Journal of Computational Mathematics
Vol.35, No.4, 2017, 421–436.

http://www.global-sci.org/jcm
doi:10.4208/jcm.1608-m2016-0561

COLLABORATIVE RESOURCE ALLOCATION OVER A HYBRID CLOUD CENTER AND EDGE SERVER NETWORK*

Houfeng Huang and Qing Ling
Department of Automation, University of Science and Technology of China, Hefei, China
Email: hhoufeng@mail.ustc.edu.cn qingling@mail.ustc.edu.cn
Wei Shi
Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, USA
Email: wilburs@illinois.edu
Jinlin Wang
National Network New Media Engineering Research Center, Institute of Acoustics
Chinese Academy of Sciences, Beijing, China
Email: wangjl@dsp.ac.cn

Abstract

This paper considers the collaborative resource allocation problem over a hybrid cloud center and edge server network, an emerging infrastructure for efficient Internet services. The cloud center acts as a pool of inexhaustible computation and storage powers. The edge servers often have limited computation and storage powers but are able to provide quick responses to service requests from end users. Upon receiving service requests, edge servers assign them to themselves, their neighboring edge servers, as well as the cloud center, aiming at minimizing the overall network cost.

This paper first establishes an optimization model for this problem. Second, in light of the separable structure of the optimization model, we utilize the alternating direction method of multipliers (ADMM) to develop a fully collaborative resource allocation algorithm. The edge servers and the cloud center autonomously collaborate to compute their local optimization variables and prices of network resources, and reach an optimal solution. Numerical experiments demonstrate the effectiveness of the hybrid network infrastructure as well as the proposed algorithm.

Mathematics subject classification: 90C25, 90C30.
Key words: Network resource allocation, Distributed network optimization, Cloud center, edge server.

1. Introduction

The fast development of communication and networking technologies in the past decades has brought unprecedented prosperity of Internet services, which has significantly shaped our daily lives, created new business models, and accelerated the process of globalization. The core of Internet services is to allocate network *resources* to meet the service requests so as to maximize the network-wide utility. The resources include network bandwidth, computation power, storage power, etc. Upon the requests such as searching for keywords, asking for recommending restaurants and watching online movies, the service providers allocate the network resources, aiming at minimizing the overall cost of network resources and maximize the quality of service

* Received February 29, 2016 / Revised version received June 25, 2016 / Accepted August 20, 2016 / Published online June 1, 2017 /

for the Internet users. These two objectives can be unified to a framework of maximizing the network-wide utility.

This paper focuses on the collaborative resource allocation problem over a hybrid cloud center and edge server network, an emerging infrastructure for efficient Internet services. The network has a cloud center that acts as a pool of computation and storage powers, as well as multiple edge servers that directly interact with end users; see Fig. 1 for an illustration. The cloud center, though may be a collection of geographically distributed components, can be abstracted as a single node in the network. It has inexhaustible computation and storage powers while the communication costs between the cloud center and the edge servers are considerable. The edge servers often have limited computation and storage powers. However, they are able to provide quick responses to service requests. If the quality of service given by an edge server to its end users is unsatisfactory, it can forward a fraction of the received service requests to the cloud center though this brings extra communication cost and latency. The edge server can also forward the service requests to neighboring edge servers who have available computation and storage powers. By "neighbors" we mean that two edge servers between whom the communication cost and the latency are relatively small.

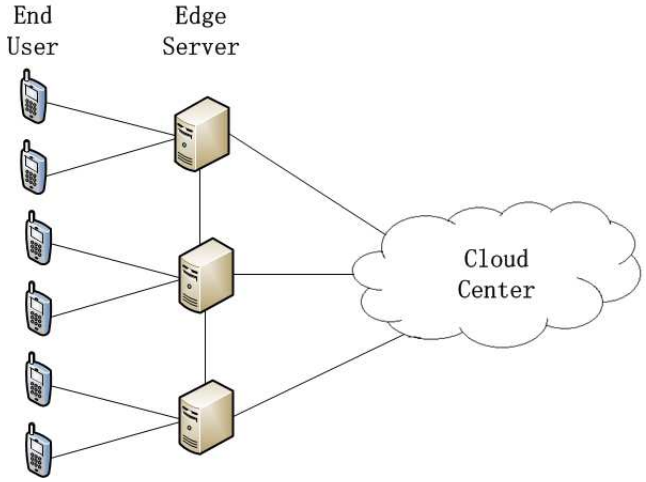

Fig. 1. Infrastructure of a hybrid cloud center and edge server network.

This novel network infrastructure takes advantages of both cloud computing that fits for computation- and storage-intensive applications [1] and edge computing (also known as fog computing) that provides fast response [2, 3], and hence brings elastic network services to the end users. We will give several illustrative examples about its applications in Section 2. However, this hybrid infrastructure leads to challenges in modeling and solving the collaborative network resource allocation problem. This paper aims at addressing these two issues. To be specific, our contributions are two-fold.

(i) We establish a collaborative resource allocation model for the hybrid cloud center and edge server network. We define the costs of the network resources, such as computation and storage on the cloud center and the edge servers as well as communication over the links, and formulate a network cost minimization (or equivalently, utility maximization) problem.

(ii) We develop a fully autonomous resource allocation algorithm so that the cloud center

and the edge servers cooperate to reach an optimal assignment of the network resources. We utilize the separable problem structure and adopt the alternating direction method of multipliers (ADMM) to decompose the decision process to the cloud center and the edge servers.

Notice that due to the hybrid network infrastructure, the proposed model and algorithm include many classical network resource allocation models and algorithms as special cases or variants [4–8]. They range from flow optimization over a computer network [9] to consensus optimization over a decentralized network [10–13], coordination of a master-slave network [14], resource allocation within a data center [15, 16], and demand response in smart grids [17, 18], to name a few. We will demonstrate their connections in Section 2.

This paper is organized as follows. Section 2 establishes a collaborative resource allocation model for the hybrid cloud center and edge server network, gives several illustrative examples about its applications, and demonstrates the connection between the proposed and existing models. Section 3 proposes a fully autonomous resource allocation algorithm where the cloud center and the edge servers cooperate to reach an optimal assignment of the network resources. The algorithm development is based on ADMM, which utilizes the separable problem structure and decomposes the decision process to the cloud center and the edge servers. Section 4 provides numerical experiments to demonstrate the advantages of the hybrid cloud center and edge server network infrastructure as well as the effectiveness of the proposed collaborative resource allocation algorithm. Section 5 concludes the paper and discusses future research directions.

