QoS-Aware Fog Computing Resource Allocation using Feasibility-Finding Benders Decomposition

Thai T. Vu, Diep N. Nguyen, Dinh Thai Hoang, Eryk Dutkiewicz

***Abstract*—We investigate a joint offloading and resource allocation under a multi-layer cooperative fog and cloud computing architecture, aiming to minimize the total energy consumption of mobile devices while meeting users' QoS requirements, e.g., delay, security, and application compatibility. Due to the mutual coupling amongst offloading decision and resource allocation variables, the resulting optimization is a mixed integer non-linear programming problem that is NP-hard. Such problem often requires exponential time to find the optimal solution. In this work, we propose a distributed approach, namely feasibility-finding Benders decomposition (FFBD), that decomposes the original problem into a master problem for the offloading decision and subproblems for resource allocation. These (simpler) subproblems can be solved in parallel at fog nodes, thereby reducing both the complexity and the computational time. The numerical results show that the FFBD always returns the optimal solution of the problem with significantly less computation time (e.g., in comparing with the branch-and-bound method).**

Keywords- Fog computing, offloading, resource allocation, QoS, security, latency, MINLP, and Benders decomposition.

I. Introduction

A new network architecture, referred to as mobile edge or fog computing, has recently received paramount interest. The key idea of fog computing is to "move" computing resources closer to mobile users [1]. In a fog network, powerful computing devices, e.g., servers, are deployed at the edges of the mobile network to support hardware resource-constrained devices to perform high-complexity tasks. As a result, the deployment of fog networks can save energy, increase operation time, and enable new applications/services for mobile devices by utilizing powerful resources at the edge. Fog computing can also reduce operating costs for mobile network operators up to 67% by reducing the total throughput and peak backhaul bandwidth consumption [2].

However, unlike public clouds, e.g., Amazon Web Services, a fog node does not possess abundant computing resource. While the number of mobile applications is huge, every fog node can support only a small number of application types with different qualities of service (QoS), e.g., security. From mobile users' perspectives, computation offloading demand also varies in both types and QoS requirements. For example, some applications may require higher security levels (e.g., finance/health applications), whereas others may need lower latency (e.g., games). Moreover, not all computational tasks benefit from being offloaded to the fog node. Some tasks even consume more energy when being offloaded than being processed locally due to the communication overhead, i.e., transmitting requests and receiving results[1]. Given the above, this work considers the joint task offloading and resource allocation optimization problem, aiming to minimize the energy consumption for mobile devices under the fog nodes' resource constraints, and mobile applications' delay, security and compatibility requirements.

The above problem has been visited from different angles and using different approaches over the last few years [3]–[7]. In [3], the offloading decision problem in a two-tier architecture (i.e., the mobile users and the fog nodes) was formulated as a non-cooperative game. To minimize the completion time for each tasks, the authors of [4] developed a game-based distributed algorithm to optimally allocate the computational tasks among nearby devices and the edge cloud. The authors of [5] proposed a decomposition technique to maximize the weighted sum of all users' offloading utilities, which comprises both the energy and delay required for each task. The optimal task offloading problem was formulated and addressed in [6] to minimize the average processing time of each task. Focusing on IoT devices, the authors of [7] aimed at minimizing average transmission energy consumption while guaranteeing the average latency requirement. However, most existing works did not consider offloading computing task to the cloud servers, i.e., considering only the two-tier architecture with a fog server.

As aforementioned, fog nodes are also limited in computing power, in comparison to cloud servers. Therefore, in this paper, we develop a multi-layer cooperative fog network including mobile devices, multiple fog nodes, and a cloud cluster which includes many cloud servers for different application types. Under our architecture, the tasks can be offloaded to either a fog node or a cloud server. To minimize the total energy consumption for mobile users in the network while meeting all tasks requirements, we first formulate the joint task offloading and resource allocation optimization problem for all mobile users and edge nodes. The resulting problem is a mixed integer non-linear programming (MINLP) which is proven to be NP-hard. Its optimal solution can be found by using the improve branch-and-bound (BB) algorithm (IBBA) [8]. However, the intermediate problems of IBBA still have a large size due to mixed offloading and resource allocation variables. Consequently, the computation time of the IBBA is still high. In this work, we propose a distributed approach, namely feasibility-finding Benders decomposition (FFBD), that decomposes the original problem into a master problem for the offloading decision and subproblems for resource allocation. These (simpler) subproblems can be solved in parallel at fog nodes, thus help reduce both the complexity and the computational time. Especially, while other Benders decomposition (BD) methods only work with linear problems, our FFBD directly solves the non-linear ones. Under the FFBD, Benders cutting-planes are generated directly based on the resource limitation of fog nodes and the results of subproblems. The extensive numerical results show that the FFBD method always returns

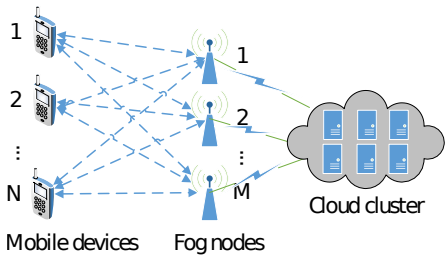

Fig. 1: Three-tier cooperative mobile edge computing network.

the optimal solution of the original problem with significantly less computation time in comparing with the IBBA.

II. System Model and Problem Formulation

A. System Model

Fig. 1 illustrates a three-tier fog computing system with N mobile devices $\mathcal{N} = \{1, \ldots, N\}$, M cooperative fog nodes $\mathcal{M} = \{1, \ldots, M\}$, and one cluster of cloud servers. Mobile devices can run programs belonging to a set of Q applications $\mathcal{Q} = \{1, \ldots, Q\}$. Each program contains independent computing tasks which can be processed locally or offloaded to fog nodes or the cloud cluster. Mobile devices, fog nodes and the cloud servers have different security levels, denoted as $\mathcal{S} = \{1, \ldots, S\}$ [9], in which 1 is the highest security level and S is the lowest one.

