Hybrid-DCA: A Double Asynchronous Approach for Stochastic Dual Coordinate Ascent

Soumitra Pal   MITRA@UCONN.EDU
Tingyang Xu   TIX11001@ENGR.UCONN.EDU
Computer Science and Engineering, University of Connecticut, Storrs, CT 06269 USA

Tianbao Yang   TIANBAO-YANG@UIOWA.EDU
Computer Science, University of Iowa, Iowa City, IW 52242 USA

Sanguthevar Rajasekaran   RAJASEK@ENGR.UCONN.EDU
Jinbo Bi   JINBO@ENGR.UCONN.EDU
Computer Science and Engineering, University of Connecticut, Storrs, CT 06269 USA

Abstract

In prior works, stochastic dual coordinate ascent (SDCA) has been parallelized in a multi-core environment where the cores communicate through shared memory, or in a multi-processor distributed memory environment where the processors communicate through message passing. In this paper, we propose a hybrid SDCA framework for multi-core clusters, the most common high performance computing environment that consists of multiple nodes each having multiple cores and its own shared memory. We distribute data across nodes where each node solves a local problem in an asynchronous parallel fashion on its cores, and then the local updates are aggregated via an asynchronous across-node update scheme. The proposed double asynchronous method converges to a global solution for L-Lipschitz continuous loss functions, and at a linear convergence rate if a smooth convex loss function is used. Extensive empirical comparison has shown that our algorithm scales better than the best known shared-memory methods and runs faster than previous distributed-memory methods. Big datasets, such as one of 280 GB from the LIBSVM repository, cannot be accommodated on a single node and hence cannot be solved by a parallel algorithm. For such a dataset, our hybrid algorithm takes 30 seconds to achieve a duality gap of 10^{-6} on 16 nodes each using 8 cores, which is significantly faster than the best known distributed algorithms, such as CoCoA+, that take more than 300 seconds on 16 nodes.

Keywords: dual coordinate descent, distributed computing, optimization

1. Introduction

The immense growth of data has made it important to efficiently solve large scale machine learning problems. It is necessary to take advantage of modern high performance computing (HPC) environments such as multi-core settings where the cores communicate through shared memory, or multi-processor distributed memory settings where the processors communicate by passing messages. In particular, a large class of supervised learning formulations, including support vector machines (SVMs), logistic regression, ridge regression and many others, solve the following generic regularized risk minimization (RRM) problem: given a set of instance-

label pairs of data points $(\mathbf{x}_i, y_i), i = 1, \ldots, n$,

$$\min_{\mathbf{w} \in \mathbb{R}^d} P(\mathbf{w}) := \frac{1}{n} \sum_{i=1}^{n} \phi(\mathbf{x}_i^\top \mathbf{w}; y_i) + \frac{\lambda}{2} g(\mathbf{w}), \tag{1}$$

where $y_i \in \mathbb{R}$ is the label for the data point $\mathbf{x}_i \in \mathbb{R}^d$, $\mathbf{w} \in \mathbb{R}^d$ is the linear predictor to be optimized, ϕ is a loss function that is convex with respect to its first argument, λ is a regularization parameter that balances between the loss and a regularizer $g(\mathbf{w})$, especially the ℓ_2-norm penalty $\|\mathbf{w}\|_2^2$.

Many efficient sequential algorithms have been developed in the past decades to solve (1), *e.g.,* stochastic gradient descent (SGD) Zhang (2004), or alternating direction method of multipliers (ADMM) Boyd et al. (2011). Especially, (stochastic) dual coordinate ascent (DCA) algorithm Shalev-Shwartz and Zhang (2013) has been one of the most widely used algorithms for solving (1). It efficiently optimizes the following dual formulation (2)

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^n} D(\boldsymbol{\alpha}) := -\frac{1}{n} \sum_{i=1}^{n} \phi^*(-\alpha_i) - \frac{\lambda}{2} g^* \left(\frac{1}{\lambda n} \mathbf{X} \boldsymbol{\alpha} \right), \tag{2}$$

$$\text{using} \qquad \mathbf{w}(\boldsymbol{\alpha}) = \nabla g^* \left(\frac{1}{\lambda n} \mathbf{X} \boldsymbol{\alpha} \right), \tag{3}$$

where ϕ^*, g^* are the convex conjugates of ϕ, g, respectively, defined as, *e.g.,* $\phi^*(u) = \max_z(zu - \phi(z))$ and it is known that if $\boldsymbol{\alpha}^*$ is an optimal dual solution then $\mathbf{w}^* = \mathbf{w}(\boldsymbol{\alpha}^*)$ is an optimal primal solution and $P(\mathbf{w}^*) = D(\boldsymbol{\alpha}^*)$. The dual objective has a separate dual variable associated with each training data point. The stochastic DCA updates dual variables, one at a time, while maintaining the primal variables by calculating (3) from the dual variables.

Recently, many efforts have been undertaken to solve (1) in a distributed or parallel framework. It has been shown that distributed DCA algorithms have comparable and sometimes even better convergence than SGD-based or ADMM-based distributed algorithms Yang (2013). The distributed DCA algorithms can be grouped into two sets. The first set contains synchronous algorithms in which a random dual variable is updated by each processor and the primal variables are synchronized across the processors in every iteration Jaggi et al. (2014); Ma et al. (2015b); Yang (2013). This approach incurs a large communication overhead. The second set of algorithms avoids communication overhead by exploiting the shared memory in a multi-core setting Hsieh et al. (2015) where the primal variables are stored in a primary memory shared across all the processors. Further speedups have been obtained by using (asynchronous) atomic memory operations instead of costly locks for shared memory updates Hsieh et al. (2015); Peng et al. (2015). Nevertheless, this approach is difficult to scale up for big datasets that cannot be fully accommodated in the shared memory. This leads to a challenging question: how do we scale up the asynchronous shared memory approach for big data while maintaining the speed up?

We address this challenge by proposing and implementing a hybrid strategy. The modern HPC platforms can be viewed as a collection of K nodes interconnected through a network as shown in Fig. 1(a). Each node contains a memory shared among R processing cores. Our strategy exploits this architecture by equally distributing the data across the local
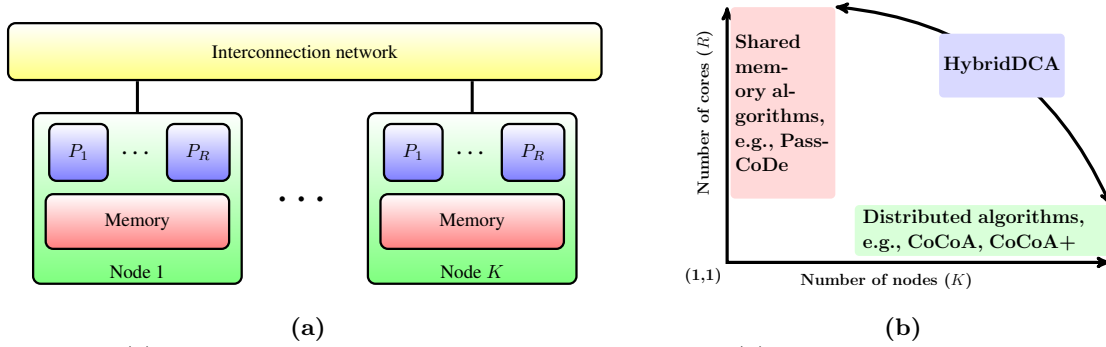
Figure 1: (a) A simplified view of the modern HPC system and (b) Algorithms on this architecture.

shared memory of the K nodes. Each of the R cores within a node runs a computing thread that asynchronously updates a random dual variable from those associated with the data allocated to the node. Each node also runs a communicating thread. One of the communicating threads is designated as *master* and the rest are *workers*. After every round of H local iterations in each computing thread, each worker thread sends the local update to the master. After accumulating the local updates from S of the K workers, the master broadcasts the global update to the contributing workers. However, to avoid a slower worker falling back too far, the master ensures that in every Γ consecutive global updates there is at least one local update from each worker. Fig. 1(b) shows how our scheme is a generalization of the existing approaches: for $K = 1$, our setup coincides with the shared memory multi-core setting Hsieh et al. (2015) and for $R = 1, S = K$ our setup coincides with the synchronous algorithms in distributed memory setting Jaggi et al. (2014); Ma et al. (2015b); Yang (2013). With a proper adjustment of the parameters H, S, Γ our strategy could balance the computation time of the first setting with the communication time of the second one, while ensuring scalability in big data applications.

