FBRC: Optimization of task scheduling in Fog-based Region and Cloud

Thanh Dat Dang
University of Technology Sydney, Australia
School of Computing and Communications
Thanh.D.Dang@student.uts.edu.au

Doan Hoang
University of Technology Sydney, Australia
School of Computing and Communications
Doan.Hoang@uts.edu.au

Abstract— **Fog computing preserves benefits of cloud computing and is strategically positioned to address effectively many local and performance issues because its resources and specific services are virtualized and located at the edge of the customer premises. Resource management is a critical issue affecting system performance significantly. Due to the complex distribution and high mobility of fog devices, computation resources still experience high latencies in fog's large coverage area. This paper considers a Fog-based Region and Cloud (FBRC) in which requests are locally handled not just by a region but multiple regions when additional resources are needed. An efficient task scheduling mechanism is thus essential to minimize the completion time of tasks and improve user experiences. To this end, two issues are investigated in the paper: 1) designing a fog-based region architecture to provide nearby computing resources; 2) investigating efficient scheduling algorithms to distribute tasks among regions and remote clouds. To deal with the complexity of scheduling tasks, a heuristic-based algorithm is proposed based on our formulation and validated by extensive simulations.**

Keywords—fog computing; task scheduling; fog resource; sensitive latency; region; fog cloud.

I. INTRODUCTION

Fog computing has become a new computing model in providing local computing resources and storage for end-users rather than cloud computing. It provides a popular platform to facilitate a wide range of applications such as smart transports, healthcare, smart grid applications. Users' applications and data, however, are increasing not only in number, volume, and variety but also in complexity with strict latency requirements. As a large number of physical devices move about in a large area, processing tasks may experience high latencies and jitters as needed computing resources may not be optimally distributed and are located far from their users. This paper introduces a local computing concept called "Region" to deal with these issues. A Region centres on a physical location which provides services for users within its coverage. It includes all fog devices such as high-end servers, smart phones, and vehicles connected to one another via wire or wireless connections in a defined geo-graphic location. Some fog devices, which may share computing resources in multiple regions, move to a new region but still request and/or provide computing services.

Fog computing has been recently introduced as "Cloud at edge network" to provide computing services, storage, and network services locally to its users. It bridges between users and clouds by providing single-hop wireless communications such as Wi-Fi, Bluetooth. Integrating resources at the edge of the network into computation groups enables fog to handle requests from nearby clients individually and efficiently. Nevertheless, heavy computation tasks still need to be processed at a remote cloud due to the limitation of fog's resources. Research on computation offloading has been explored on methods and in frameworks over the last few years [1-3]. These approaches showed that processing requests at local fog platforms results in faster response times in general compared to handling them at centralized clouds. Handling a large number of fog nodes, however, results in an excessive increase in transmission time for sending requests and receiving results. Other efforts focus on scheduling tasks for fog and cloud resources but they only considered costs and energy consumption [4-7]. Research on scheduling tasks on fog and cloud resources have not been well-established yet due to the lack of fog architecture that manages and allocates resources efficiently.

This paper proposes a concept of "Region" which provides computation resources to nearly clients in order to reduce data transmission times. We do not consider a region as a pre-established entity but as a dynamic region in terms of computation resources and locations relative to their clients. Furthermore, task processing involves not just a single region but multiple regions and/or possibly cloud servers in order to minimize the its completion time. The resources of a region are, however, limited and their availability varies. Some tasks may be processed on one region for faster response but others may have to be distributed and executed over multiple regions or even at remote cloud servers as they have more computational resources. Although more computation resources result in shorter processing time, data transmission between them and their users leads to higher latency. Thus, how to systematically manage the resources and schedule the tasks on a Fog-based region is still an open issue. An effective resource management and task scheduling mechanism is required in order to provide

high user experience, e.g., task completion time minimization. The following questions shall be answered to achieve minimum task completion time.

