2016

Asynchronous Distributed ADMM for Large-Scale Optimization—Part I: Algorithm and Convergence Analysis

Tsung-Hui Chang
The Chinese University of Hong Kong

Mingyi Hong
Iowa State University, mingyi@iastate.edu

Wei-Cheng Liao
University of Minnesota - Twin Cities

Xiangfeng Wang
East China Normal University

Asynchronous Distributed ADMM for Large-Scale Optimization- Part I: Algorithm and Convergence Analysis

Tsung-Hui Chang⋆, Mingyi Hong†, Wei-Cheng Liao§ and Xiangfeng Wang‡

Abstract

Aiming at solving large-scale optimization problems, this paper studies distributed optimization methods based on the alternating direction method of multipliers (ADMM). By formulating the optimization problem as a consensus problem, the ADMM can be used to solve the consensus problem in a fully parallel fashion over a computer network with a star topology. However, traditional synchronized computation does not scale well with the problem size, as the speed of the algorithm is limited by the slowest workers. This is particularly true in a heterogeneous network where the computing nodes experience different computation and communication delays. In this paper, we propose an asynchronous distributed ADMM (AD-ADMM) which can effectively improve the time efficiency of distributed optimization. Our main interest lies in analyzing the convergence conditions of the AD-ADMM, under the popular *partially asynchronous* model, which is defined based on a maximum tolerable delay of the network. Specifically, by considering general and possibly non-convex cost functions, we show that the AD-ADMM is guaranteed to converge to the set of Karush-Kuhn-Tucker (KKT) points as long as the algorithm parameters are chosen appropriately according to the network delay. We further illustrate that the asynchrony of the ADMM has to be handled with care, as slightly modifying the implementation of the AD-ADMM can jeopardize the algorithm convergence, even under the standard convex setting.

Keywords– Distributed optimization, ADMM, Asynchronous, Consensus optimization

Part of this work was submitted to IEEE ICASSP, Shanghai, China, March 20-25, 2016 [1]. Tsung-Hui Chang is supported by NSFC, China, Grant No. 61571385. Mingyi Hong is supported by NFS Grant No. CCF-1526078 , and AFOSR, Grant No. 15RT0767. Xiangfeng Wang is supported by Shanghai YangFan No. 15YF1403400 and NSFC No. 11501210.

⋆Tsung-Hui Chang is the corresponding author. Address: School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China 518172. E-mail: tsunghui.chang@ieee.org.

†Mingyi Hong is with Department of Industrial and Manufacturing Systems Engineering, Iowa State University, Ames, 50011, USA, E-mail: mingyi@iastate.edu

†Wei-Cheng Liao is with Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455, USA, E-mail: liaox146@umn.edu

‡Xiangfeng Wang is with Shanghai Key Lab for Trustworthy Computing, School of Computer Science and Software Engineering, East China Normal University, Shanghai, 200062, China, E-mail: xfwang@sei.ecnu.edu.cn

I. Introduction

A. Background

Scaling up optimization algorithms for future data-intensive applications calls for efficient distributed and parallel implementations, so that modern multi-core high performance computing technologies can be fully utilized [2]–[4]. In this work, we are interested in developing distributed optimization methods for solving the following optimization problem

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \ \sum_{i=1}^{N} f_i(\boldsymbol{x}) + h(\boldsymbol{x}), \tag{1}$$

where each $f_i : \mathbb{R}^n \to \mathbb{R}$ is a (smooth) cost function; $h : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is a convex (proper and lower semi-continuous) but possibly non-smooth regularization function. The latter is used to impose desired structures on the solution (e.g., sparsity) and/or used to enforce certain constraints. Problem (1) includes as special cases many important statistical learning problems such as the LASSO problem [5], logistic regression (LR) problem [6], support vector machine (SVM) [7] and the sparse principal component analysis (PCA) problem [8]. In this paper, we focus on solving large-scale instances of these learning problems with either a large number of training samples or a large number of features (n is large) [3]. These are typical data-intensive machine learning scenarios in which the data sets are often distributedly located in a few computing nodes. Traditional centralized optimization methods, therefore, fails to scale well due to their inability to handle distributed data sets and computing resources.

Our goal is to develop efficient distributed optimization algorithms over a computer network with a star topology, in which a master node coordinates the computation of a set of distributed workers (see Figure 1 for illustration). Such star topology represents a common architecture for distributed computing, therefore it has been used widely in distributed optimization [4], [9]–[16]. For example, under the star topology, references [10], [11] presented distributed stochastic gradient descent (SGD) methods, references [12], [13] parallelized the proximal gradient (PG) methods, while references [14]–[17] parallelized the block coordinate descent (BCD) method. In these works, the distributed workers iteratively calculate the gradients related to their local data, while the master collects such information from the workers to perform SGD, PG or BCD updates.

However, when scaling up these distributed algorithms, node synchronization becomes an important issue. Specifically, under the synchronous protocol, the master is triggered at each iteration only if it receives the required information from all the distributed workers. On the one hand, such synchronization is beneficial to make the algorithms well behaved; on the other hand, however, the speed of the algorithms

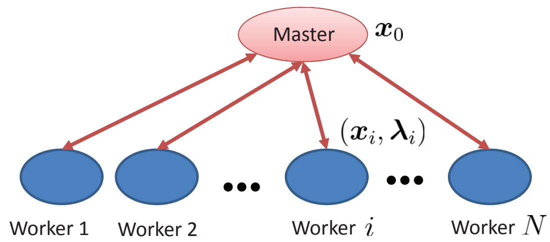


Fig. 1: A star computer cluster with one master and N workers.

would be limited by the "slowest" worker especially when the workers have different computation and communication delays. To address such dilemma, a few recent works [10]–[14] have introduced "asynchrony" into the distributed algorithms, which allows the master to perform updates when not all, but a small subset of workers have returned their gradient information. The asynchronous updates would cause "delayed" gradient information. A few algorithmic tricks such as delay-dependent step-size selection have been introduced to ensure that the staled gradient information does not destroy the stability of the algorithm. In practice, such asynchrony does make a big difference. As has been consistently reported in [10]–[14], under such an asynchronous protocol, the computation time can decrease almost linearly with the number of workers.

B. Related Works

A different approach for distributed and parallel optimization is based on the alternating direction method of multipliers (ADMM) [9, Section 7.1.1]. In the distributed ADMM, the original learning problem is partitioned into N subproblems, each containing a subset of training samples or the learning parameters. At each iteration, the workers solve the subproblems and send the up-to-date variable information to the master, who summarizes this information and broadcasts the result to the workers. In this way, a given large-scale learning problem can be solved in a parallel and distributed fashion. Notably, other than the standard convex setting [9], the recent analysis in [18] has shown that such distributed ADMM is provably convergent to a Karush-Kuhn-Tucker (KKT) point even for non-convex problems.

Recently, the synchronous distributed ADMM [9], [18] has been extended to the asynchronous setting, similar to [10]–[14]. Specifically, reference [19] has considered a version of AD-ADMM with bounded delay assumption and studied its theoretical and numerical performances. However, only convex cases are considered in [19]. Reference [20] has studied another version of AD-ADMM for non-convex problems, which considers inexact subproblem updates and, similar to [10]–[14], the workers compute gradient information only. This type of distributed optimization schemes, however, may not fully utilize the computation powers of distributed nodes. Besides, due to inexact update, such schemes usually require more iterations to converge and thus may have higher communication overhead. References [21]–[23] have respectively considered asynchronous ADMM methods for decentralized optimization over networks. These works consider network topologies beyond the star network, but their definition of asynchrony is different from what we propose here. Specifically, the asynchrony in [21] lies in that, at each iteration, the nodes are randomly activated to perform variable update. The method presented in [22] further allows that the communications between nodes can succeed or fail randomly. It is shown in [22] that such asynchronous ADMM can converge in a probability-one sense, provided that the nodes and communication links satisfy certain statistical assumption. Reference [23] has considered an asynchronous dual ADMM method. The asynchrony is in the sense that the nodes are partitioned into groups based on certain coloring scheme and only one group of nodes update variable in each iteration.

C. Contributions

In this paper[1], we generalize the state-of-the-art synchronous distributed ADMM [9], [18] to the asynchronous setting. Like [10]–[14], [19], [20], the asynchronous distributed ADMM (AD-ADMM) algorithm developed in this paper gives the master the freedom of making updates only based on variable information from a partial set of workers, which further improves the computation efficiency of the distributed ADMM.

Theoretically, we show that, for general and possibly non-convex problems in the form of (1), the AD-ADMM converges to the set of KKT points if the algorithm parameters are chosen appropriately according to the maximum network delay. Our results differ significantly from the existing works [19], [21], [22] which are all developed for convex problems. Therefore, the analysis and algorithm proposed here are applicable not only to standard convex learning problems but also to important non-convex problems such as the sparse PCA problem [8] and matrix factorization problems [24]. To the best of our knowledge, except the inexact version in [20], this is the first time that the distributed ADMM is rigorously shown to be convergent for non-convex problems under the asynchronous protocol. Moreover, unlike [19], [21], [22] where the convergence analyses all rely on certain statistical assumption on the nodes/workers, our convergence analysis is deterministic and characterizes the worst-case convergence conditions of the AD-ADMM under a bounded delay assumption only. Furthermore, we demonstrate that the asynchrony of ADMM has to be handled with care – as a slight modification of the algorithm may

[1]In contrast to the conference paper [1], the current paper presents detailed proofs of theorems and more simulation results.

lead to completely different convergence conditions and even destroy the convergence of ADMM for convex problems. Some numerical results are presented to support our theoretical claims.

In the companion paper [25], the linear convergence conditions of the AD-ADMM is further analyzed. In addition, the numerical performance of the AD-ADMM is examined by solving a large-scale LR problem on a high-performance computer cluster.

Synopsis: Section II presents the applications of problem (1) and reviews the distributed ADMM in [9]. The proposed AD-ADMM and its convergence conditions are presented in Section III. Comparison of the proposed AD-ADMM with an alternative scheme is presented in Section IV. Some simulation results are presented in Section V. Finally, concluding remarks are given in Section VI.

II. Applications and Distributed ADMM

A. Applications

We target at solving problem (1) over a star computer network (cluster) with one master node and N workers/slaves, as illustrated in Figure 1. Such distributed optimization approach is extremely useful in modern big data applications [3]. For example, let us consider the following regularized empirical risk minimization problem [7]

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \ \sum_{j=1}^{m} \ell(\boldsymbol{a}_j^T\boldsymbol{w}, y_j) + \Omega(\boldsymbol{w}), \tag{2}$$

where m is the number of training samples and $\ell(\boldsymbol{a}_j^T\boldsymbol{w}, y_j)$ is a loss function (e.g., regression or classification error) that depends on the training sample $\boldsymbol{a}_j \in \mathbb{R}^n$, label y_j and the parameter vector $\boldsymbol{w} \in \mathbb{R}^n$. Here, n denotes the dimension of the parameters (features); $\Omega(\boldsymbol{w})$ is an appropriate convex regularizer. Problem (2) is one of the most important problems in signal processing and statistical learning, which includes the LASSO problem [26], LR [6], SVM [7] and the sparse PCA problem [8], to name a few. Obviously, solving (2) can be challenging when the number of training samples is very large. In that case, it is natural to split the training samples across the computer cluster and resort to a distributed optimization approach. Suppose that the m training samples are uniformly distributed and stored by the N workers, with each node i getting $q_i = \lfloor m/N \rfloor$ samples. By defining $f_i(\boldsymbol{w}) \triangleq \sum_{j=(i-1)q_i+1}^{iq_i} \ell(\boldsymbol{a}_j^T\boldsymbol{w}, y_j)$, $i = 1, \ldots, N$, and $h(\boldsymbol{w}) \triangleq \Omega(\boldsymbol{w})$, it is clear that (2) is an instance of (1).

When the number of training samples is moderate but the dimension of the parameters is very large ($n \gg m$), problem (2) is also challenging to solve. By [9, Section 7.3], one can instead consider the Lagrangian dual problem of (2) provided that (2) has zero duality gap. Specifically, let the training matrix $\boldsymbol{A} \triangleq [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m]^T \in \mathbb{R}^{m\times n}$ be partitioned as $\boldsymbol{A} = [\boldsymbol{A}_1, \ldots, \boldsymbol{A}_N]$, and let the parameter vector

$\boldsymbol{w}$ be partitioned conformally as $\boldsymbol{w} = [\boldsymbol{w}_1^T, \ldots, \boldsymbol{w}_N^T]^T$; moreover, assume that Ω is separable as $\Omega(\boldsymbol{w}) = \sum_{i=1}^N \Omega_i(\boldsymbol{w}_i)$. Then, following [9, Section 7.3], one can obtain the dual problem of (2) as

$$\min_{\boldsymbol{\nu}\in\mathbb{R}^m} \ \sum_{i=1}^N \Omega_i^*(\boldsymbol{A}_i^T\boldsymbol{\nu}) + \Phi^*(\boldsymbol{\nu}), \tag{3}$$

where $\boldsymbol{\nu} \triangleq [\nu_1, \ldots, \nu_m]^T$ is a dual variable, $\Phi^*(\boldsymbol{\nu}) = \sum_{j=1}^m \ell^*(\nu_j, y_j)$, and ℓ^* and Ω_i^* are respectively the conjugate functions of ℓ and Ω_i. Note that (3) is equivalent to splitting the n parameters across the N workers. Clearly, problem (3) is an instance of (1).

It is interesting to mention that many emerging problems in smart power grid can also be formulated as problem (1); see, for example, the power state estimation problem considered in [27] is solved by employing the distributed ADMM. The energy management problems (i.e., demand response) in [28]–[30] can potentially be handled by the distributed ADMM as well.

B. Distributed ADMM

In this section, we present the distributed ADMM [4], [9] for solving problem (1). Let us consider the following consensus formulation of problem (1)

$$\min_{\substack{\boldsymbol{x}_0,\boldsymbol{x}_i\in\mathbb{R}^n,\\ i=1,\ldots,N}} \quad \sum_{i=1}^N f_i(\boldsymbol{x}_i) + h(\boldsymbol{x}_0) \tag{4a}$$

$$\text{s.t.} \quad \boldsymbol{x}_i = \boldsymbol{x}_0, \quad \forall i = 1, \ldots, N. \tag{4b}$$

In (4), the $N+1$ variables $\boldsymbol{x}_i$, $i = 0, 1, \ldots, N$, are subject to the consensus constraint in (4b), i.e., $\boldsymbol{x}_0 = \boldsymbol{x}_1 = \cdots = \boldsymbol{x}_N$. Thus, problem (4) is equivalent to (1).

It has been shown that such a consensus problem can be efficiently solved by the ADMM [9]. To describe this method, let $\boldsymbol{\lambda} \in \mathbb{R}^n$ denote the Lagrange dual variable associated with constraint (4b) and define the following augmented Lagrangian function

$$\begin{aligned}\mathcal{L}_\rho(\boldsymbol{x}, \boldsymbol{x}_0, \boldsymbol{\lambda}) = &\sum_{i=1}^N f_i(\boldsymbol{x}_i) + h(\boldsymbol{x}_0) \\ &+ \sum_{i=1}^N \boldsymbol{\lambda}_i^T(\boldsymbol{x}_i - \boldsymbol{x}_0) + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i - \boldsymbol{x}_0\|^2,\end{aligned} \tag{5}$$

where $\boldsymbol{x} \triangleq [\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_N^T]^T$, $\boldsymbol{\lambda} \triangleq [\boldsymbol{\lambda}_1^T, \ldots, \boldsymbol{\lambda}_N^T]^T$ and $\rho > 0$ is a penalty parameter. According to [4], the standard synchronous ADMM iteratively updates the primal variables $\boldsymbol{x}_i, i = 0, 1, \ldots, N$, by minimizing (5) in a (one-round) Gauss-Seidel fashion, followed by updating the dual variable $\boldsymbol{\lambda}$ using an approximate gradient ascent method. The ADMM algorithm for solving (4) is presented in Algorithm 1,

Algorithm 1 (Synchronous) Distributed ADMM for (4) [9]

1: **Given** initial variables $\boldsymbol{x}^0$ and $\boldsymbol{\lambda}^0$; set $\boldsymbol{x}_0^0 = \boldsymbol{x}^0$ and $k = 0$.

