Novel Transform-Based Optimization for Resource Allocation and Task Offloading in Communication Networks

Yitong Wang and Jun Zhao, *Member, IEEE*, Liangxin Qian, and Chang Liu

***Abstract*—Fractional Programming (FP) and Multiplicative Programming (MP) are fundamental methodologies widely employed in solving non-convex optimization problems prevalent in wireless communication and edge computing networks. Prior work introduced a remarkable method to solve non-convex functions with multiple ratios by deriving novel tight upper bounds, which we formalize as the `UpperBound` transform in this paper. However, the `UpperBound` transform faces critical limitations, particularly when directly extended to MP problems involving non-negative functions or discrete optimization variables. In this paper, we introduce a generalized transform termed the `UP` transform to overcome these limitations. We rigorously prove that the `UP` transform guarantees convergence to a Karush-Kuhn-Tucker (KKT) point for a broader class of MP problems, including scenarios where variables can be zero or discrete. We comprehensively illustrate the `UP` transform's utility through two practical applications: partial task offloading in mobile edge computing, optimizing computation and energy efficiency; and user association coupled with resource allocation in heterogeneous networks, addressing mixed discrete-continuous optimization challenges. Comparative evaluations against conventional methods demonstrate superior convergence speed, efficiency, solution quality, and reduced computational complexity of the proposed `UP` transform based algorithms.**

***Index Terms*—Fractional programming, multiplicative programming, non-convex optimization, wireless networks, edge computing.**

I. Introduction

FRACTIONAL programming (FP), which solves the optimization problem whose objective and constraint functions consist of ratio term(s), is a critical tool in developing current wireless networks and enhancing the performance of communication services [1]. In practical communication and networking systems, FP is particularly suitable for ratio-based optimization objectives, such as maximizing resource utilization efficiency (e.g., utility-to-cost ratio, throughput-to-power ratio). General FP optimization problem is formulated as follows with $A_n(\boldsymbol{x}) \geq 0$, $B_n(\boldsymbol{x}) > 0$, and $G(\boldsymbol{x})$ being scalar functions of a variable $x \in \mathbb{R}^M$, for $n = 1, 2, \ldots, N$, where N is a positive integer:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & K(\boldsymbol{x}) \stackrel{\text{def}}{=} G(\boldsymbol{x}) + \sum_{n=1}^{N} \frac{A_n(\boldsymbol{x})}{B_n(\boldsymbol{x})} \\ \text{subject to} \quad & \boldsymbol{x} \in \mathcal{X}, \end{aligned} \tag{1}$$

where $\mathcal{X} \subseteq \mathbb{R}^M$ is a closed feasible set. The objective function in (1) is typically non-convex of the optimization variable $\boldsymbol{x}$ due to the sum of the N ratios in (1), as explained in [2], [3]. To derive effective resource allocation schemes of communication networks, constructing tractable surrogate functions to tightly approximate the non-convex objective functions of the problem (1) is a conventional idea for finding the optimal solutions [4].

Yitong Wang, Jun Zhao, and Liangxin Qian are with Nanyang Technological University, Singapore. Email: yitong002@e.ntu.edu.sg, junzhao@ntu.edu.sg, qian0080@e.ntu.edu.sg, liuc0063@e.ntu.edu.sg

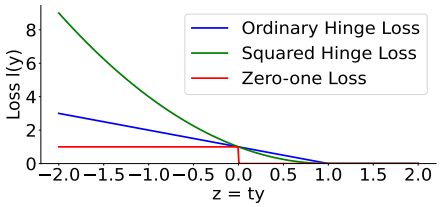


Fig. 1: Hinge loss and zero-one loss functions: zero-one loss function for fixed $t = 1$ is red. Two variants of the hinge loss as a function of $z = ty$: the ordinary variant is blue, and its square is green.

The idea of finding surrogate functions as the upper bound of non-convex functions also appears in machine learning. In support vector machines (SVM), hinge loss, as the continuous and differentiable upper bound of zero-one loss, is a specific type of loss function designed to maximize the margin between classifications. Its ordinary variant is defined as $\max(0, 1 - t \cdot y)$, where t represents the ground truth label and y is the predicted value, respectively. The key approach in optimizing the zero-one loss involves indirectly controlling the actual loss by optimizing its hinge loss. Fig. 1 presents zero-one loss functions and two variants of hinge loss functions. Since the hinge loss is non-negative and convex, its expectation can be regarded as an upper bound of the loss function. This means attempting to minimize this upper bound to ensure that the overall loss is effectively controlled. Thus, the idea of optimizing such an upper bound in machine learning can also be further evolved into non-convex optimization techniques, which are not only efficient but also guarantee the convergence of algorithms to an optimal solution.

Zhao *et al.* [2] in IEEE JSAC 2024 enhanced the performance of the human-centric Metaverse by minimizing the energy consumption and maximizing the utility-cost ratio (UCR) of the system. The proposed algorithm was inspired by a similar idea of choosing a well-approximated upper bound of the non-convex functions to ensure convergence of the algorithm. However, this transform is highly dependent on the specific settings and has not been abstracted into a theorem that can guide and be adopted into a series of FP optimization problems formulated on networking. In Theorem 1 of Section II-A, we give the details of the abstracted theorem and refer to this transform as the `UpperBound` transform.

Multiplicative programming (MP), as another foundational tool, is more appropriate for problems involving product terms between decision variables, such as the joint optimization

of task offloading ratios and communication/computation resources. Such product terms typically arise in edge computing, joint scheduling, or load balancing scenarios where performance metrics depend on the product of multiple factors. To further unlock the potential of `UpperBound` transform, we consider the MP optimization problems, which can typically be formulated as

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & H(\boldsymbol{x}) \overset{\text{def}}{=} G(\boldsymbol{x}) + \sum_{n=1}^{N} [A_n(\boldsymbol{x}) B_n(\boldsymbol{x})] \\ \text{subject to} \quad & \boldsymbol{x} \in \mathcal{X}, \end{aligned}$$

where the objective function is also typically non-convex with respect to the optimization variable $\boldsymbol{x}$.

However, when adopting the idea of finding the upper bound of the non-convex functions in MP problems [5], some inherent characters existing in FP problems are destroyed, resulting in the proposed algorithm of MP problems with `UpperBound` transform failing to converge. We illustrate the inherent characters and corresponding limitations in Section II and update the `UpperBound` transform to make it more general and adaptable to more general scenarios. Updated `UpperBound` transform is referred to as `UP` transform, and its details are presented in Theorem 2.

Additionally, recent works [6]–[8] on resource allocation in wireless communication networks also focus on solving non-convex optimization problems. Specifically, these studies generally adopt the block coordinate descent (BCD) method and propose an iterative algorithm where a single block of the variables (consisting of objective variables and/or introduced auxiliary variables) is optimized while the remaining blocks are kept fixed during each iteration. Different updating orders of blocks can significantly affect the convergence rate of the algorithm. The important **novelty** of our proposed `UP` transform is that all the proposed optimization variables can be optimized in one single block at each iteration of the proposed BCD algorithm, and the rest of the introduced variables (i.e., auxiliary variables) are optimized in another block.

The **goals** of this paper are as follows. We first abstract the `UpperBound` transform and introduce the algorithm of adopting this transform. Then, we illustrate the challenges when directly adopting `UpperBound` transform to MP optimization problems and generalize it to the `UP` transform. We furthermore prove the effectiveness of our proposed transforms in typical communication networks. The main **contributions** of this paper are as follows:

- We abstract the `UpperBound` transform and introduce the adoption of this transform in MP problems with the same settings of functions $A_n(\boldsymbol{x})$ and $B_n(\boldsymbol{x})$ as our recent work [2]. We then illustrate the details of the optimization algorithm and clarify the correspondence between the primal optimization problem and the recast problem.
- We refine the `UpperBound` transform to `UP` transform and focus on the complicated MP optimization problems of $B_n(\boldsymbol{x}) = 0$ with discrete or mixed discrete-continuous variables. After illustrating and resolving corresponding challenges, we analyze the essential properties of `UP` transform, verify it converges to a KKT point, and further present the extended results of this transform.
- To validate the effectiveness, we apply the `UP` transform to the minimization problems of a latency-sensitive communication network by jointly optimizing the partial offloading rate and computation frequencies of the mobile edge server and local devices (i.e., the product of the offloading ratio of computational task and the cost function of the communication system). We obtain the optimum by adopting KKT conditions analysis and then analyze the computational complexity of the `UP` transform-based algorithm.
- We apply the `UP` transform into a heterogeneous network (HetNet) to perform an in-depth analysis of non-convex problems with mixed discrete-continuous optimization variable, i.e., the minimization of the energy consumption and processing delay by jointly optimizing the user association decision vector, computation offloading vector, computation resource, and transmission power. In this application scenario, we propose a novel algorithm with the integration of `UP` transform and successive convex approximation (SCA) technique to tackle this NP-hard optimization problem.

Notation. Let $\mathbb{R}$ be the set of real numbers, while $\mathbb{R}_+$ (resp., $\mathbb{R}_{++}$) denotes the set of non-negative (resp., positive) real numbers; i.e., $\mathbb{R}_+ = [0, \infty)$ and $\mathbb{R}_{++} = (0, \infty)$. We use boldfaced lower-case letters (e.g., $\boldsymbol{x}$) to denote vectors. For an M-dimensional vector $\boldsymbol{x}$, its m-th dimension is given by x_m. The symbol "$\overset{\text{def}}{=}$" and ":=" mean "by definition".

Roadmap. The rest of the paper is organized as follows: Section II presents the intuitions and main results of our proposed transforms. Section III introduces the guidelines for adopting proposed transforms in different practical scenarios of wireless communication networks, and further evaluates and presents the effectiveness of our proposed transforms. Section IV compares the computational complexity of the proposed algorithm with traditional optimization techniques, and improvements compared with prior related papers are given in Section V. Section VI concludes this paper.

II. Main Results

In this section, we first give an intuitive overview of the proposed transforms introduced in Theorems 1 and 2, and then present the details of these theorems.

Intuition of Theorem 1 (`UpperBound` transform). In Theorem 1, we propose an approach to solve a non-convex optimization problem which involves multiplicative term(s)[1,2]. Such a problem belongs to the class of **MP problems**. Specifically, in an MP problem $\mathbb{P}_1$ with the variable being a vector $\boldsymbol{x}$, we can replace any multiplicative term $A_n(\boldsymbol{x})B_n(\boldsymbol{x})$ ("n" is the index for each multiplicative term) by approximation functions

$$K_n(\boldsymbol{x}, y_n) \overset{\text{def}}{=} [A_n(\boldsymbol{x})]^2 y_n + \frac{[B_n(\boldsymbol{x})]^2}{4y_n}, \tag{2}$$

where $\boldsymbol{x} \in \mathcal{X}, y_n \in \mathbb{R}_{++}$ and solve the resulting problem $\mathbb{P}_2$ with the variables being $\boldsymbol{x}$ and $\boldsymbol{y}$, for $\boldsymbol{y} \overset{\text{def}}{=} [y_1, y_2, \ldots, y_N]$.

[1]Multiplicative term(s) are typically non-convex. For instance, even the simple product $f(x, y) \overset{\text{def}}{=} x \times y$ is neither jointly convex nor concave with respect to x and y, because $f(2,2) < \frac{f(1,1)+f(3,3)}{2}$ and $f(2,2) > \frac{f(1,3)+f(3,1)}{2}$.

[2]Section II-D1 shows that fraction term(s) can also be handled by our optimization approach.

Intuition of Theorem 2 (UP transform). Theorem 1 presents a way to handle the non-convex multiplicative term(s) in MP problems, but requires both the multiplier and multiplicand to be strictly positive. As will be explained at the beginning of Section II-C, when one of the multiplier and multiplicand takes the zero value, Theorem 1 does not work, and this motivates us to propose Theorem 2 as a refinement.

Applicability to general optimization involving multiplicative or fractional terms. We present Theorems 1 and 2 in a way of solving MP. Actually, as we will discuss in Section II-D, our approaches in Theorems 1 and 2 can be used for solving **fractional programming (FP)** [9], where FP is related to optimize a ratio, sum of ratios, or related functions. In fact, our approaches are generally applicable to any optimization problems involving multiplicative or fractional terms, not limited to common MP or FP problems.

A. Theorem 1 (UpperBound Transform)

Let $\mathcal{N}$ denote the set $\{1, 2, ..., N\}$. We now present Theorem 1, and some parts are placed in boxes since we will replace them when we present Theorem 2 in Section II-C. In this way, we reuse other parts of Theorem 1 when stating Theorem 2 without repeating those texts.

Theorem 1. *Let $\boldsymbol{x} \in \mathbb{R}^M$ be an M-dimensional real vector. Functions $A_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_{++}$, $\boxed{B_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_{++}}$ for $n \in \mathcal{N}$, and $G(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}$ all have definitions on a feasible set $\mathcal{X} \subseteq \mathbb{R}^M$. We aim to solve the following minimization problem $\mathbb{P}_1$:*

$$\textit{Problem } \mathbb{P}_1\textit{: } \min_{\boldsymbol{x}} \ H(\boldsymbol{x}) \ \textit{subject to } \boldsymbol{x} \in \mathcal{X}, \tag{3}$$

$$\textit{for } H(\boldsymbol{x}) \overset{def}{=} G(\boldsymbol{x}) + \sum_{n=1}^{N} [A_n(\boldsymbol{x}) B_n(\boldsymbol{x})] \tag{4}$$

To solve Problem $\mathbb{P}_1$, we turn to reformulated Problem $\mathbb{P}_2$ below. After defining $K_n(\boldsymbol{x}, y_n)$ based on (2) for $n \in \mathcal{N}$, and further defining

$$W(\boldsymbol{x}, \boldsymbol{y}) \overset{def}{=} G(\boldsymbol{x}) + \sum_{n=1}^{N} K_n(\boldsymbol{x}, y_n), \tag{5}$$

where $\boldsymbol{y} \overset{def}{=} [y_1, y_2, \ldots, y_N]$, we consider

$$\textit{Problem } \mathbb{P}_2\textit{: } \min_{\boldsymbol{x}, \boldsymbol{y}} \ W(\boldsymbol{x}, \boldsymbol{y}) \tag{6}$$

$$\textit{subject to } \boldsymbol{x} \in \mathcal{X} \textit{ and } \boldsymbol{y} \in \mathbb{R}^N_{++}. \tag{6a}$$

We solve $\mathbb{P}_2$ using an alternating optimization (AO) process: With an initial $\boldsymbol{x}^{(0)} \in \mathcal{X}$, we perform the following operations iteratively to get $\boldsymbol{y}^{(j)}$ and $\boldsymbol{x}^{(j)}$ for $j = 1, 2, \ldots$:

① *Fix $\boldsymbol{x}^{(j-1)}$ and solve Problem $\mathbb{P}_2(\boldsymbol{x}^{(j-1)})$ to obtain $\boldsymbol{y}^{(j)} \overset{def}{=} \left[y_1^{(j)}, \ldots, y_N^{(j)}\right]$ for*

$$y_n^{(j)} \overset{def}{=} \frac{B_n(\boldsymbol{x}^{(j-1)})}{2A_n(\boldsymbol{x}^{(j-1)})}, \ \forall n \in \mathcal{N}, \tag{7}$$

which is the closed-form minimizer of $W(\boldsymbol{x}^{(j-1)}, \boldsymbol{y})$ with respect to $\boldsymbol{y}$.

It is clear that $\boldsymbol{y}^{(j)}$ optimally solves $\mathbb{P}_2(\boldsymbol{x}^{(j-1)})$, with $\mathbb{P}_2(\boldsymbol{x})$ denoting optimizing $\boldsymbol{y}$ for $\mathbb{P}_2$ given $\boldsymbol{x}$; i.e.,

$$\textit{Problem } \mathbb{P}_2(\boldsymbol{x})\textit{: } \min_{\boldsymbol{y}} \ W(\boldsymbol{x}, \boldsymbol{y}) \textit{ subject to } \boldsymbol{y} \in \mathbb{R}^N_{++}; \tag{8}$$

② *Fix $\boldsymbol{y}^{(j)}$ and solve Problem $\mathbb{P}_2(\boldsymbol{y}^{(j)})$ to obtain $\boldsymbol{x}^{(j)}$ as a KKT point of $\mathbb{P}_2(\boldsymbol{y}^{(j)})$, where $\mathbb{P}_2(\boldsymbol{y})$ means optimizing $\boldsymbol{x}$ for $\mathbb{P}_2$ given $\boldsymbol{y}$; i.e.,*

$$\textit{Problem } \mathbb{P}_2(\boldsymbol{y})\textit{: } \min_{\boldsymbol{x}} \ W(\boldsymbol{x}, \boldsymbol{y}) \textit{ subject to } \boldsymbol{x} \in \mathcal{X}. \tag{9}$$

Then we have the following result on the relationship between Problems $\mathbb{P}_1$ and $\mathbb{P}_2$:

Result:
With $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ denoting the convergence[a] of $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ as $j \to \infty$, then $\boldsymbol{x}^*$ is a KKT point for Problem $\mathbb{P}_1$. Hence, supposing the convergence criteria[b] is met after J iterations (i.e., after running $j = 1, 2, \ldots, J$), we can consider $\boldsymbol{x}^{(J)}$ as solving $\mathbb{P}_1$ by finding a KKT point (subject to arbitrarily small error tolerance).