1) Where should the computation of required tasks take place, on regions or cloud servers? Generally, a cloud server has a larger amount of computation resources and faster processing time than regions. Nevertheless, it is located far from clients such that high latency occurs in task processing due to data transmission between clients at a region and cloud servers. In contrast, when a region is located near the clients, the transmission time is shortened, but low computation performance may result in an excessive increase the response time. If a task is not appropriately scheduled, the task completion time may be even longer when placed in a cloud server or a region. Efficiently scheduling tasks on regions and cloud server is thus a critical issue to reduce task completion time.

2) Which cloud servers or regions shall process the task when requests at original region are submitted? The tasks processing time on cloud servers and regions may be different from time to time due to workloads on them. It is desirable to allocate heavy tasks to light load cloud servers or regions depending on their conditions and locations to balance the loads and improve both computation and response times.

The main contributions of the paper are summarized as follows:

We propose the use of Region that addresses requirements of application's latency-sensitive in fog computing and in combination with cloud computing to provision computation resources on demand. To our best knowledge, we are the first investigate the task scheduling problem for Fog-based Region and Cloud (FBRC). In particular, we consider a scenario where computation can be processed either at local regions and/or remote cloud servers. By obtaining efficient task schedules at both regions and cloud servers, we choose to minimize the computation and transmission latency of all requests. We formulate the task scheduling problem for the FBRC as an Integer Program (called as FBRC-IP). The optimal scheduling problem is NP-hard problem. Thus, we design an efficient heuristic algorithm to solve the FBRC-IP problem.

We conduct simulations to demonstrate that the FBRC can significantly complement the Cloud-based only option to optimally minimize response latency.

The remaining of the paper is organized as follows. Section II presents the overview of our proposed model. Section III presents the formulation of the scheduling problem. Section IV shows numeric results for the proposed model. Section V discusses related work. The conclusion is drawn in Section VI.

II. PRELIMINARIES

A. System model

The logical view of FBRC is shown in Figure 1, where there is a set *R* of fog-based regions, a set *C* of cloud servers. All these nodes are inter-connected. Clients at the original region can request computation resources from other regions placed around it or cloud servers. Thus, tasks can be scheduled to minimize the completion time either at regions or cloud servers. In each region, a fog node is selected to manage the region. This fog node handles tasks submitted .in the region such as a join/leave requests, receiving and submitting computation requests, scheduling tasks to appropriate computation nodes. Hence, these fog nodes also collect and update throughputs and resources availability of closest regions and cloud servers. In our model, regions and cloud servers are required to periodically notify each other the average throughput and the amount of data that they can sustain.

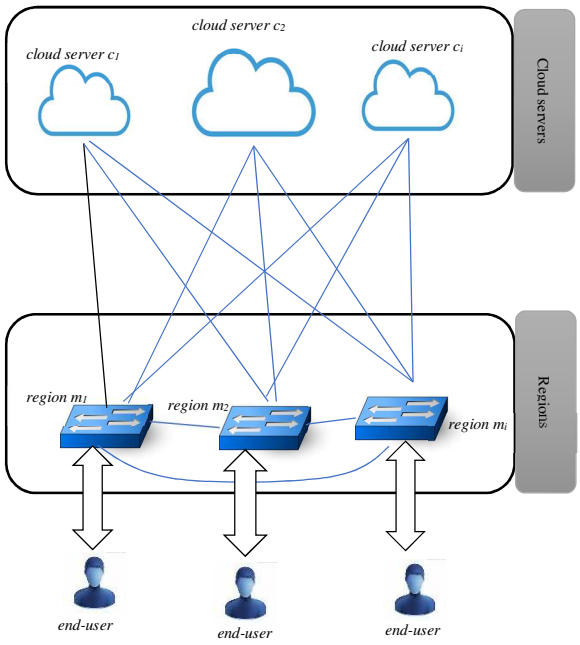

Fig. 1. The Logic view of FBRC

B. Fog-based Region scenario

Figure 2 presents a fog-based region scenario. In this design, a region can be structured by one or several fog nodes. A fog node consists of several fog devices with weak performance which are deployed at edge network. It can provide computation, network resources and storages. The fog devices are heterogeneous ranging from high-end servers to end devices such as mobile devices, wearable devices. For example, Region 1 is structured by fog node 1 and fog node 2 while Region 2 is formed by fog node 3 and fog node 4.

