Network Constrained Distributed Dual Coordinate Ascent for Machine Learning

Myung Cho *1, Lifeng Lai †2 and Weiyu Xu ‡3

1Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27606
2Department of Electrical and Computer Engineering, University of California, Davis, CA 95616
3Department of Electrical and Computer Engineering, University of Iowa, Iowa City, IA 52242

October 31, 2018

Abstract

With explosion of data size and limited storage space at a single location, data are often distributed at different locations. We thus face the challenge of performing large-scale machine learning from these distributed data through communication networks. In this paper, we study how the network communication constraints will impact the convergence speed of distributed machine learning optimization algorithms. In particular, we give the convergence rate analysis of the distributed dual coordinate ascent in a general tree structured network. Furthermore, by considering network communication delays, we optimize the network-constrained dual coordinate ascent to maximize its convergence speed. From numerical experiments, we demonstrate that under different network communication delays, one needs to adopt delay-dependent numbers of local and global iterations for distributed dual coordinate ascent to achieve maximum convergence speed.

1 Introduction

In modern society, the abundance of sensors enables data to be ubiquitously collected at an astonishing speed. The amount of data that we can access and learn actionable information from is skyrocketing. This propels our society into an era of big data. “Big data” is dramatically impacting our everyday lives, and almost every aspect of our society, including education, science, engineering, finance, healthcare, and management [1]. The Google flu trend service is one example of using big data for better healthcare services [2].

To make use of big data, one often uses large-scale convex optimizations to learn actionable information from these big data [3]. However, very often data are collected and stored at different locations. Due to the constraints of limited storage volumes and network communication bandwidths, sometimes it is not possible to pool these distributed data at one central location, and then perform centralized machine learning over these data. This necessitates performing machine learning in a distributed manner.

Solving large scale optimization problems dealing with distributed big data is an on-going challenge, due to various obstacles such as the size of data and the distributed nature of data. In order
to answer the challenge of distributed data, researchers have studied various optimization methods such as synchronous Stochastic Gradient Decent (SGD) \[3,4\], synchronous Stochastic Dual Coordinate Ascent (SDCA) \[5–7\], asynchronous SGD \[8,9\], and asynchronous SDCA \[10,11\]. Even though the convergence of SGD does not depend on the size of data, it is reported in \[12\] that SDCA can outperform SGD when we need relatively high solution accuracy. Furthermore, asynchronous updating scheme can suffer from the conflicts between intermediate results.

Motivated by these facts, the authors in \[5–7\] considered a synchronous distributed dual coordinate ascent for solving regularized loss minimization problems in a star network. In this star network, data are distributed over a few local workers, which can individually communicate with a central station. In \[5–7\], the authors derived the convergence rate of the distributed dual coordinate ascent with respect to the number of iterations. The proposed distributed optimization framework in \[6,7\] is free of tuning parameters or learning rates, compared with SGD-based methods. Moreover, the duality gap in \[6,7\] readily provides a fair stopping criterion and efficient accuracy certificates. In \[10,11\], the authors investigated the performance of asynchronous SDCA.

In practical communication networks, computers are not always organized in a star work, and can have very different network topologies. It is unclear how to design and analyze dual coordinate ascent algorithms for a network with general topologies. In addition, it is unknown how network communication delays (not merely the number of communication rounds) will affect the design and convergence rate of distributed dual coordinate ascent algorithms \[5–7\]. We remark that, in \[13\], the authors considered communication delays and provided the convergence bound in terms of time for consensus based distributed optimization.

In this paper, we study how the network communication constraints will impact the convergence speed of distributed dual coordinate ascent optimization algorithms. Firstly, we consider the design and analysis of distributed dual coordinate ascent algorithms for regularized loss minimization, in a general tree structured network. We give the convergence rate analysis of the distributed dual coordinate ascent for the considered tree network. Secondly, by considering network communication delays, we optimize the network-constrained dual coordinate ascent to maximize its convergence speed. Our numerical experiments show that under different network communication delays, one needs to adopt delay-dependent number of local iterations for distributed dual coordinate ascent to achieve maximum convergence speed.

**Notations:** We denote the set of real numbers as \( \mathbb{R} \). We use \([k]\) to denote the index set of the coordinates in the \(k\)-th coordinate block. For an index set \( Q \), we use \( \overline{Q} \) and \( |Q| \) to represent the complement and the cardinality of \( Q \) respectively. We use bold letters to represent vectors. If we use an index set as a subscript of a vector (matrix), we refer to the partial vector (partial matrix) over the index set (with columns over the index set). The superscript \((t)\) is used to denote the \(t\)-th iteration. For example, \( \alpha_{[k]}(t) \) represents a partial vector \( \alpha \) over the \(k\)-th block coordinate at the \(t\)-th iteration. We use the superscript \( \star \) to denote the optimal solution.

## 2 Problem setup

We have the following regularized loss minimization problem for machine learning applications \[5,7,10,11,14\]:

\[
\minimize_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \| \mathbf{w} \|^2 + \frac{1}{m} \sum_{i=1}^{m} \ell_i(\mathbf{w}^T \mathbf{x}_i), \tag{1}
\]

where \( \mathbf{x}_i \in \mathbb{R}^d, i = 1, \ldots, m \), are dataset, \( \ell_i(\cdot) \), \( i = 1, \ldots, m \), are loss functions, and \( \lambda \) is the regularization parameter. Depending on the loss functions, one can consider \[11\] as various machine learning
problems ranging from regression to classification. For example, if the loss function is the hinge loss function, the optimization problem with labeled dataset \( \{(x_i, y_i)\}, \) \( i = 1, ..., m, \) where \( y_i \in \mathbb{R} \) is label information, is the Support Vector Machine (SVM).

