Resource Allocation and Distributed Uplink Offloading Mechanism in Fog Environment

Linna Ruan, Zhoubin Liu, Xuesong Qiu, Zixiang Wang, Shaoyong Guo, and Feng Qi

Abstract: **Applying fog computing technology to the shared pattern has two problems to cope with. One is to formulate a rational mechanism for resource allocation, and the other is to design computation offloading strategy of tasks based on resource allocation result. For solving these problems, we construct a three-layer F-RAN architecture first, which consists of terminal layer, access layer and network layer. Second, we adopt differential game and bipartite graph multiple matching algorithm to solve bandwidth resource allocation problem of fog node (FN)-access point (AP) and AP-shared terminal (ST), respectively. Third, Lyapunov theory and proposed deviation update decision algorithm (DUDA) are used to solve computation offloading decision-making and offloading update order-making. At last, simulation results show that our strategy can save 30%-60% system consumption, and the resource demand satisfaction rate can be guaranteed to reach 80% or more.**

Index Terms: **Differential game theory, Distributed uplink offloading, F-RAN, Resource allocation, Sharing pattern.**

I. INTRODUCTION

SHARING pattern realizes green development through reuse of resources, which is an effective way to cope with growing needs of the community. In view of Hawking's reiteration of humanity's extinction on June 20 and report of 15,000 scientists on "independence" in 2017, we can draw a conclusion that humans will be destroyed if remain indifferent to the issue of resources. The pattern of sharing then has drawn wide attention from all walks of life; This reform of kinds of resources has given birth to the formation of a shared economic model and is expected to really alleviate the problem of resource shortage. Therefore, more and more shared concept products assimilate into life like shared bicycle. Considering the high demand of shared products in terms of geographic location, delay and energy consumption, this paper uses fog computing model to realize applications in this scenario. Sharing pattern and fog computing are introduced in the following parts to illustrate the significance of this study.

A. Sharing Pattern

Real-time upload of terminal information is a key factor to ensure the quality of shared services. Due to the large number of terminals and limited bandwidth resources, it is particularly important to design a reasonable resource allocation strategy to alleviate competition. In addition, terminals also have some computing tasks to cope with, such as generating random passwords, uploading fault information, etc. How to design a suitable offloading mechanism is also one of our concerns.

B. Fog Computing

As an extension of cloud computing, fog computing sinks computing ability to the edge of network, resulting in less delay. This feature can effectively support real-time upload of shared terminals' information. In addition, its geo-location awareness feature can effectively support shared terminals' high demand for location information. Therefore, this paper adopt fog computing to support shared services.

The main contribution of this paper can be summarized as follows:

Build shared pattern oriented network model based on F-RAN. Some existing researches have also built models based on F-RAN, but we have not seen a study focus on shared pattern scenario. In this paper, we build a three-layer network architecture model for shared pattern. By the way, a partial offloading network model is also built, which is closer to actual scene than completely local execution or offloading.

Design a resource allocation strategy based on differential game and bipartite graph multiple matching. For the problem of how FN resources be reasonably allocated among all users in each cell, we use differential game and bipartite graph multiple matching. First, the problem of resource allocation between FNs and APs is formulated as a differential game. Feedback Nash equilibrium is achieved by solving how many resources each AP can obtained. Second, bipartite graph multiple matching is adopted to achieve maximum traffic matching between APs and STs.

Propose a distributed uplink computation offloading strategy with Lyapunov theory and deviation update decision algorithm (DUDA). Most researches on offloading mechanism considered system overhead while neglected system stability and offloading update order-making. In this paper, we design a distributed uplink computation offloading strategy with Lyapunov theory and DUDA mechanism. System stability is guaranteed first when pursuing for minimum system consumption with Lyapunov. Inter-cell competition of decision update opportunities is also solved with proposed DUDA.

The remainder of this article is organized as follows. Related work is introduced in Section II. A network architecture is constructed in Section III. The resource allocation strategy based on differential game and bipartite graph multiple matching is shown in Section IV. Computation offloading system model in-

Manuscript received December 14, 2017.

This work was supported by State Grid Technology Project 'Edge Computing Research in Smart Grid Application and Security' $(Grant.52110118001H)$; National Natural Science Foundation of China $(Grant.61702048)$

L. Ruan, X. Qiu, S. Guo, and F. Qi are with the State Key Laboratory of Networking & Switching Technology, Beijing University of Posts and Telecommunications, email: {linnaRuan, xsqiu, syguo, qifeng}@bupt.edu.cn.

Z. Liu and Z. Wang are with the State Grid Zhejiang Electric Company, email: {liuzhoubin, wangzixiang}@zj.sgcc.com.cn.

X. Qiu is the corresponding author.

Digital Object Identifier: 10.1109/JCN.2018.000037

cluding communication and computing model is formulated in Section V. Section VI introduces the best offloading strategy set based on Lyapunov and offloading mechanism with proposed DUDA. Section VII describes the performance analysis of our algorithm. Section VIII shows simulations and results analysis of the paper. The work of this paper and prospects for possible future research are summarized in Section IX.

II. RELATED WORK

In order to solve the problem of resource allocation and computation offloading for fog computing, some scholars have carried on related research.

A. Resource Allocation

Game theory is one of the common methods in resource allocation. Based on Stackelberg game and matching principal, an optimization resource allocation approach was proposed in [1]. It constructed a three-tier IoT fog network architecture consists of data service subscribers (DSSs), FNs and data service operators (DSOs) to clarify IoT issue, but the formulation of both the pricing and preference tables in FN-DSO mapping was not clear. In [2], resource allocation strategy for fog computing based on priced timed Petri nets (PTPNs) was studied, which considered price cost and time cost comprehensively. A hierarchical game framework for resource management was built in [3]. There were also some works pay attention to resource allocation for edge computing [4]–[6].

B. Computation Offloading

Hybrid computation offloading problem was discussed in [7], which took cloud computing servers and fog computing servers into consideration. Aiming to minimize total energy consumption for both communication and computation, four subproblems according to computation energy efficiency were constructed. In [8], workload allocation problem was discussed. With constrained service delay, it proposed an efficiency workload allocation method to minimize power consumption by solving three subproblems. An optimization framework of offloading from a single mobile device (MD) to multiple edge devices was proposed in [9]. In [10], device-to-device (D2D) Fogging was studied. It formulated an optimization problem that aims at minimizing time-average energy consumption and developed an online task offloading algorithm based on Lyapunov to reduce energy consumption.

Disadvantage of the above research is that only resource allocation or computation offloading are considered, failed to complete the entire process of optimization program formulation. In addition, there are also several kinds of research focus on the joint solution to solve problems. An integrated framework for computation offloading and interference management in wireless cellular networks with MEC was proposed in [11]. It proved one-sidedness of considering computation offloading strategy alone. Supply lags behind demand bought by fixed resource possession cannot adapt to the change of offloading decision will significantly affecting the performance improvement that offloading strategy formulation should bring. In [12], a joint resource allocation and coordinated computation offloading method was designed. But it only considered the case of a single fog node. F-RAN was also mentioned in [13]. It was a review article focus on issues and challenges. A F-RAN architecture was also proposed in this article, but cell concept in the terminal layer was ignored.

