A Parallel Optimal Task Allocation Mechanism for Large-Scale Mobile Edge Computing

Xiaoxiong Zhong, Member, IEEE, Xinghan Wang, Yuanyuan Yang, Fellow, IEEE,
Yang Qin, senior Member, IEEE, Xiaoke Ma, Tingting Yang, Member, IEEE

Abstract—We consider the problem of intelligent and efficient task allocation mechanism in large-scale mobile edge computing (MEC), which can reduce delay and energy consumption in a parallel and distributed optimization. In this paper, we study the joint optimization model to consider cooperative task management mechanism among mobile terminals (MT), macro cell base station (MBS), and multiple small cell base station (SBS) for large-scale MEC applications. We propose a parallel multi-block Alternating Direction Method of Multipliers (ADMM)-based method to model both requirements of low delay and low energy consumption in the MEC system which formulates the task allocation under those requirements as a nonlinear 0-1 integer programming problem. To solve the optimization problem, we develop an efficient combination of conjugate gradient, Newton and linear search techniques based algorithm with Logarithmic Smoothing (for global variables updating) and the Cyclic Block coordinate Gradient Projection (CBGP, for local variables updating) methods, which can guarantee convergence and reduce computational complexity with a good scalability. Numerical results demonstrate the effectiveness of the proposed mechanism and it can effectively reduce delay and energy consumption for a large-scale MEC system.

Index Terms—large scale mobile edge computing; optimal task allocation; ADMM

I. INTRODUCTION

With the rapid development of the wireless technology and Internet, more and more mobile terminals (MT) have different wireless access requirements for bandwidth and computing, which promotes mobile applications, such as online playing game, virtual reality, intelligent data processing and other new services continue to emerge [1-3]. However, these new mobile applications have high energy consumption and high latency, which pose a huge challenge for computing and battery capacity of the MT. Mobile Edge Computing (MEC) is a new promotion technology that supports cloud computing capabilities and edge service environments at the edge of cellular networks. The legacy base station (BS) is updated to a MEC-enabled base station (MEC-BS) by being equipped with a computing function (such as an MEC server), hence the MEC-BSs can implement MT capability enhancement, which can reduce MT application execution time and MT energy consumption. The development of these new applications and services is limited by the computing capability and battery of these MTs. If data consumption and computationally intensive tasks need to be offloaded to the cloud for execution, it can address the problem that MTs’ have a lack of computing capability. However, a relatively large delay will be induced when an MT connects to the cloud over a wireless network, which is not suitable for delay-sensitive tasks.

MEC allows MT to perform computational offloading to offload its computational tasks to the MEC-BS that overwrites it. When the execution of the task at the MEC-BS is completed, the MEC-BS returns the result of the task to the MT. Due to limited computing resources, MEC-BS is unable to provide unlimited computational offload services for all tasks in the MT within coverage. Therefore, how to effectively manage MEC-BS resources (e.g., energy and computational resource), maximizing system performance, is critical. On the other hand, if there are too many tasks to be uninstalled from the MT, the MEC-BS will still be overloaded. Hence, how to design an efficient task management scheme, which can reduce delay and energy consumption, is a challenging issue.

Distributed resource allocation can benefit from being deployed at the edge of network, reducing delay and energy consumption. However, distribution optimization in a parallel manner for performance improvement brings to light several challenges. How could we design an efficient parallel optimization mechanism for large-scale task scheduling? How could we guarantee convergence to a resource allocation scheme and performance optimality given task management problem?

To answer these questions, we present an optimal task allocation mechanism for three-tier model based edge computing system, which can make full use of resource among MT, small cell base station (SBS), and macro cell base station (MBS). The proposed mechanism can dispatch task in resource allocation among different MTs, SBS, and MBS.
featuring distributed optimization and efficient task allocation mechanism, to reach intelligent and efficient resource management in MEC. The contributions of this article are as follows:

In this paper, we exploit a three-tier model for task management, which can make full use of resource among MTs (local terminals, LTs), SBS and MBS, reducing delay and energy.

1) It is more difficult to guarantee convergence in multi-block ADMM, only if its optimal function is linear. In the proposed optimal scheme, we exploit regulation linear method to make the optimal function to be linear, guaranteeing its convergence and it can run in a parallel manner and its global and local variables can update simultaneously, which can reduce computational complexity for achieving good scalability.

2) Our joint optimization problem is a nonlinear 0-1 integer programming problem, which can be solved using Logarithmic Smoothing by modifying the inequality constrain as equality constrain in decision variables and smoothing function. Also, we prove the convergence of the proposed Logarithmic Smoothing based optimization scheme.

3) In order to obtain the optimal values of the global variables, we use Newton method to calculate the partial derivative of the global variables. For the purpose of accelerating the convergence of the global variables, we adopt the conjugate gradient to solve it. After obtaining the partial derivative of the global variables, we use linear search technique to get the optimal step and update the global variables.

4) In order to solve the sub-optimal problem, in which the objective function is convex with more variables updating simultaneously, we adopt the cyclic block coordinate gradient projection (CBGP) method to address the sub-optimal problem, which can guarantee convergence and reduce computational complexity.

5) We conduct extensive experiments to evaluate the performance of the proposed mechanism. With the numerical results, we show that the proposed method can guarantee the convergence and effectively reduce delay and energy consumption.

The remainder of this paper is organized as follows. We review the related work in Section II. In Section III, we describe the system model, energy model and computation model. We propose an improved parallel multi-block ADMM based task allocation framework for MEC in Section IV. We evaluate the performance of the proposed mechanism and provide illustrative results in Section V and conclude the paper in Section VI.