By using the conjugate function, i.e., \( \ell_i(a) = \sup_b ab - \ell_i^*)(b), \) where \( a, b \in \mathbb{R} \) and \( \ell_i(\cdot) \) is convex, we can obtain the dual problem of (1) as follows:

\[
\text{maximize } D(\alpha) \triangleq \frac{\lambda}{2} \|A\alpha\|^2 - \frac{1}{m} \sum_{i=1}^{m} \ell_i^*(-\alpha_i), \tag{2}
\]

where \( \alpha_i \) is the \( i \)-th element of the dual vector \( \alpha, \) and the data matrix \( A \in \mathbb{R}^{d \times m} \) has the normalized training data \( \frac{1}{\lambda_m} x_i \) in its \( i \)-th column; namely \( A_i = \frac{1}{\lambda_m} x_i. \) Since we have the primal-dual relationship as \( w(\alpha) \triangleq A\alpha, \) we have the duality gap as \( P(w(\alpha)) - D(\alpha). \)

In this paper, we consider a distributed dual coordinate ascent for the regularized loss minimization problem over distributed data in a network of computers. Let us review the previous research on the distributed dual coordinate ascent in a star network in the following section.

3 Review of distributed dual coordinate ascent in a star network

The authors in [5, 7, 10] consider a star network as shown in Fig. 1 and assume that each local worker has disjoint parts of dataset. Specifically, the \( k \)-th local worker has training data \( \{(x_i, y_i)\}, \) \( i \in [k], \) where \( [k] \) is the index set for the training data of the \( k \)-th local worker. Hence, if the star network has \( K \) local workers, \( \bigcup_{k=1}^{K} [k] = m. \) In [6], the author introduced Algorithm 1 for the distributed dual coordinate ascent. In Algorithm 1, LocalDualMethod(\cdot) represents any dual method to solve (2). The Stochastic Dual Coordinate Ascent (SDCA), denoted by LocalSDCA(\cdot), is a possible candidate for LocalDualMethod(\cdot) [6]. The convergence rate of the algorithm is given as follows [6].

**Theorem 1** ( [6 Theorem 2] ). Assume that Algorithm 1 is run for \( T \) outer iterations of \( K \) local computers, with the procedure LocalSDCA(\cdot) having local geometric improvement \( \Theta. \) Further, assume the loss functions \( \ell_i(\cdot) \) are \( 1/\gamma \)-smooth. Then, the following geometric convergence rate holds for the
Figure 1: Illustration of a distributed star network, where $W_i$, $i = 1, 2, 3$, are local workers.

**Global (dual) objective:**

$$
E[D(\alpha^*) - D(\alpha^{(T)})] \leq \left(1 - (1 - \Theta) \frac{1}{K} \frac{\lambda m \gamma}{\rho + \lambda m \gamma}\right)^T \left(D(\alpha^*) - D(\alpha^{(0)})\right),
$$

(3)

where $\rho$ is any real number satisfying

$$
\rho \geq \rho_{\min} = \max_{\alpha \in \mathbb{R}^m} \lambda^2 m^2 \sum_{k=1}^K \frac{\|A[k]\alpha[k]\|^2 - \|A\alpha\|^2}{\|\alpha\|^2} \geq 0.
$$

For LocalSDCA($\cdot$), $\Theta$ can be set to the following value with $s \in [0, 1]$:

$$
\Theta = \left(1 - \frac{s}{\tilde{m}}\right)^H,
$$

(4)

where $\tilde{m} = \max_{k=1,\ldots,K} m_k$ is the size of the largest block of coordinates among $K$ local workers, $H$ is the number of local (or inner) iterations in LocalSDCA($\cdot$), and $s \in [0, 1]$ is a step size which determines how far the next solution will be from the current solution at each iteration.

Since a star network that the previous research considered is a simple network model, and a network of computers can have various topologies, we study the distributed dual coordinate ascent in a generalized network, which is a tree structured network model.

4 Generalized distributed coordinate ascent in tree-structured networks

Earlier works [5][6] provide the convergence analysis of distributed dual coordinate ascent algorithms in a star network as illustrated in Fig. 1. However, the communication network connecting different local workers is not necessarily a simple star network, but instead can be an arbitrary undirected connected graph. The design and analysis of the distributed dual coordinate ascent algorithm in a communication network of general topology is not well understood [5]. One may argue that, in a connected communication network, we can always form a virtual star network by connecting local workers to a central station through the relays of other computers. However, the communication delay from one particular local worker to the central station can be very large (long relays), significantly slowing down the convergence of the distributed learning algorithm. Thus it is necessary to spend more computational resources on performing distributed optimization among local workers close to each other first, before communicating intermediate computational results to a central station.
Figure 2: A tree-structured network, which has two layers. In the network, a central station (root node) has three direct child nodes $S_1$, $S_2$ and $S_3$. Each node $S_i$ has three direct local workers $W_{ij}$, $j = 1, 2, 3$.

