A Novel Forwarding Policy under Cloud Radio Access Network with Mobile Edge Computing Architecture

Dian-Yu Lin
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
r04921057@ntu.edu.tw

Yung-Lin Hsu
Graduate Institute of
Communication Engineering
National Taiwan University
Taipei, Taiwan
d04942010@ntu.edu.tw

Hung-Yu Wei
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
hywei@ntu.edu.tw

***Abstract*—Nowadays, dozens of low-latency required application are emerging, traditional mobile network architecture would not be able to support such applications anymore in the future. Cloud radio access network (C-RAN) combined with Multi-access/mobile edge computing (MEC) seems to be one of the most feasible new RAN architectures to fulfill the requirement. With the assistance of MEC, the computing resource could be allocated more efficiently. In this paper, firstly the advantage of generalized-processor-sharing model (GPS) compared with first-in-first-out (FIFO) and processor-sharing (PS) are discussed in order to figure out the practical queueing behavior in MEC system. Next, the relationship between theoretical traffic intensity factor and realistic system CPU utilization condition is correlated. Finally, based on the discussion, a two threshold forwarding policy (TTFP) algorithm is proposed to dynamically arrange the data traffic according to current system traffic states. The result of the simulation articulates that TTFP algorithm could efficiently fulfill the applications who requires low entire waiting time as possible in high intensity traffic condition.**

***Index Terms*—C-RAN, Multi-access/mobile edge computing (MEC), generalized-processor-sharing model (GPS), data forwarding policy**

I. Introduction

The proliferation of mobile data traffic especially video and voice streaming has been anticipated to be dramatic and unprecedented in the future. To accommodate such large traffic loads, deploying much more small cells seems to be a intuitive way to increase the system capacity [1], [2]. However, such implementation may increase nor only capital expenditure but also operating expense to operators. How to reduce the network deploying and operating costs when meeting the data traffic demands is a critical issue. In addition, new types of serving application might be introduced in the next generation mobile networks. Researches and literatures already exposed the end-to-end latency demand of some mission critical applications is limited to be within a few milliseconds [3]. Ultra-reliable low latency communication (URLLC) defined in 3GPP is an iconic example for such applications. That is to say, latency is one of the critical attributes which should be carefully considered in the future mobile network architectures [4].

The cloud radio access network(C-RAN) is composed of a centralized baseband unit(BBU) pool and remote radio heads (RRHs) [5]. In order to cater trementous data traffic and critical requirements such as URLLC in next generation communication systems, the system converging Multi-access/mobile edge computing(MEC), fog computing and C-RAN has been considered[6]. Fig. 1 illustrates a paradigm architecture combining of MEC system into C-RAN. In this figure, the MEC service entities are located on the original data path between end devices and cloud computing data centers. In such framework, instead of original basestation RRH is used to be the transceiver. As the figure shown, to improve system scalability and flexibility, plenty of RRHs need to be deployed close to user equipments(UEs). The MEC platform equipped with several MEC entities, each MEC entity may enable to administer the BBU pool, based on the deploying strategy. The BBU pool is a set of computing entities, which is able to perform baseband processing and/or data computing. The function of baseband processing is that the system would decrypt a packet first before computing, some information would be revealed in this stage, such as application intention, routing path and priority, etc. This kinds of MEC paradigm and function are introduced in [7], [8]. With C-RAN and MEC architecture, fulfilling the low-latency requirement of URLLC applications seems to be feasible in the next generation mobile network.

Comparing to conventional RANs, C-RAN possesses more data computational advantages such as flexibility and reliability. Also, [9] introduced an epochal concept, which is device-centric architecture. Combining with the Software Defined Radio (SDR), the radio system can be able to support different kinds of application requirements [10], [11]. Moreover, centralized baseband processing could make the utilization of computing resource more efficient and further reduce the computational power consumption. [12], [13]. By using the reconfigurable fronthaul [14], [15], the connections between

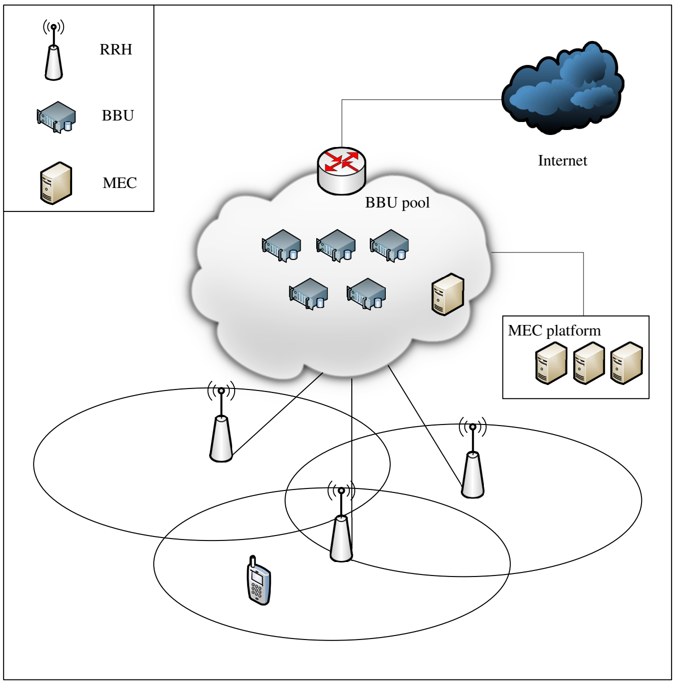


Fig. 1: C-RAN and MEC system combination architecture

BBUs and RRHs could be changed accordingly and dynamically to meet the varying traffic loads [16]. Therefore, C-RAN is known as a soft and green architecture in 5G mobile network [17]. To decrease the data processing latency, performing handover mechanism in C-RAN BBU pool is inevitable, and the implementation of BBU pool is based on the performing policy in general purpose processor (GPP) [18], [19]. GPP platform is capable to run some particular applications, which means that there are some additional computing resources in GPP platform.

There are already many attentions on the emerging low-latency applications in future mobile networks. In order to cater the requirement, distributing a part of the computing services from cloud computing center to network edge is a potential solution [20], [21]. This kind of deployment is known as fog computing [22], [23], [24] or MEC [25], [26]. [27] proposed a system which makes the nearby unused mobile devices to do the computation at the network edge or fog nodes. Roughly speaking, MEC entities would be deployed by operators, whereas fog nodes are mostly to be privacy belonging.