2: **repeat**

3: **update**

$$\boldsymbol{x}_0^{k+1} = \arg\min_{\boldsymbol{x}_0 \in \mathbb{R}^n} \Big\{ h(\boldsymbol{x}_0) - \boldsymbol{x}_0^T \textstyle\sum_{i=1}^N \boldsymbol{\lambda}_i^k + \frac{\rho}{2} \sum_{i=1}^N \|\boldsymbol{x}_i^k - \boldsymbol{x}_0\|^2 \Big\}, \tag{6}$$

$$\boldsymbol{x}_i^{k+1} = \arg\min_{\boldsymbol{x}_i \in \mathbb{R}^n} \ f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \boldsymbol{\lambda}_i^k + \frac{\rho}{2} \|\boldsymbol{x}_i - \boldsymbol{x}_0^{k+1}\|^2, \quad \forall\, i = 1, \ldots, N, \tag{7}$$

$$\boldsymbol{\lambda}_i^{k+1} = \boldsymbol{\lambda}_i^k + \rho(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1}), \ \forall\, i = 1, \ldots, N. \tag{8}$$

4: **set** $k \leftarrow k + 1$.

5: **until** a predefined stopping criterion is satisfied.

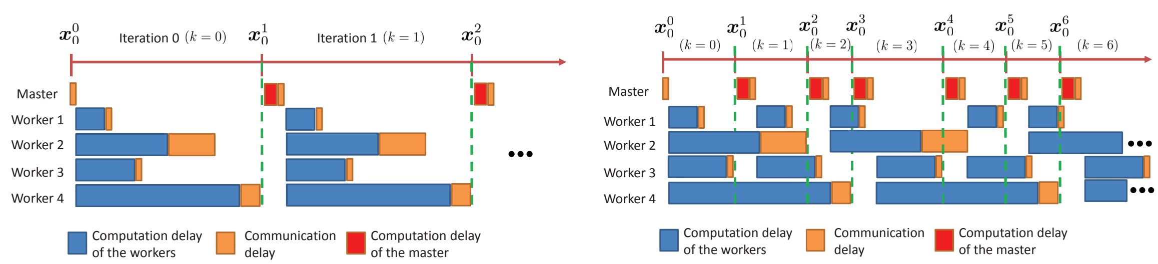
As seen, Algorithm 1 is naturally implementable over the star computer network illustrated in Figure 1. Specifically, the master node takes charge of optimizing $\boldsymbol{x}_0$ by (6), and each worker i is responsible for optimizing $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$ by (7)-(8). Through exchanging the up-to-date $\boldsymbol{x}_0$ and $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$ between the master and the workers, Algorithm 1 solves problem (1) in a fully distributed and parallel manner. Convergence properties of the distributed ADMM have been extensively studied; see, e.g., [9], [18], [31]–[33]. Specifically, [31] shows that the ADMM, under general convex assumptions, has a worst-case $\mathcal{O}(1/k)$ convergence rate; while [32] shows that the ADMM can have a linear convergence rate given strong convexity and smoothness conditions on f_i's. For non-convex and smooth f_i's, the work [18] shows that Algorithm 1 can converge to the set of KKT points with a $\mathcal{O}(1/\sqrt{k})$ rate as long as ρ is large enough.

However, Algorithm 1 is a synchronous algorithm, where the operations of the master and the workers are "locked" with each other. Specifically, to optimize $\boldsymbol{x}_0$ at each iteration, the master has to wait until receiving all the up-to-date variables $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$, $i = 1, \ldots, N$, from the workers. Since the workers may have different computation and communication delays[2], the pace of the optimization would be determined by the "slowest" worker. As an example illustrated in Figure 2(a), the master updates $\boldsymbol{x}_0$ only when it has received the variable information for the four workers at every iteration. As a result, under such synchronous protocol, the master and speedy workers (e.g., workers 1 and 3 in Figure 2) would spend most of the time idling, and thus the parallel computational resources cannot be fully utilized.

III. Asynchronous Distributed ADMM

A. Algorithm Description

In this section, we present an AD-ADMM. The asynchronism we consider is in the same spirit of [10]–[14], [19], [20], where the master does not wait for all the workers. Instead, the master updates $\boldsymbol{x}_0$ whenever it receives $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$ from a partial set of the workers. For example, in Figure 2(b), the master updates $\boldsymbol{x}_0$ whenever it receives the variable information from at least two workers. This implies that none of the workers have to be synchronized with each other and the master does not need to wait for the slowest worker either. As illustrated in Figure 2(b), with the lock removed, both the master and speedy workers can update their variables more frequently.



(a) Synchronous distributed ADMM

(b) Asynchronous distributed ADMM

Fig. 2: Illustration of synchronous and asynchronous distributed ADMM.

Let us denote $k \geq 0$ as the iteration number of the master (i.e., the number of times for which the master updates $\boldsymbol{x}_0$), and denote $\mathcal{A}_k \subseteq \mathcal{V} \triangleq \{1, \ldots, N\}$ as the index subset of workers from which the master receives variable information during iteration k (for example, in Figure 2(b), $\mathcal{A}_0 = \{1, 3\}$ and

[2] In a heterogeneous network, the workers can have different computational powers, or the data sets can be non-uniformly distributed across the network. Thus, the workers can require different computational times in solving the local subproblems. Besides, the communication delays can also be different, e.g., due to probabilistic communication failures and message retransmission.

$\mathcal{A}_1 = \{1, 2\})$[3]. We say that worker i is "arrived" at iteration k if $i \in \mathcal{A}_k$ and "unarrived" otherwise. Clearly, unbounded delay will jeopardize the algorithm convergence. Therefore throughout this paper, we will assume that the asynchronous delay in the network is bounded. In particular, we follow the popular *partially asynchronous* model [4] and assume:

Assumption 1 (Bounded delay) *Let $\tau \geq 1$ be a maximum tolerable delay. For all $i \in \mathcal{V}$ and iteration $k \geq 0$, it must be that $i \in \mathcal{A}_k \cup \mathcal{A}_{k-1} \cdots \cup \mathcal{A}_{\max\{k-\tau+1,-1\}}$.*

Assumption 1 implies that every worker i is arrived at least once within the period $[k - \tau + 1, k]$. In another word, the variable information $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$ used by the master must be at most τ iterations old. To guarantee the bounded delay, at every iteration the master should wait for the workers who have been inactive for $\tau - 1$ iterations, if such workers exist. Note that, when $\tau = 1$, one has $i \in \mathcal{A}_k$ for all $i \in \mathcal{V}$ (i.e., $\mathcal{A}_k = \mathcal{V}$), which corresponds to the synchronous case and the master always waits for all the workers at every iteration.

In Algorithm 2, we present the proposed AD-ADMM, which specifies respectively the steps for the master and the distributed workers. Here, $\mathcal{A}_k^c$ denotes the complementary set of $\mathcal{A}_k$, i.e., $\mathcal{A}_k \cap \mathcal{A}_k^c = \emptyset$ and $\mathcal{A}_k \cup \mathcal{A}_k^c = \mathcal{V}$. Algorithm 2 has five notable differences compared with Algorithm 1. First, the master is required to update $\{(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\}_{i \in \mathcal{V}}$, and such update is only performed for those variables with $i \in \mathcal{A}_k$. Second, $\boldsymbol{x}_0$ is updated by solving a problem with an additional proximal term $\frac{\gamma}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_0^k\|^2$, where $\gamma > 0$ is a penalty parameter (cf. (12)). Adding such proximal term is crucial in making the algorithm well-behaved in the asynchronous setting. As will be seen in the next section, a proper choice of γ guarantees the convergence of Algorithm 2. Third, the variables d_i's are introduced to count the delays of the workers. If worker i is arrived at the current iteration, then d_i is set to zero; otherwise, d_i is increased by one. So, to ensure Assumption 1 hold all the time, in Step 4 of *Algorithm of the Master*, the master waits if there exists at least one worker whose $d_i \geq \tau - 1$. Fourth, in addition to the bounded delay, we assume that the master proceeds to update the variables only if there are at least $A \geq 1$ arrived workers, i.e., $|\mathcal{A}_k| \geq A$ for all k [19]. Note that when $A = N$, the algorithm reduces to the synchronous distributed ADMM. Fifth, in Step 6 of *Algorithm of the Master*, the master sends the up-to-date $\boldsymbol{x}_0$ only to the arrived workers.

We emphasize again that both the master and fast workers in the AD-ADMM can have less idle time and update more frequently than its synchronous counterpart. As illustrated in Figure 2, during the

[3]Without loss of generality, we let $\mathcal{A}_{-1} = \mathcal{V}$, as seen from Figure 2.

same period of time, the synchronous algorithm only completes two updates whereas the asynchronous algorithm updates six times already. On the flip side, the asynchronous algorithm introduces delayed variable information and thereby requires a larger number of iterations to reach the same solution accuracy than its synchronous counterpart. In practice we observe that the benefit of improved update frequency can outweigh the cost of increased number of iterations, and as a result the asynchronous algorithm can still converge faster in time. This is particularly true when the workers have different computation and communication delays and when the computation and communication delays of the master for solving (12) is much shorter than the computation and communication delays of the workers for updating (13) and (14)[4]; e.g., see Figure 2. Detailed numerical results will be reported in Section V of the companion paper [25].

B. Convergence Analysis

In this subsection, we analyze the convergence conditions of Algorithm 2. We first make the following standard assumption on problem (1) (or equivalently problem (4)):

Assumption 2 *Each function f_i is twice differentiable and its gradient ∇f_i is Lipschitz continuous with a Lipschitz constant $L > 0$; the function h is proper convex (lower semi-continuous, but not necessarily smooth) and dom(h) (the domain of h) is compact. Moreover, problem (1) is bounded below, i.e., $F^\star > -\infty$ where $F^\star$ denotes the optimal objective value of problem (1).*

Notably, we do not assume any convexity on f_i's. Indeed, we will show that the AD-ADMM can converge to the set of KKT points even for non-convex f_i's. Our main result is formally stated below.

Theorem 1 *Suppose that Assumption 1 and Assumption 2 hold true. Moreover, assume that there exists a constant $S \in [1, N]$ such that $|\mathcal{A}_k| < S$ for all k and that*

$$\infty > \mathcal{L}_\rho(\boldsymbol{x}^0, \boldsymbol{x}_0^0, \boldsymbol{\lambda}^0) - F^\star \geq 0, \tag{15}$$

$$\rho > \frac{(1 + L + L^2) + \sqrt{(1 + L + L^2)^2 + 8L^2}}{2}, \tag{16}$$

$$\gamma > \frac{S(1 + \rho^2)(\tau - 1)^2 - N\rho}{2}. \tag{17}$$

[4]Note that, for many practical cases (such as $h(\boldsymbol{x}_0) = \|\boldsymbol{x}_0\|_1$) for which (12) has a closed-form solution, the computation delay of the master is negligible. For high-performance computer clusters connected by large-bandwidth fiber links, the communication delays between the master and the workers can also be short. However, for cases in which the computation and communication delays of the master is significant, the AD-ADMM could be less time efficient than the synchronous ADMM due to the increased number of iterations.

Then, $(\{\boldsymbol{x}_i^k\}_{i=1}^N, \boldsymbol{x}_0^k, \{\boldsymbol{\lambda}_i^k\}_{i=1}^N)$ *generated by* (9), (10) *and* (12) *are bounded and have limit points which satisfy KKT conditions of problem* (4).

Theorem 1 implies that the AD-ADMM is guaranteed to converge to the set of KKT points as long as the penalty parameters ρ and γ are sufficiently large. Since $1/\gamma$ can be viewed as the step size of $\boldsymbol{x}_0$, (17) indicates that the master should be more cautious in moving $\boldsymbol{x}_0$ if the network allows a longer delay τ. In particular, the value γ in the worst case should increase with the order of τ^2. When $\tau = 1$ (the synchronous case), $\gamma = -(N\rho)/2 < 0$ and thus the proximal term $\frac{\gamma}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_0^k\|^2$ can be removed from (12). On the other hand, we also see from (17) that γ should increase with N if $\tau > 1$ is fixed[5]. This is because in the worst case the more workers, the more outdated information introduced in the network. Finally, we should mention that a large ρ may be essential for the AD-ADMM to converge properly, especially for non-convex problems, as we demonstrate via simulations in Section V.

Let us compare Theorem 1 with the results in [19], [22]. First, the convergence conditions in [19], [22] are only applicable for convex problems, whereas our results hold for both convex and non-convex problems. Second, [19], [22] have made specific statistical assumptions on the behavior of the workers, and the convergence results presented therein are in an expectation sense. Therefore it is possible, at least theoretically, that a realization of the algorithm fails to converge despite satisfying the conditions given in [19]. On the contrary, our convergence results hold deterministically.

Note that for non-convex f_i's, subproblem (13) is not necessarily convex. However, given $\rho \geq L$ in (16) and twice differentiability of f_i (Assumption 2), subproblem (13) becomes a (strongly) convex problem[6] and hence is globally solvable. When f_i's are all convex functions, Theorem 1 reduces to the following corollary.

Corollary 1 *Assume that f_i's are all convex functions. Under the same premises of Theorem 1, and for γ satisfying* (17) *and*

$$\rho \geq \frac{(1+L^2) + \sqrt{(1+L^2)^2 + 8L^2}}{2}, \tag{18}$$

$(\{\boldsymbol{x}_i^k\}_{i=1}^N, \boldsymbol{x}_0^k, \{\boldsymbol{\lambda}_i^k\}_{i=1}^N)$ *generated by* (9), (10) *and* (12) *are bounded and have limit points which satisfy KKT conditions of problem* (4).

[5]Note that, for a fixed τ, S should increase with N.

[6]By [34, Lemma 1.2.2], the minimum eigenvalue of the Hessian matrix of $f_i(\boldsymbol{x}_i)$ is no smaller than $-L$. Thus, for $\rho > L$, subproblem (13) is a strongly convex problem.