II. RELATED WORK

Recently, some optimization algorithm based resource management schemes have been proposed for MEC [4-17]. Chen et al. [4] developed an optimization framework for software defined ultra-dense networks, minimizing the delay and saving the energy, which includes two sub-optimization problems: task placement and resource allocation. Consider cooperation behaviors, Xu et al. [5] proposed an online algorithm for service caching and task offloading in MEC systems, which jointly considers service heterogeneity, system dynamics, spatial demand and distributed coordination. Similarly, for heterogeneous services, Tan et al. [6] formulated a virtual resource allocation problem, and proposed ADMM algorithm to solve it, with jointly considering user association, power control and resources allocation. Zhou et al. [7] formulated the virtual resource allocation strategy as a joint optimization problem and used ADMM to address it, considering virtualization, caching and computing. Considering edge nodes and mobile users with time-dependent requests, Zheng et al. [8] proposed a convergent and scalable Stackelberg game for edge caching, and used a Stackelberg game based ADMM to solve storage allocation game and user allocation game in a distributed manner. Wang et al. [9] presented a parallel optimization framework which leverages the vertical cooperation among devices, edge nodes and cloud servers, and the horizontal cooperation between edge nodes by ADMM method to address the resource allocation optimal problem. Zhou et al. [10] studied the energy-efficient workload offloading problem and propose a low-complexity distributed solution based on consensus alternating direction method of multipliers (ADMM) for vehicular networks edge computing service provisioning. In [11], Dai et al. proposed a novel two-tier computation offloading framework in heterogeneous networks, which formulated joint computation offloading and user association problem for multi-task MEC system to minimize overall energy consumption through computation resource allocation and transmission power allocation. Wang et al. [12] proposed a heterogeneous multi-layer MEC framework, where data that cannot be timely processed at the edge are allowed to be offloaded to the upper layer MEC servers and the cloud center. Their goal is minimizing the system latency, i.e., the total computing and transmission time on all layers for the data generated by the edge devices, by jointly coordinating the task assignment, computing, and transmission resources in the framework. Li et al. [13] studied collaborative cache allocation and task scheduling in edge computing and modeled the task scheduling problem as a weighted bipartite graph, which can reduce latency. In [14], Alameddine et al. mathematically formulated the dynamic task offloading and scheduling as a mixed integer program and exploited a logic based bender decomposition approach to efficiently address the problem to optimality for low latency IoT service in MEC. For the ultra-reliable low-latency requirements in mission-critical applications, Liu et al. [15] proposed a new system design, where probabilistic and statistical constraints are imposed on task queue lengths, by applying extreme value theory and marrying tools from Lyapunov optimization and matching theory for user-server association, and dynamic task offloading and resource allocation. Meng et al. [16] studied the online deadline-aware task dispatching and scheduling in edge computing, which jointly considers with bandwidth constraint using joint optimization of networking and
computing resource to meet the deadlines and proposed an online algorithm Dedas, which greedily schedules newly arriving tasks and considers whether to replace some existing tasks in order to make the new deadlines satisfied. In [17], Kim et al. proposed dual-side control algorithms for cost-delay tradeoff in MEC, including two optimization problems where the objective is to minimize costs subject to queue stability under two scenarios: a competition scenario and a cooperation scenario.

Most of the existing resource allocation algorithms in MEC were proposed based on the environment without large-scale model and they perform well only when the assumptions do hold. However, the resource allocation algorithms in real-world large-scale MEC are often too complex to be designed by traditional cloud computing. Till now, there is still little attention being paid on applying optimization algorithms for resource management of large-scale MEC in a parallel manner, which will cause a poor system performance. Hence, how to design an efficient parallel task allocation scheme for large-scale MEC is a challenging issue. This paper aims to propose a solution to address this problem.

III. MODEL FOR TASK ALLOCATION MECHANISM

In this section, we will describe the system model, computation model and energy consumption model for the proposed task allocation mechanism in MEC.

1. System model

In this paper, we consider a large scale mobile edge computing system, which is composed of many groups, and we denote the set of groups as \( GR = \{g_1, g_2, ..., g_s\} \), each group is made up of a bunch of BS, including an MBS and some SBS, denoted as \( BS = \{h_1, h_2, ..., h_i\} \), where \( h \) represent an MBS and each element of set \( \{h_i, ..., h_j\} \) represents a SBS, each BS is equipped with MEC-server.

Each task that enters the coverage of group will be processed in local or in there-tier computation model or in MBS. Each task can be separated into subtasks. The set of tasks is \( H = \{h_1, h_2, ..., h_n\} \), each task \( h \) can be expressed through a vector \( h = (c, h_{\text{max}}, u) \), \( c \) denotes the data size of the task, \( h_{\text{max}} \) is the deadline of \( h \), and \( u \) is the number of CPU cycle for computing one bit of task \( h \).

In order to make task algorithm suitable for the realistic large-scale environment, and leverage a large number of the computation server resources, we adopt a parallel approach of local computation, MBS computation and there tier computation, which jointly considers path in resource allocation mechanism, as shown in Fig. 1. The path 1 \{task3, Local3, Virtual destination\} is an example for local computation model, denoted by the blue line. The path 2 \{task1, Local1, AP_1, SBS1, FU_2, MBS, Virtual destination\} is an example for three-tier computing model, denoted by the red line (also, the paths \{task2, AP_2, SBS2, FU_3, MBS, Virtual destination\}, \{task1, Local1, AP_1, FU_1, MBS, Virtual destination\} and \{task3, Local3, AP_3, SBS3, Virtual destination\} are examples for three-tier computing model). The path 3 \{task1, SBS1, Virtual destination\} is an example for SBS computing model, denoted by the green line. The path 4 \{task2, MBS, Virtual destination\} is an example for MBS computing model, denoted by the purple line. If an adjacent unit help local task transmit data, we call adjacent unit the parent of local task. By extension, the path can be defined as \( \text{Path } h_j\rightarrow SBS/MBS/Local = (\{\text{task}, \text{link}^1, \text{forwarding}^1(h_j)\}, \{\text{forwarding}^2(h_j), \text{link}^2, \text{forwarding}^3(h_j)\}, \ldots \{\text{forwarding}^{n-1}(h_j), \text{link}^{n-1}, SBS/MBS/Local\}) \).

The large-scale edge computing network can be described as a graph \( G = (V, E) \), where \( V \) and \( E \) denote vertices and edges, respectively. The vertices are computing units and forwarding units, in which computing units (like MEC server deployed on SBS or MBS) provide computing services, and forwarding units (like access point and switch) provide directing function. A virtual destination is introduced in order to let all the tasks flow to the same destination. Virtual destination is a sign of the end of tasks processing. Each vertex is associated with a source and a sink pair \( \{V_{\text{source}}, V_{\text{destination}}\} \). We use \( f \) to denote a path that traversing the vertex \( v \) and let \( F = \bigcup f \) to denote the feasible paths from local terminal to processing node in the large-scale edge computing. A path \( f \) consists of a sequence of links, a series forwarding units, a source and a processing node. We use the \( Y_{h_j} = \{y_{h_j} / f \in F \} \) represents possible paths of the part of task \( h_j \) indexed by \( F \).

So we can conclude the aggregation load of each forwarding unit (i.e., load_{forwarding}) and each link load_{link}.

\[
\text{load}_{\text{forwarding}} = \sum_{h_j \in \text{forwarding}^i} \sum_{f \in \text{forwarding}^i} y_{h_j} f c_j \quad \forall \text{forwarding} \in V(1)
\]

and
\( \text{load}_{\text{link}} = \sum_{k \in \text{link}_{i}} \sum_{f \in \text{link}_{i}} y_{k,f}c_{j} \quad \forall \text{link} \in \mathcal{E} \) \hspace{1cm} (2)

We adopt a linear latency model since buffer utilization and packet loss increases as the bit rate grows [18]. Hence, we can get the latency of the forwarding unit \( \text{Delay}_{\text{forwarding}} \) and the overall transmission latency on wireless channels \( \text{Delay}_{\text{link}} \), which can be expressed as:

\[
\text{Delay}_{\text{forwarding}} = (\alpha_{1}\text{load}_{\text{forwarding}} + \alpha_{2}) \times \text{load}_{\text{forwarding}} \quad (3)
\]

and

\[
\text{Delay}_{\text{link}} = \frac{\text{load}_{\text{link}}}{\text{capacity}_{\text{link}}} \quad (4)
\]

where \( \alpha_{1} \) and \( \alpha_{2} \) are parameters for linear model, which are related to forwarding unit, \( \text{capacity}_{\text{link}} \) is the capacity of link.