Motivated by these network constraints, in this section, we investigate the design and analysis of a recursive distributed dual coordinate ascent algorithm over a general tree structured network, instead of a simple star network. We choose to investigate a tree network, because every connected communication network has a spanning tree. In addition, the tree structured network is a generalization of the star network.

We first describe a general tree network, with a 2-layer tree network example illustrated in Fig. 2. In the considered tree network, the root node corresponds to the central station. Any other tree node corresponds to a local worker. Each tree node may have several direct child nodes. Without loss of generality, we assume that only the local workers corresponding to the leaf nodes have access to the distributed data, namely disjoint segmented blocks of the data matrix $A$ (In fact, if a non-leaf node $Q$ stores data, one can create a virtual leaf node $L$ attached to $Q$, and “stores” data in $L$). For a tree node $Q$, we also use $Q$ to denote the set of indices of data points stored in the subtree with $Q$ as the root node (the subtree includes $Q$, indirect and direct child nodes of $Q$). We use $[Q, k]$ to denote the set of indices of data points stored in the subtree whose root node is the $k$-th direct child node of $Q$. If $Q$ is a leaf node, we use $m_Q$ to denote the number of data points stored in $Q$.

In a tree network, a node can only communicate with its child nodes or parent nodes.

We are ready to introduce the generalized distributed dual coordinate ascent algorithm (which we call TreeDualMethod) for solving (2) dealing with data stored in a general tree structure network. Algorithm 2, Algorithm 3 and Procedure 4 describe respectively the computational steps of TreeDualMethod for a general tree node (not root or leaf), the root node, and a leaf node. In the following section, we provide the convergence analysis of the generalized distributed dual coordinate ascent algorithm in the tree structured network model.

5 Convergence analysis of TreeDualMethod for a tree network

In this section, we show that for a tree network, there is a recursive relation between the convergence rate of the algorithm at a tree node $Q$ and the convergence rate of the algorithm at $Q$’s direct child nodes.

We assume that $Q$ has $K$ direct child nodes, and denote the dual variable vector corresponding
Let us consider a tree node $Q$. We assume that there exists $\Theta \in [0, 1)$ such that for any given $\alpha$, TreeDualMethod for $Q$’s $k$-th direct child node returns an update $\Delta \alpha_{[Q,k]}$ such that

$$E[\epsilon_{Q,k}((\alpha_{[Q,1]}, \ldots, \alpha_{[Q,k]-1}, \alpha_{[Q,k]} + \Delta \alpha_{[Q,k]}, \ldots, \alpha_{[Q,K]}, \alpha_{Q}))] \leq \Theta \cdot \epsilon_{Q,k}(\alpha).$$

(6)

Note that the suboptimality gap for the $k$-th child node of $Q$ is defined when $\alpha_{[Q,i]}$’s ($i \neq k$) and $\alpha_{Q}$ are fixed. We further assume that we have the following local geometric improvement for the $k$-th direct child node of $Q$.

**Assumption 1** (Direct child node geometric improvement of TreeDualMethod). *Let us consider a tree node $Q$. We assume that there exists $\Theta \in [0, 1)$ such that for any given $\alpha$, TreeDualMethod for $Q$’s $k$-th direct child node returns an update $\Delta \alpha_{[Q,k]}$ such that*

$$E[\epsilon_{Q,k}((\alpha_{[Q,1]}, \ldots, \alpha_{[Q,k]-1}, \alpha_{[Q,k]} + \Delta \alpha_{[Q,k]}, \ldots, \alpha_{[Q,K]}, \alpha_{Q}))] \leq \Theta \cdot \epsilon_{Q,k}(\alpha).$$

(6)

For a leaf node, TreeDualMethod uses LocalSDCA in [4] as described in Procedure [4]. We remark that this geometric improvement condition holds true if the $k$-th direct child node of $Q$ is a leaf

**Algorithm 2: TreeDualMethod: General Distributed Dual Coordinate Ascent for a General Tree Node $Q$ (not root or leaf)**

**Input:** $T \geq 1$, $\alpha_Q$, $w$

**Initialization:** $\alpha^{(0)}_{[Q,k]} \leftarrow \alpha_{[Q,k]}$ for all direct child nodes $k$ of node $Q$, $w^{(0)} \leftarrow w$

**for** $t = 1$ **to** $T$

***for*** all direct child nodes $k = 1, 2, \ldots, K$ of $Q$ **in parallel** **do**

$(\Delta \alpha_{[Q,k]}, \Delta w_k) \leftarrow$ TreeDualMethod($\alpha^{(t-1)}_{[Q,k]}, w^{(t-1)}$)

$\alpha^{(t)}_{[Q,k]} \leftarrow \alpha^{(t-1)}_{[Q,k]} + \frac{1}{K} \Delta \alpha_{[Q,k]}$

end

$w^{(t)} \leftarrow w^{(t-1)} + \frac{1}{K} \sum_{k=1}^{K} \Delta w_k$

**Output:** $\Delta \alpha_Q \triangleq \alpha^{(T)}_Q - \alpha^{(0)}_Q$, and $\Delta w_Q \triangleq w^{(T)} - w^{(0)} = A_Q \Delta \alpha_Q$