Thus, our contributions are 1) we propose and analyze a hybrid asynchronous shared memory and asynchronous distributed memory implementation (*Hybrid-DCA*) of the mostly used DCA algorithm to solve (1); 2) we prove a strong guarantee of convergence for L-Lipschitz continuous loss functions, and further linear convergence when a smooth convex loss function is used; and 3) the experimental results using our light-weight OpenMP+MPI implementation show that our algorithms are much faster than existing distributed memory algorithms Jaggi et al. (2014); Ma et al. (2015b), and easily scale up with the volume of data in comparison with the shared memory based algorithms Hsieh et al. (2015) as the shared memory size is limited.

2. Related Work

Sequential Algorithms. SGD is the oldest and simplest method of solving problem (1). Though SGD is easy to implement and converges to modest accuracy quickly, it requires a long tail of iterations to reach 'good' solutions and also requires adjusting a step-size parameter. On the other hand, SDCA methods are free of learning-rate parameters and have faster convergence rate around the end Moulines and Bach (2011); Needell et al. (2014). A modified SGD has also been proposed with faster convergence by switching to SDCA after

quickly reaching a modest solution Shalev-Shwartz and Zhang (2013). Recently, 'variance reduced' modifications to the original SGD have also caught attention. These modifications estimate gradients with small variance as they approach to an optimal solution. Mini-batch algorithms are also proposed to update several dual variables (data points) in a batch rather than a single data point per iterationTakac et al. (2013). Mini-batch versions of both SGD and SDCA have slower convergence when the batch size increasesRichtárik and Takáč (2013); Shalev-Shwartz and Zhang (2016). All these sequential algorithms become ineffective when the datasets get bigger.

Distributed Algorithms. In the early single communication scheme McWilliams et al. (2014); Heinze et al. (2015); Mcdonald et al. (2009), a dataset is 'decomposed' into smaller parts that can be solved independently. The final solution is reached by 'accumulating' the partial solutions using a single round of communications. This method has limited utility because most datasets cannot be decomposed in such a way. Using the primal-dual relationship (3), fully distributed algorithms of DCA are later developed where each processor updates a separate α_i which is then used to update $\mathbf{w}(\boldsymbol{\alpha})$, and synchronizes $\mathbf{w}$ across all processors (*e.g.,* CoCoA Jaggi et al. (2014)). To trade off communications vs computations, a processor can solve its subproblem with H dual updates before synchronizing the primal variable (*e.g.,* CoCoA+ Ma et al. (2015b),DisDCA Yang (2013)). In Yang (2013); Ma et al. (2015b), a more general framework is proposed in which the subproblem can be solved using not only SDCA but also any other sequential solver that can guarantee a Θ-approximation of the local solution for some $\Theta \in (0, 1]$. Nevertheless, the synchronized update to the primal variables has the inherent drawback that the overall algorithm runs at a speed of the slowest processor even when there are fast processors Agarwal et al. (2014).

Parallel Algorithms. Multi-core shared memory systems have also been exploited, where the primal variables are maintained in a shared memory, removing the communication cost. However, updates to shared memory requires synchronization primitives, such as locks, which again slows down computation. Recent methods Hsieh et al. (2015); Liu et al. (2015) avoid locks by exploiting (asynchronous) atomic memory updates in modern memory systems. There is even a wild version in Hsieh et al. (2015) that takes arbitrarily one of the simultaneous updates. Though the shared memory algorithms are faster than the distributed versions, they have an inherent drawback of being not scalable, as there can be only a few cores in a processor board.

Other Distributed Methods for RRM. Besides distributed DCA methods, there are several recent distributed versions of other algorithms with faster convergence, including distributed Newton-type methods (DISCO Zhang and Xiao (2015), DANE Shamir et al. (2014)) and distributed stochastic variance reduced gradient method (DSVRG Lee et al. (2015)). It has been shown that they can achieve the same accurate solution using fewer rounds of communication, however, with additional computational overhead. In particular, DISCO and DANE need to solve a linear system in each round, which could be very expensive for higher dimensions. DSVRG requires each machine to load and store a second subset of the data sampled from the original training data, which also increase its running time.

The ADMM Boyd et al. (2011) and quasi-Newton methods such as L-BFGS also have distributed solutions. These methods have low communication cost, however, their inherent drawback of computing the full batch gradient does not give computation vs communications

trade-off. In the context of consensus optimization, Zhang and Kwok (2014) gives an asynchronous distributed ADMM algorithm but that does not directly apply to solving (1).

To the best of our knowledge, this paper is the first to propose, implement and analyze a hybrid approach exploiting modern HPC architecture. Our approach is the amalgamation of three different ideas – 1) CoCoA+/DisDCA distributed framework, 2) asynchronous multi-core shared-memory solver Hsieh et al. (2015) and 3) asynchronous distributed approach Zhang and Kwok (2014) – taking the best of each of them. In a sense ours is the first algorithm which asynchronously uses updates which themselves have been computed using asynchronous methods.

3. Algorithm

At the core of our algorithm, the data is distributed across K nodes and each node, called a *worker*, repeatedly solves a perturbed dual formulation on its data partition and sends the local update to one of the nodes designated as the *master* which merges the local updates and sends back the accumulated global update to the workers to solve the subproblem once again, unless a global convergence is reached. Let $\mathcal{I}_k \subseteq \{1, 2, \ldots, n\}, k = 1, \ldots, K$ denote the indices of the data and the dual variables residing on node k and $n_k = |\mathcal{I}_k|$. For any $\mathbf{z} \in \mathbb{R}^n$ let $\mathbf{z}_{[k]}$ denote the vector in $\mathbb{R}^n$ defined in such a way that the ith component $(\mathbf{z}_{[k]})_i = z_i$ if $i \in \mathcal{I}_k$, 0 otherwise. Let $\mathbf{X}_{[k]} \in \mathbb{R}^{d \times n}$ denote the matrix consisting of the columns of the $\mathbf{X} \in \mathbb{R}^{d \times n}$ indexed by $\mathcal{I}_k$ and replaced with zeros in all other columns.