At each time slot, mobile user i can request to offload a computing task $I_i \left(D_i^i, D_i^o, C_i, s_i, t_i^r, q_i\right)$, in which D_i^i and D_i^o respectively are the input (including input data and execution code) and output/result data lengths, C_i is the number of CPU cycles that are required to execute the task, and $q_i \in \mathcal{Q}$ is the application type of task I_i. The QoS of task I_i is defined by the delay t_i^r and security requirements $s_i \in \mathcal{S}$. Only mobile device, fog nodes, or the cloud servers satisfying these requirements are eligible to process the task.

1) Local Processing: Device i has a security level $s_i^l \in \mathcal{S}$ and a CPU processing rate f_i^l (cycles per second). We assume that the security level of a mobile device always satisfies its task requirement, i.e., $s_i^l \leq s_i$. If task I_i is processed locally, the necessary time T_i^l to perform the task is given by

$$T_i^l = C_i / f_i^l. \tag{1}$$

Similar to [3], the CPU power consumption rate P_i^l can be modeled by a super-linear function of f_i^l, as $P_i^l = \alpha (f_i^l)^\gamma$, where α and γ are pre-configured parameters depending on the chip architecture. The consumed energy E_i^l of the mobile device for local computation is given by

$$E_i^l = P_i^l T_i^l = \alpha (f_i^l)^{\gamma - 1} C_i. \tag{2}$$

2) Fog Node Processing: A fog node j has capabilities denoted by a tuple $(R_j^u, R_j^d, R_j^f, s_j^f)$ in which R_j^u, R_j^d, R_j^f, and $s_j^f \in \mathcal{S}$ are the total uplink rate, total downlink rate, the CPU cycle rate, and its security level, respectively. If task I_i is processed at fog node j, then this node will allocate bandwidth and computation resources for mobile device i, defined by a tuple $\mathbf{r}_{ij} = (r_{ij}^u, r_{ij}^d, r_{ij}^f)$, in which r_{ij}^u, r_{ij}^d, r_{ij}^f respectively are uplink, downlink, and CPU cycle rates for input, output transmissions, and task execution. In this case, the energy consumption at the mobile user is for both transferring input to and receiving output from the fog node j. The delay includes time for transmitting input, receiving output and task processing at the fog node.

Let e_{ij}^u and e_{ij}^d denote the energy consumption for transmitting and receiving a unit of data, respectively. The consumed energy of mobile device E_{ij}^f and the delay T_{ij}^f are given by:

$$E_{ij}^f = E_{ij}^u + E_{ij}^d, \text{ and } T_{ij}^f = D_i^i / r_{ij}^u + D_i^o / r_{ij}^d + C_i / r_{ij}^f, \tag{3}$$

where $E_{ij}^u = e_{ij}^u D_i^i$ and $E_{ij}^d = e_{ij}^d D_i^o$.

3) Cloud Server Processing: Assume that all fog nodes are connected to the cloud cluster. Let $s_{q_i}^c$ be the security level of the cloud computing toward application type q_i. If task I_i has $s_i \geq s_{q_i}^c$, then a fog node can forward I_i to a suitable cloud server in the cluster. We denote the data rate between a fog node and the cluster as r^{fc}, and the processing rate assigned to each task on the cloud as f^c.

If fog node j forwards task I_i to the cloud cluster, it will allocate resources for mobile device i, defined by a tuple $\mathbf{r}_{ij} = (r_{ij}^u, r_{ij}^d, r_{ij}^f)$, in which r_{ij}^u, r_{ij}^d are uplink rate, downlink rate for input and output transmissions, and $r_{ij}^f = 0$. After receiving the task, fog node j sends the input data to the cloud server for processing, then receives and sends the result back to the mobile user. In this case, the consumed energy E_{ij}^c at the mobile user is only for transmitting input and output data directly to and from fog node j as in the case of fog node processing, while the delay T_{ij}^c includes the time for transmitting the input from mobile user to the fog node, time from the fog node to the cloud cluster, time for receiving the output from the cloud cluster to mobile user via the edge node, and task-execution time at the cloud server. These performance metrics are as follows:

$$E_{ij}^c = E_{ij}^f = E_{ij}^u + E_{ij}^d, \tag{4}$$

and

$$T_{ij}^c = D_i^i / r_{ij}^u + D_i^o / r_{ij}^d + (D_i^i + D_i^o) / r^{fc} + C_i / f^c. \tag{5}$$

Because a fog node can support only some types of mobile applications, let $G(q_i)$ be the set of all fog nodes that can support the application type q_i. Equivalently, $\overline{G}(q_i) = \mathcal{M} / G(q_i)$ is the set of all fog nodes that do not support the application type q_i. In other words, the task $I_i \left(D_i^i, D_i^o, C_i, s_i, t_i^r, q_i\right)$ cannot be processed at any fog node in $\overline{G}(q_i)$.

B. Problem Formulation

We denote the binary offloading decision variable for task I_i by $\mathbf{x}_i = (x_i^l, x_{i1}^f, \ldots, x_{iM}^f, x_{i1}^c, \ldots, x_{iM}^c)$, in which $x_i^l = 1$, $x_{ij}^f = 1$, and $x_{ij}^c = 1$ respectively indicate that task I_i is processed locally at the mobile device, fog node j, the cloud server (via fog node j), respectively. From Eq. (1)–(5), the consumed energy E_i of the mobile user and the delay T_i when task I_i is processed are given as

$$E_i = \mathbf{e}_i^\top \mathbf{x}_i, \text{ and } T_i = \mathbf{h}_i^\top \mathbf{x}_i, \tag{6}$$

where $\mathbf{e}_i = (E_i^l, E_{i1}^f, ..., E_{iM}^f, E_{i1}^c, ..., E_{iM}^c)$ and $\mathbf{h}_i = (T_i^l, T_{i1}^f, ..., T_{iM}^f, T_{i1}^c, ..., T_{iM}^c)$.

