Asynchronous Dual Free Stochastic Dual Coordinate Ascent for Distributed Data Mining

Zhouyuan Huo[1], Xue Jiang[3], Heng Huang[1,2]
[1]Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA, United States
[2]JD.com, [3]School of Electronic Information, Wuhan University, Wuhan, China
Email: zhouyuan.huo@pitt.edu, jxt@whu.edu.cn, heng.huang@pitt.edu

***Abstract*—The primal-dual distributed computational methods have broad large-scale data mining applications. Previous primal-dual distributed methods are not applicable when the dual formulation is not available, *e.g.* the sum-of-non-convex objectives. Moreover, these algorithms and theoretical analysis are based on the fundamental assumption that the computing speeds of multiple machines in a cluster are similar. However, the straggler problem is an unavoidable practical issue in the distributed system because of the existence of slow machines. Therefore, the total computational time of the distributed optimization methods is highly dependent on the slowest machine. In this paper, we address these two issues by proposing novel distributed asynchronous dual free stochastic dual coordinate ascent algorithm for distributed data mining. Our method does not need the dual formulation of the target problem in the computation. We tackle the straggler problem through asynchronous communication and the negative effect of slow machines is significantly alleviated. We also analyze the convergence rate of our method and prove the linear convergence rate even if the individual functions in objective are non-convex. Experiments on both convex and non-convex loss functions are used to validate our statements.**

I. Introduction

In this paper, we consider solving the ℓ_2-norm regularized empirical loss minimization problem which is arising ubiquitously in supervised machine learning and data mining problems:

$$\min_{w\in\mathbb{R}^d} P(w) := \min_{w\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n}\phi_i(w) + \frac{\lambda}{2}\|w\|_2^2. \tag{1}$$

We let $f(w) = \frac{1}{n}\sum_{i=1}^{n}\phi_i(w)$ and $w \in \mathbb{R}^d$ be the linear predictor to be optimized. There are many applications falling into this formulation, such as classification, regression, and principal component analysis (PCA). In classification, given features $x_i \in \mathbb{R}^d$ and labels $y_i \in \{1, -1\}$, we obtain Support Vector Machine (SVM) when we let $\phi_i(w) = \max\{0, 1 - y_i x_i^T w\}$. In regression, given features $x_i \in \mathbb{R}^d$ and response $y_i \in \mathbb{R}$, we have Ridge Regression problem if $\phi_i(w) = (y_i - x_i^T w)^2$. Recently, [1], [2] showed that the problem of PCA can be solved through convex optimization. Supposing $C = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$ be normalized covariance matrix, [1] showed that approximating the principle component of A is equivalent to minimizing $f(w) = \frac{1}{2}w^T(\mu I - C)w - b^T w$ given $\mu > 0$ and $b \in \mathbb{R}^d$. Defining $\phi_i(w) = \frac{1}{2}w^T((\mu - \frac{\lambda}{2})I - x_i x_i^T)w - b^T w$ and $\mu > \sigma_1(C) + \frac{\lambda}{2}$ where $\sigma_1(C)$ denotes the largest singular value of C, it also falls into problem (1). In this case, $f(w)$ is convex while each $\phi_i(w)$ is probably non-convex.

Distributed machine learning and data mining methods are required to solve the problem (1) when the data are distributed over multiple machines. In [3], the authors proposed communication-efficient distributed dual coordinate ascent (CoCoA) for primal-dual distributed optimization. In each iteration, the CoCoA framework allows workers to optimize subproblems independently at first. After that, it calls "Reduce" operation to collect local solution from all workers, and updates global variable and broadcasts the up-to-date global variable to workers in the end. It uses stochastic dual coordinate ascent (SDCA) as the local solver which is one of the most successful methods proposed for solving the problem (1) [4], [5]. In [6], the authors proved that SDCA has linear convergence if the convex function $\phi_i(w)$ is smooth, which is much faster than stochastic gradient descent (SGD). [7], [8] also proposed distributed SDCA and analyzed the tradeoff between computation and communication. [9], [10] accelerated the CoCoA by allowing for more aggressive updates, and proved that CoCoA has linear primal-dual convergence for the smooth convex problem and sublinear convergence for the non-smooth convex problem. However, there are two issues for these primal-dual distributed methods. Firstly, all of them use SDCA as the local solver. SDCA is not applicable when the dual problem is unknown, *e.g.* ϕ_i is non-convex. Therefore, the applications of these primal-dual distributed methods are limited. Secondly, all of these methods assume that the workers have similar computing speed, which is not true in practice. Straggler problem is an unavoidable practical issue in the distributed data mining. Thus, the computing time of CoCoA and distributed SDCA is dependent on the slowest worker. Even if there is only one bad worker, they will work far slower than expectation.

In [11], [12], the authors proposed dual free stochastic dual coordinate ascent (dfSDCA). It was proved to admit similar convergence rate to SDCA while it did not rely on duality at all. However there is no distributed machine learning method using dfSDCA, and its convergence analysis is still unknown yet.

In this paper, we solve the above two challenging issues in previous primal-dual distributed machine learning methods by proposing novel Distributed Dual Free Stochastic Dual Coordinate Ascent (Dis-dfSDCA). We use dfSDCA as the local solver such that Dis-dfSDCA can be applied to the non-convex problem easily. We alleviate the effect of straggler

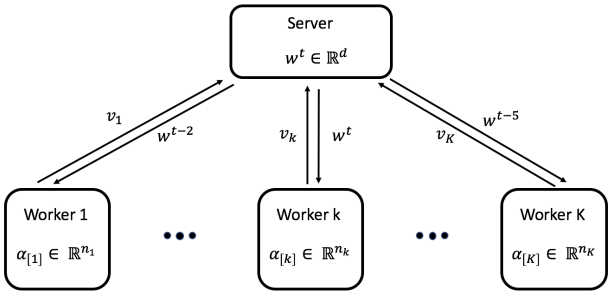


Fig. 1: Distributed asynchronous dual free stochastic dual coordinate ascent for parameter server framework. In iteration t, the server receives gradient message v_k from worker k, and sends the up-to-date w^t back to the worker k. Global variables in other workers are stale. For example worker 1 and K store stale global variables w^{t-2} and w^{t-5} respectively.

problem by allowing asynchronous communication between server and workers. As shown in Figure 1, the server does not wait and workers may store the stale global variable in the local. We also analyze the convergence rate of our method and prove that it admits linear convergence rate even if the individual losses (ϕ_i) are non-convex, as long as the sum of losses f is convex. Finally, we conduct simulation on the distributed system with straggler problem. Experimental results verify our theoretical conclusions and show that our method works well in practice.

II. Preliminary

To optimize the primal problem (1), we often derive and optimize its dual problem alternatively:

$$\max_{\alpha\in\mathbb{R}^n} D(\alpha) := \max_{\alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n} -\phi_i^*(-\alpha_i) - \frac{\lambda}{2}\|\frac{1}{\lambda n}A\alpha\|_2^2\,, \quad (2)$$

where ϕ_i^* is the convex conjugate function to ϕ_i, $A = [x_1, x_2, ...x_n] \in \mathbb{R}^{d\times n}$ denotes data matrix and $\alpha \in \mathbb{R}^n$ denotes dual variable. We can use stochastic gradient descent (SGD) to optimize primal problem (1), however, there are always two issues: (1) SGD is too aggressive at the beginning of the optimization; (2) it does not have a clear stopping criterion. One of the biggest advantages of optimizing the dual problem is that we can keep tracking the duality gap $G(\alpha)$ to monitor the progress of optimization. Duality gap is defined as: $G(\alpha) = P(w(\alpha)) - D(\alpha)$, where $P(w(\alpha))$ and $D(\alpha)$ denote objective values of primal problem and dual problem respectively. If w^* is the optimal solution of primal problem (1) and α^* is the optimal solution of dual problem (2), the primal-dual relation always holds that:

$$w^* = w(\alpha^*) = \frac{1}{\lambda n}A\alpha^*\,. \quad (3)$$

A. Stochastic Dual Coordinate Ascent

In [6], the authors proposed stochastic dual coordinate ascent (SDCA) to optimize the dual problem (2). The pseudocode of SDCA is presented in Algorithm 1. In iteration t, given sample i and other dual variables $\alpha_{j\neq i}$ fixed, we maximize the following subproblem:

$$\max_{\Delta\alpha_i\in\mathbb{R}} -\frac{1}{n}\phi_i^*(-(\alpha_i^t + \Delta\alpha_i)) - \frac{\lambda}{2}\|w^t + \frac{1}{\lambda n}\Delta\alpha_i x_i\|_2^2 \quad (4)$$

e_i denotes coordinate vector of size n, where element i is 1 and other elements are 0. In their paper, the authors proved that SDCA admits linear convergence rate for smooth loss, which is much faster than stochastic gradient descent (SGD). An accelerated SDCA was also proposed in [5]. However, SDCA is not applicable when it is difficult to derive the dual problem, *e.g.* ϕ_i are non-convex.