Ideally, the dual problem solved by node k is (2) with $\mathbf{X}, \boldsymbol{\alpha}$ replaced by $\mathbf{X}_{[k]}, \boldsymbol{\alpha}_{[k]}$, respectively, and hence is independent of other nodes. However, following the efficient practical implementation idea in Yang (2013); Ma et al. (2015b), we let the workers communicate among them a vector $\mathbf{v} \in \mathbb{R}^d$, an estimate of $\mathbf{w}(\boldsymbol{\alpha}) = \frac{1}{\lambda n} \mathbf{X}\boldsymbol{\alpha}$ that summarizes the last known global solution $\boldsymbol{\alpha}$. Also following Yang (2013); Ma et al. (2015b) for faster convergence, each worker in our algorithm solves the following perturbed local dual problem, which we henceforth call the *subproblem*:

$$\max_{\boldsymbol{\delta}_{[k]} \in \mathbb{R}^n} Q_k^\sigma(\boldsymbol{\delta}_{[k]}; \mathbf{v}, \boldsymbol{\alpha}_{[k]}) := -\frac{1}{n_k} \sum_{i \in \mathcal{I}_k} \phi^*(-\alpha_i - \delta_i) - \frac{\lambda}{S} g^*(\mathbf{v})$$

$$- \langle \frac{1}{n} X_{[k]}^\top \nabla g^*(\mathbf{v}), \boldsymbol{\delta}_{[k]} \rangle - \frac{\lambda \sigma}{2} \left\| \frac{1}{\lambda n} \mathbf{X}_{[k]} \boldsymbol{\delta}_{[k]} \right\|^2 \tag{4}$$

where $\boldsymbol{\delta}_{[k]}$ denotes the local (incremental) update to the dual variable $\boldsymbol{\alpha}_{[k]}$ and the *scaling parameter* σ measures the difficulty of solving the given data partition (see Yang (2013); Ma et al. (2015b)) and must be chosen such that

$$\sigma \geq \sigma_{min} := \nu \max_{\boldsymbol{\alpha} \in \mathbb{R}^n} \frac{\|\mathbf{X}\boldsymbol{\alpha}\|^2}{\sum_{k=1}^K \|\mathbf{X}\boldsymbol{\alpha}_{[k]}\|^2} \tag{5}$$

where the *aggregation parameter* $\nu \in [\frac{1}{S}, 1]$ is the weight given by the master to each of local updates from the contributing workers while computing the global update. The second term in the objective of our subproblem has denominator S in stead of K. Unlike the synchronous all reduce approach in Ma et al. (2015b), our asynchronous method merges the local updates

5

Algorithm 1: Hybrid-DCA: Worker node k

Input: Initial $\boldsymbol{\alpha}_{[k]} \in \mathbb{R}^n$, data partition I_k,
            scaling parameter σ, aggregation parameter ν

1   $\mathbf{v} \leftarrow \frac{1}{\lambda n} X \boldsymbol{\alpha}_{[k]}$;

2   **for** $t \leftarrow 0, 1, \ldots$ **do**

3      $\boldsymbol{\delta}_{[k]} \leftarrow \mathbf{0}, \quad \mathbf{v}_{old} \leftarrow \mathbf{v}$;

4   **for** *cores* $r \leftarrow 1, \ldots, R$ *in parallel* **do**

5   **for** $h \leftarrow 0, 1, \ldots, H-1$ **do**

6   Randomly pick i from $I_{k,r}$;

7              $\varepsilon \leftarrow \operatorname{argmax}_{\varepsilon} Q_k^{\sigma}(\varepsilon \mathbf{e}_i; \mathbf{v}, \boldsymbol{\alpha}_{[k]} + \boldsymbol{\delta}_{[k]})$;

8              $\boldsymbol{\delta}_{[k]} \leftarrow \boldsymbol{\delta}_{[k]} + \varepsilon \mathbf{e}_i$;

9              $\mathbf{v} \xleftarrow{atomic} \mathbf{v} + \nabla g^* \left(\frac{1}{\lambda n} X \varepsilon \mathbf{e}_i \right)$;

10   send $\Delta \mathbf{v} \leftarrow \mathbf{v} - \mathbf{v}_{old}$ to the master;

11   receive $\mathbf{v}$ from the master;

12     $\boldsymbol{\alpha}_{[k]} \leftarrow \boldsymbol{\alpha}_{[k]} + \nu \boldsymbol{\delta}_{[k]}$;

---

from only S out of K nodes in each global update. By Lemma 3.2 in Ma et al. (2015b), $\sigma := \nu S$ is a safe choice to hold condition (5).

3.1 Asynchronous updates by cores in a worker node

In each communication round, each worker k solves its subproblem using a parallel asynchronous DCA method Hsieh et al. (2015) on the R cores. Let the data partition $\mathcal{I}_k$ stored in the shared memory be logically divided into R subparts where subpart $\mathcal{I}_{k,r} \subseteq \mathcal{I}_k, r = 1, \ldots, R$, is exclusively used by core r. In each of the H iterations, core r chooses a random coordinate $i \in \mathcal{I}_{k,r}$ and updates $\boldsymbol{\delta}_{[k]}$ in the ith unit direction by a step size ε computed using a single variable optimization problem:

$$\varepsilon = \operatorname*{argmax}_{\varepsilon \in \mathbb{R}} Q_k^{\sigma}(\varepsilon \mathbf{e}_i; \mathbf{v}, \boldsymbol{\alpha}_{[k]} + \boldsymbol{\delta}_{[k]}) \tag{6}$$

which has a closed form solution for SVM problems Fan et al. (2008), and a solution using an iterative solver for logistic regression problems Yu et al. (2011). The local updates to $\mathbf{v}$ are also maintained appropriately. While the coordinates used by any two cores and hence the corresponding updates to $\boldsymbol{\delta}_{[k]}$ are independent of each other, there might be conflicts in the updates to $\mathbf{v}$ if the corresponding columns in X have nonzero values in some common row. We use lock-free *atomic* memory updates to handle such conflicts. When all cores complete H iterations, worker k sends the accumulated update $\Delta \mathbf{v}$ from the current round to the master; waits until it receives the globally updated $\mathbf{v}$ from the master; and repeats for another round unless the master indicates termination.

3.2 Merging updates from workers by master

If the master had to wait for the updates from all the workers, it could compute the global updates only after the slowest worker finished. To avoid this problem, we use *bounded barrier*: in each round, the master waits for updates from only a subset $\mathcal{P}_S$ of $S \leq K$

Algorithm 2: Hybrid-DCA: Master node

Input: Initial $\boldsymbol{\alpha} \in \mathbb{R}^n$, aggregation parameter ν,
         barrier bound S, delay bound Γ

1   $\mathbf{v}^{(0)} \leftarrow \frac{1}{\lambda n}\mathbf{X}\boldsymbol{\alpha}; \qquad \mathcal{P} = \emptyset;$

2   **for** $t \leftarrow 0, 1, \ldots$ **do**

3   **while** $|\mathcal{P}| < S$ **or** $\max_k \Gamma_k > \Gamma$ **do**

4   receive update $\Delta\mathbf{v}_k$ from some worker k;

5       $\mathcal{P} \leftarrow \mathcal{P} \cup \{k\}; \qquad \Gamma_k \leftarrow 1;$

6     $\mathcal{P}_S^{(t)} \leftarrow S$ workers in $\mathcal{P}$ with oldest updates;

7     $\mathbf{v}^{(t+1)} \leftarrow \mathbf{v}^{(t)} + \nu \sum_{k \in \mathcal{P}_S^{(t)}} \Delta\mathbf{v}_k; \quad \mathcal{P} \leftarrow \mathcal{P} \setminus \mathcal{P}_S^{(t)};$

8   **foreach** $k \notin \mathcal{P}_S^{(t)}$ **do** $\Gamma_k \leftarrow \Gamma_k + 1$ broadcast $\mathbf{v}^{(t+1)}$ to all workers in $\mathcal{P}_S^{(t)}$;

---

workers, and sends them back the global update $\mathbf{v} = \mathbf{v} + \nu \sum_{k \in \mathcal{P}_S} \Delta\mathbf{v}_k$. However, due to this relaxation, there might be some slow workers with out-of-date $\mathbf{v}$. When updates from such workers are merged by the master, it may degrade the quality of the global solution and hence may cause slow convergence or even divergence. We ensure sufficient freshness of the updates using *bounded delay*: the master makes sure that no worker has a stale update older than Γ rounds. This asynchronous approach has two benefits: 1) the overall progress is no more bottlenecked by the slowest processor, and 2) the total number of communications is reduced. On the flip side, convergence may get slowed down for very small S or very large Γ.