2. Problem Statement

This section gives a collaborative resource allocation model for the hybrid cloud center and edge server network, followed by several illustrative examples about its applications. The model rooted in the hybrid network infrastructure covers many classical network resource allocation models as special cases or variants as we will explain.

2.1. Network Resource Allocation Model

As illustrated in Fig. 1, the hybrid network consists of several entities: the cloud center, the edge servers, and the communication links between them. We model the cloud center as a node whose label is 0, though in practice it is often composed of geographically distributed components. The edge servers are nodes labelled from 1 to L. Every edge server i is able to communicate with the cloud center, and the communication channel is denoted by an undirected link $(0, i)$. Notice that the communication costs over these links are generally very high. Two edge servers i and j having an undirected communication link (i, j) in between means that the corresponding communication cost over the link is small enough. These two edge servers are called as neighbors. The set of neighbors of edge server i is denoted as $\mathcal{N}_i$. We do not allow non-neighboring edge servers to communicate with each other since this brings extra management cost.

The network resources include network bandwidth that is attached to the communication links, as well as computation and storage powers that are attached to the nodes (both the cloud center and the edge servers). These resources are allocated to handle service requests from end users. We assume that there are P classes of service requests whose amounts can be quantified.

For simplicity we can assume that $P = 1$, namely, there is only one class of service request. We give detailed explanations of the settings as follows.

(i) Edge server i receives service request from several end users, whose amount is denoted by a nonnegative vector $s_i \in \mathcal{R}_+^P$. The p-th element $s_i(p)$ stands for the amount of the p-th class of service request. Edge server i can assign this amount to itself and other nodes (its neighboring edge servers and the cloud center). Denote s_{ii}, s_{ij}, and s_{i0}, all $\in \mathcal{R}_+^P$, as the amount assigned to itself, its neighboring edge servers $j \in \mathcal{N}_i$, and the cloud center, respectively. To satisfy the service request we must have $\sum_{j\in\mathcal{N}_i\cup i\cup 0} s_{ij} = s_i$. Notice that if two edge servers i and j are not neighbors, then by assumption they cannot assign requests to each other. Therefore, $s_{ij} = 0$ if $j \notin \mathcal{N}_i \cup i \cup 0$.

(ii) The cloud center and the edge servers need to spend computation and storage resources to handle the amount of service requests assigned by themselves and/or other nodes. If service request s_{ij} is assigned to node j by node i, node j spends a certain amount of resource to process it, which leads to the computation/storage cost determined by s_{ij}.

(iii) When node j receives service request s_{ij} assigned by node i, the two nodes may need to transmit data (for example, data to process or to store). When node j finishes processing, it returns the final result to node i because the latter is the user interface of the corresponding service request. Both transmissions incur the communication cost determined by s_{ij} over link (i, j).

In the network resource allocation problem, the goal is to minimize the summed costs of the resources given that the service requests are satisfied. If edge server i spends a total amount of resource $\sum_{j\in\mathcal{N}_i\cup i} s_{ji}$ to process its assigned service requests, we define the cost occurred at node i as $f_i(\sum_{j\in\mathcal{N}_i\cup i} s_{ji})$. For every neighboring edge server i, the cost occurred over link (i, j) for edge server j processing the service request s_{ij} is defined as $g_{ij}(s_{ij})$. Similarly if the cloud center, which is labelled as node 0, spends a total amount of resource $\sum_{j=1}^{L} s_{j0}$ to process its assigned service requests, we define the cost occurred at the cloud server as $f_0(\sum_{j=1}^{L} s_{j0})$. For every edge server i, the cost occurred over link $(0, i)$ for the cloud center processing the service request s_{i0} is defined as $g_{i0}(s_{i0})$. Notice that in practice the computation and storage costs at the cloud center are much less than those at an edge server, while the communication cost between the cloud center and an edge server is much larger than that between two neighboring edge servers. This prior knowledge will be reflected in the design of the cost functions f_0, $f_i, i = 1, \cdots, L$, $g_{i0}, i = 1, \cdots, L$, and $g_{ij}, j \in \mathcal{N}_i, i = 1, \cdots, L$.

Based on the discussions above, we formulate the collaborative resource allocation problem over a hybrid cloud center and edge server network as follows.

$$\begin{aligned} \min_{\{s_{ij}\}} \quad & f_0\left(\sum_{j=1}^{L} s_{j0}\right) + \sum_{i=1}^{L} f_i\left(\sum_{j\in\mathcal{N}_i\cup i} s_{ji}\right) + \sum_{i=1}^{L}\sum_{j\in\mathcal{N}_i\cup 0} g_{ij}(s_{ij}), \qquad (2.1)\\ s.t. \quad & s_i = \sum_{j\in\mathcal{N}_i\cup i\cup 0} s_{ij},\ \forall i = 1, \cdots, L,\\ & s_{ij} \geq 0,\ \forall j \in \mathcal{N}_i \cup i \cup 0,\ \forall i = 1, \cdots, L. \end{aligned}$$

The objective function (2.1) is the summed cost of the nodes and the links. The first line of the constraints means satisfaction of the user requests, and the second line of the constraints

requires that all the assigned amount of requests are nonnegative. The goal of collaborative resource allocation is to find the optimal values of s_{ij}, $j \in \mathcal{N}_i \cup i \cup 0$, $i = 1, 2, \cdots, L$ by solving (2.1). An illustration of service requests, task assignments, and cost functions is given by Fig. 2, where the network contains one cloud center and three edge servers.

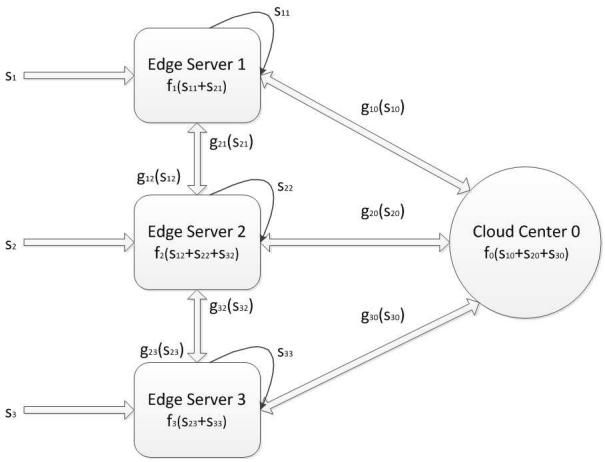

Fig. 2. Illustration of service requests, task assignments, and cost functions.