C. Proof of Theorem 1 and Corollary 1

Let us write Algorithm 2 from the master's point of view. Define $\bar{k}_i$ as the last iteration number before iteration k for which worker $i \in \mathcal{A}_k$ is arrived[7], i.e., $i \in \mathcal{A}_{\bar{k}_i}$. Then Algorithm 2 from the master's point of view is as follows: for master iteration $k = 0, 1, \ldots,$

$$\boldsymbol{x}_i^{k+1} = \begin{cases} \arg\min_{\boldsymbol{x}_i} \Big\{ f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \boldsymbol{\lambda}_i^{\bar{k}_i+1} \\ \qquad\qquad + \frac{\rho}{2}\|\boldsymbol{x}_i - \boldsymbol{x}_0^{\bar{k}_i+1}\|^2 \Big\}, & \forall i \in \mathcal{A}_k \\ \boldsymbol{x}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{19}$$

$$\boldsymbol{\lambda}_i^{k+1} = \begin{cases} \boldsymbol{\lambda}_i^{\bar{k}_i+1} + \rho(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{\bar{k}_i+1}) & \forall i \in \mathcal{A}_k \\ \boldsymbol{\lambda}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{20}$$

$$\boldsymbol{x}_0^{k+1} = \arg\min_{\boldsymbol{x}_0 \in \mathbb{R}^n} \Big\{ h(\boldsymbol{x}_0) - \boldsymbol{x}_0^T \textstyle\sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} \\ \qquad + \frac{\rho}{2} \sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0\|^2 + \frac{\gamma}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_0^k\|^2 \Big\}.$$

Now it is relatively easy to see that the master updates $\boldsymbol{x}_0$ using the delayed $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)_{i \in \mathcal{A}_k}$ and the old $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)_{i \in \mathcal{A}_k^c}$. Under Assumption 1, it must hold

$$\max\{k - \tau, -1\} \leq \bar{k}_i < k \quad \forall\, k \geq 0. \tag{21}$$

Moreover, by the definition of $\bar{k}_i$ it holds that $i \notin \mathcal{A}_{k-1} \cup \cdots \cup \mathcal{A}_{\bar{k}_i+1}$, therefore we have that

$$\boldsymbol{\lambda}_i^{\bar{k}_i+1} = \boldsymbol{\lambda}_i^{\bar{k}_i+2} = \cdots = \boldsymbol{\lambda}_i^k, \quad \forall\, i \in \mathcal{A}_k. \tag{22}$$

By applying (22) to (19) and (20) (replacing $\boldsymbol{\lambda}_i^{\bar{k}_i+1}$ with $\boldsymbol{\lambda}_i^k$), we rewrite the master-point-of-view algorithm in Algorithm 3.

Inspired by [18], our analysis for Theorem 1 investigates how the augmented Lagrangian function, i.e.,

$$\mathcal{L}_\rho(\boldsymbol{x}^k, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^k) = \sum_{i=1}^N f_i(\boldsymbol{x}_i^k) + h(\boldsymbol{x}_0^k) + \sum_{i=1}^N (\boldsymbol{\lambda}_i^k)^T(\boldsymbol{x}_i^k - \boldsymbol{x}_0^k) \\ + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i^k - \boldsymbol{x}_0^k\|^2 \tag{26}$$

evolves with the iteration number k, where $\boldsymbol{x}^k \triangleq [(\boldsymbol{x}_1^k)^T, \ldots, (\boldsymbol{x}_N^k)^T]^T$ and $\boldsymbol{\lambda}^k \triangleq [(\boldsymbol{\lambda}_1^k)^T, \ldots, (\boldsymbol{\lambda}_N^k)^T]^T$. The following lemma is one of the keys to prove Theorem 1.

[7]Note that $\bar{k}_i = -1$ for $k = 0$ and $\bar{k}_i \geq -1$ for $k \geq 0$

Lemma 1 *Suppose that Assumption 2 holds and $\rho \geq L$. Then, it holds that*

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) - \mathcal{L}_\rho(\boldsymbol{x}^{k}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k}) \\
&\leq -\frac{2\gamma + N\rho}{2}\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2 \\
&\quad + \left(\frac{1}{\rho} + \frac{1}{2}\right)\sum_{i\in\mathcal{A}_k}\|\boldsymbol{\lambda}_i^{k+1} - \boldsymbol{\lambda}_i^k\|^2 \\
&\quad + \frac{1+\rho^2}{2}\sum_{i\in\mathcal{A}_k}\|\boldsymbol{x}_0^{\bar{k}_i+1} - \boldsymbol{x}_0^k\|^2 \\
&\quad + \frac{(1-\rho)+L}{2}\sum_{i\in\mathcal{A}_k}\|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k\|^2.
\end{aligned} \tag{27}$$

Proof: See Appendix A. ■

Equation (27) shows that $\mathcal{L}_\rho(\boldsymbol{x}^k, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^k)$ is not necessarily decreasing due to the error terms $\sum_{i\in\mathcal{A}_k}\|\boldsymbol{\lambda}_i^{k+1} - \boldsymbol{\lambda}_i^k\|^2$ and $\sum_{i\in\mathcal{A}_k}\|\boldsymbol{x}_0^{\bar{k}_i+1} - \boldsymbol{x}_0^k\|^2$. Next we bound the sizes of these two terms.

First consider $\sum_{i\in\mathcal{A}_k}\|\boldsymbol{\lambda}_i^{k+1} - \boldsymbol{\lambda}_i^k\|^2$. Note from (24) and the optimality condition of (23) that, $\forall\, i \in \mathcal{A}_k$,

$$\begin{aligned}
\mathbf{0} &= \nabla f_i(\boldsymbol{x}_i^{k+1}) + \boldsymbol{\lambda}_i^k + \rho(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{\bar{k}_i+1}) \\
&= \nabla f_i(\boldsymbol{x}_i^{k+1}) + \boldsymbol{\lambda}_i^{k+1}.
\end{aligned} \tag{28}$$

For any $i \in \mathcal{A}_k^c$, denote $\widetilde{k}_i < k$ as the last iteration number for which worker i is arrived. Then, $i \in \mathcal{A}_{\widetilde{k}_i}$ and thus $\nabla f_i(\boldsymbol{x}_i^{\widetilde{k}_i+1}) + \boldsymbol{\lambda}_i^{\widetilde{k}_i+1} = \mathbf{0}$. Since $\boldsymbol{x}_i^{\widetilde{k}_i+1} = \boldsymbol{x}_i^{\widetilde{k}_i+2} = \cdots = \boldsymbol{x}_i^k = \boldsymbol{x}_i^{k+1}$ and $\boldsymbol{\lambda}_i^{\widetilde{k}_i+1} = \boldsymbol{\lambda}_i^{\widetilde{k}_i+2} = \cdots = \boldsymbol{\lambda}_i^k = \boldsymbol{\lambda}_i^{k+1}$, we obtain that $\nabla f_i(\boldsymbol{x}_i^{k+1}) + \boldsymbol{\lambda}_i^{k+1} = \mathbf{0}\ \forall i \in \mathcal{A}_k^c$. Therefore, we conclude that

$$\nabla f_i(\boldsymbol{x}_i^{k+1}) + \boldsymbol{\lambda}_i^{k+1} = \mathbf{0}, \quad \forall\ i \in \mathcal{V} \text{ and } \forall\ k. \tag{29}$$

By (29) and the Lipschitz continuity of ∇f_i (Assumption 2), we can bound

$$\begin{aligned}
\|\boldsymbol{\lambda}_i^{k+1} - \boldsymbol{\lambda}_i^k\|^2 &\leq \|\nabla f_i(\boldsymbol{x}_i^{k+1}) - \nabla f_i(\boldsymbol{x}_i^k)\|^2 \\
&\leq L^2\|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k\|^2, \quad \forall\ i \in \mathcal{V}.
\end{aligned} \tag{30}$$

By applying (30), we can further write (27) as

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) \\
&\leq \mathcal{L}_\rho(\boldsymbol{x}^{k}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k}) + \left(\frac{1+\rho^2}{2}\right)\sum_{i\in\mathcal{A}_k}\|\boldsymbol{x}_0^k - \boldsymbol{x}_0^{\bar{k}_i+1}\|^2 \\
&- \left(\frac{2\gamma + N\rho}{2}\right)\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2 \\
&+ \left(\frac{L + L^2 + (1-\rho)}{2} + \frac{L^2}{\rho}\right)\sum_{i\in\mathcal{A}_k}\|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k\|^2.
\end{aligned} \tag{31}$$

From (31), one can observe that the error term $(\frac{1+\rho^2}{2})\sum_{i\in\mathcal{A}_k}\|\boldsymbol{x}_0^k-\boldsymbol{x}_0^{\bar{k}_i+1}\|^2$ is present due to the asynchrony of the network. The next lemma bounds this error term:

Lemma 2 *Suppose that Assumption 1 holds and assume that* $|\mathcal{A}_k|<S$ *for all* k*, for some constant* $S\in[1,N]$*. Then, it holds that*

$$\sum_{j=0}^{k}\sum_{i\in\mathcal{A}_j}\|\boldsymbol{x}_0^j-\boldsymbol{x}_0^{\bar{j}_i+1}\|^2\leq S(\tau-1)^2\sum_{j=0}^{k-1}\|\boldsymbol{x}_0^{j+1}-\boldsymbol{x}_0^j\|^2. \tag{32}$$

Proof: See Appendix B. ■

The last lemma shows that $\mathcal{L}_\rho(\boldsymbol{x}^k,\boldsymbol{x}_0^k,\boldsymbol{\lambda}^k)$ is bounded below:

Lemma 3 *Under Assumption 2 and for* $\rho\geq L$*, it holds that*

$$\mathcal{L}_\rho(\boldsymbol{x}^{k+1},\boldsymbol{x}_0^{k+1},\boldsymbol{\lambda}^{k+1})\geq F^\star>-\infty. \tag{33}$$

Proof: See Appendix C. ■

Given the three lemmas above, we are ready to prove Theorem 1.

Proof of Theorem 1: Note that any KKT point $(\{\boldsymbol{x}_i^\star\}_{i=1}^N,\boldsymbol{x}_0^\star,\{\boldsymbol{\lambda}_i^\star\}_{i=1}^N)$ of problem (4) satisfies the following conditions

$$\nabla f_i(\boldsymbol{x}_i^\star)+\boldsymbol{\lambda}_i^\star=\mathbf{0},\quad\forall\ i\in\mathcal{V}, \tag{34a}$$

$$\boldsymbol{s}_0^\star-\textstyle\sum_{i=1}^N\boldsymbol{\lambda}_i^\star=\mathbf{0}, \tag{34b}$$

$$\boldsymbol{x}_i^\star=\boldsymbol{x}_0^\star,\quad\forall\ i\in\mathcal{V}, \tag{34c}$$

where $\boldsymbol{s}_0^\star\in\partial h(\boldsymbol{x}_0^\star)$ denotes a subgradient of h at $\boldsymbol{x}_0^\star$ and $\partial h(\boldsymbol{x}_0^\star)$ is the subdifferential of h at $\boldsymbol{x}_0^\star$. Since (34) also implies

$$\sum_{i=1}^N\nabla f_i(\boldsymbol{x}^\star)+\boldsymbol{s}_0^\star=\mathbf{0}, \tag{35}$$

where $\boldsymbol{x}^\star\triangleq\boldsymbol{x}_0^\star=\cdots=\boldsymbol{x}_N^\star$, $\boldsymbol{x}^\star$ is also a stationary point of the original problem (1).

To prove the desired result, we take a telescoping sum of (31), which yields

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) - \mathcal{L}_\rho(\boldsymbol{x}^0, \boldsymbol{x}_0^0, \boldsymbol{\lambda}^0) \\
&\leq \left(\frac{1+\rho^2}{2}\right) \sum_{j=0}^{k} \sum_{i \in \mathcal{A}_j} \|\boldsymbol{x}_0^j - \boldsymbol{x}_0^{\bar{j}_i+1}\|^2 \\
&\quad + \left(\frac{L + L^2 + (1-\rho)}{2} + \frac{L^2}{\rho}\right) \sum_{j=0}^{k} \sum_{i \in \mathcal{A}_j} \|\boldsymbol{x}_i^{j+1} - \boldsymbol{x}_i^j\|^2 \\
&\quad - \left(\frac{2\gamma + N\rho}{2}\right) \sum_{j=0}^{k} \|\boldsymbol{x}_0^{j+1} - \boldsymbol{x}_0^j\|^2.
\end{aligned} \tag{36}$$

By substituting (32) in Lemma 2 into (36), we obtain

$$\begin{aligned}
&\left(\frac{2\gamma + N\rho - S(1+\rho^2)(\tau-1)^2}{2}\right) \sum_{j=0}^{k-1} \|\boldsymbol{x}_0^{j+1} - \boldsymbol{x}_0^j\|^2 \\
&+ \left(\frac{(1-\rho) - (L + L^2)}{2} - \frac{L^2}{\rho}\right) \sum_{j=0}^{k} \sum_{i=1}^{N} \|\boldsymbol{x}_i^{j+1} - \boldsymbol{x}_i^j\|^2 \\
&\leq \mathcal{L}_\rho(\boldsymbol{x}^0, \boldsymbol{x}_0^0, \boldsymbol{\lambda}^0) - \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) \\
&= (\mathcal{L}_\rho(\boldsymbol{x}^0, \boldsymbol{x}_0^0, \boldsymbol{\lambda}^0) - F^\star) - (\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) - F^\star) \\
&\leq \mathcal{L}_\rho(\boldsymbol{x}^0, \boldsymbol{x}_0^0, \boldsymbol{\lambda}^0) - F^\star < \infty,
\end{aligned} \tag{37}$$

where the second inequality is obtained by applying Lemma 3, and the last strict inequality is due to Assumption 2 where the optimal value $F^\star$ is assumed to be lower bounded.

Then, (16) and (17) imply that the left hand side (LHS) of (37) is positive and increasing with k. Since the RHS of (37) is finite, we must have, as $k \to \infty$,

$$\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k \to \mathbf{0}, \quad \boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k \to \mathbf{0}, \quad \forall\ i \in \mathcal{V}. \tag{38}$$

Given (30), (38) infers

$$\boldsymbol{\lambda}_i^{k+1} - \boldsymbol{\lambda}_i^k \to \mathbf{0}, \quad \forall i \in \mathcal{V}. \tag{39}$$

We use (38) and (39) to show that every limit point of $(\{\boldsymbol{x}_i^k\}_{i=1}^N, \boldsymbol{x}_0^k, \{\boldsymbol{\lambda}_i^k\}_{i=1}^N)$ is a KKT point of problem (4). Firstly, by applying (39) to (24) and by (38), one obtains $\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_i^{k+1} \to \mathbf{0}\ \forall i \in \mathcal{A}_k$. For $i \in \mathcal{A}_k^c$, note that $i \in \mathcal{A}_{\widetilde{k}_i}$ (see the definition of $\widetilde{k}_i$ above (29)) and thus, by (24),

$$\boldsymbol{\lambda}_i^{\widetilde{k}_i+1} = \boldsymbol{\lambda}_i^{\widetilde{k}_i} + \rho(\boldsymbol{x}_i^{\widetilde{k}_i+1} - \boldsymbol{x}_0^{\overline{(\widetilde{k}_i)}_i+1}),$$

where $\overline{(\widetilde{k}_i)}_i$ denotes the last iteration number before iteration $\widetilde{k}_i$ for which worker i is arrived. Moreover, since $\boldsymbol{x}_i^{\widetilde{k}_i+1} = \boldsymbol{x}_i^{\widetilde{k}_i+2} = \cdots = \boldsymbol{x}_i^{k} = \boldsymbol{x}_i^{k+1}\ \forall i \in \mathcal{A}_k^c$, and by (24), (38) and (39), we have $\forall i \in \mathcal{A}_k^c$,

$$\begin{aligned}
&\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_i^{k+1}\| = \|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_i^{\widetilde{k}_i+1}\| \\
&= \|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^{\overline{(\widetilde{k}_i)}_i+1} + \boldsymbol{x}_0^{\overline{(\widetilde{k}_i)}_i+1} - \boldsymbol{x}_i^{\widetilde{k}_i+1}\| \\
&\leq \|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^{\overline{(\widetilde{k}_i)}_i+1}\| + \frac{1}{\rho}\|\boldsymbol{\lambda}_i^{\widetilde{k}_i+1} - \boldsymbol{\lambda}_i^{\widetilde{k}_i}\| \\
&\to \mathbf{0}.
\end{aligned} \tag{40}$$

So we conclude

$$\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_i^{k+1} \to \mathbf{0}\ \forall i \in \mathcal{V}. \tag{41}$$

Secondly, the optimality condition of (25) gives

$$\begin{aligned}
&\boldsymbol{s}_0^{k+1} - \sum_{i=1}^{N}\boldsymbol{\lambda}_i^{k+1} - \rho\sum_{i=1}^{N}(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1}) \\
&\qquad + \gamma(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^{k}) = \mathbf{0},
\end{aligned} \tag{42}$$

for some $\boldsymbol{s}_0^{k+1} \in \partial h(\boldsymbol{x}_0^{k+1})$. By applying (41) and (38) to (42), we obtain that

$$\boldsymbol{s}_0^{k+1} - \sum_{i=1}^{N}\boldsymbol{\lambda}_i^{k+1} \to \mathbf{0}. \tag{43}$$

Equations (29), (41) and (43) imply that $(\{\boldsymbol{x}_i^k\}_{i=1}^N, \boldsymbol{x}_0^k, \{\boldsymbol{\lambda}_i^k\}_{i=1}^N)$ asymptotically satisfy the KKT conditions in (34).