2. Local computing model & energy consumption model

First, the local process time should be defined according to the following equation:

\[
T_{\text{local}}^{(h_{i})} = z_{j} \frac{c_{j}H_{j}}{f_{\text{local}}} \quad (5)
\]

For local task computing, we define the local energy consumption model:

\[
E_{\text{local}}^{(h_{i})} = c_{j}H_{j}e_{\text{local}} \quad (6)
\]

where \( f_{\text{local}} \) is the process capability of local devices, \( z_{j} \) is a binary variable and has \( z_{j} \in \{0,1\} \), it has \( z_{j}=1 \) if the task \( h_{i} \) decides to process locally; it has \( z_{j}=0 \) if the task is not processed locally and \( e_{\text{local}} \) is the energy consumption per CPU cycle of local device.

3. MBS computation model

In this paper, we describe the computation model from a delay perspective. The overall delay in MBS \( h_{i} \) can be expressed as:

\[
T_{\text{MBS}}^{(h_{i})} = T_{\text{up}}^{(h_{i})} + T_{\text{execute}}^{(h_{i})} \quad (7)
\]

where \( T_{\text{up}}^{(h_{i})} \) is the uploading time, and \( T_{\text{execute}}^{(h_{i})} \) is the calculation time.

\[
T_{\text{up}}^{(h_{i})} = y_{u} \quad (8)
\]

where \( p^{(h_{i})} \) is the transmission power density of the MBS \( h_{i} \), \( G^{(h_{i})} \) is the channel gain between the local terminal and the MBS, \( \sigma^{2} \) is the power of the additive white Gaussian noise, \( y_{u} \) is a binary variable and has \( y_{u} \in \{0,1\} \), it has \( y_{u}=1 \) if the task \( h_{i} \) is decided to upload to MBS; it has \( y_{u}=0 \) if the task is not processed by the MBS.

The calculation time in MEC server deployed on MBS \( T_{\text{execute}}^{(h_{i})} \) can be expressed:

\[
T_{\text{execute}}^{(h_{i})} = y_{u} \frac{c_{j}H_{j}}{f_{(h_{i})}} \quad (9)
\]

where \( f_{(h_{i})} \) is the maximum number of task that MEC-Server can handle.

4. Three-tier computation model

If the MEC-server deployed on SBSs don’t have sufficient computation resources, it will upload remaining part of its to MEC-server deployed on MBS, which can effectively reduce time delay and energy consumption, otherwise, it will solve problem independently. So we divided the task into three parts \( c_{(h_{i})} \) and \( c_{(h_{i})} \), \( c_{(h_{i})} \) represents the part of subtask is processed in local terminal, \( c_{(h_{i})} \) represents the part of subtask is processed in MBS \( h_{i} \), and \( c_{(h_{i})} \) represents the remaining part of task is processed in SBS \( h_{i} \), we can conclude that \( c_{(h_{i})} + c_{(h_{i})} + c_{(h_{i})} = c_{h_{i}} \).

Therefore, the over delay in SBS \( h_{i} \) can be expressed as:

\[
T_{\text{delay}}^{(h_{i})} = y_{u} \frac{c_{j}H_{j}}{f_{(h_{i})}} + y_{u} \frac{c_{j}H_{j}}{f_{(h_{i})}} + y_{u} \frac{c_{j}H_{j}}{f_{(h_{i})}} \quad (10)
\]

where the first item indicates transmission time from local terminal to SBS \( h_{i} \), the second item indicates computation time of its subtasks, the third item indicates offloading time on wired line from SBS \( h_{i} \) to MBS \( h_{i} \), and the forth part show that computation time which processed by MBS \( h_{i} \) is a binary variable and has \( h_{(h_{i})} \in \{0,1\} \), which stands for the computation resource allocation about MEC server deployed on the SBS or not.

5. Edge energy consumption model

In our model, we exploit the following energy consumption model [19]:

\[
T_{\text{delay}}^{(h_{i})} = y_{u} \frac{c_{j}H_{j}}{f_{(h_{i})}} + y_{u} \frac{c_{j}H_{j}}{f_{(h_{i})}} + y_{u} \frac{c_{j}H_{j}}{f_{(h_{i})}} \quad (10)
\]
where \( P_t \) is the transmit power, \( e_t \) is the energy consumption per CPU cycle of MBS.

\[
E^{(h,a)} = x_{ij} \left( E^{(h,a)}_{\text{upload}} + E^{(h,a)}_{\text{execute}} \right) = x_{ij} \left( P_t T_{wp}^{(h,a)} + c_i e_{\text{MBS}} \right)
\]

\[
(11)
\]

where \( P_{\text{off}} \) is the offloading power between SBS and MBS, \( P_{\text{SBS}} \) is the SBS computation power consumption (in watt), and \( e_{\text{SBS}} \) is the energy consumption per CPU cycle of SBS. The first item indicates the transmission energy consumption from local to SBS, the second item indicates the computation energy consumption of subtask \( c_i \) in SBS, and the third item indicates the energy consumption of \( c_i \) from SBS to MBS, the forth item presents us with the computation energy consumption of remaining task \( c_j \) in MBS.

IV. PARALLEL MULTI-BLOCK ADMM BASED TASK ALLOCATION MECHANISM

In this section, based on the above models, we will describe the improved multi-block ADMM based optimal task allocation mechanism in detail, which jointly considers delay and energy in optimization utility, as shown in \( P1 \).

The (13-1) guarantees that the time of the task that is offloaded into MEC Server or the time of the task that processed in local terminals cannot exceed the deadline time. The constraint (13-3) guarantees that the sum of computation resource allocated to all offloading task cannot exceed the total amount of computation resource of MEC server deployed on SBS. The constraint (13-4) indicates each task is either executed locally, or by one of MEC server which deployed either on the MBS or on the SBS.

It has \([H \times |BS|]\) variables in our optimization problem, where \(|BS|\) is the number of BSs, \(|H|\) is the number of the tasks. Thus, the computational complexity significantly increases as the number of tasks and BSs. To solve the large-scale optimization problem, we propose a scalable and practical distributed method based on multi-block ADMM with using Logarithmic Smoothing [20] by modifying the inequality constrain as equality constrains in decision variables and smoothing function and using the cyclic block coordinate gradient projection (CBGP) method [21] to address the sub-optimal problem.