Algorithm 1 SDCA

1: Initialize α^0 and $w^0 = w(\alpha^0)$;
2: **for** $t = 0, 1, 2, \ldots, T-1$ **do**
3: Randomly sample i from $\{1, 2, ..., n\}$;
4: Find $\Delta\alpha_i$ to maximize the subproblem (4);
5: Update dual variable α through:
$$\alpha^{t+1} \leftarrow \alpha^t + \Delta\alpha_i e_i;$$
6: Update primal variable w through:
$$w^{t+1} \leftarrow w^t + \tfrac{1}{\lambda n}\Delta\alpha_i x_i;$$
7: **end for**

B. Dual Free Stochastic Dual Coordinate Ascent

To address the limitation of SDCA, [11] proposes Dual Free Stochastic Coordinate Ascent (dfSDCA) which has similar convergence property to SDCA. The pseudocode of dfSDCA is presented in Algorithm 2. Although we keep vector $\alpha \in \mathbb{R}^n$ in the optimization, the derivation of dual problem is not necessary for dfSDCA. According to the update rule of α and w in the algorithm, the primal-dual relation (3) also holds for dfSDCA. The drawback of dfSDCA is that it is space-consuming to store α, whose space complexity $O(nd)$. We can reduce it to $O(n)$ if $\nabla\phi_i(w)$ can be written as $\nabla\phi_i(x_i^T w)x_i$. In [13], the authors accelerated dfSDCA by using non-uniform sampling strategy in each iteration and proved that it admits faster convergence.

Algorithm 2 Dual Free SDCA

1: Initialize dual variable $\alpha^0 = (\alpha_0^0, ..., \alpha_n^0)$ where $\forall i, \alpha_i^0 \in \mathbb{R}^d$, primal variable $w^0 = w(\alpha^0)$;
2: **for** $t = 0, 1, 2, \ldots, T-1$ **do**
3: Randomly sample i from $\{1, 2, ..., n\}$;
4: Compute dual residue κ through:
$$\kappa \leftarrow \nabla\phi_i(w^t) + \alpha_i^t;$$
5: Update dual variable α_i through:
$$\alpha_i^{t+1} \leftarrow \alpha_i^t - \eta\lambda n\kappa;$$
6: Update primal variable w through:
$$w^{t+1} \leftarrow w^t - \eta\kappa;$$
7: **end for**

III. Distributed Asynchronous Dual Free Stochastic Dual Coordinate Ascent

In this section, we propose Distributed Asynchronous Dual Free Stochastic Coordinate Ascent (Dis-dfSDCA) for distributed optimization. Dis-dfSDCA fits for any parameter server framework, where the star-shape network is used. We assume that there are n samples in the dataset, and they are evenly distributed over K workers. In worker k, there are n_k samples. It is satisfied that $n = \sum_{k=1}^{K} n_k$. Different from sequential dfSDCA, we split the update of dual variable and primal variable into different nodes. The pseudocodes of Dis-dfSDCA for server node and worker nodes are presented in Algorithm 3 and Algorithm 4 respectively.

A. Update Global Variable w on Server

The up-to-date global variable $w \in \mathbb{R}^d$ is stored and updated on the server. Initially, w is set to be vector zero. At the beginning of each iteration, the server receives gradient message v_k from arbitrary worker k and let $v^t = v_k$. Then it updates the global variable through:

$$w^{s,t+1} = w^{s,t} - \eta v^t \tag{5}$$

Finally, it sends the up-to-date global variable back to the worker k for further computation. Asynchronous method is robust to straggler problem because it allows for updating the global variable when receiving from only one worker. However, if the w in the worker is too stale, it may lead the algorithm to diverge. Therefore, we induce two loops in our algorithm. Server broadcasts the latest global variable w to all workers after every T iterations. In this way, we prevent the problem of divergence and keep the advantage of asynchronous communication at the same time. Algorithm 3 summarizes the pseudocode on the server.

Algorithm 3 Dis-dfSDCA (Server)

Initialize $w \in \mathbb{R}^d$, η
for $s = 0, 1, ..., S-1$ **do**
  for $t = 0, 1, ..., T-1$ **do**
    Receive gradient message $v^{s,t} = v_k$ from worker k;
    Update global variable $w^{s+1,t+1}$ through:
      $w^{s,t+1} \leftarrow w^{s,t} - \eta v^{s,t}$;
    Send $w^{s,t+1}$ back to worker k ;
  end for
  $w^{s+1,0} = w^{s,T}$
  Broadcast the up-to-date global variable $w^{s+1,0}$ to all workers.
end for

In Algorithm 3, we use the update of vanilla dfSDCA in the server. [12] proposed accelerated dfSDCA by using "Catalyst" algorithm of [14]. It is proved to admit faster convergence rate by a constant factor. Our Algorithm 3 can also be extended to the accelerated version easily. In our paper, we only consider the vanilla version and analyze the convergence rate of our algorithm.

B. Update Local Variable α on Worker

In the distributed optimization, workers are responsible for the gradient computation which is the main workload during the optimization. We take arbitrary worker k as an example. Dual variable $\alpha_{[k]} \in \mathbb{R}^{n_k}$ is only stored and updated in the worker k, each α_i is corresponding to sample i. Initially, local variable $\alpha_{[k]}$ is set to be vector zero. After receiving stale global variable $w^{s,d(t)} \in \mathbb{R}^d$ from the server, worker k computes the dual residue κ and updates local variable α_i and gradient message v_k for H iterations. Samples I_t are randomly selected in the local dataset, and we set $|I_t| = H$. In each iteration, worker k selects a sample i randomly and computes the dual residue κ for coordinate i of the dual variable through:

$$\kappa = \nabla \phi_i(w^{s,d(t)}) + \alpha_i \tag{6}$$

Dual residue can also be viewed as the gradient in Stochastic Gradient Descent. When we obtain optimal dual variable α^* and primal variable w^*, κ should be 0. Therefore, it is satisfied that $\alpha_i^* = -\nabla \phi_i(w^*)$. Then worker k updates local dual variable α_i and gradient message v_k separately through:

$$\alpha_i = \alpha_i - \eta \lambda n \kappa, \quad i \in I_t \tag{7}$$

$$v_k = v_k + \kappa \tag{8}$$

Because there is only one α_i in the cluster, it is always up-to-date. After H iterations, the worker k sends gradient message v_k to the server. From the update rule in our algorithm, it is easy to know that the well-known primal-dual relation in the equation (3) is always satisfied. The pseudocode of Dis-dfSDCA in worker node k is described in Algorithm 4.