Lastly, let us show that $(\{\boldsymbol{x}_i^k\}_{i=1}^N, \boldsymbol{x}_0^k, \{\boldsymbol{\lambda}_i^k\}_{i=1}^N)$ is bounded and has limit points. Since $\text{dom}(h)$ is compact and $\boldsymbol{x}_0^k \in \text{dom}(h)$, $\boldsymbol{x}_0^k$ is a bounded sequence and thus has limit points. From (41), $\boldsymbol{x}_i^k$, $i \in \mathcal{V}$, are bounded and have limit points. Moreover, by (29), $\boldsymbol{\lambda}_i^k$, $i \in \mathcal{V}$, are bounded and have limit points as well. In summary, $(\{\boldsymbol{x}_i^k\}_{i=1}^N, \boldsymbol{x}_0^k, \{\boldsymbol{\lambda}_i^k\}_{i=1}^N)$ converges to the set of KKT points of problem (4) . ■

Proof of Corollary 1: The proof exactly follows that of Theorem 1. The only difference is that the coefficient of the term $\frac{(1-\rho)+L}{2}\sum_{i\in\mathcal{A}_k}\|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k\|^2$ in (27) reduces from $\frac{(1-\rho)+L}{2}$ to $\frac{(1-\rho)}{2}$; see the footnote in Appendix A. ■

IV. Comparison with an Alternative Scheme

In Algorithm 2, the workers compute $(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$, $i \in \mathcal{V}$, and the master is in charge of computing $\boldsymbol{x}_0$. While such distributed implementation is intuitive and natural, one may wonder whether there exist other valid implementations, and if so, how they compare with Algorithm 2. To shed some light on this question, we consider in this section an alternative scheme in Algorithm 4.

Algorithm 4 differs from Algorithm 2 in that the master handles not only the update of $\boldsymbol{x}_0$ but also that of $\{\boldsymbol{\lambda}_i\}_{i\in\mathcal{V}}$; so the workers only updates $\{\boldsymbol{x}_i\}$. In essence, in a synchronous network, Algorithm 2 and Algorithm 4 are equivalent up to a change of update order[8] and have the same convergence conditions. However, intriguingly, in an asynchronous network, the two algorithms may require distinct convergence conditions and behave very differently in practice. To analyze the convergence of Algorithm 4, we make the following assumption.

Assumption 3 *Each function f_i is strongly convex with modulus $\sigma^2 > 0$ and the function h is convex.*

Under the strong convexity assumption, we are able to show the following convergence result for Algorithm 4.

Theorem 2 *Suppose that Assumption 1 and Assumption 3 hold true. Moreover, let $\gamma = 0$ and*

$$0 < \rho \leq \frac{\sigma^2}{(5\tau - 3)\max\{2\tau, 3(\tau-1)\}}, \tag{48}$$

and define $\bar{\boldsymbol{x}}_i^k = \frac{1}{k}\sum_{\ell=1}^k \boldsymbol{x}_i^k$ $\forall i = 0, 1, \ldots, N$, where $(\{\boldsymbol{x}_i^k\}_{i=1}^N, \boldsymbol{x}_0^k)$ are generated by (44) *and* (45)*. Then, it holds that*

$$\left| \left[\sum_{i=1}^N f_i(\bar{\boldsymbol{x}}_i^k) + h(\bar{\boldsymbol{x}}_0^k) \right] - F^\star \right| + \sum_{i=1}^N \|\bar{\boldsymbol{x}}_i^k - \bar{\boldsymbol{x}}_0^k\| \leq \frac{(2+\delta_\lambda)C}{k} \tag{49}$$

for all k, where $C < \infty$ is a finite constant and $\delta_\lambda \triangleq \max\{\|\boldsymbol{\lambda}_1^\star\|, \ldots, \|\boldsymbol{\lambda}_N^\star\|\}$, in which $\{\boldsymbol{\lambda}_i^\star\}$ denote the optimal dual variables of (4)*.*

The proof is presented in Appendix D. Theorem 2 somehow implies that Algorithm 4 may require stronger convergence conditions than Algorithm 2 in the asynchronous network, as f_i's are assumed to be strongly convex. Besides, different from Theorem 1 where ρ is advised to be large for Algorithm 2, Theorem 2 indicates that ρ needs to be small for Algorithm 4. Since ρ is the step size of the dual gradient ascent in (46), (48) implies that the master should move $\boldsymbol{\lambda}_i$'s slowly when τ is large. Such insight is reminiscent of the recent convergence results for multi-block ADMM in [33].

Interestingly and surprisingly, our numerical results to be presented shortly suggest that the strongly convex f_i's and a small ρ are necessary for the convergence of Algorithm 4.

[8]Algorithm 2 under the synchronous protocol is the same as Algorithm 1 with the order of (6) and (7) interchanged.

V. Simulation Results

The main purpose of this section is to examine the convergence behavior of the AD-ADMM with respect to the master's iteration number k. So, the simulation results to be presented are obtained by implementing Algorithm 3 on a desktop computer. First, we present the simulation results of the AD-ADMM for solving the non-convex sparse PCA problem. Second, we consider the LASSO problem and compare Algorithm 4 with Algorithm 2.

A. *Example 1: Sparse PCA*

Theorem 1 has shown that the AD-ADMM can converge for non-convex problems. To verify this point, let us consider the following sparse PCA problem [8]

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \; -\sum_{j=1}^{N} \boldsymbol{w}^T\boldsymbol{B}_j^T\boldsymbol{B}_j\boldsymbol{w} + \theta\|\boldsymbol{w}\|_1, \tag{50}$$

where $\boldsymbol{B}_j \in \mathbb{R}^{m\times n}$, $\forall j = 1,\ldots,N$, and $\theta > 0$ is a regularization parameter. The sparse PCA problem above is not a convex problem. We display in Figure 3 the convergence performance of the AD-ADMM for solving (50). In the simulations, each matrix $\boldsymbol{B}_j \in \mathbb{R}^n$ is a 1000×500 sparse random matrix with approximately 5000 non-zero entries; θ is set to 0.1 and $N = 32$. The penalty parameter ρ is set to $\rho = \beta \max_{j=1,\ldots,N} \lambda_{\max}(\boldsymbol{B}_j^T\boldsymbol{B}_j)$ and $\gamma = 0$. To simulate an asynchronous scenario, at each iteration, half of the workers are assumed to have a probability 0.1 to be arrived independently, and half of the workers are assumed to have a probability 0.8 to be "arrived" independently. At each iteration, the master proceeds to update the variables as long as there is at least one arrived worker, i.e., $A = 1$. The accuracy is defined as

$$\text{accuracy} = \frac{|\mathcal{L}_\rho(\boldsymbol{x}^k, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^k) - \hat{F}|}{\hat{F}} \tag{51}$$

where $\hat{F}$ denotes the optimal objective value for the synchronous case $(\tau = 1)$ which is obtained by running the distributed ADMM (with $\beta = 3$) for 10000 iterations (it is found in the experiments that the AD-ADMM converges to the same KKT point for different values of τ). One can observe from Figure 3 that the AD-ADMM (with $\beta = 3$) indeed converges properly even though (50) is a non-convex problem.

Interestingly, we note that for the example considered here, the AD-ADMM with $\gamma = 0$ works well for different values of τ, even though Theorem 1 suggests that γ should be a larger value in the worst-case. However, we do observe from Figure 3 that if one sets $\beta = 1.5$ (i.e., a smaller value of ρ), then the AD-ADMM diverges even in the synchronous case ($\tau = 1$). This implies that the claim of a large enough ρ is necessary for the non-convex sparse PCA problem.

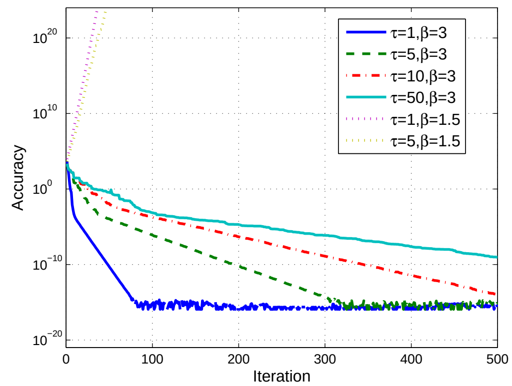


Fig. 3: Convergence curves of the AD-ADMM (Algorithm 2) for solving the sparse PCA problem (50); $N = 32$, $\theta = 0.1$, $\rho = \beta \max_{j=1,\ldots,N} \lambda_{\max}(\boldsymbol{B}_j^T \boldsymbol{B}_j)$ and $\gamma = 0$.

B. Example 2: LASSO

In this example, we compare the convergence performance of Algorithm 4 with Algorithm 2. We consider the following LASSO problem

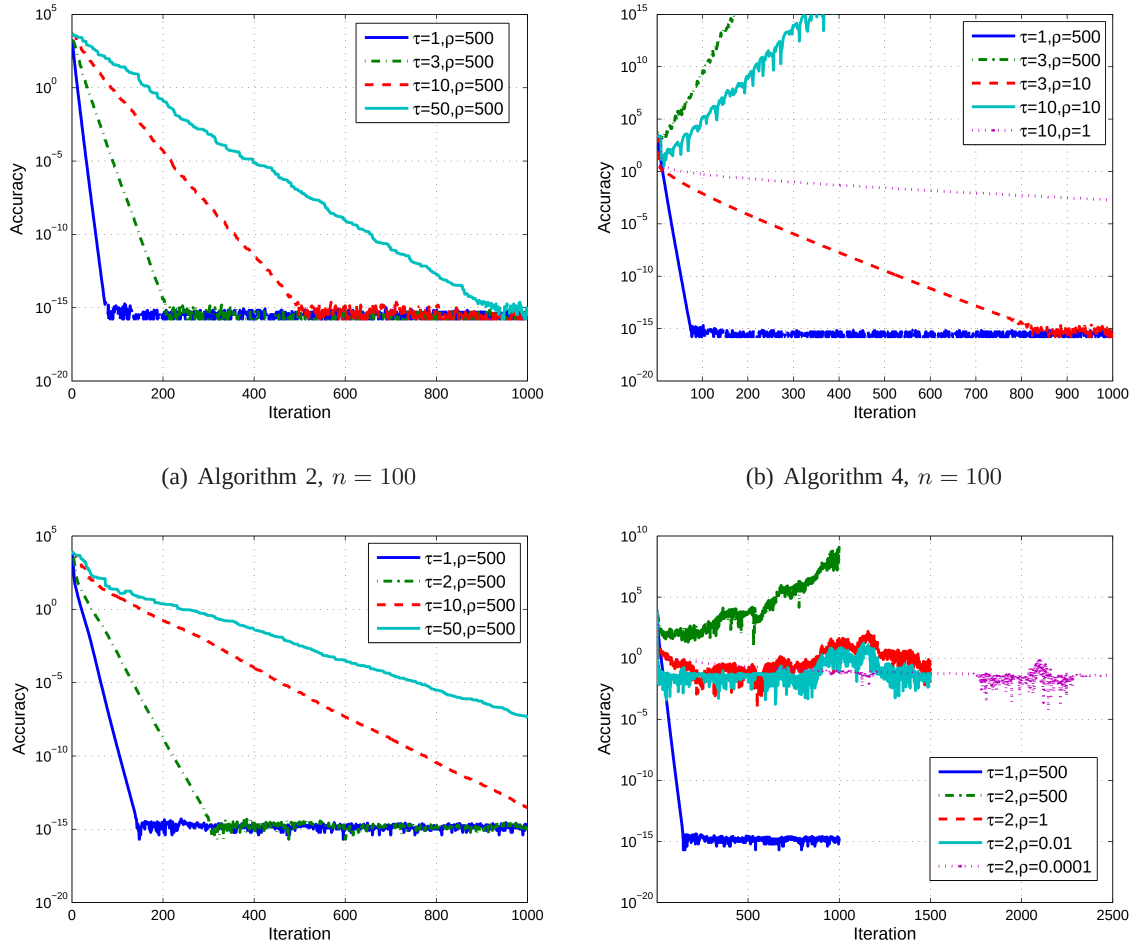
$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \sum_{i=1}^{N} \|\boldsymbol{A}_i\boldsymbol{w} - \boldsymbol{b}_i\|^2 + \theta\|\boldsymbol{w}\|_1, \tag{52}$$

where $\boldsymbol{A}_i \in \mathbb{R}^{m\times n}$, $\boldsymbol{b}_i \in \mathbb{R}^m$, $i = 1, \ldots, N$, and $\theta > 0$. The elements of $\boldsymbol{A}_i$'s are randomly generated following the Gaussian distribution with zero mean and unit variance, i.e., $\sim \mathcal{N}(0, 1)$; each $\boldsymbol{b}_i$ is generated by $\boldsymbol{b}_i = \boldsymbol{A}_i\boldsymbol{w}^0 + \boldsymbol{\nu}_i$ where $\boldsymbol{w}^0 \in \mathbb{R}^n$ is an $n \times 1$ sparse random vector with approximately $0.05n$ non-zero entries and $\boldsymbol{\nu}_i$ is a noise vector with entries following $\mathcal{N}(0, 0.01)$. A star network with 16 ($N = 16$) workers is considered. To simulate an asynchronous scenario, at each iteration, half of the workers are assumed to have a probability 0.1 to be arrived independently, 4 workers are assumed to have a probability 0.3 to be arrived independently, and the remaining 4 workers are assumed to have a probability 0.8 to be arrived independently.

Figure 4(a) and Figure 4(b) respectively display the convergence curves (accuracy versus iteration number) of Algorithm 2 and Algorithm 4 for solving (52) with $N = 16$, $m = 200$, $n = 100$ and $\theta = 0.1$. The accuracy is defined as

$$\text{accuracy} = \frac{|\mathcal{L}_\rho(\boldsymbol{x}^k, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^k) - F^\star|}{F^\star} \tag{53}$$

where $F^\star$ denotes the optimal objective value of problem (52). One can see from Figure 4(a) that Algorithm 2 (with $\rho = 500$, $\gamma = 0$) converges well for various values of delay τ. From Figure 4(b),



(a) Algorithm 2, $n = 100$

(b) Algorithm 4, $n = 100$

(c) Algorithm 2, $n = 1000$

(d) Algorithm 4, $n = 1000$

Fig. 4: Convergence curves of Algorithm 2 and Algorithm 4 for solving the LASSO problem in (52) with $N = 16$, $m = 200$ and $\theta = 0.1$. The parameter γ is set to zero.

one can observe that, under the synchronous setting (i.e., $\tau = 1$), Algorithm 4 (with $\rho = 500$) exhibits a similar behavior as Algorithm 2 in Figure 4(a). However, under the asynchronous setting of $\tau = 3$, Algorithm 4 (with $\rho = 500$) diverges as shown in Figure 4(b); Algorithm 4 can become convergent if one decrease ρ to 10. Analogously, for $\tau = 10$, one has to further reduce ρ to 1 in order to have Algorithm 4 convergent. However, the convergence speed of Algorithm 4 with $\rho = 1$ is much slower when comparing to Algorithm 2 in Figure 4(a).