Fog election: It is essential to delegate a fog node to manage a region's activities and computing resources due to frequent join and leave node requests. Furthermore, task executions with a region need to be secure to protect sensitive data. We use a decentralized method [8] to select the delegated fog node in the region. Each fog node sends its vote for other fog nodes. In turn, it receives votes from other fog nodes. Thus, the votes in the region are collected into high capacity nodes among which the delegated node may be selected. A heartbeat is sent by every fog node to other fog nodes in a region periodically, at a heartbeat interval. Heartbeats are used by a fog node as a means to inform all fog nodes it is alive. A delegated fog also sends all fog nodes in its region every time the region changes by the detection of

an event, which is either a new fog that entered the region or one that left or crashed.

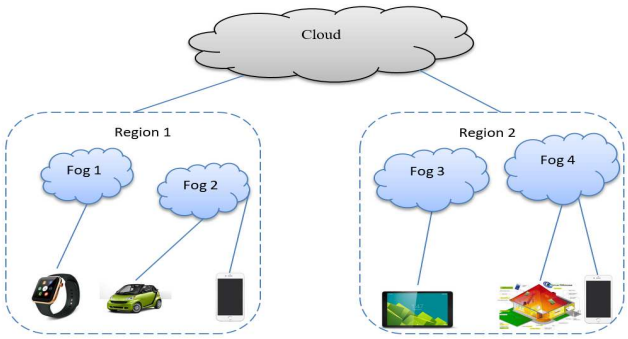

Fig. 2. The overview of Region-based Trust-Aware scenario

C. Problem statement

The transmission latency between any two nodes, e.g., is denoted as l_{ij} and other major symbols are summarized in Table I.

TABLE I. NOTATIONS

Basic Notation	Description
I	A set of fog nodes in the original region
T	A set of tasks
R	A set of regions
C	A set of cloud servers
t	A pending task in T.
R'	A set of fog nodes in other regions ($R \backslash I$).
γ_{om}	Transmission latency between the original region and other region $m \in R'$
γ_{oc}	Transmission latency between the original region and cloud server $c \in C$
η_c	Service rate for cloud server $c \in C$
μ_r	Service rate for other region $m \in R'$
st	Average request rate of task t from client i
x_{tic}	The $\{0,1\}$ variable indicates whether the cloud server c is selected.
x_{tim}	The $\{0,1\}$ variable indicates whether another region m is selected.
x_{tio}	The $\{0,1\}$ variable indicates whether the original region is selected.

We consider applications in fog computing consisting of a set of T tasks that need to be completed in a required time. Tasks can be run in parallel. Let R and C be the set of computation resources of regions and cloud servers for the applications. The computation resources in regions and cloud have different capacities and characteristics. Generally, cloud servers own more computation resources and process tasks faster compared to those of regions, but they are often located far away from their clients. In other words, these servers can provide faster computation but have higher latency because of the long transfer time between clients and servers. In contrast, the computational results may be transferred quickly from a region to its surrounding clients but the processing time to complete a client's request may be very large because of limited computing resources. For these reasons, FBRC effectively deploys resources of multiple regions as well as cloud servers to reduce task completion time. We assume that the processing time of tasks and the number of hops for transferring data between regions and cloud servers are known. In doing so, we could pre-compute the processing and transmission time of tasks along with the amount of data throughput.