Let $\mathbf{e} = (\mathbf{e}_1, \ldots, \mathbf{e}_N)$ and $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$. Then, the total consumed energy of mobile devices is given as

$$E = \mathbf{e}^\top \mathbf{x}. \tag{7}$$

In this paper, we address a joint offloading decision ($\mathbf{x}$) and resource allocation ($\mathbf{r}$) problem that aims to minimize the total energy consumption of all mobile devices under the delay, security, and application compatibility requirements. The problem is formally stated as follows

$$(\mathbf{P}_0) \quad \min_{\mathbf{x},\mathbf{r}} \mathbf{e}^\top \mathbf{x}, \tag{8}$$

s.t.

$$(\mathbf{R}_0)\begin{cases} (\mathcal{C}_1)\ T_i \le t_i^r, \forall i \in \mathcal{N}, \\ (\mathcal{C}_2)\ \sum_{i=1}^{N} r_{ij}^f \le R_j^f, \forall j \in \mathcal{M}, \\ (\mathcal{C}_3)\ \sum_{i=1}^{N} r_{ij}^u \le R_j^u, \forall j \in \mathcal{M}, \\ (\mathcal{C}_4)\ \sum_{i=1}^{N} r_{ij}^d \le R_j^d, \forall j \in \mathcal{M}, \\ r_{ij}^u, r_{ij}^d, r_{ij}^f \ge 0, \forall (i,j) \in \mathcal{N} \times \mathcal{M}, \end{cases} \tag{9}$$

and

$$(\mathbf{X}_0)\begin{cases} (\mathcal{C}_5)\ x_i^l + \sum_{j=1}^{M} x_{ij}^f + \sum_{j=1}^{M} x_{ij}^c = 1, \forall i \in \mathcal{N}, \\ (\mathcal{C}_6)\ x_i^l s_i^l + \sum_{j=1}^{M} x_{ij}^f s_j^f + \sum_{j=1}^{M} x_{ij}^c s_{q_i}^c \le s_i, \forall i \in \mathcal{N}, \\ (\mathcal{C}_7)\ x_{ij}^f = 0, \forall (i,j) \in \mathcal{N} \times \overline{G}(q_i), \\ x_i^l, x_{ij}^f, x_{ij}^c \in \{0,1\}, \forall (i,j) \in \mathcal{N} \times \mathcal{M}. \end{cases} \tag{10}$$

where $(\mathcal{C}_1), (\mathcal{C}_6)$ and $(\mathcal{C}_7)$, respectively, are the delay, security and application compatibility requirements of tasks, $(\mathcal{C}_2), (\mathcal{C}_3)$ and $(\mathcal{C}_4)$ are resource constraints at fog nodes, and $(\mathcal{C}_5)$ is offloading decision constraints.

III. Proposed Optimal Solutions

The problem $(\mathbf{P}_0)$ is NP-hard (the proof is omitted for brevity, interested readers are referred to the technical report [8]). Hence, it may take standard optimization solvers exponential time. We observe that by relaxing its binary variables to real numbers $x_i^l, x_{ij}^f, x_{ij}^c \in [0,1], \forall (i,j) \in \mathcal{N} \times \mathcal{M}$, both the objective function and the constraints in $(\mathbf{P}_0)$ are the sum of linear and linear-fractional functions of the forms x, r, and x/r with positive coefficients. Thus, they are either convex or concave functions with respect to $\mathbf{x}$ and $\mathbf{r}$. The relaxing problem is hence a convex optimization problem [10]. In the sequel, using on this characteristic, we propose an effective approach to solve the problem $(\mathbf{P}_0)$.

A. Feasibility-Finding Benders Decomposition

Due to the multiple non-linear constraints in $(\mathbf{P}_0)$, the Benders decomposition (BD) method in [11] is inapplicable. Additionally, this linearization method with the dual multipliers faces the zig-zagging issue [12] which increases the computation time. This method also has difficulty in achieving the optimal solution even though the lower bound is close to the upper bound. Thus, we introduce a distributed algorithm named Feasibility-Finding Benders decomposition (FFBD) as illustrated in Fig. 2. The key point of FFBD is the generation of *Benders cuts* that exclude superfluous solutions, based on the set theory. This completely differs from the work [11], where the Benders cuts are created by solving the dual problem. Due to the finite number of Benders cuts, the FFBD is alway return the optimal solution after a limited number of iterations.

Specifically, we first decompose $(\mathbf{P}_0)$ into a master problem $(\mathbf{MP}_0)$ for the offloading decision and a subproblem $(\mathbf{SP}_0)$ for the resource allocation. Then, FFBD algorithm finds the optimal solution of $(\mathbf{P}_0)$ by iteratively solving $(\mathbf{MP}_0)$ and $(\mathbf{SP}_0)$ at either the cloud server or fog nodes.

$$(\mathbf{MP}_0) \quad \min_{\mathbf{x} \in \mathbf{X}_0} \{\mathbf{e}^\top \mathbf{x} | cuts^{(k)}\}, \tag{11}$$

and

$$(\mathbf{SP}_0) \quad \min_{\mathbf{x}^{(k)}, \mathbf{r} \in \mathbf{R}_0} \{0\}, \tag{12}$$

where $\{0\}$ is the zero constant function, Benders cutting-planes $cuts^{(k)}$ are restrictions on integer offloading solution $\mathbf{x}^{(k)}$ of $(\mathbf{MP}_0)$ at iteration (k).