**end**

**Algorithm 3: TreeDualMethod: General Distributed Dual Coordinate Ascent for the Root Node $Q$**

**Input:** $R \geq 1$

**Initialization:** $\alpha^{(0)}_{[Q,k]} \leftarrow 0$ for all direct child nodes $k$ of node $Q$, $w^{(0)} \leftarrow 0$

**for** $t = 1$ **to** $R$

***for*** all direct child nodes $k = 1, 2, \ldots, K$ **in parallel** **do**

$(\Delta \alpha_{[Q,k]}, \Delta w_k) \leftarrow$ TreeDualMethod($\alpha^{(t-1)}_{[Q,k]}, w^{(t-1)}$)

$\alpha^{(t)}_{[Q,k]} \leftarrow \alpha^{(t-1)}_{[Q,k]} + \frac{1}{K} \Delta \alpha_{[Q,k]}$

end

$w^{(t)} \leftarrow w^{(t-1)} + \frac{1}{K} \sum_{k=1}^{K} \Delta w_k$

**Output:** $\alpha^{(R)}$, and $w^{(R)}$

end
child node. The following proposition gives a bound on the convergence for a leaf node $B$ even when the input $w$ in Procedure $P$ is also determined by $\alpha_Q$ and $\alpha_{Q,B}$.

**Proposition 1** ([6, Proposition 1]). Let us consider a tree node $Q$ whose direct child node $B$ is a leaf node. Assume loss functions $\ell_i(\cdot)$ are $\frac{1}{\gamma}$-smooth. Then for the leaf node $B$, Assumption $7$ holds with

$$\Theta = \left(1 - \frac{\lambda m \gamma}{1 + \lambda m \gamma m_B}\right)^H. \quad (7)$$

where $m_B$ is the size of data stored at node $B$.

Additionally, Theorem 2, which is our main result, shows that if the geometric improvement condition holds true for direct child nodes of $Q$, then the geometric improvement condition also holds true for $Q$; thus it leads to a recursive calculation of the convergence rate for the tree network.

**Theorem 2.** Let us consider a tree node $Q$ which has $K$ direct child nodes. The $K$ direct child nodes satisfy the local geometric improvement requirement Assumption $4$ with parameters $\Theta_1, \Theta_2, \ldots, \Theta_K$. We assume that Algorithm $2$ (or Algorithm $3$) has an input $w$, and Algorithm $2$ (or Algorithm $3$) is run for $T$ iterations. We further make the assumption that the loss functions $\ell_i(\cdot)$ are $\frac{1}{\gamma}$-smooth.

Then, for any input $w$ to Algorithm $2$ (or Algorithm $3$), the following geometric convergence rate holds for $Q$:

$$E[D(\alpha^*_Q, \alpha_Q) - D(\alpha^{(T)}_Q, \alpha_Q)] \leq \left(1 - (1 - \Theta) \frac{1}{K \rho + \lambda m \gamma}\right)^T \left(D(\alpha^*_Q, \alpha_Q) - D(\alpha^{(0)}_Q, \alpha_Q)\right), \quad (8)$$

where $\Theta = \max_k \Theta_k$, and $\rho$ is any real number satisfying

$$\rho \geq \rho_{\min} \triangleq \max_{\alpha \in \mathbb{R}^{|Q|}} \lambda^2 m^2 \sum_{k=1}^K \|A_{[Q,k]} \alpha_{[Q,k]}\|^2 - \|A_Q \alpha_Q\|^2 \geq 0.$$

Because Theorem 2 works for any non-leaf tree node, by combining it with Proposition 1 we can recursively obtain the convergence rate of the generalized distributed dual coordinate ascent.
algorithm for the whole tree network. Note that \( \left( 1 - (1 - \Theta) \frac{1}{R \rho^s \lambda m \gamma^2} \right)^T \) becomes the “\( \Theta \)” for \( Q \), and \( (8) \) is seen as the direct child node geometric improvement of TreeDualMethod by the direct parent node of \( Q \).

Theorem 2 is different from Theorem 2 of [6] in two aspects. Firstly, Theorem 2 works for any tree node in a general tree network, beyond the star network discussed in [6]. Secondly, Theorem 2 is true, even when the input \( w \) of Algorithm 2 is not only determined by \( \alpha^Q \), but also determined by \( \alpha^Q \). To see this, we note that, at the root node, \( w = A_Q^r \alpha^Q + A_{Q}^r \alpha^Q \), and the root node will pass \( w \) to tree node \( Q \) by recalling TreeDualMethod(\( \cdot \)) for the root node’s child nodes. Our proof of Theorem 2 addresses this challenge that the input \( w \) is also affected by \( \alpha^Q \). For the readability, we place the proof of Theorem 2 in Appendix 9.1.

So far we have discussed how the network topology can affect the convergence rate of the distributed dual coordinate ascent, which is expressed in terms of the number of iterations. However, it is natural to consider communication delays, since the communication is a bottleneck of the convergence of distributed algorithms. In the next section, we consider how the communication delays, another major network constraint, impact the convergence of distributed dual coordinate ascent algorithms. By considering communication delays, we optimize the number of local iterations \( H \) for maximum convergence speed.

