Distributed Optimization Using the Primal-Dual Method of Multipliers

Guoqiang Zhang and Richard Heusdens

***Abstract*—In this paper, we propose the primal-dual method of multipliers (PDMM) for distributed optimization over a graph. In particular, we optimize a sum of convex functions defined over a graph, where every edge in the graph carries a linear equality constraint. In designing the new algorithm, an augmented primal-dual Lagrangian function is constructed which smoothly captures the graph topology. It is shown that a saddle point of the constructed function provides an optimal solution of the original problem. Further under both the synchronous and asynchronous updating schemes, PDMM has the convergence rate of $O(1/K)$ (where K denotes the iteration index) for general closed, proper, and convex functions. Other properties of PDMM such as convergence speeds versus different parameter-settings and resilience to transmission failure are also investigated through the experiments of distributed averaging.**

***Index Terms*—ADMM, distributed optimization, PDMM, sublinear convergence.**

I. Introduction

In recent years, distributed optimization has drawn increasing attention due to the demand for big-data processing and easy access to ubiquitous computing units (e.g., a computer, a mobile phone or a sensor equipped with a CPU). The basic idea is to have a set of computing units collaborate with each other in a distributed way to complete a complex task. Popular applications include telecommunication [3], [4], wireless sensor networks [5], cloud computing and machine learning [6]. The research challenge is on the design of efficient and robust distributed optimization algorithms for those applications.

To the best of our knowledge, almost all the optimization problems in those applications can be formulated as optimization over a graphic model $G = (\mathcal{V}, \mathcal{E})$:

$$\min_{\{\boldsymbol{x}_i\}} \sum_{i\in\mathcal{V}} f_i(\boldsymbol{x}_i) + \sum_{(i,j)\in\mathcal{E}} f_{ij}(\boldsymbol{x}_i, \boldsymbol{x}_j), \qquad (1)$$

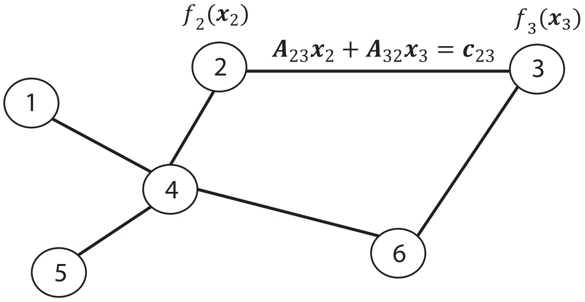

$f_2(\boldsymbol{x}_2)$
$f_3(\boldsymbol{x}_3)$
$\boldsymbol{A}_{23}\boldsymbol{x}_2 + \boldsymbol{A}_{32}\boldsymbol{x}_3 = \boldsymbol{c}_{23}$
1
2
3
4
5
6

Fig. 1. Demonstration of Problem (1) for edge-functions being linear constraints. Every edge in the graph carries an equality constraint.

where $\{f_i | i \in \mathcal{V}\}$ and $\{f_{ij} | (i,j) \in \mathcal{E}\}$ are referred to as node and edge-functions, respectively. For instance, for the application of distributed quadratic optimization, all the node and edge-functions are in the form of scalar quadratic functions (see [7]–[9]).

In the literature, a large number of applications (see [10]) require that every edge function $f_{ij}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $(i,j) \in \mathcal{E}$, is essentially a linear equality constraint in terms of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Mathematically, we use $\boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{A}_{ji}\boldsymbol{x}_j = \boldsymbol{c}_{ij}$ to formulate the equality constraint for each $(i,j) \in \mathcal{E}$, as demonstrated in Fig. 1. In this situation, (1) can be described as

$$\min_{\{\boldsymbol{x}_i\}} \sum_{i\in\mathcal{V}} f_i(\boldsymbol{x}_i) + \sum_{(i,j)\in\mathcal{E}} I_{\boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{A}_{ji}\boldsymbol{x}_j = \boldsymbol{c}_{ij}}(\boldsymbol{x}_i, \boldsymbol{x}_j), \qquad (2)$$

where $I_{(\cdot)}$ denotes the indicator or characteristic function defined as $I_{\mathcal{C}}(\boldsymbol{x}) = 0$ if $\boldsymbol{x} \in \mathcal{C}$ and $I_{\mathcal{C}}(\boldsymbol{x}) = \infty$ if $\boldsymbol{x} \notin \mathcal{C}$. In this paper, we focus on convex optimization of form (2), where every node-function f_i is closed, proper and convex.

The majority of recent research have been focusing on a specialized form of the convex problem (2), where every edge-function f_{ij} reduces to $I_{\boldsymbol{x}_i = \boldsymbol{x}_j}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The above problem is commonly known as the *consensus problem* in the literature. Classic methods include the dual-averaging algorithm [11], the subgradient algorithm [12], the diffusion adaptation algorithm [13]. For the special case that $\{f_i | i \in \mathcal{V}\}$ are scalar quadratic functions (referred to as the *distributed averaging* problem), the most popular methods are the randomized gossip algorithm [5] and the broadcast algorithm [14]. See [15] for an overview of the literature for solving the distributed averaging problem.

The alternating-direction method of multipliers (ADMM) can be applied to solve the general convex optimization (2). The key step is to decompose each equality constraint $\boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{A}_{ji}\boldsymbol{x}_j =$

Manuscript received March 12, 2016; revised October 15, 2016 and January 11, 2017; accepted February 6, 2017. Date of publication February 20, 2017; date of current version February 19, 2018. This paper was presented in part at the International Conference on Acoustics, Speech, and Signal Processing, Florence, Italy, May 2014, and in part at International Conference on Acoustics, Speech, and Signal Processing, South Brisbane, Qld., Australia, April 2015. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Sergio Barbarossa.

G. Zhang is with the School of Computing and Communications, University of Technology Sydney, Ultimo, NSW 2007, Australia (e-mail: guoqiang.zhang@uts.edu.au).

R. Heusdens is with the Department of Microelectronics, Circuits and Systems group, Delft University of Technology, Delft 2628 CD, The Netherlands (e-mail: r.heusdens@tudelft.nl).

Digital Object Identifier 10.1109/TSIPN.2017.2672403

$\boldsymbol{c}_{ij}$ into two constraints such as $\boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{z}_{ij} = \boldsymbol{c}_{ij}$ and $\boldsymbol{z}_{ij} = \boldsymbol{A}_{ji}\boldsymbol{x}_j$ with the help of the auxiliary variable $\boldsymbol{z}_{ij}$. As a result, (2) can be reformulated as

$$\min_{\boldsymbol{x},\boldsymbol{z}} f(\boldsymbol{x}) + g(\boldsymbol{z}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c}, \tag{3}$$

where $f(\boldsymbol{x}) = \sum_{i\in\mathcal{V}} f_i(\boldsymbol{x}_i)$, $g(\boldsymbol{z}) = 0$ and $\boldsymbol{z}$ is a vector obtained by stacking up $\boldsymbol{z}_{ij}$ one after another. See [16] for using ADMM to solve the consensus problem of (2) (with edge-function $I_{\boldsymbol{x}_i=\boldsymbol{x}_j}(\boldsymbol{x}_i, \boldsymbol{x}_j)$). The graphic structure is implicitly embedded in the two matrices $(\boldsymbol{A}, \boldsymbol{B})$ and the vector $\boldsymbol{c}$. The reformulation essentially converts the problem on a general graph with many nodes (2) to a graph with only two nodes (3), allowing the application of ADMM. Based on (3), ADMM then constructs and optimizes an augmented Lagrangian function iteratively with respect to $(\boldsymbol{x}, \boldsymbol{z})$ and a set of Lagrangian multipliers. We refer to the above procedure as synchronous ADMM as it updates all the variables at each iteration. Recently, the work of [17] proposed asynchronous ADMM, which optimizes the same function over a subset of the variables at each iteration.

We note that besides solving (2), ADMM has found many successful applications in the fields of signal processing and machine learning (see [10] for an overview). For instance, in [18] and [19], variants of ADMM have been proposed to solve a (possibly nonconvex) optimization problem defined over a graph with a star topology, which is motivated from big data applications. The work of [20] considers solving the consensus problem of (2) (with edge-function $I_{\boldsymbol{x}_i=\boldsymbol{x}_j}(\boldsymbol{x}_i, \boldsymbol{x}_j)$) over a general graph, where each node function f_i is further expressed as a sum of two component functions. The authors of [20] propose a new algorithm which includes ADMM as a special case when one component function is zero. In general, ADMM and its variants are quite simple and often provide satisfactory results after a reasonable number of iterations, making it a popular algorithm in recent years.

In this paper, we tackle the convex problem (2) directly instead of relying on the reformulation (3). Specifically, we construct an augmented primal-dual Lagrangian function for (2) without introducing the auxiliary variable $\boldsymbol{z}$ as is required by ADMM. We show that solving (2) is equivalent to searching for a saddle point of the augmented primal-dual Lagrangian. We then propose the primal-dual method of multipliers (PDMM) to iteratively approach one saddle point of the constructed function. It is shown that for both the synchronous and asynchronous updating schemes, the PDMM converges with the rate of $\mathcal{O}(1/K)$ for general closed, proper and convex functions.

Further we evaluate PDMM through the experiments of distributed averaging. Firstly, it is found that the parameters of PDMM should be selected by a rule (see VI-C1) for fast convergence. Secondly, when there are transmission failures in the graph, transmission losses only slow down the convergence speed of PDMM. Finally, experimental comparison suggests that PDMM outperforms ADMM and the two gossip algorithms in [5] and [14].

This work is mainly devoted to the theoretical analysis of PDMM. In the literature, PDMM has already been successfully applied for solving a few other problems. The work of [21] investigates the efficiency of ADMM and PDMM for distributed dictionary learning. In [22], we have used both ADMM and PDMM for training a support vector machine (SVM). In the above examples it is found that PDMM outperforms ADMM in terms of convergence rate. In [23], the authors describes an application of the linearly constrained minimum variance (LCMV) beamformer for use in acoustic wireless sensor networks. The proposed algorithm computes the optimal beamformer output at each node in the network without the need for sharing raw data within the network. PDMM has been successfully applied to perform distributed beamforming. This suggests that PDMM is not only theoretically interesting but also might be powerful in real applications.

II. Problem Setting

In this section, we first introduce basic notations needed in the rest of the paper. We then make a proper assumption about the existence of optimal solutions of the problem. Finally, we derive the dual problem to (2) and its Lagrangian function, which will be used for constructing the augmented primal-dual Lagrangian function in Section III.

A. Notations and Functional Properties

We first introduce notations for a graphic model. We denote a graph as $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, m\}$ represents the set of nodes and $\mathcal{E} = \{(i,j)|i,j \in \mathcal{V}\}$ represents the set of edges in the graph, respectively. We use $\vec{\mathcal{E}}$ to denote the set of all directed edges. Therefore, $|\vec{\mathcal{E}}| = 2|\mathcal{E}|$. The directed edge $[i,j]$ starts from node i and ends with node j. We use $\mathcal{N}_i$ to denote the set of all neighboring nodes of node i, i.e., $\mathcal{N}_i = \{j|(i,j) \in \mathcal{E}\}$. Given a graph $G = (\mathcal{V}, \mathcal{E})$, only neighboring nodes are allowed to communicate with each other directly.

Next we introduce notations for mathematical description in the remainder of the paper. We use bold small letters to denote vectors and bold capital letters to denote matrices. The notation $\boldsymbol{M} \succeq 0$ (or $\boldsymbol{M} \succ 0$) represents a symmetric positive semi-definite matrix (or a symmetric positive definite matrix). The superscript $(\cdot)^T$ represents the transpose operator. Given a vector $\boldsymbol{y}$, we use $\|\boldsymbol{y}\|$ to denote its l_2 norm.

Finally, we introduce the conjugate function. Suppose $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is a closed, proper and convex function. Then the conjugate of $h(\cdot)$ is defined as [24, Definition 2.1.20]

$$h^*(\boldsymbol{\delta}) \stackrel{\Delta}{=} \max_{\boldsymbol{y}} \boldsymbol{\delta}^T\boldsymbol{y} - h(\boldsymbol{y}), \tag{4}$$

where the conjugate function h^* is again a closed, proper and convex function. Let $\boldsymbol{y}'$ be the optimal solution for a particular $\boldsymbol{\delta}'$ in (4). We then have

$$\boldsymbol{\delta}' \in \partial_{\boldsymbol{y}} h(\boldsymbol{y}'), \tag{5}$$

where $\partial_{\boldsymbol{y}} h(\boldsymbol{y}')$ represents the set of all subgradients of $h(\cdot)$ at $\boldsymbol{y}'$ (see [24, Definition 2.1.23]). As a consequence, since $h^{**} = h$, we have

$$h(\boldsymbol{y}') = \boldsymbol{y}'^T\boldsymbol{\delta}' - h^*(\boldsymbol{\delta}') = \max_{\boldsymbol{\delta}} \boldsymbol{y}'^T\boldsymbol{\delta} - h^*(\boldsymbol{\delta}), \tag{6}$$

and we conclude that $\boldsymbol{y}' \in \partial_{\boldsymbol{\delta}} h^*(\boldsymbol{\delta}')$ as well.

B. Problem Assumption

With the notation $G=(\mathcal{V},\mathcal{E})$ for a graph, we first reformulate the convex problem (2) as

$$\min_{\boldsymbol{x}} \sum_{i\in\mathcal{V}} f_i(\boldsymbol{x}_i) \text{ s.t. } \boldsymbol{A}_{ij}\boldsymbol{x}_i+\boldsymbol{A}_{ji}\boldsymbol{x}_j=\boldsymbol{c}_{ij}\ \forall(i,j)\in\mathcal{E}, \tag{7}$$

where each function $f_i:\mathbb{R}^{n_i}\to\mathbb{R}\cup\{+\infty\}$ is assumed to be closed, proper and convex, and $\boldsymbol{x}=[\boldsymbol{x}_1^T,\boldsymbol{x}_2^T,\ldots,\boldsymbol{x}_m^T]^T$. For every edge $(i,j)\in\mathcal{E}$, we let $(\boldsymbol{c}_{ij},\boldsymbol{A}_{ij},\boldsymbol{A}_{ji})\in(\mathbb{R}^{n_{ij}},\mathbb{R}^{n_{ij}\times n_i},\mathbb{R}^{n_{ij}\times n_j})$. The vector $\boldsymbol{x}$ is thus of dimension $n_{\boldsymbol{x}}=\sum_{i\in\mathcal{V}} n_i$. In general, $\boldsymbol{A}_{ij}$ and $\boldsymbol{A}_{ji}$ are two different matrices. The matrix $\boldsymbol{A}_{ij}$ operates on $\boldsymbol{x}_i$ in the linear constraint of edge $(i,j)\in\mathcal{E}$. The notation s.t. in (7) stands for "subject to". We take the reformulation (7) as the *primal* problem.

