t_{ij} = \frac{1}{\mu_{ij}/l_j - \lambda_j}. \tag{1}$$

B. Network Delay

When a UE request is sent to a cloudlet, the request goes through the BS and the SDN-based cellular core network. Hence, the E2E delay between a UE and its cloudlet consists of two parts: first, the E2E delay between the UE and its BS, i.e., the wireless delay; second, the E2E delay between the BS and the cloudlet that hosts the UE's workload. However, the cloudlet selection for a UE does not affect its wireless delay, which only depends on the UE's service plan and the mobile provider's bandwidth allocation strategy [6]. Thus, in this letter, we just define the network delay between a UE and its associated cloudlet as the E2E delay between the UE's BS and the cloudlet. Note that we assume that each UE can be associated with only one BS. Denote τ_{ki} as the E2E delay between BS k and cloudlet i, and $\mathcal{Y}$ as a given indicator matrix to reflect the UE-BS association at the beginning of each time slot, in which $y_{kj} \in \mathcal{Y}$ represents UE j being covered by BS k or not. As each UE is associated with only one BS, we have $\sum_k y_{kj} = 1, \forall j \in \mathcal{J}$. Note that the value of τ_{ki} can be measured and recorded by the SDN controller [7]. Thus, the network delay between UE j and its potential cloudlet $i \in \mathcal{I}_j$ can be expressed as

$$d_{ij} = \sum_k y_{kj}\tau_{ki}, \quad \forall j \in \mathcal{J}, \ \forall i \in \mathcal{I}_j. \tag{2}$$

III. PROBLEM FORMULATION

The main purpose of this letter is to minimize the average response time for all UEs in the hierarchical cloudlet network. Thus, we formulate the workload allocation problem, i.e., minimizing the average response time by offloading UEs' workload among different tiers of cloudlets and optimally allocating the computing resources of each cloudlet to UEs in each time slot, as follows:

$$P1: \min_{x_{ij},\mu_{ij}} \sum_{j\in J}\sum_{i\in I_j} x_{ij}(d_{ij} + \frac{1}{\mu_{ij}/l_j - \lambda_j}) \tag{3}$$

$$s.t. \sum_{j\in J} x_{ij}\mu_{ij} \le C_i, \forall i \in I, \tag{4}$$

$$\sum_{i\in I_j} x_{ij} = 1, \forall j \in J, \tag{5}$$

$$x_{ij}(d_{ij} + \frac{1}{\mu_{ij}/l_j - \lambda_j}) \le x_{ij}T_j, \forall i \in I, \forall j \in J, \tag{6}$$

$$x_{ij} \in \{0,1\}, \forall i \in I, \forall j \in J. \tag{7}$$

Here, Constraint (4) imposes computing resources allocated to UEs to be no more than the cloudlet's computing capacity.

Constraint (5) ensures that the workload of each UE is assigned to only one cloudlet. Constraint (6) imposes the response time of UE j to be smaller than its delay threshold.

The above optimization problem is a mixed integer non-linear programming problem. The resource allocation in each cloudlet (i.e., determining μ_{ij}) depends on the UE-cloudlet association x_{ij}. In order to achieve the optimal UE-cloudlet association, a brute-force search leads to $O(M^N)$ iterations, where N represents the number of UEs and M is the number of cloudlet tiers. Obviously, the computational complexity of the brute-force search increases exponentially with respect to the number of UEs. Thus, it is not practical to execute the optimization in a time slot especially for large scale networks.

IV. The WALL Algorithm

In this section, we propose the WALL algorithm to effectively assign UEs to suitable cloudlets that approaches the optimal solution with low computational complexity. For brevity, we define UEs covered by a tier-2 cloudlet as the UEs covered by BSs connected with the tier-2 cloudlet. The basic idea is to iteratively select a suitable UE which has the maximum workload among UEs covered by a tier-2 cloudlet, and assign it to the cloudlet that incurs the lowest response time. When UEs covered by a tier-2 cloudlet are assigned, we will sequentially check for the next tier-2 cloudlet.