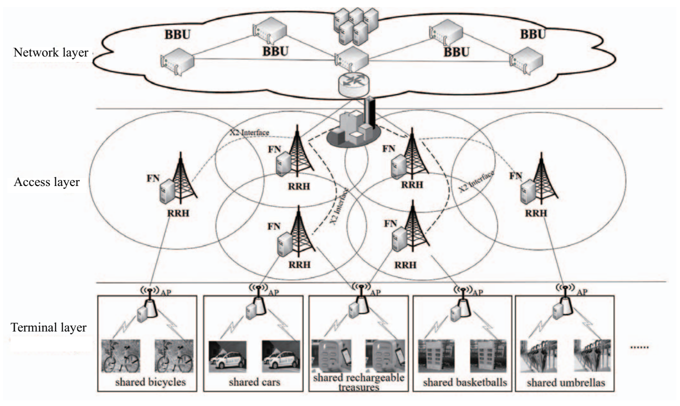

Fig. 1. Network architecture.

Compared with existing research, our main contributions can be summarized as considering the situation of multiple terminals with multiple fog nodes, and discussing the joint implementation of resource allocation as well as task uplink partial offloading for rapidly developing shared service scenario.

III. NETWORK ARCHITECTURE

Resource allocation and task computation offloading in shared pattern under F-RAN architecture are discussed in this paper. We propose a system model with three-layer architecture as shown in Fig. 1, which comprises of terminal layer, access layer, and network layer. Shared bicycles, shared cars, shared rechargeable treasures, shared basketballs and shared umbrellas are used as an example on behalf of the shared terminals, and their information is uploaded through local AP. Access layer include BSs and servers, which act as FNs. They are connected in a tree-like topology via X2 interfaces, confirmed about 50 percent lower with significant reduced network deployment and maintenance cost compared with mesh topology [14]. Network layer consists of cloud data storage center as well as BBU pool. FNs has certain computing capabilities which are numbered as $i = \{1, 2, \cdots, M\}$. APs are numbered as $j = \{1, 2, \cdots, N\}$. They act as bridging devices, connecting shared terminals to upper-layer base stations. STs are numbered as $k = \{1, 2, \cdots, K\}$. In this paper, we assume each shared terminal has a task, which can be divided.

Notations that will be used in the rest of this paper are summarized in Table 1. The overall calculation process is shown in Fig. 2. We will describe it in detail in the following subsections.

IV. RESOURCE ALLOCATION STRATEGY BASED ON DIFFERENTIAL GAME AND BIPARTITE GRAPH MULTIPLE MATCHING

According to the architecture constructed in this paper, bandwidth resource allocation between FN-AP and AP-ST needs to

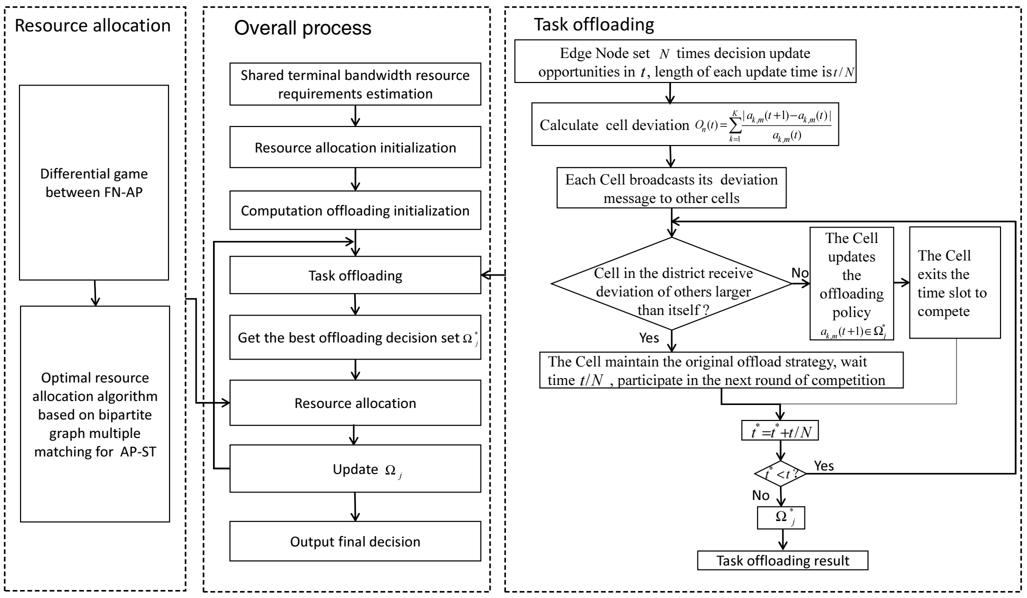
Resource allocation
Differential game between FN-AP
Optimal resource allocation algorithm based on bipartite graph multiple matching for AP-ST
Overall process
Shared terminal bandwidth resource requirements estimation
Resource allocation initialization
Computation offloading initialization
Task offloading
Get the best offloading decision set Ω^*_j
Resource allocation
Update Ω_j
Output final decision
Task offloading
Edge Node set N times decision update opportunities in t, length of each update time is t/N
Calculate cell deviation $O_n(t)=\sum_{k=1}^{K}\frac{|a_{k,m}(t+1)-a_{k,m}(t)|}{a_{k,m}(t)}$
Each Cell broadcasts its deviation message to other cells
Cell in the district receive deviation of others larger than itself?
No
The Cell updates the offloading policy $a_{k,m}(t+1)\in\Omega^*_j$
The Cell exits the time slot to compete
Yes
The Cell maintain the original offload strategy, wait time t/N, participate in the next round of competition
$t^*=t^*+t/N$
$t^*<t$?
Yes
No
Ω^*_j
Task offloading result

Fig. 2. Overall calculation process.

Table 1. Summary of definitions and symbols.

Symbol	Definition
M	Total number of FNs
N	Total number of Cells(APs)
K	Total number of shared terminals in one Cell
$\phi(AP_j)$	Bandwidth capacity of APj
C_{sk}	Bandwidth request of STk
$a_{k,m}(t)$	Offloading decision of STk to FNm
$R_{k,m}(t)$	User uplink data transfer rate of STk to FNm
$P_k(t)$	Transmission power of STk
$g_{k,m}(t)$	Large scale channel gain from STk to FNm
$h_k(t)$	Small-scale channel fading gain of STk
$W_k(t)$	Bandwidth obtained of shared terminal
$\omega_k(t)$	Background interference power of STk
$I_k(t)$	Computing task of STk
$B_k(t)$	Input data size of STk
$D_k(t)$	Total number of CPU cycles required for I_k
$F_k^{(l)}(t)$	Local computing capability of STk
$F_{k,m}^{(f)}(t)$	Computation capability assigned to STk by FNm
$E_k^{(l)}(t)$	Energy consumed by local computing part of task
$\mu_k(t)$	Energy consumption in each CPU cycles
$Z_k^{(l)}(t)$	Total cost of STk in local calculation section
$T_{k,\text{off}}^{(f)}(t)$	Time consumption of STk during offloading
$E_{k,\text{off}}^{(f)}(t)$	Energy consumption of STk during offloading
$\lambda_k^{(T)}(t)$	Weight of computational time
$\lambda_k^{(E)}(t)$	Weight of computational energy
Ω_j	Offloading decision set of Cell$j$
O_j	Deviation of Cell$j$

be considered.