In order to make the problem separable, we develop parallel optimization based ADMM and introduce the local copies of the global variables and the local copies of the variables \( x_{ij} \), \( z_j \), \( y_{ij} \), which are defined as \( \hat{x}_{ij} \), \( \hat{z}_j \), \( \hat{y}_{ij} \).

\[
P1 : \min \alpha \left( \sum_{j=1}^{\#SBS} \sum_{i=1}^{\#Tasks} \hat{x}_{ij} + \sum_{j=1}^{\#Tasks} \hat{z}_{ij} + \sum_{j=1}^{\#Tasks} \hat{y}_{ij} \right) + \left( 1 - \alpha \right) \left( \sum_{j=1}^{\#SBS} \sum_{i=1}^{\#Tasks} \hat{x}_{ij} + \sum_{j=1}^{\#SBS} \sum_{i=1}^{\#Tasks} \hat{y}_{ij} + \sum_{j=1}^{\#Tasks} \hat{y}_{ij} \right)
\]

\[
\leq (13-1)
\]

\[
x_{ij} \in \{0,1\}, y_{ij} \in \{0,1\}, z_i \in \{0,1\}
\]

\[
\sum_{j=1}^{\#Tasks} x_{ij} \leq 1, 0 < b^{(h,a)} \leq 1
\]

\[
\sum_{j=1}^{\#Tasks} \hat{x}_{ij} + \hat{z}_j + \hat{y}_{ij} = 1
\]

\[
0 \leq \alpha \leq 1
\]

\[
c_{ij} + c_{ij} + c_{ij} = c_{ij}, 0 \leq c_{ij} \leq c_{ij}, 0 \leq c_{ij} \leq c_{ij}
\]

\[
i \in \{2,3,...,|BS|\}, j \in \{1,2,3,...,|H|\}
\]

\[
(13-7)
\]

Then we can obtain the equivalent global consensus version as \( P2 \).

In order to express more easily, we define the local variable set with some constrains. The term \( \frac{1}{b^{(h,a)}} \) in contains of the objective function \( P1 \) will lead to be non-convex. To deal with these sub-problems, we set \( \frac{1}{b^{(h,a)}} \) as \( f^{(h,a)} \) we define auxiliary variables as follows \( R_{ij} = x_{ij} f^{(h,a)} \).

We add some constrains to above equation in order to overcome the difficulty of the problem intractable. We use the reformulation linearization technology to replace above equation.
P2: \[ \min \alpha \left( \sum_{j=1}^{H} \sum_{j=1}^{T} z_{ij} T_{ij}^{(h)} + \sum_{j=1}^{T} \sum_{j=1}^{H} x_{ij} T^{(h)} + \sum_{j=1}^{T} y_{ij} T^{(h)} \right) + (1-\alpha) \left( \sum_{j=1}^{H} \sum_{j=1}^{T} z_{ij} E^{(h)} + \sum_{j=1}^{T} \sum_{j=1}^{H} x_{ij} E^{(h)} + \sum_{j=1}^{T} y_{ij} E^{(h)} \right) \]

\[ \begin{aligned} & \quad \sum_{j=1}^{H} \sum_{j=1}^{T} z_{ij} T_{ij}^{(h)} + \sum_{j=1}^{T} \sum_{j=1}^{H} x_{ij} T^{(h)} + \sum_{j=1}^{T} y_{ij} T^{(h)} \leq \epsilon - \alpha \sum_{j=1}^{H} \sum_{j=1}^{T} z_{ij} t_{ij} \leq \epsilon = \alpha \sum_{j=1}^{H} \sum_{j=1}^{T} z_{ij} T_{ij}^{(h)} + \sum_{j=1}^{T} \sum_{j=1}^{H} x_{ij} T^{(h)} + \sum_{j=1}^{T} y_{ij} T^{(h)} \leq \epsilon - \alpha \sum_{j=1}^{H} \sum_{j=1}^{T} z_{ij} t_{ij} \leq \epsilon \\ & \quad x_{ij} \in \{0, 1\}, \quad y_{ij} \in \{0, 1\}, \quad z_{ij} \in \{0, 1\} \\ & \quad \sum_{j=1}^{H} \sum_{j=1}^{T} h^{(h)} T_{ij}^{(h)} x_{ij} \leq 1 \\ & \quad 0 < h^{(h)} \leq 1 \\ & \quad \sum_{j=1}^{T} x_{ij} + z_{ij} + y_{ij} = 1 \\ & \quad x_{ij} = x_{ij}, \quad z_{ij} = z_{ij} \quad y_{ij} = y_{ij} \\ & \quad c_{ij} + c_{ij} + c_{ij} = c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij} \quad 0 \leq \alpha \leq 1, \quad i \in \{2, 3, \ldots, [BS]\}, \quad j \in \{1, 2, 3, \ldots, [T]\} \end{aligned} \] (14)

**Theorem 1:** The objective function in P2 can be changed as a linear function.

**Proof:** Please refer to the Appendix A.

Therefore, it is easy to divide the objective function into independent terms. As a result, P2 can be decomposed equivalently into the following sub-problems (P3, P4 and P5).

### 1. Augmented Lagrangian and ADMM Sequential iterations

The local variables can be expressed as P3, P4 and P5.

**P3:** \[ \min \alpha \sum_{j=1}^{H} \sum_{j=1}^{T} x_{ij} T^{(h)} + (1-\alpha) \sum_{j=1}^{T} \sum_{j=1}^{H} x_{ij} E^{(h)} + \sum_{j=1}^{T} \sum_{j=1}^{H} \left( x_{ij} - x_{ij} \right)^2 \]

\[ \begin{aligned} & \quad x_{ij} \in \{0, 1\} \\ & \quad \sum_{j=1}^{H} \sum_{j=1}^{T} x_{ij} T^{(h)} \leq 1 \\ & \quad x_{ij} \leq R_{ij} \leq \frac{1}{h_{ij}} \\ & \quad R_{ij} \geq h^{(h)} + x_{ij} - 1 \\ & \quad R_{ij} \geq h^{(h)} + x_{ij} - 1 \\ & \quad c_{ij} + c_{ij} + c_{ij} = c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad 0 \leq c_{ij} \leq c_{ij}, \quad \alpha \leq \alpha, \quad i \in \{2, 3, \ldots, [BS]\}, \quad j \in \{1, 2, 3, \ldots, [T]\} \end{aligned} \] (15)

**P5:** \[ \min \beta \sum_{j=1}^{H} \sum_{j=1}^{T} x_{ij} T^{(h)} + (1-\alpha) \sum_{j=1}^{T} \sum_{j=1}^{H} x_{ij} E^{(h)} + \sum_{j=1}^{T} \sum_{j=1}^{H} \left( x_{ij} - x_{ij} \right)^2 \]

\[ \begin{aligned} & \quad x_{ij} \in \{0, 1\} \end{aligned} \] (17)

The global variables can be expressed as P6.