A. Resource Allocation

In each iteration, when a new UE is assigned to a cloudlet, based on workloads of UEs in the cloudlet, we can optimally allocate the computing resources of the cloudlet to the corresponding UEs such that their average response time is minimized. In other words, based on a given UE-cloudlet association, the original problem P1 can be transformed into a resource allocation problem for each cloudlet i as follows:

$$P2: \quad \min_{\mu_{ij}} \sum_{j\in J} x_{ij}(d_{ij} + \frac{1}{\mu_{ij}/l_j - \lambda_j}) \quad (8)$$
$$s.t. \;\; Constraints(4), (5), (6), (7).$$

We then have the following lemma:

Lemma 1. *When each x_{ij} is determined, P_2 is a convex optimization problem.*

Proof: For brevity, let $f = \sum_{j\in J} x_{ij}(d_{ij} + \frac{1}{\mu_{ij}/l_i - \lambda_{ij}})$, and we use μ_j to substitute μ_{ij} in cloudlet i. Thus, we have

$$\frac{\partial^2 f}{\partial \mu_k \partial \mu_j} = \begin{cases} 2l_j^{-2}(\mu_j/l_j - \lambda_j)^{-3}, & \text{if } k = j, \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

Since $(\mu_j/l_j - \lambda_j) > 0$, the Hessian matrix $\boldsymbol{H} = \frac{\partial^2 f}{\partial \mu_k \partial \mu_j}$ of f is a positive definite matrix. As a result, f is convex. Moreover, because Constraints (4), (5), (6), (7) are linear, the optimization problem P_2 is a convex optimization problem. ∎

As P_2 is a convex problem, we can derive the optimal solution of P_2 by solving the corresponding KKT conditions [8]. Consequently, the optimal response time for each UE in cloudlet i can be obtained.

B. UE-cloudlet Association

Denote $\mathcal{Z}$ as the set of tier-2 cloudlets and J_z as the set of UEs covered by tier-2 cloudlet z. For UEs covered by tier-2 cloudlet z, we can find the UE with the maximum workload, which has not been assigned, as follows:

$$j^* = \underset{j}{argmax} \left\{ \lambda_j | \sum_{i\in\mathcal{I}} x_{ij} = 0, \; j \in J_z \right\}. \quad (10)$$

Meanwhile, the optimal cloudlet i^* that incurs the minimum average response time for UE j^* can be expressed as follows:

$$i^* = \underset{i\in I_{j^*}}{\arg\min} \left\{ d_{ij^*} + \frac{1}{\mu_{ij^*}/l_{j^*} - \lambda_{j^*}} \right\}. \quad (11)$$

Specifically, WALL, as shown in Algorithm 1 below, starts with an initial UE-cloudlet association matrix $\mathcal{X}$, which is set to be a zero matrix. Second, for each tier-2 cloudlet, we sort its covered UEs in descending order based on their workloads. Then, we sequentially assign each UE j to a potential cloudlet $i \in I_j$ that incurs the minimum response delay. Third, when all UEs covered by a tier-2 cloudlet have already been assigned, we execute the same procedure for users covered by the next tier-2 cloudlet. Fourth, the algorithm terminates when all UEs in the cloudlet network are assigned to different tiers of cloudlets. The complexity of the sorting operation (i.e., Step 3 of Algorithm 1) is $O(|J_z|\ log(|J_z|))$ and Step 3 is repeated for $|Z|$ times. After the sorting operation, the complexity of each iteration is $O(M)$, where M is the number of cloudlet tiers; the total number of iterations can be expressed as $|\mathcal{J}|$. Thus, the complexity of Algorithm 1 is $O(M|J| + \sum_{z\in Z} |J_z| log(|J_z|))$.