[a] From "①" and "②" above, $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})|_{j=1,2,\ldots}$ are from solving Problem $\mathbb{P}_2$ of (6) via alternating optimization (AO), and many studies on AO show the convergence of $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ as $j \to \infty$; viz., the convergence of AO for $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ is guaranteed [10].

[b] An example convergence criteria is that the objective-function values under $(\boldsymbol{x}^{(J-1)}, \boldsymbol{y}^{(J-1)})$ and $(\boldsymbol{x}^{(J)}, \boldsymbol{y}^{(J)})$ respectively differ negligibly; e.g., when $W(\boldsymbol{x}^{(J)}, \boldsymbol{y}^{(J)})/W(\boldsymbol{x}^{(J-1)}, \boldsymbol{y}^{(J-1)})$ is close to 1.

Proof: Please refer to Appendix A. □

Remark 1. *$A_n(\boldsymbol{x})B_n(\boldsymbol{x})$ also covers $A_n(\boldsymbol{y})B_n(\boldsymbol{z})$. When considering $A_n(\boldsymbol{y})B_n(\boldsymbol{z})$, the process of forming $\boldsymbol{x}$ is by concatenating variables $\boldsymbol{y}$ and $\boldsymbol{z}$.*

B. Limitations of Theorem 1

Theorem 1 demonstrates the validity of the abstracted UpperBound transform under specific assumptions; however, applying this transform to practical problems arising in communication networks presents two key challenges:

- **Challenge 1 (Non-strict Positivity):** In network optimization problems, functions $A_n(\boldsymbol{x})$ or $B_n(\boldsymbol{x})$ typically is non-negative rather than strictly positive; that is, $A_n, B_n : \mathbb{R}^M \to \mathbb{R}_+$ with $A_n(\boldsymbol{x}) = 0$ or $B_n(\boldsymbol{x}) = 0$ occurring at feasible points. In practical scenarios, this condition corresponds to many common situations (e.g., inactive links, null offloading decisions, or user disconnection). This highlights a fundamental limitation of the UpperBound transform: it cannot handle such cases. In such cases, the update process in AO may prevent convergence, and **numerical instability** may become a critical consideration during the iterations, undermining the correctness and stability of the transform:
  - **Division by Small Values:** When $A_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_+$ holds, it is possible that $A_n(\boldsymbol{x}^{(j-1)})$ approaches zero during j-th iteration. In this case, the denominator of $y_n^{(j)}$ in the update step "①" becomes exceedingly small, causing $y_n^{(j)}$ in (7) to grow unboundedly large, breaking the convergence of the AO process.
  - **Zero Value Numerator:** When $B_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_+$ holds, it is possible that $B_n(\boldsymbol{x}^{(j-1)}) = 0$ during update process. In this case, the update rule (7) yields $y_n^{(j)} = 0$, which leads to an undefined value of $K_n(\boldsymbol{x}, y_n^{(j)})$ due to a division by zero in its second term $\frac{[B_n(\boldsymbol{x})]^2}{4y_n}$. As a result, the surrogate objective $W(\boldsymbol{x}, \boldsymbol{y}^{(j)})$ becomes invalid, the subproblem $\mathbb{P}_2(\boldsymbol{y}^{(j)})$ is no longer well-defined, and the update step "②" can no longer be proceed..
- **Challenge 2 (Discrete or Mixed Domains):** Communication network problems often involve integer variables

(e.g., user association indicators [6], [7]) or mixed discrete-continuous decision spaces. When $\boldsymbol{x}$ includes such discrete elements, the $\boldsymbol{x}$-subproblem in (9) in each AO step becomes a combinatorial optimization, potentially NP-hard, and no longer guarantees tractability or convergence under the framework of Theorem 1.

To solve the limitations and challenge 1 of UpperBound transform, we refine this transform and propose a generalized reformulation, referred to as the UP transform, whose detail is introduced in Section II-C. The solution to challenge 2 is introduced in Section III-B. Before presenting the new theoretical framework, we first formalize the essential **properties** that the UP transform must satisfy to remain valid in more general MP settings:

- Decoupling: The proposed UP transform should have the same decoupling property with the form $K_n(\boldsymbol{x}, y_n) = K_{n,1}(A_n(\boldsymbol{x}))f_1(y_n) + K_{n,2}(B_n(\boldsymbol{x}))f_2(y_n)$, where y_n is an auxiliary variable;
- Equivalent Solution: Variable $\boldsymbol{x}^*$ is the optimum of $A_n(\boldsymbol{x})B_n(\boldsymbol{x})$ if and only if $\boldsymbol{x}^*$ and y_n^* minimize $K_n(\boldsymbol{x}, y_n)$;
- Relaxed Objective: For $B_n(\boldsymbol{x}) > 0$ (resp. $B_n(\boldsymbol{x}) = 0$), $K_n(\boldsymbol{x}, y_n^*) = A_n(\boldsymbol{x})B_n(\boldsymbol{x})$ (resp. $\min_y K_n(\boldsymbol{x}, y_n)$) for variable $\boldsymbol{x}$ if and only if $y_n^* = \arg\min_{y_n} K_n(\boldsymbol{x}, y_n)$;
- Convexity: Function $K_n(\boldsymbol{x}, y_n)$ is convex of y_n for fixed variable $\boldsymbol{x}$, i.e., its *Hessian* matrix is positive semi-definite;
- Robustness: Updated transform is not only applicable for positive-value functions but also for non-negative functions, i.e., $\{\boldsymbol{x} \mid A_n(\boldsymbol{x}) \geq 0 \vee B_n(\boldsymbol{x}) \geq 0\} \subseteq \mathcal{X}$.

The above properties guarantee that the recast optimization problem after adopting UP transform converges to the same optimal solutions as the original optimization problem.

C. Theorem 2 (UP Transform)

We provide Theorem 2 below, where the intuition has been discussed on Page 3. We consider the case where either the multiplier $A_n(\boldsymbol{x})$ or multiplicand $B_n(\boldsymbol{x})$ in the multiplicative term $A_n(\boldsymbol{x})B_n(\boldsymbol{x})$ takes zero value. Without loss of generality, we can just regard the case of $A_n(\boldsymbol{x}) > 0$ and $B_n(\boldsymbol{x}) \geq 0$. The reason is that for the case of $A_n(\boldsymbol{x}) \geq 0$ and $B_n(\boldsymbol{x}) > 0$, we can name $A_n(\boldsymbol{x})$ and $B_n(\boldsymbol{x})$ as $B'_n(\boldsymbol{x})$ and $A'_n(\boldsymbol{x})$ respectively, and consider the multiplicative term $A'_n(\boldsymbol{x})B'_n(\boldsymbol{x})$, where $A'_n(\boldsymbol{x}) = B_n(\boldsymbol{x}) > 0$ and $B'_n(\boldsymbol{x}) = A_n(\boldsymbol{x}) \geq 0$.

Theorem 2. *First, we use "$\boxed{B_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_+}$" to replace the boxed text "$\boxed{B_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_{++}}$" of Theorem 1, but keep $A_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_{++}$ of Theorem 1. Then with an initial $\boldsymbol{x}^{(0)} \in \mathcal{X}$, we iteratively perform the following operations to obtain $\boldsymbol{y}^{(j)}$ and $\boldsymbol{x}^{(j)}$ for $j = 1, 2, \ldots$:*

- *replace the boxed "①" of Theorem 1 by the boxed "❶" below to ensure that $y_n^{(j)}$ is strictly positive:*

> ❶ *Fix $\boldsymbol{x}^{(j-1)}$ and obtain $\boldsymbol{y}^{(j)} \overset{def}{=} [y_1^{(j)}, \ldots, y_N^{(j)}]$ with*
> $$y_n^{(j)} \overset{def}{=} \max\left\{\frac{B_n(\boldsymbol{x}^{(j-1)})}{2A_n(\boldsymbol{x}^{(j-1)})},\ c_n\right\},\ \forall n \in \mathcal{N}, \quad (10)$$
> *where $c_n > 0$ for $n \in \mathcal{N}$ are hyperparameters that we can adaptively choose;*

- *we still follow "②" of Theorem 1 to obtain $\boldsymbol{x}^{(j)}$ as a KKT point of $\mathbb{P}_2(\boldsymbol{y}^{(j)})$, for $\mathbb{P}_2(\boldsymbol{y})$ defined in (9).*

Finally, we can replace the boxed result part of Theorem 1 with the following:

> ***Results:***
> *(i) The sequence $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ converges as $j \to \infty$;*
> *(ii) With $(\boldsymbol{x}^{\#}, \boldsymbol{y}^{\#})$ denoting the convergence of $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ as $j \to \infty$, then $\boldsymbol{x}^{\#}$ is a KKT point for Problem $\mathbb{P}'_1$ defined as follows.*
> $$\mathbb{P}'_1\text{: } \underset{\boldsymbol{x}}{\text{minimize}}\ L_{\boldsymbol{c}}(\boldsymbol{x}) \text{ subject to } \boldsymbol{x} \in \mathcal{X}, \quad (11)$$
> $$\text{for } L_{\boldsymbol{c}}(\boldsymbol{x}) \overset{def}{=} G(\boldsymbol{x}) + \sum_{n=1}^{N} \max\left\{A_n(\boldsymbol{x})B_n(\boldsymbol{x}), [A_n(\boldsymbol{x})]^2 c_n + \frac{[B_n(\boldsymbol{x})]^2}{4c_n}\right\}, \quad (12)$$
> *where the subscript "$\boldsymbol{c}$" in $L_{\boldsymbol{c}}(\boldsymbol{x})$ means $[c_1, \ldots, c_N]$.*
> *(iii) $L_{\boldsymbol{c}}(\boldsymbol{x})$ is an upper bound of $H(\boldsymbol{x})$ in the sense that $L_{\boldsymbol{c}}(\boldsymbol{x}) \geq H(\boldsymbol{x})$ and $\lim_{\boldsymbol{c}\to\boldsymbol{0}} L_{\boldsymbol{c}}(\boldsymbol{x}) = H(\boldsymbol{x})$. Following the method of minimizing an upper bound to solve an optimization problem, which is widely used in various optimization and machine learning problems (viz., Section I), we consider the iterative process of updating $\boldsymbol{y}^{(j)}$ and $\boldsymbol{x}^{(j)}$ as a way to solve $\mathbb{P}_1$; i.e., $\boldsymbol{x}^{\#}$ in Result "(ii)" above can be considered a "decent" solution with appropriately set $\boldsymbol{c}$.*

Proof: Please refer to Appendix B. □

Remark 2. *Result "(iii)" of Theorem 2 mentions a "decent" solution for Problem $\mathbb{P}_1$ (Experiments in Section III will confirm such "decency") when $A_n(\boldsymbol{x}) > 0$ and $B_n(\boldsymbol{x}) \geq 0$, while Theorem 1 gives a KKT solution for $\mathbb{P}_1$ when $A_n(\boldsymbol{x}) > 0$ and $B_n(\boldsymbol{x}) > 0$. The relaxation that $B_n(\boldsymbol{x})$ may be 0 covers the special case of $B_n(\boldsymbol{x})$ being binary. Binary variables appear in many resource allocation scenarios, for example, heterogeneous mobile edge computing systems. In Section III, we will also give an algorithm based on the convergence of Theorem 2 and SCA to implement the relaxation and obtain the optimal solution.*

Remark 3. *Hyperparameters c_n for $n \in \mathcal{N}$ in (10) maintain the convergence of the whole algorithm and do not result in redundant iterations. While the use of the $\max(\cdot, c_n)$ operator in the auxiliary variable update slightly sacrifices the tightness of the surrogate objective, it significantly enhances numerical stability and robustness.*

Remark 4. *We use c_n in (10) for any iteration index j, but in principle, it can differ across iterations. This means replacing c_n in (10) with $c_n^{(j)} > 0$, and we can choose the hyperparameters $c_n^{(j)}$ for $j \in \{1, 2, \ldots\}$ and $n \in \mathcal{N}$. However, in that case, the results will not be as neat as what we have now in Result "(ii)" of Theorem 2, since $c_n^{(j)}$ depending on j cannot be simply written in the objective function of an optimization problem (i.e., what we do now in $L_{\boldsymbol{c}}(\boldsymbol{x})$ of (12) cannot work if $c_n^{(j)} > 0$ differ for $j \in \{1, 2, \ldots\}$).*

D. Extending Results of Theorems 1 and 2

In this subsection, we extend the results of Theorems 1 and 2 as follows.

1) From multiplicative programming (MP) to fractional programming (FP): Theorems 1 and 2 are for multiplicative

programming (MP). Clearly, multiplicative term(s) can be easily written as fractional terms via a change of variables. Taking Theorem 1 as an example, we can obtain the corresponding result for FP after replacing $A_n(\boldsymbol{x})$ with $\frac{1}{D_n(\boldsymbol{x})}$ for $D_n(\boldsymbol{x}) : \mathbb{R}^M \to \mathbb{R}_{++}$ at necessary places, as elaborated below:

- replacing $K_n(\boldsymbol{x}, y_n)$ of (2) by $K_n^{\mathrm{F}}(\boldsymbol{x}, y_n) \stackrel{\text{def}}{=} \frac{y_n}{[D_n(\boldsymbol{x})]^2} + \frac{[B_n(\boldsymbol{x})]^2}{4y_n}$, where $\boldsymbol{x} \in \mathcal{X}, y_n \in \mathbb{R}_{++}$ and the superscript "F" means "fractional",
- replacing $H(\boldsymbol{x})$ of (4) by $H^{\mathrm{F}}(\boldsymbol{x}) \stackrel{\text{def}}{=} G(\boldsymbol{x}) + \sum_{n=1}^{N} \frac{B_n(\boldsymbol{x})}{D_n(\boldsymbol{x})}$,
- replacing $y_n^{(j)} \stackrel{\text{def}}{=} \frac{B_n(\boldsymbol{x}^{(j-1)})}{2A_n(\boldsymbol{x}^{(j-1)})}$ in (7) by $y_n^{(j)} \stackrel{\text{def}}{=} \frac{D_n(\boldsymbol{x}^{(j-1)})B_n(\boldsymbol{x}^{(j-1)})}{2}$.

For Theorem 2, we can obtain the corresponding result for FP after replacing $A_n(\boldsymbol{x})$ with $\frac{1}{D_n(\boldsymbol{x})}$.

2) Applicability to general optimization problems involving multiplicative or fractional terms: Although Theorems 1 and 2 are about the *sum* of multiplicative term(s) $A_n(\boldsymbol{x})B_n(\boldsymbol{x})$ (after ignoring the "$G(\boldsymbol{x})$" part of (5)), the approach presented in Theorems 1 and 2 can be extended to beyond optimizing the sum. Below, we take Theorem 1 as an example for elaboration and omit the corresponding discussion for Theorem 2. Actually, for any optimization problem $\mathbb{P}$ (with the variable being a vector $\boldsymbol{x}$) involving multiplicative term(s), we can replace any multiplicative term $A_n(\boldsymbol{x})B_n(\boldsymbol{x})$ by $K_n(\boldsymbol{x}, y_n)$ defined in (2), as long as such substitution results in an upper bound of the objective function in minimization problems or a lower bound of the objective function in maximization problems. Then we can solve the resulting optimization problem $\mathbb{Q}$ with the variables being $\boldsymbol{x}$ and $\boldsymbol{y}$. Similarly, for any optimization problem $\mathbb{P}$ involving fractional term(s), we can replace any fractional term $\frac{B_n(\boldsymbol{x})}{D_n(\boldsymbol{x})}$ by $K_n^{\mathrm{F}}(\boldsymbol{x}, y_n)$ defined in Section II-D1. To see the above concretely, a specific example is as follows. For the problem $\mathbb{P}$ of minimizing $G(\boldsymbol{x}) + f_n\big(\sum_{n=1}^{N}[A_n(\boldsymbol{x})B_n(\boldsymbol{x})]\big)$ subject to $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^M$ for non-decreasing functions f_n, we can consider the problem $\mathbb{Q}$ of minimizing $Z(\boldsymbol{x}, \boldsymbol{y}) \stackrel{\text{def}}{=} G(\boldsymbol{x}) + f_n\big(\sum_{n=1}^{N} K_n(\boldsymbol{x}, y_n)\big)$, where $K_n(\boldsymbol{x}, y_n)$ is defined in (2). In view of the alternating optimization (AO) method in Theorem 1, we also use AO to solve problem $\mathbb{Q}$. Specifically, with an initial $\boldsymbol{x}^{(0)} \in \mathcal{X}$, we perform the following two operations iteratively to get $\boldsymbol{y}^{(j)}$ and $\boldsymbol{x}^{(j)}$ for $j = 1, 2, \ldots$:

- use (7) to set $\boldsymbol{y}^{(j)}$, which optimally solves $\mathbb{Q}(\boldsymbol{x}^{(j-1)})$), with $\mathbb{Q}(\boldsymbol{x})$ denoting optimizing $\boldsymbol{y}$ for $\mathbb{Q}$ given $\boldsymbol{x}$;
- solve Problem $\mathbb{Q}(\boldsymbol{y}^{(j)})$ to obtain $\boldsymbol{x}^{(j)}$ as a KKT point of $\mathbb{Q}(\boldsymbol{y}^{(j)})$, where $\mathbb{Q}(\boldsymbol{y})$ means optimizing $\boldsymbol{x}$ for $\mathbb{Q}$ given $\boldsymbol{y}$.