Example: Figure 2 shows a possible sequence of important events in a run of our algorithm on a dataset having $n = 12$ data points in $d = 3$ dimensions using $K = 3$ nodes each having $R = 2$ cores such that each core works with only $|I_{k,r}| = 2$ data points. The activities in solving the subproblem using $H = 1$ local iterations in a round is shown in a rectangular box. For the first subproblem, core 1 and core 2 in worker 1 randomly selects dual coordinates such that the corresponding data points have nonzero entries in the dimensions $\{1,3\}, \{1,2,3\}$, respectively. Each core first reads the entries of $\mathbf{v}$ corresponding to these nonzero data dimensions, and then computes the updates $[0.1, 0, 0.7], [0.15, 0.5, 0.4]$, respectively, and finally applies these updates to $\mathbf{v}$. The atomic memory updates ensure that all the conflicting writes to $\mathbf{v}$, such as v_1 in the first write-cycle, happen completely. At the end of H local iterations by each core, worker 1 sends $\Delta\mathbf{v} = [0.25, 0.5, 1.1]$ to the master, the responsibility of which is shared by one of the 3 nodes, but shown separately in the figure. By this time, the faster workers 2 and 3 already complete 3 rounds. As $S = 2$, the master takes first 2 updates from $\mathcal{P}_S^{(1)} = \mathcal{P}_S^{(2)} = \{2, 3\}$ and computes the global updates using $\nu = 1$. However, as $\Gamma = 2$, the master holds back the third updates from workers 2,3 until the first update from worker 1 reaches master. The subsequent events in the run are omitted in the figure.

4. Convergence Analysis

In this section we prove the convergence of the global solution computed by our hybrid algorithm. For ease we prove for $g(w) = \|w\|^2$; the proof can be similarly extended for other
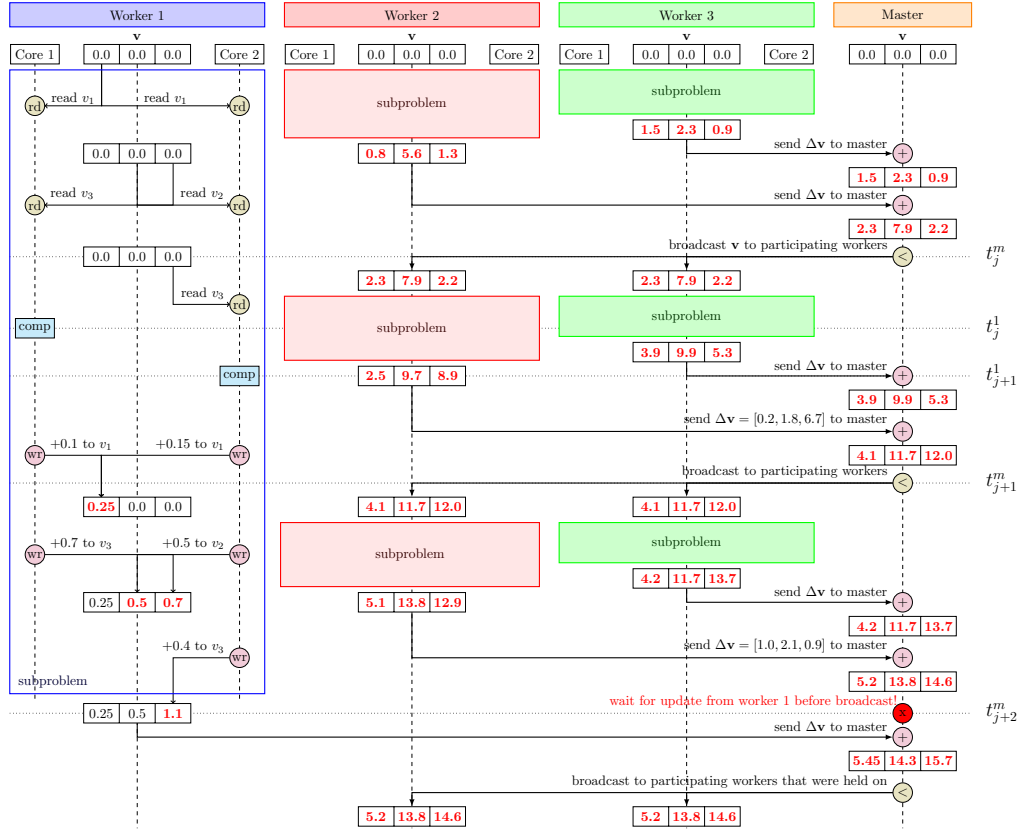
Figure 2: Sequence of important events in an example run of Hybrid-DCA where $n = 12, d = 3, K = 3, R = 2, S = 2, \Gamma = 2, \nu = 1$.

regularizers $g(w)$. The analysis is divided into two parts. First we show that the solution of the subproblem computed by each node locally is indeed not far from the optimum. Using this result on the subproblem, we next show the convergence of the global solution. Though our proofs for the two parts are based on the works Hsieh et al. (2015) and Ma et al. (2015a), respectively, we need to make significant adjustments in the proofs due to our modified framework handling two cascaded levels of asynchronous updates. Here we outline the modifications, the complete proofs are given in the appendix.

4.1 Near optimality of the solution to the local subproblem

Definition 1 *For given* $\mathbf{v}, \boldsymbol{\alpha}_{[k]}$*, a solution* $\boldsymbol{\delta}_{[k]}$ *to the subproblem* (4) *is said to be* Θ-*approximate,* $\Theta \in [0, 1)$*, if*

$$\mathbb{E}\left[Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \mathbf{v}, \boldsymbol{\alpha}_{[k]}) - Q_k^\sigma(\boldsymbol{\delta}_{[k]}; \mathbf{v}, \boldsymbol{\alpha}_{[k]})\right] \leq \Theta\left[Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \mathbf{v}, \boldsymbol{\alpha}_{[k]}) - Q_k^\sigma(\mathbf{0}; \mathbf{v}, \boldsymbol{\alpha}_{[k]})\right] \tag{7}$$

where $\boldsymbol{\delta}_{[k]}^*$ *is the optimum solution to* (4)*.*

The main challenge in using the results of Hsieh et al. (2015) to prove (7) for the solution returned by the parallel asynchronous stochastic DCA solver used by each worker in Algorithm 1 is to tackle the following changes: 1) the solver here solves only a part of the

dual problem and 2) the subproblem is now perturbed (see Section 3). While the first change is simply handled by considering the updates by the cores in worker k only, the second change needs modifications in each step of the proof in Hsieh et al. (2015), including the definition of the proximal operators T_i and the assumptions. Here we rewrite the statements of the modified lemmas and assumptions.