Algorithm 4 Dis-dfSDCA (Worker k)

Initialize $\alpha_{[k]} \in \mathbb{R}^{d \times n_k}$, η, H
repeat
  Receive global variable $w^{s,d(t)}$ from server;
  Initialize gradient message: $v_k \leftarrow 0$;
  Randomly select samples I_t from $\{1, \cdots, n_k\}$ where $|I_t| = H$;
  for sample i in I_t **do**
    Compute dual residue κ through:
      $\kappa \leftarrow \nabla \phi_i(w^{s,d(t)}) + \alpha_i$;
    Update local dual variable α_i through:
      $\alpha_i \leftarrow \alpha_i - \eta \lambda n \kappa$;
    Update gradient message v_k through:
      $v_k \leftarrow v_k + \kappa$;
  end for
  Send gradient message v_k to server;
until Termination

In Algorithm 4, we use vanilla dfSDCA in the worker which samples with uniform distribution. There are also other sampling techniques proposed to accelerate dfSDCA. As per the sampling strategy in [11], [13], [12], [15], there are three options: uniform sampling, importance sampling, and adaptive sampling. In importance sampling strategy [12], it first computes the fixed probability distribution p_i using smoothness

parameter of each function ϕ_i, then selects samples following this probability. In adaptive sampling strategy [13], it computes the adaptive probability distribution p_i using dual residue κ for each sample every iteration, then selects samples following this probability. Both of them are proved to admit faster convergence than vanilla dfSDCA with uniform sampling. We only consider the uniform sampling strategy, and analyze its corresponding convergence rate in our paper. However, other sampling techniques are straightforward to be applied to our distributed method.

IV. Convergence Analysis

In this section, we provide the theoretical convergence analysis of Dis-dfSDCA. For the case of convex losses ϕ_i, we prove that Dis-dfSDCA admits linear convergence rate. If losses ϕ_i are non-convex, we also prove linear convergence rate as long as the sum-of-non-convex objectives f is convex.

We make the following assumptions for the primal problem (1) for further analysis. All of them are common assumptions in the theoretical analysis for the asynchronous stochastic methods.

Assumption 1 (Lipschitz Constant): We assume $\nabla\phi_i$ is Lipschitz continuous, and there is Lipschitz constant L such that $\forall x, y \in \mathbb{R}^d$:

$$\|\nabla\phi_i(x) - \nabla\phi_i(y)\|_2 \leq L\|x-y\|_2 \tag{9}$$

We can also know that P is $(L+\lambda)$-smooth:

$$\|\nabla P(x) - \nabla P(y)\|_2 \leq (L+\lambda)\|x-y\|_2 \tag{10}$$

Assumption 2 (Maximum Time Delay): We assume that the maximum time delay of the global variable in each worker is upper bounded by τ, such that:

$$d(t) \geq t - \tau \tag{11}$$

τ is relevant to the number of workers K in the system. We can also control τ through inner iteration T in our algorithm.

A. Convex Case

In this section, we assume that the losses ϕ_i are convex, and prove that our method admits linear convergence.

Assumption 3 (Convexity): We assume losses ϕ_i are convex, such that $\forall x, y \in \mathbb{R}^d$:

$$\phi_i(x) \geq \phi_i(y) + \nabla\phi_i(y)^T(x-y)\,. \tag{12}$$

In our algorithm, dual variables $\alpha_{[1]}, ..., \alpha_{[K]}$ are stored in local workers. For worker k, there is no update of $\alpha_{[k]}$ from $d(t)$ to t. Therefore, it is always true that $\alpha_{[k]}^{s,t} = \alpha_{[k]}^{s,d(t)}$. For brevity, we write $v^{s,t}$, $w^{s,t}$ and $\alpha^{s,t}$ as v^t, w^t and α^t. According to our algorithm, we know that:

$$v^t = \sum_{i\in I_t}\left(\nabla\phi_i(w^{d(t)}) + \alpha_i^{d(t)}\right) = \sum_{i\in I_t} v_i^t \tag{13}$$

where $|I_t| = H$ and $\mathbb{E}[v_i^t] = \nabla P(w^{d(t)})$. In our analysis, we also assume that there are no duplicate samples in I_t. To analyze the convergence rate of our method, we need to prove the following Lemma 1 at first.

Lemma 1: Let w^* be the global solution of $P(w)$, and $\alpha_i^* = -\nabla\phi_i(w^*)$. Following the proof in [11], we define A_t and B_t as follows:

$$A_t = \mathbb{E}\|\alpha_i^t - \alpha_i^*\|^2 \tag{14}$$

$$B_t = \mathbb{E}\|w^t - w^*\|^2 \tag{15}$$

According to our algorithm, we can prove that A_{t+1} and B_{t+1} are upper bounded:

$$\begin{aligned}
&\mathbb{E}[A_{t+1} - A_t] \\
\leq\ & -\eta\lambda H\mathbb{E}\|\alpha_i^t - \alpha_i^*\|^2 - 2\eta HL\lambda^2\mathbb{E}\|w^t - w^*\|^2 \\
& +4\eta\lambda HL\left(P(x^t) - P(w^*)\right) - \eta\lambda(1-\eta\lambda n)\mathbb{E}\|v^t\|^2 \\
& +2\lambda\tau HL^2\eta^3\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2
\end{aligned} \tag{16}$$

$$\begin{aligned}
&\mathbb{E}[B_{t+1} - B_t] \\
\leq\ & -2\eta\left(P(w^{d(t)}) - P(w^*)\right) + \eta^2\mathbb{E}\|v^t\|^2 \\
& -2\eta\left\langle w^t - w^{d(t)}, \nabla P(x^{d(t)})\right\rangle
\end{aligned} \tag{17}$$

Theorem 1: Suppose losses ϕ_i are convex and $\nabla\phi_i$ are Lipschitz continuous. Let w^* be the optimal solution to $P(w)$, and $\alpha_i^* = -\nabla\phi_i(w^*)$. Define $C_t = \frac{1}{2\lambda L}A_t + B_t$. We can prove that as long as:

$$\eta \leq \frac{1}{4HL\tau^2 + \lambda n + 2L} \tag{18}$$

the following inequality holds:

$$\mathbb{E}[C_T] \leq (1 - \eta\lambda H)\mathbb{E}[C_0] \tag{19}$$

Proof sketch:[1] Substituting A_{t+1} and B_{t+1} according to Lemma 1, the following inequality holds that:

$$\begin{aligned}
\mathbb{E}[C_{t+1}] &= \frac{1}{2\lambda L}A_{t+1} + B_{t+1} \\
&\leq (1-\eta\lambda H)\mathbb{E}[C_t] + 2\tau HL\eta^3\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \\
&\quad + \left(\frac{\eta^2\lambda n}{2L} + \eta^2 - \frac{\eta}{2L}\right)\mathbb{E}\|v^t\|^2
\end{aligned} \tag{20}$$

Adding the above inequality from $t=0$ to $t=T-1$, we have:

$$\begin{aligned}
&\sum_{t=0}^{T-1}\mathbb{E}[C_{t+1}] \leq \sum_{t=0}^{T-1}(1-\eta\lambda H)\mathbb{E}[C_t] \\
&+\left(2H\tau^2\eta^2 + \tfrac{\eta^2\lambda n}{2L} + \eta^2 - \tfrac{\eta}{2L}\right)\sum_{t=0}^{T-1}\mathbb{E}\|v^t\|^2
\end{aligned} \tag{21}$$

where the inequality follows from Assumption 2 and $\eta L \leq 1$. If $2H\eta^2\tau^2 + \frac{\eta^2\lambda n}{2L} + \eta^2 - \frac{\eta}{2L} \leq 0$, such that:

$$\eta \leq \frac{1}{4HL\tau^2 + \lambda n + 2L}, \tag{22}$$

[1]We provide the proof sketch here, please check the Appendix for details.

we have the following inequality:

$$\begin{aligned}\sum_{t=0}^{T-1}\mathbb{E}[C_{t+1}] &\leq \sum_{t=0}^{T-1}(1-\eta\lambda H)\mathbb{E}[C_t] \\ &\leq \sum_{t=1}^{T-1}\mathbb{E}[C_t]+(1-\eta\lambda H)C_0 \end{aligned} \quad (23)$$

Because $C_t \geq 0$, then we complete the proof that $\mathbb{E}[C_T] \leq (1-\eta\lambda H)\mathbb{E}[C_0]$. □

Because $\nabla P(w)$ is Lipschitz continuous, we know that:

$$P(w^t)-P(w^*) \leq \frac{L+\lambda}{2}\|w^t-w^*\|^2 \leq \frac{L+\lambda}{2}C_t \quad (24)$$

Theorem 2: We consider the outer iteration s, and write C^t as $C^{s,t}$. According to Algorithm 3, we know $C^{s+1,0}=C^{s,T}$. Following Theorem 1 and applying (19) for S iterations, it is satisfied that:

$$\mathbb{E}[C_{S,0}] \leq (1-\eta\lambda H)^S\mathbb{E}[C_{0,0}] \quad (25)$$

In particular, to achieve $\mathbb{E}[P(w^{S,0})-P(w^*)] \leq \varepsilon$, it suffices to set $\eta=\frac{1}{4HL\tau^2+\lambda n+2L}$ and

$$S \geq O\left(\left(\frac{L}{\lambda}\left(\tau^2+\frac{1}{H}\right)+\frac{n}{H}\right)\log\left(\frac{1}{\varepsilon}\right)\right) \quad (26)$$

From Theorem 1 and 2, we know that our Dis-dfSDCA admits linear convergence if losses ϕ_i are convex. According to Theorem 2, we observe that τ affects the speed of our convergence, if $\tau \to \infty$, it may lead our algorithm to diverge. Therefore, it is important to keep τ within a reasonable bound. In our algorithm, τ is relevant to the number of workers and less than T. When we let $H=1$ and $\tau=0$, S is relevant to $O(\frac{L}{\lambda}+n)$. It is compatible with the convergence analysis of sequential dfSDCA in [11].