Task processing requests are submitted randomly at each fog node. For each task t in a set T of tasks, the average task arrival to a fog node $i \in R$ is τ_{ti}. Without loss of generality, we assume that a region $r \in R$ has the computational capacity μ_r for FBRC and a cloud server $c \in C$ has computation rate η_c. Let s_t denote the size of task $t \in T$.

In the FBRC, a task can be processed by the fog nodes in a region itself, or in multiple regions, and/or in cloud servers. Although static scheduling may be feasible for task processing locally, limited resources prevent handling all tasks locally in a region. In fact, it is difficult to estimate accurately computational requirements of task requests. Alternatively, more flexibility and higher efficiency could be obtained if the task scheduling process can choose stochastic strategies based on the distributions of submitted tasks and their requirements. We denote p_{tim} as the probability that a task $t \in T$ submitted from a fog node $i \in I$ in the original region to other regions R' where $R' \subseteq R$. Let p_{tic} be the probability that a task t request submitted from the original region is sent to cloud node $c \in C$ for handling disk reading and task processing. Let p_{tio} be the probability that task $t \in T$ request is processed at the original region itself.

A task may be processed at different regions or cloud servers other than always in a particular region due to the availability of computing resources. Usually, cloud servers have higher service rates than fog nodes in regions since they are shared by multiple clients for many tasks. How to balance the requests among the original region, other regions, and cloud servers is a critical issue to task completion time. In addition, the transmission latency is also another critical issue because tasks may be processed fasters at cloud servers or multiple regions that have available resources but transferring data among nodes results in a high delay in the whole process. Clearly, the probability distribution of submitted requests and strategies of the scheduling process play big roles in minimizing task completion times.

III. PROBLEM FORMULATION

In this section, we provide a formal description of our problem with consideration of task scheduling by formulating it into an Integer program problem.

A. Task completion time analysis

1) Computation time

The computation time of a task depends on where the processing is scheduled. If t is distributed on cloud server $c \in C$ with the service rate η_c, the server c may be shared by multiple clients for different tasks. The overall task arrival rate at cloud server c thus can be calculated as

$$\theta_c = \sum_{t\in T}\sum_{i\in I} p_{tic}\,\tau_{ti}, \forall c \in C \quad (1)$$

Recall that the task computation time is exponentially distributed on a cloud server, which is based on an M/M/1 queue. The average computation time of all tasks at cloud server $c \in C$ can be calculated as

$$\sigma_c^{tc} = \frac{1}{\eta_C - \sum_{t\in T}\sum_{i\in I} p_{tic}\tau_{ti}}, \forall c \in C, \quad (2)$$

Where we must ensure that

$$\eta_C > \sum_{t\in T}\sum_{i\in I} p_{tic}\,\tau_{ti}, \forall c \in C \quad (3)$$

If t is sent to another region $m \in R'$ with the service rate μ_r, the region m may be requested from clients for different tasks. Therefore, the overall task arrival rate at region m can be calculated as

$$\theta_m = \sum_{t\in T}\sum_{i\in I} p_{tim}\,\tau_{ti}, \forall m \in R' \quad (4)$$

Similarly, the computation on another region is also based on an M/M/1 queue. It can be also calculated as

$$\sigma_m^{tc} = \frac{1}{\mu_r - \sum_{t\in T}\sum_{i\in I} p_{tim}\tau_{ti}}, \forall m \in R'. \quad (5)$$

We shall also guaranty that

$$\mu_r > \sum_{t\in T} p_{tim}\,\tau_{ti} = 1, \forall r \in R' \quad (6)$$

Finally, the computation is processed at the original region. The overall task arrival rate at the original region can be calculated as follows.

$$\theta_0 = \sum_{t\in T}\sum_{i\in I} p_{tio}\,\tau_{ti}, \forall i \in I. \quad (7)$$

We can derive the average computation time on the fog node $i \in I$ at the original region as

$$\sigma_o^{tc} = \frac{1}{\mu_o - \sum_{t\in T}\sum_{i\in I} p_{tio}\tau_{ti}}, \forall i \in I. \quad (8)$$