The MEC platform is considered to play an important role in the future mobile networks. Because the transmission delay from end devices to the MEC platform is much shorter than end devices to cloud data centers. However, there are some challenges in MEC system need to be conquered. One of the most critical challenge is that the resources on the MEC platform may not be enough if the data traffic goes heavy. When the end devices forwarding a large amount of data traffic to the MEC platform, the packets might experience larger processing and queueing delay, which may fail the low-latency requirement. Thus, designing a data forwarding policy to optimize and reconcile the volume of the traffic computed in the MEC node is necessary. In other words, allocating the computing resources between the cloud side baseband processing and the MEC node is an important topic.

To meet the low-latency requirement in MEC platform, the appropriate amount of traffic forwarded to the MEC platform need to be determined accordingly. However, quantizing the appropriate forwarding data in MEC is not easy. Since that there might be some invisible or unpredictable constraints in practical MEC equipment. Thanks to the MEC platform and BBU pool is implemented according to the deploying strategy in GPP, analyzing the GPP processing behaviors could help the MEC system make a decent forwarding decision. There are several mathematical queueing models could be used to compare with the GPP processing behavior, such as first-in-first-out (FIFO) [28] and processor-sharing (PS) [29]. Nevertheless, FIFO and PS are not that suitable to represent the practical data traffic behavior, the reason will be discussed in the next section. Hence, generalized-processor-sharing (GPS) model is adopted to mimic the practical data traffic and compared to GPP processing behaviors. In this paper, by comparing the mathematical results to the simulation and real GPP platform results, the relationship between theoretical system service rate and practical system traffic intensity value will be revealed. Besides, according to the relationship, a two threshold forwarding policy algorithm is proposed, which enables the MEC GPP platform to dynamically arrange the data path according to the current traffic.

To implement C-RAN architecture, [18] proposed a GPP based C-RAN architecture. The main concept is to efficiently reduce power consumption by allocating computing resources according to the traffic. In [30], the authors considered both the computing resources for C-RAN and MEC application as an integrated computing resource pool. Furthermore, the authors provided some discussion about the CPU load measurement on the testbed. Through the observation of the CPU load, it would be possible to predict the remaining computing resources for the MEC application service.

The rest of this paper is in the following. In Section II, the defect of FIFO and PS are going to be discussed. Section III explains the reason why GPS is adopted and gives some derivation consequences. Section IV provides the observation results in the testbed and the comparison the processing behaviors between testbed PSS and GPS model. In Section V, according to the observation and comparison results appears in Section IV, a two threshold forwarding policy algorithm is proposed. The simulations of algorithm are also presented in this section. Finally, here comes the conclusion in Section VI.

II. Related works

In order to figure out the GPP processing behaviors, using queueing theorem and model to approximate the processing appearances seems to be feasible. There are several candidate queueing models, such as first-in-first-out (FIFO) [28], processor-sharing (PS) [29], [28] and generalized-processor-sharing (GPS) [28]. In GPP, there might be multiple applications running on the MEC platform simultaneously. Now, consider a simple case, the MEC platform only possesses a virtual machine (VM) with a single-core processor. In order to meet the low-latency requirement, the packet waiting time (queueing time plus processing time) in the MEC platform is the main observational factor. The general traffic model will be introduced in the following contents. With the traffic model, the waiting time of FIFO and PS are going to follow out.

A. *Traffic Model*

Assume that there are totally N applications running on the single-core processor, the packet arrival pattern of each application is set to be a Poisson distribution, and the average arrival rates are denoted as $\lambda_1, \lambda_2, ..., \lambda_N$, respectively. Moreover, according to the superposition characteristic of Poisson distribution [28], the overall sum of each packet arrival rate λ is

$$\lambda = \lambda_1 + \lambda_2 ... + \lambda_N \tag{1}$$

B. *First In First Out*

Fig. 2a illustrates FIFO queueing model. In this queueing model, the processor always deals with the packet according to the arrival time order. If a packet comes before the end of the previous packet processing time, this packet will go to the tail of the queue. If the system queue capacity is infinite, the FIFO system can be viewed as an M/M/1 queueing system.

By using the theory of continuous-time Markov chains with exponential distribution, the mean waiting time of M/M/1 can be calculated, and the result is shown as eq.(2) . μ represents for the mean service rate of processor when dealing with only one packet. Since that the mathematical formula could be easily obtained [28], the derived process is skipped here.

$$W_{FIFO} = \frac{\rho}{\lambda(1-\rho)} \tag{2}$$

Where the traffic intensity index $\rho = \frac{\lambda}{\mu}$.