From Eq. (11) and (12), the cost function of $(\mathbf{P}_0)$ is identical with that of $(\mathbf{MP}_0)$. $(\mathbf{SP}_0)$ only verifies if the integer offloading solution $\mathbf{x}^{(k)}$ of $(\mathbf{MP}_0)$ leads to a feasible resource allocation solution $\mathbf{r}$. Theorem 1 shows that the iteration can stop when a feasible solution $(\mathbf{x}, \mathbf{r})$ is found or $(\mathbf{MP}_0)$ is infeasible.

Theorem 1. *At any iteration (k), if a feasible solution $(\mathbf{x})$ of $(\mathbf{MP}_0)$ leads to a feasible solution $(\mathbf{r})$ of $(\mathbf{SP}_0)$. Then, $(\mathbf{x}, \mathbf{r})$ is the optimal solution of the original problem $(\mathbf{P}_0)$.*

At any iteration (k), if the master problem $(\mathbf{MP}_0)$ is infeasible, then the original problem $(\mathbf{P}_0)$ is infeasible.

Proof: The detailed proof is presented in [8].

B. Distributed Subproblems

With fixed offloading decisions $\mathbf{x}^{(k)}$, $(\mathbf{SP}_0)$ is equivalently divided into M independent resource allocation subproblems of M fog nodes. Without loss of generality, we assume $\mathcal{N}_j^t$ and $\mathcal{N}_j^s$, respectively, be the sets of tasks to be processed at fog node j and at the cloud server via fog node j. Here, $\mathcal{N}_j^t$ and $\mathcal{N}_j^s$ are equivalently determined by two offloading decision variables $\mathbf{x}_j^{f(k)} = (x_{1j}^f, \ldots, x_{Nj}^f)^{(k)}$ and $\mathbf{x}_j^{c(k)} = (x_{1j}^c, \ldots, x_{Nj}^c)^{(k)}$ in $\mathbf{x}^{(k)}$. We can write $\mathcal{N}_j^t = \{1, \ldots, t\}$, $\mathcal{N}_j^s = \{t+1, \ldots, t+s\}$, and $\mathcal{N}_j^{t+s} = \mathcal{N}_j^t \cup \mathcal{N}_j^s = \{1, \ldots, t+s\}$ is defined by $\mathbf{x}_j^{(k)} = (\mathbf{x}_j^{f(k)}, \mathbf{x}_j^{c(k)})$. Variable $\mathbf{r}_j = (\mathbf{r}_{1j}, \ldots \mathbf{r}_{(t+s)j})$ denotes resource allocation of fog node j towards its assigned set of tasks $\mathcal{N}_j^{t+s}$. The resource allocation problem at fog node j can be defined as

$$(\mathbf{SP}_1) \quad \min_{\mathbf{x}_j^{(k)}, \mathbf{r}_j \in \mathbf{R}_j} \{0\}, \tag{13}$$

where

$$(\mathbf{R}_j)\begin{cases} (\mathcal{C}_{1j})\ T_i \le t_i^r, \forall i \in \mathcal{N}_j^{t+s}, \\ (\mathcal{C}_{2j})\ \sum_{i \in \mathcal{N}_j^t} r_{ij}^f \le R_j^f, \\ (\mathcal{C}_{3j})\ \sum_{i \in \mathcal{N}_j^{t+s}} r_{ij}^u \le R_j^u, \\ (\mathcal{C}_{4j})\ \sum_{i \in \mathcal{N}_j^{t+s}} r_{ij}^d \le R_j^d, \\ r_{ij}^f, r_{ij}^u, r_{ij}^d \ge 0, \forall i \in \mathcal{N}_j^{t+s}, \\ r_{ij}^f = 0, \forall i \in \mathcal{N}_j^s. \end{cases} \tag{14}$$

All subproblems $(\mathbf{SP}_1)$ can be solved distributedly among fog nodes in cooperation with the cloud server for $(\mathbf{MP}_0)$. Besides, all these subproblems $(\mathbf{SP}_1)$ can be solved in parallel.

At iteration (k), if $(\mathbf{SP}_1)$ is feasible at every fog nodes, then $\mathbf{x}^{(k)}$ and $\mathbf{r} = (\mathbf{r}_1, \ldots, \mathbf{r}_M)$ are optimal solution of $(\mathbf{P}_0)$.

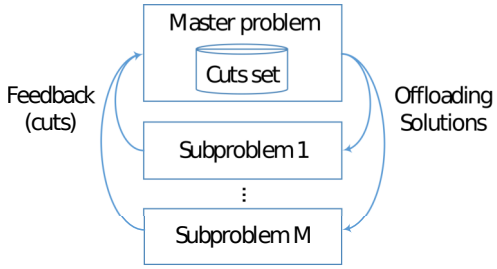

Fig. 2: Feasibility-Finding Benders Decomposition.

Otherwise, if $(\mathbf{SP}_1)$ is infeasible at fog node j, a new cutting-plane $c_j^{(k)}$ will be added to the cut set of $(\mathbf{MP}_0)$ for the next iteration: $cuts^{(k+1)} = cuts^{(k)} \cup c_j^{(j)}$. The following section develops theoretical analysis which helps to improve the efficiency of FFBD algorithm.