### 6 Impacts of communication delays on the convergence rate of distributed dual coordinate ascent algorithms

Earlier works [5][6] bounded the convergence of distributed dual coordinate ascent algorithms with respect to the number of inner and outer iterations. However, in distributed dual coordinate ascent algorithms, there may be significant communication delays between computers. Thus the convergence of distributed dual coordinate ascent algorithms not only depends on how many iterations of these algorithm have been run, but also depend on the communication delays in performing these iterations. Thus we aim to investigate the convergence of distributed dual coordinate ascent algorithms with respect to total time used, including computational time and communication delays.

In this paper, for simplicity, we consider the star network as shown in Fig. 1 and the corresponding Algorithm 1 even though our analysis can be generalized to a tree network. We assume that the round-trip communication delay between a local worker and the central station is \( t_{delay} \). Intuitively, if \( t_{delay} \) is close to 0, local workers might want to perform a small number of local iterations, and communicate with the central station at a higher frequency; on the other hand, if \( t_{delay} \) is large, namely, there is a large communication cost, then local workers may want to perform more local iterations before communicating with the central station in order to speed up convergence. We use \( t_{lp} \) to denote the computational time for one local iteration at a worker, and use \( t_{cp} \) to denote the computational time for one parameter update at the central station.

We assume each local worker performs \( H \) local iterations before communicating with the central station, and, in total, there are \( T \) outer iterations. The total experienced time is given by

\[
t_{total} = (t_{lp}H + t_{delay} + t_{cp}) \cdot T.
\]  

Hence, the number \( T \) of outer iterations is given by

\[
T = \frac{t_{total}}{(t_{lp}H + t_{delay} + t_{cp})}. \tag{10}
\]
Figure 3: Duality gap at center node as the operation time of algorithms goes. The distributed dual coordinate ascent in a tree network (red) and a star network (blue), i.e., CoCoA, are considered when the communication delay, $t_{delay}$, exists between the center node and its direct child nodes. $t_{delay} = 10^5 \times t_{lp}$, where $t_{lp}$ represents the computational time for one local iteration at a worker, and the average $t_{lp} \approx 10^{-5}$.

In (8), for $T$ outer iterations, the expected gap between the optimal objective value and the current objective value for Algorithm 1 is given by:

$$\left(1 - \left(1 - \left[1 - \delta\right]^H \right) \frac{C}{K}\right)^T,$$

where $\delta = \frac{s}{m}$ and $C = \frac{\lambda m \gamma}{(\rho + \lambda m \gamma)}$.

Our goal is to minimize the gap in objective value (11) under a given total time $t_{total}$, by optimizing over the number $H$ of local iterations. Hence, by plugging (10) into (11), we obtain the following optimization problem

$$\min_H \left(1 - \left(1 - \left[1 - \delta\right]^H \right) \frac{C}{K}\right)^T \frac{t_{total}}{t_{lp} H + t_{delay} + t_{cp}}.$$

(12)

Through (12), we can obtain the fastest convergence speed by adjusting $H$ according to $t_{cp}$, $t_{delay}$, and $t_{lp}$. In numerical experiments, we demonstrate that as $t_{delay}$ increases, the optimal number $H$ of local iterations also increases.

7 Numerical experiments

We demonstrate the convergence of the generalized distributed dual coordinate ascent in a tree network model. Since the authors in [6,7] compared the distributed dual coordinate ascent in a star network, so-called CoCoA, with other methods including mini-batch SDCA [15], local SGD and mini-batch-SGD [10], we only compare our generalized distributed dual coordinate ascent in a tree network with the CoCoA [6]. Additionally, since we are interested in the distributed dual coordinate ascent considering different network topologies, we do not consider the CoCoA+ [7], which is the updated version of the CoCoA.

In the numerical experiment, we assume that lots of communication delays exist between the center node and local workers for the CoCoA. For the generalized distributed dual coordinate
Figure 4: (a) The objective value of (12) when the number of iterations $H$ is varied from 1 to 2000, where $(C, K, \delta, t_{total}, t_{lp}, t_{cp}) = (0.5, 3, 1/300, 1, 4 \times 10^{-5}, 3 \times 10^{-5})$ and $t_{delay} = r \times t_{lp}$. The red line represents the optimal number of local iterations to achieve the fastest convergence rate. (b) Optimal number of iterations to achieve the fastest convergence rate, when the parameters are the same as (a) and $r$ is varied from 0 (no delay) to $10^{10}$.

Ascent in a tree network, the same communication delays exist between the center node and the sub-center nodes (assuming that communication delays between sub-center nodes and local workers are negligible). We test our algorithm for the ridge regression problem with the wine quality dataset [17]. We consider a tree network model having four local workers, two sub-center nodes (each having two local workers), and one center node. The simulated star network has four local workers and one center node. In both cases, we evenly split the data to four local workers. As shown in Fig. 3, the operation time of the distributed dual coordinate ascent can be further reduced by sharing local results via sub-center nodes when communication delays between the center node and local workers are large.