Then similar to the proof of Theorem 1, $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ converges as $j \to \infty$, and with $(\boldsymbol{x}^{\diamond}, \boldsymbol{y}^{\diamond})$ denoting its convergence, $\boldsymbol{x}^{\diamond}$ is a KKT point for Problem $\mathbb{P}$.

III. Application Scenarios

In this section, we delve into two significant practical applications of the UP transform in wireless communication networks. In subsection III-A, we consider the partial offloading scenario in wireless networking and discuss the problems with continuous optimization variables. In subsection III-B, we investigate the strategies for efficient resource and service allocation across the hierarchical network [11], [12], and focus on the scenario concerning mixed-integer optimization (MIO) mentioned in **Challenge 2**. A pivotal contribution of this application scenario in mixed discrete-continuous cases is the development of an innovative algorithm designed to tackle binary variables (mentioned in *Remark* 2) by adopting our proposed transform. The specific parameter settings and experimental evaluations of these application scenarios are shown in Section III-C.

A. Scenario 1: Partial Offloading with Resource Allocation

Resource allocation and the strategy of partial offloading [13] have gained significant attention in the research of federated learning [14], mobile computing [15], and the Internet of Things (IoT) [16], to tackle the limited computational capacities and battery life of mobile devices [17]. These techniques typically involve offloading a portion of the computational tasks from the mobile device to more powerful external servers, either in the cloud or at the edge of the network [18]. In addition, such offloading in networking also helps address latency-sensitive applications by leveraging the proximity and processing power of edge servers.

System Model: To formulate the optimization problem from practical application scenarios, we consider a general mobile edge computing system with N user equipment (UEs) and one edge server. Assume that each UE n processes a computation-intensive task that commonly arises in the vehicular edge computing (VEC) [19], Metaverse [20], or encryption computation [21], and the size of the task is denoted as C_n. In order to enhance spectral efficiency and minimize cross-channel interference, OFDMA (Orthogonal Frequency-Division Multiple Access) is adopted in this proposed system. Due to computational limitations, each UE offloads a fraction $x_n \in [0, 1]$ of its tasks. $x_n = 0$ indicates the UE n's task will be processed locally on the mobile device, while $x_n = 1$ indicates that the computation task is fully offloaded to the edge server and processed. When $0 < x_n < 1$, it indicates the proportion of tasks that are offloaded to the edge server. By denoting the number of CPU cycles required to process task one bit in the local device and edge server as q_n^{local} and q_n^{edge}, we can then obtain the delay for UE n's computation task at different points as

$$T_n(x_n, f_n^l, f_n^e) = T_n^{\text{local}}(x_n, f_n^l) + T_n^{\text{edge}}(x_n, f_n^e). \tag{13}$$

Specifically, $T_n^{\text{local}}(x_n, f_n^l) = \frac{C_n^{\text{local}} q_n^{\text{local}}}{f_n^l} = \frac{(1-x_n)C_n q_n^{\text{local}}}{f_n^l}$, and $T_n^{\text{edge}}(x_n, f_n^e) = \frac{C_n^{\text{edge}} q_n^{\text{edge}}}{f_n^e} = \frac{x_n C_n q_n^{\text{edge}}}{f_n^e}$, where f_n^l and f_n^e represent the computation frequency of UE n and edge server respectively. Compared with the latency of processing, we omit transmission latency due to the proximity of the edge server to mobile devices.

Therefore, the total energy consumption is derived as $E_n(x_n, f_n^l, f_n^e) = k_n T_n^{\text{local}}(x_n, f_n^l) {f_n^l}^3 + k_e T_n^{\text{edge}}(x_n, f_n^e) {f_n^e}^3$, where k_n is the coefficient reflecting the power efficiency of UE n and k_e represents the analogous coefficient related to the power efficiency of the edge server.

We next infer the cost function which jointly considers the offloading ratio and power limitation of each UE as

$O_n(x_n, f_n^l, f_n^e) = w_1 T_n(x_n, f_n^l, f_n^e) + w_2 E_n(x_n, f_n^l, f_n^e)$ and after mathematical transformation, it can be expressed as:

$$O_n(x_n, f_n^l, f_n^e) = (1 - x_n) H_{n,1}(f_n^l) + x_n H_{n,2}(f_n^e),$$

where $H_{n,1}(f_n^l) = C_n q_n^{\text{local}} (\frac{w_1}{f_n^l} + w_2 k_n {f_n^l}^2)$ and $H_{n,2}(f_n^e) = C_n q_n^{\text{edge}} (\frac{w_1}{f_n^e} + w_2 k_e {f_n^e}^2)$. w_1 and w_2 serve as weight parameters specifically designated to modulate the magnitudes of the cost components.

Problem Formulation: UE n can choose the offloading decision $\boldsymbol{x} \coloneqq (x_1, x_2, ..., x_N)$ based on its own and the edge server's computational ability. Meanwhile, the task computation frequency locally $\boldsymbol{f^l} \coloneqq (f_1^l, f_2^l, ..., f_N^l)$, the edge computation frequency $\boldsymbol{f^e} \coloneqq (f_1^e, f_2^e, ..., f_N^e)$ allocated to each mobile equipment can be jointly optimized to achieve optimum. Then, the joint optimization problem incorporating energy cost expenditure, offloading determinations, and computation resources is formulated as follows:

$$\underset{\boldsymbol{x},\ \boldsymbol{f^l},\ \boldsymbol{f^e}}{\text{minimize}} \quad \sum_{n \in \mathcal{N}} (1 - x_n) H_{n,1}(f_n^l) + x_n H_{n,2}(f_n^e) \tag{14}$$

$$\text{subject to} \quad (\mathbf{C}_1)\text{: } 0 \le x_n \le 1, \forall n \in \mathcal{N}, \tag{14a}$$

$$(\mathbf{C}_2)\text{: } \sum\nolimits_{n=1}^{N} f_n^e \le F^e, \tag{14b}$$

$$(\mathbf{C}_3)\text{: } 0 \le f_n^l \le F_n^l, \forall n \in \mathcal{N}, \tag{14c}$$

$$(\mathbf{C}_4)\text{: } 0 \le f_n^e \le F_n^e, \forall n \in \mathcal{N}, \tag{14d}$$

where $\mathcal{N} = \{1, 2, ..., N\}$ is the set of UEs. F^e in $(\mathbf{C}_2)$ and F_n^l in $(\mathbf{C}_3)$ indicate the maximum computation frequency of the edge server and UE n. F_n^e in $(\mathbf{C}_4)$ represented the maximum allocated computation frequency by the edge server to UE n. For each constraint, $(\mathbf{C}_1)$ indicates the offloading decision and the ratio of the offloaded computation task of each UE. $(\mathbf{C}_2)$ restricts the total computation frequency of the edge server. $(\mathbf{C}_3)$ represents that the local processing frequency cannot exceed the device limit. For constraint $(\mathbf{C}_4)$, it mitigates the potential for specific computation tasks to greedily consume computational resources on the edge server.

Solution with UP Transform: Based on Theorem 2, the original optimization is divided into the sub-problem and the coupling of the optimization variable $\boldsymbol{x}$, $\boldsymbol{f^l}$, and $\boldsymbol{f^e}$ can be resolved by introducing auxiliary variables in (7). In the $(k+1)$-th iteration, the objective function is approximated as:

$$G(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{f^e} | \boldsymbol{u}^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{c}^{(k)}) = \sum_{n \in \mathcal{N}} g_n(x_n, f_n^l, f_n^e | u_n^{(k)}, v_n^{(k)}, c_n^{(k)}),$$

where $u_n^{(k)} = \frac{(1 - x_n^{(k)})}{2 H_{n,1}(f_n^{l\,(k)})}$ and $v_n^{(k)} = \frac{x_n^{(k)}}{2 H_{n,2}(f_n^{e})^{(k)}} \in \mathbb{R}_+$ are the introduced auxiliary variables related to the convergence of the proposed algorithm when $0 < x_n < 1$ according to (7). Function $g_n(x_n, f_n^l, f_n^e \mid u_n^{(k)}, v_n^{(k)})$ is defined as shown below, where $c_n^{(k)}$ is the introduced constant according to (10):

$$g_n(x_n, f_n^l, f_n^e | u_n^{(k)}, v_n^{(k)}) = [H_{n,1}(f_n^l)]^2 (u_n^{(k)} + c_n^{(k)}) + \frac{(1 - x_n)^2}{4\big(u_n^{(k)} + c_n^{(k)}\big)}$$

$$+ [H_{n,2}(f_n^e)]^2 (v_n^{(k)} + c_n^{(k)}) + \frac{x_n^2}{4(v_n^{(k)} + c_n^{(k)})} \tag{15}$$

Then sub-problem of the original optimization problem in the $(k+1)$-th iteration is reformulated as:

$$\underset{\boldsymbol{x},\ \boldsymbol{f^l},\ \boldsymbol{f^e}}{\text{minimize}} \quad G(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{f^e} \mid \boldsymbol{u}^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{c}^{(k)}) \tag{16}$$

$$\text{subject to (14a), (14b), (14c), (14d).}$$

Until now, the original optimization problem can be solved iteratively, and the process of solution is listed in Algorithm 1. Instead of using the general convex toolboxes, we adopt Karush–Kuhn–Tucker (KKT) conditions analysis to obtain the solution to the problem in a faster way.

Algorithm 1: Partial Offloading Programming

1 Initialize the index of iteration: $k = 0$; optimization variable $\boldsymbol{\mathcal{X}}^{(0)} = [\boldsymbol{x}^{(0)}, \boldsymbol{f^l}^{(0)}, \boldsymbol{f^e}^{(0)}]$;
2 Calculate and derive auxiliary variable space: $\boldsymbol{\mathcal{A}}^{(0)} = [\boldsymbol{u}^{(0)}, \boldsymbol{v}^{(0)}]$, where $u_n^{(0)} = \frac{(1 - x_n^{(0)})}{2 H_{n,1}(f_n^{l\,(0)})}$, $v_n^{(k)} = \frac{x_n^{(0)}}{2 H_{n,2}(f_n^{e\,(0)})}$, $\forall n \in \mathcal{N}$; Derive the constant vector: $\boldsymbol{c}^{(0)}$ based on (10).
3 **repeat**
4 Obtain the optimal variable $\boldsymbol{\mathcal{X}}^{(k+1)}$ of the $(k+1)$-th iteration by adopting the Algorithm 2 when given auxiliary variable space $\boldsymbol{\mathcal{A}}^{(k)}$;
5 Update $\boldsymbol{\mathcal{A}}^{(k+1)} = [\boldsymbol{u}^{(k+1)}, \boldsymbol{v}^{(k+1)}]$ and $\boldsymbol{c}^{(k+1)}$ with given $\boldsymbol{\mathcal{X}}^{(k+1)}$;
6 $k \leftarrow k + 1$;
7 **until** *Convergence or Maximum iteration number* K;

KKT Analysis: Before proceeding with the analysis of each KKT condition, we first define the Lagrange function by introducing multipliers for constraints as:

$$L = G(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{f^e} | \boldsymbol{u}^{(k)}, \boldsymbol{v}^{(k)}, \boldsymbol{c}^{(k)}) + \sum_{n \in \mathcal{N}} [\gamma_n \cdot (x_n - 1) - \beta_n x_n] + \delta \cdot [(\sum_{n \in \mathcal{N}} f_n^e)$$

$$- F^e] + \sum_{n \in \mathcal{N}} [\zeta_n \cdot (f_n^l - F_n^l) - \epsilon_n f_n^l] + \sum_{n \in \mathcal{N}} [\theta_n \cdot (f_n^e - F_n^e) - \eta_n f_n^e] \tag{17}$$

Then, based on the multipliers and Lagrange function, we get KKT conditions as shown below:

Stationary:

$$\frac{\partial L}{\partial x_n} = D_n(x_n) - \beta_n + \gamma_n = 0, \tag{18a}$$

$$\frac{\partial L}{\partial f_n^l} = Q_n(f_n^l) - \epsilon_n + \zeta_n = 0, \tag{18b}$$

$$\frac{\partial L}{\partial f_n^e} = R_n(f_n^e, \delta) - \eta_n + \theta_n = 0. \tag{18c}$$

where $D_n(x_n) = \frac{x_n - 1}{2(u_n^{(k)} + c_n^{(k)})} + \frac{x_n}{2(v_n^{(k)} + c_n^{(k)})}$ relating to the variable x_n, $Q_n(f_n^l) = 2 H_{n,1}(f_n^l) H'_{n,1}(f_n^l)(u_n^{(k)} + c_n^{(k)})$ relating to the variable f_n^l, and $R_n(f_n^e, \delta) = 2 H_{n,2}(f_n^e) H'_{n,2}(f_n^e)(v_n^{(k)} + c_n^{(k)}) + \delta$ relating to the variable f_n^e and the multiplier δ.

Complementary Slackness:

$$\begin{aligned}
&\text{(19a): } \beta_n \cdot (-x_n) = 0; \quad \text{(19b): } \gamma_n \cdot (x_n - 1) = 0; \\
&\text{(19c): } \delta \cdot [(\sum\nolimits_{n \in \mathcal{N}} f_n^e) - F^e] = 0; \\
&\text{(19d): } \epsilon_n \cdot (-f_n^l) = 0; \quad \text{(19e): } \zeta_n \cdot (f_n^l - F_n^l) = 0; \\
&\text{(19f): } \eta_n \cdot (-f_n^e) = 0; \quad \text{(19g): } \theta_n \cdot (f_n^e - F_n^e) = 0;
\end{aligned} \tag{19}$$

Primal Feasibility: (14a), (14b), (14c), (14d);

Dual Feasibility:

$$\text{(20a)-(20e)} : \beta_n, \gamma_n, \delta, \epsilon_n, \zeta_n, \eta_n, \theta_n \ge 0, \forall n \in \mathcal{N}. \tag{20}$$

Under the KKT conditions, we proceed to seek the optimal solutions by employing the proposed algorithm listed in Algorithm 2.