C. Fast Feasibility and Infeasibility Detection

From Eq. (3) and (5), the delay constraints $(\mathcal{C}_{1j})$ $T_i \leq t_i^r$ in $(\mathbf{R}_j)$ of $(\mathbf{SP}_1)$ can be rewritten as

$$\begin{cases} \left(\frac{D_i^i}{r_{ij}^u} + \frac{D_i^o}{r_{ij}^d} + \frac{C_i}{r_{ij}^f}\right) \leq t_i^r, & \forall i \in \mathcal{N}_j^t \\ \left(\frac{D_i^i}{r_{ij}^u} + \frac{D_i^o}{r_{ij}^d}\right) \leq t_i^r - \left(\frac{(D_i^i+D_i^o)}{r^{fc}} + \frac{C_i}{f^c}\right), & \forall i \in \mathcal{N}_j^s. \end{cases} \tag{15}$$

Remarkably, the component $\left(\frac{(D_i^i+D_i^o)}{r^{fc}} + \frac{C_i}{f^c}\right)$ is a constant. If $\exists i \in \mathcal{N}_j^s, t_i^r - \left(\frac{(D_i^i+D_i^o)}{r^{fc}} + \frac{C_i}{f^c}\right) \leq 0$, then processing task i at the cloud server does not satisfy its delay requirement $T_i \leq t_i^r$. In other words, $(\mathbf{SP}_1)$ is infeasible. A Benders cut to prevent offloading task i to the cloud server can be created directly for this case. Otherwise, if $t_i^r - \left(\frac{(D_i^i+D_i^o)}{r^{fc}} + \frac{C_i}{f^c}\right) > 0, \forall i \in \mathcal{N}_j^s$, then we define the relative size $(D_i^{i'}, D_i^{o'}, C_i')$ of task i as below.

$$\begin{cases} \left(\frac{D_i^i}{t_i^r}, \frac{D_i^o}{t_i^r}, \frac{C_i}{t_i^r}\right), & \forall i \in \mathcal{N}_j^t \\ \left(\frac{D_i^i}{\left(t_i^r - \frac{(D_i^i+D_i^o)}{r^{fc}} - \frac{C_i}{f^c}\right)}, \frac{D_i^o}{\left(t_i^r - \frac{(D_i^i+D_i^o)}{r^{fc}} - \frac{C_i}{f^c}\right)}, 0\right), & \forall i \in \mathcal{N}_j^s. \end{cases} \tag{16}$$

Let $\beta_i = \left(\frac{D_i^{i'}}{r_{ij}^u} + \frac{D_i^{o'}}{r_{ij}^d} + \frac{C_i'}{r_{ij}^f}\right)$ be the satisfaction rate of Task i. The delay constraint in Eq. (15) becomes

$$\beta_i = \left(\frac{D_i^{i'}}{r_{ij}^u} + \frac{D_i^{o'}}{r_{ij}^d} + \frac{C_i'}{r_{ij}^f}\right) \leq 1, \forall i \in \mathcal{N}_j^{t+s}. \tag{17}$$

THEOREM 2. *Define balancing rates* $\beta_{bal}^u = \frac{\sum_{i \in \mathbb{N}_j^{t+s}} D_i^{i'}}{R_j^u}$, $\beta_{bal}^d = \frac{\sum_{i \in \mathbb{N}_j^{t+s}} D_i^{o'}}{R_j^d}$, *and* $\beta_{bal}^f = \frac{\sum_{i \in \mathbb{N}_j^{t+s}} C_i'}{R_j^f}$. *If* $\beta_{bal} = \beta_{bal}^u + \beta_{bal}^d + \beta_{bal}^f \leq 1$, *then the problem* $(\mathbf{SP}_1)$ *is feasible.*

Proof. We need to show a feasible solution of $(\mathbf{SP}_1)$.

Task I_i will be allocate resources r_{ij}^u, r_{ij}^d and r_{ij}^f as $r_{ij}^u = \frac{D_i^{i'}}{\beta_{bal}^u}$, $r_{ij}^d = \frac{D_i^{o'}}{\beta_{bal}^d}$, and $r_{ij}^f = \frac{C_i'}{\beta_{bal}^f}$. We have $\beta_i = \frac{D_i^i}{r_{ij}^u} + \frac{D_i^{o'}}{r_{ij}^d} + \frac{C_i'}{r_{ij}^f} = \left(\beta_{bal}^u + \beta_{bal}^d + \beta_{bal}^f\right)$. Here, $r_{ij}^f = 0$ and $\frac{C_i'}{r_{ij}^f} = 0, \forall i \in \mathcal{N}_j^s$. Thus, $\beta_i = \beta_{bal} \leq 1, \forall i \in \mathcal{N}_j^{t+s}$.

Besides, $\sum_{i \in \mathcal{N}_j^{t+s}} r_{ij}^u = R_j^u$, $\sum_{i \in \mathcal{N}_j^{t+s}} r_{ij}^d = R_j^d$ and $\sum_{i \in \mathcal{N}_j^{t+s}} r_{ij}^f = R_j^f$ satisfying resource limit conditions. In conclusion, the problem $(\mathbf{SP}_1)$ is feasible. ■

THEOREM 3. *If* $\frac{\sum_{i \in \mathcal{N}_j^{t+s}} D_i^{i'}}{R_j^u} > 1$ *or* $\frac{\sum_{i \in \mathcal{N}_j^{t+s}} D_i^{o'}}{R_j^d} > 1$ *or* $\frac{\sum_{i \in \mathcal{N}_j^{t+s}} C_i'}{R_j^f} > 1$, *then the problem* $(\mathbf{SP}_1)$ *is infeasible.*

Proof: The detailed proof is presented in [8].

D. Cutting-Plane Generation

Here, we introduce three types of cutting-planes which will be updated in $(\mathbf{MP}_0)$.