The primal Lagrangian for (7) can be constructed as

$$L_p(\boldsymbol{x},\boldsymbol{\delta})=\sum_{i\in\mathcal{V}} f_i(\boldsymbol{x}_i)+\sum_{(i,j)\in\mathcal{E}}\boldsymbol{\delta}_{ij}^T(\boldsymbol{c}_{ij}-\boldsymbol{A}_{ij}\boldsymbol{x}_i-\boldsymbol{A}_{ji}\boldsymbol{x}_j), \tag{8}$$

where $\boldsymbol{\delta}_{ij}$ is the Lagrangian multiplier (or the dual variable) for the corresponding edge constraint in (7), and the vector $\boldsymbol{\delta}$ is obtained by stacking all the dual variables $\boldsymbol{\delta}_{ij}$, $(i,j)\in\mathcal{E}$, on top of one another. Therefore, $\boldsymbol{\delta}$ is of dimension $n_{\boldsymbol{\delta}}=\sum_{(i,j)\in\mathcal{E}} n_{ij}$. The Lagrangian function is convex in $\boldsymbol{x}$ for fixed $\boldsymbol{\delta}$, and concave in $\boldsymbol{\delta}$ for fixed $\boldsymbol{x}$. Throughout the rest of the paper, we will make the following (common) assumption:

Assumption 1: There exists a saddle point $(\boldsymbol{x}^\star,\boldsymbol{\delta}^\star)$ to the Lagrangian function $L_p(\boldsymbol{x},\boldsymbol{\delta})$ such that for all $\boldsymbol{x}\in\mathbb{R}^{n_{\boldsymbol{x}}}$ and $\boldsymbol{\delta}\in\mathbb{R}^{n_{\delta}}$ we have

$$L_p(\boldsymbol{x}^\star,\boldsymbol{\delta})\le L_p(\boldsymbol{x}^\star,\boldsymbol{\delta}^\star)\le L_p(\boldsymbol{x},\boldsymbol{\delta}^\star).$$

Or equivalently, the following optimality (KKT) conditions hold for $(\boldsymbol{x}^\star,\boldsymbol{\delta}^\star)$:

$$\sum_{j\in\mathcal{N}_i}\boldsymbol{A}_{ij}^T\boldsymbol{\delta}_{ij}^\star\in\partial f_i(\boldsymbol{x}_i^\star)\qquad\forall i\in\mathcal{V} \tag{9}$$

$$\boldsymbol{A}_{ji}\boldsymbol{x}_j^\star+\boldsymbol{A}_{ij}\boldsymbol{x}_i^\star=\boldsymbol{c}_{ij}\qquad\forall(i,j)\in\mathcal{E}. \tag{10}$$

C. Dual Problem and Its Lagrangian Function

We first derive the dual problem to (7). Optimizing $L_p(\boldsymbol{x},\boldsymbol{\delta})$ over $\boldsymbol{\delta}$ and $\boldsymbol{x}$ yields

$$\begin{aligned}
&\max_{\boldsymbol{\delta}}\min_{\boldsymbol{x}} L_p(\boldsymbol{x},\boldsymbol{\delta})\\
&=\max_{\boldsymbol{\delta}}\sum_{i\in\mathcal{V}}\min_{\boldsymbol{x}_i}\Big(f_i(\boldsymbol{x}_i)-\sum_{j\in\mathcal{N}_i}\boldsymbol{\delta}_{ij}^T\boldsymbol{A}_{ij}\boldsymbol{x}_i\Big)+\sum_{(i,j)\in\mathcal{E}}\boldsymbol{\delta}_{ij}^T\boldsymbol{c}_{ij}\\
&=\max_{\boldsymbol{\delta}}\sum_{i\in\mathcal{V}}-f_i^*\left(\sum_{j\in\mathcal{N}_i}\boldsymbol{A}_{ij}^T\boldsymbol{\delta}_{ij}\right)+\sum_{(i,j)\in\mathcal{E}}\boldsymbol{\delta}_{ij}^T\boldsymbol{c}_{ij},
\end{aligned}\tag{11}$$

where $f_i^*(\cdot)$ is the conjugate function of $f_i(\cdot)$ as defined in (4), satisfying Fenchel's inequality

$$f_i(\boldsymbol{x}_i)+f_i^*\left(\sum_{j\in\mathcal{N}_i}\boldsymbol{A}_{ij}^T\boldsymbol{\delta}_{ij}\right)\ge\sum_{j\in\mathcal{N}_i}\boldsymbol{\delta}_{ij}^T\boldsymbol{A}_{ij}\boldsymbol{x}_i. \tag{12}$$

Under Assumption 1, the dual problem (11) is equivalent to the primal problem (7). That is suppose $(\boldsymbol{x}^\star,\boldsymbol{\delta}^\star)$ is a saddle point of L_p. Then $\boldsymbol{x}^\star$ solves the primal problem (7) and $\boldsymbol{\delta}^\star$ solves the dual problem (11).

At this point, we need to introduce auxiliary variables to decouple the node dependencies in (11). Indeed, every $\boldsymbol{\delta}_{ij}$, associated to edge (i,j), is used by two conjugate functions f_i^* and f_j^*. As a consequence, all conjugate functions in (11) are dependent on each other. To decouple the conjugate functions, we introduce for each edge $(i,j)\in\mathcal{E}$ *two* auxiliary *node* variables $\boldsymbol{\lambda}_{i|j}\in\mathbb{R}^{n_{ij}}$ and $\boldsymbol{\lambda}_{j|i}\in\mathbb{R}^{n_{ij}}$, one for each node i and j, respectively. The node variable $\boldsymbol{\lambda}_{i|j}$ is owned by and updated at node i and is related to neighboring node j. Hence, at every node i we introduce $|\mathcal{N}_i|$ new node variables. With this, we can reformulate the original dual problem as

$$\begin{aligned}
&\max_{\boldsymbol{\delta},\{\boldsymbol{\lambda}_i\}}\ -\sum_{i\in\mathcal{V}}f_i^*(\boldsymbol{A}_i^T\boldsymbol{\lambda}_i)+\sum_{(i,j)\in\mathcal{E}}\boldsymbol{\delta}_{ij}^T\boldsymbol{c}_{ij}\\
&\qquad\text{s. t.}\quad\boldsymbol{\lambda}_{i|j}=\boldsymbol{\lambda}_{j|i}=\boldsymbol{\delta}_{ij}\quad\forall(i,j)\in\mathcal{E},
\end{aligned}\tag{13}$$

where $\boldsymbol{\lambda}_i$ is obtained by vertically concatenating all $\boldsymbol{\lambda}_{i|j}$, $j\in\mathcal{N}_i$, and $\boldsymbol{A}_i^T$ is obtained by horizontally concatenating all $\boldsymbol{A}_{ij}^T$, $j\in\mathcal{N}_i$. To clarify, the product $\boldsymbol{A}_i^T\boldsymbol{\lambda}_i$ in (13) equals to

$$\boldsymbol{A}_i^T\boldsymbol{\lambda}_i=\sum_{j\in\mathcal{N}_i}\boldsymbol{A}_{ij}^T\boldsymbol{\lambda}_{i|j}. \tag{14}$$

Consequently, we let $\boldsymbol{\lambda}=[\boldsymbol{\lambda}_1^T,\boldsymbol{\lambda}_2^T,\ldots,\boldsymbol{\lambda}_m^T]^T$. In the above reformulation (13), each conjugate function $f_i^*(\cdot)$ only involves the *node* variable $\boldsymbol{\lambda}_i$, facilitating distributed optimization.

Next we tackle the equality constraints in (13). To do so, we construct a (dual) Lagrangian function for the dual problem (13), which is given by

$$\begin{aligned}
L_d'(\boldsymbol{\delta},\boldsymbol{\lambda},\boldsymbol{y})=&-\sum_{i\in\mathcal{V}}f_i^*(\boldsymbol{A}_i^T\boldsymbol{\lambda}_i)+\sum_{(i,j)\in\mathcal{E}}\boldsymbol{\delta}_{ij}^T\boldsymbol{c}_{ij}\\
&+\sum_{(i,j)\in\mathcal{E}}\Big[\boldsymbol{y}_{i|j}^T(\boldsymbol{\delta}_{ij}-\boldsymbol{\lambda}_{i|j})+\boldsymbol{y}_{j|i}^T(\boldsymbol{\delta}_{ij}-\boldsymbol{\lambda}_{j|i})\Big],
\end{aligned}\tag{15}$$

where $\boldsymbol{y}$ is obtained by concatenating all the Lagrangian multipliers $\boldsymbol{y}_{i|j}$, $[i,j]\in\vec{\mathcal{E}}$, one after another.

We now argue that each Lagrangian multiplier $\boldsymbol{y}_{i|j}$, $[i,j]\in\vec{\mathcal{E}}$, in (15) can be replaced by an affine function of $\boldsymbol{x}_j$. Suppose $(\boldsymbol{x}^\star,\boldsymbol{\delta}^\star)$ is a saddle point of L_p. By letting $\boldsymbol{\lambda}_{i|j}^\star=\boldsymbol{\delta}_{ij}^\star$ for every $[i,j]\in\vec{\mathcal{E}}$, Fenchel's inequality (12) must hold with equality at $(\boldsymbol{x}^\star,\boldsymbol{\lambda}^\star)$ from which we derive that

$$\begin{aligned}
\boldsymbol{0}&\in\partial_{\boldsymbol{\lambda}_{i|j}}\left[f_i^*(\boldsymbol{A}_i^T\boldsymbol{\lambda}_i^\star)\right]-\boldsymbol{A}_{ij}\boldsymbol{x}_i^\star\\
&=\partial_{\boldsymbol{\lambda}_{i|j}}\left[f_i^*(\boldsymbol{A}_i^T\boldsymbol{\lambda}_i^\star)\right]+\boldsymbol{A}_{ji}\boldsymbol{x}_j^\star-\boldsymbol{c}_{ij}\quad\forall[i,j]\in\vec{\mathcal{E}}.
\end{aligned}$$

One can then show that $(\boldsymbol{\delta}^\star,\boldsymbol{\lambda}^\star,\boldsymbol{y}^\star)$ where $\boldsymbol{y}_{i|j}^\star=\boldsymbol{A}_{ji}\boldsymbol{x}_j^\star-\boldsymbol{c}_{ij}$ for every $[i,j]\in\vec{\mathcal{E}}$, is a saddle point of L_d'. We therefore restrict the Lagrangian multiplier $\boldsymbol{y}_{i|j}$ to be of the form $\boldsymbol{y}_{i|j}=\boldsymbol{A}_{ji}\boldsymbol{x}_j-\boldsymbol{c}_{ij}$ so that the dual Lagrangian becomes

$$\begin{aligned}
L_d(\boldsymbol{\delta},\boldsymbol{\lambda},\boldsymbol{x})=&\sum_{i\in\mathcal{V}}\Big(-f_i^*(\boldsymbol{A}_i^T\boldsymbol{\lambda}_i)-\sum_{j\in\mathcal{N}_i}\boldsymbol{\lambda}_{j|i}^T(\boldsymbol{A}_{ij}\boldsymbol{x}_i-\boldsymbol{c}_{ij})\Big)\\
&-\sum_{(i,j)\in\mathcal{E}}\boldsymbol{\delta}_{ij}^T(\boldsymbol{c}_{ij}-\boldsymbol{A}_{ij}\boldsymbol{x}_i-\boldsymbol{A}_{ji}\boldsymbol{x}_j).
\end{aligned}\tag{16}$$

We summarize the result in a lemma below:

Lemma 1: If $(\boldsymbol{x}^\star, \boldsymbol{\delta}^\star)$ is a saddle point of $L_p(\boldsymbol{x}, \boldsymbol{\delta})$, then $(\boldsymbol{\delta}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{x}^\star)$ is a saddle point of $L_d(\boldsymbol{\delta}, \boldsymbol{\lambda}, \boldsymbol{x})$, where $\boldsymbol{\lambda}^\star_{i|j} = \boldsymbol{\delta}^\star_{ij}$ for every $[i,j] \in \vec{\mathcal{E}}$.

We note that $L_d(\boldsymbol{\delta}, \boldsymbol{\lambda}, \boldsymbol{x})$ might not be equivalent to $L'_d(\boldsymbol{\delta}, \boldsymbol{\lambda}, \boldsymbol{y})$. By inspection of the optimality conditions of (16), not every saddle point $(\boldsymbol{\delta}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{x}^\star)$ of L_d might lead to $\{\boldsymbol{\lambda}^\star_{i|j} = \boldsymbol{\lambda}^\star_{j|i}, (i,j) \in \mathcal{E}\}$ due to the generality of the matrices $\{\boldsymbol{A}_{ij}, [i,j] \in \vec{\mathcal{E}}\}$. In next section we will introduce quadratic penalty functions w.r.t. $\boldsymbol{\lambda}$ to implicitly enforce the equality constraints $\{\boldsymbol{\lambda}^\star_{i|j} = \boldsymbol{\lambda}^\star_{j|i}, (i,j) \in \mathcal{E}\}$.

To briefly summarize, one can alternatively solve the dual problem (13) instead of the primal problem. Further, by replacing $\boldsymbol{y}$ with an affine function of $\boldsymbol{x}$ in (15), the dual Lagrangian $L_d(\boldsymbol{\delta}, \boldsymbol{\lambda}, \boldsymbol{x})$ share two variables $\boldsymbol{x}$ and $\boldsymbol{\delta}$ with the primal Lagrangian $L_p(\boldsymbol{x}, \boldsymbol{\delta})$. We will show in next section that the special form of L_d in (16) plays a crucial role for constructing the augmented primal-dual Lagrangian.

III. Augmented Primal-Dual Lagrangian

In this section, we first build and investigate a primal-dual Lagrangian from L_p and L_d. We show that a saddle point of the primal-dual Lagrangian does not always lead to an optimal solution of the primal or the dual problem.

To address the above issue, we then construct an *augmented* primal-dual Lagrangian by introducing two additional penalty functions. We show that any saddle point of the augmented primal-dual Lagrangian leads to an optimal solution of the primal and the dual problem, respectively.

A. Primal-Dual Lagrangian

By inspection of (8) and (16), we see that in both L_p and L_d, the edge variables $\boldsymbol{\delta}_{ij}$ are related to the terms $\boldsymbol{c}_{ij} - \boldsymbol{A}_{ij}\boldsymbol{x}_i - \boldsymbol{A}_{ji}\boldsymbol{x}_j$. As a consequence, if we add the primal and dual Lagrangians, $\boldsymbol{\delta}_{ij}$ will cancel out and the resulting function contains node variables $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ only.

We hereby define the new function as the *primal-dual* Lagrangian below:

Definition 1: The primal-dual Lagrangian is defined as

$$\begin{aligned} L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda}) &= L_p(\boldsymbol{x}, \boldsymbol{\delta}) + L_d(\boldsymbol{\delta}, \boldsymbol{\lambda}, \boldsymbol{x}) \\ &= \sum_{i \in \mathcal{V}} \Big[f_i(\boldsymbol{x}_i) - \sum_{j \in \mathcal{N}_i} \boldsymbol{\lambda}^T_{j|i}(\boldsymbol{A}_{ij}\boldsymbol{x}_i - \boldsymbol{c}_{ij}) - f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i) \Big]. \end{aligned} \tag{17}$$

$L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda})$ is convex in $\boldsymbol{x}$ for fixed $\boldsymbol{\lambda}$ and concave in $\boldsymbol{\lambda}$ for fixed $\boldsymbol{x}$, suggesting that it is essentially a saddle-point problem (see [25], [26] for solving different saddle point problems). For each edge $(i,j) \in \mathcal{E}$, the node variables $\boldsymbol{\lambda}_{i|j}$ and $\boldsymbol{\lambda}_{j|i}$ substitute the role of the edge variable $\boldsymbol{\delta}_{ij}$. The removal of $\boldsymbol{\delta}_{ij}$ enables to design a distributed algorithm that only involves node-oriented optimization (see next section for PDMM).

Next we study the properties of saddle points of $L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda})$:

Lemma 2: If $\boldsymbol{x}^\star$ solves the primal problem (7), then there exists a $\boldsymbol{\lambda}^\star$ such that $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ is a saddle point of $L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda})$.

Proof: If $\boldsymbol{x}^\star$ solves the primal problem (7), then there exists a $\boldsymbol{\delta}^\star$ such that $(\boldsymbol{x}^\star, \boldsymbol{\delta}^\star)$ is a saddle point of $L_p(\boldsymbol{x}, \boldsymbol{\delta})$ and by Lemma 1, there exist $\boldsymbol{\lambda}^\star_{i|j} = \boldsymbol{\delta}^\star_{ij}$ for every $[i,j] \in \vec{\mathcal{E}}$ so that $(\boldsymbol{\delta}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{x}^\star)$ is a saddle point of $L_d(\boldsymbol{\delta}, \boldsymbol{\lambda}, \boldsymbol{x})$. Hence

$$\begin{aligned} L_{pd}(\boldsymbol{x}^\star, \boldsymbol{\lambda}) &= L_p(\boldsymbol{x}^\star, \boldsymbol{\delta}) + L_d(\boldsymbol{\delta}, \boldsymbol{\lambda}, \boldsymbol{x}^\star) \\ &\le L_p(\boldsymbol{x}^\star, \boldsymbol{\delta}^\star) + L_d(\boldsymbol{\delta}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{x}^\star) \\ &= L_{pd}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \\ &\le L_p(\boldsymbol{x}, \boldsymbol{\delta}^\star) + L_d(\boldsymbol{\delta}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{x}) \qquad = L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda}^\star). \end{aligned}$$

■

The fact that $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ is a saddle point of $L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda})$, however, is *not* sufficient for showing $\boldsymbol{x}^\star$ (or $\boldsymbol{\lambda}^\star$) being optimal for solving the primal problem (7) (for solving the dual problem (13)).

Example 1 ($\boldsymbol{x}^\star$ not optimal): Consider the following problem

$$\min_{x_1, x_2} f_1(x_1) + f_2(x_2) \quad \text{s.t.} \quad x_1 - x_2 = 0, \tag{18}$$

where

$$f_1(x_1) = f_2(-x_1) = \begin{cases} x_1 - 1 & x_1 \ge 1 \\ 0 & \text{otherwise} \end{cases}.$$

With this, the primal Lagrangian is given by $L_p(\boldsymbol{x}, \delta_{12}) = f_1(x_1) + f_2(x_2) + \delta_{12}(x_2 - x_1)$, so that the dual function is given by $-f_1^*(\delta_{12}) - f_2^*(-\delta_{12})$, where

$$f_1^*(\delta_{12}) = f_2^*(-\delta_{12}) = \begin{cases} \delta_{12} & 0 \le \delta_{12} \le 1 \\ +\infty & \text{otherwise} \end{cases}.$$

Hence, the optimal solution for the primal and dual problem is $x_1^\star = x_2^\star \in [-1, 1]$ and $\delta_{12}^\star = 0$, respectively. The primal-dual Lagrangian in this case is given by

$$\begin{aligned} L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda}) = f_1(x_1) + f_2(x_2) - f_1^*(\lambda_{1|2}) - f_2^*(-\lambda_{2|1}) \\ - x_1\lambda_{2|1} + x_2\lambda_{1|2}. \end{aligned} \tag{19}$$

One can show that every point $(x'_1, x'_2, \lambda'_{1|2}, \lambda'_{2|1}) \in \{(x_1, x_2, 0, 0) | -1 \le x_1, x_2 \le 1\}$ is a saddle point of $L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda})$, which does not necessarily lead to $x'_1 = x'_2$.