B. Non-convex Case

In this section, we assume that the losses ϕ_i are non-convex, while the sum-of-non-convex objectives f is convex. We also prove that Dis-dfSDCA admits linear convergence rate for this case. Firstly, we get the following Lemma 2.

Lemma 2: Let w^* be optimal solution to $P(w)$, and $\alpha_i^*=-\nabla\phi_i(w^*)$. Following the definition of A_t and B_t in Lemma 1, we prove that A_{t+1} and B_{t+1} are upper bounded:

$$\begin{aligned}&\mathbb{E}[A_{t+1}-A_t] \\ \leq\ & -\eta\lambda H\mathbb{E}\|\alpha_i^t-\alpha_i^*\|^2+2\eta\lambda HL^2\mathbb{E}\|w^t-w^*\|^2 \\ & -\eta\lambda(1-\eta\lambda n)\mathbb{E}\|v^t\|^2+2\lambda\tau HL^2\eta^3\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \\ &\mathbb{E}[B_{t+1}-B_t] \\ \leq\ & -\frac{3\eta\lambda H}{4}\mathbb{E}\|w^t-w^*\|^2+\eta^2\mathbb{E}\|v^t\|^2 \\ & \frac{2H\tau H^2(L+\lambda)^2\eta^3}{\lambda}\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \end{aligned} \quad (27)$$

Theorem 3: Suppose f is convex and $\nabla\phi_i$ is Lipschitz continuous. Let w^* be the optimal solution to $P(w)$, and $\alpha_i^*=-\nabla\phi_i(w^*)$. Define $C_t=\frac{1}{4L^2}A_t+B_t$. We can prove that as long as:

$$\eta \leq \frac{\lambda^2}{2HL\tau^2\lambda^2+8HL\tau^2(L+\lambda)^2+4\lambda L^2+n\lambda^3} \quad (28)$$

the following inequality holds:

$$\mathbb{E}[C_T] \leq (1-\eta\lambda H)\mathbb{E}[C_0] \quad (29)$$

Proof sketch: Substituting A_{t+1} and B_{t+1} according to Lemma 2, the following inequality holds that:

$$\begin{aligned}&\mathbb{E}[C_{t+1}]=\frac{1}{4L^2}A_{t+1}+B_{t+1} \\ \leq\ & (1-\eta\lambda H)\mathbb{E}[C_t] \\ & +\left(\frac{\lambda H\tau\eta^3}{2}+\frac{2H\tau(L+\lambda)^2\eta^3}{\lambda}\right)\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \\ & +\left(\eta^2+\frac{n\eta^2\lambda^2}{4L^2}-\frac{\eta\lambda}{4L^2}\right)\mathbb{E}\|v^t\|^2 \end{aligned} \quad (30)$$

Adding the above inequality from $t=0$ to $t=T-1$, we have:

$$\begin{aligned}\sum_{t=0}^{T-1}\mathbb{E}[C_{t+1}] \leq \sum_{t=0}^{T-1}(1-\eta\lambda H)\mathbb{E}[C_t]+\Bigg(\eta^2+\frac{n\eta^2\lambda^2}{4L^2} \\ +\frac{\lambda H\tau^2\eta^2}{2L}+\frac{2H\tau^2(L+\lambda)^2\eta^2}{\lambda L}-\frac{\eta\lambda}{4L^2}\Bigg)\sum_{t=0}^{T-1}\mathbb{E}\|v^t\|^2 \end{aligned} \quad (31)$$

where the inequality follows from Assumption 2 and $\eta L \leq 1$. If $\eta^2+\frac{n\eta^2\lambda^2}{4L^2}+\frac{\lambda H\tau^2\eta^2}{2L}+\frac{2H\tau^2(L+\lambda)^2\eta^2}{\lambda L}-\frac{\eta\lambda}{4L^2} \leq 0$, such that:

$$\eta \leq \frac{\lambda^2}{2HL\tau^2\lambda^2+8HL\tau^2(L+\lambda)^2+4\lambda L^2+n\lambda^3} \quad (32)$$

we have the following inequality:

$$\begin{aligned}\sum_{t=0}^{T-1}\mathbb{E}[C_{t+1}] &\leq \sum_{t=0}^{T-1}(1-\eta\lambda H)\mathbb{E}[C_t] \\ &\leq \sum_{t=1}^{T-1}\mathbb{E}[C_t]+(1-\eta\lambda H)C_0 \end{aligned} \quad (33)$$

Because $C_t \geq 0$, then we complete the proof that $\mathbb{E}[C_T] \leq (1-\eta\lambda H)\mathbb{E}[C_0]$. □

Theorem 4: We consider the outer iteration s, and write C^t as $C^{s,t}$. According to Algorithm 3, we know $C^{s+1,0}=C^{s,T}$. Following Theorem 3 and applying (29) for S iterations, it is satisfied that:

$$\mathbb{E}[C_{S,0}] \leq (1-\eta\lambda H)^S\mathbb{E}[C_{0,0}] \quad (34)$$

In particular, to achieve $\mathbb{E}[P(w^{S,0})-P(w^*)] \leq \varepsilon$, it suffices to set $\eta=\frac{\lambda^2}{2HL\tau^2\lambda^2+8HL\tau^2(L+\lambda)^2+4\lambda L^2+n\lambda^3}$ and

$$S \geq O\left(\left(\frac{\left(\tau^2+1/H\right)L^2}{\lambda^2}+\frac{\tau^2L^3}{\lambda^3}+\frac{n}{H}\right)\log\left(\frac{1}{\varepsilon}\right)\right) \quad (35)$$

From Theorems 3 and 4, we know that our Dis-dfSDCA admits linear convergence even if losses ϕ_i are non-convex, as long as the sum-of-non-convex objectives is convex. Comparing Theorems 2 with 4, we can observe that our method needs more iterations to converge to the similar accuracy when ϕ_i are non-convex. It is reasonable because non-convex problem is known to be harder to be optimized than convex problem. When we let $H = 1$ and $\tau = 0$, S is relevant to $O(\frac{L^2}{\lambda^2} + n)$. It is also compatible with the convergence analysis of sequential dfSDCA in [11].

V. Experiments

In this section, we conduct two simulated experiments on the distributed system with straggler problem. There are mainly three goals, firstly, we want to verify that our Dis-dfSDCA has linear convergence rate for the convex and smooth problem; secondly, we would like to make sure that our method has better speedup property than other primal-dual methods; thirdly, we would like to show that our method is also fit for non-convex losses.

Our algorithm is implemented using C++, and the point-to-point communication between worker and server is handled by openMPI [16]. We use Armadillo library [17] for efficient matrix computation. Experiments are performed on Amazon Web Services, and each node is a t2.medium instance which has two virtual CPUs. In our distributed system, we simulate the straggler problem by forcing one selected worker node to the delaying state for m times as long as the normal computing time of other normal workers with probability p. In our experiments, we set $p = 0.2$ and m is selected from $[0, 10]$ randomly. In practice, all nodes have a tiny possibility of being delayed. The setting in our experiments is to verify that our algorithm is robust to straggler problem, even in the extreme situation.