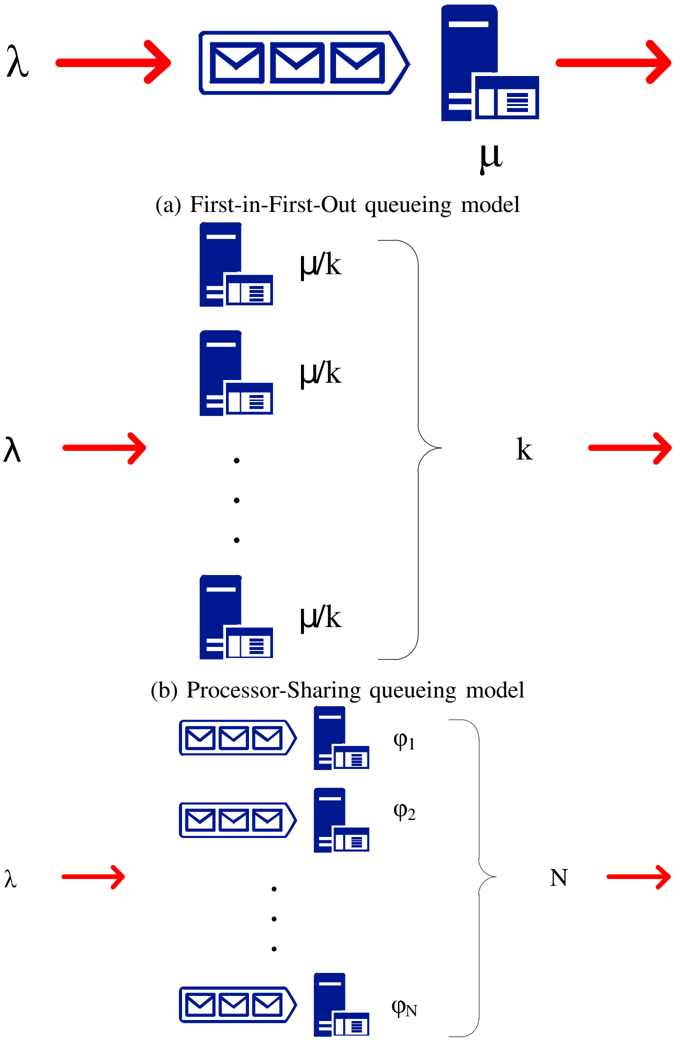
C. *Processor Sharing*

In Fig. 2b, the mechanism of the processor-sharing (PS) model has been derived. In the PS model, once arrived, all the packets are serviced concurrently regardless of application types and sources. Which implies that there is no queueing time in PS system. Therefore, the mean service rate grows proportionally depends on the number of packets arrived in the system.

As same as the assumption in FIFO system, the mean service rate for a packet being served alone is μ. Thus, when the processor deals with k packets, the service rate goes to $\frac{\mu}{k}$. [29] shows that the mean waiting time spent in the PS system.

$$W_{PS} = \frac{t}{1-\rho} \tag{3}$$

Where t is a packet requiring serving time, $t \geq \frac{\rho}{\lambda}$.



(a) First-in-First-Out queueing model

(b) Processor-Sharing queueing model

(c) Generalized-Processor-Sharing Model

Fig. 2: FIFO, PS and GPS queueing model

However, here are some drawbacks in FIFO and PS model when modeling the practical data traffic behaviors. In general GPP platform, a single-core processor would distribute the computing resources according to the incoming application number through time slicing. The defect of the FIFO queue is that it ignore multitasking factor, all the packets need to wait in the queue line no matter the application type. And the flaw of PS model is just at the opposite side, it shares the computing resource only according to the number of incoming packet, a single application could occupied almost the entire resource if its packet arrival rate is much greater than others. Hence, considering the drawbacks of FIFO and PS model, generalized-processor-sharing (GPS) model seems to be more rational to be the real-contrast model.

III. Generalized-Processor-Sharing Model Adoption

In order to be free from the flaw of FIFO and PS model, GPS is adopted in this paper. In Fig. 2c, as the same, GPS server service rate is set to be μ, and N applications are treated in the server. Differ from FIFO and PS, the computing resources are shared based on the number of running application, each application has its own queue for the packets. So, GPS is more fair than FIFO and PS in terms of computing resource allocation. In addition, the resource allocation is not necessarily to be even. According to some specific rule such as application priority or allocation policy, each application may enjoy different portion of resource.

In order to make the analysis easier, the GPS model would allocate the computing resource to the application evenly here. ϕ_i represents the allocation weighting index of application i. If application i has at least one packet at the processor, $\phi_i = 1$, otherwise $\phi_i = 0$. μ_i is the partial computing resource allocated to application i. The individual resource allocation formula can be formed as

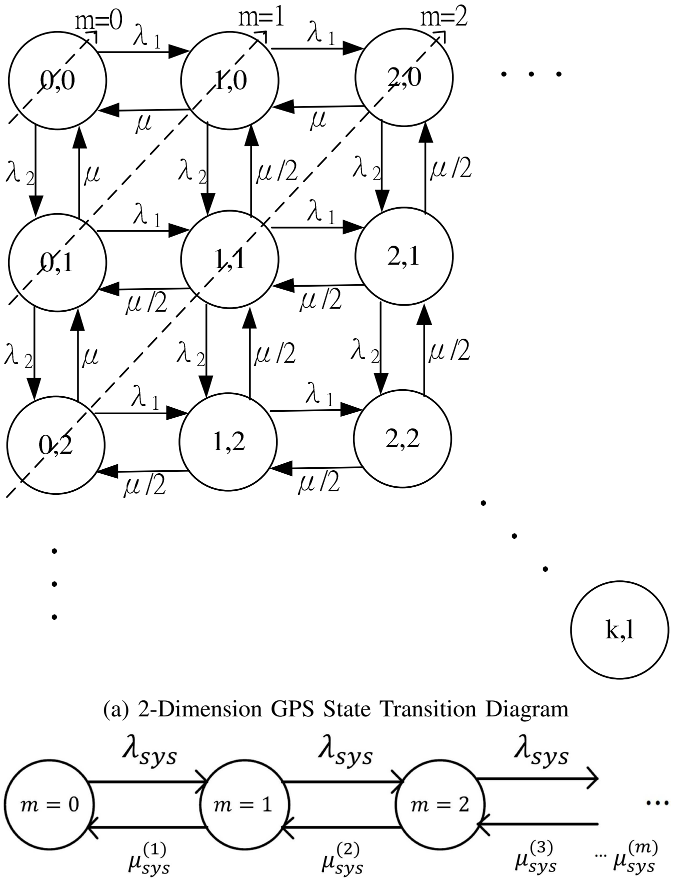
$$\mu_i = \frac{\phi_i}{\sum_{j=1}^{N} \phi_j}\mu,\ i = 1, ..., N \tag{4}$$

Based on eq.(4), it is easy to observe that the maximum value of μ_i, says μ_{max}, is equal to μ, and the minimum value of μ_i, says μ_{min}, is $\frac{\mu}{N}$. with the following conditions, the upper bound and the lower bound waiting time could be derived respectively. Where $\rho_{lower} = \frac{\lambda_i}{\mu_{max}}$ and $\rho_{upper} = \frac{\lambda_i}{\mu_{min}}$.

$$\frac{\rho_{lower}}{\lambda_i(1-\rho_{lower})} \leq W_{GPS,i} \leq \frac{\rho_{upper}}{\lambda_i(1-\rho_{upper})} \tag{5}$$

Now, consider a processing scenario in GPS system. In this scenario, a new incoming packet arrives when a previous packet is still under processing in the processor, there are two possible cases may happen. Case one, the new incoming packet and the previous packet belong to the same application. In this case the incoming packet is put in the queue line owned to the application. The other case is the incoming packet belongs to another application. In this case the processor will share partial of the computing resource to handle the incoming packet in parallel. In consequences, the serving time of the previous packet is stretched proportionally since that the computing resource is shared. For the sake of discussion, here comes a simple two-application existing scenario. The packet arrival state in the GPS system can be interpreted as two parameters. Fig. 3a gives the state transition diagram. The parameter at the left side is the number of arrival packet from the first application, says k, and the right-side parameter depicts the number of packets from the second application, says l.