Theorem 3. *The optimal solution of the proposed objective*

Algorithm 2: Analyze Corresponding KKT Conditions

1: Given the auxiliary variable space $\boldsymbol{\mathcal{A}} = [\boldsymbol{u}, \boldsymbol{v}]$;
2: **for** $n \leftarrow 1$ to N **do**
3: Obtain $\widetilde{x}_n$ by assuming $D_n(x_n) = 0$ on (18a);
4: Obtain $\widetilde{f}_n^l$ by assuming $Q_n(f_n^l) = 0$ on (18b);
5: Obtain $\widehat{f}_n^e(\delta)$ assuming $R_n(f_n^l, \delta) = 0$ on (19c);
6: **end for**
7: $\delta^* \leftarrow \begin{cases} 0, \text{if} \sum_{n\in\mathcal{N}} \widetilde{f}_n^e(0) \le F^e; \\ \text{Solution to } \sum_{n\in\mathcal{N}} \widetilde{f}_n^e(0) > F^e; \end{cases}$
8: **update** x_n^*, ${f_n^l}^*$, and ${f_n^e}^*$ based on Theorem 3;
9: **return** The optimal variable value $\mathcal{X}^* = [\boldsymbol{x}^*, \boldsymbol{f^l}^*, \boldsymbol{f^e}^*]$.

function is obtained by ***Algorithm*** *2 and is expressed as:*

$$\begin{cases} x_n^* = \max\{\min\{\widetilde{x}_n, 1\}, 0\}; \\ {f_n^l}^* = \max\{\min\{\widetilde{f}_n^l, F_n^l\}, 0\}; \\ {f_n^e}^* = \max\{\min\{\widehat{f}_n^e(\delta^*), F_n^e\}, 0\}; \end{cases} \tag{21}$$

where $\widetilde{x}_n$ *meets the condition (18a) with* $D_n(x_n)|_{x_n=\widetilde{x}_n} = 0$, $\widetilde{f}_n^l$ *and* $\widehat{f}_n^e(\delta)$ *meets the condition (18b) and (18c) with* $Q_n(f_n^l)|_{f_n^l=\widetilde{f}_n^l} = 0$ *and* $R_n(f_n^e, \delta)|_{f_n^e=\widehat{f}_n^e(\delta)} = 0$, *respectively.*

Proof: Please refer to Appendix C. □

B. Scenario 2: User Association with Resource Allocation

In this subsection, we consider a heterogeneous mobile edge computing system [22]–[24] consisting of one Macro Base Station (MBS) [25] and one Small Base Station (SBS) [26] as an example and propose a novel two-tier computation offloading algorithm with the convergence of the UP transform and successive convex approximation (SCA) approach [27].

System Model: In the proposed system, we assume that each base station also has one MEC server to execute the offloaded computation task, and we denote the set of base stations as $\mathcal{M} = \{1, 2\}$ where $m = 1$ is the SBS and $m = 2$ represents the MBS. SBS is connected to the MBS via wired links. There are N mobile users, denoted as $\mathcal{N} = \{1, 2, ..., N\}$. The SBS and mobile users are randomly distributed within the convergence of MBS. Each mobile user is connected to the SBS via wireless links and has a computation-intensive and delay-sensitive task. We define the task of each user n as $D_n = (d_n, c_n)$, where d_n denotes the data size of the computation task, c_n is the number of CPU cycles for computing one bit of task D_n.

For this two-tier computation offloading framework, a mobile user can partition the task into two parts. One part is offloaded to the MEC server which is associated with the base stations, and the other part is executed locally. As the computation resource of the MEC server in SBS is limited, the part offloaded will further be partitioned into two parts when the computation resource of the MEC server on SBS is exhausted. The SBS will offload the remaining part of the task to the MEC server on the MBS. Furthermore, each mobile user can also offload the task directly to the MBS. We denote the part offloaded to the base stations as $d_{n,m}(0 \le d_{n,m} \le d_n)$ and the part processed locally as $d_n - d_{n,m}$. In the second tier, SBS further divides the fragment $d_{n,1}$ as $d_{n,1} - d_{n,1}^{'}$ and $d_{n,1}^{'}(0 \le d_{n,1}^{'} \le d_{n,1} \le d_n)$, and the latter part is offloaded to the MBS. For the offloading decision of each user, we denote $x_{n,m} = \{0, 1\}$ as the association variable, where $x_{n,m} = 0$ if user $n \in \mathcal{N}$ is associated with base station $m \in \mathcal{M}$ and $x_{n,m} = 0$ otherwise.

For the wireless communication model, we also adopt the OFDMA technique between mobile users and base stations, and the channel power gain is denoted as $H_{n,m}$. Then, when mobile user n is associated with the SBS, the signal-to-interference plus noise ratio (SINR) of the uplink will be expressed as $\gamma_{n,1} = \frac{P_n H_{n,1}}{\sigma^2}$, where P_n is the transmission power of mobile user n, and σ^2 is defined as the noise power. The inter-user interference is not taken into consideration with the adoption of OFDMA. Assuming the SBS allocates bandwidth equally to its associated users, the achievable uplink data rate of mobile user n can be inferred as $R_{n,1} = \frac{B_1}{\sum_{n=1}^{N} x_{n,1}} \ln(1 + \gamma_{n,1})$, where B_1 represents the bandwidth of the SBS and $\sum_{n=1}^{N} x_{n,1}$ denotes the total number of mobile users associated with the SBS. Similarly, if a mobile user is associated with MBS, the SINR is expressed as $\gamma_{n,2} = \frac{P_n H_{n,2}}{\sigma^2}$, and the achievable data rate will be $R_{n,2} = \frac{B_2}{\sum_{n=1}^{N} x_{n,2}} \ln(1+\gamma_{n,2})$ where B_2 is the total bandwidth of the MBS and $\sum_{n=1}^{N} x_{n,2}$ is the number of users associated with MBS.

Then, we further infer the cost function of the local computation in terms of latency and energy consumption. We denote f_n^l as the computation resource when processing the task locally. Then we obtain the computation time as $T_{n,m}^{\text{local}} = \frac{(d_n - d_{n,m})c_n}{f_n^l}$, and the corresponding energy consumption is $E_{n,m}^{\text{local}} = k^l (d_n - d_{n,m}) c_n {f_n^l}^2$ where k_n is the effective switched capacitance depending on the chip architecture.

For the task offloaded to the SBS, we denote the allocated computation frequency by the SBS to the task as f_n^s and assume that MBS allocates fixed computation frequency f_0 to each offloaded task. Then the processing time in SBS is:

$$T_{n,1}^{\text{SBS}} = \frac{d_{n,1}}{R_{n,1}} + \frac{(d_{n,1} - d_{n,1}^{'})c_n}{f_n^s} + \frac{d_{n,1}^{'}}{r_0} + \frac{d_{n,1}^{'} c_n}{f_0},$$

where r_0 represents the data rate of the wired link. Then, the corresponding energy consumption is derived below:

$$E_{n,1}^{\text{SBS}} = \frac{P_n d_{n,1}}{R_{n,1}} + k^S (d_{n,1} - d_{n,1}^{'}) c_n {f_n^s}^2 + \frac{\bar{P} d_{n,1}^{'}}{r_0} + k^M d_{n,1}^{'} c_n f_0^2,$$

where $\bar{P}$ is the offloading power via wired line, k^S and k^M are the coefficients reflecting the power efficiency of the MEC server in SBS and MBS, respectively. When the mobile user directly offloads the task to the MEC server in MBS, the execution time is $T_{n,2}^{\text{MBS}} = \frac{d_{n,2}}{R_{n,2}} + \frac{d_{n,2} c_n}{f_0}$, and the energy consumption is $E_{n,2}^{\text{MBS}} = \frac{P_n d_{n,2}}{R_{n,2}} + k^M d_{n,2} c_n f_0^2$.

Then, the total latency and energy consumption of the task for each user can be concluded as follows:

$$T_n = (\sum\nolimits_{m\in\mathcal{M}} x_{n,m} T_{n,m}^{\text{local}}) + x_{n,1} T_{n,1}^{\text{SBS}} + x_{n,2} T_{n,2}^{\text{MBS}},$$

$$E_n = (\sum\nolimits_{m\in\mathcal{M}} x_{n,m} E_{n,m}^{\text{local}}) + x_{n,1} E_{n,1}^{\text{SBS}} + x_{n,2} E_{n,2}^{\text{MBS}}.$$

Problem Formulation: We define $\boldsymbol{x} \coloneqq \{x_{n,m}\}$ as the user association decision vector, $\boldsymbol{f^l} \coloneqq \{f_n^l\}$ and $\boldsymbol{f_n^s} \coloneqq \{f_n^s\}$ as the vector of local and SBS computation frequency respectively, $\boldsymbol{P} \coloneqq [P_n|_{n\in\mathcal{N}}]$ as the transmission power vector, and $\boldsymbol{d} \coloneqq \{d_{n,m}, d_{n,1}^{'}\}$ as the computation offloading vector. The

optimization problem is formulated as:

$$\underset{\boldsymbol{x},\boldsymbol{f^l},\boldsymbol{f^s},\boldsymbol{P},\boldsymbol{d}}{\text{minimize}} \quad \sum_{n=1}^{N}(w_1T_n + w_2E_n) \tag{22}$$

$$\text{subject to} \quad (\mathbf{C}_1) \sum_{m\in\mathcal{M}} x_{n,m} = 1, \forall n \in \mathcal{N}, \tag{22a}$$

$$(\mathbf{C}_2) \quad 0 \le f_n^l \le F_n, \forall n \in \mathcal{N}, \tag{22b}$$

$$(\mathbf{C}_3) \quad \sum_{n=1}^{N} x_{n,1} f_n^s \le F^s, \tag{22c}$$

$$(\mathbf{C}_4) \quad 0 \le P_n \le P_n^{\max}, \forall n \in \mathcal{N}, \tag{22d}$$

$$(\mathbf{C}_5) \quad 0 \le d_{n,1}^{'} \le d_{n,1}, \forall n \in \mathcal{N}, \tag{22e}$$

$$(\mathbf{C}_6) \quad 0 \le d_{n,m} \le d_n, \forall n \in \mathcal{N}, m \in \mathcal{M}, \tag{22f}$$

$$(\mathbf{C}_7) \quad f_n^s \ge 0, \forall n \in \mathcal{N}, \tag{22g}$$

$$(\mathbf{C}_8) \quad x_{n,m} \in \{0,1\}, \forall n \in \mathcal{N}, m \in \mathcal{M}. \tag{22h}$$

where w_1 and w_2 represent the weight parameters designed to modulate the magnitudes of the cost components. The offloading decision constraints relate to constrains $(\mathbf{C}_1)$, $(\mathbf{C}_3)$ and $(\mathbf{C}_8)$. Constraints $(\mathbf{C}_2)$, $(\mathbf{C}_3)$, and $(\mathbf{C}_7)$ limit the frequency of computation locally and in SBS, respectively. Also, constraints $(\mathbf{C}_5)$ and $(\mathbf{C}_6)$ also constructed as the limitation of computation resource. $(\mathbf{C}_4)$ limits the power resource of mobile user n.

Solution with Updated Transform and SCA: Based on the constraints of the user association vector, we first adopt SCA technology to solve the discrete optimization variable $\boldsymbol{x}$ resulting in the problem as NP-hard. Without loss of equivalence, $(\mathbf{C}_8)$ can be rewritten as:

$$x_{n,m} \in [0,1], \ \forall n \in \mathcal{N}, m \in \mathcal{M}, \tag{23}$$

$$\textbf{and} \ \sum\nolimits_{n\in\mathcal{N}} \sum\nolimits_{m\in\mathcal{M}} x_{n,m}(1 - x_{n,m}) \le 0. \tag{24}$$

Note that the optimization problem has been transitioned into a continuous optimization problem, resulting in a notable reduction in computational complexity when contrasted with the direct resolution of the original discrete variable $x_{n,m}$. However, the function $\sum_{n\in\mathcal{N}} \sum_{m\in\mathcal{M}} x_{n,m}(1-x_{n,m})$ in constraint (24) is a concave function. To facilitate the solution, we adopt a method that introduces a penalty term for this concave constraint into the objective function, which is expressed as:

$$\sum_{n=1}^{N}(w_1T_n + w_2E_n) - \tau \cdot \sum\nolimits_{n\in\mathcal{N}} \sum\nolimits_{m\in\mathcal{M}} x_{n,m}(x_{n,m} - 1),$$

where τ is the penalty parameter with $\tau > 0$. Then, the objective function becomes concave due to the concavity of the second term. Simultaneously, given the second term is differentiable, we utilize the first-order Taylor series to linearize it at each iteration. Specifically, in the $(i+1)$-th iteration,we approximate each $\sum_{n\in\mathcal{N}} \sum_{m\in\mathcal{M}} x_{n,m}(x_{n,m}-1)$ with $\sum_{n\in\mathcal{N}} \sum_{m\in\mathcal{M}} x_{n,m}^{(i)}(x_{n,m}^{(i)} - 1) + (2x_{n,m}^{(i)} - 1)(x_{n,m} - x_{n,m}^{(i)})$ denoted as $H(\boldsymbol{x} \mid \boldsymbol{x}^{(i)})$, where $\boldsymbol{x}^{(i)}$ is the optimal solution of the i-th sub-problem. Therefore, the objective function is reformulated as:

$$\sum_{n=1}^{N}(w_1T_n + w_2E_n) - \tau \cdot H(\boldsymbol{x} \mid \boldsymbol{x}^{(i)}). \tag{25}$$

After transformation, we obtain the objective function as:

$$\sum_{n=1}^{N} Q_n(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{f^s}, \boldsymbol{P}, \boldsymbol{d}) - \tau \cdot H(\boldsymbol{x} \mid \boldsymbol{x}^{(i)}),$$

where $Q_n(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{f^s}, \boldsymbol{P}, \boldsymbol{d}) = O_n(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{d}) + S_n(\boldsymbol{x}, \boldsymbol{P}, \boldsymbol{d}) + U_n(\boldsymbol{x}, \boldsymbol{f^s}, \boldsymbol{d}) + V_n(\boldsymbol{x}, \boldsymbol{d})$ and specifically,

$$V_n(\boldsymbol{x}, \boldsymbol{d}) = x_{n,1}d_{n,1}^{'}C + x_{n,2}d_{n,2}(C - \frac{w_1 + w_2\bar{P}}{r_0}),$$

$$U_n(\boldsymbol{x}, \boldsymbol{f^s}, \boldsymbol{d}) = x_{n,1}(d_{n,1} - d_{n,1}^{'})(\frac{w_1}{f_n^s} + w_2k^S{f_n^s}^2)c_n,$$

$$O_n(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{d}) = \sum_{m\in\mathcal{M}} x_{n,m}(d_n - d_{n,m})(\frac{w_1}{f_n^l} + w_2k^l{f_n^l}^2)c_n,$$

$$S_n(\boldsymbol{x}, \boldsymbol{P}, \boldsymbol{d}) = \sum_{m\in\mathcal{M}} x_{n,m}\frac{d_{n,m}(\sum_{n=1}^{N} x_{n,m})}{B_m \ln(1 + \frac{P_nH_{n,m}}{\sigma^2})}(w_1 + w_2P_n),$$

and the constant $C = \frac{w_1 + w_2\bar{P}}{r_0} + (\frac{w_1}{f_0} + w_2k^M f_0^2)c_n$.

Theorem 4. *Proposed objective function $Q_n(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{f^s}, \boldsymbol{P}, \boldsymbol{d})$ with coupled optimization variable can solved by optimizing convex function $\widetilde{Q}_n(\boldsymbol{\mathcal{X}} \mid \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}})$ derived from the* UP *transform, where $\mathcal{X} \coloneqq [\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{f^s}, \boldsymbol{P}, \boldsymbol{d}]$ is the space of optimization variable, $\boldsymbol{\mathcal{A}} \coloneqq [\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\delta}, \boldsymbol{\epsilon}, \boldsymbol{\zeta}, \boldsymbol{\eta}, \boldsymbol{\theta}]$ is the auxiliary variables space, and $\boldsymbol{\mathcal{B}} \coloneqq [\boldsymbol{b}, \boldsymbol{e}, \boldsymbol{g}, \boldsymbol{h}, \boldsymbol{j}, \boldsymbol{l}, \boldsymbol{q}, \boldsymbol{z}]$ is the constant space.*

Proof: Please refer to Appendix D. □

Then the transformed objective function can be derived as: $\sum_{n=1}^{N} \widetilde{Q}_n(\boldsymbol{\mathcal{X}} \mid \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}) - \tau \cdot H(\boldsymbol{x} \mid \boldsymbol{x}^{(i)})$, where $\widetilde{Q}_n(\boldsymbol{\mathcal{X}} \mid \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}) = \widetilde{O}_n + \widetilde{S}_n + \widetilde{U}_n + \widetilde{V}_n$ is given in Appendix D.

However, this optimization problem is not convex as the optimization variable is still coupled in the constraint (22c). We apply the proposed UP transform to decouple the variables:

$$\sum_{n=1}^{N} \Big[(f_n^s)^2(\lambda_n^{(k)} + u_n^{(k)}) + \frac{x_{n,1}^2}{4(\lambda_n^{(k)} + u_n^{(k)})}\Big] - F^s \le 0, \tag{26}$$

where auxiliary variable $\lambda_n^{(k)}$ and constant $u_n^{(k)}$ obey the rules in Theorem 2. And $\lambda_n^{(k)} = \frac{x_{n,1}^{(k)}}{2f_n^{s(k)}}$ in the $(k+1)$-th iteration.