Hence, the Lagrange multipliers can be expressed by (19).

\[ \begin{aligned} & \quad \alpha_{ij}^{k+1} = \alpha_{ij}^k + \rho \left( x_{ij} - x_{ij}^{k+1} \right) \\ & \quad \beta_{ij}^{k+1} = \beta_{ij}^k + \rho \left( z_{ij} - z_{ij}^{k+1} \right) \\ & \quad \gamma_{ij}^{k+1} = \gamma_{ij}^k + \rho \left( y_{ij} - y_{ij}^{k+1} \right) \\ & \quad i \in \{2, 3, \ldots, [BS]\}, \quad j \in \{1, 2, 3, \ldots, [T]\} \end{aligned} \] (19)

**B. Local Variables Update**

Obviously, the binary variable is non-convex, in order to transform non-convex set into convex set, we need to relax binary variable \( x_{ij} \) with a real value such that \( \sum_{j=1}^{H} \sum_{j=1}^{T} x_{ij} \cdot x_{ij} \leq 0 \) and \( 0 \leq x_{ij} \leq 1 \).

**Theorem 2:** P3 can be changed as P7.

**Proof:** Please refer to the Appendix B.
According to the observations in P7, the objective function is convex, therefore we have Theorem 3.

Theorem 3: The P7 can be solved by CBGP method.

Proof: Please refer to the Appendix C.

According to Theorem 3, we solve the problem by fixing four of five variables and deriving the remaining one, as shown in Algorithm 1.

Algorithm 1: Variables update in CBGP

Input:
- The amount of computing resource are required for the task: \( c_y \), Parameter setting: \( \alpha \in (0,1) \)
- The rounds of iterations: \( k \)

Output: \( x_y, c_{yj}, c_{ij}, R_y \)

1. Initialize \( x_y, c_{yj}, c_{ij}, y_y, \theta_y, \sigma_y, \theta_{ij}, \sigma_{ij}, c_y, c_{ij}, R_y \)
2. while \( k < \alpha \) do
   3. \( R_y^{(k)} = P_{\text{Dc}} \left( R_y^{(k)} - 2 \nabla Q(R_y^{(k)}) \right) \)
   4. \( c_{yj}^{(k+1)} = P_{\text{Dc}} \left( c_{yj}^{(k)} - 2 \nabla Q(c_{yj}^{(k)}) \right) \)
   5. \( c_{ij}^{(k+1)} = P_{\text{Dc}} \left( c_{ij}^{(k)} - 2 \nabla Q(c_{ij}^{(k)}) \right) \)
   6. \( c_{ij} = c_{ij} - c_{ij} \)
   7. \( x_y^{(k+1)} \) solving the problem with fixed \( c_{yj}, c_y, c_{ij}, R_y \)
   8. \( k++ \)
9. end while

The objective function of P4 is linear and strongly convex, which can be solved by comparing the two possible solutions. So the solution of P4 can be solved as following:

\[
\begin{align*}
  z_j = 0 & \quad \text{if} \quad d\theta^{(k)} + (1-\alpha)E^{(k)} + \frac{\rho}{2}(1-z_j) > 0 \\
  z_j = 1 & \quad \text{if} \quad d\theta^{(k)} + (1-\alpha)E^{(k)} + \frac{\rho}{2}(1-z_j) < 0
\end{align*}
\]

Similarly, the objective function of P5 is linear and strongly convex, which can be solved by comparing the two possible solutions. So the solution of P5 can be solved as following:

\[
\begin{align*}
  y_{ij} = 0 & \quad \text{if} \quad d\theta^{(k)} + (1-\alpha)E^{(k,ij)} + \frac{\rho}{2}(1-y_{ij}) > 0 \\
  y_{ij} = 1 & \quad \text{if} \quad d\theta^{(k)} + (1-\alpha)E^{(k,ij)} + \frac{\rho}{2}(1-y_{ij}) < 0
\end{align*}
\]

In the sub-problem P6, the objective function is nonlinear due to the nonlinear integer value \( x_y \), \( y_{ij} \) and \( z_j \), the similar method can be adopted in solving the integer constrain problems. Here, we use Logarithmic Smoothing to solve the nonlinear integer problem, the smoothing function aims to eliminate all the integer constrains. Secondly, in order to make the value close to zero or one, the objective function must be added to some penalty functions. So the objective function can be described as P8.
Next we use Newton’s method to solve the problems. Firstly, we add some auxiliary variables to transform the inequality constraints to the equality constraints, (24) and (25).

\[
z_j T_{\text{local}} + \sum_{i=1}^{m_i} x_j T_i^{(h, \cdot)} + y_j T_j^{(\cdot, \cdot)} + m_j = r_j^{-}\quad(24)
\]
\[
\begin{align*}
\nabla_G(x, y, z, m) + T_j^{(h, \cdot)} v_j + \zeta_j &= 0 \\
\nabla_G(x, y, z, m) + T_i^{(\cdot, \cdot)} v_i + \zeta_i &= 0 \\
\nabla_G(x, y, z, m) + T_{\text{local}}^{(h, \cdot)} v_j + \zeta_j &= 0 \\
\n\sum_{i=1}^{m_i} x_j T_i^{(h, \cdot)} + y_j T_j^{(\cdot, \cdot)} + m_j - r_j^{\text{max}} &= 0 \\
\end{align*}
\]

where \( m_j \geq 0 \).

Then, the P8 can be changed as P9. Hence, the Lagrangian equation is shown in (27). From (27), we can obtain the first order partial derivative about \( x, y, z, m, v_j, \zeta_j \).

**Theorem 4:** P9 can be solved by the combination of conjugate gradient, Newton and linear search techniques based algorithm with Logarithmic Smoothing.

**Proof:** Applying Newton’s method directly, we can obtain the following equation:

\[
\begin{bmatrix}
 g_1 & g_2 & g_3 & g_4 & T^{(h, \cdot)} \\
 g_5 & g_6 & g_7 & g_8 & T^{(\cdot, \cdot)} \\
 g_9 & g_{10} & g_{11} & g_{12} & T_{\text{local}}^{(h, \cdot)} \end{bmatrix} \begin{bmatrix}
 \Delta x \\
 \Delta y \\
 \Delta m \\
 \Delta v_j \\
 \Delta \zeta_j \\
 \end{bmatrix} = \begin{bmatrix}
 h_1 \\
 h_2 \\
 h_3 \\
 h_4 \\
 h_5 \\
 \end{bmatrix}
\]

where \( h_j = \begin{bmatrix}
 v_j G(x, y, z, m) + T^{(h, \cdot)} v_j + \zeta_j \\
 v_j G(x, y, z, m) + T^{(\cdot, \cdot)} v_i + \zeta_i \\
 v_j G(x, y, z, m) + T_{\text{local}}^{(h, \cdot)} v_j + \zeta_j \\
 \sum_{i=1}^{m_i} x_j T_i^{(h, \cdot)} + y_j T_j^{(\cdot, \cdot)} + m_j - r_j^{\text{max}} \\
 \end{bmatrix} \) (31)