We consider the updates made in the current round by all the cores in the ascending order of the actual time point (U_j^k in Figure 2) when the step size ε of the update is computed (breaking ties arbitrarily) and prove (7) by showing sufficient progress in between two successive updates in this order, however, under some assumptions similar to those used in Hsieh et al. (2015).

Because of the atomic updates, the step size computation may not include all the latest updates, however, we assume all the updates before the $(j - \gamma)$-th update have already been written into $\mathbf{v}$. Let $\bar{\mathbf{X}}_{[k]}$ denote the normalized data matrix where each row is $\bar{\mathbf{x}}_i^\top = \mathbf{x}_i^\top / \|\mathbf{x}_i\|, i \in \mathcal{I}_k$. Define $M_{[k],i} = \max_{\mathcal{D} \subseteq [d]} \| \sum_{t \in \mathcal{D}} \bar{\mathbf{X}}_{[k](:,t)} X_{[k](i,t)} \|$, $M = \max_k \max_i M_{[k],i}$, where $[d]$ is the set of all the feature indices, and $\bar{\mathbf{X}}_{[k](:,t)}$ is the t-th column of $\bar{\mathbf{X}}_{[k]}$. Moreover, R_{min} is defined as the minimum value of global data matrix, *i.e.*, $R_{min} = \min_{i=1,...,n} \|\mathbf{x}_i\|^2$.

Assumption 1 (Bounded delay of updates γ)

$$(\gamma + 1)^2 \leq \frac{\sqrt{n_k}}{6eM}, \; \text{where } e \text{ is the Euler's number.} \tag{8}$$

Definition 2 (Global error bound) *For a convex function $f : \mathbb{R}^n \to \mathbb{R}$, the optimization problem: $\min_{\boldsymbol{\beta}} f(\boldsymbol{\beta})$ admits a global error bound if there is a constant κ such that*

$$\|\boldsymbol{\beta} - P_S(\boldsymbol{\beta})\| \leq \kappa \|T(\boldsymbol{\beta}) - \boldsymbol{\beta}\|, \tag{9}$$

where $P_S(\cdot)$ is the Euclidean projection to the set of optimal solutions, and $T : \mathbb{R}^n \to \mathbb{R}^n$ is the operator defined as

$$T_i(\boldsymbol{\beta}) = \arg\min_u f\left(\boldsymbol{\beta} + (u - \beta_i)\mathbf{e}_t\right) \; \forall i \in [n].$$

The optimization problem admits a relaxed condition called global error bound from the beginning if (9) holds for any $\boldsymbol{\beta}$ satisfying $f(\boldsymbol{\beta}) \leq F$ for some constant F.

Assumption 2 *The local subproblem formulation (4) admits global error bound from the beginning for $F = Q(\boldsymbol{\delta}_{[k]}^{(j)}; \mathbf{v}^{(j)}, \boldsymbol{\alpha}_{[k]}^{(j)})$ and any update j.*

The global error bound helps prove that our subproblem solver achieves significant improvement after each update. It has been shown that when the loss functions are hinge loss or squared hinge loss, the local subproblem formulation (4) does indeed satisfy global error bound condition Hsieh et al. (2015).

Assumption 3 *The local subproblem objective (4) is L_{max}-Lipschitz continuous.*

Assumption 4 (Bounded M, L_{max})

$$2L_{max}\left(1+\frac{e^2\gamma^2M^2}{\sigma^2 n_k}\right)\left(\frac{e^2\gamma^2M^2}{\sigma^2 n_k}\right)\leq 1$$

Lemma 3 *When Assumptions 1-4 hold, the solutions computed in two successive updates by the* local subproblem *solver has a linear convergence rate in expectation, i.e.,*

$$\mathbb{E}\left[Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*)-Q_k^\sigma(\boldsymbol{\delta}_{[k]}^{(j)})\right]\leq \eta\left[Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*)-Q_k^\sigma(\boldsymbol{\delta}_{[k]}^{(j-1)})\right]$$

where $\boldsymbol{\delta}_{[k]}^{(j)}$ is the $\boldsymbol{\delta}_{[k]}$ after the jth update,

$$\eta = 1 - \frac{\kappa R_{min}}{2nL_{max}}\left(1-\frac{2L_{max}}{R_{min}}\left(1+\frac{e^2\gamma^2M^2}{\sigma^2\tilde{n}}\right)\left(\frac{e^2\gamma^2M^2}{\sigma^2\tilde{n}}\right)\right),$$

and $\tilde{n}=\max_k n_k$ is the size of the largest data part. Moreover, $\boldsymbol{\delta}_{[k]}^{(H)}$ is a Θ-approximate solution for

$$\Theta = \eta^H. \tag{10}$$

4.2 Convergence of global solution

Although we showed that the local subproblem solver outputs a Θ-approximate solution, we cannot directly apply the results of Ma et al. (2015a) for the global solution because our algorithm uses updates from only a subset of workers which is unlike the synchronous all-reduce of the updates from all workers used in Ma et al. (2015a). We need to handle this asynchronous nature of the global updates, just like we handled asynchronous updates for the local subproblem. Let us consider the global updates in the order the master computed them (at time U_t^m in Figure 2). Let $\boldsymbol{\alpha}^{(t)}$ denote the value of $\boldsymbol{\alpha}$ distributed across all the nodes at the time master computed tth global update $\mathbf{v}^{(t)}$. If $k\in\mathcal{P}_S^{(t)}$ then the update $\boldsymbol{\delta}_{[k]}^{(t)}$ has already been included in $\mathbf{v}^{(t)}$. However, if $k\notin\mathcal{P}_S^{(t)}$ then it may not be included. Let ξ be such that for all $l\leq\xi$ and for all k, $\boldsymbol{\delta}_{[k]}^{(l)}$ has been included in $\mathbf{v}^{(t)}$. By the design of our algorithm, $t-\Gamma\leq\xi\leq t-1$. Let $\hat{\boldsymbol{\alpha}}^{(t)}$ be defined as follows: $\hat{\boldsymbol{\alpha}}_{[k]}^{(t)}=\boldsymbol{\alpha}_{[k]}^{(t)},\forall k\in\mathcal{P}_S^{(t)}$ and $=\boldsymbol{\alpha}_{[k]}^{(\xi)}$ for the latest ξ for which the update is already included in global $\mathbf{v}$, $\forall k\notin\mathcal{P}_S^{(t)}$. Let $\mathbf{w}^{(t)},\hat{\mathbf{w}}^{(t)}$ be $\mathbf{w}(\boldsymbol{\alpha}^{(t)})$ and $\mathbf{w}(\hat{\boldsymbol{\alpha}}^{(t)})$ respectively. Note that $\mathbf{w}^{(t)}=\hat{\mathbf{w}}^{(t)}+\frac{1}{\lambda n}\sum_{l=\xi}^{t-1}\mathbf{X}\boldsymbol{\delta}^{(l)}$.