The critical problem to simplify the flow balance equation is how to map 2-dimension states to 1-dimension states rationally. Recap the characteristic of superposition of Poisson distribution (eq.(1)), each application packet arrival rate could be summed up as λ_{sys} and regarded as the arrival rate of single application. Similarly, all the arrival packet number could also be summed up. Therefore, the overall system could be simplify as a 1-dimension state transition diagram, the result is shown as Fig. 3b. Parameter m is the sum of k and l. In this model, the number of the application and the individual arrival rate



(a) 2-Dimension GPS State Transition Diagram

(b) 1-Dimension GPS State Transition Diagram

Fig. 3: GPS state transition diagram systematization

(λ_i) are assumed to be known and fixed, thus the overall system arrival rate λ_{sys} is fixed as well. However, without any state probability information in Fig. 3a, it's almost impossible to get the exact values of system mean service rate $\mu_{sys}^{(m)}$ in Fig. 3b. Fortunately, the system mean waiting time could still be approximated once the overall system traffic intensity ρ_{sys} is obtained as eq.(6). The formula is shown as eq.(7).

$$\begin{aligned}\rho_{sys} = \sum_{i=1}^{N} \frac{\lambda_i}{N \cdot \mu_i} &= \frac{\sum_{i=1}^{N} \frac{1}{\kappa_i}(\prod_{j=1}^{N} \kappa_j)\lambda_i}{N \cdot (\prod_{j=1}^{N} \kappa_j)\mu}, \quad \kappa_i \neq 0 \\ &= \frac{1}{N \cdot \mu} \sum_{i=1}^{N} \frac{\lambda_i}{\kappa_i}, \quad \kappa_i \neq 0\end{aligned} \tag{6}$$

where κ_i is the allocated computing resource ratio of application i, $\kappa_i = \frac{\phi_i}{\sum_{j=1}^{N} \phi_j}$.

$$W_{GPS,sys} = \frac{\rho_{sys}}{\lambda_{sys}(1-\rho_{sys})} \tag{7}$$

It should be noticed that the system mean waiting time depicts the overall applications mean waiting time but not individual application. Obviously, more the number of application be processed in the system, longer the waiting time each application should take. The following are some validations of simulation and theoretical results.

A. GPS Processor under Fairly Distributed Traffic

The first simulation is under a fairly distributed traffic scenario, which means that the arrival rate of each application is equal. As shown in Fig. 4,the mean service rate of processor μ is set to be 50 (packets/s). The overall packet mean arrival rate λ is 40 (packets/s). For example, if there are 4 applications running on the processor simultaneously, the packet arrival rate becomes 10 (packets/s) to each application. Apparently, when the number of participating application increases, the curve of individual mean waiting time is almost fixed and approaches to the theoretical line. The reason is that when the number of participating application arises, even though the computing resource for each application is reduced, individual application packet mean arrival rate descends as well, which makes the mean waiting time is about to fixed. According to the result shows in Fig. 4, it could be asserted that the assumption of GPS model in the previous content seems to be rational at least in fairly distributed traffic scenario. In the next subsection, some testbed experiment results are considered jointly.

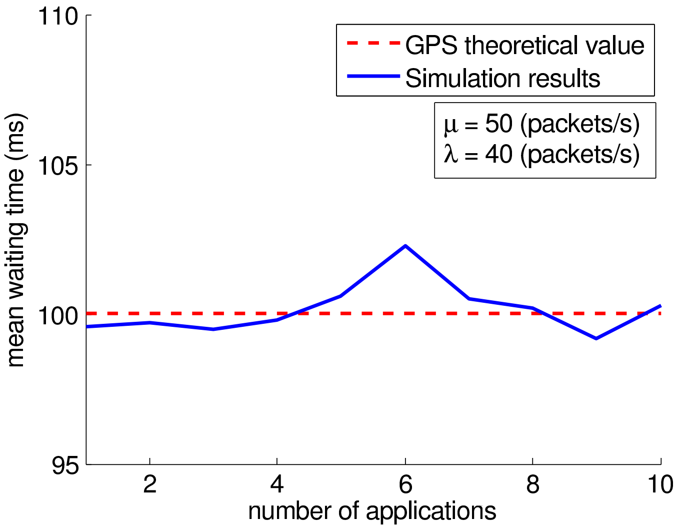


Fig. 4: Mean APP waiting time under fairly distributed traffic

B. GPS Processor under Unfairly Distributed Traffic

In this simulation, the setting of the mean service rate μ and overall packet mean arrival rate λ are set to be 100 (packets/s) and 90 (packets/s). In order to observe the GPS system in unfairly distributed traffic scenario, the number of the serving application is fixed to be two. λ_1 and λ_2 represent to the mean packet arrival rate of application 1 and 2, respectively. In Fig. 5, λ_2 always equal or greater than λ_1. In addition, considering the joint comparison of the upper and lower bound in eq.(5), the mean waiting time of application 1 is compared to its upper bond, whereas the mean waiting time of application 2 is compared to its lower bound. The reason is that application 2 has the higher probability to transmit a packet to processor, which implies application 2 might take all the computing resource more often. Theoretically, the mean waiting time of application 2 has no chance to break down to its lower bound. Vice versa, since that application 1 needs to share the computing resource with application 2 almost all the time, the mean waiting time may not exceed its upper bound. As shown in 5, the mean waiting time curve of application 2 always higher than its lower bound, whereas the curve of application 1 approaches but not crosses its upper bound. Here is an unpredictable phenomenon appears at the curve of application 2 mean waiting time. There are some large values emerge when the value of $\frac{\lambda_2}{\lambda_1}$ is less than 10. These peak entails that when the difference of the application arrival rate is not that big, the application who has higher arrival rate would suffer from resource contention more severely.