The the sub-problem of the original problem in the $(k+1)$-th iteration of UP transform under the $(i+1)$-th iteration of SCA approach is reformulated as:

$$\underset{\boldsymbol{x},\boldsymbol{f^l},\boldsymbol{f^s},\boldsymbol{P},\boldsymbol{d}}{\text{minimize}} \quad \sum_{n=1}^{N} \widetilde{Q}_n(\boldsymbol{\mathcal{X}} | \boldsymbol{\mathcal{A}}^{(k)}, \boldsymbol{\mathcal{B}}^{(k)}) - \tau \cdot H(\boldsymbol{x} | \boldsymbol{x}^{(i)}) \tag{27}$$

$$\text{subject to} \quad (22a), (22b), (22d), (22e), (22f), (22g), (23), (26)$$

Then, the original optimization problem can be addressed through an iterative process. For this user association practical application, various analysis methods can be considered. To prevent unnecessary repetition in analyses, the CVX toolboxes are utilized for solving each convex sub-problem efficiently.

Proposed Algorithm: In the proposed method with the convergence of SCA technique, a penalty function is employed to further facilitate the transformation of various conditions, effectively segmenting the algorithm into two complementary components: *Inter-Sub-Problem Programming* and *Intra-Sub-Problem Programming*. For the inter-sub-problem programming as illustrated in Algorithm 3, it focused on deriving the optimal solution $\boldsymbol{x}^{(i)}$ in the i-th sub-problem. This solution is subsequently utilized in the succeeding iteration of the SCA method. Then the proposed intra-sub-problem programming, as illustrated in Algorithm 4, which is obtained after adopting the UP transform, will be utilized to derive the optimal solution in the succeeding k-th iteration. SCA progressively approximates feasible convex subsets within the non-convex space, while the UP transform addresses instability issues caused by multiplicative terms. Their integration ensures feasibility and

Algorithm 3: Inter-Sub-Problem Programming

Initialization of the iteration index: $i = 1$;
Initialization of the optimal solution space: $\boldsymbol{\mathcal{S}}^{(0)}=[\boldsymbol{x}^{(0)}, \boldsymbol{f^l}^{(0)}, \boldsymbol{f^s}^{(0)}, \boldsymbol{P}^{(0)}, \boldsymbol{d}^{(0)}]$;
Adopt the SCA method to obtain the problem (27);
repeat
  Obtain the i-th sub-problem by using $\boldsymbol{x}^{(i-1)}$;
  Solve the i-th sub-problem by using Algorithm 4 to get the optimal optimal variable value $\mathcal{X}^*=[\boldsymbol{x}^*, \boldsymbol{f^l}^*, \boldsymbol{f^s}^*, \boldsymbol{P}^*, \boldsymbol{d}^*]$;
  Let $\boldsymbol{\mathcal{S}}^i \leftarrow \mathcal{X}^*$ of i-th sub-problem;
  Set $i \leftarrow i + 1$;
until $|\boldsymbol{\mathcal{S}}^{(i)} - \boldsymbol{\mathcal{S}}^{(i-1)}| \leq \bar{\epsilon}_0$ *or Maximum iteration number* I;
return $\boldsymbol{\mathcal{S}}^{(i)} = [\boldsymbol{x}^{(i)}, \boldsymbol{f^l}^{(i)}, \boldsymbol{f^s}^{(i)}, \boldsymbol{P}^{(i)}, \boldsymbol{d}^{(i)}]$ as the optimal solution of the original optimization problem.

convergence under non-convex scenarios. Therefore, the proposed algorithm will converge and get the optimal solutions.

C. Experimental Results

1) Baseline Settings: To evaluate the performance of our proposed UP transform-based algorithm, we compare it with several baselines commonly used in edge computing and task offloading scenarios:

- **Full Offloading**: Each UE offloads its entire computation task to the edge server. The edge server allocates its computation resources to users proportionally.
- **No Offloading**: All tasks are processed locally at the UE side, with computation frequencies constrained by each device's maximum capability.
- **Random Offloading**: For each UE, the offloading ratio is randomly sampled from a uniform distribution in $[0, 1]$, and the computation resources are allocated accordingly.
- **Gradient Descent / Newton's Method / Interior Point Method / B&B Method / Heuristic Allocation**: These are classical optimization methods applied to solve the original objective without using our proposed transform. Computational complexity comparisons of our proposed algorithm with these classical methods are further presented in Section IV-B.

2) Scenario 1: To evaluate the effectiveness of our proposed transform in partial offloading scenarios, we consider the MEC system involving $N = 30$ UEs and one edge server. The computing capacity of the edge server F^e is set to 10GHz, while each UE operates with a local computation frequency F_n^l of 1.5GHz. Each UE can be allocated an edge computation frequency F_n^e up to 10GHz. The computation energy efficiency coefficients k_n and k_e for UEs and the edge server are both set to 10^{-26}. The task sizes C_n for UEs are uniformly distributed between 100MB and 500MB. The reverse diffusion process incorporates fixed discretization steps $\Delta t_{n,0} = 1/500$ and $\Delta t_{n,1} = 1/1000$, ensuring stability in the iterative computation. To simulate an initial noisy state, the average error rate is set to 1, implying that the original content is fully Gaussian noise.

Algorithm 4: Intra-Sub-Problem Programming

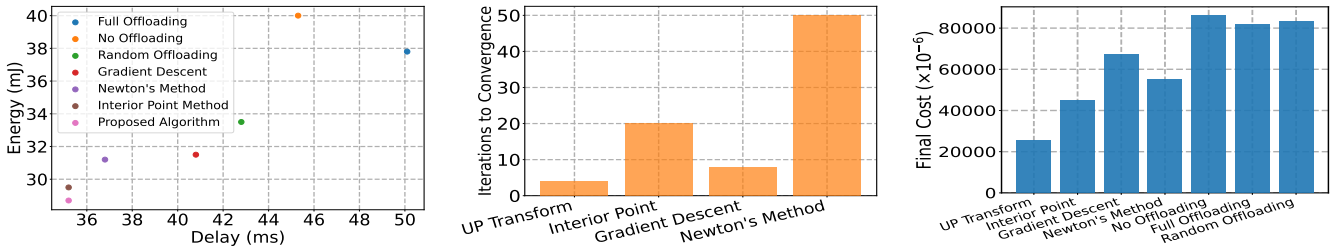
Initialization of the iteration index: $k = 0$;
Initialization of the optimization variable space: $\boldsymbol{\mathcal{X}}^{(0)} = [\boldsymbol{x}^{(0)}, \boldsymbol{f^l}^{(0)}, \boldsymbol{f^s}^{(0)}, \boldsymbol{P}^{(0)}, \boldsymbol{d}^{(0)}]$;
Calculate and derive auxiliary variable space: $\boldsymbol{\mathcal{A}}^{(0)}$ and constant space: $\boldsymbol{\mathcal{B}}^{(0)}$ based on the analysis of Theorem 4;
repeat
  Obtain the optimal variable $\boldsymbol{\mathcal{X}}^{(k+1)}$ of the $(k+1)$-th iteration by adopting CVX toolboxes when given auxiliary variable space $\boldsymbol{\mathcal{A}}^{(k)}$ and constant $\boldsymbol{\mathcal{B}}^{(k)}$;
  Update $\boldsymbol{\mathcal{A}}^{(k+1)}$ and $\boldsymbol{\mathcal{B}}^{(k+1)}$ with given $\boldsymbol{\mathcal{X}}^{(k+1)}$;
  $k \leftarrow k + 1$;
until *Convergence or Maximum iteration number* K;
return $\boldsymbol{\mathcal{X}}^* \leftarrow \boldsymbol{\mathcal{X}}^{(k)}$ as the optimal solution of the i-th sub-problem.

Delay-Energy Tradeoff. Fig. 2(a) shows the delay-energy tradeoff, illustrating the operational efficiency of each method. Our proposed algorithm achieves the best overall tradeoff, attaining the lowest energy consumption and one of the shortest delays. Compared to full offloading and no offloading, which suffer from either excessive energy usage or prolonged delays, the proposed method effectively balances the computation between edge and local resources. Notably, random offloading and gradient descent show moderate performance but fail to match the low-delay, low-energy profile of our approach. The interior-point method performs comparably in delay but consumes slightly more energy, while Newton's method, despite achieving good delay, is less energy efficient.

Convergence Efficiency. In Fig. 2(b), we compare the convergence speed of different optimization methods with our proposed transform method. Our proposed UP transform-based algorithm demonstrates superior performance, requiring only a few iterations to converge. This significantly outperforms traditional optimization approaches: gradient descent, interior-point, and Newton's method. The efficiency of our method can be attributed to the well-designed surrogate function and the adaptive auxiliary variable strategy, which accelerates convergence without sacrificing accuracy.

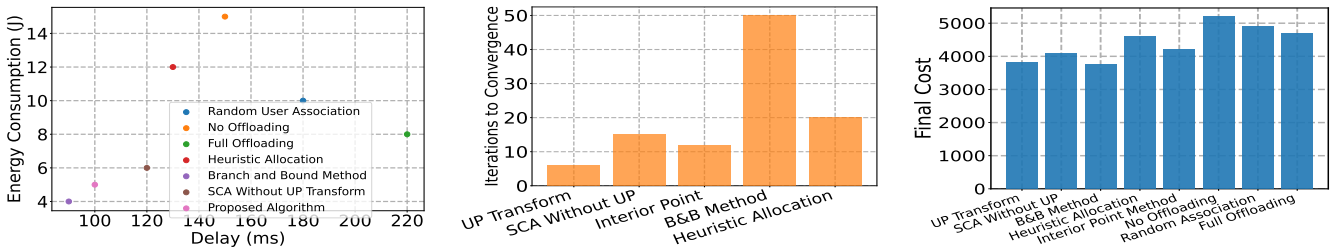
Cost Function. In terms of the cost function values, we present the experimental results in Fig. 2(c). The proposed algorithm consistently achieves the lowest final cost function value, outperforming all baselines. Interior-point and Newton's methods follow but still lag behind, with cost values nearly double those of the proposed method. Full and no offloading strategies result in the highest costs, indicating suboptimal resource utilization and task allocation. This proves that static offloading decisions are insufficient for dynamic, resource-constrained edge computing scenarios.

3) Scenario 2: To evaluate the performance of the proposed transform in heterogeneous communication networks, we consider a network consisting of $N = 20$ users. The location of the MBS is fixed at the centre of the network. Locations of the SBS and mobile users are randomly distributed. We



(a) Delay-Energy Tradeoff (b) Convergence Comparison (c) Final Cost Function Value Comparison

Fig. 2: Evaluation results of Scenario 1



(a) Delay-Energy Tradeoff (b) Convergence Comparison (c) Final Cost Function Value Comparison

Fig. 3: Evaluation results of Scenario 2

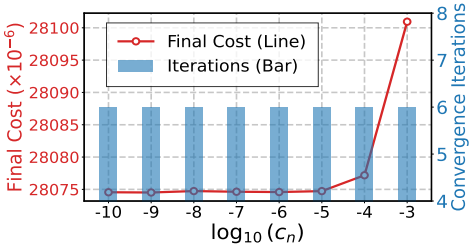
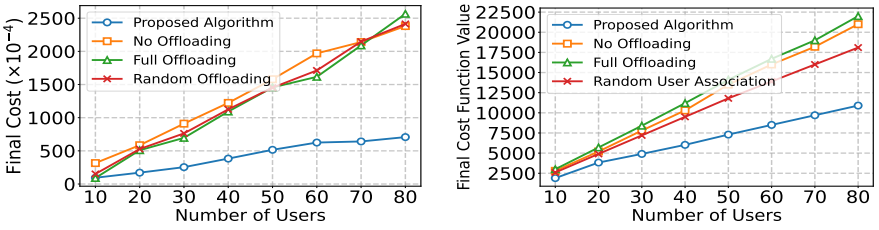


Fig. 4: Sensitivity Analysis to c_n



(a) Partial Offloading Scenario (b) User Association Scenario

Fig. 5: Cost function values vs. Number of users in proposed scenarios

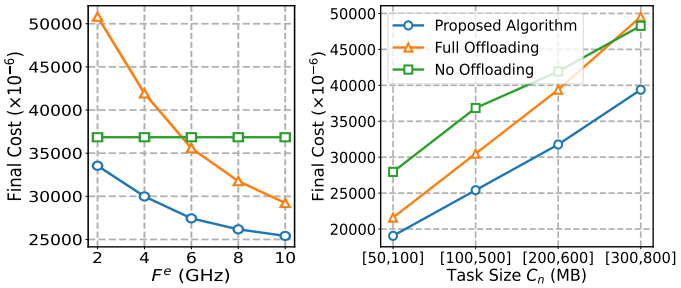
set the maximum transmission power of each mobile user as 100mW. The path-loss between the MBS and the mobile user is simulated by $128.1 + 36.7\log_{10}(d) + \mu$ (in dB), and that between the SBS and the mobile user is set as $140.7 + 36.7\log_{10}(d) + \mu$ (in dB), where d is the distance in km and μ satisfies the log-distribution $\mathcal{N}(0, 8\text{dB})$. We set the AWGN noise power as $\sigma^2 = -110$dBm. The bandwidth of MBS and SBS are set as 10MHz and 5MHz, respectively. For the computation frequency of the SBS and mobile user, we set $F^s = 20$GHz and $F_n = 1$GHz. The data size and required number of CPU cycles per bit follow the of each computation task are set as $d_n = 350$KB and $c_n = 75$ cycles/bit. For the computation resource of MBS, we set the $f_0 = 5$GHz and $r_0 = 1$Gbps. The computation energy efficiency coefficient k^l, k^S, and k^M are set as 1×10^{-25}. For the penalty parameter τ, we set the value as 10^5.

Convergence Speed. We similarly compare our optimization performance with other baselines: random user association, branch and bound (B&B) method, heuristic allocation, SCA without UP transform, full offloading, and no offloading. The evaluation results are presented in Fig. 3. As shown in Fig. 3(b), our proposed method converges in the fewest iterations, significantly outperforming all other baselines. This highlights the computational efficiency of UP transform, particularly in contrast to the B&B method, which suffers from high convergence latency due to its combinatorial search.

Cost Efficiency Evaluation. In terms of the delay-energy tradeoff and cost function, as shown in Fig. 3(b) and 3(c), while the B&B algorithm yields the globally optimal value, it does so at the cost of substantially slower convergence, whereas our algorithm reaches a near-optimal solution with far greater efficiency. Together, these results validate our proposed method's capability to balance cost, computational speed, and resource efficiency—making it highly suitable for real-time optimization in heterogeneous edge computing environments.

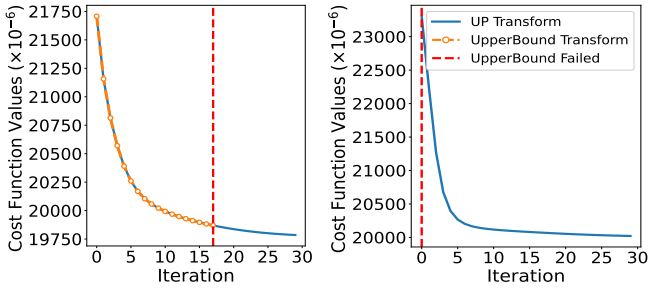
D. Hyperparameter Sensitivity Analysis

The sensitivity analysis is shown in Fig. 4 and reveals that c_n meets the properties introduced in *Remark* 3 (e.g., 10^{-8} to 10^{-6}), the final cost remains stable, indicating that the UP transform maintains high optimization accuracy across a wide range of small auxiliary parameters. However, as c_n increases



(a) Impact of server computing (b) Impact of take size

Fig. 6: Scalability under computation and memory constraints



(a) Initialized $\boldsymbol{x}$ without 0 Element (b) Initialized $\boldsymbol{x}$ with 0 Element

Fig. 7: UpperBound Transform vs UP Transform

beyond 10^{-4}, the final cost sharply increases, suggesting that an excessively large c_n hampers the optimization performance. Meanwhile, the number of convergence iterations remains nearly constant for all c_n values, demonstrating that the convergence speed is largely independent of the choice of c_n. This confirms that `UP` transform ensures stable and efficient convergence, with c_n mainly influencing the final solution quality rather than the optimization dynamics.

E. Scalability and Robustness

Cost Scalability with Varying User Loads. In Fig. 5, we evaluate the cost function values with different numbers of users. Regardless of the kind of scenarios mentioned (i.e., partial offloading shown in Fig. 5(a) and user association shown in Fig. 5(b)), our proposed algorithm achieves the lowest cost values across all user counts, as it effectively optimizes offloading decisions, associations, and resource allocation by dynamically adjusting to system constraints and user demands.