We observe that the (40) can be reduced as follow:

\[
\begin{bmatrix}
 H \quad A_i \quad U \\
 A_i \quad 0 \\
 \end{bmatrix} \begin{bmatrix}
 \Delta x \\
 \Delta y \\
 \Delta m \\
 \Delta v_j \\
 \Delta \zeta_j \\
 \end{bmatrix} = \begin{bmatrix}
 M \\
 h_k \\
 \end{bmatrix} \quad(32)
\]

where

\[
U = [\Delta x, \Delta y, \Delta m]^T, \quad H = \text{Diag}(D_1, D_2, D_3)
\]

\[
A_i = [1,1,1]^T, \quad M = [W_i, W_j, W_k]
\]

\[
D_i = D_2 = D_3 = [g_9, g_{10}, g_{11}, g_{12}, T_{\text{local}}^{(h, \cdot)}],
\]

\[
W_i = [g_1, g_2, g_3, g_4, T^{(h, \cdot)}, T^{(\cdot, \cdot)}, T_{\text{local}}^{(h, \cdot)}],
\]

\[
W_2 = [g_9, g_{10}, g_{11}, g_{12}, T_{\text{local}}^{(h, \cdot)}],
\]

\[
W_3 = [g_9, g_{10}, g_{11}, g_{12}, T_{\text{local}}^{(h, \cdot)}],
\]

and

\[
\begin{bmatrix}
 g_1 = \nabla G(x, y, z, m) \\
 g_2 = \nabla G(x, y, z, m) \\
 g_3 = \nabla G(x, y, z, m) \\
 g_4 = \nabla G(x, y, z, m) \\
 g_5 = \nabla G(x, y, z, m) \\
 g_6 = \nabla G(x, y, z, m) \\
 g_7 = \nabla G(x, y, z, m) \\
 g_8 = \nabla G(x, y, z, m) \\
 g_9 = \nabla G(x, y, z, m) \\
 g_{10} = \nabla G(x, y, z, m) \\
 g_{11} = \nabla G(x, y, z, m) \\
 g_{12} = \nabla G(x, y, z, m) \\
 g_{13} = \nabla G(x, y, z, m) \\
 g_{14} = \nabla G(x, y, z, m) \\
 g_{15} = \nabla G(x, y, z, m) \\
 g_{16} = \nabla G(x, y, z, m)
\end{bmatrix}
\]

(30)
Theorem 5: From Theorem 4, it has $Z^THZX = Z^TM$.

Proof: Let $Z$ be null-space matrix of the $A$, then we can have $A_{Z} = 0$, we assume the $[x_0, y_0, z_0]$ is feasible point that $\forall i \in 1, i \sum x_i + y_i + z_i = 1$, so we conclude that $h_x = 0$ and $A_{Z}U = 0$, and we can get that $U = Z^TX$. Substituting this into the above equation we can get $Z^THZX = Z^TM$.

From Theorem 5, we can get the equation $Z^THZX = Z^TM$. Next, we use the conjugate gradient method and transform the equation into quadratic equation:

$$f(X) = \frac{1}{2}X^T Z^THZ X - Z^TMX$$

where $X^*$ can be solved as the solution points of $AX = B$.

Then we can get $X^*$ by the conjugate gradient method, and denote $x = x + \rho x, y = y + \rho y, z = z + \rho z$. We can use linear research technology to obtain the step $\rho$.

In this part, we evaluate the performance of the proposed task allocation mechanism. The parameters setting are shown in Table 1.

From Theorem 5, we can get the equation $Z^THZX = Z^TM$. Next, we use the conjugate gradient method and transform the equation into quadratic equation:

$$f(X) = \frac{1}{2}X^T Z^THZ X - Z^TMX$$

where $X^*$ can be solved as the solution points of $AX = B$.

Then we can get $X^*$ by the conjugate gradient method, and denote $x = x + \rho x, y = y + \rho y, z = z + \rho z$. We can use linear research technology to obtain the step $\rho$.
this circumstance. So we can conclude that the utility will increase with increasing value of parameter $\alpha$.

Table 1. Simulation parameters setting.

| Parameter                      | Value                      |
|--------------------------------|----------------------------|
| The transmit power of user     | 0.1w                       |
| Input data size                | 5 kbits-10kbits            |
| Delay threshold                | 15s-30s                    |
| Computational capability       | 10-100GHz CPU cycles/s     |
| The path loss exponent         | 4                          |
| The bandwidth                  | 20MHz                      |
| Computation workload           | 18000 CPU cycles/bit       |
| Computation efficiency coefficient | $10^{-20}$                 |
| The power of noise             | -172dBm/Hz                 |
| The transmit power of BS       | 40                         |
| BS coverage area               | 200m*200m                  |
| $\alpha$                       | 0.5                        |
| $M^p_T$                        | 100M                       |
| $\beta$                       | 0.1                        |
| $T$                            | 1000                       |

In Fig. 3, we verify the convergence of the improved ADMM algorithm in different step values. The number of tasks is 100, the number of BS is 6, the computational capacity of local terminal is 5GHz cycle/s, the computation capacity of SBS is 20GHz, the computation capacity of MBS is 100GHz, and the parameter $\alpha$ is 0.5. The rate of convergence in $\rho=1$ is slower than that in $\rho=1.2$. However, they are eventually converged in similar value. We can conclude that if the objective function is linear, we can get the optimal value.

Fig. 4 shows the changes in utility for different numbers of BSs among different numbers of tasks. The computational capacity is 5GHz cycle/s, and the step size $\rho$ is 1. The computational capacity of local terminal is 5GHz cycle/s, the computation capacity of SBS is 20GHz, and the computation capacity of MBS is 100GHz. The parameter $\alpha$ is 0.8. It is obvious that as the number of tasks increases, the utility will increase as well. With increasing the number of BSs (this means that the density of BS deployment increases), the distance between task and base station will decrease. From the above-mentioned analysis, we can know that the delay accounts for a large proportion of the total utility. Hence, if the distance decreases, it is obvious that utility will decrease as the density of base station increasing, as shown in Fig. 4.

In Fig. 5, we can draw several observations. The computational requirement of tasks is linearly related the size of tasks. When the size of task is smaller than 2.52kb, this task will be calculated in LT (local terminal). When the size of task is larger than 20Mb, our police will make task offload to MBS which can reduce energy consumption. When the size of tasks are 760.793kb, 867.93kb, 666.793kb, 467.93kb, 367.93kb, we use three-tier model for task management, as shown in Fig. 5, which can make full use of resource among LT, SBS, and MBS, reducing delay and energy.
In this paper, we present a parallel optimal task allocation mechanism for a large-scale MEC system, which includes local computing model, three-tier model and MBS computation model. It can make full use of resource among MT, SBS, and MBS, featuring large-scale distributed optimization and efficient task allocation mechanism, to reach intelligent and efficient resource management in MEC. In the proposed mechanism, we formulate the task allocation as nonlinear 0-1 integer programming problem, which can use Logarithmic Smoothing by modifying the inequality constrain as equality constrain in decision variables and smoothing function. In the meantime, we exploit regulation linear method to make the optimal function to be linear, guaranteeing its convergence and it can run in a parallel manner. For the solutions, we exploit the conjugate gradient and Newton methods, and CBGP method to solve the optimal problem and sub-optimal problem, respectively, in which the global and local variables can update simultaneously. The simulation results show that the proposed mechanism can guarantee the convergence and effectively reduce delay and energy consumption in a large-scale MEC system.