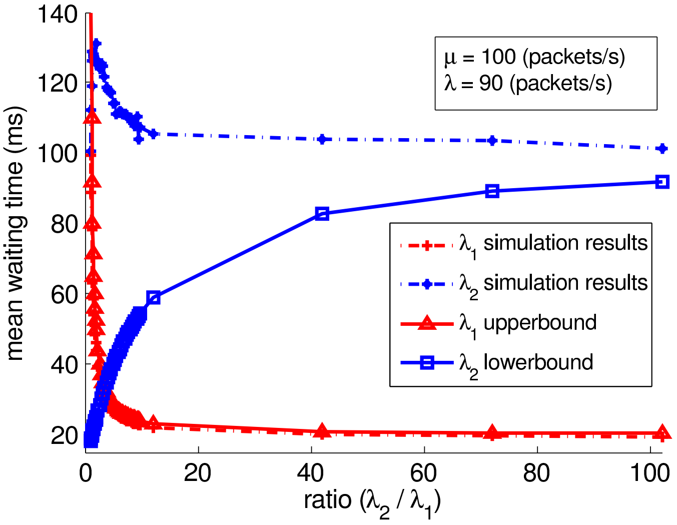


Fig. 5: Mean APP waiting time under unfairly distributed traffic

IV. Testbed Experiments

Sincerely, embedded an MEC platform in realistic wireless environment is a huge project and costly, building a virtual MEC platform environment seems to be a compromised way. The specifications of the virtual testbed equipment and operation system is in Table.I.

TABLE I: Testbed specifications

Type	Acer E5-475G-56US
CPU	Intel Core i5-6200U 2.3GHz
RAM	4GB DDR4
OS	Linux 14.04 LTS
Language	C11

A. Traffic Intensity and the CPU Utilization Comparison

The traffic intensity index ρ proposed in the queueing theorem is defined as the ratio of the theoretical average service rate and the packet average arrival rate, that is $\rho = \frac{\lambda}{\mu}$.

On the other hand, the definition of a processor CPU utilization U is the percentage of the executing period over one specific time slot, which can be read from the system report. Although the concept of these two indices are not exactly the same, there should be a strong relationship between them. Fig. 6 points out the result. To normalize the CPU utilization data, the entire CPU efficiency monitoring time slot is set to be one minute. FIFO, PS and GPS queue model are jointly tested in this experiment. Here, there are two applications running on each queue model. The mean arrival rate of each application is λ in FIFO, PS and GPS. Besides, the mean processor service rate is μ. As a result, the behavior of traffic intensity and CPU utilization act similarly in the three models. The growing curve of CPU utilization percentage is almost equal to traffic intensity, which means that the traffic intensity state could be probed.

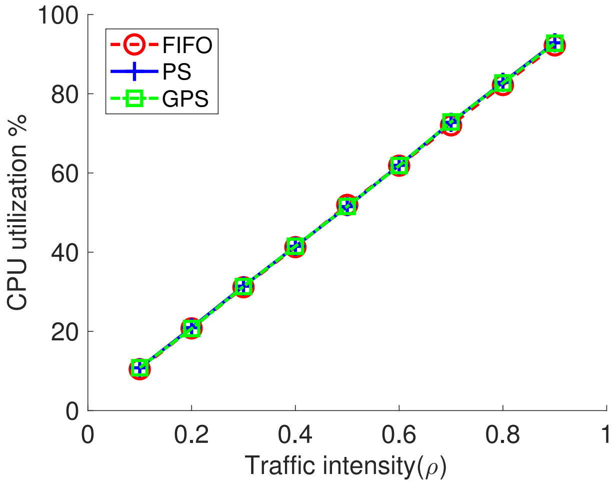
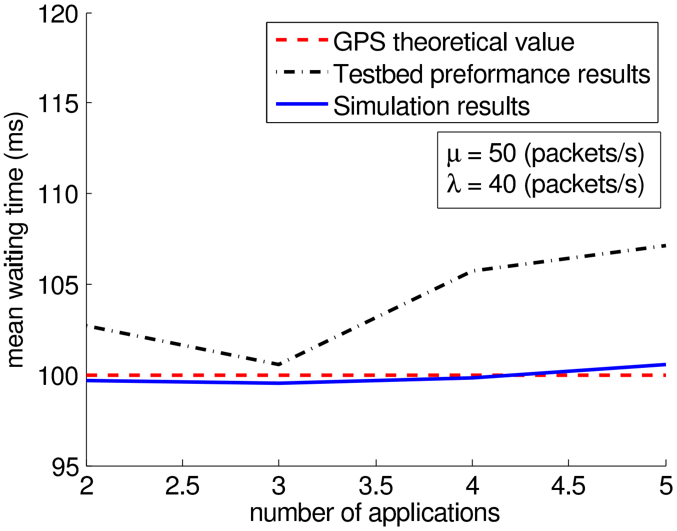
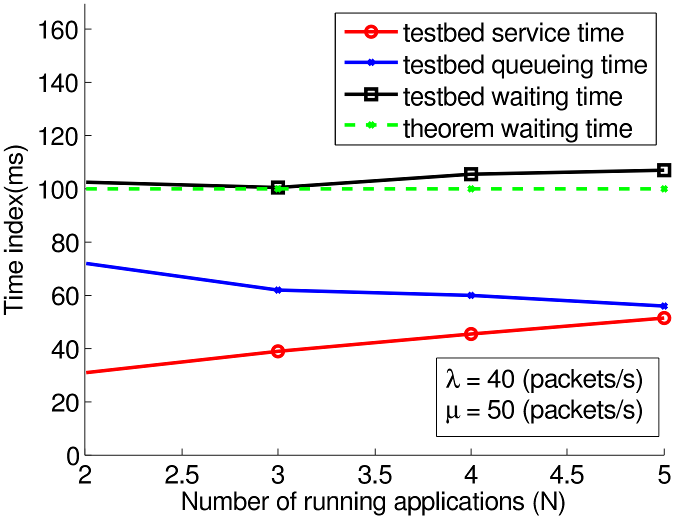


Fig. 6: Relationship between CPU utilization and traffic intensity



(a) APP mean waiting time in testbed environment



(b) Testbed CUP queueing and service time exploration

Fig. 7: GPS queueing model simulation

B. GPS Platform in testbed experiment

1) under Fairly Distributed Traffic: In this experiment, the mean service rate μ of processor is set to be 50 (packets/s). The total mean arrival rate λ is 40 (packets/s). As shown in Fig. 7a, while the total mean arrival rate is fixed, with the number of application increase, the individual mean waiting time read from the testbed data base almost matches with the theoretical (eq.(7)) and simulation waiting time. The experiment result points out that GPS model is suitable to describe MEC platform processing behavior. In MEC platform, when the total mean arrival rate is fixed, no matter how many applications are, the mean waiting time should stay.