A. Convex Case

In our experiment, we optimize quadratic loss with ℓ_2 regularization term to solve binary classification problem:

$$\min_{w\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(x_i^T w - y_i)^2 + \frac{\lambda}{2}\|w\|^2 \tag{36}$$

where $\lambda = 0.1$. Datasets in our experiments are from LIBSVM [18]. Table I shows brief details of each dataset. In this problem, because $\nabla\phi_i(w)$ can be written as $\nabla\phi_i(x_i^T w)$, we just need to store $\hat{\alpha} \in \mathbb{R}^n$, and recover $\alpha \in \mathbb{R}^{d\times n}$ through $a_i = x_i\hat{\alpha}_i$. Therefore the space complexity is $O(n)$.

We compare our method with CoCoA+ [9], which is the state-of-the-art distributed primal-dual optimization framework. We reimplement CoCoA+ framework using C++, and use SDCA as the local solver. Learning rate η in our method is selected from $\eta = \{1, 0.1, 0.001, 0.0001\}$.

1) Convergence of Duality Gap: We compare the duality gap convergence of compared methods in terms of time and epoch number respectively, where duality gap is well defined in [6]. Experimental results are presented in Figure 2. We distribute IJCNN1 dataset over 4 workers. Figures 2a in the first column show the duality gap convergence in terms of time and epoch on IJCNN1 dataset. From the second figure, it is easy to know that Dis-dfSDCA and CoCoA+ have similar convergence rate. Since CoCoA+ has linear convergence if the problem is convex and smooth, it is verified that Dis-dfSDCA has linear convergence rate as well. In the experiment, we evaluate Dis-dfSDCA when we set different amount of local computations, $H = 10^2$ and $H = 10^3$. Results show that our method is faster than CoCoA+ method in both two cases. The reason is that CoCoA+ is affected by the straggler problem in the distributed system. We also optimize problem (36) with COVTYPE dataset using 8 workers, and RCV1 dataset using 16 workers. We can draw the similar conclusion from the results of other two datasets.

Dataset	# of samples	Dimension	Sparsity
IJCNN1	49,990	22	41 %
COVTYPE	581,012	54	22 %
RCV1	677,399	47,236	0.16%

TABLE I: Experimental datasets from LIBSVM.

2) Speedup: In this section, we evaluate the scaling up ability of compared methods. The first row of Figure 3 presents the speedup of compared methods on IJCNN1 and COVTYPE datasets. Speedup is defined as follows:

$$\text{Time speedup } = \frac{\text{Running time for serial computation}}{\text{Running time of using } K \text{ workers}} \tag{37}$$

Figure in the second row shows the convergence of duality gap on RCV1 on multiple machines. It is obvious that Dis-dfSDCA always converges faster than CoCoA+ when they have the same number of workers. Experimental results verify that Dis-dfSDCA has better speedup property than CoCoA+ when there is straggler problem.

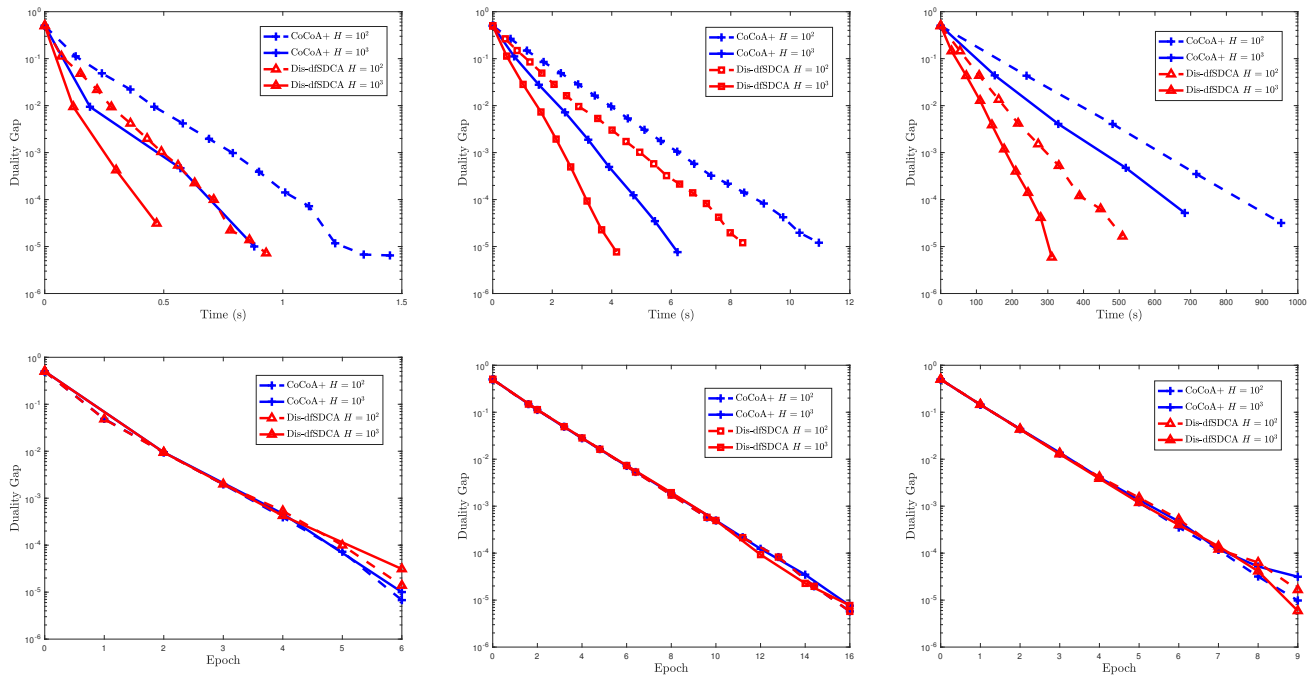
B. Non-convex Case

In this experiment, we optimize the following convex objective, which is an essential step for principal component analysis in [1]:

$$\min_{w\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2} w^T\left((\mu-\lambda) - x_i x_i^T\right) w - b^T w + \frac{\lambda}{2}\|w\|^2 \tag{38}$$

We conduct the experiment on synthetic data and generate $n = 500,000$ random vectors $\{x_1, ..., x_{500,000}\} \in \mathbb{R}^{500}$ which are mean subtracted and normalized to have Euclidean norm 1. $C = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$ denotes covariance matrix, $b \in \mathbb{R}^d$ denotes a random vector and we let $\mu = 100$, $\lambda = 10^{-4}$ in the experiment. Because each ϕ_i is probably non-convex, CoCoA is not able to solve this problem. In this experiment, we compare with Distributed asynchronous SVRG [19].

In Figure 4, it is obvious that Dis-dfSDCA runs faster than Distributed SVRG when there are 4 workers. We can observe the similar phenomenon when there are 8 workers. This observation is reasonable because Distributed SVRG needs to compute two gradients in each inner iteration and full gradient in each outer iteration. Dis-dfSDCA is faster because it only



(a) IJCNN1 (b) COVTYPE (c) RCV1

Fig. 2: Figures (a) - (c) present the convergence of duality gap of compared methods in terms of time. Figures (d) - (f) present the convergence of duality gap of compared methods in terms of epoch number. We train IJCNN1 dataset with 4 workers, COVTYPE dataset with 8 workers and RCV1 dataset with 16 workers.

needs to compute one gradient in each iteration. However, Dis-dfSDCA needs $O(nd)$ space for storing α , because $\nabla\phi_i(w)$ cannot be written as $\nabla\phi_i(x_i^T w)x_i$ in this problem.

VI. Conclusion

In this paper, we proposed Distributed Asynchronous Dual Free Coordinate Ascent (Dis-dfSDCA) method for distributed machine learning. We addressed two challenging issues in previous primal-dual distributed optimization methods: firstly, Dis-dfSDCA does not rely on the dual formulation, and can be used to solve the non-convex problem; secondly, Dis-dfSDCA uses asynchronous communication and can be applied on the complicated distributed system where there is straggler problem. We also analyze the convergence rate of Dis-dfSDCA and prove linear convergence even if the loss functions are non-convex, as long as the sum of non-convex objectives is convex. We conduct experiments on the simulated distributed system with straggler problem, and all experimental results consistently verify our theoretical analysis.