Figure 4(c) and Figure 4(d) show the comparison results of Algorithm 2 and Algorithm 4 for solving

(52) with n increased to 1000. Note that, given $m = 200$ and $n = 1000$, the cost functions $f_i(\boldsymbol{w}_i) \triangleq \|\boldsymbol{A}_i\boldsymbol{w}_i - \boldsymbol{b}_i\|^2$ in (52) are no longer strongly convex. One can observe from Figure 4(c) that Algorithm 2 (with $\rho = 500$, $\gamma = 0$) still converges properly for various values of τ. However, as one can see from Figure 4(d), Algorithm 4 always diverges for various values of ρ even when the delay τ is as small as two. As a result, the strong convexity assumed in Theorem 2 may also be necessary in practice. We conclude from these simulation results that Algorithm 2 significantly outperforms Algorithm 4 in the asynchronous network, even though the two have the same convergence behaviors in the synchronous network.

VI. Concluding Remarks

In this paper, we have proposed the AD-ADMM (Algorithm 2) aiming at solving large-scale instances of problem (1) over a star computer network. Under the partially asynchronous model, we have shown (in Theorem 1) that the AD-ADMM can deterministically converge to the set of KKT points of problem (4), even in the absence of convexity of f_i's. We have also compared the AD-ADMM (Algorithm 2) with an alternative asynchronous implementation (Algorithm 4), and illustrated the interesting fact that a slight modification of the algorithm can significantly change the algorithm convergence conditions/behaviors in the asynchronous setting.

From the presented simulation results, we have observed that the AD-ADMM may exhibit linear convergence for some structured instances of problem (1). The conditions under which linear convergence can be achieved are presented in the companion paper [25]. Numerical results which demonstrate the time efficiency of the proposed AD-ADMM on a high performance computer cluster are also presented in [25].

Appendix A
Proof of Lemma 1

Notice that

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) - \mathcal{L}_\rho(\boldsymbol{x}^{k}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k}) \\
&= \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) - \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k+1}) \\
&\quad + \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k+1}) - \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k}) \\
&\quad + \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k}) - \mathcal{L}_\rho(\boldsymbol{x}^{k}, \boldsymbol{x}_0^{k}, \boldsymbol{\lambda}^{k}).
\end{aligned} \tag{A.1}$$

We bound the three pairs of the differences on the right hand side (RHS) of (A.1) as follows. Firstly, since $-\boldsymbol{x}_0^T \sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0\|^2 + \frac{\gamma}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_0^k\|^2$ in (25) is strongly convex with respect to (w.r.t.) $\boldsymbol{x}_0$ with modulus $\gamma + N\rho$, by [34, Definition 2.1.2], we have

$$\begin{aligned}
&\Big(-(\boldsymbol{x}_0^k)^T \sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^k\|^2 \Big) \\
&- \Big(-(\boldsymbol{x}_0^{k+1})^T \sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} \\
&\qquad + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1}\|^2 + \frac{\gamma}{2}\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2 \Big) \\
&\geq \Big(-\sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} + \rho \sum_{i=1}^N (\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_i^{k+1}) \\
&+ \gamma(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k) \Big)^T (\boldsymbol{x}_0^k - \boldsymbol{x}_0^{k+1}) + \frac{\gamma + N\rho}{2}\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2. \qquad \text{(A.2)}
\end{aligned}$$

By the optimality condition of (25) and the convexity of h, we respectively have

$$\begin{aligned}
&\Big(\boldsymbol{s}_0^{k+1} - \sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} + \rho \sum_{i=1}^N (\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_i^{k+1}) \\
&\qquad + \gamma(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k) \Big)^T (\boldsymbol{x}_0^k - \boldsymbol{x}_0^{k+1}) \geq 0, \qquad \text{(A.3)}
\end{aligned}$$

$$h(\boldsymbol{x}_0^k) \geq h(\boldsymbol{x}_0^{k+1}) + (\boldsymbol{s}_0^{k+1})^T (\boldsymbol{x}_0^k - \boldsymbol{x}_0^{k+1}). \qquad \text{(A.4)}$$

By subsequently applying (A.3) and (A.4) to (A.2), we obtain

$$\begin{aligned}
&\Big(h(\boldsymbol{x}_0^k) - (\boldsymbol{x}_0^k)^T \sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^k\|^2 \Big) \\
&- \Big(h(\boldsymbol{x}_0^{k+1}) - (\boldsymbol{x}_0^{k+1})^T \sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} \\
&\qquad + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1}\|^2 + \frac{\gamma}{2}\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2 \Big) \\
&\geq \frac{\gamma + N\rho}{2}\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2, \qquad \text{(A.5)}
\end{aligned}$$

that is,

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) - \mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^{k+1}) \\
&\leq -\frac{2\gamma + N\rho}{2}\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2. \qquad \text{(A.6)}
\end{aligned}$$

Secondly, it directly follows from (26) that

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1},\boldsymbol{x}_0^k,\boldsymbol{\lambda}^{k+1})-\mathcal{L}_\rho(\boldsymbol{x}^{k+1},\boldsymbol{x}_0^k,\boldsymbol{\lambda}^{k})\\
&=\sum_{i=1}^N(\boldsymbol{\lambda}_i^{k+1}-\boldsymbol{\lambda}_i^k)^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^k)\\
&=\sum_{i\in\mathcal{A}_k}(\boldsymbol{\lambda}_i^{k+1}-\boldsymbol{\lambda}_i^k)^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{\bar{k}_i+1})\\
&\qquad\qquad+\sum_{i\in\mathcal{A}_k}(\boldsymbol{\lambda}_i^{k+1}-\boldsymbol{\lambda}_i^k)^T(\boldsymbol{x}_0^{\bar{k}_i+1}-\boldsymbol{x}_0^k)\\
&=\frac{1}{\rho}\sum_{i\in\mathcal{A}_k}\|\boldsymbol{\lambda}_i^{k+1}-\boldsymbol{\lambda}_i^k\|^2\\
&\qquad\qquad+\sum_{i\in\mathcal{A}_k}(\boldsymbol{\lambda}_i^{k+1}-\boldsymbol{\lambda}_i^k)^T(\boldsymbol{x}_0^{\bar{k}_i+1}-\boldsymbol{x}_0^k),
\end{aligned}\tag{A.7}$$

where the second equality is due to the fact that $\boldsymbol{\lambda}_i^{k+1}=\boldsymbol{\lambda}_i^k\ \forall i\in\mathcal{A}_k^c$ and the last equality is obtained by applying

$$\boldsymbol{\lambda}_i^{k+1}=\boldsymbol{\lambda}_i^k+\rho(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{\bar{k}_i+1})\ \forall i\in\mathcal{A}_k\tag{A.8}$$

as shown in (24).

Thirdly, define $\mathcal{L}_i(\boldsymbol{x}_i,\boldsymbol{x}_0^k,\boldsymbol{\lambda}^k)=f_i(\boldsymbol{x}_i)+\boldsymbol{x}_i^T\boldsymbol{\lambda}_i^k+\frac{\rho}{2}\|\boldsymbol{x}_i-\boldsymbol{x}_0^k\|^2$ and assume that $\rho\geq L$. Since, by [34, Lemma 1.2.2], the minimum eigenvalue of the Hessian matrix of $f_i(\boldsymbol{x}_i)$ is no smaller than $-L$, $\mathcal{L}_i(\boldsymbol{x}_i,\boldsymbol{x}_0^k,\boldsymbol{\lambda}^k)$ is strongly convex w.r.t. $\boldsymbol{x}_i$ and the convexity parameter is given by $\rho-L\geq 0$ [9]. Therefore, one has

$$\begin{aligned}
\mathcal{L}_i(\boldsymbol{x}_i^k,\boldsymbol{x}_0^k,\boldsymbol{\lambda}^k)&\geq\mathcal{L}_i(\boldsymbol{x}_i^{k+1},\boldsymbol{x}_0^k,\boldsymbol{\lambda}^k)\\
&+(\nabla f_i(\boldsymbol{x}_i^{k+1})+\boldsymbol{\lambda}_i^k+\rho(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^k))^T(\boldsymbol{x}_i^k-\boldsymbol{x}_i^{k+1})\\
&+\frac{\rho-L}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^k\|^2.
\end{aligned}\tag{A.9}$$

Also, by the optimality condition of (23), one has, $\forall i\in\mathcal{A}_k$,

$$\mathbf{0}=\nabla f_i(\boldsymbol{x}_i^{k+1})+\boldsymbol{\lambda}_i^k+\rho(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{\bar{k}_i+1})\tag{A.10}$$

$$\begin{aligned}
&=(\nabla f_i(\boldsymbol{x}_i^{k+1})+\boldsymbol{\lambda}_i^k+\rho(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^k))\\
&\qquad\qquad+\rho(\boldsymbol{x}_0^k-\boldsymbol{x}_0^{\bar{k}_i+1}).
\end{aligned}\tag{A.11}$$

[9]When f_i is a convex function, the minimum eigenvalue of the Hessian matrix of $f_i(\boldsymbol{x}_i)$ is zero. So, the convexity parameter of $\mathcal{L}_i(\boldsymbol{x}_i,\boldsymbol{\lambda}^k,\boldsymbol{x}_0^k)$ is ρ instead.

By substituting (A.11) into (A.9) and by (26), we have

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^k) - \mathcal{L}_\rho(\boldsymbol{x}^k, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^k) \\
&= \sum_{i=1}^{N} (\mathcal{L}_i(\boldsymbol{x}_i^{k+1}, \boldsymbol{\lambda}^k, \boldsymbol{x}_0^k) - \mathcal{L}_i(\boldsymbol{x}_i^k, \boldsymbol{\lambda}^k, \boldsymbol{x}_0^k)) \\
&= \sum_{i \in \mathcal{A}_k} (\mathcal{L}_i(\boldsymbol{x}_i^{k+1}, \boldsymbol{\lambda}^k, \boldsymbol{x}_0^k) - \mathcal{L}_i(\boldsymbol{x}_i^k, \boldsymbol{\lambda}^k, \boldsymbol{x}_0^k)) \\
&\leq -\frac{\rho - L}{2} \sum_{i \in \mathcal{A}_k} \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k\|^2 \\
&\qquad + \rho \sum_{i \in \mathcal{A}_k} (\boldsymbol{x}_0^{\bar{k}_i+1} - \boldsymbol{x}_0^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k),
\end{aligned} \tag{A.12}$$

where the second equality is due to $\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^k \ \forall i \in \mathcal{A}_k^c$ from (23).

After substituting (A.6), (A.7) and (A.12) into (A.1), we obtain

$$\begin{aligned}
&\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) - \mathcal{L}_\rho(\boldsymbol{x}^k, \boldsymbol{x}_0^k, \boldsymbol{\lambda}^k) \\
&\leq -\frac{2\gamma + N\rho}{2} \|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k\|^2 + \frac{1}{\rho} \sum_{i \in \mathcal{A}_k} \|\boldsymbol{\lambda}_i^{k+1} - \boldsymbol{\lambda}_i^k\|^2 \\
&\quad -\frac{\rho - L}{2} \sum_{i \in \mathcal{A}_k} \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k\|^2 \\
&\quad + \sum_{i \in \mathcal{A}_k} (\boldsymbol{\lambda}_i^{k+1} - \boldsymbol{\lambda}_i^k)^T (\boldsymbol{x}_0^{\bar{k}_i+1} - \boldsymbol{x}_0^k) \\
&\quad + \rho \sum_{i \in \mathcal{A}_k} (\boldsymbol{x}_0^{\bar{k}_i+1} - \boldsymbol{x}_0^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^k).
\end{aligned} \tag{A.13}$$

Recall the Young's inequality, i.e.,

$$\boldsymbol{a}^T \boldsymbol{b} \leq \frac{1}{2\delta} \|\boldsymbol{a}\|^2 + \frac{\delta}{2} \|\boldsymbol{b}\|^2, \tag{A.14}$$

for any $\boldsymbol{a}$, $\boldsymbol{b}$ and $\delta > 0$, and apply it to the fourth and fifth terms in the RHS of (A.13) with $\delta = 1$ and $\delta = 1/\rho$ for some $\epsilon > 0$, respectively. Then (27) is obtained. ■

Appendix B

Proof of Lemma 2

It is easy to show that

$$\sum_{j=0}^{k}\sum_{i\in\mathcal{A}_j}\|\boldsymbol{x}_0^{j}-\boldsymbol{x}_0^{\bar{j}_i+1}\|^2=\sum_{j=0}^{k}\sum_{i\in\mathcal{A}_j}\|\sum_{\ell=\bar{j}_i+1}^{j-1}(\boldsymbol{x}_0^{\ell}-\boldsymbol{x}_0^{\ell+1})\|^2$$
$$\leq\sum_{j=0}^{k}\sum_{i\in\mathcal{A}_j}(j-\bar{j}_i-1)\sum_{\ell=\bar{j}_i+1}^{j-1}\|\boldsymbol{x}_0^{\ell}-\boldsymbol{x}_0^{\ell+1}\|^2$$
$$\leq\sum_{j=0}^{k}\sum_{i\in\mathcal{A}_j}(\tau-1)\sum_{\ell=j-\tau+1}^{j-1}\|\boldsymbol{x}_0^{\ell}-\boldsymbol{x}_0^{\ell+1}\|^2$$
$$\leq S(\tau-1)\sum_{j=0}^{k}\sum_{\ell=j-\tau+1}^{j-1}\|\boldsymbol{x}_0^{\ell}-\boldsymbol{x}_0^{\ell+1}\|^2 \tag{A.15}$$

where, in the second inequality, we have applied the fact of $j-\tau\leq\bar{j}_i<j$ from (21); in the last inequality, we have applied the assumption of $|\mathcal{A}_k|<S$ for all k. Notice that, in the summation $\sum_{j=0}^{k}\sum_{\ell=j-\tau+1}^{j-1}\|\boldsymbol{x}_0^{\ell}-\boldsymbol{x}_0^{\ell+1}\|^2$, each $\|\boldsymbol{x}_0^{j}-\boldsymbol{x}_0^{j+1}\|^2$, where $j=0,\ldots,k-1$, appears no more than $\tau-1$ times. Thus, one can upper bound

$$\sum_{j=0}^{k}\sum_{\ell=j-\tau+1}^{j-1}\|\boldsymbol{x}_0^{\ell}-\boldsymbol{x}_0^{\ell+1}\|^2\leq(\tau-1)\sum_{j=0}^{k-1}\|\boldsymbol{x}_0^{j+1}-\boldsymbol{x}_0^{j}\|^2, \tag{A.16}$$

which, combined with (A.15), yields (32). ■

Appendix C

Proof of Lemma 3

The proof is similar to [18, Lemma 2.3]. We present the proof here for completeness. By recalling equation (29) and applying it to (26), one obtains

$$\mathcal{L}_\rho(\boldsymbol{x}^{k+1},\boldsymbol{x}_0^{k+1},\boldsymbol{\lambda}^{k+1})=h(\boldsymbol{x}_0^{k+1})+\sum_{i=1}^{N}f_i(\boldsymbol{x}_i^{k+1})$$
$$-\sum_{i=1}^{N}(\nabla f_i(\boldsymbol{x}_i^{k+1}))^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{k+1})+\frac{\rho}{2}\sum_{i=1}^{N}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{k+1}\|^2. \tag{A.17}$$