2.2. Examples of Applications

Below we give several illustrative examples for the network resource allocation model (2.1).

Distributed surveillance video processing. Suppose that L cameras are deployed in a large monitoring area. These cameras take videos from which we can identify terrorists, discover hazardous events, etc. The video processing task can be roughly done on-site by the cameras, but with a low accuracy since they only have limited computation powers. Thus, the task can be decomposed so that other nearby cameras are able to contribute their idle computation powers or we allow a cloud center to do fine processing in an offline manner. In this example, the cameras work as edge servers and the service requests are to process the taken videos. The edge server and the cloud center have computation powers as their resources; their costs are determined by the utilized computation powers. The costs of the resources of the communication links are represented by latencies due to offline video processing. The combination of these two costs constitutes the network-wide objective.

Distributed online stream media service. We consider the problem of providing online steam media service to a city. The city is divided into L districts, each having an edge server to receive service requests from end users inside the corresponding district. An edge server stores some popular stream media contents (movies, songs, etc). When end users request these contents, the edge server is able to provide fast and high-quality service. However, the storage power of the edge server is limited. If by chance end users ask for some other contents, the edge server has to forward these service requests to neighboring edge servers or a cloud center. The edge servers and the cloud center have storage powers as their resources, which determine the costs of the nodes. The costs of the resources of the communication links are quantified by the qualities of services (for example, latencies).

2.3. Connections with Existing Models

The network resource allocation model (2.1) for the hybrid cloud center and edge server network include many existing models as special cases or variants. The hybrid network infrastructure enables collaboration among the decentralized edge servers, as well as collaboration between the edge servers and the cloud center. This generalizes most existing works that allow either collaboration among decentralized nodes, or collaboration between distributed nodes and a fusion center.

Coordination of a master-slave network. A master-slave network is composed of L distributed slave nodes and one centralized master node [14]. Slave node i has its local cost v_i determined by its local argument y_i. The master node also has its cost v_0 that is determined by all the local arguments $y_1, y_2, \cdots, y_L$. The goal is to minimize the summed costs of all the slave nodes and the master node. The problem can be formulated as

$$\min_{\{y_i\}} \quad \sum_{i=1}^{L} v_i(y_i) + v_0(y_1, y_2, \cdots, y_L). \tag{2.2}$$

Compared with (2.1), (2.2) does not allow collaboration among the slave nodes and is hence a simplified model.

Resource allocation within a data center. Suppose that a data center having L facilities serves N users [15, 16]. The amount of service given by facility i to user j is denoted by a nonnegative variable s_{ij}. Every facility i has a local cost $f_i(\sum_{j=1}^{N} s_{ij})$, while every user j also has a local cost $g_j(s_{1j}, s_{2j}, \cdots, s_{Lj})$. The problem formulation is

$$\begin{aligned} \min_{\{s_{ij}\}} \quad & \sum_{i=1}^{L} f_i(\sum_{j=1}^{N} s_{ij}) + \sum_{j=1}^{N} g_j(s_{1j}, s_{2j}, \cdots, s_{Lj}), \\ s.t. \quad & s_{ij} \geq 0, \ \forall j = 1, \cdots, N, \ \forall i = 1, \cdots, L. \end{aligned} \tag{2.3}$$

If we remove the cloud center and let all the edge servers be neighbors of each other, (2.1) degenerates to the case that L facilities (edge servers) serve L users (edge servers), namely, the model of (2.3) with $L = N$.

3. Algorithm Development

Solving the collaborative resource allocation problem (2.1) is challenging because the optimization variables s_{ij} are entangled. First, the requests s_{j0} assigned by all the edge servers j to the cloud center are coupled in the term $f_0(\sum_{j=1}^{L} s_{j0})$. Second, for every edge server i, the requests s_{ji} assigned to it from all $j \in \mathcal{N}_i \cup i$ are coupled in the term $f_i(\sum_{j\in\mathcal{N}_i\cup i} s_{ji})$. Last, for every edge server i, the requests s_{ij} assigned by it to all $j \in \mathcal{N}_i \cup i \cup 0$ are coupled in the constraint $s_i = \sum_{j\in\mathcal{N}_i\cup i\cup 0} s_{ij}$.

Existing works on the classical network resource allocation problem rely on the technique of dual decomposition to handle the entangled variables [4–8]. In the dual domain, the entangled variables are naturally decoupled, which enables decentralized and autonomous collaboration of the nodes. However, the dual decomposition algorithm often suffers from slow convergence speed and sensitivity to algorithm parameters.

3.1. An ADMM Approach

In this paper, we propose to apply the alternating direction method of multipliers (ADMM), a powerful primal-dual operator splitting algorithm, to solve (2.1). For a minimization problem with two blocks of optimization variables, which are coupled with linear constraints, at every iteration ADMM minimizes the augmented Lagrangian with respect to the two blocks of optimization variables in an alternating direction manner, followed by a dual ascent step to update the dual variable [19,20]. ADMM has been widely applied in various applications due to its fast convergence speed and remarkable numerical stability [21, 22], particularly in the decentralized consensus optimization problem [11–13] and the data center resource allocation problem [15].