In Fig. 7b, there are two additional data abstracted from the virtual MEC platform database, the service time and queueing time. As a result, the service time grows with the application number. It is quit rational because the processor have to separate the computing resource to accommodate all the applications, the serving time thus increases. And, the queueing time decreases with the application number. With the declination of individual packet mean arrival rate, it is considerable to observe the queueing time goes down. Finally, the mean waiting time could be calculated as the summation of the service time and queueing time.

2) Under Unfairly Distributed Traffic: Differ from III-B, in order to perform this experiment readily, some of the settings are changed here. There are also two applications running on the MEC platform, the mean service rate of processor μ is still fixed as 50 (packets/s), but the means packet arrival rate is different. Here, application 2 is set to be a background application with a constant arrival rate $\lambda_2 = 8$ (packets/s). On the other hand, the arrival rate of application 1 λ_1 becomes an independent variable varies with X-axis. Fig. 8 demonstrates the preforming result in virtual testbed. In the figure, the mean waiting time of application 1 increases seriously because of the self-congestion at the queueing part. However, although the mean waiting time of application 2 increases as well, the change is insignificant. This result concurs with the extrapolation in III, that is, No matter how big the difference of mean arrival rate is, the GPS processor could guarantee a upper bound waiting time for each application. Therefore, the traffic loading dominated application would never overwhelm the system.

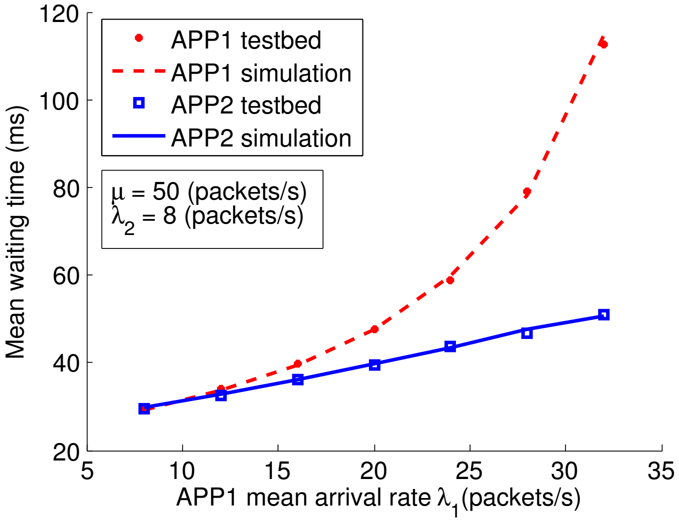


Fig. 8: Testbed experiment under unfairly distributed traffic

V. Two-Thresholds Forwarding Policy

In section IV-A, the relationship between system traffic intensity and processor CPU utilization factor has been certified. By reading the CPU utilization value, the entire traffic intensity could be easily collated. Moreover, some good features of GPS platform have been discussed and attested in section III. Hence, GPS model and platform becomes the one used to develop C-RAN and MEC system in this paper. On the merits point, GPS system could protect the light-traffic application from be overwhelmed by the heavy-traffic application. On the RAN deployment point, the structure of GPS system is more similar to the MEC structure proposed by ETSI [7].

A. Scenario

Refer to ETSI MEC document [7], the MEC structure could be briefly presented as shown in Fig. 9. When a new packet is coming, the BBU will check the headers and labels firstly to figure out the attribute of the packet. After, the packets would be forwarded to either the local server such as MEC BBU data computing part or to the cloud server, based on the demand of application, resource allocating situation and traffic condition. Obviously, the path to cloud is much longer than the path to the local servers, the propagation delay thus also needs to be taken into consideration when making a forwarding decision. In general, the computing capability in cloud would be more powerful than in MEC. Besides, there are some resource limitations in MEC platform such as computing capability, buffer size and power consuming restraint. Therefore, it is intuitive that more traffic accepted by the MEC platform, longer the processing and queueing delay would be experienced by the applications. In ETSI MEC system, there are several virtual machines (VMs) running on the MEC platform to serve the applications. Generally speaking, one VM is established with parts of the computing resource to cope with one specific application. Once the application is terminated, the corresponding VM would be closed and the resource would be released as well. Next, the algorithm mechanism based on such MEC scenario is going to be introduced.

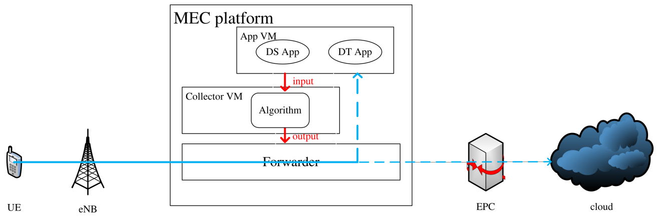


Fig. 9: Perspective of ETSI MEC structure

B. Two-Thresholds Forwarding Policy

To guarantee the quality of service provided by the MEC platform, the amount of data forwarded to the MEC platform should be well controlled under a policy. Based on the mathematical and simulation results, under a decent system traffic intensity, the mean waiting time could be statistically acceptable for each application. In this scenario (Fig. 9), assuming there are totally two types of application transmitting their packets to RAN, the delay sensitive (DS) application and delay tolerant (DT) application. In the proposed Two-Thresholds Forwarding Policy (TTFP), there are two thresholds to be set to improve the packet route of the applications. The first threshold is system busy state threshold, which avoids the overall MEC computing resource overloading. The second threshold is traffic intensity threshold for each application, which could maintain the traffic intensity situation of individual application. The setting of the second threshold is based on the latency requirement of each application. The following is the proposed algorithm.

In the script of TTFP algorithm, three possible cases are considered. First, when the current system CPU utilization exceeds the defined upper bound, in order to fulfill the latency requirement of DS APPs, the DT APP who has the greatest traffic intensity would be forward to cloud. Second, when the current system CPU utilization is below to the defined lower bound, the DT APP who has the lowest would be invited to process in MEC site, in case such invitation would not make the CPU utilization overflowed. Tertiary, the CPU utilization is within the range of defined upper and lower bounds. The system is under a balanced condition in this case, such condition would be hold until next execution. The following is the algorithm simulation.