It is clear from Example 1 that finding a saddle point of L_{pd} does not necessarily solve the primal problem (7). Similarly, one can also build another example illustrating that a saddle point of L_{pd} does not necessarily solve the dual problem (13).

B. Augmented Primal-Dual Lagrangian

The problem that not every saddle point of $L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda})$ leads to an optimal point of the primal or dual problem can be solved by adding two quadratic penalty terms to $L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda})$ as

$$L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda}) = L_{pd}(\boldsymbol{x}, \boldsymbol{\lambda}) + h_{\mathcal{P}_p}(\boldsymbol{x}) - h_{\mathcal{P}_d}(\boldsymbol{\lambda}), \tag{20}$$

where $h_{\mathcal{P}_p}(\boldsymbol{x})$ and $h_{\mathcal{P}_d}(\boldsymbol{\lambda})$ are defined as

$$h_{\mathcal{P}_p}(\boldsymbol{x}) = \sum_{(i,j) \in \mathcal{E}} \frac{1}{2} \left\| \boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{A}_{ji}\boldsymbol{x}_j - \boldsymbol{c}_{ij} \right\|^2_{\boldsymbol{P}_{p,ij}} \tag{21}$$

$$h_{\mathcal{P}_d}(\boldsymbol{\lambda}) = \sum_{(i,j) \in \mathcal{E}} \frac{1}{2} \left\| \boldsymbol{\lambda}_{i|j} - \boldsymbol{\lambda}_{j|i} \right\|^2_{\boldsymbol{P}_{d,ij}}, \tag{22}$$

where $\mathcal{P} = \mathcal{P}_p \cup \mathcal{P}_d$ and

$$\mathcal{P}_p = \{\boldsymbol{P}_{p,ij}^T = \boldsymbol{P}_{p,ij} \succ 0 | (i,j) \in \mathcal{E}\}$$
$$\mathcal{P}_d = \{\boldsymbol{P}_{d,ij}^T = \boldsymbol{P}_{d,ij} \succ 0 | (i,j) \in \mathcal{E}\}.$$

The $2|\mathcal{E}|$ positive definite matrices in $\mathcal{P}$ remain to be specified.

Let $X = \{\boldsymbol{x} | \boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{A}_{ji}\boldsymbol{x}_j = \boldsymbol{c}_{ij}, \forall (i,j) \in \mathcal{E}\}$ and $\Lambda = \{\boldsymbol{\lambda} | \boldsymbol{\lambda}_{i|j} = \boldsymbol{\lambda}_{j|i}, \forall (i,j) \in \mathcal{E}\}$ denote the primal and dual feasible set, respectively. It is clear that $h_{\mathcal{P}_p}(\boldsymbol{x}) \geq 0$ (or $-h_{\mathcal{P}_d}(\boldsymbol{\lambda}) \leq 0$) with equality if and only if $\boldsymbol{x} \in X$ (or $\boldsymbol{\lambda} \in \Lambda$). The introduction of the two penalty functions essentially prevents non-feasible $\boldsymbol{x}$ and/or $\boldsymbol{\lambda}$ to correspond to saddle points of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. As a consequence, we have a saddle point theorem for $L_{\mathcal{P}}$ which states that $\boldsymbol{x}^\star$ solves the primal problem (7) if and only if there exits $\boldsymbol{\lambda}^\star$ such that $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ is a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. To prove this result, we need the following lemma.

Lemma 3: Let $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ and $(\boldsymbol{x}', \boldsymbol{\lambda}')$ be two saddle points of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. Then

$$L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}^\star) = L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}') = L_{\mathcal{P}}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) = L_{\mathcal{P}}(\boldsymbol{x}^\star, \boldsymbol{\lambda}'). \quad (23)$$

Further, $(\boldsymbol{x}', \boldsymbol{\lambda}^\star)$ and $(\boldsymbol{x}^\star, \boldsymbol{\lambda}')$ are two saddle points of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$ as well.

Proof: Since $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ and $(\boldsymbol{x}', \boldsymbol{\lambda}')$ are two saddle points of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$, we have

$$L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}^\star) \leq L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}') \leq L_{\mathcal{P}}(\boldsymbol{x}^\star, \boldsymbol{\lambda}')$$
$$L_{\mathcal{P}}(\boldsymbol{x}^\star, \boldsymbol{\lambda}') \leq L_{\mathcal{P}}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \leq L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}^\star).$$

Combining the above two inequality chains produces (23). In order to show that $(\boldsymbol{x}', \boldsymbol{\lambda}^\star)$ is a saddle point, we have $L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}) \leq L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}') = L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}^\star) = L_{\mathcal{P}}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \leq L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda}^\star)$. The proof for $(\boldsymbol{x}^\star, \boldsymbol{\lambda}')$ is similar. ■

We are ready to prove the saddle point theorem for $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$.

Theorem 1: If $\boldsymbol{x}^\star$ solves the primal problem (7), there exists $\boldsymbol{\lambda}^\star$ such that $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ is a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. Conversely, if $(\boldsymbol{x}', \boldsymbol{\lambda}')$ is a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$, then $\boldsymbol{x}'$ and $\boldsymbol{\lambda}'$ solves the primal and the dual problem, respectively. Or equivalently, the following optimality conditions hold

$$\sum_{j \in \mathcal{N}_i} \boldsymbol{A}_{ij}^T \boldsymbol{\lambda}'_{j|i} \in \partial_{\boldsymbol{x}_i} f_i(\boldsymbol{x}'_i) \qquad \forall i \in \mathcal{V} \quad (24)$$

$$\boldsymbol{A}_{ij}\boldsymbol{x}'_i + \boldsymbol{A}_{ji}\boldsymbol{x}'_j - \boldsymbol{c}_{ij} = \boldsymbol{0} \qquad \forall (i,j) \in \mathcal{E} \quad (25)$$

$$\boldsymbol{\lambda}'_{i|j} - \boldsymbol{\lambda}'_{j|i} = \boldsymbol{0} \qquad \forall (i,j) \in \mathcal{E}. \quad (26)$$

Proof: If $\boldsymbol{x}^\star$ solves the primal problem, then there exists a $\boldsymbol{\lambda}^\star$ such that $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ is a saddle point of L_{pd} by Lemma 2. Since $\boldsymbol{x}^\star \in X$ and $\boldsymbol{\lambda}^\star \in \Lambda$, we have $h_{\mathcal{P}_p}(\boldsymbol{x}^\star) - h_{\mathcal{P}_d}(\boldsymbol{\lambda}^\star) = 0$, $\partial_{\boldsymbol{x}} h_{\mathcal{P}_p}(\boldsymbol{x}^\star) = \boldsymbol{0}$ and $\partial_{\boldsymbol{\lambda}} h_{\mathcal{P}_d}(\boldsymbol{\lambda}^\star) = \boldsymbol{0}$, from which we conclude that $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ is a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$ as well.

Conversely, let $(\boldsymbol{x}', \boldsymbol{\lambda}')$ be a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. We first show that $\boldsymbol{x}'$ solves the primal problem. We have from Lemma 3 that $L_{\mathcal{P}}(\boldsymbol{x}', \boldsymbol{\lambda}^\star) = L_{\mathcal{P}}(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$, which can be simplified as

$$L_p(\boldsymbol{x}', \boldsymbol{\delta}^\star) + L_d(\boldsymbol{\delta}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{x}') + h_{\mathcal{P}_p}(\boldsymbol{x}')$$
$$= L_p(\boldsymbol{x}^\star, \boldsymbol{\delta}^\star) + L_d(\boldsymbol{\delta}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{x}^\star),$$

from which we conclude that $h_{\mathcal{P}_p}(\boldsymbol{x}') = L_p(\boldsymbol{x}^\star, \boldsymbol{\delta}^\star) - L_p(\boldsymbol{x}', \boldsymbol{\delta}^\star) \leq 0$ and thus $h_{\mathcal{P}_p}(\boldsymbol{x}') = 0$ so that $\boldsymbol{x}' \in X$. In addition, since $(\boldsymbol{x}', \boldsymbol{\lambda}^\star)$ is a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$ by Lemma 3, we have

$$\sum_{j \in \mathcal{N}_i} \boldsymbol{A}_{ij}^T \boldsymbol{\delta}_{ij}^\star = \sum_{j \in \mathcal{N}_i} \boldsymbol{A}_{ij}^T \boldsymbol{\lambda}_{j|i}^\star \in \partial_{\boldsymbol{x}_i} f_i(\boldsymbol{x}'_i), \forall i \in \mathcal{V},$$

and we conclude that $\boldsymbol{x}'$ solves the primal problem as required. Similarly, one can show that $\boldsymbol{\lambda}'$ solves the dual problem.

Based on the above analysis, we conclude that the optimality conditions for $(\boldsymbol{x}', \boldsymbol{\lambda}')$ being a saddle point of $L_{\mathcal{P}}$ are given by (24)–(26). The set of optimality conditions $\{\boldsymbol{c}_{ij} - \boldsymbol{A}_{ji}\boldsymbol{x}'_j \in \partial_{\boldsymbol{\lambda}_{i|j}} \left[f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}'_i)\right] | [i,j] \in \vec{\mathcal{E}}\}$ is redundant and can be derived from (24)–(26) (see (4)–(6) for the argument). ■

Theorem 1 states that instead of solving the primal problem (7) or the dual problem (13), one can alternatively search for a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. To briefly summarize, we consider solving the following min-max problem in the rest of the paper

$$(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) = \arg\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda}} L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda}). \quad (27)$$

We will explain in next section how to iteratively approach the saddle point $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ in a distributed manner.

IV. Primal-Dual Method of Multipliers

In this section, we present a new algorithm named *primal-dual method of multipliers* (PDMM) to iteratively approach a saddle point of $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. We propose both the synchronous and asynchronous PDMM for solving the problem.

A. Synchronous Updating Scheme

The synchronous updating scheme refers to the operation that at each iteration, all the variables over the graph update their estimates by using the most recent estimates from their neighbors from last iteration. Suppose $(\hat{\boldsymbol{x}}^k, \hat{\boldsymbol{\lambda}}^k)$ is the estimate obtained from the $k-1$th iteration, where $k \geq 1$. We compute the new estimate $(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1})$ at iteration k as

$$\left(\hat{\boldsymbol{x}}_i^{k+1}, \hat{\boldsymbol{\lambda}}_i^{k+1}\right) = \arg\min_{\boldsymbol{x}_i} \max_{\boldsymbol{\lambda}_i} L_{\mathcal{P}}\Big(\left[\ldots, \hat{\boldsymbol{x}}_{i-1}^{k,T}, \boldsymbol{x}_i^T, \hat{\boldsymbol{x}}_{i+1}^{k,T}, \ldots\right]^T,$$
$$\left[\ldots, \hat{\boldsymbol{\lambda}}_{i-1}^{k,T}, \boldsymbol{\lambda}_i^T, \hat{\boldsymbol{\lambda}}_{i+1}^{k,T}, \ldots\right]^T\Big) \quad i \in \mathcal{V}. \quad (28)$$

By inserting the expression (20) for $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$ into (28), the updating expression can be further simplified as

$$\hat{\boldsymbol{x}}_i^{k+1} = \arg\min_{\boldsymbol{x}_i} \Bigg[\sum_{j \in \mathcal{N}_i} \frac{1}{2} \left\| \boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{c}_{ij} \right\|_{\boldsymbol{P}_{p,ij}}^2$$
$$- \boldsymbol{x}_i^T \left(\sum_{j \in \mathcal{N}_i} \boldsymbol{A}_{ij}^T \hat{\boldsymbol{\lambda}}_{j|i}^k\right) + f_i(\boldsymbol{x}_i)\Bigg] \quad i \in \mathcal{V} \quad (29)$$

$$\hat{\boldsymbol{\lambda}}_i^{k+1} = \arg\min_{\boldsymbol{\lambda}_i} \Bigg[\sum_{j \in \mathcal{N}_i} \Bigg(\frac{1}{2} \left\| \boldsymbol{\lambda}_{i|j} - \hat{\boldsymbol{\lambda}}_{j|i}^k \right\|_{\boldsymbol{P}_{d,ij}}^2 + \boldsymbol{\lambda}_{i|j}^T \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k$$
$$- \boldsymbol{\lambda}_{i|j}^T \boldsymbol{c}_{ij}\Bigg) + f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i)\Bigg] \quad i \in \mathcal{V}. \quad (30)$$

Equation (29)–(30) suggest that at iteration k, every node i performs parameter-updating independently once the estimates $\{\hat{\boldsymbol{x}}_j^k, \hat{\boldsymbol{\lambda}}_{j|i}^k | j \in \mathcal{N}_i\}$ of its neighboring variables are available. In addition, the computation of $\hat{\boldsymbol{x}}_i^{k+1}$ and $\hat{\boldsymbol{\lambda}}_i^{k+1}$ can be carried out in parallel since $\boldsymbol{x}_i$ and $\boldsymbol{\lambda}_i$ are not directly related in $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$. We refer to (29)–(30) as *node-oriented* computation.

In order to run PDMM over the graph, each iteration should consist of two steps. Firstly, every node i computes $(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{\lambda}}_i)$ by following (29)–(30), accounting for *information-fusion*. Secondly, every node i sends $(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{\lambda}}_{i|j})$ to its neighboring node j for all neighbors, accounting for *information-spread*. We take $\hat{\boldsymbol{x}}_i$ as the common message to all neighbors of node i and $\hat{\boldsymbol{\lambda}}_{i|j}$ as a node-specific message only to neighbor j. In some applications, it may be preferable to exploit broadcast transmission rather than point-to-point transmission in order to save energy. We will explain in Section IV-C that the transmission of $\hat{\boldsymbol{\lambda}}_{i|j}$, $j \in \mathcal{N}_i$, can be replaced by broadcast transmission of an intermediate quantity.

Finally, we consider terminating the iterates (29)–(30). One can check if the estimate $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}})$ becomes stable over consecutive iterates (see Corollary 1 for theoretical support).

B. Asynchronous Updating Scheme

The asynchronous updating scheme refers to the operation that at each iteration, only the variables associated with one node in the graph update their estimates while all other variables keep their estimates fixed. Suppose node i is selected at iteration k. We then compute $(\hat{\boldsymbol{x}}_i^{k+1}, \hat{\boldsymbol{\lambda}}_i^{k+1})$ by optimizing $L_{\mathcal{P}}$ based on the most recent estimates $\{\hat{\boldsymbol{x}}_j^k, \hat{\boldsymbol{\lambda}}_{j|i}^k | j \in \mathcal{N}_i\}$ from its neighboring nodes. At the same time, the estimates $(\hat{\boldsymbol{x}}_j^k, \hat{\boldsymbol{\lambda}}_j^k)$, $j \neq i$, remain the same. By following the above computational instruction, $(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1})$ can be obtained as

$$\left(\hat{\boldsymbol{x}}_i^{k+1}, \hat{\boldsymbol{\lambda}}_i^{k+1}\right) = \arg\min_{\boldsymbol{x}_i} \max_{\boldsymbol{\lambda}_i} L_{\mathcal{P}}\Big(\left[\ldots, \hat{\boldsymbol{x}}_{i-1}^{k,T}, \boldsymbol{x}_i^T, \hat{\boldsymbol{x}}_{i+1}^{k,T}, \ldots\right]^T, \left[\ldots, \hat{\boldsymbol{\lambda}}_{i-1}^{k,T}, \boldsymbol{\lambda}_i^T, \hat{\boldsymbol{\lambda}}_{i+1}^{k,T}, \ldots\right]^T\Big) \quad (31)$$

$$(\hat{\boldsymbol{x}}_j^{k+1}, \hat{\boldsymbol{\lambda}}_j^{k+1}) = (\hat{\boldsymbol{x}}_j^k, \hat{\boldsymbol{\lambda}}_j^k) \qquad j \in \mathcal{V}, j \neq i. \quad (32)$$

Similarly to (29)–(30), $\hat{\boldsymbol{x}}_i^{k+1}$ and $\hat{\boldsymbol{\lambda}}_i^{k+1}$ can also be computed separately in (31). Once the update at node i is complete, the node sends the common message $\hat{\boldsymbol{x}}_i^{k+1}$ and node-specific messages $\{\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}, j \in \mathcal{N}_i\}$ to its neighbors. We will explain in next subsection how to exploit broadcast transmission to replace point-to-point transmission.