Algorithm 1 WALL algorithm

Input: The user-BS association matrix $\mathcal{Y} = \{y_{kj} | k \in \mathcal{K}, \; j \in \mathcal{I}\}$. The matrix of E2E delay between BSs and cloudlets $\mathcal{T} = \{\tau_{ki} | k \in \mathcal{K}, \; i \in \mathcal{I}\}$. The vector of the average task arrival rate for UEs $\Lambda = \{\lambda_j | j \in \mathcal{J}\}$.

Output: The UE-cloudlet association matrix, i.e., $\mathcal{X} = \{x_{ij} | i \in \boldsymbol{\mathcal{I}}, j \in \boldsymbol{\mathcal{J}}\}$.

1: Initialize $\mathcal{X} = 0$ and let $z = 1$;
2: **while** $z \leq |\mathcal{Z}|$ **do**
3: Sort all UEs covered by tier-2 cloudlet z (i.e., J_z) in descending order of UEs' workloads;
4: Let $n = 1$;
5: **while** $n \leq |J_z|$ **do**
6: $j^* = n$ (i.e., find the optimal UE j^*);
7: Find the optimal cloudlet i^* for UE j^* by Eq. (11);
8: Let $x_{i^*j^*} = 1$ and $n = n + 1$;
9: **end while**
10: $z = z + 1$;
11: **end while**

Note that if a user cannot find an edge cloudlet to enable the user and the cloudlet's existing associated users to achieve lower response time than their delay threshold, both tier-1 and tier-2 cloudlets of the user are considered to be full and the user's tasks will be assigned to the remote cloud.

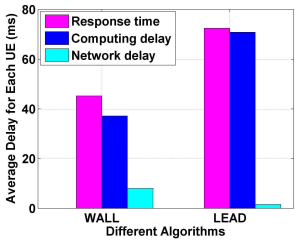

Fig. 2. Average performance of a UE for different schemes.

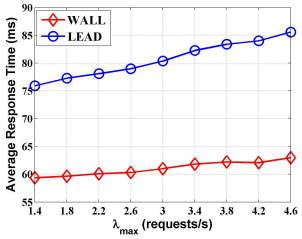

Fig. 3. Average response time with respect to λ_{max} ($C_{T1} = 1.5*10^5$ and $C_{T2} = 15*10^5$).

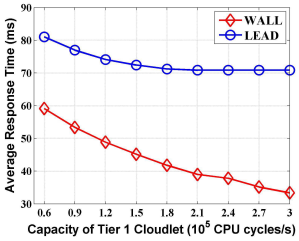

Fig. 4. Average response time with respect to capacities of tier-1 ($\lambda_{max} = 1.8$, $C_{T2} = 22.5 * 10^5$).

V. Results and Discussion

We set up the simulation to demonstrate the performance of WALL. For comparison, we select the LatEncy aware Avatar hanDoff (LEAD) algorithm [1] in which UEs' requests are offloaded to their closest cloudlets (i.e., with the minimum network delay), computing resources for UEs are given. The simulation environment consists of 2 tier-2 cloudlets and 25 tier-1 cloudlets (i.e., each tier-1 cloudlet is attached to a BS) within an area of 25 km^2, where the coverage of each BS is 1 km^2. Let tier-1 cloudlet 1-13 connect to the first tier-2 cloudlet, and the rest connect to the second tier-2 cloudlet. Meanwhile, 300 UEs are randomly distributed among the BSs and assumed to be associated to its closest BS. As each UE's task arrival rate follows a Poisson distribution, we randomly choose the average task arrival rate of each UE between 0 and λ_{max} in each time slot (i.e., 3 mins). The average size of requests in each UE is chosen according to the Normal distribution with an average of 1000 CPU cycles and a variance of 200 cycles, i.e., $N(1000, 200)$. Let C_{T1} be the capacity of each tier-1 cloudlet, and C_{T2} be the capacity of each tier-2 cloudlet, where the unit is CPU cycles/second. Moreover, the network delay between each tier-1 cloudlet and its corresponding tier-2 cloudlet is chosen according to $N(30, 10)$ (in ms); the network delay between each tier-2 cloudlet and the remote cloud is selected according to $N(90, 30)$ (in ms); the delay threshold is set as 150 ms.