A. Game between FN and AP

As the location of FNs and APs are relatively fixed, the main problem between those two kinds of objects is how each FN allocates resources for APs it matches. Taking into account dynamic changes of bandwidth resources, we use differential game to achieve real-time allocation.

FN is symbolled as F_i, and AP of the jth cell is symbolled as A_j. $F_i(\tau)$ and $A_j(\tau)$ represent allocatable resource rate of FN and available resource rate of AP, respectively. α_i means yield of resource distribution, and β_j means cost of resource obtained. According to differential game theory, the objective function of each AP can be expressed as

$$\max_{A_j}\int_0^T\left[\frac{\alpha_iF_i(\tau)-\beta_jA_j(\tau)}{F_i(\tau)}A_j(\tau)\right]\exp(-r\tau)d\tau,\quad j\in\{1,2,\cdots,N\},\ \tau\in[0,T],\tag{1}$$

where r represents constant discount rate. The goal of this resource-based game process is to maximize system revenue.

Bandwidth resource can be sold of F_i changes as

$$\dot{F_i}(\tau)=F_i(\tau)-\sum_{j=1}^{N}A_j(\tau),\ \ F_i(0)=1.\tag{2}$$

Lemma 1: Resource procession of APj under best resource decision-making is

$$\phi^*(AP_j)=\sum_{i=1}^{M}\varphi_j^*(t,F_i)=\sum_{i=1}^{M}F_i(t)[\frac{\alpha_i}{\beta_j}-G_j(t)]/2,\ \ j\in N.\tag{3}$$

Proof: To find out the optimal feedback strategies of differential games (1) and (2), Nash feedback equalization solution must satisfy the following relation according to feedback Nash equilibrium theorem of deterministic differential games:

$$\begin{aligned}&-V_t^j(t,F_i)\\&=\max_{A_j}\Bigg\{\left[\frac{\alpha_iF_i(t)-\beta_jA_j(t)}{F_i(t)}A_j(t)\right]\exp(-rt)\\&\quad+V_F^j(t,F_i)\big[\alpha_iF_i(t)-\beta_jA_j(t)-[\varphi_1^*(t,F_i)+\varphi_2^*(t,F_i)+\cdots\\&\quad+\varphi_{k-1}^*(t,F_i)+A_j(t,F_i)+\cdots+\varphi_N^*(t,F_i)]\big]\Bigg\},j\in N\\&V^j(T,F_i)=0,j\in N.\end{aligned}\tag{4}$$

In order to get $A_j(t)$ maximize (4), we find the first derivative of $A_j(t)$ on both sides of the equation. Best decision $\varphi_j^*(t, F_i)$ can be got through setting the first derivative equal to zero.

$$-\frac{\partial V_t^j(t,F_i)}{\partial(A_j(t))} = \left[-\frac{\beta_j A_j(t)}{F_i(t)} + \frac{\alpha_i F_i(t)-\beta_j A_j(t)}{F_i(t)}\right]\exp(-rt) - \beta_j V_F^j(t, F_i) = 0, \tag{5}$$

$$\varphi_j^*(t, F_i) = F_i(t)[\frac{\alpha_i}{\beta_j} - V_F^j(t, F_i)\exp(rt)]/2. \tag{6}$$

Bring the value of $\varphi_j^*(t, F_i)$ into (4), we can get

$$-V_t^j(t, F_i) = \left\{\frac{F_i(t)[\frac{\alpha_i}{\beta_j} - V_F^j(t,F_i)\exp(rt)]}{2} - \frac{F_i(t)[\frac{\alpha_i}{\beta_j} - V_F^j(t,F_i)\exp(rt)]^2}{4}\right\}\exp(-rt) + V_F^j(t, F_i)\left\{F_i(t) - \frac{F_i(t)}{2}\sum_{i=1}^{M}[\frac{\alpha_i}{\beta_j} - V_F^j(t, F_i)\exp(rt)]\right\}, \; j \in N. \tag{7}$$

Solve (7), the following equation can be obtained.

$$V^j(t, F_i) = [G_j(t)F_i + H_j(t)]\exp(-rt), \; j \in N, i \in M, \tag{8}$$

where $G_j(t)$ and $H_j(t)$ suit

$$\begin{cases} \dot{G_j}(t) = -\frac{G_j(t)^2}{4} + (r - \frac{1}{2})G_j(t) + \\ \qquad \frac{G_j(t)}{2}\sum\limits_{j=1,j\neq i}^{N}(\frac{\alpha_i}{\beta_j} - G_j(t)) - \frac{\alpha_i}{2\beta_j} + \frac{1}{4}(\frac{\alpha_i}{\beta_j})^2 \\ \dot{H_j}(t) = r * H_j(t) \\ G_j(T) = 0, H_j(T) = 0. \end{cases} \tag{9}$$

Bring $V_{F_i}^j(t, F_i)$ from (8) into (7), we can get

$$\varphi_j^*(t, F_i) = F_i(t)[\frac{\alpha_i}{\beta_j} - G_j(t)]/2, \; j \in N. \tag{10}$$

Therefore, by solving relation (9) of $G_j(t)$ and $F_i^*(t)$, and then substituting their values into (10), we can get feedback Nash equilibrium strategy of APj.

B. Bipartite Graph Multiple Matching between APs and STs

After differential game, each AP has obtained resources from FNs, and then, task has to compete for the resources of APs by shared terminals in each cell. AP-ST matching model based on bipartite graph multiple matching is shown in Fig. 3. Our goal is to meet terminals resource needs with best effort.

AP layer is considered as Part A and terminal layer is considered as Part B. As matching relationship of the two parts not follow a simple one-to-one mode, it needs to extend traditional bipartite graph matching to multiple matching. Considering the constraint amount of bandwidth resources in each AP and required by STs, it is necessary to first virtualize a source point s and a convergence point t. Next, connect s and APs with an edge, which capacity is $\phi(AP_j)$ according to (3). Set up an edge with capacity 1 between APj and STk, and finally connect all STk-t with the capacity of C_{sk}. According to bipartite graph