As ∇f_i is Lipschitz continuous under Assumption 2, the descent lemma [36, Proposition A.24] holds

$$f_i(\boldsymbol{x}_0^{k+1})\leq f_i(\boldsymbol{x}_i^{k+1})+(\nabla f_i(\boldsymbol{x}_i^{k+1}))^T(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_i^{k+1})$$
$$+\frac{L}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{k+1}\|^2\ \forall\ i=1,\ldots,N. \tag{A.18}$$

By combining (A.17) and (A.18), one can lower bound $\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1})$ as

$$\mathcal{L}_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{x}_0^{k+1}, \boldsymbol{\lambda}^{k+1}) \geq h(\boldsymbol{x}_0^{k+1}) + \sum_{i=1}^N f_i(\boldsymbol{x}_0^{k+1}) + \frac{\rho - L}{2} \sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1}\|^2, \quad \text{(A.19)}$$

which implies (33) given $\rho \geq L$ and under Assumption 2. ■

Appendix D

Proof of Theorem 2

For ease of analysis, we equivalently write Algorithm 4 as follows: For iteration $k = 0, 1, \ldots,$

$$\boldsymbol{x}_i^{k+1} = \begin{cases} \arg\min_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \boldsymbol{\lambda}_i^{\bar{k}_i+1} + \frac{\rho}{2}\|\boldsymbol{x}_i - \boldsymbol{x}_0^{\bar{k}_i+1}\|^2, & \forall i \in \mathcal{A}_k \\ \boldsymbol{x}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \quad \text{(A.20)}$$

$$\boldsymbol{x}_0^{k+1} = \arg\min_{\boldsymbol{x}_0} h(\boldsymbol{x}_0) - \boldsymbol{x}_0^T \textstyle\sum_{i=1}^N \boldsymbol{\lambda}_i^k + \frac{\rho}{2} \sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0\|^2, \quad \text{(A.21)}$$

$$\boldsymbol{\lambda}_i^{k+1} = \boldsymbol{\lambda}_i^k + \rho(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1}) \ \forall i \in \mathcal{V}. \quad \text{(A.22)}$$

Here, $\bar{k}_i$ is the last iteration number for which the master node receives message from worker $i \in \mathcal{A}_k$ before iteration k. For $i \in \mathcal{A}_k^c$, let us denote $\widetilde{k}_i$ ($k - \tau < \widetilde{k}_i < k$) as the last iteration number for which the master node receives message from worker i before iteration k, and further denote $\widehat{k}_i$ ($\widetilde{k}_i - \tau \leq \widehat{k}_i < \widetilde{k}_i$) as the last iteration number for which the master node receives message from worker i before iteration $\widetilde{k}_i$. Then, by (A.20), it must be

$$\boldsymbol{x}_i^{\widetilde{k}_i+1} = \arg\min_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \boldsymbol{\lambda}_i^{\widehat{k}_i+1} + \frac{\rho}{2}\|\boldsymbol{x}_i - \boldsymbol{x}_0^{\widehat{k}_i+1}\|^2 \quad \forall i \in \mathcal{A}_k^c, \quad \text{(A.23)}$$

$$\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^{\widetilde{k}_i+1}, \quad \text{(A.24)}$$

where the second equation is due to $\boldsymbol{x}_i^{\widetilde{k}_i+1} = \boldsymbol{x}_i^{\widetilde{k}_i+2} = \cdots = \boldsymbol{x}_i^k = \boldsymbol{x}_i^{k+1} \ \forall i \in \mathcal{A}_k^c$.

Let us consider the following update steps

$$\boldsymbol{x}_i^{k+1} = \begin{cases} \arg\min_{\boldsymbol{x}_i} \alpha f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1} + \frac{\beta}{2}\|\boldsymbol{x}_i - \boldsymbol{x}_0^{\bar{k}_i+1}\|^2, & \forall i \in \mathcal{A}_k \\ \arg\min_{\boldsymbol{x}_i} \alpha f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} + \frac{\beta}{2}\|\boldsymbol{x}_i - \boldsymbol{x}_0^{\widehat{k}_i+1}\|^2 & \forall i \in \mathcal{A}_k^c \end{cases}, \quad \text{(A.25)}$$

$$\boldsymbol{x}_0^{k+1} = \arg\min_{\boldsymbol{x}_0} \alpha h(\boldsymbol{x}_0) - \boldsymbol{x}_0^T \textstyle\sum_{i=1}^N \widetilde{\boldsymbol{\lambda}}_i^k + \frac{\beta}{2} \sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0\|^2, \quad \text{(A.26)}$$

$$\widetilde{\boldsymbol{\lambda}}_i^{k+1} = \widetilde{\boldsymbol{\lambda}}_i^k + \beta(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1}) \ \forall i \in \mathcal{V}, \quad \text{(A.27)}$$

where $\alpha, \beta > 0$. One can verify that (A.25)-(A.27) are equivalent to (A.20)-(A.22) and (A.23)-(A.24) if one considers the change of variables $\boldsymbol{\lambda}_i = \widetilde{\boldsymbol{\lambda}}_i / \alpha$ and $\rho = \beta / \alpha$.

We first consider the optimality condition of (A.25) for $i \in \mathcal{A}_k$:

$$
\begin{aligned}
0 &\geq \alpha \partial f_i(\boldsymbol{x}_i^{k+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + (\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1} + \beta(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{\bar{k}_i+1}))^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&= \alpha \partial f_i(\boldsymbol{x}_i^{k+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + (\widetilde{\boldsymbol{\lambda}}_i^{k+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&\quad + (\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i^{k})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + \beta(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{\bar{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star),
\end{aligned} \tag{A.28}
$$

where we have applied (A.27) to obtain the equality. Since, under Assumption 3, f_i is strongly convex, one has

$$
\alpha f_i(\boldsymbol{x}_i^\star) \geq \alpha f_i(\boldsymbol{x}_i^{k+1}) + \alpha \partial f_i(\boldsymbol{x}_i^{k+1})^T(\boldsymbol{x}_i^\star - \boldsymbol{x}_i^{k+1}) + \frac{\alpha\sigma^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2. \tag{A.29}
$$

Combining (A.28) and (A.29) gives rise to

$$
\begin{aligned}
&\alpha f_i(\boldsymbol{x}_i^{k+1}) - \alpha f_i(\boldsymbol{x}_i^\star) + \widetilde{\boldsymbol{\lambda}}_i^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + \frac{\alpha\sigma^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2 \\
&\quad + (\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i)^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + (\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i^{k})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&\quad + \beta(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{\bar{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \leq 0 \quad \forall i \in \mathcal{A}_k.
\end{aligned} \tag{A.30}
$$

On the other hand, consider the optimality condition of (A.25) for $i \in \mathcal{A}_k^c$:

$$
\begin{aligned}
0 &\geq \alpha \nabla f_i(\boldsymbol{x}_i^{k+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + (\widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} + \beta(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{\widehat{k}_i+1}))^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&= \alpha \nabla f_i(\boldsymbol{x}_i^{k+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&\quad + (\widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} + \widetilde{\boldsymbol{\lambda}}_i^{\widetilde{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i^{\widetilde{k}_i} - \beta(\boldsymbol{x}_i^{\widetilde{k}_i+1}-\boldsymbol{x}_0^{\widetilde{k}_i+1}) + \beta(\boldsymbol{x}_i^{\widetilde{k}_i+1}-\boldsymbol{x}_0^{\widehat{k}_i+1}))^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&= \alpha \nabla f_i(\boldsymbol{x}_i^{k+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + (\widetilde{\boldsymbol{\lambda}}_i^{\widetilde{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&\quad + (\widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} - \widetilde{\boldsymbol{\lambda}}^{\widetilde{k}_i})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + \beta(\boldsymbol{x}_0^{\widetilde{k}_i+1}-\boldsymbol{x}_0^{\widehat{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star),
\end{aligned} \tag{A.31}
$$

where (A.27) with $k = \widetilde{k}_i$ and (A.24) are used to obtain the first equality. By combining (A.29) with (A.31), one obtains

$$
\begin{aligned}
&\alpha f_i(\boldsymbol{x}_i^{k+1}) - \alpha f_i(\boldsymbol{x}_i^\star) + \widetilde{\boldsymbol{\lambda}}_i^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + \frac{\alpha\sigma^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2 \\
&\quad + (\widetilde{\boldsymbol{\lambda}}_i^{\widetilde{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i)^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) + (\widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} - \widetilde{\boldsymbol{\lambda}}^{\widetilde{k}_i})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \\
&\quad + \beta(\boldsymbol{x}_0^{\widetilde{k}_i+1}-\boldsymbol{x}_0^{\widehat{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star) \leq 0 \quad \forall i \in \mathcal{A}_k^c.
\end{aligned} \tag{A.32}
$$

By summing (A.30) for all $i \in \mathcal{A}_k$ and (A.32) for all $i \in \mathcal{A}_k^c$ and further summing the resultant two terms, we obtain that

$$\alpha \sum_{i=1}^{N} f_i(\boldsymbol{x}_i^{k+1}) - \alpha \sum_{i=1}^{N} f_i(\boldsymbol{x}_i^\star) + \sum_{i=1}^{N} \widetilde{\boldsymbol{\lambda}}_i^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i=1}^{N} \frac{\alpha\sigma^2}{2} \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star\|^2$$
$$+ \underbrace{\sum_{i\in\mathcal{A}_k} (\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i\in\mathcal{A}_k^c} (\widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)}_{\text{(a)}}$$
$$+ \sum_{i\in\mathcal{A}_k} (\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i\in\mathcal{A}_k^c} (\widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} - \widetilde{\boldsymbol{\lambda}}^{\widetilde{k}_i})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)$$
$$+ \underbrace{\sum_{i\in\mathcal{A}_k} \beta(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^{\bar{k}_i+1})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i\in\mathcal{A}_k^c} \beta(\boldsymbol{x}_0^{\widetilde{k}_i+1} - \boldsymbol{x}_0^{\widehat{k}_i+1})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)}_{\text{(b)}} \leq 0. \tag{A.33}$$

The term (a) in (A.33), after adding and subtracting $\sum_{i\in\mathcal{A}_k^c} (\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)$, can be written as

$$\text{(a)} = \sum_{i=1}^{N} (\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i\in\mathcal{A}_k^c} (\widetilde{\boldsymbol{\lambda}}_i^{\widetilde{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i^{k+1})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star). \tag{A.34}$$

The term (b) in (A.33) can be expressed as

$$\text{(b)} = \sum_{i\in\mathcal{A}_k} \beta(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k + \boldsymbol{x}_0^k - \boldsymbol{x}_0^{\bar{k}_i+1})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i\in\mathcal{A}_k^c} \beta(\boldsymbol{x}_0^{\widetilde{k}_i+1} - \boldsymbol{x}_0^{\widehat{k}_i+1})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)$$
$$= \sum_{i=1}^{N} \beta(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i\in\mathcal{A}_k^c} \beta(\boldsymbol{x}_0^{\widetilde{k}_i+1} - \boldsymbol{x}_0^{\widehat{k}_i+1} - \boldsymbol{x}_0^{k+1} + \boldsymbol{x}_0^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)$$
$$+ \sum_{i\in\mathcal{A}_k} \beta(\boldsymbol{x}_0^k - \boldsymbol{x}_0^{\bar{k}_i+1})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star). \tag{A.35}$$

Note that, by applying (A.27) and the fact of $\boldsymbol{x}_i^\star = \boldsymbol{x}_0^\star \; \forall i \in \mathcal{V}$, one can write

$$\sum_{i=1}^{N} \beta(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) = \sum_{i=1}^{N} \beta(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1} + \boldsymbol{x}_0^{k+1} - \boldsymbol{x}_i^\star)$$
$$= \sum_{i=1}^{N} (\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k)^T (\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i^k) + N\beta(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k)^T (\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^\star). \tag{A.36}$$

So, The term (b) in (A.35) is given by

$$\text{(b)} = \sum_{i=1}^{N} (\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k)^T (\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i^k) + N\beta(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^k)^T (\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^\star)$$
$$+ \sum_{i\in\mathcal{A}_k^c} \beta(\boldsymbol{x}_0^{\widetilde{k}_i+1} - \boldsymbol{x}_0^{\widehat{k}_i+1} - \boldsymbol{x}_0^{k+1} + \boldsymbol{x}_0^k)^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) + \sum_{i\in\mathcal{A}_k} \beta(\boldsymbol{x}_0^k - \boldsymbol{x}_0^{\bar{k}_i+1})^T (\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star). \tag{A.37}$$

It can be shown that

$$\sum_{i=1}^{N}(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{k})^T(\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i^{k})\geq 0. \tag{A.38}$$

To see this, consider the optimality condition of (A.26): $\forall \boldsymbol{x}_0\in\mathbb{R}^n$,

$$\begin{aligned}0&\geq \alpha h(\boldsymbol{x}_0^{k+1})-\alpha h(\boldsymbol{x}_0)-\sum_{i=1}^{N}(\widetilde{\boldsymbol{\lambda}}_i^{k}+\beta(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{k+1}))^T(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0)\\&=\alpha h(\boldsymbol{x}_0^{k+1})-\alpha h(\boldsymbol{x}_0)-\sum_{i=1}^{N}(\widetilde{\boldsymbol{\lambda}}_i^{k+1})^T(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0),\end{aligned} \tag{A.39}$$

where the equality is due to (A.27). By letting $\boldsymbol{x}_0=\boldsymbol{x}_0^k$ in (A.39) and also considering (A.39) for iteration k and $\boldsymbol{x}_0=\boldsymbol{x}_0^{k+1}$, we have

$$\begin{aligned}0&\geq \alpha h(\boldsymbol{x}_0^{k+1})-\alpha h(\boldsymbol{x}_0^{k})-\sum_{i=1}^{N}(\widetilde{\boldsymbol{\lambda}}_i^{k+1})^T(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{k}),\\0&\geq \alpha h(\boldsymbol{x}_0^{k})-\alpha h(\boldsymbol{x}_0^{k+1})-\sum_{i=1}^{N}(\widetilde{\boldsymbol{\lambda}}_i^{k})^T(\boldsymbol{x}_0^{k}-\boldsymbol{x}_0^{k+1}),\end{aligned} \tag{A.40}$$

respectively. By summing the above two equations, we obtain (A.38). Moreover, by letting $\boldsymbol{x}_0=\boldsymbol{x}_i^\star=\boldsymbol{x}_0^\star$ in (A.39), we have

$$\alpha h(\boldsymbol{x}_0^{k+1})-\alpha h(\boldsymbol{x}_0^{\star})-\sum_{i=1}^{N}\widetilde{\boldsymbol{\lambda}}_i^{T}(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_i^{\star})-\sum_{i=1}^{N}(\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i)^T(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_i^{\star})\leq 0. \tag{A.41}$$

By summing (A.41) and (A.33) followed by applying (A.34), (A.37) and (A.38), one obtains

$$\begin{aligned}&\alpha\sum_{i=1}^{N}f_i(\boldsymbol{x}_i^{k+1})+\alpha h(\boldsymbol{x}_0^{k+1})-\alpha\sum_{i=1}^{N}f_i(\boldsymbol{x}_i^{\star})-\alpha h(\boldsymbol{x}_0^{\star})+\sum_{i=1}^{N}\widetilde{\boldsymbol{\lambda}}_i^{T}(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{k+1})+\sum_{i=1}^{N}\frac{\alpha\sigma^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^{\star}\|^2\\&\quad+\frac{1}{\beta}\sum_{i=1}^{N}(\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i)^T(\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i^{k})+N\beta(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{k})^T(\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{\star})\\&\quad\quad+\sum_{i\in\mathcal{A}_k^c}(\widetilde{\boldsymbol{\lambda}}_i^{\widetilde{k}_i+1}-\widetilde{\boldsymbol{\lambda}}_i^{k+1}+\widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1}-\widetilde{\boldsymbol{\lambda}}^{\widetilde{k}_i})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^{\star})+\sum_{i\in\mathcal{A}_k}(\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1}-\widetilde{\boldsymbol{\lambda}}_i^{k})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^{\star})\\&\quad\quad+\sum_{i\in\mathcal{A}_k^c}\beta(\boldsymbol{x}_0^{\widetilde{k}_i+1}-\boldsymbol{x}_0^{\widehat{k}_i+1}-\boldsymbol{x}_0^{k+1}+\boldsymbol{x}_0^{k})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^{\star})+\sum_{i\in\mathcal{A}_k}\beta(\boldsymbol{x}_0^{k}-\boldsymbol{x}_0^{\bar{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^{\star})\leq 0,\end{aligned} \tag{A.42}$$

where the seventh term in the LHS is obtained by applying (A.27).