Lemma 4 *For any dual $\boldsymbol{\alpha}^{(t)},\boldsymbol{\delta}^{(t)}\in\mathbb{R}^n$, primal $\hat{\mathbf{w}}^{(t)}=\mathbf{w}(\hat{\boldsymbol{\alpha}}^{(t)})$ and real values ν,σ satisfying (5), it holds that*

$$D\left(\boldsymbol{\alpha}^{(t)}+\nu\sum_{k\in\mathcal{P}_S^{(t)}}\boldsymbol{\delta}_{[k]}^{(t)}\right)\geq(1-\nu)D(\hat{\boldsymbol{\alpha}}^{(t)})-\frac{\lambda}{2}(\left\|\mathbf{w}^{(t)}\right\|^2-\left\|\hat{\mathbf{w}}^{(t)}\right\|^2)$$

$$+\nu\sum_{k\in\mathcal{P}_S^{(t)}}Q_k^\sigma\left(\boldsymbol{\delta}_{[k]}^{(t)};\hat{\mathbf{w}}^{(t)},\boldsymbol{\alpha}_{[k]}^{(t)}\right)-\frac{\nu}{n}\sum_{k\in\mathcal{P}_S^{(t)}}(\mathbf{w}^{(t)}-\hat{\mathbf{w}}^{(t)})^\top\mathbf{X}\boldsymbol{\delta}_{[k]}^{(t)}. \tag{11}$$

10

Assumption 5 *There exists a $\varrho < e^{\frac{2}{\Gamma+1}}$ such that*

$$\left\| \boldsymbol{\delta}^{(t-1)} \right\|^2 \leq \varrho \left\| \boldsymbol{\delta}^{(t)} \right\|^2. \tag{12}$$

Lemma 5 *If ϕ_i^* are all $(1/\mu)$-strongly convex and Assumptions 1-5 are satisfied then for any $s \in [0,1]$, any round t of Algorithm 2 satisfies*

$$\mathbb{E}[D(\boldsymbol{\alpha}^{(t+1)}) - D(\boldsymbol{\alpha}^{(t)})] \geq \Psi(1-\Theta) \left(sG(\hat{\boldsymbol{\alpha}}) - \frac{\sigma}{2\lambda} \left(\frac{s}{n} \right)^2 \hat{R} \right) \tag{13}$$

where

$$\Psi := \nu \left(1 - \frac{\Gamma^2 e M L_{max}}{4\lambda n^2} - \frac{S M L_{max}}{4n} \right) \leq 1, \ and \tag{14}$$

$$\hat{R} := -\frac{\lambda \mu n(1-s)}{\sigma s} \|\hat{\mathbf{u}} - \hat{\boldsymbol{\alpha}}\|^2 + \sum_{k \in \mathcal{P}_S} \left\| \mathbf{X}(\hat{\mathbf{u}} - \hat{\boldsymbol{\alpha}})_{[k]} \right\|^2, \tag{15}$$

for $\hat{\mathbf{u}} \in \mathbb{R}^n$ with $-\hat{u}_i \in \partial \phi_i(\mathbf{w}(\hat{\boldsymbol{\alpha}})^\top \mathbf{x}_i)$.

Using the main results in Ma et al. (2015b) and combining Lemma 3 with Lemma 5 gives us the following two convergence results, one for smooth loss functions and the other for the Lipschitz continuous loss functions. The theorems use the quantities $\sigma_{max} = \max_k \sigma_k$, $\sigma_{sum} = \sum_k \sigma_k n_k$ where $\forall k, \sigma_k = \max_{\alpha_{[k]} \in \mathbb{R}^n} \left\| \mathbf{X}\boldsymbol{\alpha}_{[k]} \right\|^2 / \left\| \boldsymbol{\alpha}_{[k]} \right\|^2$.

Theorem 6 *If the loss functions ϕ_i are all $(1/\mu)$-smooth, then in T_1 iterations Algorithm 2 finds a solution with objective atmost ϵ_D from the optimal, i.e., $\mathbb{E}[D(\alpha^*) - D(\alpha^{(T_1)})] \leq \epsilon_D$ whenever $T_1 \geq C_1 \log \frac{1}{\epsilon_D}$ where $C_1 = \frac{1}{\Psi(1-\Theta)}(1 + \frac{\sigma_{max}\sigma}{\nu\lambda n})$ and Θ is given by (10). Furthermore, in T_2 iterations, it finds a solution with duality gap atmost ϵ_{gap}, i.e., $\mathbb{E}[P(\mathbf{w}(\alpha^{(T_2)})) - D(\alpha^{(T_2)})] \leq \epsilon_{gap}$ whenever $T_2 \geq C_1 \log \frac{C_1}{\epsilon_D}$.*

Theorem 7 *If the loss functions ϕ_i are all L-Lipschitz, then in T_1 iterations Algorithm 2 finds a solution with duality gap atmost ϵ_{gap}, i.e., $\mathbb{E}[P(\mathbf{w}(\bar{\alpha})) - D(\bar{\alpha})] \leq \epsilon_{gap}$ for the average iterate $\bar{\alpha} = \frac{1}{T_1 - T_0} \sum_{t=T_0+1}^{T_1-1} \alpha^{(t)}$ whenever $T_1 \geq T_0 + \max\{\lceil \frac{1}{\Psi(1-\Theta)} \rceil, \frac{4L^2\sigma_{sum}\sigma}{\lambda n^2 \epsilon_{gap} \Psi(1-\Theta)}\}$, and $T_0 \geq \max\{0, \lceil \frac{1}{\Psi(1-\Theta)} \log \frac{2\lambda n^2(D(\alpha^*) - D(\alpha^{(0)}))}{4L^2\sigma_{sum}\sigma} \rceil\} + \max\{0, \frac{2}{\Psi(1-\Theta)}(\frac{8L^2\sigma_{sum}\sigma}{\lambda n^2 \epsilon_{gap}} - 1)\}$ and Θ is given by (10).*

Theorem 7 establishes the convergence for L-Lipschitz continuous loss functions, and Theorem 6 proves a linear convergence rate for smooth convex loss functions.

5. Communication Cost Analysis

In each communication round, the algorithms based on synchronous updates on all K nodes require $2K$ transmissions, each consisting of all values of $\mathbf{v}$ or $\Delta\mathbf{v}$. Half of these transmissions are from the workers to the master and the rest are from the master to the workers. Whereas, our asynchronous update scheme requires $2S$ transmissions in each round.

Table 1: Datasets

Dataset	LIBSVM name	n	d	nnz	File size
rcv1	rcv1_test	677,399	47,236	49,556,258	1.2 GB
webspam	webspam	280,000	16,609,143	1,045,051,224	20 GB
kddb	kddb train	19,264,097	29,890,095	566,345,888	5.1 GB
splicesite	splice_site.t	4,627,840	11,725,480	15,383,587,858	280 GB

6. Experimental Results

We implemented our algorithm in C++ where each node runs exactly one MPI process which in turn runs one OpenMP thread on each core available within the node and the main thread handles the inter-node communication. We evaluated for hinge loss, though it applies to other loss functions too, on four datasets from LIBSVM website as shown in Table 1, using upto 16 nodes available with the Hornet cluster at University of Connecticut where each node has 24 Xeon E5-2690 cores and 128 GB main memory. The last column in Table 1 gives the total number of non-zero entries in the matrix X for each of the four dataset we used.

We experimented with the following algorithms: 1) *Baseline*: an implementation of DCA Hsieh et al. (2008), 2) *CoCoA+* Ma et al. (2015a), 3) *PassCoDe* Hsieh et al. (2015) and 4) our *Hybrid-DCA*. The algorithm parameters were varied as follows: 1) regularization parameter $\lambda \in \{10^{-3}, 10^{-4}, 10^{-5}\}$, 2) local iterations $H = \{10000, 20000, 40000\}$, 3) aggregation parameters $\nu = 1$, and 4) scaling parameter $\sigma = K, S$ for *CoCoA+*, *Hybrid-DCA*, respective, as recommended in Ma et al. (2015a). For different combinations of λ, H, we observed similar patterns of results and report for $\lambda = 10^{-4}, H = 40000$ only. The details of other parameter values are given later.