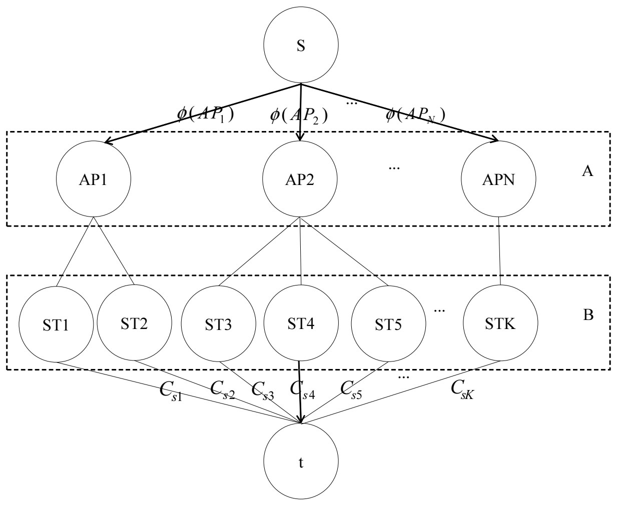

Fig. 3. Bipartite graph multiple matching between APs and STs.

multiple matching algorithm, we can get the final network flow graph, from which the maximum flow allocation scheme can be obtained. In detail, edge traffic of s-APj represents bandwidth resources that participate in the allocation of APj. Edge traffic of APj-STk indicates whether APj allocates bandwidth resource to STk. If allocates, the traffic is 1, otherwise, the traffic is 0. Edge traffic of STk-t means the amount of bandwidth resources that STk obtained. When APs cannot meet requirements of STs, scheme with largest output stream is selected. If there exist different cases with the largest output stream, we choose to satisfy needs of the terminal with higher service priority first. The algorithm is shown as below.

Algorithm 1: Optimal resource allocation algorithm based on bipartite graph multiple matching of AP-ST

```
input          : Each AP has a resource capacity value
                 $\phi(AP_j)$ according to resource it obtained
                 from FNs; each ST has a resource
                 requesting value $C_{sk}$;
output         : Resource that each ST can obtained;
initialization: All nodes are marked as 0, indicating that
                 they have not been visited;
1  for each time slot $\tau \in T$ do
2      summarize the number of resources for each ST
         symbolled as $SCa$;
3      get maxflow through finding augment path;
4      modify graph;
5  end
6  if maxflow= $SCa$ then
7      all of the STs' resource needs can be met;
8  else
9      APs' resources can not meet the needs of STs;
10     choose to satisfy needs of terminal with higher service
         priority first;
11 end
```

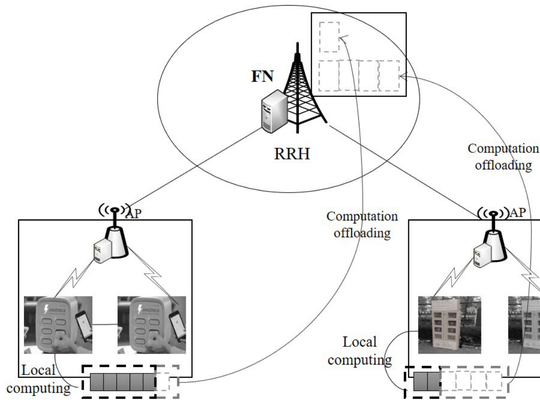

Fig. 4. Partial offloading network model.

V. Computation Offloading System Model

After finishing resource allocation in each time slot, we should make effort on computation offloading next. System consumption including time and energy aspects will be discussed in this part. Partial offloading is considered in this paper as shown in Fig. 4. Since communication and computing are the main aspects that generate system consumption in data transmission and processing, they are modeled and described in this section. $a_{k,m}(t)$ is used to represent offloading decision of STk to FNm. As we consider, user's task can be divided, so the value of $a_{k,m}(t)$ is in the range of 0 to 1 decided by our strategy. If task can be executed in local directly, $a_{k,m}(t) = 0$ can be got. Correspondingly, $a_{k,m}(t) = 1$ means full task of user should be offloaded to FNm. Other values represent part of the task is executed locally and the remaining part is offloaded to FN.

A. Communication Model

Offloading communication model is constructed for uplink computing from STs to APs. In this paper, we consider OFDMA is used in data transmission process, which means interference only exist among cells. Shannon formula can be used to calculate uplink data transfer rate according to (11).

$$R_{k,m}(t) = W_k(t)\log_2\left(1+\frac{P_k(t)g_{k,m}(t)h_k(t)}{\omega_k(t)+\sum\limits_{q\in K\setminus\{k\}:a_q\neq 0} P_q(t)g_{q,m}(t)h_q(t)}\right)$$
$$(W_k(t) = C_{sk}), \tag{11}$$

where $P_k(t)$ represents transmission power of STk, $g_{k,m}(t)$ represents large-scale channel gain from STk to FNm, including path loss and shadow fading. $h_k(t)$ represents small-scale channel fading gain that follows Rayleigh distribution. $W_k(t)$ means bandwidth resource that STk obtained, and ω_k means background interference power.

B. Computing Model

We assume each ST has a computing task $I_k(t) \triangleq (B_k(t), D_k(t))$. The unit of $B_k(t)$ is KB, which indicates size of input data, and the unit of $D_k(t)$ is megacycle, which means the total number of CPU cycles required to complete computing tasks regardless of network computing power. Their value can be got with the method mentioned in [15] and [16]. Computational consumption of local and FN is discussed next, including time as well as energy aspects according to [17].

1) Local computing

Local computing corresponds to the part of task that chooses to compute with terminal equipment. Its time and energy consumption can be expressed as

$$T_k^{(l)}(t) = (1-\sum_{m=1}^{M} a_{k,m}(t))\frac{D_k(t)}{F_k^{(l)}(t)} \tag{12}$$

and

$$E_k^{(l)}(t) = (1-\sum_{m=1}^{M} a_{k,m}(t))\mu_k(t)D_k(t), \tag{13}$$

respectively. According to the actual measurement results shown in [18], μ_k is a parameter that represents energy consumed by each CPU cycle, and it can be set by $\mu_k(t) = 10^{-11}(F_k^{(l)}(t))^2$.

2) Fog computing

The offloaded part of the task $I_k(t)$ is calculated in FN. Task offloading and execution process are the main aspects bring time and energy consumption. The part that generated from offloading process can be expressed as

$$T_{k,\text{off}}^{(f)}(t) = \sum_{m=1}^{M} a_{k,m}(t)\frac{B_k(t)}{R_{k,m}(t)} \tag{14}$$

and

$$E_{k,\text{off}}^{(f)}(t) = P_k(t)T_{k,\text{off}}^{(f)}(t) = \sum_{m=1}^{M} a_{k,m}(t)\frac{P_k(t)B_k(t)}{R_{k,m}(t)}, \tag{15}$$

respectively. Since APs transmits data to FNs in wired way, no interference as well as consumption exist in this process. Therefore, the above two formulas can show consumption of offloading process.

The other part of consumption is generated during execution of tasks. Use $F_{k,m}^{(f)}(t)$ to indicate computing capability (the number of CPU cycles executed per second) that FNm assigned to STk. Time spent in execution process is given by

$$T_{k,\text{exe}}^{(f)}(t) = \sum_{m=1}^{M} a_{k,m}(t)\frac{D_k(t)}{F_{k,m}^{(f)}(t)}. \tag{16}$$

Since this article is based on ST's point of view, energy consumed by offloading task in FN is omitted here [19].