Computation and Memory Constraints. Fig. 6 illustrates the scalability of different offloading strategies under varying system constraints, specifically edge server computing power F^e and take size C_n. In Fig. 6(a), we examine the impact of server-side computation capacity. As the edge server's computing power increases from 2GHz to 10GHz, our proposed algorithm consistently outperforms both baselines, showing a significant reduction in cost with increasing F^e. Fig. 6(b) illustrates how varying task sizes affect overall system performance. As the task size range increases from $[50, 100]$MB to $[300, 800]$MB, our proposed algorithm exhibits the slowest cost growth, demonstrating its robustness in handling large workloads through dynamic local-edge balancing. In contrast, full offloading becomes increasingly costly due to edge-side congestion and energy consumption.

F. `UpperBound` Transform versus `UP` Transform

Fig. 7 presents a comparative analysis of the convergence process of `UpperBound` transform (Theorem 1) and `UP` transform (Theorem 2) under two different initial settings of the optimization variables.

In Fig 7(a), we initialize the optimization variable $\boldsymbol{x}$ with strictly positive elements within the feasible domain $[0, 1]$. Though `UpperBound` transform performs comparably to the `UP` transform during the first few iterations, it meets convergence failure when $\boldsymbol{x}$ is optimized to 0 values. While in Fig. 7(b), some elements of $\boldsymbol{x}$ are initialized as zero. It is presented that the `UpperBound` transform fails to converge when the initial variable contains zero elements. This result is consistent with the theoretical analysis in Section II-B. In contrast, the `UP` transform successfully handles such scenarios by using hyperparameters to avoid undefined operations, ensuring stable updates of auxiliary variables throughout the iterative process.

These results empirically validate the robustness and general applicability of the `UP` transform in practical optimization problems, especially in settings with non-negative or mixed domain variables. The improved convergence behavior of the `UP` transform across varying initial conditions confirms its suitability for broader non-convex optimization tasks in edge computing and communication networks.

IV. Computation Complexity Analysis

In this section, we analyze the computational complexity of our proposed transforms and compare the complexity with other traditional algorithms.

A. Complexity of `UP` Transform

Before analyzing the complexity of the problem `UP` transform, the overview of the proposed algorithm adopting `UP` transform is presented in Algorithm 5. The KKT analysis is applied to update the optimization variable during each iteration. For the update phase of $\boldsymbol{y}^*$, step 8 requires $\mathcal{O}(N)$ per iteration. To evaluate the complexity of updating $\boldsymbol{x}^*$, we first assume the number of constraints in problem $\mathbb{P}_1$ is C and each $A_n(\boldsymbol{x})$ and $B_n(\boldsymbol{x})$ evaluation has complexity $\mathcal{O}(M)$ where M represents the dimension of $\boldsymbol{x}$. Then calculating the gradient $\nabla_{\boldsymbol{x}} W(\boldsymbol{x}, \boldsymbol{y})$ has complexity $\mathcal{O}(N \cdot M)$ if $A_n(\boldsymbol{x})$ and $B_n(\boldsymbol{x})$ are linear or sparse forms. Calculating the gradients of constraints also contributes $\mathcal{O}(C \cdot M)$. Then, the total cost of gradient computation is $\mathcal{O}((N+C) \cdot M)$. Based on paper [28] and analysis of Appendix C and D, we approximate K as the running times, then the complexity of obtaining $\boldsymbol{x}^{(j)}$ through KKT analysis is $\mathcal{O}(K \cdot (M + C))$ as the KKT conditions involve a system of $M + C$ equations (from stationarity and primal feasibility) and $M+C$ unknowns (i.e., M optimization variable and N multipliers). Consequently, the worst complexity of Algorithm 5 is $\mathcal{O}(J \cdot ((N+C+K) \cdot M + K \cdot C + N))$, where J is the maximum iteration number.

Algorithm 5: Algorithm of Adopting UP Transform

Input: Optimization problem $\mathbb{P}_1$, feasible set $\mathcal{X}$, optimization variable $\boldsymbol{x}$, and auxiliary variables $\boldsymbol{y}$;
Output: Optimal solutions $\boldsymbol{x}^*$ and auxiliary variables $\boldsymbol{y}^*$;
Randomly set the value of optimization variable $\boldsymbol{x}^{(0)}$ with $\boldsymbol{x}^{(0)} \in \mathcal{X}$;
Set the iteration number $j = 0$;
Initialize the current optimal solution $\boldsymbol{x}^* = \boldsymbol{x}^{(0)}$;
Initialize optimal auxiliary variables $\boldsymbol{y}^*$;
repeat
  Calculate values of the corresponding auxiliary variables $\boldsymbol{y}^{(j)}$ with $\boldsymbol{x}^*$ based on (10);
  Update optimal auxiliary variables $\boldsymbol{y}^* = \boldsymbol{y}^{(j)}$;
  Update the iteration number $j = j + 1$;
  Solve the problem $\mathbb{P}_2$ with given $\boldsymbol{y}^*$ by adopting KKT analysis;
  Obtain the current solution $\boldsymbol{x}^{(j)}$;
  Update the optimal solution $\boldsymbol{x}^* = \boldsymbol{x}^{(j)}$;
until *Convergence or Maximum iteration number* J;

B. Complexity Comparison

In this subsection, we compare the computation complexities of our updating $\boldsymbol{x}^{(j)}$ algorithm with other traditional methods, consisting of Gradient Descent, Newton's Method, and the Interior-Point Method. The comprehensive comparison is concluded in Table I, and each method is analyzed as follows:

Gradient Descent [29], as a first-order method, iteratively updates the variable by moving a certain amount $\eta > 0$ in the direction $-\nabla f(\boldsymbol{x}^{(k)})$ of the steepest descent of the approximated objective function, where the parameter η represents the learning rate [30]. The update rule is given by

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \eta \nabla_{\boldsymbol{x}} W(\boldsymbol{x}, \boldsymbol{y}), \text{and } \boldsymbol{x}^{(j)} \leftarrow \boldsymbol{x}^{(K_{GD})},$$

where k represents the iteration number within gradient method and K_{GD} is the maximum gradient descent iterations required for convergence of $\boldsymbol{x}^{(j)}$. When adopting gradient descent to update $\boldsymbol{x}^{(j)}$, the complexity per iteration is $\mathcal{O}((N + C) \cdot M)$ accounting for the gradient computation $W(\boldsymbol{x}, \boldsymbol{y})$. Considering another complexity for updating $\boldsymbol{y}$, the total complexity of the gradient method is derived as $\mathcal{O}(J_{GD} \cdot (K_{GD} \cdot (N + C) \cdot M + N))$, where J_{GD} denotes the total number of outer iterations.

Newton's Method [31] is a second-order optimization technique that determines the descent direction of movement by focusing on the second-order Taylor expansion of the objective function. Considering K_{NT} iterations of convergence of $\boldsymbol{x}^{(j)}$, the update rule is given by

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \eta [\nabla^2_{\boldsymbol{x}} W(\boldsymbol{x}, \boldsymbol{y})]^{-1} \nabla_{\boldsymbol{x}} (\boldsymbol{x}, \boldsymbol{y}),$$
$$\text{and } \boldsymbol{x}^{(j)} \leftarrow \boldsymbol{x}^{(K_{NT})},$$

where $\eta > 0$ is the learning rate. The gradient computation, as in the gradient descent method, has a complexity of $\mathcal{O}((N + C) \cdot M)$. Computing the Hessian matrix contributes $\mathcal{O}(M^2 \cdot (N+C))$, while inverting the Hessian matrix incurs an additional cost of $\mathcal{O}(M^3)$. Combining these, the complexity of updating $\boldsymbol{x}^{(j)}$ using Newton's method is $\mathcal{O}(M \cdot (N+C) \cdot (M+1) + M^3)$. Considering J_{NT} total outer iterations, the total complexity of adopting Newton's method, including updates for $\boldsymbol{y}$, is $\mathcal{O}(J_{NT} \cdot (K_{NT} \cdot (M(N+C)(M+1) + M^3) + N))$.

The **Interior-Point Method** (IPM) [32] is effective for solving constrained optimization, especially when convexity and smoothness can be leveraged. When adopting IPM, optimization problem $\mathbb{P}_2$ is reformulated to include a logarithmic barrier function for constraints:

$$W_{barrier}(\boldsymbol{x}, \boldsymbol{y}) = W(\boldsymbol{x}, \boldsymbol{y}) - \mu \sum_i \ln(c_i(\boldsymbol{x})),$$

where μ is the barrier parameter and $c_i(\boldsymbol{x})$ are the inequality constraints. In each iteration, the gradient computation involves evaluating $\nabla_{\boldsymbol{x}} W(\boldsymbol{x}, \boldsymbol{y})$, which has a complexity of $\mathcal{O}((N + C) \cdot M)$. The Hessian computation, which accounts for second-order derivatives of $W(\boldsymbol{x}, \boldsymbol{y})$ and the barriers terms, has a complexity of $\mathcal{O}((N + C) \cdot M^2)$, assuming linear or sparse structures for $A_n(\boldsymbol{x})$, $B_n(\boldsymbol{x})$, and $c_i(\boldsymbol{x})$. Solving the Newton system, involving the $M \times M$ Hessian matrix and gradient, requires $\mathcal{O}(M^3)$ operations. Thus, the total cost per iteration of the interior point method is $\mathcal{O}((N+C) \cdot M + (N+C) \cdot M^2 + M^3)$. The maximum iteration number of IPM, denoted as K_{IPM}, depends logarithmically on the desired precision ϵ and barrier parameter reduction, approximately $\mathcal{O}(1/\epsilon)$. Consequently, the total complexity of the algorithm adopting IPM is derived as $\mathcal{O}(J_{IPM} \cdot (N + K_{IPM} \cdot ((N + C) \cdot M + (N + C) \cdot M^2 + M^3)))$. For large M, the M^3 term dominates, simplifying the total complexity to $\mathcal{O}(J_{IPM} \cdot K_{IPM} \cdot (M^3 + (N + C) \cdot M^2))$.

The **comparison** of Gradient Descent, Newton's Method, Interior-Point Method, and our algorithm highlights the strengths of our proposed UP transform in solving optimization problems. Gradient Descent, with a per-iteration complexity of $\mathcal{O}((N+C) \cdot M)$, is simple and scales well for large dimensions M, but its slow convergence $\mathcal{O}(1/\epsilon^2)$ makes it inefficient for high-precision solutions. Newton's Method achieves faster quadratic convergence with a per-iteration complexity of $\mathcal{O}(M(N + C)(M + 1) + M^3)$, but the high cost of Hessian computation and solving linear systems limits its applicability to moderate-dimensional problems. The Interior-Point Method is highly effective for constrained convex problems, converging logarithmically in $\mathcal{O}(1/\epsilon)$ iterations; however, its high per-iteration complexity $\mathcal{O}(M^3 + (N + C) \cdot M^2)$ makes it computationally expensive for large M. In contrast, our algorithm offers a balance of efficiency and scalability by directly solving optimality conditions with a per-iteration complexity of $\mathcal{O}(M+C)$. This approach avoids the computational burden of M^3-scaling seen in Newton's and Interior-Point methods, making it particularly advantageous for problems where feasible solutions can be efficiently identified. The strengths of our UP transform with KKT Analysis lie in its ability to directly address optimality conditions, its relatively lower dependence on dimensionality M, and its suitability for constrained problems where solving the KKT equations is computationally viable.

V. Comparison with Related Work

In this section, we give comparisons of the representative papers on FP and MP techniques in communication networks

TABLE I: Computation complexity comparison of different optimization methods

Algorithm	Per-iteration Complexity	Total Complexity
Our Algorithm	$\mathcal{O}(M+C)$	$\mathcal{O}(J((N+C+K)M+KC+N))$
Gradient Descent	$\mathcal{O}((N+C)\cdot M)$	$\mathcal{O}(J_{GD}(K_{GD}(N+C)\cdot M+N))$
Newton's Method	$\mathcal{O}(M(N+C)(M+1)+M^3)$	$\mathcal{O}(J_{NT}(K_{NT}(M(N+C)(M+1)+M^3)+N))$
Interior Point Method	$\mathcal{O}(M(N+C)(M+1)+M^3)$	$\mathcal{O}(J_{IPM}K_{IPM}(M^3+(N+C)\cdot M^2))$

with our paper and emphasize the effectiveness of our proposed transforms.

A. Comparison with Related Studies

Comparing Our Theorem 1 with [2]. The result in the special case of convex $A_n(\boldsymbol{x})$, concave $C_n(\boldsymbol{x})$, convex $G(\boldsymbol{x})$, and convex set $\mathcal{X}$ has been presented in Section IV of work [2]. Theorem 1 in this paper does not impose the special requirements above and hence is useful for solving a wider set of optimization problems compared with [2]. Note that although we consider $\boldsymbol{x}^{(j)}$ as a KKT point of Problem $\mathbb{P}_2(\boldsymbol{y}^{(j)})$ in "②", this may still hold when $A_n(\boldsymbol{x})$ is not convex, $C_n(\boldsymbol{x})$ is not concave (resp., $B_n(\boldsymbol{x})$ is not convex), $G(\boldsymbol{x})$ is not convex, or set $\mathcal{X}$ is not convex, because KKT conditions may still hold for certain non-convex optimization problems [33].

Comparing Our Theorem 2 with [2]. From the above, we present Theorem 1 as a general transform form of the minimization optimization problem and always holds when the introduced auxiliary variable is not zero (i.e., $\boldsymbol{y} \neq \mathbf{0}$). Therefore, the comparison between Theorem 2 and [2] is inherently a comparison between Theorem 1 and Theorem 2. With the introduction of Theorem 2, it can be adapted to the more general minimization problem, i.e., Theorem 2 can be adapted to the case of $\boldsymbol{y} = \mathbf{0}$ while also being applicable to the cases mentioned by Theorem 1.

Comparison with [34], [35]. Not like the `UpperBound` transform, the *quadratic transform* proposed in [34], [35] can't be applied in the minimization case when $A_n(\boldsymbol{x})$ is convex and $B_n(\boldsymbol{x})$ is concave in Theorem 1. The minimization, in this case, would be $-\infty$ when using the *quadratic transform*. Besides, some minor issues arise from "*Corollary 1*" in [34] regarding its accuracy. This corollary posits an equivalence between the solutions of the original sum-of-ratios problem and those derived via the *quadratic transform*. However, this assertion is flawed because the properties of the solutions found through the *quadratic transform* differ significantly from those of the original problem. The *quadratic transform* primarily identifies a KKT point, which does not necessarily coincide with the optimal point of the original sum-of-ratios problem. Therefore, declaring these two problems as equivalent is incorrect.

Comparison with [36]. The authors in [36] proposed a methodology for identifying the global optimum solution to the sum-of-ratios problem under the condition that $G(x) = 0$ in Theorem 1. But this method fails in cases where $G(x) \neq 0$. In contrast, our theorems still hold when $G(x) \neq 0$.

B. Other Related Work

We introduce representative studies on communication networks by adopting the alternating optimization (AO) method which fixes part of the optimization variable. The overview of these six papers is given in Table II. It can be observed that most wireless communication scenarios, especially concerning the associations within unmanned aerial vehicles (UAVs) [37]–[39], ground user equipment (GUE) and high altitude platforms (HAPs) [40], and UAV-mounted base stations (UBSs) [41], and other end-to-end (E2E) devices [42], are not directly solved by starting with transforming the objective function and all the constraints of the original optimization problems. Rather, the optimization problems are solved by alternatively assuming that some of the optimization variable is fixed and analyzing the constraints associated with the remaining optimization variable. Such an idea can find the corresponding optimal solution, but it increases the complexity and computation time of the algorithm, especially when each sub-problem needs to be further solved with a distinct algorithm(e.g., brute-force search, gradient ascent method, and quadratic transform method). Therefore, our proposed algorithms (`UP` transform and `UpperBound` transform) can be applied to these papers' algorithms to solve the drawbacks brought by the current alternating optimization of objective variables and provide researchers with a novel scheme of finding the optimal solution by optimizing all optimization variable together.

TABLE II: Overview of representative papers adopting the AO method to find the optimal solutions.

Paper	Optimization variables which are optimized alternatively
[37]	User association $\boldsymbol{A}$; Computation capacity, power, and 3D location planning $\{\boldsymbol{F}, \boldsymbol{P}, \boldsymbol{Z}\}$
[38]	UAV scheduling and association$\{\boldsymbol{A}, \boldsymbol{Q}\}$; Transmitting power $\boldsymbol{P}$
[39]	User association $\boldsymbol{\mu}$; Transmission power allocation $\boldsymbol{p}$; Location of UAV $\boldsymbol{f}$
[40]	GUE association strategy $\boldsymbol{A}$; Transmit precoding matrix $\{\mathbb{Q}, \mathbb{W}\}$ of GUE and HAP
[41]	UBS resource allocation β; UBS placement $\boldsymbol{x}^{\boldsymbol{ubs}}$, and user association $\boldsymbol{z}$
[42]	E2E association $\boldsymbol{x}$; Allocation of uplink and downlink transmit bandwidth, computation, power, and time duration $\{\bar{\boldsymbol{b}}, \underline{\boldsymbol{b}}, \tilde{\boldsymbol{f}}, \underline{\boldsymbol{p}}, \boldsymbol{\tau}\}$

VI. Conclusion

In this paper, we introduced the generalized `UP` transform to overcome the limitations of the original `UpperBound` transform in solving complex non-convex optimization problems characterized by multiple ratio structures. Through rigorous theoretical analysis, we established that our proposed `UP` transform reliably converges to a KKT point. Extensive simulations validated its superior performance, demonstrating significant improvements in convergence speed and solution quality in practical communication networks.