We then give numerical results showing how communication delays impact the convergence of the distributed dual coordinate ascent. Fig. 4 shows the optimal number $H$ of local iterations from optimizing (12). In the optimization, we set $(C, K, \delta, t_{total}, t_{lp}, t_{cp}) = (0.5, 3, 1/300, 1, 4 \times 10^{-5}, 3 \times 10^{-5})$, which were measured in the numerical experiments for the star network model (unit of time is second). We set $t_{delay} = r \times t_{lp}$, where $r$ is a parameter indicating how severe the communication delay is. Figure 4(a) shows the objective values of (12) when $H$ is varied from 1 to 2000. The red line represents the optimal convergence bound at the optimal number of local iterations. Fig. 4(b) shows the optimal number of local iterations to achieve the fastest convergence rate for different communication delays, where $r$ is varied from 0 to $10^{10}$.

We further experimented with solving synthetic linear regression problems for the star network shown in Fig. 1 in order to demonstrate how communication delays affect the optimal number of local iterations. We generated dataset $A \in \mathbb{R}^{100 \times 600}$ with i.i.d. zero-mean unit-variance Gaussian elements. We assigned evenly divided dataset to three local workers, i.e., 200 vectors for each local worker. We measured the average processing time $t_{lp}$ for one iteration by running LocalSDCA over the three disjoint dataset. $t_{delay}$ is set as $t_{delay} = r \times t_{lp}$, with $r = 10$ and $10e5$ for Fig. 5(a) and Fig. 5(b) respectively. For fixed $r$, we varied the number $H$ of local iterations from 10 to 10e4. When $t_{delay} = 10e5 \times t_{lp}$, we can see that the best number of local iterations $H$ is 1000 or 10000. When
Figure 5: Convergence rate in terms of operation time with different $t_{delay} = r \times t_{lp}$ in communication.

$t_{delay} = 10 \times t_{lp}, H = 100$ or 1000 provides the best convergence results. Those numerical experiments are consistent with our predictions in Fig. [4]

8 Discussion

In this paper, we considered the distributed dual coordinate ascent in the synchronous updating scheme. In fact, our analysis of the tree-structured network can be used for analyzing the convergence speed of asynchronous dual coordinate ascent algorithms [10] for a star network. In asynchronous dual coordinate ascent algorithms, a set of nodes communicate with the center and perform parameter updates more frequently. This set of nodes can be seen as forming a subtree (the network topology is a star, but in analysis we can change it to a tree accommodating asynchronous updating). Thus our results on the tree structured network can also be used for analysis of asynchronous dual coordinate ascent algorithm. We leave the detailed analysis of distributed dual coordinate ascent for updating for future work.

References

[1] M. Chen, S. Mao, and Y. Liu, “Big data: A survey,” Mobile Networks and Applications, vol. 19, no. 2, pp. 171–209, 2014.

[2] J. Ginsberg, M. H. Mohebbi, R. S. Patel, L. Brammer, M. S. Smolinski, and L. Brilliant, “Detecting influenza epidemics using search engine query data,” Nature, vol. 457, no. 7232, pp. 1012–1014, 2009.

[3] J. Chen, R. Monga, Bengio S., and R. Jozefowicz, “Revisiting distributed synchronous SGD,” in Proceedings of the International Conference on Learning Representations Workshop Track, 2016.

[4] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis, “Large-scale matrix factorization with distributed stochastic gradient descent,” in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2011, pp. 69–77.
Appendix

9.1 Proof of Theorem 2

Proof. Suppose the tree node \( Q \) has \( K \) direct child nodes, and we simply represent the child nodes from 1 to \( K \). The convergence rate of the algorithm at a tree node \( Q \) is obtained by considering the
updating scheme at the node $Q$ as follows. For the sake of simplicity, we omit $Q$ in the subscript of $\alpha$.

$$
\alpha^{(t+1)} = (\alpha^{(t+1)}_{[1:K]}, \alpha^Q_t) = (\alpha^{(t)}_{[1:K]} + \frac{1}{K} \sum_{k=1}^{K} \Delta \alpha_{[k]'}, \alpha^Q_t), \tag{13}
$$

where $\alpha_{[k]'}$ is the zero-padding version of $\alpha_{[k]}$ and $Q = [1 : K] = \cup_{k=1}^{K} [k]$ is the index set corresponding to workers connected to the node $Q$. The optimal value at the node $Q$ is stated as follows:

$$
D(\alpha_Q, \alpha^Q) = -\frac{\lambda}{2} \| A_Q \alpha_Q + A^Q \alpha^Q \|^2 - \frac{1}{m} \sum_{i \in Q} \ell_i^t (-\alpha_i) - \frac{1}{m} \sum_{i \not\in Q} \ell_i^t (\alpha_i)
$$

$$
= -\frac{\lambda}{2} \| A_{[1:K]} \alpha_{[1:K]} + \bar{w} \|^2 - \frac{1}{m} \sum_{i \in [1:K]} \ell_i^t (-\alpha_i) - \frac{1}{m} \sum_{i \not\in Q} \ell_i^t (\alpha_i), \tag{14}
$$

where $A_Q$ is the partial matrix of $A$ by choosing the columns of $A$ over the index set $Q$, and $A^Q \alpha^Q$ is denoted as $\bar{w}$. From (13), we have

$$
D(\alpha^{(t+1)}_{[1:K]}, \alpha^Q_t) = D(\alpha^{(t)}_{[1:K]} + \frac{1}{K} \sum_{k=1}^{K} \Delta \alpha_{[k]'}, \alpha^Q_t)
$$