3) System consumption

The consumption of task executed in local area and offloaded to FN is considered to be total consumption of the system. Serial mode is selected in this paper. As a result, time and energy consumption of system is expressed as

$$T_k(t) = T_k^{(l)}(t) + T_{k,\text{off}}^{(f)}(t) + T_{k,\text{exe}}^{(f)}(t) \tag{17}$$

and

$$E_k(t) = E_k^{(l)}(t) + E_{k,\text{off}}^{(f)}(t), \tag{18}$$

respectively. While, because of the units of response time and energy expenditure are different, the two indicators should be normalized as

$$\gamma_k^T(t) = \frac{T_k(t)}{T_{k,\max}(t)} \tag{19}$$

and

$$\gamma_k^E(t) = \frac{E_k(t)}{E_{k,\max}(t)}, \tag{20}$$

where, $T_{k,\max}(t)$ and $E_{k,\max}(t)$ correspond to completely compute in local area. $\gamma_k^T(t)$ and $\gamma_k^E(t)$ are the representations of time and energy consumption after normalized. Total system consumption can be expressed as

$$\begin{aligned} &Z_k(t) = \lambda_k^{(T)}(t)\gamma_k^T(t) + \lambda_k^{(E)}(t)\gamma_k^E(t) \\ &(0 \le \lambda_k^{(T)}(t), \lambda_k^{(E)}(t) \le 1 \text{ and } \lambda_k^{(T)}(\text{t})+\lambda_k^{(E)}(\text{t}) = 1), \end{aligned} \tag{21}$$

where, $\lambda_k^{(T)}(t)$ and $\lambda_k^{(E)}(t)$ represent weights of time and energy consumption, which are selected according to STs' business requirements. That is, if the ST is in a state of energy shortage, energy consumption $\lambda_k^{(E)}(t)$ should be the main consideration factors and should be set larger, while if the ST performs a delay-sensitive business, $\lambda_k^{(T)}(t)$ should be set larger.

In the following section, we combine Lyapunov theory with proposed DUDA to develop an adaptive offloading mechanism.

VI. DISTRIBUTED UPLINK OFFLOADING MECHANISM

A. *Best Offloading Strategy*

Two salient features make Lyapunov suitable for dynamically changing network environment and assignment of tasks. One is that it corresponds to online stochastic optimization problem and it just utilizes system information at the current time period. The other is that it can joint optimize queue stability and performance objective we pursuing by building a drift-plus-penalty function. Based on the above considerations, this paper adopts Lyapunov theory to obtain best offloading decision-making of each cell. Considering that tasks of terminals need to be partially offloaded to FNs, stability of the access layer is crucial. Therefore, system stability is measured by the change of input queue in FN.

Input queue $Q(t)$ is defined as

$$\begin{aligned} Q(t+1) = \max\big[Q(t) - \textstyle\sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t) \\ + \textstyle\sum_{k=1}^{K}\sum_{m=1}^{M} a_{k,m}(t)B_k(t), 0\big]. \end{aligned} \tag{22}$$

The quadratic Lyapunov function is defined as

$$L(t) = \frac{1}{2}Q^2(t). \tag{23}$$

We can get Lyapunov drift function according to

$$L(t+1) - L(t) = \frac{1}{2}\big[Q^2(t+1) - Q^2(t)\big]. \tag{24}$$

As

$$\begin{aligned} &Q^2(t+1) \\ &= \left(\max[Q(t) - \sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t) + \sum_{k=1}^{K}\sum_{m=1}^{M} a_{k,m}(t)B_k(t)0]\right)^2 \\ &\le Q^2(t) + \left(\sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)\right)^2 \\ &\quad +2Q(t)\left(\left(\sum_{k=1}^{K}\sum_{m=1}^{M} a_{k,m}(t)B_k(t)\right) - \sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)\right), \end{aligned} \tag{25}$$

the drift function can be collated as

$$\begin{aligned} &L(t+1) - L(t) \\ &\le Q(t)\left(\left(\sum_{k=1}^{K}\sum_{m=1}^{M} a_{k,m}(t)B_k(t)\right)\right. \\ &\quad \left. - \sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)) + \frac{1}{2}\left(\sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)\right)\right)^2. \end{aligned} \tag{26}$$

Our objective is to minimize system consumption, thus a drift plus penalty function is built as

$$L(t+1) - L(t) + V\sum_{k=1}^{K} Z_k(t). \tag{27}$$

With result of (26), (27) satisfies

$$\begin{aligned} &L(t+1) - L(t) + V\sum_{k=1}^{K} Z_k(t) \\ &\le Q(t)\left(\sum_{k=1}^{K}\sum_{m=1}^{M} a_{k,m}(t)B_k(t) - \sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)\right) \\ &\quad + \frac{1}{2}\left(\sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)\right)^2 + V\sum_{k=1}^{K} Z_k(t). \end{aligned} \tag{28}$$

As the unit of data queuing length and system consumption is different, V is used to convert between units. With $Z_k(t)$ obtained in Section V and (28), we can get

$$\begin{aligned} &L(t+1) - L(t) + V\sum_{k=1}^{K} Z_k(t) \\ &\le Q(t)\left(\sum_{k=1}^{K}\sum_{m=1}^{M} a_{k,m}(t)B_k(t) - \sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)\right) \\ &\quad + \frac{1}{2}\left(\sum_{k=1}^{K}\sum_{m=1}^{M} F_{k,m}^{(f)}(t)\right)^2 + V\sum_{k=1}^{K} Z_k(t) \end{aligned} \tag{29}$$

$$
\begin{aligned}
=&Q(t)\Bigg(\sum_{k=1}^{K}\sum_{m=1}^{M}a_{k,m}(t)B_k(t)\\
&-\sum_{k=1}^{K}\sum_{m=1}^{M}F_{k,m}^{(f)}(t)\Bigg)+\frac{1}{2}\Bigg(\sum_{k=1}^{K}\sum_{m=1}^{M}F_{k,m}^{(f)}(t)\Bigg)^2\\
&+V\sum_{k=1}^{K}\Bigg[\lambda_k^{(T)}(t)\frac{T_k^{(l)}(t)+T_{k,\text{off}}^{(f)}(t)+T_{k,\text{exe}}^{(f)}(t)}{T_{k,\max}(t)}\\
&+\lambda_k^{(E)}(t)\frac{E_k^{(l)}(t)+E_{k,\text{off}}^{(f)}(t)}{E_{k,\max}(t)}\Bigg]\\
=&\sum_{k=1}^{K}\sum_{m=1}^{M}a_{k,m}(t)\Bigg[V\Bigg(\frac{\lambda_k^{(T)}(t)}{T_{k,\max}(t)}\Bigg(\frac{B_k(t)}{R_{k,m}(t)}+\frac{D_k(t)}{F_{k,m}^{(f)}(t)}\\
&-\frac{D_k(t)}{F_k^{(l)}(t)}\Bigg)+\frac{\lambda_k^{(E)}(t)}{E_{k,\max}(t)}\Bigg(\frac{P_k(t)B_k(t)}{R_{k,m}(t)}-\mu_k(t)D_k(t)\Bigg)\Bigg)\\
&+Q(t)B_k(t)\Bigg]+\sum_{k=1}^{K}V\Bigg(\frac{\lambda_k^{(T)}(t)D_k(t)}{T_{k,\max}(t)F_k^{(l)}(t)}\\
&+\frac{\lambda_k^{(E)}(t)\mu_k(t)D_k(t)}{E_{k,\max}(t)}\Bigg)-Q(t)\sum_{k=1}^{K}\sum_{m=1}^{M}F_{k,m}^{(f)}(t)\\
&+\frac{1}{2}\Bigg(\sum_{k=1}^{K}\sum_{m=1}^{M}F_{k,m}^{(f)}(t)\Bigg)^2. \qquad (30)
\end{aligned}
$$