To solve (2.1) by ADMM, we begin with reformulating (2.1). Recall that for all the edge servers $i = 1, \cdots, L$, optimization variables s_{ij}, $j \in \mathcal{N}_i \cup i \cup 0$ denote the amount of requests assigned to themselves, their neighbors and the cloud center. Now we introduce auxiliary variables $r_{ji} \in \mathcal{R}^P$, such that $r_{ji} = s_{ij}$, $i = 1, \cdots, L$, $j \in \mathcal{N}_i \cup i \cup 0$, to denote the amount of resources that must be spent to handle the requests. Meanwhile, denote $r_0 = \sum_{j=1}^{L} r_{0j} \in \mathcal{R}^P$ as the amount of resource contributed by the cloud center and $r_i = \sum_{j \in \mathcal{N}_i \cup i} r_{ij} \in \mathcal{R}^P$ as that by every edge server i. This way, (2.1) is equivalent to

$$\begin{aligned} \min_{r_0, \{r_i\}, \{s_{ij}\}, \{r_{ij}\}} \quad & f_0(r_0) + \sum_{i=1}^{L} f_i(r_i) + \sum_{i=1}^{L} \sum_{j \in \mathcal{N}_i \cup 0} g_{ij}(s_{ij}), \\ s.t. \quad & r_{ji} = s_{ij}, \ \forall j \in \mathcal{N}_i \cup i \cup 0, \ \forall i = 1, \cdots, L, \\ & r_0 = \sum_{j=1}^{L} r_{0j}, \ r_i = \sum_{j \in \mathcal{N}_i \cup i} r_{ij}, \ \forall i = 1, 2, \cdots, L, \\ & s_i = \sum_{j \in \mathcal{N}_i \cup i \cup 0} s_{ij}, \ \forall i = 1, \cdots, L, \\ & s_{ij} \geq 0, \ \forall j \in \mathcal{N}_i \cup i \cup 0, \ \forall i = 1, \cdots, L. \end{aligned} \tag{3.1}$$

For (3.1), write its augmented Lagrangian as

$$\begin{aligned} & L_\rho(r_0, \{r_i\}, \{s_{ij}\}, \{r_{ij}\}, a_0, \{a_i\}, \{b_i\}, \{c_{ij}\}) \\ =& f_0(r_0) + \sum_{i=1}^{L} f_i(r_i) + \sum_{i=1}^{L} \sum_{j \in \mathcal{N}_i \cup 0} g_{ij}(s_{ij}) + \sum_{i=1}^{L} \sum_{j \in \mathcal{N}_i \cup i \cup 0} \langle c_{ij}, r_{ji} - s_{ij} \rangle \\ & + \frac{\rho}{2} \sum_{i=1}^{L} \sum_{j \in \mathcal{N}_i \cup i \cup 0} (r_{ji} - s_{ij})^2 + \langle a_0, r_0 - \sum_{j=1}^{L} r_{0j} \rangle + \frac{\rho}{2} \left(r_0 - \sum_{j=1}^{L} r_{0j} \right)^2 + \sum_{i=1}^{L} \langle a_i, r_i - \sum_{j \in \mathcal{N}_i \cup i} r_{ij} \rangle \\ & + \frac{\rho}{2} \sum_{i=1}^{L} \left(r_i - \sum_{j \in \mathcal{N}_i \cup i} r_{ij} \right)^2 + \sum_{i=1}^{L} \langle b_i, s_i - \sum_{j \in \mathcal{N}_i \cup i \cup 0} s_{ij} \rangle + \frac{\rho}{2} \sum_{i=1}^{L} \left(s_i - \sum_{j \in \mathcal{N}_i \cup i \cup 0} s_{ij} \right)^2, \end{aligned} \tag{3.2}$$

subject to $s_{ij} \geq 0, \forall j \in \mathcal{N}_i \cup i \cup 0, \forall i = 1, \cdots, L$. Here c_{ij}, a_0, a_i and b_i, all $\in \mathcal{R}^P$, are the Lagrange multipliers attached to the constraints

$$r_{ji} = s_{ij}, \quad r_0 - \sum_{j=1}^{L} r_{0j}, \ r_i - \sum_{j \in \mathcal{N}_i \cup i} r_{ij}, \quad s_i = \sum_{j \in \mathcal{N}_i \cup i \cup 0} s_{ij},$$

respectively. The positive constant ρ is the coefficient of the augmented quadratic terms. We can choose different coefficients for the augmented quadratic terms, but simply setting them as the

same is already able to provide us satisfactory numerical performance, as we will demonstrate in Section 4.

We divide the primal variables into two groups: r_0, $\{r_i\}$, $\{s_{ij}\}$ in the first group and $\{r_{ij}\}$ in the second group. At time $k+1$, ADMM first fixes the primal variables $\{r_{ij}\}$ and the dual variables $a_0, \{a_i\}, \{b_i\}, \{c_{ij}\}$ to minimize the augmented Lagrangian (3.2) with respect to $r_0, \{r_i\}, \{s_{ij}\}$. Notice that r_0, $\{r_i\}$, $\{s_{ij}\}$ are not coupled, meaning that they can be optimized separately. For the cloud center, the update of r_0 is

$$r_0^{k+1} = \arg\min_{r_0} \ f_0(r_0) + \frac{\rho}{2}\left(r_0 + \frac{a_0^k}{\rho} - \sum_{j=1}^{L} r_{0j}^k\right)^2. \tag{3.3}$$

For edge server i, the update of r_i is

$$r_i^{k+1} = \arg\min_{r_i} \ f_i(r_i) + \frac{\rho}{2}\left(r_i + \frac{a_i^k}{\rho} - \sum_{j\in\mathcal{N}_i\cup i} r_{ij}^k\right)^2. \tag{3.4}$$

For every edge server i, consider its neighboring edge server j if $j \neq 0$ or the cloud center if $j = 0$, the update of s_{ij} is

$$\begin{aligned}\{s_{ij}^{k+1}\} = \arg\min_{\{s_{ij}\geq 0\}} &\sum_{j\in\mathcal{N}_i\cup 0} g_{ij}(s_{ij}) + \frac{\rho}{2}\sum_{j\in\mathcal{N}_i\cup i\cup 0}\left(r_{ji}^k + \frac{c_{ij}^k}{\rho} - s_{ij}\right)^2 \\ &+ \frac{\rho}{2}\left(s_i + \frac{b_i^k}{\rho} - \sum_{j\in\mathcal{N}_i\cup i\cup 0} s_{ij}\right)^2.\end{aligned} \tag{3.5}$$

In (3.5), the number of the optimization variables is $|\mathcal{N}_i|+2$ where $|\mathcal{N}_i|$ denotes the cardinality of the set $\mathcal{N}_i$. This fact implies that the optimization problem is not difficult if for every edge server i the number of its neighbors $|\mathcal{N}_i|$ is limited, even though the whole number of the edge servers L is very large.