We sum (A.42) for $k = 0, \ldots, K-1$ and take the average, which yields

$$\frac{\alpha}{K}\sum_{k=0}^{K-1}\left[\sum_{i=1}^{N} f_i(\boldsymbol{x}_i^{k+1}) + h(\boldsymbol{x}_0^{k+1})\right] - \alpha\left[\sum_{i=1}^{N} f_i(\boldsymbol{x}_i^\star) + h(\boldsymbol{x}_0^\star)\right] + \frac{1}{K}\sum_{k=0}^{K-1}\sum_{i=1}^{N}\widetilde{\boldsymbol{\lambda}}_i^T(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{k+1})$$

$$+ \frac{1}{\beta K}\underbrace{\sum_{k=0}^{K-1}\sum_{i=1}^{N}(\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i)^T(\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i^{k})}_{\text{(a)}} + \frac{N\beta}{K}\underbrace{\sum_{k=0}^{K-1}(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^{k})^T(\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^\star)}_{\text{(b)}}$$

$$\le -\frac{1}{K}\sum_{k=0}^{K-1}\sum_{i=1}^{N}\frac{\alpha\sigma^2}{2}\|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star\|^2$$

$$+ \underbrace{\frac{1}{K}\sum_{k=0}^{K-1}\left(-\sum_{i\in\mathcal{A}_k^c}(\widetilde{\boldsymbol{\lambda}}_i^{\widetilde{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i^{k+1} + \widetilde{\boldsymbol{\lambda}}_i^{\widehat{k}_i+1} - \widetilde{\boldsymbol{\lambda}}^{\widetilde{k}_i})^T(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) - \sum_{i\in\mathcal{A}_k}(\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1} - \widetilde{\boldsymbol{\lambda}}_i^{k})^T(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)\right)}_{\text{(c)}}$$

$$+ \underbrace{\frac{1}{K}\sum_{k=0}^{K-1}\left(-\sum_{i\in\mathcal{A}_k^c}\beta(\boldsymbol{x}_0^{\widetilde{k}_i+1} - \boldsymbol{x}_0^{\widehat{k}_i+1} - \boldsymbol{x}_0^{k+1} + \boldsymbol{x}_0^{k})^T(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star) - \sum_{i\in\mathcal{A}_k}\beta(\boldsymbol{x}_0^{k} - \boldsymbol{x}_0^{\bar{k}_i+1})^T(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_i^\star)\right)}_{\text{(d)}}. \tag{A.43}$$

It is easy to see that term (a)

$$\text{(a)} = \frac{1}{2}\sum_{k=0}^{K-1}\left(\|\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i\|^2 - \|\widetilde{\boldsymbol{\lambda}}_i^{k} - \widetilde{\boldsymbol{\lambda}}_i\|^2 + \|\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i^{k}\|^2\right)$$

$$= \frac{1}{2}\|\widetilde{\boldsymbol{\lambda}}_i^{K} - \widetilde{\boldsymbol{\lambda}}_i\|^2 - \frac{1}{2}\|\widetilde{\boldsymbol{\lambda}}_i^{0} - \widetilde{\boldsymbol{\lambda}}_i\|^2 + \frac{1}{2}\sum_{k=0}^{K-1}\|\widetilde{\boldsymbol{\lambda}}_i^{k+1} - \widetilde{\boldsymbol{\lambda}}_i^{k}\|^2, \tag{A.44}$$

and similarly, term (b)

$$\text{(b)} = \frac{1}{2}\sum_{k=0}^{K-1}\left(\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^\star\|^2 - \|\boldsymbol{x}_0^{k} - \boldsymbol{x}_0^\star\|^2 + \|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^{k}\|^2\right)$$

$$= \frac{1}{2}\|\boldsymbol{x}_0^{K} - \boldsymbol{x}_0^\star\|^2 - \frac{1}{2}\|\boldsymbol{x}_0^{0} - \boldsymbol{x}_0^\star\|^2 + \frac{1}{2}\sum_{k=0}^{K-1}\|\boldsymbol{x}_0^{k+1} - \boldsymbol{x}_0^{k}\|^2. \tag{A.45}$$

Notice that one can bound the term $\sum_{k=0}^{K-1}\sum_{i\in\mathcal{A}_k}(\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1}-\widetilde{\boldsymbol{\lambda}}_i^{k})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star)$ in (c) as follows

$$\sum_{k=0}^{K-1}\sum_{i\in\mathcal{A}_k}(\widetilde{\boldsymbol{\lambda}}_i^{\bar{k}_i+1}-\widetilde{\boldsymbol{\lambda}}_i^{k})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star)=\sum_{k=0}^{K-1}\sum_{i\in\mathcal{A}_k}\sum_{\ell=\bar{k}_i+1}^{k-1}(\widetilde{\boldsymbol{\lambda}}_i^{\ell}-\widetilde{\boldsymbol{\lambda}}_i^{\ell+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star)$$

$$\leq\sum_{k=0}^{K-1}\sum_{i\in\mathcal{A}_k}\sum_{\ell=k-\tau+1}^{k-1}\|\widetilde{\boldsymbol{\lambda}}_i^{\ell}-\widetilde{\boldsymbol{\lambda}}_i^{\ell+1}\|\cdot\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|$$

$$\leq\sum_{i=1}^{N}\sum_{k=0}^{K-1}\sum_{\ell=k-\tau+1}^{k-1}\left(\frac{1}{2\beta^2}\|\widetilde{\boldsymbol{\lambda}}_i^{\ell}-\widetilde{\boldsymbol{\lambda}}_i^{\ell+1}\|^2+\frac{\beta^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2\right) \tag{A.46}$$

$$\leq\sum_{i=1}^{N}\sum_{k=0}^{K-1}\left(\frac{\tau-1}{2\beta^2}\|\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i^{k}\|^2+\frac{(\tau-1)\beta^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2\right), \tag{A.47}$$

where the second inequality is obtained by applying the Young's inequality:

$$\boldsymbol{a}^T\boldsymbol{b}\leq\frac{1}{2\delta}\|\boldsymbol{a}\|^2+\frac{\delta}{2}\|\boldsymbol{b}\|^2 \tag{A.48}$$

for any $\boldsymbol{a}$, $\boldsymbol{b}$ and $\delta>0$; the last inequality is caused by the fact that the term $\|\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i^{k}\|^2$ for each k does not appear more than $\tau-1$ times in the RHS of (A.46). By applying a similar idea to the first term of (c) and by (A.47), one eventually can bound (c) as follows

$$\text{(c)}\leq\frac{3(\tau-1)}{2\beta^2}\sum_{i=1}^{N}\sum_{k=0}^{K-1}\|\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i^{k}\|^2+\frac{3(\tau-1)\beta^2}{2}\sum_{i=1}^{N}\sum_{k=0}^{K-1}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2. \tag{A.49}$$

Similarly, the term $\sum_{k=0}^{K-1}\sum_{i\in\mathcal{A}_k}\beta(\boldsymbol{x}_0^{k}-\boldsymbol{x}_0^{\bar{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star)$ in (d) can be upper bounded as follows

$$\sum_{k=0}^{K-1}\sum_{i\in\mathcal{A}_k}\beta(\boldsymbol{x}_0^{k}-\boldsymbol{x}_0^{\bar{k}_i+1})^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star)\leq\sum_{k=0}^{K-1}\sum_{i\in\mathcal{A}_k}\sum_{\ell=k-\tau+1}^{k-1}\beta\|\boldsymbol{x}_0^{k}-\boldsymbol{x}_0^{\bar{k}_i+1}\|\cdot\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|$$

$$\leq\sum_{i=1}^{N}\sum_{k=0}^{K-1}\sum_{\ell=k-\tau+1}^{k-1}\left(\frac{1}{2}\|\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{k}\|^2+\frac{\beta^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2\right) \tag{A.50}$$

$$\leq\sum_{i=1}^{N}\sum_{k=0}^{K-1}\left(\frac{\tau-1}{2}\|\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{k}\|^2+\frac{(\tau-1)\beta^2}{2}\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2\right). \tag{A.51}$$

By applying a similar idea to the first term of (d) and by (A.51), one can bound (d) as follows

$$\text{(d)}\leq\sum_{i=1}^{N}\sum_{k=0}^{K-1}\left(\tau\|\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^{k}\|^2+\tau\beta^2\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2\right). \tag{A.52}$$

After substituting (A.44), (A.45), (A.49) and (A.52) into (A.43), we obtain that

$$\begin{aligned}
&\alpha\Bigg[\sum_{i=1}^N f_i(\bar{\boldsymbol{x}}_i^K)+h(\bar{\boldsymbol{x}}_0^K)\Bigg]-\alpha\Bigg[\sum_{i=1}^N f_i(\boldsymbol{x}_i^\star)+h(\boldsymbol{x}_0^\star)\Bigg]+\sum_{i=1}^N \widetilde{\boldsymbol{\lambda}}_i^T(\bar{\boldsymbol{x}}_i^K-\bar{\boldsymbol{x}}_0^K)\\
&\le \frac{\alpha}{K}\sum_{k=0}^{K-1}\Bigg[\sum_{i=1}^N f_i(\boldsymbol{x}_i^{k+1})+h(\boldsymbol{x}_0^{k+1})\Bigg]-\alpha\Bigg[\sum_{i=1}^N f_i(\boldsymbol{x}_i^\star)+h(\boldsymbol{x}_0^\star)\Bigg]+\frac{1}{K}\sum_{k=0}^{K-1}\sum_{i=1}^N \widetilde{\boldsymbol{\lambda}}_i^T(\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_0^{k+1})\\
&\le \frac{1}{2\beta K}\sum_{i=1}^N\|\widetilde{\boldsymbol{\lambda}}_i^0-\widetilde{\boldsymbol{\lambda}}_i\|^2-\frac{1}{2\beta K}\sum_{i=1}^N\|\widetilde{\boldsymbol{\lambda}}_i^K-\widetilde{\boldsymbol{\lambda}}_i\|^2+\frac{N\beta}{2K}\|\boldsymbol{x}_0^0-\boldsymbol{x}_0^\star\|^2-\frac{N\beta}{2K}\|\boldsymbol{x}_0^K-\boldsymbol{x}_0^\star\|^2\\
&+\left(\frac{3(\tau-1)}{2K\beta^2}-\frac{1}{2\beta K}\right)\sum_{i=1}^N\sum_{k=0}^{K-1}\|\widetilde{\boldsymbol{\lambda}}_i^{k+1}-\widetilde{\boldsymbol{\lambda}}_i^k\|^2+\left(\frac{N\tau}{K}-\frac{N\beta}{2K}\right)\sum_{k=0}^{K-1}\|\boldsymbol{x}_0^{k+1}-\boldsymbol{x}_0^k\|^2\\
&\qquad+\frac{1}{K}\sum_{k=0}^{K-1}\sum_{i=1}^N\left(\frac{3(\tau-1)\beta^2+2\tau\beta^2-\alpha\sigma^2}{2}\right)\|\boldsymbol{x}_i^{k+1}-\boldsymbol{x}_i^\star\|^2 \qquad \text{(A.53)}
\end{aligned}$$

where the first inequality is by the convexity of f_i's and h.

According to (A.53), by choosing

$$\beta\ge\max\{2\tau,3(\tau-1)\},\quad \alpha\ge\frac{(5\tau-3)\beta^2}{\sigma^2}, \qquad \text{(A.54)}$$

and recalling that $\boldsymbol{\lambda}_i=\widetilde{\boldsymbol{\lambda}}_i/\alpha$ and $\rho=\beta/\alpha$, one can obtain

$$\begin{aligned}
&\Bigg[\sum_{i=1}^N f_i(\bar{\boldsymbol{x}}_i^K)+h(\bar{\boldsymbol{x}}_0^K)\Bigg]-\Bigg[\sum_{i=1}^N f_i(\boldsymbol{x}_i^\star)+h(\boldsymbol{x}_0^\star)\Bigg]+\sum_{i=1}^N \boldsymbol{\lambda}_i^T(\bar{\boldsymbol{x}}_i^K-\bar{\boldsymbol{x}}_0^K)\\
&\le\frac{1}{2\rho K}\sum_{i=1}^N\|\boldsymbol{\lambda}_i^0-\boldsymbol{\lambda}_i\|^2+\frac{N\rho}{2K}\|\boldsymbol{x}_0^0-\boldsymbol{x}_0^\star\|^2. \qquad \text{(A.55)}
\end{aligned}$$

Note that (A.54) is equivalent to

$$\rho=\beta/\alpha\le\frac{\sigma^2}{(5\tau-3)\beta}\le\frac{\sigma^2}{(5\tau-3)\max\{2\tau,3(\tau-1)\}}. \qquad \text{(A.56)}$$

Now, let $\boldsymbol{\lambda}_i=\boldsymbol{\lambda}_i^\star+\frac{\bar{\boldsymbol{x}}_i^K-\bar{\boldsymbol{x}}_0^K}{\|\bar{\boldsymbol{x}}_i^K-\bar{\boldsymbol{x}}_0^K\|}$ $\forall i\in\mathcal{V}$ in (A.57), and note that, by the duality theory [37],

$$\Bigg[\sum_{i=1}^N f_i(\bar{\boldsymbol{x}}_i^K)+h(\bar{\boldsymbol{x}}_0^K)\Bigg]-\Bigg[\sum_{i=1}^N f_i(\boldsymbol{x}_i^\star)+h(\boldsymbol{x}_0^\star)\Bigg]+\sum_{i=1}^N(\boldsymbol{\lambda}_i^\star)^T(\bar{\boldsymbol{x}}_i^K-\bar{\boldsymbol{x}}_0^K)\ge 0.$$

Thus, we obtain that

$$\sum_{i=1}^N\|\bar{\boldsymbol{x}}_i^K-\bar{\boldsymbol{x}}_0^K\|\le\frac{1}{K}\left[\frac{1}{2\rho}\max_{\|\boldsymbol{a}\|\le 1}\left\{\sum_{i=1}^N\|\boldsymbol{\lambda}_i^0-\boldsymbol{\lambda}_i^\star+\boldsymbol{a}\|^2\right\}+\frac{N\rho}{2}\|\boldsymbol{x}_0^0-\boldsymbol{x}_0^\star\|^2\right]\triangleq\frac{C_1}{K}. \qquad \text{(A.57)}$$

On the other hand, let $\boldsymbol{\lambda}_i = \boldsymbol{\lambda}_i^\star$ in (A.57), and note that,

$$\left[\sum_{i=1}^N f_i(\bar{\boldsymbol{x}}_i^K) + h(\bar{\boldsymbol{x}}_0^K)\right] - \left[\sum_{i=1}^N f_i(\boldsymbol{x}_i^\star) + h(\boldsymbol{x}_0^\star)\right] + \sum_{i=1}^N (\boldsymbol{\lambda}_i^\star)^T(\bar{\boldsymbol{x}}_i^K - \bar{\boldsymbol{x}}_0^K)$$

$$\geq \left|\left[\sum_{i=1}^N f_i(\bar{\boldsymbol{x}}_i^K) + h(\bar{\boldsymbol{x}}_0^K)\right] - \left[\sum_{i=1}^N f_i(\boldsymbol{x}_i^\star) + h(\boldsymbol{x}_0^\star)\right]\right| - \delta_\lambda \sum_{i=1}^N \|\bar{\boldsymbol{x}}_i^K - \bar{\boldsymbol{x}}_0^K\| \tag{A.58}$$

where $\delta_\lambda \triangleq \max\{\|\boldsymbol{\lambda}_1^\star\|, \ldots, \|\boldsymbol{\lambda}_N^\star\|\}$. Thus, we obtain that

$$\left|\left[\sum_{i=1}^N f_i(\bar{\boldsymbol{x}}_i^K) + h(\bar{\boldsymbol{x}}_0^K)\right] - \left[\sum_{i=1}^N f_i(\boldsymbol{x}_i^\star) + h(\boldsymbol{x}_0^\star)\right]\right|$$

$$\leq \frac{\delta_\lambda C_1}{K} + \frac{1}{2\rho K}\sum_{i=1}^N \|\boldsymbol{\lambda}_i^0 - \boldsymbol{\lambda}_i^\star\|^2 + \frac{N\rho}{2K}\|\boldsymbol{x}_0^0 - \boldsymbol{x}_0^\star\|^2 = \frac{\delta_\lambda C_1 + C_2}{K}. \tag{A.59}$$

Finally, combining (A.57) and (A.59) gives rise to (52). ■

References

[1] T.-H. Chang, M. Hong, W.-C. Liao, and X. Wang, "Asynchronous distributed alternating direction method of multipliers: Algorithm and convergence analysis," submitted to *IEEE ICASSP*, Shanghai, China, March 20-25, 2016.