Resource Cutting-Plane: From Theorem 3, it dictates that every subset of tasks $\mathcal{N}_j \subseteq \mathcal{N}$ must not violate the uplink, downlink and computation resources constraints at every fog node j. Let $\mathbf{c}_j^{u(fog)} = (D_1^{i'}, \ldots, D_N^{i'})/R_j^u$, $\mathbf{c}_j^{d(fog)} = (D_1^{o'}, \ldots, D_N^{o'})/R_j^d$ and $\mathbf{c}_j^{f(fog)} = (C_1', \ldots, C_N')/R_j^f$. Here, $(D_i^{i'}, D_i^{o'}, C_i')$ is calculated as in Eq. (16) for $i \in \mathcal{N}_j^t$.

Let $\mathbf{c}_j^{u(cloud)} = (D_1^{i'}, \ldots, D_N^{i'})/R_j^u$, $\mathbf{c}_j^{d(cloud)} = (D_1^{o'}, \ldots, D_N^{o'})/R_j^d$ and $\mathbf{c}_j^{f(cloud)} = (C_1', \ldots, C_N')/R_j^f$. Here, $(D_i^{i'}, D_i^{o'}, C_i')$ is calculated as in Eq. (16) for $i \in \mathcal{N}_j^s$.

To eliminate all the subsets of $\mathcal{N}$ which violates the resources constraints at edge node j, we add three following Benders cuts into $cuts$ set of the Master problem $(\mathbf{MP}_0)$: $c_j^u = \{\mathbf{c}_j^{u(fog)\top}\mathbf{x}_j^f + \mathbf{c}_j^{u(cloud)\top}\mathbf{x}_j^c \leq 1\}$, $c_j^d = \{\mathbf{c}_j^{d(fog)\top}\mathbf{x}_j^f + \mathbf{c}_j^{d(cloud)\top}\mathbf{x}_j^c \leq 1\}$, and $c_j^f = \{\mathbf{c}_j^{f(fog)\top}\mathbf{x}_j^f \leq 1\}$.

Subproblem Cutting-Plane: At iteration (k), fog node j is assigned a set of tasks $\mathcal{N}_j^{t+s} = \mathcal{N}_j^t \cup \mathcal{N}_j^s$, which is defined by offloading decision $\mathbf{x}_j^{(k)} = (\mathbf{x}_j^{f(k)}, \mathbf{x}_j^{c(k)})$.

If $(\mathbf{SP}_1)$ at fog node j is infeasible, then any resource allocation problem at edge node j with assigned tasks $\mathcal{N}_j \supseteq \mathcal{N}_j^{t+s}$ is infeasible. Thus, to eliminate the all subproblems at edge node j containing $\mathcal{N}_j^{t+s}$, a new Benders cut $c_j^{(k)}$ is added into $cuts$ set of the Master problem $(\mathbf{MP}_0)$ after iteration (k).

$$c_j^{(k)} = \{\mathbf{x}_j^{f(k)\top}\mathbf{x}_j^f + \mathbf{x}_j^{c(k)\top}\mathbf{x}_j^c \leq t + s - 1\}$$

Prefixed Decision Cutting-Plane: If task I_i satisfies $E_i^l < E_{ij}^f$ and $T_i^l \leq t_i^r$, then it can be pre-decided as local processing. As mentioned in *Fast Feasibility and Infeasibility Detection*, if $\left(t_i^r - \frac{(D_i^i+D_i^o)}{r^{fc}} - \frac{C_i}{f^c}\right) \leq 0$, then task I_i could not be processed at the cloud cluster. In these cases, the suitable cutting-planes can be created and added to set $cuts$ of $(\mathbf{MP}_0)$.

E. FFBD Algorithm

Using three types of cutting-planes above, the distributed FFBD, Algorithm 1, is summarized as below.

- **Initialization:** Set the iterator $k = 1$. Then, initialize $cuts^{(k)}$ in $(\mathbf{MP}_0)$ with $3M$ resource cutting-planes as in *Resource Cutting-Plane*: $cuts^{(k)} = \bigcup_{j=1}^{M}\{c_j^u, c_j^d, c_j^f\}$. Other Benders cuts, as in *Prefixed Decision Cutting-Plane*, are also added to the $cuts^{(k)}$ of $(\mathbf{MP}_0)$.
- **Master Problem:** At iteration (k), $(\mathbf{MP}_0)$ is solved to find $\mathbf{x}^{(k)} \in X_0$ satisfying $cuts^{(k)}$. Here, $\mathbf{x}^{(k)}$ defines M

Algorithm 1: FFBD Algorithm

Input : Sets $\mathcal{N}, \mathcal{M}, \mathcal{Q}, \mathcal{S}$, $\{I_i\}$, $\{(R_j^u, R_j^d, R_j^f, s_j^f)\}$
Cloud cluster r^{fc}, f^c, $\{s_{q_i}^c\}$
Output: Optimal solution $(\mathbf{x}, \mathbf{r})$ of Problem $(\mathbf{P}_0)$

begin
  Initialize k and $cuts^{(k)}$ as in **Initialization**.
  while solution $(\mathbf{x}, \mathbf{r})$ has not been found **do**
    $\mathbf{x} \leftarrow$ Solve $(\mathbf{MP}_0)$ with $cuts^{(k)}$ as in **Master Problem**. ▷ $\mathbf{x}$ store solution $\mathbf{x}^{(k)}$ at iteration k
    if $\mathbf{x}$ is feasible **then**
      Based on $\mathbf{x}$, create M subproblems $(\mathbf{SP}_1)$ with asigned tasks $\mathcal{N}_1^{t+s}, \ldots, \mathcal{N}_M^{t+s}$.
    end
    else
      Return Problem $(\mathbf{P}_0)$ is infeasible.
    end
    for $(j = 1;\ j \le M;\ j = j + 1)$ **do**
      $\mathbf{r}_j \leftarrow$ Solve $(\mathbf{SP}_1)$ at fog node j with task set $\mathbb{N}_j^{t+s}$ as in **Subproblems**.
      if $\mathbf{r}_j$ is infeasible **then**
        Add a new Benders cut $c_j^{(k)}$ into $cuts^{(k+1)}$ as in **Subproblems**.
      end
    end
    if $\mathbf{r} = (\mathbf{r}_1 \cup \mathbf{r}_2 \cup \cdots \cup \mathbf{r}_M)$ is feasible **then**
      Return $\mathbf{x}$ and $\mathbf{r}$ ▷ Solution is found
    end
    $k \leftarrow (k + 1)$ ▷ Increase iteration index
  end
end