$$
= D\left( \frac{1}{K} \sum_{k=1}^{K} (\alpha^{(t)}_{[1:K]} + \Delta \alpha_{[k]'}), \alpha^Q_t \right)
$$

$$
\geq \frac{1}{K} \sum_{k=1}^{K} D(\alpha^{(t)}_{[1:K]} + \Delta \alpha_{[k]'}, \alpha^Q_t),
$$

where the inequality is obtained from the Jensen’s inequality. Then, we have

$$
D(\alpha^{(t+1)}_{[1:K]}, \alpha^Q_t) - D(\alpha^{(t)}_{[1:K]}, \alpha^Q_t) \geq \frac{1}{K} \sum_{k=1}^{K} \left[ D(\alpha^{(t)}_{[1:K]} + \Delta \alpha_{[k]'}, \alpha^Q_t) - D(\alpha^{(t)}_{[1:K]}, \alpha^Q_t) \right]
$$

$$
= \frac{1}{K} \sum_{k=1}^{K} \left[ D(\alpha^{(t)}_{[1:K]} + \Delta \alpha_{[k]'}, \alpha^Q_t) - D((\alpha^{(t)}_{[Q,1]}, \ldots, \alpha^{(t)}_{[Q,k]}, \ldots, \alpha^{(t)}_{[Q,K]}), \alpha^Q_t) \right]
$$

$$
+ D((\alpha^{(t)}_{[Q,1]}, \ldots, \alpha^{(t)}_{[Q,k]}, \ldots, \alpha^{(t)}_{[Q,K]}), \alpha^Q_t) - D(\alpha^{(t)}_{[1:K]}, \alpha^Q_t)
$$

$$
= \frac{1}{K} \sum_{k=1}^{K} \left[ \epsilon_{Q,k}(\alpha^{(t)}_{[1:K]}, \alpha^Q_t) - \epsilon_{Q,k}(\alpha^{(t)}_{[1:K]} + \Delta \alpha_{[k]'}, \alpha^Q_t) \right]
$$

Then, the expectation is stated as follows:

$$
E[D(\alpha^{(t+1)}_{[1:K]}, \alpha^Q_t) - D(\alpha^{(t)}_{[1:K]}, \alpha^Q_t)] \geq \frac{1}{K} \sum_{k=1}^{K} \left[ E[\epsilon_{Q,k}(\alpha^{(t)}_{[1:K]}, \alpha^Q_t)] - E[\epsilon_{Q,k}(\alpha^{(t)}_{[1:K]} + \Delta \alpha_{[k]'}, \alpha^Q_t)] \right]
$$

$$
\geq \frac{1}{K} (1 - \Theta) \sum_{k=1}^{K} \epsilon_{Q,k}(\alpha^{(t)}_{[1:K]}, \alpha^Q_t),
$$

where the last inequality is obtained from Assumption [1].
\[
\sum_{k=1}^{K} \epsilon_{Q,k}(\alpha_{1:K}^{(t)}, \alpha_{Q})
= \sum_{k=1}^{K} \maximize_{\alpha_{(Q,k)}} \left[ D((\alpha_{1:1}^{(t)}, \ldots, \hat{\alpha}_{(Q,k)}, \ldots, \alpha_{(Q,K)}^{(t)}, \alpha_{Q})) - D((\alpha_{1:1}^{(t)}, \ldots, \alpha_{(Q,k)}^{(t)}, \ldots, \alpha_{(Q,K)}^{(t)}, \alpha_{Q})) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \sum_{k=1}^{K} \left[ D((\alpha_{1:1}^{(t)}, \ldots, \hat{\alpha}_{(Q,k)}, \ldots, \alpha_{(Q,K)}^{(t)}, \alpha_{Q})) - D((\alpha_{1:1}^{(t)}, \ldots, \alpha_{(Q,k)}^{(t)}, \ldots, \alpha_{(Q,K)}^{(t)}, \alpha_{Q})) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \sum_{k=1}^{K} \left[ - \frac{\lambda}{2} \|A_{1:K}(\alpha_{1:1}^{(t)}, \ldots, \hat{\alpha}_{(Q,k)}, \ldots, \alpha_{(Q,K)}^{(t)}) + \bar{w} \|^2 + \frac{\lambda}{2} \|A_{1:K}^{(t)} + \bar{w} \|^2 \right. \\
\left. - \frac{1}{m} \sum_{i \in [1:K]} \ell_i^*(-\hat{\alpha}_i) + \frac{1}{m} \sum_{i \in [1:K]} \ell_i^*(-\alpha_i^{(t)}) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \left[ - \frac{\lambda}{2} \sum_{k=1}^{K} \left[ \|A_{1:K}(\alpha_{1:1}^{(t)}, \ldots, \hat{\alpha}_{(Q,k)}, \ldots, \alpha_{(Q,K)}^{(t)}) - A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)}) + \bar{w} \|^2 - \|A_{1:K}^{(t)} + \bar{w} \|^2 \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \left[ - \frac{\lambda}{2} \sum_{k=1}^{K} \left[ \|A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)})\|^2 - 2(A_{1:K}^{(t)} + \bar{w})^T A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)}) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \left[ - \frac{\lambda}{2} \sum_{k=1}^{K} \left[ \|A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)})\|^2 + \lambda \|A_{1:K}^{(t)} + \bar{w}\|^2 (A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)}) + \bar{w} - \bar{w}) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \left[ - \frac{\lambda}{2} \sum_{k=1}^{K} \left[ \|A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)})\|^2 + \lambda \|A_{1:K}^{(t)} + \bar{w}\|^2 (A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)}) + \bar{w} - \bar{w}) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \left[ - \frac{\lambda}{2} \sum_{k=1}^{K} \left[ \|A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)})\|^2 + \lambda \|A_{1:K}^{(t)} + \bar{w}\|^2 - \lambda (A_{1:K}^{(t)} + \bar{w})^T (A_{1:K}^{(t)} + \bar{w}) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \left[ - \frac{\lambda}{2} \sum_{k=1}^{K} \left[ \|A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)})\|^2 + \lambda \|A_{1:K}^{(t)} + \bar{w}\|^2 - 2\lambda (A_{1:K}^{(t)} + \bar{w})^T (A_{1:K}^{(t)} + \bar{w}) \right] \\
= \maximize_{\alpha \in \mathbb{R}^{1:K}} \left[ - \frac{\lambda}{2} \sum_{k=1}^{K} \left[ \|A_{[k]}(\alpha_{(k)}^{(t)} - \hat{\alpha}_{(k)})\|^2 - \|A_{1:K}^{(t)}(\hat{\alpha}_{[1:K]} - \alpha_{[1:K]}^{(t)})\|^2 \right] \right]
\right]
= (A) \tag{15}
\]
We can lower-bound \((15)\) by upper-bounding \((A)\). We define the upper bound as
\[
\rho \geq \rho_{\min} \triangleq \max_{\alpha} \lambda^2 \frac{\sum_{k=1}^{K} \|A[k]\alpha[k]\|^2 - \|A[1:K]\alpha\|^2}{\|\alpha\|^2}
\] (16)