Second, ADMM fixes the primal variables r_0, $\{r_i\}$, $\{s_{ij}\}$ and the dual variables a_0, $\{a_i\}$, $\{b_i\}$, $\{c_{ij}\}$ to minimize the augmented Lagrangian (3.2) with respect to $\{r_{ij}\}$. For the cloud center (namely, $i = 0$), the update of $\{r_{0j}\}$ is

$$\begin{aligned}\{r_{0j}^{k+1}\} &= \arg\min_{\{r_{0j}\}} \ \frac{\rho}{2}\sum_{i=1}^{L}\left(r_{0i} + \frac{c_{i0}^k}{\rho} - s_{i0}^{k+1}\right)^2 + \frac{\rho}{2}\left(r_0^{k+1} + \frac{a_0^k}{\rho} - \sum_{j=1}^{L} r_{0j}\right)^2 \\ &= \arg\min_{\{r_{0j}\}} \ \frac{\rho}{2}\sum_{j=1}^{L}\left(r_{0j} + \frac{c_{j0}^k}{\rho} - s_{j0}^{k+1}\right)^2 + \frac{\rho}{2}\left(r_0^{k+1} + \frac{a_0^k}{\rho} - \sum_{j=1}^{L} r_{0j}\right)^2.\end{aligned} \tag{3.6}$$

For all edge servers i and for all $j \in \mathcal{N}_i \cup i$, the update of $\{r_{ij}\}$ is

$$\begin{aligned}\{r_{ij}^{k+1}\} &= \arg\min_{\{r_{ij}\}} \ \frac{\rho}{2}\sum_{i=1}^{L}\sum_{j\in\mathcal{N}_i\cup i}\left(r_{ji} + \frac{c_{ij}^k}{\rho} - s_{ij}^{k+1}\right)^2 + \frac{\rho}{2}\sum_{i=1}^{L}\left(r_i^{k+1} + \frac{a_i^k}{\rho} - \sum_{j\in\mathcal{N}_i\cup i} r_{ij}\right)^2 \\ &= \arg\min_{\{r_{ij}\}} \ d\frac{\rho}{2}\sum_{i=1}^{L}\sum_{j\in\mathcal{N}_i\cup i}\left(r_{ij} + \frac{c_{ji}^k}{\rho} - s_{ji}^{k+1}\right)^2 + \frac{\rho}{2}\sum_{i=1}^{L}\left(r_i^{k+1} + \frac{a_i^k}{\rho} - \sum_{j\in\mathcal{N}_i\cup i} r_{ij}\right)^2,\end{aligned}$$

where the second equality is due to the fact that the network is undirected. Therefore, we can see that the computation of $\{r_{ij}\}$ can be decomposed to every edge server i. For every edge server i and for every $j \in \mathcal{N}_i \cup i$, the update of $\{r_{ij}\}$ is

$$\{r_{ij}^{k+1}\} = \arg\min_{\{r_{ij}\}} \frac{\rho}{2} \sum_{j \in \mathcal{N}_i \cup i} \left(r_{ij} + \frac{c_{ji}^k}{\rho} - s_{ji}^{k+1} \right)^2 + \frac{\rho}{2} \left(r_i^{k+1} + \frac{a_i^k}{\rho} - \sum_{j \in \mathcal{N}_i \cup i} r_{ij} \right)^2. \tag{3.7}$$

Observe that both (3.6) and (3.7) are simple quadratic programs and have explicit solutions.

Last, ADMM updates the dual variables a_0, $\{a_i\}$, $\{b_i\}$, $\{c_{ij}\}$ using the latest values of the primal variables. The cloud center updates a_0 by

$$a_0^{k+1} = a_0^k + \rho \left(r_0^{k+1} - \sum_{j=1}^{L} r_{0j}^{k+1} \right). \tag{3.8}$$

Edge server i updates a_i and b_i by

$$a_i^{k+1} = a_i^k + \rho \left(r_i^{k+1} - \sum_{j \in \mathcal{N}_i \cup i} r_{ij}^{k+1} \right), \tag{3.9}$$

$$b_i^{k+1} = b_i^k + \rho \left(s_i - \sum_{j \in \mathcal{N}_i \cup i \cup 0} s_{ij}^{k+1} \right). \tag{3.10}$$

For every $j \in \mathcal{N}_i \cup i \cup 0$, edge server i updates c_{ij} by

$$c_{ij}^{k+1} = c_{ij}^k + \rho (r_{ji}^{k+1} - s_{ij}^{k+1}). \tag{3.11}$$

When the cost functions f_0, f_i and g_{ij} (2.1) are convex, which is common in network resource allocation problems due to the law of diminishing marginal utility, ADMM guarantees global convergence to the optimal solution [20].

3.2. Algorithm Implementation

The collaborative resource allocation algorithm is summarized as follows. We breakdown the algorithm into two parts, the one run at the cloud center (see Table I) and the one run at every edge server i (see Table II).

At time $k+1$, the cloud center first updates the total amount of resource r_0 by (3.3) using a_0^k and $r_{0j}^k, \forall j = 1, \cdots, L$ that are locally available. Then from every edge server j, it collects c_{j0}^k and s_{j0}^{k+1}. The service assignments s_{j0}^{k+1} are available after all the edge servers j finish computing the values (see line 3, Table II). The values of c_{j0}^k and s_{j0}^{k+1} are used in updating the amount of resource r_{0j} given to every edge server j by (3.6), as well as updating a_0 by (3.8). The other variables in computing (3.6) and (3.8) are locally available.

Table I: Collaborative Resource Allocation Algorithm: Run at Cloud Center

1. **for** $k = 0, 1, \cdots$ **do**
2. Update the total amount of resource r_0 by (3.3)
$$r_0^{k+1} = \arg\min_{r_0} f_0(r_0) + \frac{\rho}{2}\left(r_0 + \frac{a_0^k}{\rho} - \sum_{j=1}^{L} r_{0j}^k\right)^2.$$
3. From every edge server j, collect c_{j0}^k and s_{j0}^{k+1}.
4. Update the amount of resource r_{0j} given to every edge server j by (3.6)
$$\{r_{0j}^{k+1}\} = \arg\min_{\{r_{0j}\}} \frac{\rho}{2}\sum_{j=1}^{L}\left(r_{0j} + \frac{c_{j0}^k}{\rho} - s_{j0}^{k+1}\right)^2 + \frac{\rho}{2}\left(r_0^{k+1} + \frac{a_0^k}{\rho} - \sum_{j=1}^{L} r_{0j}\right)^2.$$
5. Update a_0 by (3.8)
$$a_0^{k+1} = a_0^k + \rho\left(r_0^{k+1} - \sum_{j=1}^{L} r_{0j}^{k+1}\right).$$
6. **end for**