In practice, the nodes in the graph can either be randomly activated or follow a predefined order for asynchronous parameter-updating. One scheme for realizing random node-activation is that after a node finishes parameter-updating, it randomly activates one of its neighbors for next iteration. Another scheme is to introduce a clock at each node which ticks at the times of a (random) Poisson process (see [5] for detailed information). Each node is activated only when its clock ticks. As for node-activation in a predefined order, cyclic updating scheme is most straightforward. Once node i finishes parameter-updating, it informs node $i+1$ for next iteration. For the case that node i and $i+1$ are not neighbors, the path from node i to $i+1$ can be pre-stored at node i to facilitate the process. In Section V-D, we provide convergence analysis only for the cyclic updating scheme. We leave the analysis for other asynchronous schemes for future investigation.

Remark 1: To briefly summarize, synchronous PDMM scheme allows faster information-spread over the graph through parallel parameter-updating while asynchronous PDMM scheme requires less effort from node-coordination in the graph. In practice, the scheme-selection should depend on the graph (e.g., wireless sensor networks) properties such as the feasibility of parallel computation, the complexity of node-coordination and the life time of nodes.

C. Simplifying Node-Based Computations and Transmissions

It is clear that for both the synchronous and asynchronous schemes, each activated node i has to perform two minimizations: one for $\hat{\boldsymbol{x}}_i$ and the other one for $\hat{\boldsymbol{\lambda}}_i$. In this subsection, we show that the computations for the two minimizations can be simplified. We will also study how the point-to-point transmission can be replaced with broadcast transmission. To do so, we will consider two scenarios:

1) Avoiding Conjugate Functions: In the first scenario, we consider using $f_i(\cdot)$ instead of $f_i^*(\cdot)$ to update $\hat{\boldsymbol{\lambda}}_i$. Our goal is to simplify computations by avoiding the derivation of $f_i^*(\cdot)$.

By using the definition of f_i^* in (4), the computation (30) for $\hat{\boldsymbol{\lambda}}_i^{k+1}$ (which also holds for asynchronous PDMM) can be rewritten as

$$\hat{\boldsymbol{\lambda}}_i^{k+1} = \arg\min_{\boldsymbol{\lambda}_i}\left[\sum_{j\in\mathcal{N}_i}\left(\frac{1}{2}\left\|\boldsymbol{\lambda}_{i|j} - \hat{\boldsymbol{\lambda}}_{j|i}^k\right\|_{\boldsymbol{P}_{d,ij}}^2 + \boldsymbol{\lambda}_{i|j}^T \boldsymbol{A}_{ji} \hat{\boldsymbol{x}}_j^k - \boldsymbol{\lambda}_{i|j}^T \boldsymbol{c}_{ij}\right) + \max_{\boldsymbol{w}_i}\left(\boldsymbol{w}_i^T \boldsymbol{A}_i^T \boldsymbol{\lambda}_i - f_i(\boldsymbol{w}_i)\right)\right]. \quad (33)$$

We denote the optimal solution for $\boldsymbol{w}_i$ in (33) as $\boldsymbol{w}_i^{k+1}$. The optimality conditions for $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}$, $j \in \mathcal{N}_i$, and $\boldsymbol{w}_i^{k+1}$ can then be derived from (33) as

$$\boldsymbol{0} \in \boldsymbol{A}_i^T \hat{\boldsymbol{\lambda}}_i^{k+1} - \partial_{\boldsymbol{w}_i} f_i(\boldsymbol{w}_i^{k+1}) \quad (34)$$

$$\boldsymbol{c}_{ij} = \boldsymbol{P}_{d,ij}(\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \hat{\boldsymbol{\lambda}}_{j|i}^k) + \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k + \boldsymbol{A}_{ij}\boldsymbol{w}_i^{k+1} \;\; j \in \mathcal{N}_i, \quad (35)$$

where (14) is used in deriving (35). Since $\boldsymbol{P}_{d,ij}$ is a nonsingular matrix, (35) defines a mapping from $\boldsymbol{w}_i^{k+1}$ to $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}$:

$$\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} = \hat{\boldsymbol{\lambda}}_{j|i}^k + \boldsymbol{P}_{d,ij}^{-1}(\boldsymbol{c}_{ij} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{A}_{ij}\boldsymbol{w}_i^{k+1}), j \in \mathcal{N}_i, \quad (36)$$

With this mapping, (34) can then be reformulated as

$$\sum_{j\in\mathcal{N}_i} \boldsymbol{A}_{ij}^T \left(\hat{\boldsymbol{\lambda}}_{j|i}^k + \boldsymbol{P}_{d,ij}^{-1}(\boldsymbol{c}_{ij} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{A}_{ij}\boldsymbol{w}_i^{k+1})\right) \in \partial_{\boldsymbol{w}_i} f_i(\boldsymbol{w}_i^{k+1}). \quad (37)$$

By inspection of (37), it can be shown that (37) is in fact an optimality condition for the following optimization problem

$$\boldsymbol{w}_i^{k+1} = \arg\min_{\boldsymbol{w}_i} \Big[f_i(\boldsymbol{w}_i) + \frac{1}{2}\|\boldsymbol{c}_{ij} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{A}_{ij}\boldsymbol{w}_i\|^2_{\boldsymbol{P}_{d,ij}^{-1}} - \boldsymbol{w}_i^T \sum_{j\in\mathcal{N}_i} \boldsymbol{A}_{ij}^T \hat{\boldsymbol{\lambda}}_{j|i}^k \Big]. \quad (38)$$

The above analysis suggests that $\hat{\boldsymbol{\lambda}}_i^{k+1}$ can be alternatively computed through an intermediate quantity $\boldsymbol{w}_i^{k+1}$. We summarize the result in a proposition below.

Proposition 1: Considering a node $i \in \mathcal{V}$ at iteration k, the new estimate $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}$ for each $j \in \mathcal{N}_i$ can be obtained by following (36), where $\boldsymbol{w}_i^{k+1}$ is computed by (38).

Proposition 1 suggests that the estimate $\hat{\boldsymbol{\lambda}}_i^{k+1}$ can be easily computed from $\boldsymbol{w}_i^{k+1}$. We argue in the following that the point-to-point transmission of $\{\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}, j \in \mathcal{N}_i\}$ can be replaced with broadcast transmission of $\boldsymbol{w}_i^{k+1}$.

We see from (36) that the computation of the node-specific message $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}$ (from node i to node j) only consists of the quantities $\boldsymbol{w}_i^{k+1}$, $\hat{\boldsymbol{\lambda}}_{j|i}^k$ and $\hat{\boldsymbol{x}}_j^k$. Since $\hat{\boldsymbol{\lambda}}_{j|i}^k$ and $\hat{\boldsymbol{x}}_j^k$ are available at node j, the message $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}$ can therefore be computed at node j once the common message $\boldsymbol{w}_i^{k+1}$ is received. In other words, it is sufficient for node i to broadcast both $\hat{\boldsymbol{x}}_i^{k+1}$ and $\boldsymbol{w}_i^{k+1}$ to all its neighbors. Every node-specific message $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}$, $j \in \mathcal{N}_i$, can then be computed at node j alone.

Finally, in order for the broadcast transmission to work, we assume there is no transmission failure between neighboring nodes. The assumption ensures that there is no estimate inconsistency between neighboring nodes, making the broadcast transmission reliable.

2) Reducing Two Minimizations to One: In the second scenario, we study under what conditions the two minimizations (29)–(30) (which also hold for asynchronous PDMM) reduce to one minimization.

Proposition 2: Considering a node $i \in \mathcal{V}$ at iteration k, if the matrix $\boldsymbol{P}_{d,ij}$ for every neighbor $j \in \mathcal{N}_i$ is chosen to be $\boldsymbol{P}_{d,ij} = \boldsymbol{P}_{p,ij}^{-1}$, then there is $\hat{\boldsymbol{x}}_i^{k+1} = \boldsymbol{w}_i^{k+1}$. As a result,

$$\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} = \hat{\boldsymbol{\lambda}}_{j|i}^k + \boldsymbol{P}_{p,ij}(\boldsymbol{c}_{ij} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{A}_{ij}\hat{\boldsymbol{x}}_i^{k+1}) \; j \in \mathcal{N}_i. \quad (39)$$

Proof: The proof is trivial. By inspection of (29) and (38) under $\boldsymbol{P}_{d,ij} = \boldsymbol{P}_{p,ij}^{-1}$, $j \in \mathcal{N}_i$, we obtain $\hat{\boldsymbol{x}}_i^{k+1} = \boldsymbol{w}_i^{k+1}$. ■

Similarly to the first scenario, broadcast transmission is also applicable for the second scenario. Since $\hat{\boldsymbol{x}}_i^{k+1} = \boldsymbol{w}_i^{k+1}$, node i only has to broadcast the estimate $\hat{\boldsymbol{x}}_i^{k+1}$ to all its neighbors. Each message $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1}$ from node i to node j can then be computed at node j directly by applying (39). See Table I for the procedure of synchronous PDMM.

TABLE I
SYNCHRONOUS PDMM WHERE FOR EACH $i \in \mathcal{V}$, $\boldsymbol{P}_{d,ij} = \boldsymbol{P}_{p,ij}^{-1}$

```
Initialization: $\{\boldsymbol{x}_i\}$ and $\{\boldsymbol{\lambda}_{i|j}\}$
Repeat
    for all $i \in \mathcal{V}$ do
        $\hat{\boldsymbol{x}}_i^{k+1} = \arg\min_{\boldsymbol{x}_i}\Big[f_i(\boldsymbol{x}_i) - \boldsymbol{x}_i^T(\sum_{j\in\mathcal{N}_i}\boldsymbol{A}_{ij}^T\hat{\boldsymbol{\lambda}}_{j|i}^k)$
                $+ \sum_{j\in\mathcal{N}_i}\frac{1}{2}\|\boldsymbol{A}_{ij}\boldsymbol{x}_i + \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{c}_{ij}\|^2_{\boldsymbol{P}_{p,ij}}\Big]$
    end for
    for all $i \in \mathcal{V}$ and $j \in \mathcal{N}_i$ do
        $\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} = \hat{\boldsymbol{\lambda}}_{j|i}^k + \boldsymbol{P}_{p,ij}(\boldsymbol{c}_{ij} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{A}_{ij}\hat{\boldsymbol{x}}_i^{k+1})$
    end for
    $k \leftarrow k + 1$
Until some stopping criterion is met
```

V. CONVERGENCE ANALYSIS

In this section, we analyze the convergence rates of PDMM for both the synchronous and asynchronous schemes. Inspired by the convergence analysis of ADMM [27], [28], we construct a special inequality (presented in V-B) for $L_{\mathcal{P}}(\boldsymbol{x}, \boldsymbol{\lambda})$ and then exploit it to analyze both synchronous PDMM (presented in V-C) and asynchronous PDMM (presented in V-D).

Before constructing the inequality, we first study how to properly choose the matrices in the set $\mathcal{P}$ (presented in V-A) in order to enable convergence analysis.

A. Parameter Setting

In order to analyze the algorithm convergence later on, we first have to select the matrix set $\mathcal{P}$ properly. We impose a condition on each pair of matrices ($\boldsymbol{P}_{p,ij} \succ \boldsymbol{0}, \boldsymbol{P}_{d,ij} \succ \boldsymbol{0}$), $(i,j) \in \mathcal{E}$, in $L_{\mathcal{P}}$:

Condition 1: In the function $L_{\mathcal{P}}$, each matrix $\boldsymbol{P}_{d,ij}$ can be represented in terms of $\boldsymbol{P}_{p,ij}$ as

$$\boldsymbol{P}_{d,ij} = \boldsymbol{P}_{p,ij}^{-1} + \Delta\boldsymbol{P}_{d,ij} \quad \forall (i,j) \in \mathcal{E}, \quad (40)$$

where $\Delta\boldsymbol{P}_{d,ij} \succeq \boldsymbol{0}$.

Equation (40) implies that $\boldsymbol{P}_{p,ij}$ and $\boldsymbol{P}_{d,ij}$ can not be chosen arbitrarily for our convergence analysis. If $\boldsymbol{P}_{p,ij}$ is small, then $\boldsymbol{P}_{d,ij}$ has to be chosen big enough to make (40) hold, and vice versa. One special setup for $(\boldsymbol{P}_{p,ij}, \boldsymbol{P}_{d,ij})$ is to let $\boldsymbol{P}_{d,ij} = \boldsymbol{P}_{p,ij}^{-1}$, or equivalently, $\Delta\boldsymbol{P}_{d,ij} = \boldsymbol{0}$. This leads to the application of Proposition 2, which reduces two minimizations to one minimization for each activated node.

One simple setup in Condition 1 is to l l the matrices in $\mathcal{P}$ take scalar form. That is setting $(\boldsymbol{P}_{p,ij}, \boldsymbol{P}_{d,ij})$, $(i,j) \in \mathcal{E}$, to be identity matrices multiplied by positive parameters:

$$(\boldsymbol{P}_{p,ij}, \boldsymbol{P}_{d,ij}) = (\gamma_{p,ij}\boldsymbol{I}_{n_{ij}}, \gamma_{d,ij}\boldsymbol{I}_{n_{ij}}) \quad (41)$$

where $\gamma_{p,ij} > 0$, $\gamma_{d,ij} > 0$ and $\gamma_{d,ij}\gamma_{p,ij} \geq 1$. It is worth noting that matrix form of $(\boldsymbol{P}_{p,ij}, \boldsymbol{P}_{d,ij})$ might lead to faster convergence for some optimization problems.

B. Constructing an Inequality

Before introducing the inequality, we first define a new function which involves $\{f_i, i \in \mathcal{V}\}$ and their conjugates:

$$p(\boldsymbol{x}, \boldsymbol{\lambda}) = \sum_{i\in\mathcal{V}} \Big[f_i(\boldsymbol{x}_i) + f_i^*(\boldsymbol{A}_i^T\boldsymbol{\lambda}_i) - \frac{1}{2}\sum_{j\in\mathcal{N}_i} \boldsymbol{c}_{ij}^T \boldsymbol{\lambda}_{i|j} \Big]. \quad (42)$$

By studying (7) and (13) at a saddle point $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ of $L_\mathcal{P}$, one can show that $p(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) = 0$.

With $p(\boldsymbol{x}, \boldsymbol{\lambda})$, the inequality for $L_\mathcal{P}$ can be described as:

Lemma 4: Let $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ be a saddle point of $L_\mathcal{P}$. Then for any $(\boldsymbol{x}, \boldsymbol{\lambda})$, there is

$$0 \leq \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{N}_i} \Big[(\boldsymbol{\lambda}_{i|j} - \boldsymbol{\lambda}_{i|j}^\star)^T \Big(\boldsymbol{A}_{ji}\boldsymbol{x}_j - \frac{\boldsymbol{c}_{ij}}{2} \Big) \\ - (\boldsymbol{x}_i - \boldsymbol{x}_i^\star)^T \boldsymbol{A}_{ij}^T \boldsymbol{\lambda}_{j|i} \Big] + p(\boldsymbol{x}, \boldsymbol{\lambda}), \tag{43}$$

where equality holds if and only if $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfies

$$\boldsymbol{0} \in \partial_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i^\star) - \sum_{j\in\mathcal{N}_i} \boldsymbol{A}_{ij}^T \boldsymbol{\lambda}_{j|i} \qquad \forall i \in \mathcal{V} \tag{44}$$

$$\boldsymbol{0} \in \partial_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i) - \sum_{j\in\mathcal{N}_i} \boldsymbol{A}_{ij}^T \boldsymbol{\lambda}_{j|i}^\star \qquad \forall i \in \mathcal{V}. \tag{45}$$

Proof: Given a saddle point $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ of $L_\mathcal{P}$, the right hand side of the inequality (43) can be reformulated as

$$\sum_{i\in\mathcal{V}} \Bigg[\sum_{j\in\mathcal{N}_i} \Big(-\boldsymbol{\lambda}_{i|j}^{\star,T} \Big(\boldsymbol{A}_{ji}\boldsymbol{x}_j - \frac{\boldsymbol{c}_{ij}}{2} \Big) + \boldsymbol{x}_i^{\star,T} \boldsymbol{A}_{ij}^T \boldsymbol{\lambda}_{j|i} \\ - \boldsymbol{\lambda}_{i|j}^T \boldsymbol{c}_{ij} \Big) + f_i(\boldsymbol{x}_i) + f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i) \Bigg] \\ = \sum_{i\in\mathcal{V}} \Bigg[\sum_{j\in\mathcal{N}_i} \Big(-\boldsymbol{\lambda}_{j|i}^{\star,T} \boldsymbol{A}_{ij}\boldsymbol{x}_i + (\boldsymbol{A}_{ji}\boldsymbol{x}_j^\star - \boldsymbol{c}_{ij})^T \boldsymbol{\lambda}_{i|j} \Big) \\ + f_i(\boldsymbol{x}_i) + f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i) + \frac{1}{2} \sum_{j\in\mathcal{N}_i} \boldsymbol{c}_{ij}^T \boldsymbol{\lambda}_{i|j}^\star \Bigg] \\ = \sum_{i\in\mathcal{V}} \Bigg[\sum_{j\in\mathcal{N}_i} \Big(-\boldsymbol{\lambda}_{i|j}^{\star,T} \boldsymbol{A}_{ij}\boldsymbol{x}_i - \boldsymbol{x}_i^{\star,T} \boldsymbol{A}_{ij}^T \boldsymbol{\lambda}_{i|j} \Big) + f_i(\boldsymbol{x}_i) \\ + f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i) + \frac{1}{2} \sum_{j\in\mathcal{N}_i} \boldsymbol{c}_{ij}^T \boldsymbol{\lambda}_{i|j}^\star \Bigg], \tag{46}$$

where the last equality is obtained by using $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \in (X, \Lambda)$. Using Fenchel's inequalities (12), we conclude that for any $i \in \mathcal{V}$, the following two inequalities hold

$$f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i) - \boldsymbol{x}_i^{\star,T}(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i) \geq -f_i(\boldsymbol{x}_i^\star) \tag{47}$$

$$f_i(\boldsymbol{x}_i) - \boldsymbol{x}_i^T (\boldsymbol{A}_i^T \boldsymbol{\lambda}_i)^\star \geq -f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i^\star). \tag{48}$$

Finally, combining (46)–(48) and the fact that $p(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) = 0$ produces the inequality (43). The equality holds if and only if (47)–(48) hold, of which the optimality conditions are given by (44)–(45) (see (4)–(6) for the argument). ■

Lemma 4 shows that the quantity on the right hand side of (43) is always lower-bounded by zero. In the next two subsections, we will construct proper upper bounds for the quantity by replacing $(\boldsymbol{x}, \boldsymbol{\lambda})$ with real estimate of PDMM. The algorithmic convergence will be established by showing that the upper bounds approach to zero when iteration increases.