Find the minimum value of the expression on the right is our goal. In order to simplify problem, we omit the fixed value part at a certain moment, such as:

$$
\sum_{k=1}^{K}V\left(\frac{\lambda_k^{(T)}(t)D_k(t)}{T_{k,\max}(t)F_{k,m}^{(l)}(t)}+\frac{\lambda_k^{(E)}(t)\mu_k(t)D_k(t)}{E_{k,\max}(t)}\right),
$$

$$
Q(t)\sum_{k=1}^{K}\sum_{m=1}^{M}F_{k,m}^{(f)}(t),\ \frac{1}{2}\left(\sum_{k=1}^{K}\sum_{m=1}^{M}F_{k,m}^{(f)}(t)\right)^2.
$$

After simplification, our objective function can be expressed as

$$
\begin{aligned}
\min:&\\
&\sum_{k=1}^{K}\sum_{m=1}^{M}a_{k,m}(t)\Bigg[V\Bigg(\frac{\lambda_k^{(T)}(t)}{T_{k,\max}(t)}\Bigg(\frac{B_k(t)}{R_{k,m}(t)}+\frac{D_k(t)}{F_{k,m}^{(f)}(t)}-\frac{D_k(t)}{F_k^{(l)}(t)}\Bigg)\\
&+\frac{\lambda_k^{(E)}(t)}{E_{k,\max}(t)}\Bigg(\frac{P_k(t)B_k(t)}{R_{k,m}(t)}-\mu_k(t)D_k(t)\Bigg)\Bigg)+Q(t)B_k(t)\Bigg]\\
\text{s.t.}:&\ a_{k,m}(t)\in[0,1]\quad\forall t,k,m. \qquad (31)
\end{aligned}
$$

Since all STs and FNs are independent of each other on offloading decision making, this problem can be further simplified. For ease of description, the factor that removes decision is defined as $X_{k,m}(t)$. It is expressed by

$$
\begin{aligned}
X_{k,m}(t)=&\Bigg[V\Big(\frac{\lambda_k^{(T)}(t)}{T_{k,\max}(t)}\Big(\frac{B_k(t)}{R_{k,m}(t)}+\frac{D_k(t)}{F_k^{(f)}(t)}-\frac{D_k(t)}{F_k^{(l)}(t)}\Big)\\
&+\frac{\lambda_k^{(E)}(t)}{E_{k,\max}(t)}\Big(\frac{P_k(t)B_k(t)}{R_{k,m}(t)}-\mu_k(t)D_k(t)\Big)\Big)+Q(t)B_k(t)\Bigg]. \qquad (32)
\end{aligned}
$$

Based on known items we can calculate the value of $X_{k,m}(t)$. Then, best offloading strategy $a_{k,m}(t)$ can be determined according to

$$
a_{k,m}(t)=\begin{cases}\text{Maximum offloadable ratio},\ X_{k,m}(t)<0\\ 0,\ \ X_{k,m}(t)\geq 0.\end{cases} \qquad (33)
$$

This paper considers task with partial offloading capability. So, if we get $X_{k,m}(t)<0$, we should use the maximum value of $a_{k,m}(t)$ to make the right-side expression be smallest. While in the case of $X_{k,m}(t)\geq 0$, we should make $a_{k,m}(t)=0$. So far, the best offloading strategy for shared terminals in each cell has been obtained.

Algorithm 2: Deviation-based decentralized computation offloading mechanism

```
input          : Network environment, computing capability;
output         : Best offloading decision set of each cell
                 and inter-cell decision updating order;
initialization: $a_{k,m} = 0$;
1  for each user in each time slot do
2      get the best decision set $\Omega_j^*$ of each cell based on last
         section; each cell computes its deviation with
         $O_j(t) = \sum_{k=1}^{K} \frac{|a_{k,m}(t+1)-a_{k,m}(t)|}{a_{k,m}(t)}$;
3      if $O_j(t) \neq 0$ then
4          each cell broadcast its own deviation to others;
5          set time length of decision update as $t/N$;
6          if the deviation that received is higher then
7              the cell should wait $t/N$ for next opportunity;
8          else
9              the cell gets an opportunity to update
                 offloading decision in accordance with
                 $a_{k,m}(t+1) \in \Omega_j^*$;
10         end
11     end
12 end
```

B. Offloading Mechanism

Best offloading strategy for each ST in cells is illustrated in the last subsection. This part integrates offloading update order-making of cells to introduce the whole distributed uplink offloading mechanism.

To minimize the overhead of offloading process, we formulate a mechanism named DUDA to get the order of offloading decision updating between cells. In DUDA, each cell compute its deviation with $O_j(t)=\sum_{k=1}^{K}\frac{|a_{k,m}(t+1)-a_{k,m}(t)|}{a_{k,m}(t)}$, and broadcast its value to other cells. The higher the deviation, the more urgent the need for offloading decision updates as we thought. So, we give chance to the cell with deviation higher than others, while, other cells must wait for the next opportunity.

Our offloading mechanism contains decision-making and update order-making as shown in Algorithm 2.

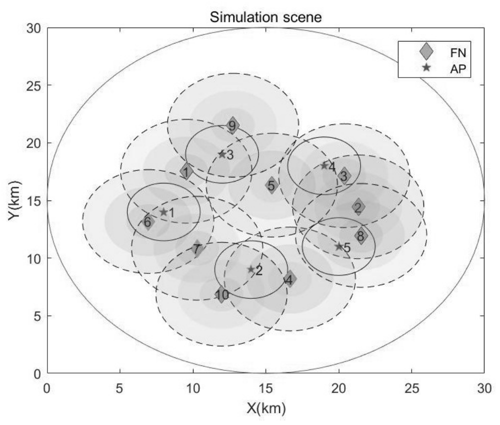

Fig. 5. Simulation scene.