6.1 Comparison with existing algorithms

Figure 3 shows the progress of duality gap achieved by the four algorithms on three smaller datasets. We chose the number of nodes ($p \leq K$) and the number of cores ($t \leq R$) per node such that the total number of worker cores ($p \times t$) is the same (16) for all algorithms except *Baseline*. The duality gap is measured as $P(\mathbf{v}) - D(\boldsymbol{\alpha})$ where $\mathbf{v}$ is the estimate of $\mathbf{w}(\boldsymbol{\alpha})$ shared across the nodes. When $S < K$, it is not possible for the master in *Hybrid-DCA* to gather the parts of $P(\mathbf{v})$ from all workers at the end of each round. We let the master temporarily store $\mathbf{v}$ in disk after each round and at the end of all stipulated rounds, the workers compute the respective parts of $P(\mathbf{v})$ from the stored $\mathbf{v}$ and the master computes the duality gap using a series of synchronous all-reduce computations from all the workers. The bottom row shows the progress of the duality gap across time, while the top row shows progress across each round that consists of a communication round in *CoCoA+* and *Hybrid-DCA* whereas consists of H local updates in *Baseline* and *PassCoDe*. In this experiment, *Hybrid-DCA* uses $S = p$ and $\Gamma = 1$ making the global updates synchronous. The progress of baseline is slow as it applies only H updates compared to $H \times p \times t$ updates by the other algorithms. In terms of time, *Hybrid-DCA* clearly outperforms both *CoCoA+*, as expected, and *PassCoDe* which incurs a larger number of cache-misses when many cores are used. In terms of the number of rounds, *PassCoDe* outperforms both *CoCoA+* and *Hybrid-DCA*, as expected. However, *PassCoDe* is not scalable beyond the number of cores in a single node.
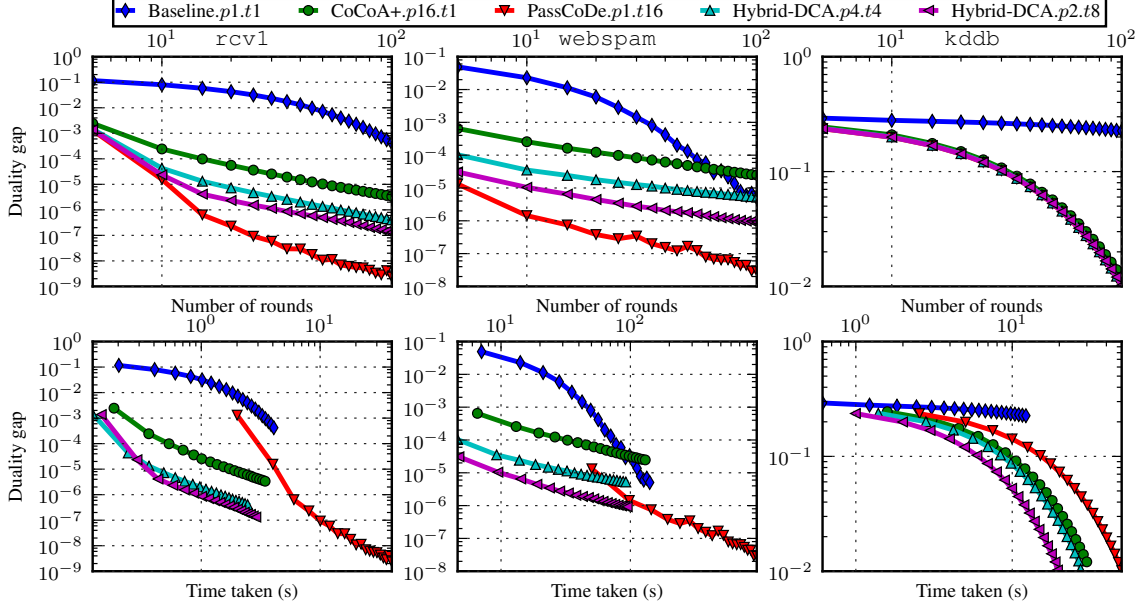
Figure 3: Performance of different solvers on three datasets, `rcv1` (left column), `webspam` (middle column), and `kddb` (right column), in terms of the progress of the duality gap across the number of rounds (top row) and across the wall time taken (bottom row).

As the number of nodes increases, the convergence of *Hybrid-DCA* becomes slower due to the costly merging process of many distributed updates.

6.2 Speedup

We ran sufficient rounds of each of the four algorithms such that the duality gap falls below a threshold and noted the time taken by the algorithms to achieve this threshold. Figure 4 shows the speedup(p, t) of all the algorithms except, *Baseline*, computed as the ratio of the time taken by *Baseline* to the time taken by the target algorithm on p nodes each with t cores. The thresholds used were $10^{-4}, 10^{-5}, 10^{-1}$ for `rcv1, webspam, kddb`, respectively. *PassCoDe* can be run only on a single node; so we vary only the number of cores. *CoCoA+* uses only 1 core per node. We ran *CoCoA+* and *Hybrid-DCA* on $p \in \{2, 4, 8, 16\}$ nodes and plotted them separately. For each p, *Hybrid-DCA* uses $t \in \{2, 4, 8, 16, 24\}$ cores per node, however, under the restriction that the total number of worker cores $(p \times t)$ does not exceed 144, a limit set by our HPC usage policy. When $t > 8$, the number of cache-misses increases due to thread switching on the physical cores and reduces speedup for both *PassCoDe* and *Hybrid-DCA*. This could be improved by carefully scheduling the OpenMP threads to the same physical cores. Nevertheless, *Hybrid-DCA* has good speedup for $t \leq 8$, as evident for $p = 16$.