Table II: Collaborative Resource Allocation Algorithm: Run at Edge Server i

1. **for** $k = 0, 1, \cdots$ **do**
2. Update the total amount of resource r_i by (3.4)
$$r_i^{k+1} = \arg\min_{r_i} f_i(r_i) + \frac{\rho}{2}(r_i + \frac{a_i^k}{\rho} - \sum_{j\in\mathcal{N}_i\cup i} r_{ij}^k)^2.$$
3. Update the amount of service assigned to $j \in \mathcal{N}_i \cup i \cup 0$ by (3.5)
$$\{s_{ij}^{k+1}\} = \arg\min_{\{s_{ij}\geq 0\}} \sum_{j\in\mathcal{N}_i\cup 0} g_{ij}(s_{ij}) + \frac{\rho}{2}\sum_{j\in\mathcal{N}_i\cup i\cup 0}(r_{ji}^k + \frac{c_{ij}^k}{\rho} - s_{ij})^2 + \frac{\rho}{2}(s_i + \frac{b_i^k}{\rho} - \sum_{j\in\mathcal{N}_i\cup i\cup 0} s_{ij})^2.$$
4. From every $j \in \mathcal{N}_i$, collect c_{ji}^k and s_{ji}^{k+1}.
5. Update the amount of resource r_{ij} given to edge server $j \in \mathcal{N}_i \cup i$ by (3.7)
$$\{r_{ij}^{k+1}\} = \arg\min_{\{r_{ij}\}} \frac{\rho}{2}\sum_{j\in\mathcal{N}_i\cup i}(r_{ij} + \frac{c_{ji}^k}{\rho} - s_{ji}^{k+1})^2 + \frac{\rho}{2}(r_i^{k+1} + \frac{a_i^k}{\rho} - \sum_{j\in\mathcal{N}_i\cup i} r_{ij})^2.$$
6. From every $j \in \mathcal{N}_i \cup 0$, collect r_{ji}^{k+1}.
7. Update a_i by (3.9)
$$a_i^{k+1} = a_i^k + \rho(r_i^{k+1} - \sum_{j\in\mathcal{N}_i\cup i} r_{ij}^{k+1}).$$
8. Update b_i by (3.10)

$$b_i^{k+1} = b_i^k + \rho(s_i - \sum_{j \in \mathcal{N}_i \cup i \cup 0} s_{ij}^{k+1}).$$

9. Update c_{ij} for every $j \in \mathcal{N}_i \cup i \cup 0$ by (3.11)

$$c_{ij}^{k+1} = c_{ij}^k + \rho(r_{ji}^{k+1} - s_{ij}^{k+1}).$$

10. **end for**

Below we briefly analyze the communication and computation costs of the proposed algorithm. Suppose that exchanging one P-dimensional variable incurs a unit of communication. At every iteration, the cloud center collects c_{j0} and s_{j0} from all edge servers j, and thus has a communication cost of $2L$. Every edge server i collects r_{ji}, s_{ji} and c_{ji} from its neighbors $j \in \mathcal{N}_i$, while collects r_{0i} from the cloud center; the resulting communication cost is $3|\mathcal{N}_i| + 1$. Therefore, the network communication cost per iteration is

$$2L + \sum_{i=1}^{L} (3|\mathcal{N}_i| + 1) = 3L + 3\sum_{i=1}^{L} |\mathcal{N}_i|.$$

Given L, a dense network causes large network communication cost per iteration; however, it often requires a smaller number of iterations to reach a target accuracy.

The computation cost of the algorithm is dominated by calculating r_{0j} given to every edge server j from (3.6) in the cloud center, as well as calculating the amount of service assigned to $j \in \mathcal{N}_i \cup i \cup 0$ from (3.5) and the amount of resource r_{ij} given to edge server $j \in \mathcal{N}_i \cup i$ from (3.7), both in every edge server i. The quadratic program (3.6) leads the computation cost of $O(L^3)$. Similarly, the cost of solving (3.7) is $O\left((|\mathcal{N}_i| + 1)^3\right)$. When the cost function g_{ij} is quadratic, the cost of solving (3.5) is $O\left((|\mathcal{N}_i| + 2)^3\right)$. Therefore, the computation cost per iteration is given by

$$O(L^3) + \sum_{i=1}^{L} O\left((|\mathcal{N}_i| + 1)^3 + (|\mathcal{N}_i| + 2)^3\right).$$

Remark 3.1 (Optimization over An Edge-Only Network). Observe that when we eliminate the cloud center, the network is only composed of edge servers and no longer hybrid. To implement the algorithm in such a network infrastructure, we can simply let every edge server i run the algorithm in Table II, while setting $s_{ij} = 0$ and $r_{ji} = 0$ for $j = 0$. In this case, the communication and computation costs per iteration are $3\sum_{i=1}^{L} |\mathcal{N}_i|$ and $\sum_{i=1}^{L} O\left((|\mathcal{N}_i| + 1)^3 + (|\mathcal{N}_i| + 2)^3\right)$, respectively.

3.3. Resources, Prices, and Decomposition

The proposed algorithm has an economic explanation. Recall that the nodes (the cloud center and the edge servers) have resources of computation and/or storage, denoted by the primal variables r_i, $i = 0, 1, \cdots, L$. Node i can assign its resource to node j with an amount of r_{ij}. Here $j = 1, \cdots, L$ if $i = 0$ and $j \in \mathcal{N}_i \cup i$ if $i \neq 0$. Meanwhile, the services carried by the communication links also cost resources, denoted by another set of primal variables s_{ij} where $i = 1, \cdots, L$ and $j \in \mathcal{N}_i \cup i \cup 0$. With particular note, self-assigned services (denoted by s_{ii}) introduce no communication costs while services assigned to the cloud center (denoted by s_{i0}) require high communication costs.

For every edge server i, its goal is to satisfy the user request s_i using the computation/storage resources of itself, its neighbors and the cloud center, as well as the communication resources of the attached communication links. Apparently, network-wide optimal resource allocation requires collaboration of all the nodes. The decomposition technique in this paper enables autonomous collaboration of the nodes through introducing the dual variables a_0, $\{a_i\}$, $\{b_i\}$, and $\{c_{ij}\}$, which stand for prices of the resources.