Appendix

Proof to Lemma 1

Proof sketch: In our proof, we suppose that there are no duplicate samples in I_t. According to our algorithm, we know that:

$$v^t = \sum_{i\in I_t}\left(\nabla\phi_i(w^{d(t)}) + \alpha_i^{d(t)}\right) = \sum_{i\in I_t} v_i^t \tag{39}$$

where $|I_t| = H$ and $\mathbb{E}[v_i^t] = \nabla P(w^{d(t)})$. Following the proof in [11], we define A_t and B_t as follows:

$$A_t = \mathbb{E}\|\alpha_i^t - \alpha_i^*\|^2 \tag{40}$$
$$B_t = \mathbb{E}\|w^t - w^*\|^2 \tag{41}$$

Defining $\beta = \eta\lambda n$, so in iteration t, $\alpha_i^{t+1} = (1-\beta)\alpha_i^t + \beta(-\nabla\phi_i(w^{d(t)}))$, we have:

$$\begin{aligned}
&\mathbb{E}[A_{t+1} - A_t] \\
=\ &\mathbb{E}[\frac{1}{n}\sum_{i\in I_t}\|(1-\beta)(\alpha_i^t - \alpha_i^*) + \beta(-\nabla\phi_i(w^{d(t)}) - \alpha_i^*)\|^2 \\
&-\frac{1}{n}\|\alpha_i^t - \alpha_i^*\|^2] \\
=\ &\mathbb{E}\Big[\frac{1}{n}\sum_{i\in I_t}\Big((1-\beta)\|\alpha_i^t - \alpha_i^*\|^2 + \beta\|\nabla\phi_i(w^{d(t)}) + \alpha_i^*\|^2 \\
&-\beta(1-\beta)\|\alpha_i^t + \nabla\phi_i(w^{d(t)})\|^2 - \|\alpha_i^t - \alpha_i^*\|^2\Big)\Big] \\
=\ &\eta\lambda H\Big(-\mathbb{E}\|\alpha_i^t - \alpha_i^*\|^2 + \mathbb{E}\|\nabla\phi_i(w^{d(t)}) + \alpha_i^*\|^2\Big) \\
&-\eta\lambda(1-\beta)\sum_{i\in I_t}\mathbb{E}\|v_i^t\|^2 \\
\le\ &\eta\lambda H\Big(-\mathbb{E}\|\alpha_i^t - \alpha_i^*\|^2 + \mathbb{E}\|\nabla\phi_i(w^{d(t)}) + \alpha_i^*\|^2\Big) \\
&-\eta\lambda(1-\beta)\mathbb{E}\|v^t\|^2
\end{aligned} \tag{42}$$

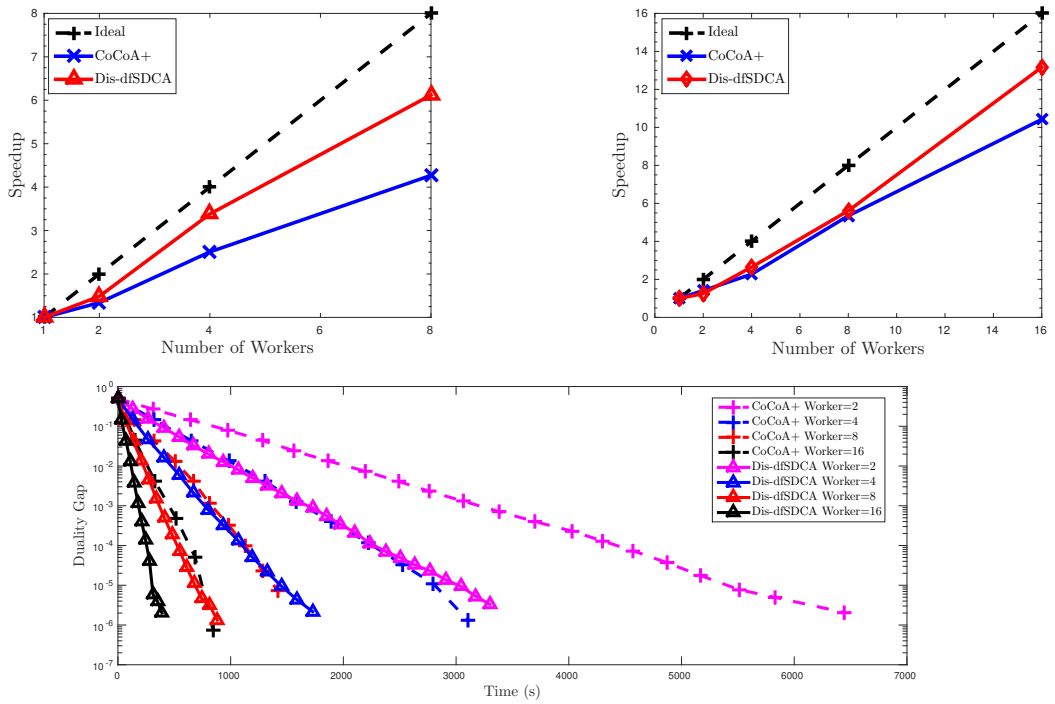


Fig. 3: Time speedup in terms of the number of workers. Row 1 left: IJCNN1; Row 1 right: COVTYPE; Row 2: RCV1.

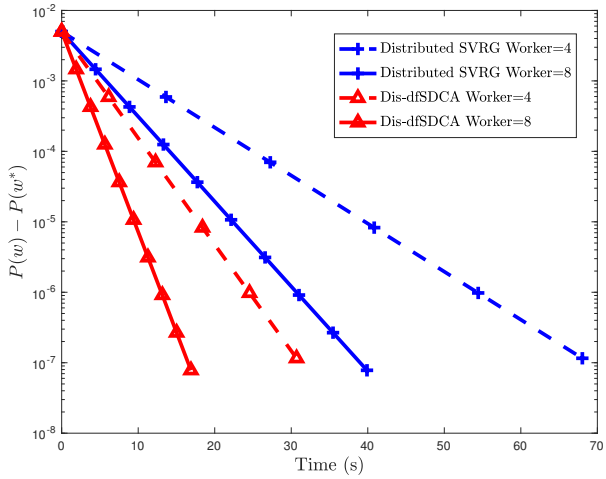


Fig. 4: Suboptimum ($P(w)-P(w^*)$) convergence of compared methods in terms of time. w^* denotes the optimal solution to problem (38) , and it is obtained by running Dis-dfSDCA until convergence.

where the last inequality follows from that $\sum_{i\in I_t}\mathbb{E}\|v_i^t\|^2 \geq \mathbb{E}\|\sum_{i\in I_t} v_i^t\|^2 = \mathbb{E}\|v^t\|^2$. Because $\alpha_i^* = -\nabla\phi_i(w^*)$, we have the following inequality:

$$\begin{aligned}
& \mathbb{E}\|\nabla\phi_i(w^{d(t)}) + \alpha^*\|^2 \\
= \;& \mathbb{E}[\|\ \nabla\phi_i(w^{d(t)}) - \nabla\phi_i(w^*)\|^2] \\
\leq \;& 2\mathbb{E}\|\nabla\phi_i(w^{d(t)}) - \nabla\phi_i(w^t)\|^2 + 2\mathbb{E}\|\nabla\phi_i(w^t) - \nabla\phi_i(w^*)\|^2 \\
\leq \;& 2L^2\mathbb{E}\|w^t - w^{d(t)}\|^2 + 2\mathbb{E}\|\nabla\phi_i(w^t) - \nabla\phi_i(w^*)\|^2 \\
\leq \;& 2L^2\eta^2\mathbb{E}\|\sum_{j=d(t)}^{t-1} v^j\|^2 + 4L\mathbb{E}\Big(P(w^t) - P(w^*) - \frac{\lambda}{2}\|w^t - w^*\|^2\Big) \\
\leq \;& 2L^2\eta^2\tau\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 + 4L\mathbb{E}\Big(P(w^t) - P(w^*) - \frac{\lambda}{2}\|w^t - w^*\|^2\Big)
\end{aligned}$$

where the first inequality follows from Lemma 4, the second inequality follows from Lemma 5, the third and the last inequalities follow from the Assumption 2. In addition, it also follows that:

$$\begin{aligned}
\mathbb{E}[B_{t+1} - B_t] &= \mathbb{E}\|w^{t+1} - w^*\|^2 - \mathbb{E}\|w^t - w^*\|^2 \\
&= -2\eta\mathbb{E}\left\langle w^t - w^*, v^t\right\rangle + \eta^2\mathbb{E}\|v^t\|^2 \quad (43)
\end{aligned}$$