We shall also guaranty that

$$\mu_i > \sum_{t\in T} p_{tio}\,\tau_{ti} = 1, \forall i \in I. \quad (9)$$

2) Transmission time

If the computation tasks and data retrieval at the original region i are handled by another region m and cloud server c, respectively, the transmission latency between i, m and c shall be considered. We use binary variables to indicate other region and cloud server selection as

$$x_{tic} = \begin{cases} 1, \text{the task } t \text{ from the original region } i \text{ is handled by cloud server } c, \\ 0, \text{otherwise} \end{cases} \quad (10)$$

Similarly, we define

$$x_{tim} = \begin{cases} 1, \text{the task } t \text{ from the original region } i \text{ is handled by region } m, \\ 0, \text{otherwise} \end{cases} \quad (11)$$

We also define

$$x_{tio} = \begin{cases} 1, \text{the task } t \text{ is handled by original region,} \\ 0, \text{otherwise} \end{cases} \quad (12)$$

Hence, we the relationship between the probabilities and decision variables as follows: 1) when $p_{tim} > 0$, the value of x_{tim} shall be 1, indicating that the region m is selected; 2) when $p_{tic} > 0$, the value of x_{tic} shall be 1, indicating that cloud server c is selected; 3) when $p_{tio} > 0$, the value of x_{tio} shall be 1, indicating that the original region is selected. Therefore, we have following relationships

$$p_{tim} < x_{tim} < A_{ptim}, \ \forall t \in T, \forall i \in I, \forall m \in R' \quad (13)$$

and

$$p_{tic} < x_{tic} < A_{ptic}, \ \forall t \in T, \forall i \in I, \forall c \in C \quad (14)$$

and

$$p_{tio} < x_{tio} < A_{ptio}, \ \forall t \in T, \forall i \in I \quad (15)$$

where A is an arbitrarily large number.

For tasks scheduled onto cloud servers and other regions, all the transmissions for data retrieval process happened between the original region i and $m \in R'$, $c \in C$. Let n_{tm} and n_{tc} are the average data retrieval during task execution from the original region i to another region m and cloud c, respectively. The expected transmission time of task t from the original region i allocated to m and c can be calculated as

$$\sigma_{tim}^{tt} = 2n_{tm} + \gamma_{om}, \forall t \in T, \forall i \in I, \forall m \in R'. \quad (16)$$

and the expected transmission time of task t between the original region i and a cloud server c can be calculated as

$$\sigma_{tic}^{tt} = 2n_{tc} + \gamma_{oc}, \forall t \in T, \forall i \in I, \forall c \in C. \quad (17)$$

3) Task completeness constraints

To ensure the Quality of service (QoS), it is required all requests submitted to the original region must be processed, either at regions or cloud servers. This results in

$$\sum_{m\in R'} p_{tim} + \sum_{i\in I} p_{tio} + \sum_{c\in C} p_{tic} = 1, \forall t \in T, \forall i \in I. \quad (18)$$

Tasks need to be completed without exceeding the deadline. This leads to

$$\sum_{c\in C} x_{tic}\,.\,(\sigma_m^{tc} + \sigma_{tim}^{tt}) + \sum_{m\in R'} x_{tim}\,.\,(\sigma_c^{tc} + \sigma_{tic}^{tt}) + \sum_{i\in I} x_{tio}\,.\,\sigma_o^{tc} \le D, \forall t \in T, \forall i \in I. \quad (19)$$