Algorithm 1 Two-Thresholds Forwarding Policy

```
Require:
    Current CPU utilization: $U$
    System CPU computing efficiency: $\mu$
    Overall APP number: $N$
    Entire APP set in MEC: $\Phi = \{\Phi_{DS}, \Phi_{DT}\}$
    DS APP set in MEC: $\Phi_{DS} = \{\phi_i\}$, $i = 1, \ldots, m$
    DT APP set in MEC: $\Phi_{TS} = \{\phi_j\}$, $j = m+1, \ldots, N$
    APP set in Cloud: $\Phi_C = \{\phi_{c,r}\}$, $r \in N, \Phi_C \neq \emptyset$
    APP packet arrival rate set:$\Lambda = \{\lambda_i, \lambda_j, \lambda_r\}$
    Given system utilization upper/lower pound: $\Theta_U / \Theta_L$
    Given DS APP traffic intensity bound set: $\{\theta_i\}$
Ensure:
if $U > \Theta_U$ then
    repeat
        $\rho_{ds,i} = \frac{\lambda_i}{(\mu/N)}, i = 1, \ldots, m$
        $\rho_{dt,i} = \frac{\lambda_i}{(\mu/N)}, i = m+1, \ldots, N$
        Let $\max\{\rho_{dt,i}\} = \rho_{dt,k}, \max\{\rho_{ds,i}\} = \rho_{ds,q}$.
        while $\rho_{ds,q} > \theta_q$ do
            $\Phi_{DT} \backslash \phi_k, \Phi = \{\Phi_{DS}, \Phi_{DT}\}, N = N - 1$
        end while
    until $\{U \leq \Theta_U\} \cap \{\rho_{ds,q} \leq \theta_q\}$
    output: $\Phi$
else if $U < \Theta_L$ then
    repeat
        $\rho_{ds,i} = \frac{\lambda_i}{(\mu/N)}, i = 1, \ldots, m$
        Let $\max\{\rho_{ds,i}\} = \rho_{ds,q}$
        while $\rho_{ds,q}) < \theta_l$ do
            $N = N + 1, \rho_{dt,q} = \frac{\lambda_q}{(\mu/N)}$.
            if $\rho_{DS,q} < \theta_l$ then
                $\rho_{c,r} = \frac{\lambda_r}{(\mu/N)}, r \in N$
                Let $\min\{\rho_{c,r}\} = \rho_{c,g}$.
                $\Phi = \Phi \cup \phi_{c,g}$
            else
                $N = N - 1$
            end if
        end while
    until $\{\Theta_L \leq U \leq \Theta_U\} \cap \{\rho_{ds,q} \leq \theta_l\}$
    output: $\Phi$
else
    Keep current state, output: $\Phi$
end if
```

C. Algorithm simulation

The main functionality of TTFP is to emigrate the packet who can endure more processing delay to the cloud side. To check the performance, here comes a preliminary simulation result. In Fig. 10, the mean system serving rate is set to be 100 (packet/s), the mean arrival rate of DS APP is fixed to be 45 (packet/s) and DT APP varies with x-axis. As the figure shown, the behavior of the two curves is exactly the same until the DT APP mean arrival rate goes to 44 (packet/s). At the critical point, the number of the overall arrival packets almost reaches the system processing limitation, which means that the processor would no longer be able to handle all the incoming packets. Therefore, the waiting time of DS APP increases dramatically without TTFP algorithm. In the contrast, the mean waiting time of DS APP is shorter than the beginning when TTFP is triggered. Since that there is only one DT APP in this simulation, once TTFP triggered, all the DT APP traffic would be steered to cloud side, only DS APP traffic is forwarded to MEC site and enjoys the entire MEC computing resource. The simulation results matches the expectation, MEC system would handle the DS packet only in case the computing limitation is reached.

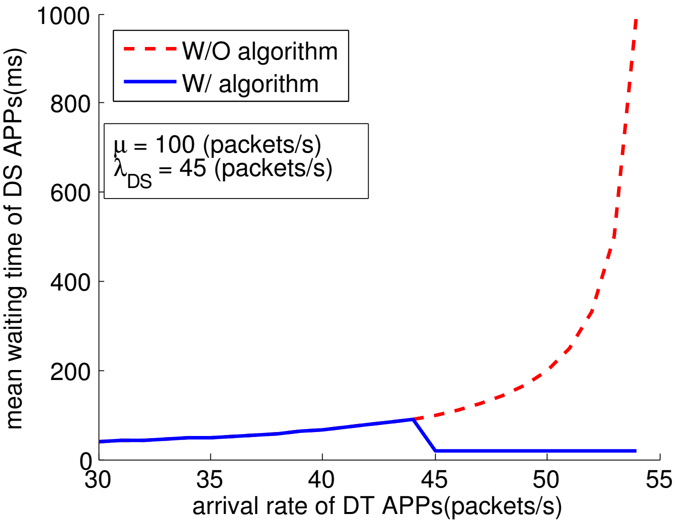


Fig. 10: Preliminary algorithm simulation result

VI. Conclusion

”Latency” is one of the most critical key words in future mobile networks. To satisfy low-latency requirements, C-RAN with MEC architecture seems to be able to achieve the goal. In general purpose processor(GPP) platform, the MEC system could handle not only baseband processing but also data computing, which makes the utilization of computing resource more efficient. In this paper, the relationship between system traffic intensity and CPU utilization is certified, and the merits of generalized-processor-sharing model(GPS) model is also presented. After discussing, GPS system is more appropriate to be the reality compared model than first-in-first-out(FIFO) and processor-sharing(PS) model. In this paper, Two-Thresholds Forwarding Policy(TTFP) algorithm is proposed to dynamically arrange the data traffic of applications according to the current system state. According to the result in algorithm simulation, implementing TTFP could fulfil the latency requirement of delay sensitive APPs as possible.

In the future, TTFP algorithm will be extended to accommodate various applications who has multi latency requirements. A wide-paving-sensor disaster alarming system could be an example. In such system, it is easy to imagine that there are many kinds of information need to be transmitted to the center, such as normal observation data, condition changed data and emergency alarming data. These data have different levels of latency requirements. Therefore, TTFP algorithm has to be able to handle three or even more kinds of latency requirement levels. Moreover, the proposed algorithm will be implemented on some quasi C-RAN and MEC environments in reality to validate the practical performance. Now our research team is looking for some chances to collaborate with some companies and organizations who has deployed C-RAN and MEC related systems.