We can know $\mathbb{E}\left\langle w^t - w^*, v^t\right\rangle$ is lower bounded that:

$$\begin{aligned}
& \mathbb{E}\left\langle w^t - w^*, v^t\right\rangle \\
= \;& \sum_{i\in I_t}\mathbb{E}\left\langle w^{d(t)} - w^*, v_i^t\right\rangle + \sum_{i\in I_t}\mathbb{E}\left\langle w^t - w^{d(t)}, v_i^t\right\rangle \\
= \;& H\left\langle w^{d(t)} - w^*, \nabla P(w^{d(t)})\right\rangle + H\left\langle w^t - w^{d(t)}, \nabla P(w^{d(t)})\right\rangle \\
\geq \;& H\Big(P(w^{d(t)}) - P(w^*)\Big) + H\left\langle w^t - w^{d(t)}, \nabla P(w^{d(t)})\right\rangle \quad (44)
\end{aligned}$$

where the equality follows form that v_i^t is not relevant to the variable before w^{t+1} and the inequality follows from the

convexity of $P(w)$. □

Proof to Theorem 1

Proof sketch: We define $C_{t+1} = c_a A_{t+1} + c_b B_{t+1}$ and set $c_a = \frac{1}{2\lambda L}$, $c_b = 1$. Inputting Lemma 1 in equation, we have:

$$\begin{aligned}
&\mathbb{E}[C_{t+1}] = c_a A_{t+1} + c_b B_{t+1} \\
\leq\ & c_a(1-\eta\lambda H)\mathbb{E}\|\alpha_i^t - \alpha_i^*\|^2 + 2c_a\lambda\tau HL^2\eta^3 \cdot \\
& \sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 - c_a\eta\lambda(1-\beta)\mathbb{E}\|v^t\|^2 \\
& +4c_a\eta\lambda HL\bigg(P(w^t) - P(w^*) - \frac{\lambda}{2}\mathbb{E}\|w^t - w^*\|^2]\bigg) \\
& +c_b\eta^2\mathbb{E}\|v^t\|^2 - 2c_b\eta\bigg(H\left(P(w^{d(t)}) - P(w^*)\right) \\
& +H\left\langle w^t - w^{d(t)}, \nabla P(w^{d(t)})\right\rangle\bigg) + c_b\mathbb{E}\|w^t - w^*\|^2 \\
\leq\ & (1-\eta\lambda H)\mathbb{E}[C_t] + 2\eta H \cdot \\
& \left(P(w^t) - P(w^{d(t)}) - \left\langle w^t - w^{d(t)}, \nabla P(w^{d(t)})\right\rangle\right) \\
& +\left(\frac{\eta^2\lambda n}{2L} + \eta^2 - \frac{\eta}{2L}\right)\mathbb{E}\|v^t\|^2 + \tau HL\eta^3\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \\
\leq\ & (1-\eta\lambda H)\mathbb{E}[C_t] + \left(\frac{\eta^2\lambda n}{2L} + \eta^2 - \frac{\eta}{2L}\right)\mathbb{E}\|v^t\|^2 \\
& +2\tau HL\eta^3\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \qquad (45)
\end{aligned}$$

where the last inequality follows from the L-smooth of $P(w)$:

$$\begin{aligned}
P(w^t) \leq\ & P(w^{d(t)}) + \left\langle w^t - w^{d(t)}, \nabla P(w^{d(t)})\right\rangle \\
& +\frac{L}{2}\|w^t - w^{d(t)}\|^2 \\
\leq\ & P(w^{d(t)}) + \left\langle w^t - w^{d(t)}, \nabla P(w^{d(t)})\right\rangle \\
& +\frac{L\tau\eta^2}{2}\sum_{j=d(t)}^{t-1}\|v^j\|^2 \qquad (46)
\end{aligned}$$

Adding the above inequality from $t = 0$ to $t = T-1$, we have that:

$$\begin{aligned}
&\sum_{t=0}^{T-1}\mathbb{E}[C_{t+1}] \\
\leq\ & \sum_{t=0}^{T-1}(1-\eta\lambda H)\mathbb{E}[C_t] + \left(\frac{\eta^2\lambda n}{2L} + \eta^2 - \frac{\eta}{2L}\right)\sum_{t=0}^{T-1}\mathbb{E}\|v^t\|^2 \\
& +2\tau HL\eta^3\sum_{t=0}^{T-1}\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \\
\leq\ & \sum_{t=0}^{T-1}(1-\eta\lambda H)\mathbb{E}[C_t] \\
& +\left(2H\tau^2\eta^2 + \frac{\eta^2\lambda n}{2L} + \eta^2 - \frac{\eta}{2L}\right)\sum_{t=0}^{T-1}\mathbb{E}\|v^t\|^2 \qquad (47)
\end{aligned}$$

where the last inequality follows from Assumption 2 and $\eta L \leq 1$. If $2H\eta^2\tau^2 + \frac{\eta^2\lambda n}{2L} + \eta^2 - \frac{\eta}{2L} \leq 0$ such that:

$$\eta \leq \frac{1}{4HL\tau^2 + \lambda n + 2L} \qquad (48)$$

Therefore, we have:

$$\begin{aligned}
\sum_{t=0}^{T-1}\mathbb{E}[C_{t+1}] \leq\ & \sum_{t=0}^{T-1}(1-\eta\lambda H)\mathbb{E}[C_t] \\
\leq\ & \sum_{t=1}^{T-1}\mathbb{E}[C_t] + (1-\eta\lambda H)C_0 \qquad (49)
\end{aligned}$$

We complete the proof. □

Proof to Lemma 2

Proof sketch: As per the smoothness of ϕ_i, we have:

$$\begin{aligned}
& \mathbb{E}\|\nabla\phi_i(w^{d(t)}) + \alpha_i^*\|^2 \\
=\ & \mathbb{E}\|\nabla\phi_i(w^{d(t)}) - \nabla\phi_i(w^*)\|^2 \\
\leq\ & L^2\mathbb{E}\|w^{d(t)} - w^*\|^2 \\
\leq\ & 2L^2\mathbb{E}\|w^{d(t)} - w^t\|^2 + 2L^2\mathbb{E}\|w^t - w^*\|^2 \\
\leq\ & 2L^2\eta^2\tau\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 + 2L^2\mathbb{E}\|w^t - w^*\|^2 \qquad (50)
\end{aligned}$$

We can also bound $-\mathbb{E}\left\langle w^t - w^*, v^t\right\rangle$ as follows:

$$\begin{aligned}
& -\mathbb{E}\left\langle w^t - w^*, v^t\right\rangle \\
=\ & -H\mathbb{E}\left\langle w^t - w^*, \nabla P(w^{d(t)})\right\rangle \\
=\ & -H\mathbb{E}\left\langle w^t - w^*, \nabla P(w^t)\right\rangle \\
& -H\mathbb{E}\left\langle w^t - w^*, \nabla P(w^{d(t)}) - \nabla P(w^t)\right\rangle \\
\leq\ & -\lambda H\mathbb{E}\|w^t - w^*\|^2 + \frac{\gamma H}{2}\mathbb{E}\|w^t - w^*\|^* \\
& +\frac{H}{2\gamma}\mathbb{E}\|\nabla P(w^t) - \nabla P(w^{d(t)})\|^2 \\
\leq\ & -(\lambda - \frac{\gamma}{2})H\mathbb{E}\|w^t - w^*\|^2 \\
& +\frac{H(L+\lambda)^2\eta^2\tau}{2\gamma}\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2
\end{aligned}$$

where the first inequality follows from the strong convexity of P such that $\langle w^t - w^*, \nabla P(w^t)\rangle \geq P(w^t) - P(w^*) + \frac{\lambda}{2}\|w^t - w^*\|^2$ and $P(w^t) - P(w^*) \geq \frac{\lambda}{2}\|w^t - w^*\|^2$. Defining $\gamma = \frac{\lambda}{2}$ and substituting above two inequalities into (42) and (43) respectively, we complete the proof. □