The dual variable a_0 can be explained as the price of the computation/storage resource at the cloud center. There is a mismatch between the overall resource r_0 and the summation of the assigned resources $\sum_{j=1}^{L} r_{0j}$, showing the scarcity of the computation/storage power and reflected by the price a_0. Similarly, a_i stands for the price of the computation/storage resource at edge server i, evaluated by the gap between the available and assigned resources, denoted by r_i and $\sum_{j\in\mathcal{N}_i\cup i} r_{ij}$, respectively. The dual variables b_i are the prices of not satisfying the user requests, which are quantified by the values $s_i - \sum_{j\in\mathcal{N}_i\cup i\cup 0} s_{ij}$. Finally, the dual variables c_{ij} show the prices of spending resources r_{ji} to satisfy the service requests s_{ij} through the communication links (i,j), $i = 1,\cdots,L$, $j \in \mathcal{N}_i \cup i \cup 0$.

4. Numerical Experiments

In the numerical experiments, we consider the following three network infrastructures.

(i) **Cloud-Only.** The edge servers only receive user requests but without any local processing. They simply forward the user requests to the cloud center.

(ii) **Edge-Only.** The edge servers collaboratively handle user requests without the help of the cloud center. The resource allocation algorithm is hence a simplified version compared to the proposed one, as we have discussed in Remark 1 in Section 3.

(iii) **Cloud-Edge.** The edge servers and the cloud center collaboratively handle user requests using the proposed algorithm.

For all the three infrastructures, the network contains 20 distributed edge servers. In the edge-only and the cloud-edge infrastructures, 66 undirected links between neighboring edge servers out of 190 possible ones are uniformly randomly chosen to be connected. The values of the user requests s_i are uniformly randomly generated in the range from 0 to 30.

For simplicity, we assume that there is only one kind of service request, namely, $P = 1$ such that all the primal and dual variables in (3.2) are scalars. Set the computation/storage cost of the cloud center as $f_0(r_0) = \exp(0.01r_0) - 1$ while those of the edge servers as $f_i(r_i) = \exp(0.2r_i) - 1$. The communication cost between two neighboring edge servers i and j is $g_{ij}(s_{ij}) = \exp(0.05s_{ij}) - 1$, and that between edge server i and the cloud center as $g_{ij}(s_{i0}) = \exp(0.1s_{i0}) - 1$. Notice that the edge-cloud communication cost is higher than that between two edge servers. Meanwhile, the computation/storage cost at a cloud center is significantly smaller than that at an edge server. We define *residual* as the performance metric, denoting the normalized difference between the current iterate s_{ij}^k and the optimal primal solution s_{ij}^* of (2.1), given by

$$\frac{\sum_{i=1}^{L}\sum_{j\in\mathcal{N}_i\cup i\cup 0}\|s_{ij}^k - s_{ij}^*\|}{\sum_{i=1}^{L}\sum_{j\in\mathcal{N}_i\cup i\cup 0}\|s_{ij}^*\|}.$$

In the first experiment, we demonstrate the convergence of the proposed algorithm in the cloud-edge infrastructure in Fig. 3. We vary the algorithm parameter ρ, the ADMM stepsize, to different values. As we can observe from the result, the proposed collaborative resource allocation algorithm always converges to the optimal solution. The ADMM stepsize ρ determines the convergence speed of the algorithm. In this experiment, setting ρ as a value between 0.1 and 0.15 achieves the fastest convergence. We also show the performance of the dual decomposition method, whose stepsize is tuned to the best value of 0.2. Observe that ADMM has much faster convergence than the dual decomposition method.

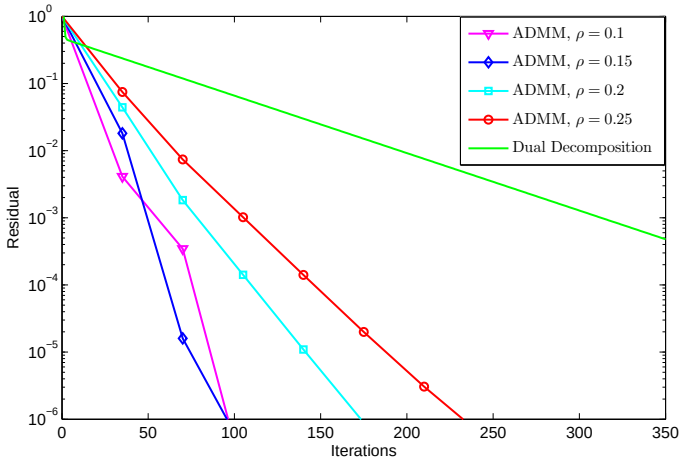

Fig. 3. Performance of the collaborative resource allocation algorithm over a cloud-edge network, with the algorithm parameter ρ being varied. We also show the dual decomposition algorithm with stepsize 0.2 as a comparison.

In the second experiment, we demonstrate the convergence of the proposed algorithm in the edge-only infrastructure. We tune the value of ρ as shown in Fig. 4. In this degenerated setting, the proposed algorithm also demonstrates similar convergence performance as in Fig. 3 and significantly outperforms the dual decomposition method.

In the third experiment, we compare the overall communication costs and the computation/storage costs of using three network infrastructures, which are given by Table III. Observe that the cloud-edge infrastructure incurs the smallest overall cost comparing to the edge-only and cloud-only infrastructures. This makes sense because the latter two are both special cases of the hybrid edge server and cloud center network, and hence only give suboptimal solutions. Compared to the edge-only network, the cloud-edge network costs more in cloud computation/storage and cloud communication. However, it slightly reduces the edge communication cost, and significantly reduces the edge computation/storage cost because the edge servers are able to assign those computation- and storage-demanding requests to the cloud center. The cloud-only network, on the other hand, has no burdens of edge communication and edge computation/storage. However, it requires the edge servers to route all the user requests to the cloud center, which wastes the computation and storage powers of the edge servers and brings high cloud computation/storage and cloud communication costs. These results demonstrate the power of allowing cloud-edge collaboration over such a hybrid network infrastructure and the effectiveness of the proposed collaborative resource allocation algorithm.