subproblems of the form $(\mathbf{SP}_1)$. If $(\mathbf{MP}_0)$ is infeasible, then FFBD is terminated with the infeasibility of $(\mathbf{P}_0)$.

- **Subproblems:** At iteration (k), fog node j independently solves $(\mathbf{SP}_1)$ toward its own assigned N_j^{t+s} tasks. Before calling a solver, Theorem 2 is used to check its feasibility. If $(\mathbf{SP}_1)$ is infeasible, then a new Benders cut $c_j^{(k)}$ as in *Subproblem Cutting-Plane* is created and added into $cuts^{(k+1)}$ of $(\mathbf{MP}_0)$ for the next iteration $(k+1)$: $cuts^{(k+1)} = cuts^{(k)} \cup \{c_j^{(k)}\}$.

In Algorithm 1, using Theorem 3 the resource cutting-planes are created at the initial stage of $(\mathbf{MP}_0)$. Thus, the subproblems violating Theorem 3 are prevented during iterations. Consequently, the computation time of FFBD is reduced.

IV. Performance Evaluation

We set up a fog network with all parameters given in Table I. There are 5 types of mobile applications $\mathcal{Q} = \{1, \ldots, 5\}$ and 3 security levels $\mathcal{S} = \{1(\text{High}), 2(\text{Medium}), 3(\text{Low})\}$. In the IoT ecosystem, mobile applications often have different characteristics in term of tasks' data size and complexity. Therefore, it is reasonable to choose randomly data size and complexity. We denote $U(a, b)$ as the descrete uniform distribution between a and b. Here, N tasks $I_i\left(D_i^i, D_i^o, C_i, s_i, t_i^r, q_i\right)$ are generated as $D_i^i \sim U(1.0, 10.0)$MB, $D_i^o \sim U(0.1, 1.0)$MB,

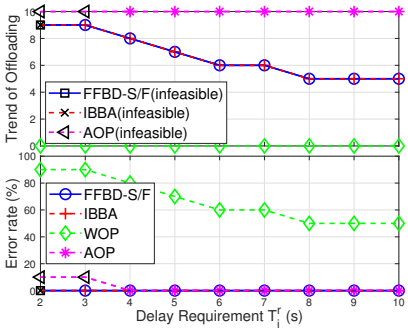

Fig. 3: Trend of offloading and error rate as the delay requirement is relaxed.

$C_i \sim U(0.1, 6.0) \times D_i^i$ Giga cycles, $s_i \sim \mathcal{S}$, $q_i \sim \mathcal{Q}$, and t_i^r varying between $(2, 10)$s. We assume that each fog node can support randomly 3 in 5 mobile applications, and the cluster of cloud servers can support all application types $\mathcal{Q}$.

TABLE I: Experimental parameters

Parameters	Value
Number of mobile devices N	10
Number of fog nodes M	4
CPU rate of mobile devices f_i^l	0.5 Giga cycles/s
Security level of mobile devices s_i^l	1(High)
Energy consumption model of devices (α, γ)	$(10^{-11}\text{Watt/cycle}^2, 2)$
Unit transmission energy consumption e_{ij}^u	0.142 J/Mb
Unit receiving energy consumption e_{ij}^d	0.142 J/Mb
Uplink data rate of each fog node R_i^u	72 Mbps
Downlink data rate of each fog node R_i^d	72 Mbps
Processing rate of each fog node R_j^f	10 Giga cycles/s
Security level of each fog node s_j^f	$\sim \mathcal{S} = \{1, \ldots, S\}$
CPU rate provided by cloud servers f^c	10 Giga cycles/s
Data rate between FNs and cloud servers r^{fc}	5 Mbps
Security level of clouds towards application q_i: $s_{q_i}^c$	$\sim \mathcal{S} = \{1, \ldots, S\}$

Here, we refer the policy in which all tasks are processed locally as "Without Offloading" (WOP), and the policy in which all tasks are offloaded to the fog nodes or the cloud server then minimized the average delay of all tasks as the "All Offloading" (AOP). The improved branch-and-bound algorithm (IBBA) in [8] is used to compare with the FFBD. To evaluate the efficiency of theoretical proposals, we develop two variants of FFBD, FFBD-S for the case using the standard MOSEK solver only, and FFBD-F for the case first using the fast solution detection method described in Theorem 2. Both FFBD-F/S and IBBA are implimented using the Optimizer API of **MOSEK** solver [13]. The results obtained by FFBD-S/F will be compared with the policies IBBA, WOP, and AOP.

In this paper, we study the impact of task delay requirements on the energy consumption of mobile devices and the computation time of the proposed methods. After creating the data set, we detect that there are 5 tasks receiving benefits from offloading $(E_i^l > E_i^f)$, and all tasks have local delay between 2s and 24s. Thus, during experiments, the delay requirement t_i^r of all tasks is varied between 2s and 10s.