B. An FBRC-IP formulation

Multiple tasks are submitted to fog nodes at an original region. These tasks will be allocated to appropriated fog nodes at the current region, other regions and cloud servers based on their requirements. If we allocate a task to high performance a cloud server which is located far from the client, this task may not be finalized in the expected time due to large transmission time. The aim of the FBRC is to minimize the maximum average task completion times. Let φ be the maximum time is introduced in completing the task t. Thus, we have

$$x_{tic}\,.\,(\sigma_m^{tc} + \sigma_{tim}^{tt}) < \varphi \quad (20)$$

and

$$x_{tim}\,.\,(\sigma_c^{tc} + \sigma_{tic}^{tt}) < \varphi, \quad (21)$$

and

$$x_{tio}\,.\,\sigma_o^{tc} < \varphi. \quad (22)$$

The problem is solved by minimizing φ. In short, we can formulate the task maximization completion time-minimization with consideration of task scheduling as an Integer Programming problem (Called FBRC-IP), as follow:

FBRC-IP:

$minimize\ \varphi,$

subject to the following constraints:

- *Service rate as (3), (6), (9)*
- *Computation resources as (13-17)*
- *Task completeness as (18), (19)*
- *Maximum completion time (20-22)*

The objective function is to minimize the task completion time when executing requests at regions and cloud servers.

C. *Algorithm design*

In this section, we present the design algorithm to find an optimal resource scheduling algorithm for FBRC that minimizes the completion time. The steps of the strategy are given as follows:

1. Requests are sorted in ascending order of latency-constrains.

2. Computation resources are allocated according to the policy that aims to minimize the computation latency for each request. This latency can be expressed as the ratio of the computation throughput and the latency requirements.

3. Pending requests are sorted in ascending order of latency-constrains.

4. Regions and cloud servers are allocated with the objective of minimizing the overall FRBC latency.

IV. NUMERICAL RESULTS

Simulation results are presented in this section to validate the task completion time by scheduling tasks to multiple regions and cloud servers. Without loss of generality, the regions and cloud servers throughout are assumed known. To evaluate the efficiency of our proposed scheduling scheme, we simulate requests, system capabilities, and scheduling strategies strictly following the system model defined in section II. Especially, in order to show the advantage of our proposed task scheduling scheme, we introduce two competitors namely Cloud-based ("Cloud") and Region-based ("Region") task scheduling schemes. The former schedules all computation tasks onto cloud servers until all the tasks are allocated or Cloud servers are fully loaded, while the latter handles all tasks on all regions until all of them are allocated or regions are fully loaded.

We select the parameter settings for the simulation as follows: each cloud server is with a total computation rate of *30* which the computation on each region is set as *10*. The current resources of regions and cloud servers are set randomly in the range of [*0.7,1*] as they are shared by tasks. The transmission latency among regions are randomly set in the range of [*0.01, 0.09*] while the transmission latency between a region and a cloud server are randomly set in the range of [*0.4,0.7*]. We investigate how FBRC performs over a range of parameters..

A. *On effects of task arrival rate*

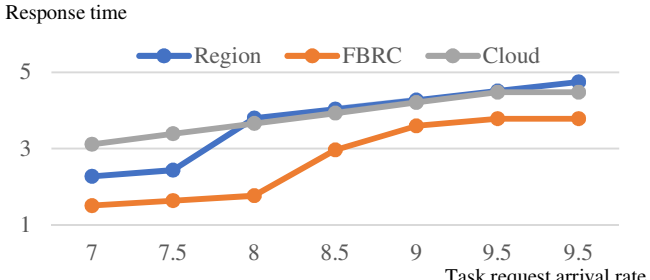

Fig. 3. Task arrival rate

We first compare the task completion time of Region, FBRC and Cloud under different task arrival rates from *7* to *10* (Figure 3). When task arrival rate increases, more regions will involve in the process to perform requests. This is because the longer queue delay leads to larger computation time. However, the benefit of our proposed scheme over "Region" and "Cloud" can be observed when task arrival rate increases. Thus, it provides the flexibility in selecting computation resources between regions and Cloud servers.