The conditions (44)–(45) in Lemma 4 are not sufficient for showing that $(\boldsymbol{x}, \boldsymbol{\lambda})$ is a saddle point of $L_\mathcal{P}$. The primal and dual feasibilities $\boldsymbol{x} \in X$ and $\boldsymbol{\lambda} \in \Lambda$ are also required to complete the argument, as shown in Lemma 5, 6 and 7 below. Lemma 5 and 6 are preliminary to show that $(\boldsymbol{x}, \boldsymbol{\lambda})$ is a saddle point of $L_\mathcal{P}$ as presented in Lemma 7. These three lemmas will be used in the next two subsections for convergence analysis.

Lemma 5: Let $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ be a saddle point of $L_\mathcal{P}$. Given $\boldsymbol{x} = \boldsymbol{x}'$ which satisfies (45) and $\boldsymbol{x}' \in X$, then $(\boldsymbol{x}', \boldsymbol{\lambda}^\star)$ is a saddle point of $L_\mathcal{P}$.

Proof: By using (45) and the fact that $\boldsymbol{x}' \in X$ and $\boldsymbol{\lambda}^\star \in \Lambda$, it is immediate from (24)–(26) that $(\boldsymbol{x}', \boldsymbol{\lambda}^\star)$ is a saddle point of $L_\mathcal{P}$. ■

Lemma 6: Let $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ be a saddle point of $L_\mathcal{P}$. Given $\boldsymbol{\lambda} = \boldsymbol{\lambda}'$ which satisfies (44) and $\boldsymbol{\lambda}' \in \Lambda$, then $(\boldsymbol{x}^\star, \boldsymbol{\lambda}')$ is a saddle point of $L_\mathcal{P}$.

Proof: The proof is similar to that for Lemma 5. ■

Lemma 7: Let $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ be a saddle point of $L_\mathcal{P}$. Given $(\boldsymbol{x}, \boldsymbol{\lambda}) = (\boldsymbol{x}', \boldsymbol{\lambda}')$ which satisfy (44)–(45) and $(\boldsymbol{x}', \boldsymbol{\lambda}') \in (X, \Lambda)$, then $(\boldsymbol{x}', \boldsymbol{\lambda}')$ is a saddle point of $L_\mathcal{P}$.

Proof: It is known from Lemma 5 and 6 that in addition to $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$, $(\boldsymbol{x}', \boldsymbol{\lambda}^\star)$ and $(\boldsymbol{x}^\star, \boldsymbol{\lambda}')$ are also the saddle points of $L_\mathcal{P}$. By using a similar argument as the one for Lemma 3, one can show that $(\boldsymbol{x}', \boldsymbol{\lambda}')$ is a saddle point of $L_\mathcal{P}$. ■

C. Synchronous PDMM

In this subsection, we show that the synchronous PDMM converges with the sub-linear rate $\mathcal{O}(K^{-1})$. In order to obtain the result, we need the following two lemmas.

Lemma 8: Let $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ be a saddle point of $L_\mathcal{P}$. The estimate $(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1})$ is obtained by performing (29)-(30) under Condition 1. Then there is

$$\sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{N}_i} \Big[(\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \boldsymbol{\lambda}_{i|j}^\star)^T \Big(\boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{k+1} - \frac{\boldsymbol{c}_{ij}}{2} \Big) - (\hat{\boldsymbol{x}}_i^{k+1} - \boldsymbol{x}_i^\star)^T \\ \cdot \boldsymbol{A}_{ij}^T \hat{\boldsymbol{\lambda}}_{j|i}^{k+1} \Big] + p(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1}) \leq \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{N}_i} d_{i|j}^{k+1}, \tag{49}$$

where $d_{i|j}^{k+1}$ is given by

$$d_{i|j}^{k+1} = \frac{1}{2} \Big(\Big\| \boldsymbol{P}_{p,ij}^{\frac{1}{2}} \boldsymbol{A}_{ji} (\hat{\boldsymbol{x}}_j^k - \boldsymbol{x}_j^\star) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}} (\boldsymbol{\lambda}_{j|i}^\star - \hat{\boldsymbol{\lambda}}_{j|i}^k) \Big\|^2 \\ - \Big\| \boldsymbol{P}_{p,ij}^{\frac{1}{2}} \boldsymbol{A}_{ji} (\hat{\boldsymbol{x}}_j^{k+1} - \boldsymbol{x}_j^\star) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}} (\boldsymbol{\lambda}_{j|i}^\star - \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}) \Big\|^2 \\ - \Big\| \boldsymbol{P}_{p,ij}^{\frac{1}{2}} (\boldsymbol{A}_{ij}\hat{\boldsymbol{x}}_i^{k+1} + \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k - \boldsymbol{c}_{ij}) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}} (\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \hat{\boldsymbol{\lambda}}_{j|i}^k) \Big\|^2 \\ + \| \Delta\boldsymbol{P}_{d,ij}^{\frac{1}{2}} (\boldsymbol{\lambda}_{j|i}^\star - \hat{\boldsymbol{\lambda}}_{j|i}^k) \|^2 - \| \Delta\boldsymbol{P}_{d,ij}^{\frac{1}{2}} (\boldsymbol{\lambda}_{j|i}^\star - \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}) \|^2 \\ - \| \Delta\boldsymbol{P}_{d,ij}^{\frac{1}{2}} (\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \hat{\boldsymbol{\lambda}}_{j|i}^k) \|^2 \Big), \tag{50}$$

where $\boldsymbol{P}_{p,ij} = \boldsymbol{P}_{p,ij}^{\frac{1}{2}} \boldsymbol{P}_{p,ij}^{\frac{1}{2}}$ and $\Delta\boldsymbol{P}_{d,ij} = \Delta\boldsymbol{P}_{d,ij}^{\frac{1}{2}} \Delta\boldsymbol{P}_{d,ij}^{\frac{1}{2}}$.

Proof: See the proof in Appendix A. ■

Lemma 9: Every pair of estimates $(\hat{\boldsymbol{x}}_i^{k+1}, \hat{\boldsymbol{\lambda}}_{i|j}^{k+1})$, $i \in \mathcal{V}, j \in \mathcal{N}_i, k \geq 0$, in Lemma 8 is upper bounded by a constant M under a squared error criterion:

$$\Big\| \boldsymbol{P}_{p,ij}^{\frac{1}{2}} \boldsymbol{A}_{ji} (\hat{\boldsymbol{x}}_j^{k+1} - \boldsymbol{x}_j^\star) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}} (\boldsymbol{\lambda}_{j|i}^\star - \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}) \Big\|^2 \leq M. \tag{51}$$

Proof: One can first prove (51) for $k=0$ by performing algebra on (49)–(50). The inequality (51) for $k>0$ can then be proved recursively. ■

Upon obtaining the results in Lemma 8 and 9, we are ready to present the convergence rate of synchronous PDMM.

Theorem 2: Let $(\hat{\boldsymbol{x}}^k, \hat{\boldsymbol{\lambda}}^k)$, $k=1,\ldots,K$, be obtained by performing (29)–(30) under Condition 1. The average estimate $(\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K) = (\frac{1}{K}\sum_{k=1}^K \hat{\boldsymbol{x}}^k, \frac{1}{K}\sum_{k=1}^K \hat{\boldsymbol{\lambda}}^k)$ satisfies

$$0 \le \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[(\bar{\boldsymbol{\lambda}}_{i|j}^K - \boldsymbol{\lambda}_{i|j}^\star)^T\Big(\boldsymbol{A}_{ji}\bar{\boldsymbol{x}}_j^K - \frac{\boldsymbol{c}_{ij}}{2}\Big) - (\bar{\boldsymbol{x}}_i^K - \boldsymbol{x}_i^\star)^T \cdot \boldsymbol{A}_{ij}^T\bar{\boldsymbol{\lambda}}_{j|i}^K\Big] + p(\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K) \le \mathcal{O}\Big(\frac{1}{K}\Big) \tag{52}$$

$$\lim_{K\to\infty}\Big[\boldsymbol{P}_{p,ij}^{\frac{1}{2}}(\boldsymbol{A}_{ij}\bar{\boldsymbol{x}}_i^K + \boldsymbol{A}_{ji}\bar{\boldsymbol{x}}_j^K - \boldsymbol{c}_{ij}) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}}(\bar{\boldsymbol{\lambda}}_{i|j}^K - \bar{\boldsymbol{\lambda}}_{j|i}^K)\Big] = \boldsymbol{0} \qquad \forall[i,j]\in\vec{\mathcal{E}}. \tag{53}$$

Proof: Summing (49) over k and simplifying the expression yields

$$\sum_{k=0}^{K-1}\Bigg(\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[(\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \boldsymbol{\lambda}_{i|j}^\star)^T\Big(\boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{k+1} - \frac{\boldsymbol{c}_{ij}}{2}\Big) - (\hat{\boldsymbol{x}}_i^{k+1} - \boldsymbol{x}_i^\star)^T\boldsymbol{A}_{ij}^T\hat{\boldsymbol{\lambda}}_{j|i}^{k+1}\Big] + p(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1}) + \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}}\Big[\Big\|\boldsymbol{P}_{p,ij}^{\frac{1}{2}}(\boldsymbol{A}_{ij}\hat{\boldsymbol{x}}_i^{k+1} + \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{k} - \boldsymbol{c}_{ij}) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}}(\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \hat{\boldsymbol{\lambda}}_{j|i}^{k})\Big\|^2\Big]\Bigg)$$
$$\le \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\frac{1}{2}\Big(\Big\|\boldsymbol{P}_{p,ij}^{\frac{1}{2}}(\boldsymbol{A}_{ji}(\hat{\boldsymbol{x}}_j^0 - \boldsymbol{x}_j^\star) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}}(\boldsymbol{\lambda}_{i|j}^\star - \hat{\boldsymbol{\lambda}}_{j|i}^0)\Big\|^2 + \|\Delta\boldsymbol{P}_{d,ij}^{\frac{1}{2}}(\boldsymbol{\lambda}_{j|i}^\star - \hat{\boldsymbol{\lambda}}_{j|i}^0)\|^2\Big). \tag{54}$$

Finally, since the left hand side of (54) is a convex function of $(\boldsymbol{x}, \boldsymbol{\lambda})$, applying Jensen's inequality to (54) and using the inequality of Lemma 4 yields (52). Similarly, applying Jensen's inequality to (54) and using the upper-bound result of Lemma 9 yields the asymptotic result (53). ■

Finally, we use the results of Theorem 2 to show that as K goes to infinity, the average estimate $(\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K)$ converges to a saddle point of $L_{\mathcal{P}}$. ■

Theorem 3: The average estimate $(\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K)$ of Theorem 2 converges to a saddle point $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ of $L_{\mathcal{P}}$ as K increases.

Proof: The basic idea of the proof is to investigate if $(\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K)$ satisfies all the conditions of Lemma 7. By investigation of Lemma 4 and (52), it is clear that the average estimate $(\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K)$ asymptotically satisfies the conditions (44)–(45) by letting $(\boldsymbol{x}, \boldsymbol{\lambda}) = (\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K)$.

Next we show that as K increases, $\bar{\boldsymbol{x}}^K$ asymptotically converges to an element of the primal feasible set X and so does $\bar{\boldsymbol{\lambda}}^K$ to an element of the dual feasible set Λ. To do so, we reconsider (53) for each pair of directed edges $[i,j]$ and $[j,i]$, which can be expressed as

$$\lim_{K\to\infty}\Big[\boldsymbol{P}_{p,ij}^{\frac{1}{2}}(\boldsymbol{A}_{ij}\bar{\boldsymbol{x}}_i^K + \boldsymbol{A}_{ji}\bar{\boldsymbol{x}}_j^K - \boldsymbol{c}_{ij}) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}}(\bar{\boldsymbol{\lambda}}_{i|j}^K - \bar{\boldsymbol{\lambda}}_{j|i}^K)\Big] = \boldsymbol{0}$$
$$\lim_{K\to\infty}\Big[\boldsymbol{P}_{p,ij}^{\frac{1}{2}}(\boldsymbol{A}_{ij}\bar{\boldsymbol{x}}_i^K + \boldsymbol{A}_{ji}\bar{\boldsymbol{x}}_j^K - \boldsymbol{c}_{ij}) + \boldsymbol{P}_{p,ij}^{-\frac{1}{2}}(\bar{\boldsymbol{\lambda}}_{j|i}^K - \bar{\boldsymbol{\lambda}}_{i|j}^K)\Big] = \boldsymbol{0}.$$

Combining the above two expressions produces

$$\lim_{K\to\infty}\boldsymbol{A}_{ij}\bar{\boldsymbol{x}}_i^K + \boldsymbol{A}_{ji}\bar{\boldsymbol{x}}_j^K = \boldsymbol{c}_{ij} \qquad \forall(i,j)\in\mathcal{E}$$
$$\lim_{K\to\infty}\bar{\boldsymbol{\lambda}}_{j|i}^K = \bar{\boldsymbol{\lambda}}_{i|j}^K \qquad \forall(i,j)\in\mathcal{E}.$$

It is straightforward from Lemma 7 that $(\bar{\boldsymbol{x}}^K, \bar{\boldsymbol{\lambda}}^K)$ converges to a saddle point of $L_{\mathcal{P}}$ as K increases. ■

Further we have the following result from Theorem 3:

Corollary 1: If for certain $i\in\mathcal{V}$, the estimate $\hat{\boldsymbol{x}}_i^k$ in Theorem 2 converges to a fixed point $\boldsymbol{x}_i'$ ($\lim_{k\to\infty}\hat{\boldsymbol{x}}_i^k = \boldsymbol{x}_i'$), we have $\boldsymbol{x}_i' = \boldsymbol{x}_i^\star$ which is the ith component of the optimal solution $\boldsymbol{x}^\star$ in Theorem 3. Similarly, if the estimate $\hat{\boldsymbol{\lambda}}_{i|j}^k$ converges to a point $\boldsymbol{\lambda}_{i|j}'$, we have $\boldsymbol{\lambda}_{i|j}' = \boldsymbol{\lambda}_{i|j}^\star$.

D. Asynchronous PDMM

In this subsection, we characterize the convergence rate of asynchronous PDMM. In order to facilitate the analysis, we consider a predefined node-activation strategy (no randomness is involved). We suppose at each iteration k, the node $i = \mathrm{mod}(k,m)+1$ is activated for parameter-updating, where $m = |\mathcal{V}|$ and $\mathrm{mod}(\cdot,\cdot)$ stands for the modulus operation. Then naturally, after a segment of m consecutive iterations, all the nodes will be activated sequentially, one node at each iteration.

To be able to derive the convergence rate, we consider segments of iterations, i.e., $k\in\{lm, lm+1, \ldots (l+1)m-1\}$, where $l\ge 0$. Each segment l consists of m iterations. With the mapping $i=\mathrm{mod}(k,m)+1$, it is immediate that $k=ml$ activates node 1 and $k=(l+1)m-1$ activates node m. Based on the above analysis, we have the following result.

Lemma 10: Let k_1, k_2 be two iteration indices within a segment $\{lm, lm+1, \ldots, (l+1)m-1\}$. If $k_1<k_2$, then $i_1<i_2$, where the node-index $i_q = \mathrm{mod}(k_q,m)+1$, $q=1,2$.

Upon introducing Lemma 10, we are ready to perform convergence analysis.