VII. PERFORMANCE ANALYSIS

In order to evaluate the effectiveness of our strategy, we make a time complexity analysis of the whole algorithm. Four sub-algorithms are mentioned and proposed to solve resource allocation and task offloading problem, which are differential game, bipartite graph multiple matching, Lyapunov and DUDA. The complexity of these parts are represented by O_d, O_b, O_L, and O_D, respectively. Since these algorithms are executed in a serial manner, the total time complexity can be expressed as

$$O_d + O_b + O_L + O_D. \tag{34}$$

The above items correspond to time complexity of the four algorithms, respectively. It can be found that the complexity of bipartite graph multiple matching algorithm is related to matching relationship. Specifically, it is determined by the number of terminals and restricted connections of APs. The complexity of Lyapunov is linearly related to the value of V in penalty function. DUDA algorithm is based on the calculation of deviation degree with each terminal and is linearly related to the number of terminals. Therefore, (34) can be expressed as

$$\begin{aligned} &O_d + O_b + O_L + O_D \\ &= O(1) + O\left(K * \sum_{j=1}^{N} \text{connection}(j)\right) + O(V) + O(K) \\ &= O\left(K * \sum_{j=1}^{N} \text{connection}(j)\right) + O(V) + O(K), \end{aligned} \tag{35}$$

where $\text{connection}(j)$ symbols the number of restricted connections of APj.

It can be seen that our strategy can maintain stability as tolerable change in time with system scale expansion.

VIII. SIMULATION CONSTRUCTION AND RESULTS

In this section, we verify effectiveness of our strategy with simulation experiments. Simulation environment construction and simulation results are shown as below.

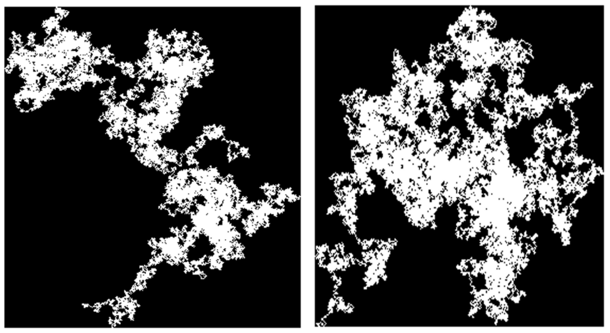

Fig. 6. Random walk path of two shared terminals.

Table 2. Parameters setting.

Simulation area S	d=30 km
Transmit power P_k	100 mW
Background noise ω_k	-100 dBm
Number of CPU cycles D_k	1000 Megacycles
Local computational capability $F_k^{(l)}$	0.7 GHz
Fog computational capability $F_k^{(f)}$	150 GHz
Weight of computational energy $\lambda_k^{(T)}$	0.3,0.7
Weight of computational time $\lambda_k^{(E)}$	$1 - \lambda_k^{(T)}$
Data size B_k	500 KB
Path loss exponent α	4

A. Simulation Environment Parameters

We construct an F-RAN scene with 10 FNs and 5 cells. They are randomly distributed within a circle of 30 km in diameter. The diameters of coverage area belongs to FN and each cell are 9km and 5km, respectively. Each cell with one AP in the center. By recording a random set of data, we describe their positional relationship in Fig. 5. Where, the coverage of FN is represented by three color circles numbered as 1,2,3 from the inside out, and which circle that APs locate in is recorded. The smaller the circle number means the closer the AP is to the FN and the smaller the resource allocation overhead. This data is used for resource allocation decision making in differential game as mentioned before. We assume that if the location of FN and AP is got, they will not change within a time slot. According to Fig. 5, we can get the matching relationship between FN and AP.

500 STs are randomly distributed within the circle of 30 km in diameter. Their bandwidth demand satisfies the uniform distribution between (0,10). To prevent terminal location jumping, we assume that terminals use random walk mode. Specifically, the neighborhood of one terminal is divided into 8 parts, the terminal chooses one neighborhood to walk in the next moment. Corresponding horizontal and vertical coordinate changes is recorded to update terminal location. Taking two terminals as an example, their randomly walking image in 700 time slots are shown in Fig. 6. In each time slot, if the location of the sharing terminal belongs to a certain cell, its demand is added to the corresponding AP. The key parameters used in the simulation are shown in Table 2.

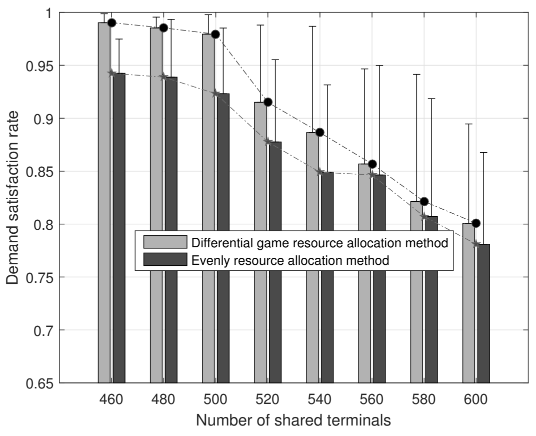

Fig. 7. Demand satisfaction rate between FN-AP with differential game and evenly resource method.

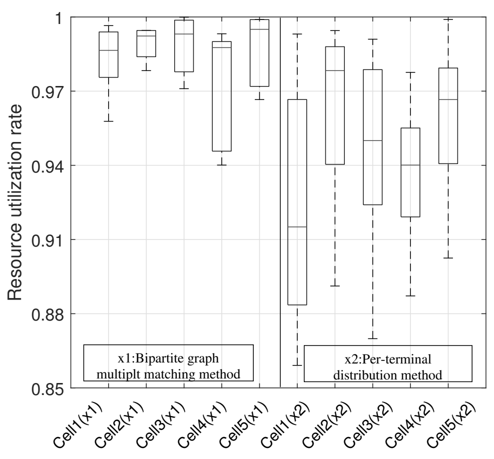

Fig. 8. Resource utilization rate of AP-ST with bipartite graph multiple matching and per-terminal distribution method.

B. Simulation Results

According to the matching relationship, we apply differential game theory proposed in this paper to allocate resources between FN and AP, measuring performance of the algorithm based on AP resource demand satisfaction rate.

We set the number of shared terminals from 460 to 600 with 20 scale division. The resource demand satisfaction rate (the amount of resources obtained/the amount of resources expected) of each cell in 700 consecutive time slots was measured and averaged to obtain the result shown in Fig. 7. The gray part represents performance of our algorithm. While the black part represents the performance of evenly resource allocation method (ERAM). Above the bar graph, stability of the results with standard deviation of the corresponding data set is shown as well. We can see that the performance of our differential game algorithm can better meet the demands when the number of shared terminals < 500. As the number of terminals increases, there will be a decrease in the satisfaction rate. However, as long as the terminal number less than 600, the demand satisfaction rate can be guaranteed to reach 0.8 or more. This algorithm can improve performance by about 5% compared to evenly distributed.

We measure the performance of our method by resource utility (the resources allocated to terminals/the resources obtained by AP from FN) of AP-ST. The simulation result is shown in Fig. 8. From the results we can see that the use of bipartite graph multiple matching method can make the average resource utilization rate of more than 97%, significantly better than per-terminal distribution method.

In each timeslot, task is offloaded according to resource allocation result. System consumption with different weight of computational time are shown in Fig. 9. It can be observed that our algorithm can reduce 30%–60% system consumption by combining resource allocation and uninterrupted time offloading which are lacked consideration in the remaining schemes.