B. *On effects of computation service rate*

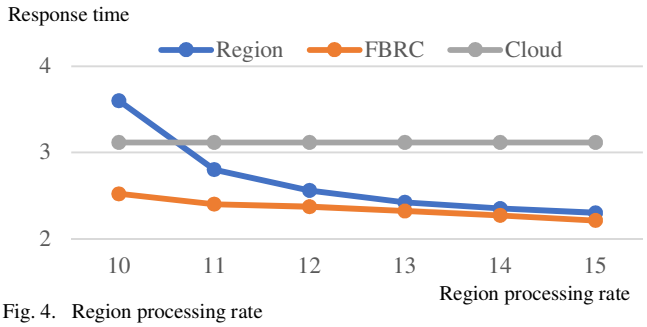

Fig. 4. Region processing rate

Figure 4 and figure 5 present the task completion time of region service rate and cloud server rate. To observe the effect on region processing rate, we increase the service rates from *10* to *15*. It can be seen from the figure 4 that the completion time of Region and FBRC shows a decreasing trend of region service rate. When the service rates increase, regions process more requests to provide with a faster response. This results in the shorter computation time. The increase of service rate, in fact, leads to a significant impact on the computation time of Region. Hence, for Region, the completion time decreases significantly. For FBRC, requests shall be processed by computation resources to obtain a faster response. Thus, more resource of regions will be used in processing requests to reduce completion time. For Cloud, as all requests are handled at cloud servers, there are no benefits from increasing the region service rate.

The similar trend is also presented in figure 5. When computation service rate on cloud servers is low, FBRC's response latency is *2.82* while that of Cloud is *3.73* at service rate *18*. It can be explained that FBRC assigns more task to regions. However, when the service rate of cloud servers increase, the difference of response latency between FBRC and Cloud becomes small. For example, when service rate is *36*, the gap decreases to *0.11*. Overall, FBRC can always schedule resource optimally to obtain the low response latency.

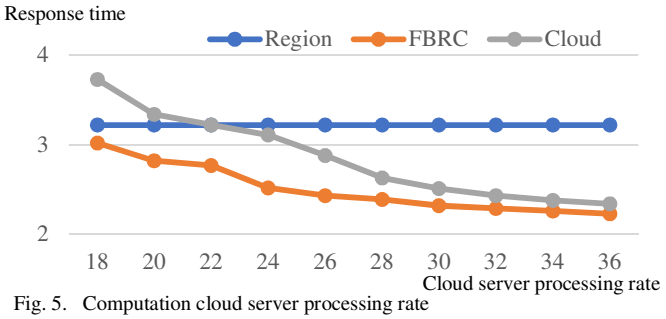

Fig. 5. Computation cloud server processing rate

V. Related work

Hassan [1] proposed an offloading mechanism using fog computing to provide nearby computing resources for mobile applications and mobile storage. It processed mobile computation tasks faster with including mobile devices and servers. In fact, mobile storage was also expanded by leveraging clients' devices nearby-access. Pu [4] proposed Device-to-Device Fogging framework that schedules mobile tasks to available resources of mobile devices. An online computationally-efficient algorithm for mobile task offloading was also introduced that minimizes the energy consumption for task execution of all users. A task scheduling approach in a cloud-fog computing system was presented in [5]. It used virtual machines at cloud as extended resources when fog nodes did not have sufficient resource to fulfill clients' requests. However, it did not consider the locality of resources nearby-access. The optimization approach focused on cost for cloud services instead of latency-sensitive responses to clients. Furthermore, the costs for fog resources should have been compared to cloud resources as these resources were also shared by multiple clients. Zeng [2] proposed a joint optimization of task scheduling and image placement in fog computing supported software-defined network embedded system. Computational resources were provided from two sources: embedded clients and fog nodes represented by computation servers. Storage servers could be shared by both clients and computation servers. This approach only structured resources locally from fog devices and embedded clients. It was, however, unavoidable to employ more resources from cloud since there could be large-scale tasks or intensive requests submitted from clients. Cardellini [9] introduced a distributed and self-adaptive QoS-aware scheduler for the extended Storm which is an open source Data Streaming Processing (DSP) system. The schedulers ran at local clusters to schedule resources for clients. The Storm was added modules that could scale the number of DSP application and network resources over fog infrastructure. Furthermore, the self-adaptive scheduler enabled reconfiguring the operator placement automatically when there were changes in fog environment. Oueis [3] proposed a low complexity small cell clusters establishment and resources management customizable algorithm to address the load balancing in fog computing. The proposed mechanism allowed small cell (SCC) to minimize the computation resource and power consumption while still satisfying users' requests. It first put the resources at small cells (SCs) to perform the requests using scheduling rules. Computation clusters were then structured to serve unsatisfied requests. Wang [10] proposed CloudFog which was a lightweight system and allocated nearby users to provide computation resources. Fog nodes played roles of supernodes to render video games and stream them. Cloud servers served heavy computation tasks and updated computation results to supernodes. In addition, the Receiver-driven encoding rate adaptation strategy and Deadline-driven sender buffer scheduling were also introduced to enhance the reliability and the latency requirement of the proposed system. Our work employs computing resources from both regions and clouds to handle requests with sensitive latency demands. In fact, these resources are scheduled optimally to allocate to each request and thus provide a better performance trade-off compared to use fog resources or cloud resources alone.