Lemma 11: Let $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ be a saddle point of $L_{\mathcal{P}}$. A segment of estimates $\{(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1}) | k = lm, \ldots, (l+1)m-1\}$, is obtained by performing (31)–(32) under Condition 1. Then there is

$$\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[\Big(\hat{\boldsymbol{\lambda}}_{i|j}^{(l+1)m} - \boldsymbol{\lambda}_{i|j}^\star\Big)^T\Big(\boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{(l+1)m} - \frac{\boldsymbol{c}_{ij}}{2}\Big) - \Big(\hat{\boldsymbol{x}}_i^{(l+1)m} - \boldsymbol{x}_i^\star\Big)^T \cdot \boldsymbol{A}_{ij}^T\hat{\boldsymbol{\lambda}}_{j|i}^{(l+1)m}\Big] + p\Big(\hat{\boldsymbol{x}}^{(l+1)m}, \hat{\boldsymbol{\lambda}}^{(l+1)m}\Big) \le \sum_{(u,v)\in\mathcal{E}}^{u<v} d_{uv}^{l+1}, \tag{55}$$

where d_{uv}^{l+1} is given by

$$\begin{aligned}
d_{uv}^{l+1} = \frac{1}{2}\Big(& \|\boldsymbol{P}_{p,uv}^{\frac{1}{2}}\boldsymbol{A}_{vu}(\hat{\boldsymbol{x}}_v^{lm} - \boldsymbol{x}_v^{\star}) + \boldsymbol{P}_{p,uv}^{-\frac{1}{2}}(\boldsymbol{\lambda}_{v|u}^{\star} - \hat{\boldsymbol{\lambda}}_{v|u}^{lm})\|^2 \\
& - \|\boldsymbol{P}_{p,uv}^{\frac{1}{2}}\boldsymbol{A}_{vu}(\hat{\boldsymbol{x}}_v^{(l+1)m} - \boldsymbol{x}_v^{\star}) + \boldsymbol{P}_{p,uv}^{-\frac{1}{2}}(\boldsymbol{\lambda}_{v|u}^{\star} - \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m})\|^2 \\
& - \|\boldsymbol{P}_{p,uv}^{\frac{1}{2}}(\boldsymbol{A}_{uv}\hat{\boldsymbol{x}}_u^{(l+1)m} + \boldsymbol{A}_{vu}\hat{\boldsymbol{x}}_v^{(l+1)m} - \boldsymbol{c}_{uv}) \\
& \qquad - \boldsymbol{P}_{p,uv}^{-\frac{1}{2}}(\hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m})\|^2 \\
& - \|\boldsymbol{P}_{p,uv}^{\frac{1}{2}}(\boldsymbol{A}_{uv}\hat{\boldsymbol{x}}_u^{(l+1)m} + \boldsymbol{A}_{vu}\hat{\boldsymbol{x}}_v^{lm} - \boldsymbol{c}_{uv}) \\
& \qquad + \boldsymbol{P}_{p,uv}^{-\frac{1}{2}}(\hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{v|u}^{lm})\|^2 + \|\Delta\boldsymbol{P}_{d,uv}^{\frac{1}{2}}(\boldsymbol{\lambda}_{u|v}^{\star} - \hat{\boldsymbol{\lambda}}_{v|u}^{lm})\|^2 \\
& - \|\Delta\boldsymbol{P}_{d,uv}^{\frac{1}{2}}(\boldsymbol{\lambda}_{u|v}^{\star} - \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m})\|^2 - \|\Delta\boldsymbol{P}_{d,uv}^{\frac{1}{2}}(\hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{v|u}^{lm})\|^2 \\
& - \|\Delta\boldsymbol{P}_{d,uv}^{\frac{1}{2}}(\hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m})\|^2\Big) \qquad u < v. \qquad (56)
\end{aligned}$$

Proof: See the proof in Appendix B. Lemma 10 will be used in the proof to simplify mathematic derivations. ■

Remark 2: We note that Lemma 11 corresponds to Lemma 8 which is for synchronous PDMM. The right hand side of (55) consists of $|\mathcal{E}|$ quantities $\{d_{uv}^{l+1}\}$ (one for each edge $(u,v) \in \mathcal{E}$) as opposed to that of (49) which consists of $|\vec{\mathcal{E}}|$ quantities $\{d_{i|j}^{k+1}\}$ (one for each directed edge $[i,j] \in \vec{\mathcal{E}}$).

Lemma 12: Every pair of estimates $(\hat{\boldsymbol{x}}_v^{(l+1)m}, \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m})$, $(u,v) \in \mathcal{E}$, $u < v$, $l \geq 0$, in Lemma 11 is upper bounded by a constant M under a squared error criterion:

$$\|\boldsymbol{P}_{p,uv}^{\frac{1}{2}}\boldsymbol{A}_{vu}(\hat{\boldsymbol{x}}_v^{(l+1)m} - \boldsymbol{x}_v^{\star}) + \boldsymbol{P}_{p,uv}^{-\frac{1}{2}}(\boldsymbol{\lambda}_{v|u}^{\star} - \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m})\|^2 \leq M.$$

Theorem 4: Let the $K \geq 1$ segments of estimates $\{(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1}) | k = 0, \ldots, Km-1\}$ be obtained by performing (31)–(32) under Condition 1. The average estimates $(\check{\boldsymbol{x}}^K, \check{\boldsymbol{\lambda}}^K) = (\frac{1}{K}\sum_{l=1}^{K}\hat{\boldsymbol{x}}^{lm}, \frac{1}{K}\sum_{l=1}^{K}\hat{\boldsymbol{\lambda}}^{lm})$ satisfies

$$\begin{aligned}
0 \leq \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[& \left(\check{\boldsymbol{\lambda}}_{i|j}^K - \boldsymbol{\lambda}_{i|j}^{\star}\right)^T\left(\boldsymbol{A}_{ji}\check{\boldsymbol{x}}_j^K - \frac{\boldsymbol{c}_{ij}}{2}\right) - \left(\check{\boldsymbol{x}}_i^K - \boldsymbol{x}_i^{\star}\right)^T \\
& \cdot \boldsymbol{A}_{ij}^T\check{\boldsymbol{\lambda}}_{j|i}^K\Big] + p\left(\check{\boldsymbol{x}}^K, \check{\boldsymbol{\lambda}}^K\right) \leq \mathcal{O}\left(\frac{1}{K}\right) \qquad (57)
\end{aligned}$$

$$\begin{aligned}
0 \leq & \left\|\boldsymbol{P}_{p,uv}^{\frac{1}{2}}(\boldsymbol{A}_{uv}\check{\boldsymbol{x}}_u^K + \boldsymbol{A}_{vu}\check{\boldsymbol{x}}_v^K - \boldsymbol{c}_{uv})\right. \\
& \left.- \boldsymbol{P}_{p,uv}^{-\frac{1}{2}}(\check{\boldsymbol{\lambda}}_{u|v}^K - \check{\boldsymbol{\lambda}}_{v|u}^K)\right\|^2 \leq \mathcal{O}\left(\frac{1}{K}\right) \ \forall (u,v) \in \mathcal{E}, u < v \qquad (58)
\end{aligned}$$

$$\begin{aligned}
\lim_{K\to\infty} & \Big[\boldsymbol{P}_{p,uv}^{\frac{1}{2}}(\boldsymbol{A}_{uv}\check{\boldsymbol{x}}_u^K + \boldsymbol{A}_{vu}\check{\boldsymbol{x}}_v^K - \boldsymbol{c}_{uv}) \\
& + \boldsymbol{P}_{p,uv}^{-\frac{1}{2}}(\check{\boldsymbol{\lambda}}_{u|v}^K - \check{\boldsymbol{\lambda}}_{v|u}^K)\Big] = \boldsymbol{0} \ \forall (u,v) \in \mathcal{E}, u < v. \qquad (59)
\end{aligned}$$

Proof: The proof is similar to that for Theorem 2. ■

Similarly to synchrounous PDMM, by using the results of Theorem 4, we can conclude that:

Theorem 5: The average estimate $(\check{\boldsymbol{x}}^K, \check{\boldsymbol{\lambda}}^K)$ of Theorem 4 converges to a saddle point $(\boldsymbol{x}^{\star}, \boldsymbol{\lambda}^{\star})$ of $L_{\mathcal{P}}$ as K increases.

Corollary 2: If for certain $u \in \mathcal{V}$, the estimate $\hat{\boldsymbol{x}}_u^{lm}$ in Theorem 4 converges to a fixed point $\boldsymbol{x}_u'$ ($\lim_{l\to\infty}\hat{\boldsymbol{x}}_u^{lm} = \boldsymbol{x}_u'$), we have $\boldsymbol{x}_u' = \boldsymbol{x}_u^{\star}$ which is the uth component of the optimal solution $\boldsymbol{x}^{\star}$ in Theorem 5. Similarly, if the estimate $\hat{\boldsymbol{\lambda}}_{u|v}^{lm}$ converges to a point $\boldsymbol{\lambda}_{u|v}'$, we hvae $\boldsymbol{\lambda}_{u|v}' = \boldsymbol{\lambda}_{u|v}^{\star}$.

VI. Application to Distributed Averaging

In this section, we consider solving the problem of distributed averaging by using PDMM. Distributed averaging is one of the basic and important operations for advanced distributed signal processing [5], [15].

A. Problem Formulation

Suppose every node i in a graph $G = (\mathcal{V}, \mathcal{E})$ carries a scalar parameter, denoted as t_i. t_i may represent a measurement of the environment, such as temperature, humidity or darkness. The problem is to compute the average value $t_{ave} = \frac{1}{m}\sum_{i\in\mathcal{V}} t_i$ iteratively only through message-passing between neighboring nodes in the graph.

The above averaging problem can be formulated as a quadratic optimization over the graph as

$$\min_{\{x_i\}} \sum_{i\in\mathcal{V}} \frac{1}{2}(x_i - t_i)^2 \quad \text{s.t. } x_i - x_j = 0 \quad \forall (i,j) \in \mathcal{E}. \qquad (60)$$

The optimal solution equals to $x_1^{\star} = \ldots = x_m^{\star} = t_{ave}$, which is the same as the averaging value.

The quadratic problem (60) is inline with (7) by letting

$$f_i(x_i) = \frac{1}{2}(x_i - t_i)^2 \quad \forall i \in \mathcal{V} \qquad (61)$$

$$(\boldsymbol{A}_{ij}, \boldsymbol{A}_{ji}, \boldsymbol{c}_{ij}) = (1, -1, 0) \quad \forall (i,j) \in \mathcal{E}, i < j. \qquad (62)$$

In next subsection, we apply PDMM for distributed averaging.

B. Parameter Computations and Transmissions

Before deriving the updating expressions for PDMM, we first configure the set $\mathcal{P}$ in $L_{\mathcal{P}}$. For distributed averaging, all the matrices in $\mathcal{P}$ become scalars. For simplicity, we set the value of the primal scalars and the dual scalars as

$$\boldsymbol{P}_{p,ij} = \gamma_p \quad \forall (i,j) \in \mathcal{E} \qquad (63a)$$

$$\boldsymbol{P}_{d,ij} = \gamma_d \quad \forall (i,j) \in \mathcal{E}, \qquad (63b)$$

where the two parameters $\gamma_p > 0$ and $\gamma_d > 0$.

We start with the synchronous PDMM. By inserting (61)–(63) into (29), (36) and (38), the updating expression for $(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1})$ at iteration k can be derived as

$$\hat{x}_i^{k+1} = \frac{t_i + \sum_{j\in\mathcal{N}_i}(\gamma_p\hat{x}_j^k + \boldsymbol{A}_{ij}\hat{\lambda}_{j|i}^k)}{1 + |\mathcal{N}_i|\gamma_p} \quad \forall i \in \mathcal{V} \qquad (64)$$

$$\hat{\lambda}_{i|j}^{k+1} = \hat{\lambda}_{j|i}^k - \frac{1}{\gamma_d}\left(\boldsymbol{A}_{ji}\hat{x}_j^k + \boldsymbol{A}_{ij}w_i^{k+1}\right) \quad \forall [i,j] \in \vec{\mathcal{E}}, \qquad (65)$$

where

$$w_i^{k+1} = \frac{\sum_{j\in\mathcal{N}_i}(\hat{x}_j^k + \gamma_d\boldsymbol{A}_{ij}\hat{\lambda}_{j|i}^k) + \gamma_d t_i}{|\mathcal{N}_i| + \gamma_d} \quad \forall i \in \mathcal{V}. \qquad (66)$$

For the case that $\gamma_d = \gamma_p^{-1}$, it is immediate from (64) and (66) that $\hat{x}_i^{k+1} = w_i^{k+1}$, which coincides with Proposition 2.

The asynchronous PDMM only activates one node per iteration. Suppose node i is activated at iteration k. Node i then updates $\hat{x}_i$ and $\hat{\lambda}_{i|j}$, $j \in \mathcal{N}_i$, by following (64)–(65) while all other nodes remain silent. After computation, node i then sends $(\hat{x}_i, \hat{\lambda}_{i|j})$ to its neighboring node j for all neighbors.

As described in Section IV-C, if no transmission fails in the graph, the transmission of $\hat{\lambda}_{i|j}$, $j \in \mathcal{N}_i$, can be replaced by broadcast transmission of w_i as given by (66). Once w_i is received by a neighboring node j, $\hat{\lambda}_{i|j}$ can be easily computed by node j alone using w_i, $\hat{x}_j$ and $\hat{\lambda}_{j|i}$ (see (65)). If instead the transmission is not reliable, we have to return to point-to-point transmission.

C. Experimental Results

We conducted three experiments for PDMM applied to distributed averaging. In the first experiment, we evaluated how different parameter-settings w.r.t. (γ_p, γ_d) affect the convergence rates of PDMM. In the second experiment, we tested the non-perfect channels for PDMM, which lacks theoretical analysis at the moment. Finally, we evaluated the convergence rates of PDMM, ADMM and two gossip algorithms.

The tested graph in the three experiments was a 10×10 two-dimensional grid (corresponding to $m = 100$), implying that each node may have two, three or four neighbors. The mean squared error (MSE) $\frac{1}{m}\|\hat{\boldsymbol{x}} - t_{ave}\mathbf{1}\|_2^2$ was employed as performance measurement.

1) Performance for Different Parameter Settings: In this experiment, we evaluated the performance of PDMM by testing different parameter-settings for (γ_p, γ_d). Both synchronous and asynchronous updating schemes were investigated.

At each iteration, the synchronous PDMM activated all the nodes for parameter-updating. As for the asynchronous PDMM, the nodes were activated sequentially by following the mapping $i = \mathrm{mod}(k, m) + 1$, where the iteration $k \geq 0$ (See Section V-D). As a result, after every segment of $m = 100$ iterations, all the nodes were activated once. In the experiment, we counted the number of iterations for the synchronous PDMM and the number of segments (each segment consists of m iterations) for the asynchronous PDMM.

For each parameter-setting, we initialized $(\hat{x}_i^0, \hat{\boldsymbol{\lambda}}_i^0) = (t_i, \mathbf{0})$ for every $i \in \mathcal{V}$. The algorithm stops when the squared error is below 10^{-4}.

Fig. 2 displays the numbers of iterations (or segments) of PDMM under different parameter-settings. Each $\circ$ or $\square$ symbol represents a particular setting for (γ_p, γ_d). The settings denoted by $\square$ are for the case that $\gamma_p\gamma_d < 1$ while the ones by $\circ$ are for the case that $\gamma_p\gamma_d \geq 1$.

It is seen from the figure that large γ_p or γ_d can only make the algorithm converge slowly. The optimal parameter-setting that leads to the fastest convergence lies on the curve $\gamma_d\gamma_p = 1$ for both the synchronous and the asynchronous updating schemes. Further, it appears that the two optimal settings for the two updating schemes are in a neighborhood.

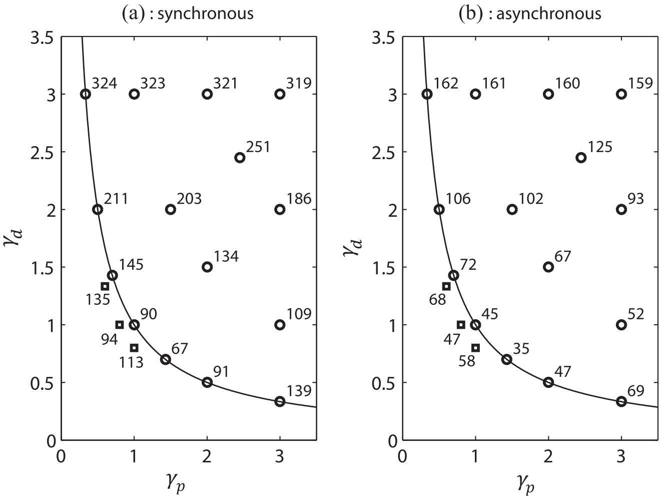


Fig. 2. Performance of PDMM for different parameter settings. Each value in subplot (a) represents the number of iterations required for the synchronous PDMM. On the other hand, each value in subplot (b) represents the number of segments of iterations for the asynchronous PDMM, where each segment consists of 100 iterations. The convex curve in each subplot corresponds to $\gamma_p\gamma_d = 1$.