We also test the relationship between offloading part and FN computing capability as well as data size. The test results are shown in Fig. 10. It can be seen that when the computational capability of FN is very low, the system tends to only offload a little part of tasks. Larger part of tasks choose to offload with FN computational capability and data size increases. When the computational capability is high enough, offloading amount will tend to be a constant.

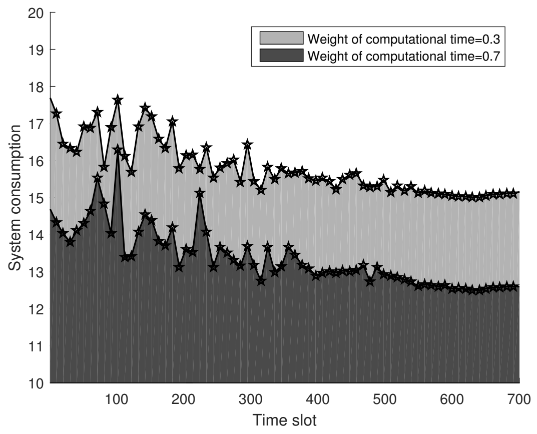

Fig. 9. System consumption with different weight of computational time.

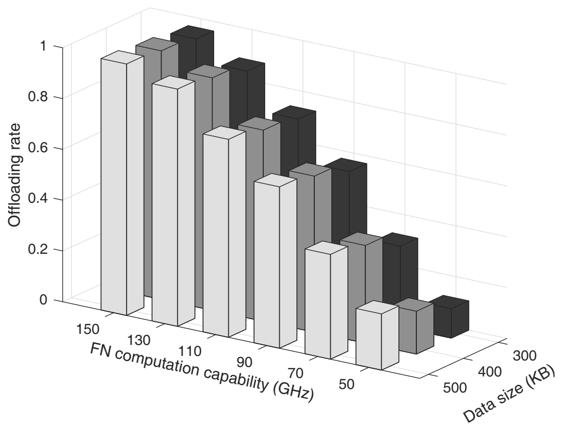

Fig. 10. Offloading rate with FN computation capability and data size changing.

IX. CONCLUSION

In this paper, we construct a shared pattern oriented F-RAN architecture. Two problems are solved based on that. One is formulating a rational mechanism for resource allocation with differential game and bipartite graph multiple matching, and the other is designing computation offloading strategy of tasks based on resource allocation result with Lyapunov and proposed DUDA algorithm. Our strategy provides a systematic, full-process solution for shared services quality promotion with F-RAN. Numerical results demonstrate that the proposed strategy is efficient. For the future work, we will make effort on safety and reliability of fog computing, hoping to improve its communication performance further with security of the transmission process ensured.

REFERENCES

[1] H. Zhang *et al.*, "Computing resource allocation in three-tier IoT fog networks: A joint optimization approach combining Stackelberg game and matching," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1204–1215, Oct. 2017.

[2] L. Ni *et al.*, "Resource allocation strategy in fog computing based on priced timed Petri nets," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1204–1215, Oct. 2017.

[3] H. Zhang *et al.*, "A hierarchical game framework for resource management in fog computing," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 52–57, Aug. 2017.

[4] A. Al-Shuwaili and O. Simeoneand, "Energy-efficient resource allocation for mobile edge computing-based augmented reality applications," *IEEE Wireless Commun. Lett.*, vol. 6, no. 3, pp. 398–401, June 2017.

[5] Y. Mao *et al.*, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sept.2017.

[6] S. Sardellitti, G. Scutari, and S. Barbarossa "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. over Netw.*, vol. 1, no.2, pp. 89–103, June 2015.

[7] X. Meng, W. Wang, and Z. Zhang "Delay-constrained hybrid computation offloading with cloud and fog computing," *IEEE Access* vol. 5, pp. 21355–21367, Sept. 2017.

[8] R. Deng *et al.*, "Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.

[9] TQ. Dinh *et al.*, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

[10] L. Pu *et al.*, "D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3887–3901, Dec. 2016.

[11] C. Wang *et al.*,"Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7432–7445, Aug. 2017.

[12] K. Liang *et al.*,"Joint resource allocation and coordinated computation offloading for fog radio access networks," *China Commun.*, vol.13, no. Supplement2 , pp. 131–139, 2016.

[13] M. Peng, S. Yan, and C. Wang,"Fog-computing-based radio access networks: issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, July-Aug. 2016.

[14] T. Biermann *et al.*, "How backhaul networks influence the feasibility of coordinated multipoint in cellular networks," *IEEE Commun. Mag.*, vol. 51, no. 8, pp. 168–176, Aug. 2013.

[15] L. Yang *et al.*,"A framework for partitioning and execution of data stream applications in mobile cloud computing," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 40, no. 4, pp. 23–32. Mar. 2013.

[16] B. Chun *et al.*,"Clonecloud: Elastic execution between mobile device and cloud," *in Proc. ACM EuroSys*, Apr. 2011, pp. 301–314.

[17] C. Xu,"Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[18] Y. Wen, W. Zhang, and H. Luo,"Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones," in *Proc. IEEE INFOCOM*, May 2012, pp. 2716–2720.

[19] T. X. Tran and D. Pompili,"Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *Comput. Sci.*, May 2017.

Linna Ruan received B.S. degree from Beijing Information Science and Technology University, Beijing, China, in 2016. She is currently studying for a Ph.D. degree at Beijing University of Posts and Telecommunications. Her research interests include access layer issues of edge computing for IoT scenarios, including resource allocation, service computation offloading.

Zhoubin Liu received B.S. degree from Institute of Computer, China University of Petroleum, and M.S. degree from College of Software, Xián Jiaotong University, in 1997 and 2015, respectively. He is currently a Senior Engineer at State Grid Zhejiang Electric Power Research Institute. His research interests include information security, energy Internet, and distributed system.

Xuesong Qiu received the Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2000. He is currently a Professor, a doctoral supervisor, and the deputy director of the State Key Laboratory of networking and switching technology, and vice president of Network Technology Research Institute of Beijing University of Posts and Telecommunications, China. He was editor of the two ITU-T standards and 4 industry standards of China. He has published more than 200 academic papers. He won twice the national scientific and technological progress prize of china. His major research interests include network and service management, information and communication technology of smart grid.

Zixiang Wang received B.S. degree from College of Information Technology, Ningbo University, and Ph.D. degree from College of Electrical Engineering, Zhejiang University, in 2010 and 2015, respectively. He is currently an engineer at State Grid Zhejiang Electric Power Research Institute. His research interests include wireless sensor networks, information fusion, and nonlinear control.

Shaoyong Guo received Ph.D. degree at Beijing University of Posts and Telecommunication. And he received B.S. degree in Information and Computing Science from HeBei University. His research interests include blockchain Application technology, mobile edge computing, and IoT in energy Internet.

Feng Qi is a Professor of Beijing University of Posts and Telecommunications, engaged in scientific research, teaching, and standardization research in information and communications. His research interests include communications software, network management, and business intelligence. He has won 2 National Science and Technology Progress Awards. He has also written more than 10 ITU-T international standards and Industry Standards. Served as vice chairman of the ITU-T Study Group 4 and Study Group 12.