APPENDIX A
PROOF OF THEOREM 1

Since $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ converges to $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ as $j \to \infty$, we obtain from (7) that

$$y_n^* = \frac{B_n(\boldsymbol{x}^*)}{2A_n(\boldsymbol{x}^*)}. \tag{28}$$

Since $\boldsymbol{x}^{(j)}$ is a KKT point of $\mathbb{P}_2(\boldsymbol{y}^{(j)})$, and $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ converges to $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ as $j \to \infty$, we know that $\boldsymbol{x}^*$ is a KKT point of $\mathbb{P}_2(\boldsymbol{y}^*)$. Then supposing that $\boldsymbol{x} \in \mathcal{X}$ means

$$\begin{cases} \mathcal{Q}_q(\boldsymbol{x}) \le 0, & \text{for } q = 1, 2, \ldots, Q, \\ \mathcal{R}_r(\boldsymbol{x}) = 0, & \text{for } r = 1, 2, \ldots, R, \\ \boldsymbol{x} \in \mathbb{R}^M, \end{cases} \tag{29}$$

we have the following set of KKT conditions, where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ denote the Lagrange multipliers:

$$\begin{cases} \text{Stationarity:} \\ \Big[\frac{\partial}{\partial x_m}\big(W(\boldsymbol{x}, \boldsymbol{y}^*) + \sum_{q=1}^{Q} \alpha_q \mathcal{Q}_q(\boldsymbol{x}) \\ + \sum_{r=1}^{R} \beta_r \mathcal{R}_r(\boldsymbol{x})\big)\Big]|_{\boldsymbol{x}=\boldsymbol{x}^*} = 0, \text{for } m = 1, \ldots, M, & \text{(30a)} \\ \text{Primal feasibility:} \\ \mathcal{Q}_q(\boldsymbol{x}^*) \le 0, \text{ for } q = 1, 2, \ldots, Q, & \text{(30b)} \\ \mathcal{R}_r(\boldsymbol{x}^*) = 0, \text{ for } r = 1, 2, \ldots, R, & \text{(30c)} \\ \text{Dual feasibility:} \\ \alpha_q \ge 0 \text{ for } q = 1, 2, \ldots, Q, & \text{(30d)} \\ \text{Complementary slackness:} \\ \alpha_q \mathcal{Q}_q(\boldsymbol{x}^*) = 0 \text{ for } q = 1, 2, \ldots, Q. & \text{(30e)} \end{cases}$$

Recalling the expressions of $W(\boldsymbol{x}, \boldsymbol{y})$ in Equations (5) and (2) and $H(\boldsymbol{x})$ in (3), we can leverage (28) to show the equivalence between $\frac{\partial}{\partial x_m} W(\boldsymbol{x}, \boldsymbol{y}^*)|_{\boldsymbol{x}=\boldsymbol{x}^*}$ and $\frac{\partial}{\partial x_m} H(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^*}$ so that (30a) is equivalent to

$$\Big[\frac{\partial}{\partial x_m}\big(H(\boldsymbol{x}) + \sum_{q=1}^{Q} \alpha_q \mathcal{Q}_q(\boldsymbol{x}) + \sum_{r=1}^{R} \beta_r \mathcal{R}_r(\boldsymbol{x})\big)\Big]\Big|_{\boldsymbol{x}=\boldsymbol{x}^*} = 0,$$
$$\text{for } m = 1, 2, \ldots, M. \tag{31}$$

Clearly, (31) (30b) (30c) (30d) (30e) together mean that $\boldsymbol{x}^*$ is a KKT point for Problem $\mathbb{P}_1$ in (3). □

APPENDIX B
PROOF OF THEOREM 2

To prove Theorem 2, we first prove that

Lemma 1. *$(\boldsymbol{y}^{(j)}, \boldsymbol{x}^{(j)})|_{j=1,2,\ldots}$ in Theorem 2 are solving Problem $\mathbb{P}_2'$ defined below via alternating optimization (AO):*

$$\mathbb{P}_2'\text{: } \underset{\boldsymbol{x}, \boldsymbol{y}}{\textit{minimize}} \;\; W(\boldsymbol{x}, \boldsymbol{y})$$
$$\textit{subject to} \;\; \begin{matrix} \boldsymbol{x} \in \mathcal{X} \textit{ and} \\ y_n \ge c_n, \; \forall n \in \mathcal{N}, \end{matrix}$$

where the expression of $W(\boldsymbol{x}, \boldsymbol{y})$ is given via Equations (2) (5).

With $\mathbb{P}_2'(\boldsymbol{x})$ (resp., $\mathbb{P}_2'(\boldsymbol{y})$) denoting the problem of optimizing $\boldsymbol{y}$ given $\boldsymbol{x}$ (resp., optimizing $\boldsymbol{x}$ given $\boldsymbol{y}$) for $\mathbb{P}_2'$, it is straightforward to see that $\boldsymbol{y}^{(j)}$ given by (10) optimally solves $\mathbb{P}_2(\boldsymbol{x}^{(j-1)})$). In addition, $\mathbb{P}_2'(\boldsymbol{y})$ is the same as $\mathbb{P}_2(\boldsymbol{y})$ in (9), so $\boldsymbol{x}^{(j)}$ as a KKT point of $\mathbb{P}_2(\boldsymbol{y}^{(j)})$ is also a KKT point of $\mathbb{P}_2'(\boldsymbol{y}^{(j)})$. Based on the above, Lemma 1 is proved. Then we have the convergence of $(\boldsymbol{y}^{(j)}, \boldsymbol{x}^{(j)})|_{j=1,2,\ldots}$ due to the AO process of solving $\mathbb{P}_2'$.

With $(\boldsymbol{x}^{\#}, \boldsymbol{y}^{\#})$ denoting the convergence of $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ as $j \to \infty$, we know from (10) that $y_n^{\#}$ being the n-th dimension of $\boldsymbol{y}^{\#}$ becomes

$$y_n^{\#} = \max\left\{\frac{B_n(\boldsymbol{x}^{\#})}{2A_n(\boldsymbol{x}^{\#})}, \; c_n\right\}, \; \forall n \in \mathcal{N}. \tag{32}$$

Readers may observe the difference between (32) above and (28) in the proof of Theorem 1. Then, as elaborated below, the remaining steps to complete showing Theorem 2 are similar to those after (28) in Theorem 1's proof.

Since $\boldsymbol{x}^{(j)}$ is a KKT point of $\mathbb{P}_2'(\boldsymbol{y}^{(j)})$, and $(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})$ converges to $(\boldsymbol{x}^{\#}, \boldsymbol{y}^{\#})$ as $j \to \infty$, we know that $\boldsymbol{x}^{\#}$ is a KKT point of $\mathbb{P}_2'(\boldsymbol{y}^{\#})$. Then supposing that $\boldsymbol{x} \in \mathcal{X}$ means (29), we have the following set of KKT conditions, where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ denote the Lagrange multipliers:

$$\begin{cases} \text{Stationarity:} \\ \Big[\frac{\partial}{\partial x_m}\big(W(\boldsymbol{x}, \boldsymbol{y}^{\#}) + \sum_{q=1}^{Q} \alpha_q \mathcal{Q}_q(\boldsymbol{x}) \\ + \sum_{r=1}^{R} \beta_r \mathcal{R}_r(\boldsymbol{x})\big)\Big]|_{\boldsymbol{x}=\boldsymbol{x}^{\#}} = 0, \text{for } m = 1, \ldots, M, & \text{(33a)} \\ \text{Primal feasibility:} \\ \mathcal{Q}_q(\boldsymbol{x}^{\#}) \le 0, \text{ for } q = 1, 2, \ldots, Q, & \text{(33b)} \\ \mathcal{R}_r(\boldsymbol{x}^{\#}) = 0, \text{ for } r = 1, 2, \ldots, R, & \text{(33c)} \\ \text{Dual feasibility:} \\ \alpha_q \ge 0 \text{ for } q = 1, 2, \ldots, Q, & \text{(33d)} \\ \text{Complementary slackness:} \\ \alpha_q \mathcal{Q}_q(\boldsymbol{x}^{\#}) = 0 \text{ for } q = 1, 2, \ldots, Q. & \text{(33e)} \end{cases}$$

Recalling the expressions of $W(\boldsymbol{x}, \boldsymbol{y})$ in Equations (5) (2) and $L_{\boldsymbol{c}}(\boldsymbol{x})$ in (3), we can use (32) to show the equivalence between $\frac{\partial}{\partial x_m} W(\boldsymbol{x}, \boldsymbol{y}^{\#})|_{\boldsymbol{x}=\boldsymbol{x}^{\#}}$ and $\frac{\partial}{\partial x_m} L_{\boldsymbol{c}}(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^{\#}}$ so that (33a) is equivalent to

$$\Big[\frac{\partial}{\partial x_m}\big(L_{\boldsymbol{c}}(\boldsymbol{x}) + \sum_{q=1}^{Q} \alpha_q \mathcal{Q}_q(\boldsymbol{x}) + \sum_{r=1}^{R} \beta_r \mathcal{R}_r(\boldsymbol{x})\big)\Big]\Big|_{\boldsymbol{x}=\boldsymbol{x}^{\#}} = 0,$$
$$\text{for } m = 1, 2, \ldots, M. \tag{34}$$

Clearly, (33b) (33c) (33d) (33e) (34) together mean that $\boldsymbol{x}^{\#}$ is a KKT point for Problem $\mathbb{P}_1'$ in (11). □

APPENDIX C
PROOF OF THEOREM 3

Deriving from the functional expression $D_n(x_n)$ as delineated in (18a), two properties of this condition can be inferred contingent upon the values of auxiliary variables and Lagrange multipliers: *1). $D_n(x_n)$ is non-decreasing for x_n and 2). Specifically, we can obtain the explicit expression $\widetilde{x}_n = D_n^{-1}(0)$ by setting $\beta = 0$ and $\gamma_n = 0$ if $D_n(x_n) = 0$ has a solution.* Then we proceed to discuss the different cases based on conditions (19a) and (19b) which are outlined below:

- Case 1: $\widetilde{x}_n \ge 1$. In this case, we can infer that $D_n(x_n)$ is equal to or less than zero (i.e. $D_n(1) \le 0$). Therefore, the optimal solution can be set that $\widetilde{x}_n = 1$ and $\beta_n = 0$ to meet the conditions, and then the value of γ_n is equal to $-D_n(1)$ which exactly meets the condition (19b).
- Case 2: $0 < \widetilde{x}_n < 1$. We can simply set the optimal value of optimization variable x_n as $\widetilde{x}_n$ with $\beta_n = 0$ and $\gamma_n = 0$ based on the second property.
- Case 3: $\widetilde{x}_n \le 0$. Conversely with Case 1, $\widetilde{x}_n \le 0$ means when $x_n = 0$, $D_n(0)$ is equal to or better than 0. We choose the feasible solution as $\widetilde{x}_n = 0$,$\gamma_n = 0$, and $\beta_n = D_n(0)$.

Thus, we summarize all of the above cases as follows:

$$\begin{cases} x_n^* = \max\{\min\{\widetilde{x}_n, 1\}, 0\}; \\ \beta_n^* = \max\{D_n(0), 0\}; \\ \gamma_n^* = -\min\{D_n(1), 0\}; \end{cases} \tag{35}$$

Based on the convexity of the function $H_{n,1}(f_n^l)$, it can be inferred that the term $[H_{n,1}(f_n^l)]^2(u_n^{(k)} + c_n^{(k)})$ maintains convexity with the scalar composition theory (Page 84 in [43]). We next obtain the *non-decreasing nature* of the function $Q_n(f_n^l)$ as the decoupling of each optimization variable and being obtained by first-order derivation of the mentioned term. Then we can discuss the similar cases with x_n by assuming $\widetilde{f}_n^l = Q_n^{-1}(0)$ and summarize the discussion as follows:

$$\begin{cases} {f_n^l}^* = \max\{\min\{\widetilde{f}_n^l, F_n^l\}, 0\}; \\ \epsilon_n^* = \max\{Q_n(0), 0\}; \\ \zeta_n^* = -\min\{Q_n(F_n^l), 0\}; \end{cases} \tag{36}$$

According to the analysis of the optimization variable f_n^l, we obtain similar characteristics of the variable f_n^e in the function $R_n(f_n^e, \delta)$. Assume that when $R_n(f_n^e, \delta) = 0$, the value expression of f_n^e is denoted as $\widehat{f}_n^e(\delta)$. We discuss the cases based on the (18c):

- Case 1: $\widehat{f}_n^e(\delta) \geq F_n^e$. In this case, we can infer that $R_n(F_n^e, \delta) \leq 0$). Therefore, the optimal solution can be set that $\widehat{f}_n^e(\delta) = F_n^e$ and $\eta_n = 0$ to meet the conditions, and then the value of θ_n is $-R_n(F_n^e, \delta)$ which exactly meets the conditions.
- Case 2: $0 < \widehat{f}_n^e(\delta) < 1$. We can set the optimal value as $\widehat{f}_n^e(\delta)$ with $\eta_n = 0$ and $\theta_n = 0$ based on the second property.
- Case 3: $\widehat{f}_n^e(\delta) \leq 0$. Unlike Case 1, $\widehat{f}_n^e(\delta) \leq 0$ means when $\widehat{f}_n^e(\delta) = 0$, $R_n(0, \delta)$ is equal to or better than 0. Therefore, we choose the feasible solution as $\widehat{f}_n^e(\delta) = 0$, $\theta_n = 0$, and $\eta_n = R_n(0, \delta)$.

Therefore, for the above analysis, we summarise that:

$$\begin{cases} \widetilde{f}_n^e(\delta) = \max\{\min\{\widehat{f}_n^e(\delta), F_n^e\}, 0\}; \\ \widetilde{\eta}_n(\delta) = \max\{R_n(0, \delta), 0\}; \\ \widetilde{\theta}_n(\delta) = -\min\{R_n(F_n^e, \delta), 0\}; \end{cases} \tag{37}$$

For the optimization variable f_n^e, we cannot directly obtain the feasible solution while getting the formulation related to the multiplier δ. Drawing from the condition (19c), we discuss the value of δ based on the scenarios assuming $\delta = 0$:

- Case 1: $\sum_{n\in\mathcal{N}} \widetilde{f}_n^e(0) \leq F^e$. In this situation, we set $\delta = 0$ to meet the condition (20c).
- Case 2: $\sum_{n\in\mathcal{N}} \widetilde{f}_n^e(0) > F^e$. Conversely, we need to find the optimal value denoted as $\widetilde{\delta}$ of the δ based on the condition (19c) by adopting the bisection method.