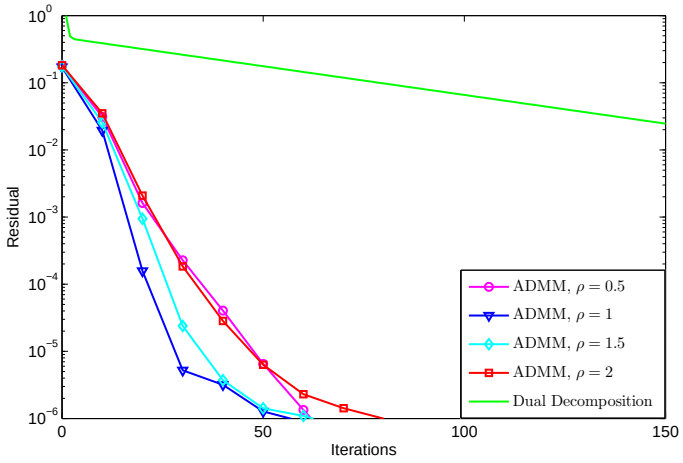

Fig. 4. Performance of the collaborative resource allocation algorithm over an edge-only network, with the algorithm parameter ρ being varied. We also show the dual decomposition algorithm with stepsize 0.2 as a comparison.

Table III: Optimal Costs of Three Network Infrastructures

type of cost	cloud-edge	edge-only	cloud-only
edge computation/storage cost	21.1890	283.6466	0
cloud computation/storage cost	6.4117	0	14.1803
edge-edge communication cost	2.1546	4.7165	0
cloud-edge communication cost	37.6300	0	81.2308
overall cost	67.3853	288.3631	95.4111

5. Conclusions

In this paper, we consider a novel network composed of one cloud center and multiple edge servers, which takes advantages of both cloud computing that fits for computation- and storage-intensive applications and edge computing (also known as fog computing) that provides fast response. Through sharing resources among neighboring edge servers and the cloud center, every edge server is able to bring elastic network services to its end users. We formulate the collaborative resource allocation problem, whose decomposable structure enables the use of the alternating direction method of multipliers. The resultant algorithm naturally decomposes computations onto the cloud center and the edge servers and leads to autonomous collaboration among these nodes. We discuss the connections between our work and the existing resource allocation models and algorithms.

One of our future research directions is to reduce the complexities in computing local variables (see the updates of (3.3), (3.4), and (3.5), all of which require to solve optimization problems). Another topic of particular interest is to apply the technique of Nesterov's acceleration in this problem so as to expedite the convergence. We will also consider applying the proposed model and algorithm in practical network service scenarios, for example, distributed surveillance video processing and distributed online stream media service.

Acknowledgements. Houfeng Huang and Qing Ling are supported by NSF China grant 61573331, NSF Anhui grant 1608085QF130 and CAS grant XDA06040602. Jinlin Wang is supported by CAS grant XDA06040602. The authors are grateful to Dr. Ming Zhao for his constructive comments during the writing of this paper.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, A view of cloud computing, *Communications of the ACM*, **53**:4 (2010), 50-58.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, Fog computing and its role in the Internet of things, *MCC*, 2012.

[3] M. Chiang, Fog networking: An overview on research opportunities, Manuscript

[4] R. Johari and J. Tsitsiklis, Efficiency loss in a network resource allocation game, *Mathematics of Operations Research*, **29**:3 (2004), 407-435.

[5] D. Palomar and M. Chiang, A tutorial on decomposition methods for network utility maximization, *IEEE Journal on Selected Areas in Communications*, **24**:8 (2006), 1439-1451.

[6] Y. Xue, B. Li, and K. Nahrstedt, Optimal resource allocation in wireless ad hoc networks: A price-based approach, *IEEE Transactions on Mobile Computing*, **5**:4 (2006), 347-364.

[7] X. Lin and N. Shroff, Utility maximization for communication networks with multipath routing, *IEEE Transactions on Automatic Control*, **15**:5 (2006), 766-781.

[8] A. Beck, A. Nedic, A. Ozdaglar, and M. Teboulle, Optimal distributed gradient methods for network resource allocation problems, Manuscript.

[9] S. Low and D. Lapsley, Optimization flow control–I: Basic algorithm and convergence, *IEEE Transactions on Networking*, **7**:6 (1999), 861-874.

[10] A. Nedic and A. Ozdaglar, Distributed subgradient methods for multiagent optimization, *IEEE Transactions on Automatic Control*, **54**:1 (2009), 48-61.

[11] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, On the linear convergence of the ADMM in decentralized consensus optimization, *IEEE Transactions on Signal Processing*, **62**:7 (2014), 1750-1761.

[12] W. Shi, Q. Ling, G. Wu, and W. Yin, EXTRA: An exact first-order algorithm for decentralized consensus optimization, *SIAM Journal on Optimization*, **25**:2 (2015), 944-966.

[13] T. Chang, M. Hong, and X. Wang, Multi-agent distributed optimization via inexact consensus ADMM, *IEEE Transactions on Signal Processing*, **63**:2 (2015), 482-497.

[14] M. Hale, A. Nedic, and M. Egerstedt, Cloud-based centralized/decentralized multi-agent optimization with communication delays, Manuscript.

[15] C. Feng, H. Xu, and B. Li, An alternating direction method approach to cloud traffic management, Manuscript.

[16] W. Wang, B. Liang, and B. Li, Multi-resource fair allocation in heterogeneous cloud computing systems, Manuscript.

[17] G. Giannakis, V. Kekatos, N. Gatsis, S. Kim, H. Zhu, and B. Wollenberg, Monitoring and optimization for power grids: A signal processing perspective, *IEEE Signal Processing Magazine*, **30**:5 (2013), 107-128.

[18] R. Deng, Z. Yang, M. Chow, and J. Chen, A survey on demand response in smart grids: Mathematical models and approaches, *IEEE Transactions on Industrial Informatics*, **11**:3 (2015), 570-582.

[19] D. Gabay and B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation, *Computers and Mathematics with Applications*, **2**:1 (1976), 17-40.

[20] J. Eckstein and D. Bertsekas, On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators, *Mathematical Programming*, **55**:1-3 (1992), 293-318.

[21] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, *Foundations and Trends in Machine Learning*, **3**:1 (2010), 1-122.

[22] W. Deng and W. Yin, On the global and linear convergence of the generalized alternating direction method of multipliers, *Journal of Scientific Computing*, **66**:3 (2016), 889-916.