Fig. 2 shows the average response time per UE, in which WALL yields lower response time as compared to LEAD. Specifically, WALL incurs higher network delay since LEAD always assigns UEs' requests to their closest cloudlet. However, WALL considers both the network delay and computing delay in the workload allocation and flexibly allocates computing resources for UEs based on their workloads, and thus significantly reduces the computing delay.

We further analyze how the workloads of UEs affect the performances of the two algorithms. Note that the value of λ_{max} reflects the workloads of UEs, i.e., increasing λ_{max} increases workloads of UEs. As shown in Fig. 3, with the increase of λ_{max}, the average response time of both WALL and LEAD increase gradually. However, the average response time of WALL is remarkably lower and grows slowly as compared to LEAD because when UEs' workloads are heavy, WALL can offload UEs' workloads to a lightly loaded cloudlet, which can allocate enough computing resources to UEs, thus remarkably reducing their average computing delay.

Moreover, we analyze the impact of tier-1 cloudlets' capacities on the average response time. Fig. 4 shows the average response time of WALL and LEAD when the capacities of tier-1 cloudlets increase. It can be seen that the average response time of WALL decreases more quickly than LEAD. As the computing resource for each UE is given, LEAD just considers the network delay, and thus assigns more UEs to tier-1 cloudlets with the increase of capacities of these cloudlets. In particular, when tier-1 cloudlets' capacities are very high, most UEs in LEAD are assigned to tier-1 cloudlets, and thus the average response time tends to be stable. In contrast, WALL can flexibly allocate the computing resources of a cloudlet to UEs. When the capacity of a tier-1 cloudlet is low, WALL allocates many UEs to the tier-2 cloudlet, where the low computing delay overweighs the increment of network delay. Meanwhile, when the capacity of a tier-1 cloudlet increases, it incurs lower computing delay; thus, WALL assigns more UEs from the tier-2 cloudlet to tier-1 cloudlets to further reduce their response time by reducing their network delay.

VI. Conclusion

In this letter, we have proposed the Workload ALLocation (WALL) scheme for hierarchical cloudlet networks. WALL assigns UEs to different tiers of cloudlets and optimally allocates the computing resources of each cloudlet to its associated UEs in each time slot. Simulation results have verified WALL.

References

[1] X. Sun and N. Ansari, "Avaptive avatar handoff in the cloudlet network," *IEEE Trans. on Cloud Computing*, 2017, DOI:10.1109/TCC.2017.2701794, early access.

[2] A. Kiani and N. Ansari, "Towards hierarchical mobile edge computing: An auction-based profit maximization approach," *IEEE Internet of Things Journal*, 2017, DOI: 10.1109/JIOT.2017.2750030, early access.

[3] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *35th Annual IEEE Intl. Conf. on Comp. Comm. (INFOCOM 2016)*, San Francisco, CA, April 2016, pp. 1–9.

[4] T. G. Rodrigues *et al.*, "Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control," *IEEE Trans. on Computers*, vol. 66, no. 5, pp. 810–819, 2017.

[5] ——, "Towards a low-delay edge cloud computing through a combined communication and computation approach," in *IEEE 84th Vehicular Technology Conf. (VTC 2016)*, Sept 2016, pp. 1–5.

[6] Q. Fan and N. Ansari, "Green energy aware user association in heterogeneous networks," in *Proc. of IEEE Wireless Comm. & Netwk. Conf. (WCNC'2016)*, Doha, Qatar, Apr. 2016.

[7] N. L. Van Adrichem *et al.*, "Opennetmon: Network monitoring in openflow software-defined networks," in *2014 IEEE Network Ops. & Mgmt. Sym. (NOMS)*, Krakow, Poland, May 2014, pp. 1–8.

[8] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2009.