Consequently, the feasible solution of δ is expressed as:

$$\delta^* = \begin{cases} 0, & \sum_{n\in\mathcal{N}} \widetilde{f}_n^e(0) \leq F^e, \\ \widetilde{\delta}, & \text{others.} \end{cases} \tag{38}$$

Until now, the optimal solution of the optimization variable $[\boldsymbol{x}^*, {\boldsymbol{f}^{\boldsymbol{l}}}^*, \boldsymbol{f}^{\boldsymbol{e}*}]$ and Lagrange multiplier $[\boldsymbol{\beta}^*, \boldsymbol{\gamma}^*, \delta^*, \boldsymbol{\epsilon}^*, \boldsymbol{\zeta}^*, \boldsymbol{\eta}^*, \boldsymbol{\theta}^*]$ can be obtained by above analysis. □

APPENDIX D
PROOF OF THEOREM 4

To decouple the optimization variable in each term of the objective function, we adopt the UP transform and introduce auxiliary variables in the $(k+1)$-th iteration as follows:

For $O_n(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{d})$, we introduce the auxiliary variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Then the function in the $(k+1)$-th iteration can be transformed into $\widetilde{O}_n(\boldsymbol{x}, \boldsymbol{f^l}, \boldsymbol{d} \mid \boldsymbol{\alpha}^{(k)}, \boldsymbol{\beta}^{(k)}, \boldsymbol{b}^{(k)}, \boldsymbol{e}^{(k)})$ which equals

$$\sum_{m\in\mathcal{M}} [\widehat{O}_{n,m}(\boldsymbol{f^l}, \boldsymbol{d} | \boldsymbol{\beta}^{(k)}, \boldsymbol{e}^{(k)})]^2(\alpha_{n,m}^{(k)} + b_{n,m}^{(k)}) + \frac{x_{n,m}^2 c_n}{4(\alpha_{n,m}^{(k)} + b_{n,m}^{(k)})},$$

where $\widehat{O}_{n,m}(\boldsymbol{f^l}, \boldsymbol{d} | \boldsymbol{\beta}^{(k)}, \boldsymbol{e}^{(k)}) = (\frac{w_1}{f_n^l} + w_2 k^l {f_n^l}^2)^2(\beta_{n,m}^{(k)} + e_{n,m}^{(k)}) + \frac{(d_n - d_{n,m})^2}{4(\beta_{n,m}^{(k)} + e_{n,m}^{(k)})}$. When $x_{n,m} \neq 0$ and $d_n - d_{n,m} \neq 0$, values of auxiliary variables are $\beta_{n,m}^{(k)} = \frac{d_n - d_{n,m}^{(k)}}{2[\frac{w_1}{f_n^{l\,(k)}} + w_2 k^l (f_n^{l\,(k)})^2]}$ and $\alpha_{n,m}^{(k)} = \frac{x_{n,m}^{(k)} c_n}{2\widehat{O}_{n,m}(\boldsymbol{f}^{\boldsymbol{l}\,(k)}, \boldsymbol{d}^{(k)} \mid \boldsymbol{\beta}^{(k)}, \boldsymbol{e}^{(k)})}$ and the selection of constant $\boldsymbol{b}^{(k)}$ and $\boldsymbol{e}^{(k)}$ follows the properties in Remark 3 and the following constants obey the same rules.

For function $S_n(\boldsymbol{x}, \boldsymbol{P}, \boldsymbol{d})$, it can be transformed as $\widetilde{S}_n(\boldsymbol{x}, \boldsymbol{P}, \boldsymbol{d} \mid \boldsymbol{\gamma}^{(k)}, \boldsymbol{\delta}^{(k)}, \boldsymbol{g}^{(k)}, \boldsymbol{h}^{(k)})$ which equals:

$$\sum_{m\in\mathcal{M}} [\widehat{S}_{n,m}(\boldsymbol{P}, \boldsymbol{d} \mid \boldsymbol{\delta}^{(k)}, \boldsymbol{h}^{(k)})]^2(\gamma_{n,m}^{(k)} + g_{n,m}^{(k)}) + \frac{x_{n,m}^2}{4(\gamma_{n,m}^{(k)} + g_{n,m}^{(k)})},$$

where $\widehat{S}_{n,m}(\boldsymbol{P}, \boldsymbol{d} | \boldsymbol{\delta}^{(k)}, \boldsymbol{h}^{(k)}) = [\overline{S}(P_n)]^2(\delta_{n,m}^{(k)} + h_{n,m}^{(k)}) + \frac{d_{n,m}^2}{4(\delta_{n,m}^{(k)} + h_{n,m}^{(k)})}$ and $\overline{S}(P_n) = \frac{w_1 + w_2 P_n}{B_m \ln(1 + \frac{P_n H_{n,m}}{\sigma^2})}$. $\boldsymbol{g}^{(k)}, \boldsymbol{h}^{(k)}$ are expressed as:

$$\delta_{n,m}^{(k)} = \frac{d_{n,m}^{(k)}}{2\overline{S}(P_n^{(k)})}, \gamma_{n,m}^{(k)} = \frac{x_{n,m}^{(k)}}{2\widehat{S}_{n,m}(\boldsymbol{P}^{(k)}, \boldsymbol{d}^{(k)} \mid \boldsymbol{\delta}^{(k)}, \boldsymbol{h}^{(k)})},$$

when the constants are equal to zero.

For function $U_n(\boldsymbol{x}, \boldsymbol{f^s}, \boldsymbol{d})$, we introduce the auxiliary $\boldsymbol{\epsilon}$ and $\boldsymbol{\zeta}$ to transform it to $\widetilde{U}_n(\boldsymbol{x}, \boldsymbol{f^s}, \boldsymbol{d} \mid \boldsymbol{\epsilon}^{(k)}, \boldsymbol{\zeta}^{(k)}, \boldsymbol{j}^{(k)}, \boldsymbol{l}^{(k)})$ being equal to $[\widehat{U}_n(\boldsymbol{f^s}, \boldsymbol{d} \mid \boldsymbol{\zeta}^{(k)}, \boldsymbol{l}^{(k)})]^2(\epsilon_n^{(k)} + j_n^{(k)}) + \frac{x_{n,1}^2}{4(\epsilon_n^{(k)} + j_n^{(k)})}$, where the constants are denoted as $\boldsymbol{j}^{(k)}$ and $\boldsymbol{l}^{(k)}$. Function $\widehat{U}_n$ is expressed as:

$$\widehat{U}_n(\boldsymbol{f^s}, \boldsymbol{d} | \boldsymbol{\zeta}^{(k)}, \boldsymbol{l}^{(k)}) = (\frac{w_1}{f_n^s} + w_2 k^S {f_n^s}^2)^2(\zeta_n^{(k)} + l_n^{(k)}) + \frac{(d_{n,1} - d_{n,1}^{'})^2}{4(\zeta_n^{(k)} + l_n^{(k)})}.$$

Based on the Eq (10), if constants are equal to 0, auxiliary variables can be inferred as:

$$\zeta_n^{(k)} = \frac{d_{n,1}^{(k)} - d_{n,1}^{'(k)}}{2\left[\frac{w_1}{f_n^{s(k)}} + w_2 k^S (f_n^{s(k)})^2\right]}, \epsilon_n^{(k)} = \frac{x_{n,1}^{(k)}}{2\widehat{U}_n(\boldsymbol{f}^{\boldsymbol{s}(k)}, \boldsymbol{d}^{(k)} \mid \boldsymbol{\zeta}^{(k)}, \boldsymbol{l}^{(k)})}.$$

For the last function $V_n(\boldsymbol{x}, \boldsymbol{d})$, it can be similarly transformed to $\widetilde{V}_n(\boldsymbol{x}, \boldsymbol{d} \mid \boldsymbol{\eta}^{(k)}, \boldsymbol{\theta}^{(k)}, \boldsymbol{q}^{(k)}, \boldsymbol{z}^{(k)})$, where $\boldsymbol{\eta}$ and $\boldsymbol{\theta}$ are auxiliary variables, and $\boldsymbol{q}$ and $\boldsymbol{z}$ are the constants. Then the details of $\widetilde{V}_n$ is: $\widetilde{V}_n = [d_{n,1}^{'2}(\eta_n^{(k)} + q_n^{(k)}) + \frac{x_{n,1}^2}{4(\eta_n^{(k)} + q_n^{(k)})}]C + [d_{n,2}^2(\theta_n^{(k)} + z_n^{(k)}) + \frac{x_{n,2}^2}{4(\theta_n^{(k)} + z_n^{(k)})}](C - \frac{w_1 + w_2 \bar{P}}{r_0})$. When auxiliary variables are not equal to zero, auxiliary variables are expressed as $\eta_n^{(k)} = \frac{x_n^{(k)}}{2d_{n,1}^{'(k)}}$ and $\theta_n^{(k)} = \frac{x_{n,2}^{(k)}}{2d_{n,2}^{(k)}}$.

Until now, the optimization variable in the term $w_1 T_n + w_2 E_n$ of the objective function in the Eq (25) is transformed into a convex function by using the UP transform. □

REFERENCES

[1] Z. Wang, Y. Wang, and J. Zhao, "Resource Allocation and Secure Wireless Communication in the Large Model based Mobile Edge Computing System," in *MobiHoc '24*, 2024, p. 111–120. [Online]. Available: https://doi.org/10.1145/3641512.3686373

[2] J. Zhao, L. Qian, and W. Yu, "Human-Centric Resource Allocation in the Metaverse Over Wireless Communications," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 42, no. 3, pp. 514–537, 2024.

[3] K. Shen and W. Yu, "Fractional Programming for Communication Systems—Part I: Power Control and Beamforming," *IEEE Transactions on Signal Processing*, vol. 66, no. 10, pp. 2616–2630, 2018.

[4] A. Seetharam, B. Jiang, D. Goeckel, J. Kurose, and R. Hancock, "Optimizing Control Overhead for Power-Aware Routing in Wireless Networks," in *MILCOM 2013-2013 IEEE Military Communications Conference*. IEEE, 2013, pp. 870–875.

[5] H. Konno and T. Kuno, "Multiplicative Programming Problems," *Handbook of Global Optimization*, pp. 369–405, 1995.

[6] D. Malak, F. V. Mutlu, J. Zhang, and E. M. Yeh, "Joint power control and caching for transmission delay minimization in wireless hetnets," *IEEE/ACM Transactions on Networking*, vol. 32, no. 2, pp. 1477–1492, 2024.

[7] Y. Deng, F. Lyu, T. Xia, Y. Zhou, Y. Zhang, J. Ren, and Y. Yang, "A communication-efficient hierarchical federated learning framework via shaping data distribution at edge," *IEEE/ACM Transactions on Networking*, vol. 32, no. 3, pp. 2600–2615, 2024.

[8] R. Liu, K. Guo, K. An, F. Zhou, Y. Wu, Y. Huang, and G. Zheng, "Resource allocation for noma-enabled cognitive satellite–uav–terrestrial networks with imperfect csi," *IEEE Transactions on Cognitive Communications and Networking*, vol. 9, no. 4, pp. 963–976, 2023.

[9] S. Schaible and T. Ibaraki, "Fractional Programming," *European journal of operational research*, vol. 12, no. 4, pp. 325–338, 1983.

[10] J. C. Bezdek and R. J. Hathaway, "Some Notes on Alternating Optimization," in *Advances in Soft Computing—AFSS 2002: 2002 AFSS International Conference on Fuzzy Systems Calcutta, India, February 3–6, 2002 Proceedings*. Springer, 2002, pp. 288–300.

[11] M. Asim, M. ELAffendi, and A. A. A. El-Latif, "Multi-IRS and Multi-UAV-assisted MEC system for 5G/6G networks: Efficient joint trajectory optimization and passive beamforming framework," *IEEE Transactions on Intelligent Transportation Systems*, pp. 4553–4564, Apr. 2023.

[12] Q. Li, A. Nayak, Y. Zhang, and F. R. Yu, "A Cooperative Recharging-Transmission Strategy in Powered UAV-Aided Terahertz Downlink Networks," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 4, pp. 5479–5484, Apr. 2023.

[13] U. Saleem, Y. Liu, S. Jangsher, X. Tao, and Y. Li, "Latency Minimization for D2D-Enabled Partial Computation Offloading in Mobile Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4472–4486, 2020.

[14] S. Zawad, X. Ma, J. Yi, C. Li, M. Zhang, L. Yang, F. Yan, and Y. He, "FedCust: Offloading Hyperparameter Customization for Federated Learning," *Performance Evaluation*, vol. 167, p. 102450, 2025.

[15] X. Qin, Q. Xie, and B. Li, "Distributed Threshold-Based Offloading for Heterogeneous Mobile Edge Computing," in *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2023, pp. 202–213.

[16] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, Opportunities, and Directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[17] Y. Huang, T. Sun, and T. He, "Overlay-based decentralized federated learning in bandwidth-limited networks," in *Proceedings of the Twenty-fifth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2024, pp. 121–130.

[18] W. Aljoby, X. Wang, D. M. Divakaran, T. Z. Fu, and R. T. Ma, "Diffperf: An in-network performance optimization for improving user-perceived qoe," in *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*. IEEE, 2021, pp. 288–292.

[19] M. S. Bute, P. Fan, L. Zhang, and F. Abbas, "An Efficient Distributed Task Offloading Scheme for Vehicular Edge Computing Networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 13 149–13 161, 2021.

[20] X. Zhou, C. Liu, and J. Zhao, "Resource Allocation of Federated Learning for the Metaverse with Mobile Augmented Reality," *IEEE Transactions on Wireless Communications*, 2023.

[21] H. Xiao, J. Zhao, J. Feng, L. Liu, Q. Pei, and W. Shi, "Joint Optimization of Security Strength and Resource Allocation for Computation Offloading in Vehicular Edge Computing," *IEEE Transactions on Wireless Communications*, 2023.

[22] X. Shang, "Enabling data-intensive workflows in heterogeneous edge-cloud networks," *ACM SIGMETRICS Performance Evaluation Review*, vol. 50, no. 3, pp. 36–38, 2023.

[23] Y. Li, H. Ma, L. Wang, S. Mao, and G. Wang, "Optimized Content Caching and User Association for Edge Computing in Densely Deployed Heterogeneous Networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 6, pp. 2130–2142, Jun. 2022.

[24] M. R. Hossen and M. A. Islam, "Mobile task offloading under unreliable edge performance," *ACM SIGMETRICS Performance Evaluation Review*, vol. 48, no. 4, pp. 29–32, 2021.

[25] C. Wang, D. Deng, L. Xu, and W. Wang, "Resource Scheduling Based on Deep Reinforcement Learning in UAV Assisted Emergency Communication Networks," *IEEE Transactions on Communications*, vol. 70, no. 6, pp. 3834–3848, 2022.

[26] M. A. Qureshi, A. Nika, and C. Tekin, "Multi-User Small Base Station Association via Contextual Combinatorial Volatile Bandits," *IEEE Transactions on Communications*, vol. 69, no. 6, pp. 3726–3740, 2021.

[27] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-Minimization Algorithms in Signal Processing, Communications, and Machine Learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 3, pp. 794–816, 2016.

[28] Y. Li, W. Yu, and J. Zhao, "PrivTuner with Homomorphic Encryption and LoRA: A P3EFT Scheme for Privacy-Preserving Parameter-Efficient Fine-Tuning of AI Foundation Models," *arXiv preprint arXiv:2410.00433*, 2024.

[29] L. Su and J. Xu, "Securing Distributed Gradient Descent in High Dimensional Statistical Learning," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 1, pp. 1–41, 2019.

[30] X. Gu, K. Lyu, L. Huang, and S. Arora, "Why (and when) does local sgd generalize better than sgd?" *The 14th International OPT Workshop on Optimization for Machine Learning*, 2022.

[31] C. T. Kelley, *Solving nonlinear equations with Newton's method*. SIAM, 2003.

[32] F. A. Potra and S. J. Wright, "Interior-point methods," *Journal of computational and applied mathematics*, vol. 124, no. 1-2, pp. 281–302, 2000.

[33] F. Flores-Bazán and G. Mastroeni, "Characterizing FJ and KKT Conditions in Nonconvex Mathematical Programming with Applications," *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 647–676, 2015.

[34] K. Shen and W. Yu, "Fractional Programming for Communication Systems—Part I: Power Control and Beamforming," *IEEE Transactions on Signal Processing*, vol. 66, no. 10, pp. 2616–2630, 2018.

[35] ——, "Fractional Programming for Communication Systems—Part II: Uplink Scheduling via Matching," *IEEE Transactions on Signal Processing*, vol. 66, no. 10, pp. 2631–2644, 2018.

[36] Y. Jong, "An efficient global optimization algorithm for nonlinear sum-of-ratios problem," *Optimization Online*, pp. 1–21, 2012.

[37] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy Efficient Resource Allocation in UAV-Enabled Mobile Edge Computing Networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, pp. 4576–4589, 2019.

[38] Q. Wu, Y. Zeng, and R. Zhang, "Joint Trajectory and Communication Design for Multi-UAV Enabled Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.

[39] F. Pervez, L. Zhao, and C. Yang, "Joint User Association, Power Optimization and Trajectory Control in an Integrated Satellite-Aerial-Terrestrial Network," *IEEE Transactions on Wireless Communications*, vol. 21, no. 5, pp. 3279–3290, 2021.

[40] C. Ding, J.-B. Wang, H. Zhang, M. Lin, and G. Y. Li, "Joint Optimization of Transmission and Computation Resources for Satellite and High Altitude Platform Assisted Edge Computing," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 1362–1377, 2021.

[41] C. Qiu, Z. Wei, X. Yuan, Z. Feng, and P. Zhang, "Multiple UAV-Mounted Base Station Placement and User Association With Joint Fronthaul and Backhaul Optimization," *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5864–5877, 2020.

[42] Y. Sun, J. Xu, and S. Cui, "User Association and Resource Allocation for MEC-Enabled IoT Networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 8051–8062, 2022.

[43] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004.