VI. CONCLUSION

...
An ADMM framework,” IEEE Transactions on Vehicular Technology 68(3): 2763-2776, 2019.

[10] Z. Zhou, J. Feng, Z. Chang, et al., “Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach,” IEEE Transactions on Vehicular Technology 68(5): 5087-5099, 2019.

[11] Y. Dai, D. Xu, S. Maharjan, et al., “Joint computation offloading and user association in multi-task mobile edge computing,” IEEE Transactions on Vehicular Technology 67(12): 12313-12325, 2018.

[12] P. Wang, Z. Zheng, B. Di, et al., “HetMEC: Latency-optimal task assignment and resource allocation for heterogeneous multi-layer mobile edge computing,” IEEE Transactions on Wireless Communications 18(10): 4942-4956, 2019.

[13] C. Li, J. Tang, H. Tang, et al., “Collaborative cache allocation and task scheduling for data-intensive applications in edge computing environment,” Future Generation Computer Systems 95: 249-264, 2019.

[14] H. Alameddine, S. Sharafeddine, S. Sebbah, et al., “Dynamic task offloading and scheduling for low-latency IoT services in multi-access edge computing,” IEEE Journal on Selected Areas in Communications 37(3): 668-682, 2019.

[15] C. Liu, M. Bennis, M. Debbah, et al., “Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing,” IEEE Transactions on Communications 67(6): 4132-4150, 2019.

[16] J. Meng, H. Tan, C. Xu, et al., “DEDAS: Online task dispatching and scheduling with bandwidth constraint in edge computing,” in Proc. of IEEE INFOCOM 2019: 2287-2295.

[17] Y. Kim, J. Kwak, S. Chong, “Dual-side optimization for cost-delay tradeoff in mobile edge computing,” IEEE Transactions on Vehicular Technology 67(2): 1765-1781, 2018.

[18] M. Gao, B. Addis, M. Bouet, et al., “Optimal orchestration of virtual network functions,” Computer Networks, (142):108-127, 2018.

[19] M. Liu, F. Yu, Y. Teng, et al., “Computation offloading and content caching in wireless blockchain networks with mobile edge computing,” IEEE Transactions on Vehicular Technology 67(11): 11008-11021, 2018.

[20] W. Murray, K. Ng, “An algorithm for nonlinear optimization problems with binary variables,” Computational Optimization and Applications 47(2): 257-288, 2010.

[21] Z Luo, P. Tseng, “On the convergence of the coordinate descent method for convex differentiable minimization,” Journal of Optimization Theory and Applications, 72(1):7-35, 1992.

Xiaoxiong Zhong (S’12-M’16) received his Ph.D degree in Computer Science and Technology from Harbin Institute of Technology, China, in 2015. He was a Postdoctoral Research Fellow with Tsinghua University, from 2016 to 2018. He is currently an assistant professor with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, China. His general research interests include network protocol design and analysis, data transmission and data analysis in internet of things and edge computing.

Xinghan Wang received his B.S. degree in Computer Science and Technology from Taiyuan University of Technology, China, in 2016. He is currently pursuing the M.S. degree in the School of Computer Science and Information Security at Guilin University of Electronic Technology, China. His research interest includes network protocol design and analysis, reinforcement learning and edge computing.

Yuanyuan Yang received the BEng and MS degrees in computer science and engineering from Tsinghua University and the MSE and PhD degrees in computer science from Johns Hopkins University. She is a SUNY Distinguished Professor of computer engineering and computer science and the Associate Dean for Academic Affairs in the College of Engineering and Applied Sciences at Stony Brook University, New York. Her research interests include wireless networks, data center networks and cloud computing. She has published over 380 papers in major journals and refereed conference proceedings and holds seven US patents in these areas. She is currently the Associate Editor-in-Chief for IEEE Transactions on Cloud Computing and an Associate Editor for ACM Computing Surveys. She has served as an Associate Editor-in-Chief and Associated Editor for IEEE Transactions on Computers and Associate Editor for IEEE Transactions on Parallel and Distributed Systems. She has also served as a general chair, program chair,
or vice chair for several major conferences and a program committee member for numerous conferences. She is an IEEE Fellow.

Yang Qin (S’98-M’01-SM’06) received her B.S (with first class honors) in Computer Science at Southwest Jiaotong University (China), in 1989, M.S in Computer Science at Huazhong University of Science & Technology, Wuhan, Hubei, in 1992, and Ph.D in Computer Science, Hong Kong University of Science & Technology, Kowloon, Hong Kong at November of 1999. From 1999 to 2000, she has visited the Washington State University as a Postdoc, USA. From 2000 to 2008, she is an assistant professor Nanyang Technological University, Singapore. Currently, she is an associate professor in the Department of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), China. Her research interests are in the areas of wireless networks, mobile computing, cross-layer design, QoS of routing and scheduling, high speed optical networks and so forth. She is a senior member of IEEE.

Xiaoke Ma received the PhD degree in computer science from Xidian University, in 2012. He is a professor with the School of Computer Science and Technology, Xidian University, China. His research interests include machine learning, optimization, data mining, and bioinformatics. He publishes more than 40 papers in the peer-reviewed international journals, such as the IEEE Transactions on Knowledge and Data Engineering, the IEEE Transactions on Cybernetics, Pattern Recognition, Bioinformatics, PLoS Computational Biology, Cell Stem Cell, Nuclear Acids Research, and the IEEE/ACM Transactions on Computational Biology and Bioinformatics.

Tingting Yang (M’13) received the Ph.D. degrees from Dalian Maritime University, China, in 2010. She is currently a professor in the School of Electrical Engineering and Intelligentization, Dongguan University of Technology, China. Since September 2012, she has been a visiting scholar at the Broadband Communications Research (BBCR) Lab at the Department of Electrical and Computer Engineering, University of Waterloo, Canada. Her research interests are in the areas of maritime wideband communication networks, DTN networks, and green wireless communication. She serves as the associate Editor-in-Chief of the IET Communications, as well as the advisory editor for SpringerPlus. She also serves as a TPC Member for IEEE ICC’14, ICC’15.
APPENDIX A

PROOF OF THEOREM 1

Proof: When $x_y=1$, from the first constraint we can get $R_y \geq 1$, and from the fourth constraint we can get $R_y \geq r^{(h,b)}$, since the variable $h^{(h,b)}$ meets that $0 \leq h^{(h,b)} \leq 1$, so we can get $r^{(h,b)} \geq 1$. The first and fourth constraints we can get $R_y \geq r^{(h,b)}$. The second constraints we can get $R_y \leq \frac{1}{h_{\min}}$. The third constraints we can get $R_y \leq r^{(h,b)}$. Since the variable meets that $r^{(h,b)} \leq \frac{1}{h_{\min}}$, the second and third constraints we can get $R_y \leq r^{(h,b)}$. So when $x_y=1$, $r^{(h,b)} \leq R_y \leq r^{(h,b)}$, as a result, we can get $R_y = r^{(h,b)}$. When $x_y=0$, from the first constraint we can get $R_y \geq 0$. From the fourth constraint we can get $R_y \geq r^{(h,b)} \frac{1}{h_{\min}}$. Since $r^{(h,b)} \leq \frac{1}{h_{\min}}$, from the first and fourth constraints we can get $R_y \geq 1$. Form the second constraint we can get $R_y \leq 1$, and from the third constraint we can get $R_y \leq r^{(h,b)}-1$. Since $r^{(h,b)} \geq 1$, from the second and third constraints we can get $R_y \leq 0$. So when $x_y=0$, $0 \leq R_y \leq 0$. As a result, we can get $R_y = 0$.