[2] V. Cevher, S. Becker, and M. Schmidt, "Convex optimization for big data," *IEEE Signal Process. Mag.*, pp. 32–43, Sept. 2014.

[3] R. Bekkerman, M. Bilenko, and J. Langford, *Scaling up Machine Learning- Parallel and Distributed Approaches*. Cambridge University Press, 2012.

[4] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: Numerical methods*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.

[5] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *J. Roy. Stat. Soc. B*, vol. 58, pp. 267–288, 1996.

[6] J. Liu, J. Chen, and J. Ye, "Large-scale sparse logistic regression," in *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining*, New York, NY, USA, June 28 - July 1, 2009, pp. 547–556.

[7] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY, USA: Springer-Verlag, 2001.

[8] P. Richtárik, M. Takáč, and S. D. Ahipasaoglu, "Alternating maximization: Unifying framework for 8 sparse PCA formulations and efficient parallel codes," [Online] http://arxiv.org/abs/1212.4137.

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

[10] F. Niu, B. Recht, C. Re, and S. J. Wright, "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent," *Proc. Advances in Neural Information Processing Systems (NIPS)*, vol. 24, pp. 693-701, 2011, [Online] http://arxiv.org/abs/1106.5730.

[11] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," *Proc. Advances in Neural Information Processing Systems (NIPS)*, vol. 24, pp. 873-881, 2011, [Online] http://arxiv.org/abs/1104.5525.

[12] M. Li, L. Zhou, Z. Yang, A. Li, F. Xia, D. G. Andersen, and A. Smola, "Parameter server for distributed machine learning," [Online] http://www.cs.cmu.edu/~muli/file/ps.pdf.

[13] M. Li, D. G. Andersen, and A. Smola, "Distributed delayed proximal gradient methods," [Online] http://www.cs.cmu.edu/~muli/file/ddp.pdf.

[14] J. Liu and S. J. Wright, "Asynchronous stochastic coordinate descent: Parallelism and convergence properties," *SIAM J. Optim.*,, vol. 25, no. 1, pp. 351–376, Feb. 2015.

[15] M. Razaviyayn, M. Hong, Z.-Q. Luo, and J. S. Pang, "Parallel successive convex approximation for nonsmooth nonconvex optimization," in *the Proceedings of the Neural Information Processing (NIPS)*, 2014.

[16] G. Scutari, F. Facchinei, P. Song, D. P. Palomar, and J.-S. Pang, "Decomposition by partial linearization: Parallel optimization of multi-agent systems," *IEEE Transactions on Signal Processing*, vol. 63, no. 3, pp. 641–656, 2014.

[17] A. Daneshmand, F. Facchinei, V. Kungurtsev, and G. Scutari, "Hybrid random/deterministic parallel algorithms for nonconvex big data optimization," to appear in *IEEE Trans. on Signal Processing* [Online] http://www.eng.buffalo.edu/~gesualdo/Papers/DanFaccKungTSPsub14.pdf.

[18] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," to appear in *SIAM J. Opt.*; available on http://arxiv.org/pdf/1410.1390.pdf.

[19] R. Zhang and J. T. Kwok, "Asynchronous distributed ADMM for consensus optimization," in *Proc. 31th ICML, , 2014.*, Beijing, China, June 21-26, 2014, pp. 1–9.

[20] M. Hong, "A distributed, asynchronous and incremental algorithm for nonconvex optimization: An ADMM based approach," technical report; available on http://arxiv.org/pdf/1412.6058.

[21] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Asynchronous distributed optimization using a randomized alternating direction method of multipliers," in *Proc. IEEE CDC*, Florence, Italy, Dec. 10-13, 2013, pp. 3671–3676.

[22] E. Wei and A. Ozdaglar, "On the $O(1/K)$ convergence of asynchronous distributed alternating direction method of multipliers," available on arxiv.org.

[23] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Puschel, "D-ADMM: A communication-efficient distributed algorithm for separable optimization," *IEEE. Trans. Signal Process.*, vol. 60, no. 10, pp. 2718–2723, May 2013.

[24] Q. Ling, Y. Xu, W. Yin, and Z. Wen, "Decentralized low-rank matrix completion," in *Proc. IEEE ICASSP*, Kyoto, Japan, March 25-30, 2012, pp. 2925–2928.

[25] T.-H. Chang, W.-C. Liao, M. Hong, and X. Wang, "Asynchronous distributed ADMM for large-scale optimization- Part II: Linear convergence analysis and numerical performance," submitted for publication.

[26] R. Tibshirani and M. Saunders, "Sparisty and smoothness via the fused lasso," *J. R. Statist. Soc. B*, vol. 67, no. 1, pp. 91–108, 2005.

[27] J. Zhang, S. Nabavi, A. Chakrabortty, and Y. Xin, "Convergence analysis of ADMM based power system mode estimation under asynchronous wide-area communication delays," in *Proc. IEEE PES General Meeting*, Denver, CO, USA, July 26-30, 2015, pp. 1–5.

[28] T.-H. Chang, A. Nedić, and A. Scaglione, "Distributed constrained optimization by consensus-based primal-dual perturbation method," *IEEE. Trans. Auto. Control.*, vol. 59, no. 6, pp. 1524–1538, June 2014.

[29] J.-Y. Joo and M. Ilic, "Multi-layered optimization of demand resources using Lagrange dual decomposition," *IEEE Trans. Smart Grid*, vol. 4, no. 4, pp. 2081–2088, Dec 2013.

[30] E. Dall'Anese, H. Zhu, and G. B. Giannakis, "Distributed optimal power flow for smart microgrids," *IEEE Trans. Smart Grid*, vol. 4, no. 3, pp. 1464–1475, Sept. 2013.

[31] B. He and X. Yuan, "On the $o(1/n)$ convergence rate of Douglas-Rachford alternating direction method," *SIAM J. Num. Anal.*, vol. 50, 2012.

[32] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," Rice CAAM technical report 12-14, 2012.

[33] M. Hong and Z.-Q. Luo, "On the linear convergence of the alternating direction method of multipliers," available on arxiv.org.

[34] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Kluwer Academic Publishers, 2004.

[35] T.-H. Chang, M. Hong, W.-C. Liao, and X. Wang, "Electronic companion for "Asynchronous distributed ADMM for large-scale optimization- Part I: Algorithm and convergence analysis," available on http://arxiv.org.

[36] D. P. Bertsekas, *Nonlinear Programming: 2nd Ed.* Cambridge, Massachusetts: Athena Scientific, 2003.

[37] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

Algorithm 2 Asynchronous Distributed ADMM for (4).

Algorithm of the Master:

Given initial variable $\boldsymbol{x}^0$ and broadcast it to the workers. Set $k=0$ and $d_1 = \cdots = d_N = 0$;

repeat

  wait until receiving $\{\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{\lambda}}_i\}_{i\in\mathcal{A}_k}$ from workers $i \in \mathcal{A}_k$ such that $|\mathcal{A}_k| \geq A$ and $d_i < \tau - 1$ $\forall i \in \mathcal{A}_k^c$.

  update

$$\boldsymbol{x}_i^{k+1} = \begin{cases} \hat{\boldsymbol{x}}_i & \forall i \in \mathcal{A}_k \\ \boldsymbol{x}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{9}$$

$$\boldsymbol{\lambda}_i^{k+1} = \begin{cases} \hat{\boldsymbol{\lambda}}_i & \forall i \in \mathcal{A}_k \\ \boldsymbol{\lambda}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{10}$$

$$d_i = \begin{cases} 0 & \forall i \in \mathcal{A}_k \\ d_i + 1 & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{11}$$

$$\begin{aligned} \boldsymbol{x}_0^{k+1} = \arg\min_{\boldsymbol{x}_0 \in \mathbb{R}^n} \Big\{ & h(\boldsymbol{x}_0) - \boldsymbol{x}_0^T \textstyle\sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} \\ & + \tfrac{\rho}{2} \textstyle\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0\|^2 + \tfrac{\gamma}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_0^k\|^2 \Big\}, \end{aligned} \tag{12}$$

  broadcast $\boldsymbol{x}_0^{k+1}$ to the workers in $\mathcal{A}_k$.

  set $k \leftarrow k+1$.

until a predefined stopping criterion is satisfied.

Algorithm of the ith Worker:

Given initial $\boldsymbol{\lambda}^0$ and set $k_i = 0$.

repeat

  wait until receiving $\hat{\boldsymbol{x}}_0$ from the master node.

  update

$$\boldsymbol{x}_i^{k_i+1} = \arg\min_{\boldsymbol{x}_i \in \mathbb{R}^n} f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \boldsymbol{\lambda}_i^{k_i} + \tfrac{\rho}{2}\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_0\|^2, \tag{13}$$

$$\boldsymbol{\lambda}_i^{k_i+1} = \boldsymbol{\lambda}_i^{k_i} + \rho(\boldsymbol{x}_i^{k_i+1} - \hat{\boldsymbol{x}}_0). \tag{14}$$

  send $(\boldsymbol{x}_i^{k_i+1}, \boldsymbol{\lambda}_i^{k_i+1})$ to the master node.

  set $k_i \leftarrow k_i + 1$.

until a predefined stopping criterion is satisfied.

Algorithm 3 Asynchronous distributed ADMM from the master's point of view.

1: **Given** initial variables $\boldsymbol{x}^0$ and $\boldsymbol{\lambda}^0$; set $\boldsymbol{x}_0^0 = \boldsymbol{x}^0$ and $k = 0$.

2: **repeat**

3: **update**

$$\boldsymbol{x}_i^{k+1} = \begin{cases} \arg\min_{\boldsymbol{x}_i \in \mathbb{R}^n} \Big\{ f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \boldsymbol{\lambda}_i^k \\ \qquad + \frac{\rho}{2} \|\boldsymbol{x}_i - \boldsymbol{x}_0^{\bar{k}_i+1}\|^2 \Big\}, & \forall i \in \mathcal{A}_k \\ \boldsymbol{x}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{23}$$

$$\boldsymbol{\lambda}_i^{k+1} = \begin{cases} \boldsymbol{\lambda}_i^k + \rho(\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0^{\bar{k}_i+1}) & \forall i \in \mathcal{A}_k \\ \boldsymbol{\lambda}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{24}$$

$$\boldsymbol{x}_0^{k+1} = \arg\min_{\boldsymbol{x}_0 \in \mathbb{R}^n} \Big\{ h(\boldsymbol{x}_0) - \boldsymbol{x}_0^T \textstyle\sum_{i=1}^N \boldsymbol{\lambda}_i^{k+1} \\ \qquad + \frac{\rho}{2} \textstyle\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0\|^2 + \frac{\gamma}{2} \|\boldsymbol{x}_0 - \boldsymbol{x}_0^k\|^2 \Big\}. \tag{25}$$

4: **set** $k \leftarrow k + 1$.

5: **until** a predefined stopping criterion is satisfied.

Algorithm 4 An Alternative Implementation of Asynchronous Distributed ADMM.

Algorithm of the Master:

Given initial variable $\boldsymbol{x}^0$ and broadcast it to the workers. Set $k = 0$ and $d_1 = \cdots = d_N = 0$;

repeat

wait until receiving $\{\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{\lambda}}_i\}_{i\in\mathcal{A}_k}$ from workers $i \in \mathcal{A}_k$ such that $|\mathcal{A}_k| \geq A$ and $d_i < \tau - 1$ $\forall i \in \mathcal{A}_k^c$.

update

$$\boldsymbol{x}_i^{k+1} = \begin{cases} \hat{\boldsymbol{x}}_i & \forall i \in \mathcal{A}_k \\ \boldsymbol{x}_i^k & \forall i \in \mathcal{A}_k^c \end{cases}, \tag{44}$$

$$d_i = \begin{cases} 0 & \forall i \in \mathcal{A}_k \\ d_i + 1 & \forall i \in \mathcal{A}_k^c \end{cases},$$

$$\boldsymbol{x}_0^{k+1} = \arg\min_{\boldsymbol{x}_0\in\mathbb{R}^n} \Big\{ h(\boldsymbol{x}_0) - \boldsymbol{x}_0^T \textstyle\sum_{i=1}^N \boldsymbol{\lambda}_i^k + \frac{\rho}{2}\sum_{i=1}^N \|\boldsymbol{x}_i^{k+1} - \boldsymbol{x}_0\|^2 + \frac{\gamma}{2}\|\boldsymbol{x}_0 - \boldsymbol{x}_0^k\|^2 \Big\}, \tag{45}$$

$$\boldsymbol{\lambda}_i^{k+1} = \boldsymbol{\lambda}_i^k + \rho(\boldsymbol{x}_i^{k_i+1} - \boldsymbol{x}_0^{k+1}) \ \forall i \in \mathcal{V}. \tag{46}$$

broadcast $\boldsymbol{x}_0^{k+1}$ and $\{\boldsymbol{\lambda}_i^{k+1}\}_{i\in\mathcal{A}_k}$ to the workers in $\mathcal{A}_k$.

set $k \leftarrow k + 1$.

until a predefined stopping criterion is satisfied.

Algorithm of the ith Worker:

Given initial $\boldsymbol{\lambda}^0$ and set $k_i = 0$.

repeat

wait until receiving $(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{\lambda}}_i)$ from the master node.

update

$$\boldsymbol{x}_i^{k_i+1} = \arg\min_{\boldsymbol{x}_i\in\mathbb{R}^n} f_i(\boldsymbol{x}_i) + \boldsymbol{x}_i^T \hat{\boldsymbol{\lambda}}_i + \frac{\rho}{2}\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_0\|^2, \tag{47}$$

send $\boldsymbol{x}_i^{k_i+1}$ to the master node.

set $k_i \leftarrow k_i + 1$.

until a predefined stopping criterion is satisfied.