Fig. 3 shows the trends of offloading tasks and error rates. Generally, while the trends of WOP and AOP are constants, i.e., 0 and 10, respectively, the offloading trends (to either fog nodes or cloud servers) of FFBD-S/F and IBBA methods

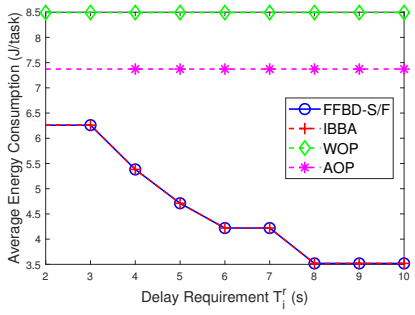

Fig. 4: Average consumed energy at mobile devices when the delay requirement is looser.

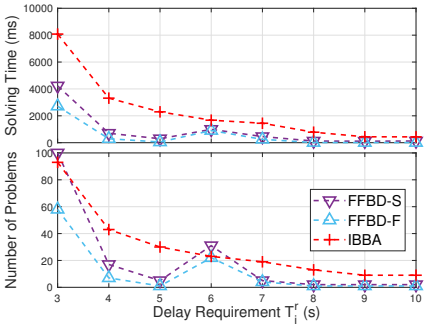

Fig. 5: Computation time and number of solved intermediate problems in order to find an optimal solution when the delay requirement is looser.

decrease from 9 to 5 tasks since the delay requirement is relaxed. Specifically, at first some tasks without offloading benefits still have to be offloaded due to their high local processing delay ($T_i^l > t_i^r$). Then, when t_i^r is larger, these tasks will be executed locally to reduce the consumed energy if $T_i^l \leq t_i^r$. Noticeably, the fog network has not enough resources to accept all offloaded tasks satisfying QoS constraints if delay requirement $t_i^r \leq 3$. Hence, it is infeasible for FFBD-S/F and IBBA at $t_i^r = 2$s and for AOP at $t_i^r = 2$s and 3s. Since $t_i^r \geq 8$s, FFBD-S/F and IBBA return the optimum solution with only 5 offloaded tasks, which get benefit from offloading. From Fig. 3, while the error rate (the proportion of tasks with unsatisfied QoS requirements) of WOP steadily decreases from 90% to 50%, it is zero for other methods at all feasible points. Additionally, at $t_i^r = 3$s, AOP is infeasible, but it is equivalent to the error rate of 10% since FFBD-S/F and IBBA are feasible with 9 offloading tasks.

The offloading trends completely match with the average energy consumption depicted in Fig. 4. Generally, while it is a constant for both WOP with 8.5J/task and AOP with 7.4J/task, the consumed energy in FFBD-S/F and IBBA decreases from 6.3J/task to 3.5J/task when increasing the delay requirement. Equivalently, both FFBD-S/F and IBBA methods reduce the consumed energy from 15% and 26% to 52% and 59%, respectively, in comparing with the AOP and WOP.

Fig. 5 shows the computation time and the number of intermediate problems being solved since the delay requirement is looser. For FFBD-F/S, the intermediate problems are either the master problem or subproblems solved by the standard solver during iterations. The subproblems solved by the fast feasible detection method as in Theorem 2 are ignored. For IBBA, intermediate problems are the relaxing ones solved during travelling the search tree. Here, the WOP and AOP are ignored due to their inadequacy of the goal. Generally, the computation time is proportional to the number of intermediate problems. The FFBD-S/F methods are effective due to their decomposition and initial Benders cuts based on Theorem 3 and the cutting-plane generation from the results of subproblems. The FFBD-F also has better results than that of the FFBD-S due to the implementation of the fast solution finding method described in Theorem 2. Especially, in the worst case, the FFBD-F, FFBD-S and IBBA algorithms, respectively, have the solving time of 2.7s, 4.2s and 8.1s equivalent to 58, 100 and 93 intermediate subproblems at $t_i^r = 3$s.

V. Conclusion

We have proposed the joint offloading and resource allocation problem for three-tier fog computing focusing on the QoS of offloading tasks. To find the optimal solution, we have developed the distributed method, namely FFBD, with two variants FFBD-S/F based on the Benders decomposition algorithm. The FFBD-F implemented with the fast feasible detection method is the fastest algorithm in comparing with the FFBD-S and IBBA. Numerical results have demonstrated the efficiency in terms of energy consumption reduction of the proposed solution.

References

[1] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 2017.

[2] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.

[3] X. Chen, *et al.*, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, 2016.

[4] S. Jošilo and G. Dán, "Decentralized algorithm for randomized task allocation in fog computing systems," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 85–97, 2019.

[5] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, 2019.

[6] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, 2018.

[7] Y. Chen, *et al.*, "Energy efficient dynamic offloading in mobile edge computing for internet of things," *IEEE Trans. Cloud Comput.*, pp. 1–1, 2019.

[8] T. T. Vu, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Optimal Task Offloading and Resource Allocation for Fog Computing," *Technical report*, 2019. [Online]. Available: https://arxiv.org

[9] H. El-Sayed, *et al.*, "Edge of things: The big picture on the integration of edge, iot and the cloud in a distributed computing environment," *IEEE Access*, vol. 6, pp. 1706–1717, 2018.

[10] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[11] Y. Yu, *et al.*, "Green fog computing resource allocation using joint benders decomposition, dinkelbach algorithm, and modified distributed inner convex approximation," in *IEEE ICC 2018*, 2018, Conference Proceedings, pp. 1–6.

[12] M. Fischetti, *et al.*, "Benders decomposition without separability: A computational study for capacitated facility location problems," *EUR J OPER RES*, vol. 253, no. 3, pp. 557–569, 2016.

[13] E. D. Andersen and K. D. Andersen, "The mosek documentation and api reference," Report, 2019. [Online]. Available: https://www.mosek.com/documentation/