Finally, we note that the settings denoted by $\square$ correspond to the situation that $\gamma_p\gamma_d < 1$. The experiment for those settings demonstrates that Condition 1 is only sufficient for algorithmic convergence. We also tested the setting $\gamma_p = \gamma_d = 0.5$. We found that the above setting led to divergence for both synchronous and synchronous schemes. This phenomenon suggests that γ_p and γ_d cannot be chosen arbitrarily in practice.

2) Performance With Transmission Failure: In this experiment, we studied how transmission failure affects the performance of PDMM given the fact that no convergence guaranty is derived at the moment. As discussed in Section IV-C, we could not use broadcast transmission in the case of transmission loss. Instead, each activated node i has to perform point-to-point transmission for $\hat{\lambda}_{i|j}$ from node i to node $j \in \mathcal{N}_i$.

Due to transmission failure, PDMM was initialized differently from the first experiment. Each time the algorithm was tested, the initial estimate $(\hat{\boldsymbol{x}}^0, \hat{\boldsymbol{\lambda}}^0)$ was set as

$$(\hat{\boldsymbol{x}}^0, \hat{\boldsymbol{\lambda}}^0) = (\mathbf{0}, \mathbf{0}), \tag{67}$$

which guarantees that every node in the graph has access to the initial estimates of neighboring nodes without transmission.

Fig. 3 demonstrates the performance of PDMM under three transmission losses: 0%, 20% and 40%. Subplot (a) and (b) are for the asynchronous and synchronous schemes, respectively. Each curve in the two subplots was obtained by averaging over 100 simulations to mitigate the effect of random transmission losses. It is seen that transmission failure only slows down the convergence speed of the algorithm. The above property is highly desirable in real applications because transmission losses might be inevitable in some networks (e.g., see [29] for investigation of packet-loss over wireless sensor networks in different environments).

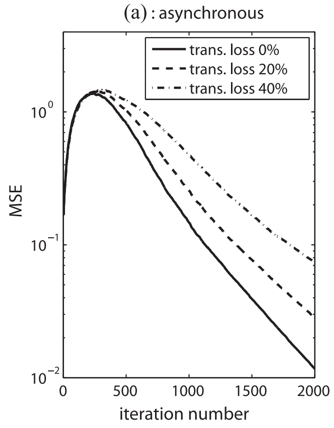


Fig. 3. Performance of synchronous/asynchronous PDMM under different transmission losses (%).

Finally, it is observed that for each transmission-loss in subplot (a), the error goes up in the first few hundred of iterations before deceasing. This may be because of the special initialization (67). We have tested the initialization $\{\hat{x}_i^0 = t_i\}$ for 0% transmission loss, where the MSE decreases along with the iterations monotonically.

3) Performance Comparison: In this experiment, we investigated the convergence speeds of four algorithms under the condition of no transmission failure. Besides PDMM, we also implemented the broadcast-based algorithm in [14] (referred to as *broadcast*), the randomized gossip algorithm in [5] (referred to as *gossip*) and ADMM. Unlike PDMM and ADMM that can work either synchronously or asynchronously, both *broadcast* and *gossip* algorithms can only work asynchronously. While *broadcast* algorithm randomly activates one node per iteration, *gossip* algorithm randomly activates one edge per iteration for parameter-updating.

Similarly to the first experiment, we also evaluated PDMM for both the synchronous and asynchronous schemes. For the asynchronous scheme, we tested all the four algorithms introduced above while for the synchronous scheme, we focused on PDMM and ADMM. The implementation of the synchronous/asynchronous ADMM follows from [10] and [17], respectively. The asynchronous ADMM [17] is similar to the *gossip* algorithm in the sense that both algorithms activates one edge per iteration.

We note that the asynchronous ADMM essentially activates two neighboring nodes per iteration. To make a fair comparison between PDMM and ADMM, we implemented two versions of PDMM for the asynchronous scheme. The first version follows Section IV-B where each iteration randomly activates one node as the *gossip* algorithm, referred to as *one-node PDMM*. The second version of PDMM randomly activates two neighboring nodes per iteration as the *broadcast* algorithm, referred to as *two-node PDMM*.

Both PDMM and ADMM have some parameters to be specified. To simplify the implementation, we let $\gamma_p = \gamma_d = 1$ in PDMM (which is not the optimal setting from Fig. 2). Similarly, we set the parameter in ADMM to be 1.

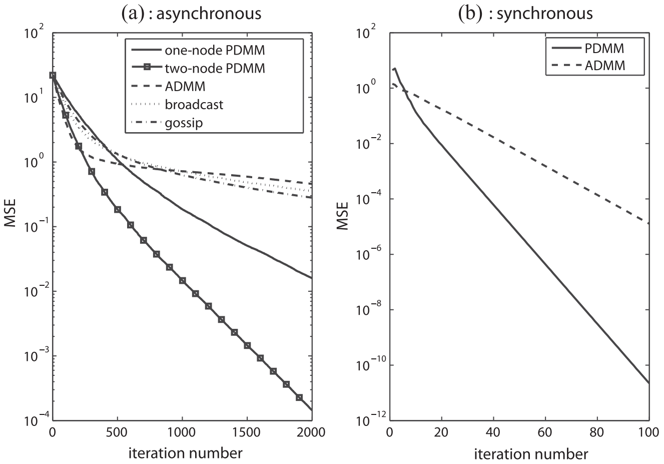


Fig. 4. Performance comparison under perfect channel. The two curves in subplot (b) at iteration 1 have a noticeable gap compared to subplot (a). This is because under the synchronous scheme, all the parameters of each method are updated per iteration, leading to a relatively big performance difference in the beginning.

TABLE II
AVERAGE EXECUTION TIMES (PER ITERATION) AND THEIR STANDARD DEVIATIONS FOR THE FOUR METHODS

	one-node PDMM	two-node PDMM	ADMM	broadcast	gossip	PDMM (syn)	ADMM (syn)
ave. (μs)	5.46	8.92	6.54	2.10	0.24	380	384
std (10^{-6})	5.04	8.58	8.09	4.55	1.73	216	285

In the experiment, the *gossip* and *broadcast* algorithms were initialized according to [5] and [14], respectively. The initialization for PDMM was the same as in the first experiment. The estimates of ADMM were initialized similarly as for PDMM.

Fig. 4 displays the MSE trajectories for the four methods while Table II lists the average execution times (per iteration) and their standard deviations. Similarly to the second experiment, the performance of each method for the asynchronous scheme was obtained by averaging over 100 simulations to mitigate the effect of randomness introduced in node or edge-activation. We now focus on the asynchronous scheme. It is seen from Fig. 4(a) that the *two-node PDMM* converges the fastest in terms of number of iterations while the *gossip* algorithm requires the least execution time on average. The above results suggest that for applications where signal transmission is more expensive than local computation (w.r.t. energy consumption), PDMM might be a good candidate as it may save number of iterations.

Fig. 4(b) demonstrates the MSE performance of PDMM and ADMM for the synchronous scheme. Both algorithms appear to have linear convergence rates. This may be because the objective functions in (60) are strongly convex and have gradients which are Lipschitz continuous. It is seen from Table II that both methods take roughly the same execution time. By combining the above results, we conclude that under synchronous scheme, PDMM converges faster than ADMM w.r.t. the execution time, which may be due to the fact that PDMM avoids the auxiliary variable $\boldsymbol{z}$ used in ADMM.

VII. Conclusion

In this paper, we have proposed PDMM for iterative optimization over a general graph. The augmented primal-dual Lagrangian function is constructed of which a saddle point provides an optimal solution of the original problem, which leads to the design of PDMM. PDMM performs broadcast transmission under perfect channel and point-to-point transmission under non-perfect channel. We have shown that both the synchronous and asynchronous PDMMs possess a convergence rate of $\mathcal{O}(1/K)$ for general closed, proper and convex functions defined over the graph. As an example, we have applied PDMM for distributed averaging, through which properties of PDMM such as proper parameter-selection and resilience against transmission failure are further investigated.

We note that PDMM is natural when performing node-oriented optimization over a graph as compared to ADMM which involves computing the edge variable $\boldsymbol{z}$ introduced in (3). A few applications in [21], [22] and [23] suggest that PDMM is practically promising. While convergence properties of ADMM under different conditions (e.g., strong convexity and/or the gradients being Lipschitz continuous) are well understood, the convergence properties of PDMM for those conditions remain to be discovered.

Appendix A
Proof for Lemma 8

Before presenting the proof, we first introduce a basic inequality, which is described in a lemma below:

Lemma 13: Let $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ be two arbitrary closed, proper and convex functions. $\boldsymbol{x}^\star$ minimizes the sum of the two functions, i.e., $\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x}}(f_1(\boldsymbol{x}) + f_2(\boldsymbol{x}))$. Then, there is

$$f_1(\boldsymbol{x}) - f_1(\boldsymbol{x}^\star) \geq (\boldsymbol{x}^\star - \boldsymbol{x})^T \boldsymbol{r}(\boldsymbol{x}^\star) \quad \forall \boldsymbol{x}, \tag{68}$$

where $\boldsymbol{r}(\boldsymbol{x}^\star) \in \partial_{\boldsymbol{x}} f_2(\boldsymbol{x}^\star)$.

The above inequality is wildly exploited for the convergence analysis of ADMM and its variants [10], [27], [28]. We will also use the inequality in our proof.

Applying (68) to the updating (29)–(30) for $(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1})$, we obtain a set of inequalities for all $(\boldsymbol{x}, \boldsymbol{\lambda}) \in (\mathbb{R}^{\sum n_i}, \mathbb{R}^{2\sum n_{ij}})$ as

$$\sum_{j\in\mathcal{N}_i} \left[\boldsymbol{P}_{d,ij}(\hat{\boldsymbol{\lambda}}_{j|i}^{k} - \hat{\boldsymbol{\lambda}}_{i|j}^{k+1}) + \boldsymbol{c}_{ij} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k\right]^T (\boldsymbol{\lambda}_{i|j} - \hat{\boldsymbol{\lambda}}_{i|j}^{k+1})$$
$$\leq f_i^*(\boldsymbol{A}_i^T \boldsymbol{\lambda}_i) - f_i^*(\boldsymbol{A}_i^T \hat{\boldsymbol{\lambda}}_i^{k+1}) \qquad \forall i \in \mathcal{V} \tag{69}$$

$$\sum_{j\in\mathcal{N}_i} \left[\left(\boldsymbol{P}_{p,ij}(\boldsymbol{c}_{ij} - \boldsymbol{A}_{ij}\boldsymbol{x}_i^{k+1} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k) + \hat{\boldsymbol{\lambda}}_{j|i}^k\right)^T \boldsymbol{A}_{ij}\right.$$
$$\left. \cdot (\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i^{k+1})\right] \leq f_i(\boldsymbol{x}_i) - f_i(\hat{\boldsymbol{x}}_i^{k+1}) \quad \forall i \in \mathcal{V}. \tag{70}$$

Adding (69)–(70) over all $i \in \mathcal{V}$, and substituting $(\boldsymbol{x}, \boldsymbol{\lambda}) = (\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$, the saddle point of $L_{\mathcal{P}}$, yields

$$\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[(\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \boldsymbol{\lambda}_{i|j}^\star)^T \left(\boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{k+1} - \frac{\boldsymbol{c}_{ij}}{2}\right) - (\hat{\boldsymbol{x}}_i^{k+1} - \boldsymbol{x}_i^\star)^T$$
$$\cdot \boldsymbol{A}_{ij}^T \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}\Big] + p(\hat{\boldsymbol{x}}^{k+1}, \hat{\boldsymbol{\lambda}}^{k+1}) - p(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$$
$$\leq \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[\Big(\boldsymbol{P}_{p,ij}(\boldsymbol{c}_{ij} - \boldsymbol{A}_{ij}\boldsymbol{x}_i^{k+1} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^k) + \hat{\boldsymbol{\lambda}}_{j|i}^k$$
$$- \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}\Big)^T \boldsymbol{A}_{ij}(\hat{\boldsymbol{x}}_i^{k+1} - \boldsymbol{x}_i^\star) + (\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \boldsymbol{\lambda}_{i|j}^\star)^T$$
$$\cdot \Big(\boldsymbol{P}_{d,ij}(\hat{\boldsymbol{\lambda}}_{j|i}^k - \hat{\boldsymbol{\lambda}}_{i|j}^{k+1}) + \boldsymbol{A}_{ji}(\hat{\boldsymbol{x}}_j^{k+1} - \hat{\boldsymbol{x}}_j^k)\Big)\Big]$$
$$= \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[\Big(\boldsymbol{P}_{p,ij}\boldsymbol{A}_{ji}(\hat{\boldsymbol{x}}_j^{k+1} - \hat{\boldsymbol{x}}_j^k) + \hat{\boldsymbol{\lambda}}_{j|i}^k - \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}\Big)^T$$
$$\cdot \boldsymbol{A}_{ij}(\hat{\boldsymbol{x}}_i^{k+1} - \boldsymbol{x}_i^\star) + (\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \boldsymbol{\lambda}_{i|j}^\star)^T$$
$$\cdot \Big(\boldsymbol{P}_{d,ij}(\hat{\boldsymbol{\lambda}}_{j|i}^k - \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}) + \boldsymbol{A}_{ji}(\hat{\boldsymbol{x}}_j^{k+1} - \hat{\boldsymbol{x}}_j^k)\Big)\Big]$$
$$- \sum_{(i,j)\in\mathcal{E}}\Big(\|\boldsymbol{c}_{ij} - \boldsymbol{A}_{ij}\boldsymbol{x}_i^{k+1} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{k+1}\|_{\boldsymbol{P}_{p,ij}}^2$$
$$+ \|\hat{\boldsymbol{\lambda}}_{i|j}^{k+1} - \hat{\boldsymbol{\lambda}}_{j|i}^{k+1}\|_{\boldsymbol{P}_{d,ij}}^2\Big), \tag{71}$$

where the last equality follows from the two optimality conditions (25)–(26).

To further simplify (71), one can first insert the alternative expression (40) for every $\boldsymbol{P}_{d,ij}$ into (71). After that, the expression (49) can be obtained by simplifying the new expression using (25)–(26) and the following identity

$$(\boldsymbol{y}_1 - \boldsymbol{y}_2)^T(\boldsymbol{y}_3 - \boldsymbol{y}_4)$$
$$\equiv \frac{1}{2}(\|\boldsymbol{y}_1 + \boldsymbol{y}_3\|^2 - \|\boldsymbol{y}_1 + \boldsymbol{y}_4\|^2 - \|\boldsymbol{y}_2 + \boldsymbol{y}_3\|^2 + \|\boldsymbol{y}_2 + \boldsymbol{y}_4\|^2).$$

Appendix B
Proof of Lemma 11

The basic idea for the proof is similar to that for Lemma 8 as presented in Appendix A. However, since asynchronous PDMM activates one node $i \in \mathcal{V}$ per iteration, it is difficult to tell which neighbors of i have been recently activated and which have not yet. The above difficulty requires careful treatment in the convergence analysis. We sketch the proof in the following for reference.