Proof to Theorem 3

Proof sketch: We define $C_{t+1} = c_a A_{t+1} + c_b B_{t+1}$ and set $c_a = \frac{1}{4L^2}$, $c_b = 1$. Inputting Lemma 2 in the equation, we have:

$$
\begin{aligned}
&\mathbb{E}[C_{t+1}] = c_a A_{t+1} + c_b B_{t+1} \\
\leq\ & c_a(1-\eta\lambda H)\mathbb{E}\|\alpha_i^t - \alpha_i^*\|^2 \\
&+(c_b + 2c_a\eta\lambda HL^2 - 2c_b\eta H(\lambda - \frac{\gamma}{2}))\mathbb{E}\|w^t - w^*\|^2 \\
&+ \left(c_b\eta^2 - c_a\eta\lambda(1-\beta)\right)\mathbb{E}\|v^t\|^2 \\
&+ \left(2c_a\lambda H\tau L^2\eta^3 + c_b\frac{H\tau(L+\lambda)^2\eta^3}{\gamma}\right)\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \\
=\ & (1-\eta\lambda H)\mathbb{E}[C_t] + \left(c_b\eta^2 - c_a\eta\lambda(1-\beta)\right)\mathbb{E}\|v^t\|^2 \\
&+ \left(2c_a\lambda H\tau L^2\eta^3 + c_b\frac{H\tau(L+\lambda)^2\eta^3}{\gamma}\right)\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2
\end{aligned}
$$

where $\gamma = \frac{\lambda}{2}$. Adding the above inequality from $t = 0$ to $t = T-1$, we have that:

$$
\begin{aligned}
&\sum_{t=0}^{T-1}\mathbb{E}[C_{t+1}] \\
\leq\ & (1-\eta\lambda H)\sum_{t=0}^{T-1}\mathbb{E}[C_t] + \left(c_b\eta^2 - c_a\eta\lambda(1-\beta)\right)\sum_{t=0}^{T-1}\mathbb{E}\|v^t\|^2 \\
&+ \left(2c_a\lambda H\tau L^2\eta^3 + c_b\frac{H\tau(L+\lambda)^2\eta^3}{\gamma}\right)\sum_{t=0}^{T-1}\sum_{j=d(t)}^{t-1}\mathbb{E}\|v^j\|^2 \\
\leq\ & (1-\eta\lambda H)\sum_{t=0}^{T-1}\mathbb{E}[C_t] + (c_b\eta^2 - c_a\eta\lambda(1-\beta) \\
&+2c_a\lambda H\tau^2 L^2\eta^3 + c_b\frac{H\tau^2(L+\lambda)^2\eta^3}{\gamma})\sum_{t=0}^{T-1}\mathbb{E}\|v^t\|^2 \\
\leq\ & \sum_{t=1}^{T-1}\mathbb{E}[C_t] + (1-\eta\lambda H)\mathbb{E}[C_0] \qquad (51)
\end{aligned}
$$

where the last inequality holds as long as:

$$
\eta \leq \frac{\lambda^2}{2HL\tau^2\lambda^2 + 8HL\tau^2(L+\lambda)^2 + 4\lambda L^2 + n\lambda^3} \qquad (52)
$$

such that $2c_a\lambda HL^2\tau^2\eta^3 + c_b\frac{H(L+\lambda)^2\tau^2\eta^3}{\gamma} + c_b\eta^2 - c_a\eta\lambda(1-\beta) \leq 0$. We complete the proof. □

EXTRA LEMMAS

Lemma 3 ([20]): For random variables $z_1, ..., z_r$ are independent and mean 0, we have:

$$
\mathbb{E}[\|z_1 + ... + z_r\|^2] \quad = \quad \mathbb{E}[\|z_1\|^2 + ... + \|z_r\|^2] \qquad (53)
$$

Lemma 4: For any $z_1, ..., z_r$, it holds that:

$$
\|z_1 + ... + z_r\|^2 \quad \leq \quad r(\|z_1\|^2 + ... + \|z_r\|^2) \qquad (54)
$$

Lemma 5 ([11]): Assume that each $\phi_i(w)$ is L-smooth and convex. Then, for every w,

$$
\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n}\|\nabla\phi_i(w) - \nabla\phi_i(w^*)\|^2 \\
\leq\ & 2L\left(P(w) - P(w^*) - \frac{\lambda}{2}\|w - w^*\|^2\right) \qquad (55)
\end{aligned}
$$

REFERENCES

[1] D. Garber and E. Hazan, "Fast and simple pca via convex optimization," *arXiv preprint arXiv:1509.05647*, 2015.

[2] Z. Allen-Zhu and Y. Yuan, "Improved svrg for non-strongly-convex or sum-of-non-convex objectives," in *International conference on machine learning*, 2016, pp. 1080–1089.

[3] M. Jaggi, V. Smith, M. Takác, J. Terhorst, S. Krishnan, T. Hofmann, and M. I. Jordan, "Communication-efficient distributed dual coordinate ascent," in *Advances in Neural Information Processing Systems*, 2014, pp. 3068–3076.

[4] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan, "A dual coordinate descent method for large-scale linear svm," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 408–415.

[5] S. Shalev-Shwartz and T. Zhang, "Accelerated mini-batch stochastic dual coordinate ascent," in *Advances in Neural Information Processing Systems*, 2013, pp. 378–385.

[6] ——, "Stochastic dual coordinate ascent methods for regularized loss," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 567–599, 2013.

[7] T. Yang, "Trading computation for communication: Distributed stochastic dual coordinate ascent," in *Advances in Neural Information Processing Systems*, 2013, pp. 629–637.

[8] M. Takáč, P. Richtárik, and N. Srebro, "Distributed mini-batch sdca," *arXiv preprint arXiv:1507.08322*, 2015.

[9] C. Ma, V. Smith, M. Jaggi, M. I. Jordan, P. Richtárik, and M. Takáč, "Adding vs. averaging in distributed primal-dual optimization," *arXiv preprint arXiv:1502.03508*, 2015.

[10] C. Ma, J. Konečnỳ, M. Jaggi, V. Smith, M. I. Jordan, P. Richtárik, and M. Takáč, "Distributed optimization with arbitrary local solvers," *Optimization Methods and Software*, pp. 1–36, 2017.

[11] S. Shalev-Shwartz, "Sdca without duality," *arXiv preprint arXiv:1502.06177*, 2015.

[12] ——, "Sdca without duality, regularization, and individual convexity," in *International Conference on Machine Learning*, 2016, pp. 747–754.

[13] X. He and M. Takáč, "Dual free sdca for empirical risk minimization with adaptive probabilities," *arXiv preprint arXiv:1510.06684*, 2015.

[14] H. Lin, J. Mairal, and Z. Harchaoui, "A universal catalyst for first-order optimization," in *Advances in Neural Information Processing Systems*, 2015, pp. 3384–3392.

[15] Z. Qu and P. Richtárik, "Stochastic dual coordinate ascent with adaptive probabilities," 2015.

[16] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, R. H. Castain, D. J. Daniel, R. L. Graham, and T. S. Woodall, "Open MPI: Goals, concept, and design of a next generation MPI implementation," in *Proceedings, 11th European PVM/MPI Users' Group Meeting*, Budapest, Hungary, September 2004, pp. 97–104.

[17] C. Sanderson and R. Curtin, "Armadillo: a template-based c++ library for linear algebra," *Journal of Open Source Software*, 2016.

[18] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[19] Z. Huo and H. Huang, "Asynchronous mini-batch gradient descent with variance reduction for non-convex optimization." in *AAAI*, 2017, pp. 2043–2049.

[20] S. J. Reddi, S. Sra, B. Poczos, and A. Smola, "Fast stochastic methods for nonsmooth nonconvex optimization," *arXiv preprint arXiv:1605.06900*, 2016.