VI. Conclusion

This paper introduced a new concept of "Region" in fog computing in providing nearby access for clients. A task scheduling for region-based cloud algorithm was proposed to satisfy resource and sensitive latency requirements and yet utilize appropriate cloud resources for heavy computation tasks. The scheduling problem was formulated as an Integer program and solved by a heuristic algorithm. The numeric results demonstrated the efficiency of the proposed model in term of latency response and resource utilization compared to Region-based and Cloud-based resource managements.

References

[1] M. A. Hassan, M. Xiao, Q. Wei, and S. Chen, "Help your mobile applications with fog computing," in *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking - Workshops (SECON Workshops)*, 2015, pp. 1-6.

[2] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, "Joint Optimization of Task Scheduling and Image Placement in Fog Computing Supported Software-Defined Embedded System," *IEEE Transactions on Computers,* vol. 65, pp. 3702-3712, 2016.

[3] J. Oueis, E. C. Strinati, S. Sardellitti, and S. Barbarossa, "Small Cell Clustering for Efficient Distributed Fog Computing: A Multi-User Case," in *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, 2015, pp. 1-5.

[4] L. Pu, X. Chen, J. Xu, and X. Fu, "D2D Fogging: An Energy-Efficient and Incentive-Aware Task Offloading Framework via Network-Assisted D2D Collaboration," *IEEE Journal on Selected Areas in Communications,* vol. PP, pp. 1-1, 2016.

[5] P. Xuan-Qui and H. Eui-Nam, "Towards task scheduling in a cloud-fog computing system," in *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2016, pp. 1-4.

[6] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog Computing May Help to Save Energy in Cloud Computing," *IEEE Journal on Selected Areas in Communications,* vol. 34, pp. 1728-1739, 2016.

[7] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal Workload Allocation in Fog-Cloud Computing Towards Balanced Delay and Power Consumption," *IEEE Internet of Things Journal,* vol. PP, pp. 1-1, 2016.

[8] G. Liu, H. Shen, and L. Ward, "An Efficient and Trustworthy P2P and Social Network Integrated File Sharing System," *IEEE Transactions on Computers,* vol. 64, pp. 54-70, 2015.

[9] V. Cardellini, V. Grassi, F. L. Presti, and M. Nardelli, "Distributed QoS-aware scheduling in storm," presented at the Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems, Oslo, Norway, 2015.

[10] Y. Wang and W. Shi, "Budget-Driven Scheduling Algorithms for Batches of MapReduce Jobs in Heterogeneous Clouds," *IEEE Transactions on Cloud Computing,* vol. 2, pp. 306-319, 2014.