Hence, the Equation (34) can be replaced by Equation (35). And then the objective function in P2 can be changed as a linear function.

\[
\begin{aligned}
\xi &= \left\{ \begin{array}{ll}
x_y \in \{0,1\} \\
h^{(h,b)} \in \{0,1\} \\
x_y \in \{0,1\} \\
y_y \in \{0,1\} \\
x \in \{0,1\} \\
z \in \{0,1\}
\end{array} \right.
\end{aligned}
\]

\[
\begin{aligned}
&\sum_{i=1}^{H} \sum_{j=1}^{i} x_{ij} \leq 1 \\
&\sum_{r=1}^{\sum_{i=1}^{H} x_{ij}} \frac{1}{h_{\min}} \leq 1 \\
&\sum_{i=1}^{H} x_{ij} \leq \frac{1}{h_{\min}} \\
&\sum_{r=1}^{\sum_{i=1}^{H} x_{ij}} \frac{1}{h_{\min}} \leq 1
\end{aligned}
\]

which means that

\[
\begin{aligned}
&x_y^{k+1} - x_y^k < 0 \\
&x_y^{k+1} < x_y^k
\end{aligned}
\]

so, we can conclude that $x_y^{k+1} - x_y^k < x_y^k - (x_y^k)^2$.

\[
\begin{aligned}
x_y^{k+1} - x_y^k &< (x_y^k)^2 \\
x_y^{k+1} &< (x_y^k)^2 + x_y^k - x_y^k
\end{aligned}
\]

Hence, the Equation (34) can be replaced by

\[
\begin{aligned}
&x_y^{k+1} - x_y^k < (x_y^k)^2, x_y^{k+1} - x_y^k
\end{aligned}
\]

APPENDIX B

PROOF OF THEOREM 2

Proof: $x_y^{k+1}$ is better than $x_y^k$, so we can conclude that $x_y^{k+1} - x_y^k < x_y^k - (x_y^k)^2$.

\[
\begin{aligned}
&x_y^{k+1} - x_y^k < x_y^k - (x_y^k)^2 \leq \left( x_y^k - \nabla (x_y^k)^2, x_y^{k+1} - x_y^k \right) \\
&\leq x_y^k - (x_y^k)^2 - \nabla (x_y^k)^2, x_y^{k+1} - x_y^k \\
= x_y^k - (x_y^k)^2
\end{aligned}
\]

which means that

\[
\begin{aligned}
x_y^{k+1} - x_y^k < (x_y^k)^2, x_y^{k+1} - x_y^k
\end{aligned}
\]

Hence, $\sum_{i=1}^{H} \sum_{j=1}^{i} x_{ij} - x_y^2$ can be replaced by

\[
\begin{aligned}
x_y^{k+1} - x_y^k < (x_y^k)^2, x_y^{k+1} - x_y^k
\end{aligned}
\]

And then if we add the equation into P3, then we can obtain P7.
\[ R_{ij}^{(k+1)} = P_{ij}^k \left( R_{ij}^{(k)} - \varepsilon_j R_{ij} \nabla Q(R_{ij}^{(k)}) \right) \]  

(36)

where \( \alpha_{ij} \) is the bounded domain constrained of variables.

We update the variables \( c_{ij}, c_j, c_y \) sequentially according to CBGP method. In updating, the objective function is convex and the constraints are linear, so we use subgradient to update the variables.

\[
\begin{align*}
\nabla_x L(x, y, \theta, \phi, \psi, \sigma) &= 0, \\
\nabla_y L(x, y, \theta, \phi, \psi, \sigma) &= 0.
\end{align*}
\]

(41)

and for each \( t \) we have

\[
\begin{align*}
\nabla_u G(u_t, v_t, z(u_t), m(u_t)) &= 0, \\
\nabla_u G(u_t, v_t, z(u_t), m(u_t)) &= 0.
\end{align*}
\]

(42)

From above equation we can conclude that:

\[
\begin{bmatrix}
T_{ij}^{(k)} \\
T^{(h,k)} \\
T^{(h,k)} \\
1
\end{bmatrix}
\begin{bmatrix}
x(u_t) - x^* \\
y(u_t) - y^* \\
z(u_t) - z^* \\
m(u_t) - m^*
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]

(43)
\[
\begin{pmatrix}
H_1 \\
H_2 \\
H_3 \\
H_4
\end{pmatrix}
+ \begin{pmatrix}
T^{(h,b)}_1 & 1 & 0 & 0 \\
T^{(h,b)}_2 & 1 & 0 & 0 \\
T^{(\nu_l)} & 1 & 0 & 0 \\
T_{\text{local}} & 1 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
\nu_j(u_j) - \nu_j^* \\
\xi_j(u_j) - \xi_j^*
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
0 \\
0
\end{pmatrix}
\] (44)

where
\[
H_1 = \nabla_x G(x(u), y(u), z(u), m(u)) - \nabla_x G(x^*, y^*, z^*, m^*)
\]
\[
H_2 = \nabla_y G(x(u), y(u), z(u), m(u)) - \nabla_y G(x^*, y^*, z^*, m^*)
\]
\[
H_3 = \nabla_z G(x(u), y(u), z(u), m(u)) - \nabla_z G(x^*, y^*, z^*, m^*)
\]
\[
H_4 = \nabla_m G(x(u), y(u), z(u), m(u)) - \nabla_m G(x^*, y^*, z^*, m^*)
\] (45)

Multiplying the both sides
\[
\begin{pmatrix}
x(u) - x^* \\
y(u) - y^* \\
z(u) - z^* \\
m(u) - m^*
\end{pmatrix}
\] , we can conclude that:
\[
\begin{pmatrix}
x(u) - x^* \\
y(u) - y^* \\
z(u) - z^* \\
m(u) - m^*
\end{pmatrix}^T
\begin{pmatrix}
H_1 \\
H_2 \\
H_3 \\
H_4
\end{pmatrix} = 0
\] (46)

Since \(H_1, H_2, H_3, H_4\) are linearly independent. So we can conclude that
\[
\begin{align*}
x(u) - x^* &= 0 \\
y(u) - y^* &= 0 \\
z(u) - z^* &= 0 \\
m(u) - m^* &= 0
\end{align*}
\] (47)