References

[1] N. Bhushan, J. Li, D. Malladi, R. Gilmore, D. Brenner, A. Damnjanovic, R. Sukhavasi, C. Patel, and S. Geirhofer, "Network densification: the dominant theme for wireless evolution into 5G," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 82–89, 2014.

[2] A. Checko, H. Holm, and H. Christiansen, "Optimizing small cell deployment by the use of C-RANs," *European Wireless 2014; 20th European Wireless Conference*, pp. 1–6, Jun. 2014.

[3] M. Weiner, M. Jorgovanovic, A. Sahai, and B. Nikolie, "Design of a low-latency, high-reliability wireless communication system for control applications," *Communications (ICC), 2014 IEEE International Conference on*, pp. 3829–3835, Aug. 2014.

[4] R. Wang, H. Hu, and X. Yang, "Potentials and challenges of C-RAN supporting multi-RATs toward 5G mobile networks," *IEEE Access*, vol. 2, pp. 1187–1195, Oct. 2014.

[5] C. Mobile, "C-RAN: the road towards green RAN," *White Paper, ver*, vol. 2, Oct. 2011.

[6] Y. J. Ku, D. Y. Lin, C. F. Lee, P. J. Hsieh, H. Y. Wei, C. T. Chou, and A. C. Pang, "5g radio access network design with the fog paradigm: Confluence of communications and computing," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 46–52, April 2017.

[7] ETSI MEC ISG, "Mobile edge computing (MEC); Technical Requirements GS MEC 002 V1.1.1," Mar. 2016.

[8] ——, "Mobile edge computing (MEC); Framework and Reference Architecture GS MEC 003 V1.1.1," Mar. 2016.

[9] F. Boccardi, R. Heath, A. Lozano, T. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 74–80, Feb. 2014.

[10] "Software-Defined radio technology overview," *White Paper*, Aug. 2002.

[11] M. Bansal, J. Mehlman, S. Katti, and P. Levis, "Openradio: a programmable wireless dataplane," *Proceedings of the first workshop on Hot topics in software defined networks*, pp. 109–114, Aug. 2012.

[12] M. Khan, R. Alhumaima, and H. Al-Raweshidy, "Reducing energy consumption by dynamic resource allocation in C-RAN," *Networks and Communications (EuCNC), 2015 European Conference on*, pp. 169–174, Jul. 2015.

[13] Z. Kong, J. Gong, C. Xu, K. Wang, and J. Rao, "eBase: A baseband unit cluster testbed to improve energy-efficiency for cloud radio access network," *Communications (ICC), 2013 IEEE International Conference on*, pp. 4222–4227, Jun. 2013.

[14] C. Liu, K. Sundaresan, M. Jiang, S. Rangarajan, and G. Chang, "The case for re-configurable backhaul in cloud-RAN based small cell networks," *Infocom, 2013 Proceedings IEEE*, pp. 1124–1132, Jul. 2013.

[15] K. Sundaresan, M. Arslan, S. Singh, S. Rangarajan, and S. Krishnamurthy, "FluidNet: A flexible cloud-based radio access network for small cells," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 915–928, Apr. 2016.

[16] S. Namba, T. Warabino, and S. Kaneko, "BBU-RRH switching schemes for centralized RAN," *Communications and Networking in China (CHINACOM), 2012 7th International ICST Conference on*, pp. 762–766, Aug. 2012.

[17] I. Chih-Lin, C. Rowell, S. Han, Z. Xu, G. Li, and Z. Pan, "Toward green and soft: a 5G perspective," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 66–73, Feb. 2014.

[18] G. Li, S. Zhang, X. Yang, F. Liao, T. Ngai, S. Zhang, and C. Kuilin, "Architecture of GPP based, scalable, large-scale C-RAN BBU pool," *Globecom Workshops (GC Wkshps), 2012 IEEE*, pp. 267–272, 2012.

[19] L. Liu, F. Yang, R. Wang, Z. Shi, A. Stidwell, and D. Gu, "Analysis of handover performance improvement in cloud-RAN architecture," pp. 850–855, Aug. 2012.

[20] E. Ahmed, A. Akhunzada, M. Whaiduzzaman, A. Gani, H. Ab, H. Siti, and R. Buyya, "Network-centric performance analysis of runtime application migration in mobile cloud computing," *Simulation Modelling Practice and Theory*, vol. 50, pp. 42–56, Jan. 2015.

[21] J. Liu, E. Ahmed, M. Shiraz, A. Gani, R. Buyya, and A. Qureshi, "Application partitioning algorithms in mobile cloud computing: Taxonomy, review and future directions," *Journal of Network and Computer Applications*, vol. 48, pp. 99–117, Feb. 2015.

[22] Y. Shih, W. Chung, A. Pang, T. Chiu, and H. Wei, "Enabling Low-Latency Applications in Fog-Radio Access Network," *IEEE Networks*, vol. 31, no. 1, pp. 52–58, Jan. 2017.

[23] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," *MCC workshop on Mobile cloud computing*, pp. 13–16, Aug. 2012.

[24] M. Chiang and T. Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, Jun. 2016.

[25] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal *et al.*, "Mobile-edge computing introductory technical white paper," *White Paper, Mobile-edge Computing (MEC) industry initiative*, Sept. 2014.

[26] M. Beck, M. Werner, S. Feld, and T. Schimper, "Mobile edge computing: A taxonomy," Jan. 2014.

[27] K. Habak, M. Ammar, K. Harras, and E. Zegura, "Femto clouds: Leveraging mobile devices to provide cloud service at the edge," *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, pp. 9–16, Jun. 2015.

[28] D. Gross, J. F. Shortle, J. M. Thompson, and C. M. Harris, *Fundamentals of Queueing Theory*, 4, Ed. Wiley, Aug. 2008.

[29] E. Coffman and L. Kleinrock, "Feedback Queueing Models for Time-Shared Systems," *Journal of the ACM*, vol. 15, no. 4, pp. 549–576, Oct. 1968.

[30] Y. Ku, D. Lin, and H. Wei, "Fog RAN over General Purpose Processor Platform," *Vehicular Technology Conference (VTC-Fall), 2016 IEEE 84th*, pp. 1–2, Sept. 2016.