Then, \((15)\), which is \(\sum_{k=1}^{K} \epsilon_{Q,k}(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]})\), is lower-bounded as follows:
\[
\begin{align*}
\sum_{k=1}^{K} \epsilon_{Q,k}(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) \\
\geq \max_{\alpha \in \mathbb{R}^{[1:K]\|}} D(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) - D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) - \frac{\rho_{\min}}{2\lambda|[1:K]|^2} \|\alpha_{[1:K]}^{(t)} - \alpha_{[1:K]}\|^2 \\
\geq \max_{\eta \in [0, 1]} \eta D(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) + (1 - \eta)D(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) - D(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) + \frac{\rho_{\min} \eta^2}{2\lambda|[1:K]|^2} \|\alpha_{[1:K]}^{(t)} - \alpha_{[1:K]}\|^2 \\
\geq \max_{\eta \in [0, 1]} \eta D(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) - \eta D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) + \frac{\gamma \eta (1 - \eta)}{2\lambda|[1:K]|^2} \|\alpha_{[1:K]}^{(t)} - \alpha_{[1:K]}\|^2 - \frac{\rho_{\min} \eta^2}{2\lambda|[1:K]|^2} \|\alpha_{[1:K]}^{(t)} - \alpha_{[1:K]}\|^2
\end{align*}
\] (17)

(17) can be lower-bounded by choosing \(\eta = \frac{\lambda|[1:K]|^\gamma}{\lambda|[1:K]|^\gamma + \rho}\) as follows:
\[
\begin{align*}
\sum_{k=1}^{K} \epsilon_{Q,k}(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) \\
\geq \frac{\lambda|[1:K]|^\gamma}{\lambda|[1:K]|^\gamma + \rho} \left( D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) - D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) \right) + \frac{1}{2\lambda|[1:K]|^2} \left( \frac{\lambda|[1:K]|^\gamma}{\lambda|[1:K]|^\gamma + \rho} \right) \|\alpha_{[1:K]}^{(t)} - \alpha_{[1:K]}\|^2 \\
\geq \frac{\lambda|[1:K]|^\gamma}{\lambda|[1:K]|^\gamma + \rho} \left( D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) - D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) \right)
\end{align*}
\]

Therefore, we have
\[
E[D(\alpha^{(t+1)}_{[1:K]}, \alpha_{[Q]}) - D(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]}) | \bar{\alpha}_{[1:K]}^{(t)}, \alpha_{[Q]}^{(t)}] \geq \frac{1}{K} (1 - \Theta) \sum_{k=1}^{K} \epsilon_{Q,k}(\alpha^{(t)}_{[1:K]}, \alpha_{[Q]})
\]
\[
\begin{align*}
\geq \frac{1}{K} (1 - \Theta) \frac{\lambda|[1:K]|^\gamma}{\lambda|[1:K]|^\gamma + \rho} \left( D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) - D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) \right)
\end{align*}
\]

Therefore,
\[
E[D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) - D(\alpha_{[1:K]}^{(t+1)}, \alpha_{[Q]}) | \alpha_{[1:K]}^{(t)}, \alpha_{[Q]}] \\
\leq \left( 1 - \frac{1}{K} (1 - \Theta) \frac{\lambda|[1:K]|^\gamma}{\lambda|[1:K]|^\gamma + \rho} \right) \left( D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) - D(\alpha_{[1:K]}^{(t)}, \alpha_{[Q]}) \right)
\]

\(\square\)