We focus on the parameter-updating for a particular segment of iterations $k \in \{ml, ml+1, \ldots, ml+m-1\}$, where $l \geq 0$. For simplicity, we denote the activated node i at iteration k as $i(k)$. To start with, we apply (68) to the updating (31) for the estimate $(\hat{\boldsymbol{x}}_{i(k)}^{k+1}, \hat{\boldsymbol{\lambda}}_{i(k)}^{k+1})$ of node $i(k)$. In order to do so, we first have to consider the estimates of its neighbors. It may happen that some neighbors have already been activated within the segment while others are still waiting to be activated. If a neighbor $j \in \mathcal{N}_{i(k)}$ is still waiting, we then have $(\hat{\boldsymbol{x}}_j^k, \hat{\boldsymbol{\lambda}}_j^k) = (\hat{\boldsymbol{x}}_j^{lm}, \hat{\boldsymbol{\lambda}}_j^{lm})$. Conversely, if a neighbor $j \in \mathcal{N}_{i(k)}$ has already been activated, we then have $(\hat{\boldsymbol{x}}_j^k, \hat{\boldsymbol{\lambda}}_j^k) = (\hat{\boldsymbol{x}}_j^{(l+1)m}, \hat{\boldsymbol{\lambda}}_j^{(l+1)m})$. From Lemma 10, it is clear that if $j < i(k)$ (or $j > i(k)$), then the neighbor j has been activated (not yet activated). For simplicity, we use a function $s(k,j)$ to denote the value lm or $(l+1)m$ for a neighbor $j \in \mathcal{N}_{i(k)}$ at iteration k

$$s(k,j) = \begin{cases} lm & j > i(k) \\ (l+1)m & j < i(k) \end{cases}. \tag{72}$$

As for the activated node $i(k)$, we have $(\hat{\boldsymbol{x}}_{i(k)}^{k+1}, \hat{\boldsymbol{\lambda}}_{i(k)}^{k+1}) = (\hat{\boldsymbol{x}}_{i(k)}^{(l+1)m}, \hat{\boldsymbol{\lambda}}_{i(k)}^{(l+1)m})$. As a result, the two inequalities for $\hat{\boldsymbol{x}}_{i(k)}^{k+1}$ and $\hat{\boldsymbol{\lambda}}_{i(k)}^{k+1}$ are given by

$$\begin{aligned}
&\sum_{j\in\mathcal{N}_{i(k)}} \Big[\boldsymbol{P}_{d,i(k)j}(\hat{\boldsymbol{\lambda}}_{j|i(k)}^{s(k,j)} - \hat{\boldsymbol{\lambda}}_{i(k)|j}^{(l+1)m}) - \boldsymbol{A}_{ji(k)}\hat{\boldsymbol{x}}_j^{s(k,j)} \\
&\qquad + \boldsymbol{c}_{i(k)j}\Big]^T \Big(\boldsymbol{\lambda}_{i(k)|j} - \hat{\boldsymbol{\lambda}}_{i(k)|j}^{(l+1)m}\Big) \\
&\leq f_{i(k)}^*\Big(\boldsymbol{A}_{i(k)}^T\boldsymbol{\lambda}_{i(k)}\Big) - f_{i(k)}^*\Big(\boldsymbol{A}_{i(k)}^T\hat{\boldsymbol{\lambda}}_{i(k)}^{(l+1)m}\Big) \qquad (73)\\
&\sum_{j\in\mathcal{N}_{i(k)}} \Big[\boldsymbol{P}_{p,i(k)j}\Big(-\boldsymbol{A}_{i(k)j}\boldsymbol{x}_{i(k)}^{(l+1)m} - \boldsymbol{A}_{ji(k)}\hat{\boldsymbol{x}}_j^{s(k,j)} \\
&\qquad + \boldsymbol{c}_{i(k)j}\Big) + \hat{\boldsymbol{\lambda}}_{j|i(k)}^{s(k,j)}\Big]^T \boldsymbol{A}_{i(k)j}\Big(\boldsymbol{x}_{i(k)} - \hat{\boldsymbol{x}}_{i(k)}^{(l+1)m}\Big) \\
&\leq f_{i(k)}\big(\boldsymbol{x}_{i(k)}\big) - f_{i(k)}\Big(\hat{\boldsymbol{x}}_{i(k)}^{(l+1)m}\Big), \qquad (74)
\end{aligned}$$

where $lm \leq k < (l+1)m$.

Next adding (73)–(74) over all $lm \leq k < (l+1)m$ and substituting $(\boldsymbol{x}, \boldsymbol{\lambda}) = (\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ yields

$$\begin{aligned}
&\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}\Big[\Big(\hat{\boldsymbol{\lambda}}_{i|j}^{(l+1)m} - \boldsymbol{\lambda}_{i|j}^\star\Big)^T\Big(\boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{(l+1)m} - \frac{\boldsymbol{c}_{ij}}{2}\Big) - \Big(\hat{\boldsymbol{x}}_i^{(l+1)m} - \boldsymbol{x}_i^\star\Big)^T \\
&\quad \cdot \boldsymbol{A}_{ij}^T\hat{\boldsymbol{\lambda}}_{j|i}^{(l+1)m}\Big] + p\Big(\hat{\boldsymbol{x}}^{(l+1)m}, \hat{\boldsymbol{\lambda}}^{(l+1)m}\Big) - p(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star) \\
&\leq \sum_{k=lm}^{(l+1)m-1}\sum_{j\in\mathcal{N}_{i(k)}}\Bigg[\Big[\boldsymbol{P}_{d,i(k)j}\Big(\hat{\boldsymbol{\lambda}}_{j|i(k)}^{s(k,j)} - \hat{\boldsymbol{\lambda}}_{i(k)|j}^{(l+1)m}\Big) \\
&\qquad + \boldsymbol{A}_{ji(k)}\Big(\hat{\boldsymbol{x}}_j^{(l+1)m} - \hat{\boldsymbol{x}}_j^{s(k,j)}\Big)\Big]^T\Big(\hat{\boldsymbol{\lambda}}_{i(k)|j}^{(l+1)m} - \boldsymbol{\lambda}_{i(k)|j}^\star\Big) \\
&\quad + \Big[\boldsymbol{P}_{p,i(k)j}\Big(\boldsymbol{c}_{i(k)j} - \boldsymbol{A}_{i(k)j}\boldsymbol{x}_{i(k)}^{(l+1)m} - \boldsymbol{A}_{ji(k)}\hat{\boldsymbol{x}}_j^{s(k,j)}\Big) \\
&\qquad + \hat{\boldsymbol{\lambda}}_{j|i(k)}^{s(k,j)} - \hat{\boldsymbol{\lambda}}_{j|i(k)}^{(l+1)m}\Big]^T\boldsymbol{A}_{i(k)j}\Big(\hat{\boldsymbol{x}}_{i(k)}^{(l+1)m} - \boldsymbol{x}_{i(k)}^\star\Big)\Bigg] \\
&= \sum_{k=lm}^{(l+1)m-1}\sum_{j\in\mathcal{N}_{i(k)}} g(k, i(k), j) - \sum_{(i,j)\in\mathcal{E}}\Big(\Big\|\hat{\boldsymbol{\lambda}}_{i|j}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{j|i}^{(l+1)m}\Big\|_{\boldsymbol{P}_{d,ij}}^2 \\
&\qquad + \Big\|\boldsymbol{c}_{ij} - \boldsymbol{A}_{ij}\hat{\boldsymbol{x}}_i^{(l+1)m} - \boldsymbol{A}_{ji}\hat{\boldsymbol{x}}_j^{(l+1)m}\Big\|_{\boldsymbol{P}_{p,ij}}^2\Big), \qquad (75)
\end{aligned}$$

where the function $g(k, i(k), j)$ is defined as

$$\begin{aligned}
&g(k, i(k), j) \\
&= \Big[\boldsymbol{P}_{d,i(k)j}\Big(\hat{\boldsymbol{\lambda}}_{j|i(k)}^{s(k,j)} - \hat{\boldsymbol{\lambda}}_{j|i(k)}^{(l+1)m}\Big) \\
&\qquad + \boldsymbol{A}_{ji(k)}\Big(\hat{\boldsymbol{x}}_j^{(l+1)m} - \hat{\boldsymbol{x}}_j^{s(k,j)}\Big)\Big]^T\Big(\hat{\boldsymbol{\lambda}}_{i(k)|j}^{(l+1)m} - \boldsymbol{\lambda}_{i(k)|j}^\star\Big) \\
&\quad + \Big[\boldsymbol{P}_{p,i(k)j}\boldsymbol{A}_{ji(k)}\Big(\hat{\boldsymbol{x}}_j^{(l+1)m} - \hat{\boldsymbol{x}}_j^{s(k,j)}\Big) \\
&\qquad + \hat{\boldsymbol{\lambda}}_{j|i(k)}^{s(k,j)} - \hat{\boldsymbol{\lambda}}_{j|i(k)}^{(l+1)m}\Big]^T\boldsymbol{A}_{i(k)j}\Big(\hat{\boldsymbol{x}}_{i(k)}^{(l+1)m} - \boldsymbol{x}_{i(k)}^\star\Big),
\end{aligned}$$

where $lm \leq k < (l+1)m$ and $j \in \mathcal{N}_{i(k)}$.

Now we are in a position to analyze the right hand side of (75). By using the fact that each node i has $|\mathcal{N}_i|$ different functions $g(k, i(k), j)$, we can conclude that each edge $(u,v) \in \mathcal{E}$ is associated with two functions $g(k_1, u(k_1), v)$ and $g(k_2, v(k_2), u)$, where iteration k_1 and k_2 activate u and v, respectively. From (75), it is clear that each edge (u,v) is also associated with the other two functions $\|\boldsymbol{c}_{uv} - \boldsymbol{A}_{uv}\hat{\boldsymbol{x}}_u^{(l+1)m} - \boldsymbol{A}_{vu}\hat{\boldsymbol{x}}_v^{(l+1)m}\|_{\boldsymbol{P}_{p,uv}}^2$ and $\|\hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m}\|_{\boldsymbol{P}_{d,uv}}^2$. We show in the following that the combination of the above four functions for every edge $(u,v) \in \mathcal{E}$ is independent of k_1 and k_2. In order to do so, we assume $k_1 < k_2$ (or equivalently, $u < v$ from Lemma 10). From (72), we know that $s(k_1, v) = lm$ and $s(k_2, u) = (l+1)m$. Based on the above information, the four functions for $(u,v) \in \mathcal{E}$ can be simplified as

$$\begin{aligned}
&g(k_1, u(k_1), v) + g(k_2, v(k_2), u) - \|\hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m}\|_{\boldsymbol{P}_{d,uv}}^2 \\
&- \|\boldsymbol{c}_{uv} - \boldsymbol{A}_{uv}\hat{\boldsymbol{x}}_u^{(l+1)m} - \boldsymbol{A}_{vu}\hat{\boldsymbol{x}}_v^{(l+1)m}\|_{\boldsymbol{P}_{p,uv}}^2 \\
&= g(k_1, u(k_1), v) - \|\hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m}\|_{\boldsymbol{P}_{d,uv}}^2 \\
&- \|\boldsymbol{c}_{uv} - \boldsymbol{A}_{uv}\hat{\boldsymbol{x}}_u^{(l+1)m} - \boldsymbol{A}_{vu}\hat{\boldsymbol{x}}_v^{(l+1)m}\|_{\boldsymbol{P}_{p,uv}}^2 \\
&= \Big[\boldsymbol{P}_{d,uv}\Big(\hat{\boldsymbol{\lambda}}_{v|u}^{lm} - \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m}\Big) + \boldsymbol{A}_{vu}\Big(\hat{\boldsymbol{x}}_v^{(l+1)m} - \hat{\boldsymbol{x}}_v^{lm}\Big)\Big]^T \\
&\cdot\Big(\hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m} - \boldsymbol{\lambda}_{u|v}^\star\Big) + \Big[\boldsymbol{P}_{p,uv}\boldsymbol{A}_{vu}\Big(\hat{\boldsymbol{x}}_v^{(l+1)m} - \hat{\boldsymbol{x}}_v^{lm}\Big) + \hat{\boldsymbol{\lambda}}_{v|u}^{lm} \\
&\ - \hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m}\Big]^T\boldsymbol{A}_{uv}\Big(\hat{\boldsymbol{x}}_u^{(l+1)m} - \boldsymbol{x}_u^\star\Big) - \|\hat{\boldsymbol{\lambda}}_{v|u}^{(l+1)m} - \hat{\boldsymbol{\lambda}}_{u|v}^{(l+1)m}\|_{\boldsymbol{P}_{d,uv}}^2 \\
&\quad - \|\boldsymbol{c}_{uv} - \boldsymbol{A}_{uv}\hat{\boldsymbol{x}}_u^{(l+1)m} - \boldsymbol{A}_{vu}\hat{\boldsymbol{x}}_v^{(l+1)m}\|_{\boldsymbol{P}_{p,uv}}^2 \qquad (76)\\
&= d_{uv}^{l+1} \qquad u < v, \qquad (77)
\end{aligned}$$

where d_{uv}^{l+1} is given by (56), of which the derivation is similar to that for $d_{i|j}^{k+1}$ in (50). The term $u(k_1)$ in (76) is simplified as u since we already assume that at iteration k_1, node u is activated. The quantity d_{uv}^{l+1} is a function of m and l instead of k_1. Finally, combining (75) and (77) produces (55).

References

[1] G. Zhang, R. Heusdens, and W. B. Kleijn, "On the convergence rate of the Bi-Alternating direction method of multipliers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2014, pp. 3897–3901.

[2] G. Zhang and R. Heusdens, "Bi-Alternating direction method of multipliers over graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 2015, pp. 3571–3575.

[3] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge U.K.: Cambridge Univ. Press, 2008.

[4] G. Zhang, R. Heusdens, and W. B. Kleijn, "Large scale LP decoding with low complexity," *IEEE Commun. Lett.*, vol. 17, no. 11, pp. 2152–2155, Nov. 2013.

[5] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.

[6] D. Sontag, A. Globerson, and T. Jaakkola, "Introduction to dual decomposition for inference," in *Optimization for Machine Learning*. Cambridge, MA, USA: MIT Press, 2011.

[7] Y. Zeng and R. Heusdens, "Linear coordinate-descent message-passing for quadratic optimization," *Neural Comput.*, vol. 24, no. 12, pp. 3340–3370, 2012.

[8] C. C. Moallemi and B. V. Roy, "Convergence of min-sum message passing for quadratic optimization," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2413–2423, May 2009.

[9] G. Zhang and R. Heusdens, "Convergence of min-sum-min message-passing for Quadratic Optimization," in *Proc. Eur. Conf. Mach. Learn.*, 2014, pp. 353–368.

[10] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[11] J. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Trans. Autom. Control*, vol. 57, no. 3, 2012, pp. 592–606, Mar. 2012.

[12] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[13] J. Chen and A. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[14] F. Lutzeler, P. Ciblat, and W. Hachem, "Analysis of sum-weight-like algorithms for averaging in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 61, no. 11, pp. 2802–2814, Jun. 2013.

[15] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.

[16] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 7, pp. 1750–1761, Apr. 2014.

[17] E. Wei and A. Ozdaglar, "On the O(1/k) convergence of asynchronous distributed alternating direction method of multipliers," [Online]. Available: available on arxiv.org.

[18] R. Zhang and J. T. Kwok, "Asynchronous distributed ADMM for consensus optimization," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 1701–1709.

[19] T.-H. Chang, M. Hong, W.-C. Liao, and X. Wang, "Asynchronous distributed ADMM for Large-Scale Optimization- Part I: algorithm and convergence analysis," *IEEE Trans. Signal Processing*, vol. 64, no. 12, pp. 3118–3130, 2016.

[20] P. Bianchi, W. Hachem, and F. Iutzeler, "A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization," *IEEE Trans. Autom. Control*, vol. 61, no. 10, pp. 2947–2957, Oct. 2016.

[21] H. M. Zhang, "Distributed convex optimization: A study on the primal-dual method of multipliers," M.S. thesis, Delft Univ. Technol., Delft, The Netherlands, 2015.

[22] G. Zhang and R. Heusdens, "On simplifying the primal-dual method of multipliers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 2016, pp. 4826–4830.

[23] T. Sherson, W. B. Kleijn, and R. Heusdens, "A distributed algorithm for robust LCMV beamforming," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 101–105.

[24] Y. Sawaragi, H. Nakayama, and T. Tanino, *Theory of Multiobjective Optimization*. Amsterdam, The Netherlands: Elsevier, 1985.

[25] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *J. Math. Imag. Vis.*, vol. 40, pp. 120–145, 2011.

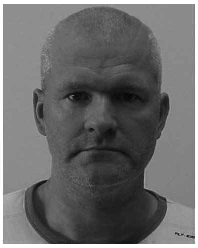
[26] B. He and X. Yuan, "Convergence analysis of primal-dual algorithms for a saddle-point problem: From contraction perspective," *SIAM J. Imag. Sci.*, vol. 5, no. 1, pp. 119–149, 2012.

[27] H. Wang and A. Banerjee, "Online alternating direction method," in *Proc. Int. Conf. Mach. Learning*, Jun. 2012.

[28] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," *J. Sci. Comput.*, vol. 66, no. 3, pp. 889–916, 2016.

[29] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *Proc. 1st Int. Conf. Embedded Netw. Sensor Syst.*, 2003, pp. 1–13.

Guoqiang Zhang received the B.Eng. degree from the University of Science and Technology of China, Hefei, China, in 2003, the M.Phil. degree from the University of Hong Kong, Hong Kong, in 2006, and the Ph.D. degree from the Royal Institute of Technology, Stockholm, Sweden, in 2010. From the spring of 2011, he worked as a Postdoctoral Researcher at Delft University of Technology. From the spring of 2015, he worked as a Senior Researcher at the Ericsson AB, Sweden. Since 2017, he has been a Senior Lecturer in the School of Computing and Communications, University of Technology Sydney, Ultimo NSW, Australia. His current research interests include algorithm design and performance analysis for distributed processing, signal processing over wireless sensor networks, and application of distributed optimization in training artificial neural networks.



Richard Heusdens received the M.Sc. and Ph.D. degrees from the Delft University of Technology, Delft, The Netherlands, in 1992 and 1997, respectively. Since 2002, he has been an Associate Professor in the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology. In the spring of 1992, he joined the Digital Signal Processing Group at the Philips Research Laboratories, Eindhoven, The Netherlands. He has worked on various topics in the field of signal processing, such as image/video compression and VLSI architectures for image processing algorithms. In 1997, he joined the Circuits and Systems Group of Delft University of Technology, where he was a Postdoctoral Researcher. In 2000, he moved to the Information and Communication Theory (ICT) Group, where he became an Assistant Professor responsible for the audio/speech signal processing activities within the ICT group. He held visiting positions at KTH Royal Institute of Technology, Sweden, in 2002 and 2008, and from 2014 to 2016, he was a Guest Professor with Aalborg University. He is involved in research projects that cover subjects such as audio and acoustic signal processing, speech enhancement, and distributed signal processing.