6.3 Effects of the parameter S

Figure 5 shows the results of varying $S \in \{2, 3, 4, 6, 8\}$ with fixed $\Gamma = 10$ on $p = 8$ nodes each with $t = 8$ cores. When $S < p/2$, only a minority of the workers contribute in a round and the duality gap does not progress below some certain level. On the other hand, when at
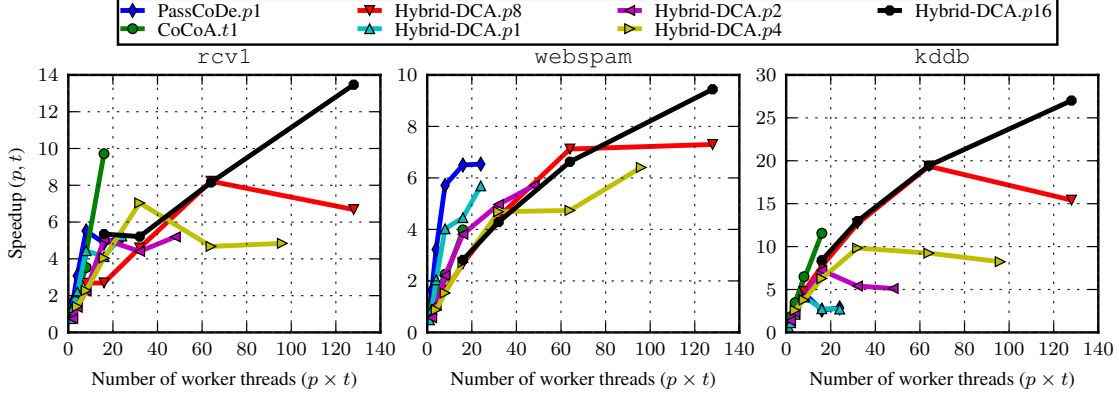
Figure 4: Speedup of different parallel or distributed solvers with respect to the sequential implementation *Baseline*.
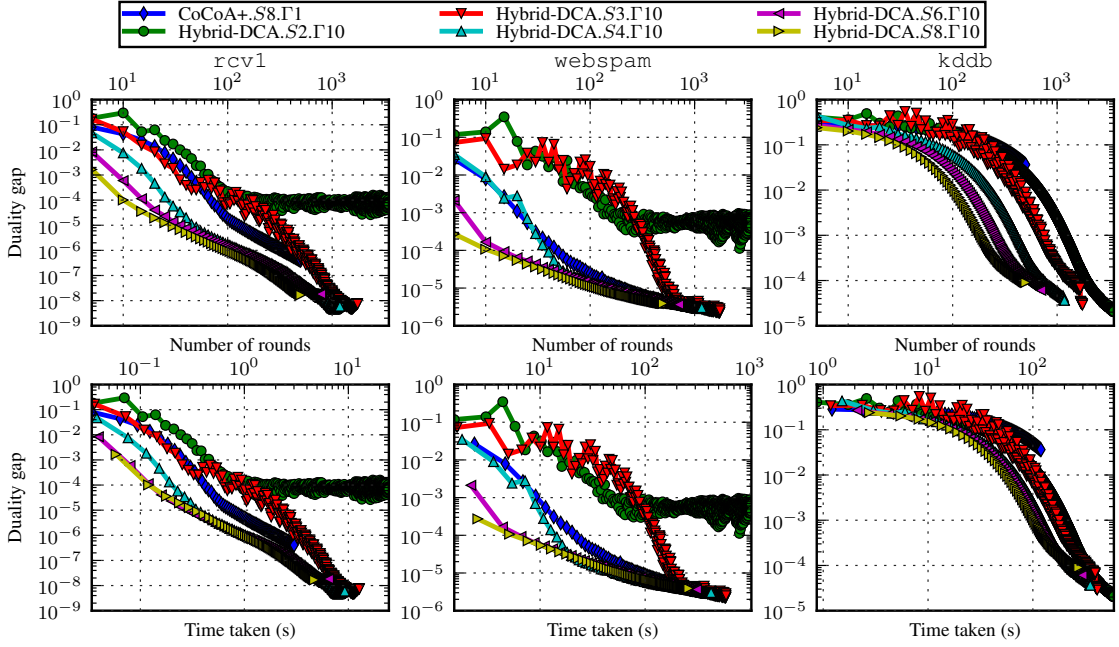

Figure 5: Effect of varying S on $p = 8$ worker nodes, with Γ fixed at 10.

least half of the workers contribute in each round, it is possible to achieve the same duality level obtained using all the workers. However, the reduction in time per round is eventually eaten by the larger number of rounds required to achieve the same duality gap. Nevertheless, the approach is useful for HPC platforms with heterogeneous nodes, unlike ours, where the waiting for updates from all workers has larger penalty per round, or for the case, where the need is to run for a specified number of rounds and quickly achieve a reasonably good duality gap.

6.4 Effects of the parameter Γ

Figure 6 shows the results of varying $\Gamma \in \{1, 2, 3, 4, 10\}$ with fixed $S = 6$ on $p = 8$ nodes each with $t = 8$ cores. We do not see much effect of Γ as the HPC platform used for our
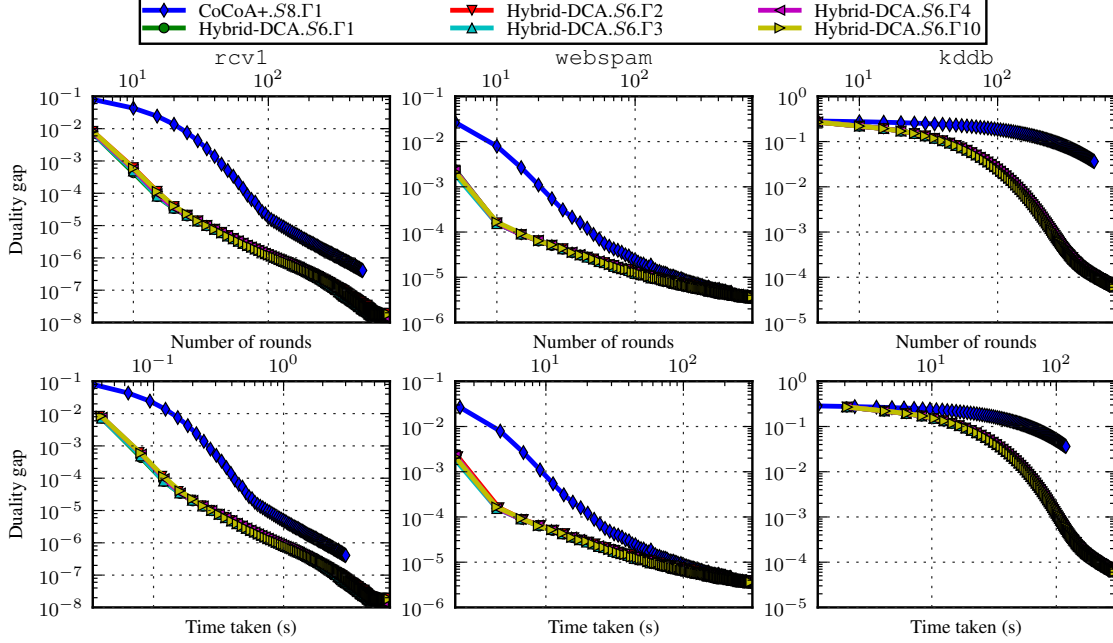
Figure 6: Effect of varying Γ on $p = 8$ worker nodes, with S fixed at 8.

experiments has homogeneous nodes. Our experimentation showed that even if we use $\Gamma = 10$, the stale value at any worker was for at most 4 rounds. We expect to see a larger variance of staleness in case of heterogeneous nodes.

6.5 Performance on a big dataset

We experimented our hybrid algorithm on the big dataset `splicesite` of size about 280 GB and compared with the previous best algorithm *CoCoA+*. Because of the enormous size, the dataset cannot be accommodated on a single node and hence *PassCoDe* cannot be run on this dataset. In this experiment, we used the number of local iterations $H = 10,000$. The results are shown in Figure 7 where the progress of duality gap across the rounds of communication is shown on the left and across the wall time on the right. To achieve a duality gap of at least 10^{-6} on 16 nodes, *CoCoA+* took more than 300 seconds which somewhat matches the 1200 seconds (20 minutes) time on 4 nodes reported in Ma et al. (2015a). *Hybrid-DCA* on 16 nodes each using 8 cores took about 30 seconds to achieve the same duality gap, showing enough evidence about the scalability of our algorithm. One could also argue that *CoCoA+* can be run on all these 16x8=128 cores, treating each core as a distributed node. We also experimented with this configuration which achieved better progress on duality gap than *CoCoA+* on 16 nodes, however, still performed far worse than *Hybrid-DCA* in terms of both the number of rounds and the time taken.

7. Conclusions

In this paper, we present a hybrid parallel and distributed asynchronous stochastic dual coordinate ascent algorithm utilizing modern HPC platforms with many nodes of multi-core shared-memory systems. We analyze the convergence properties of this novel algorithm which
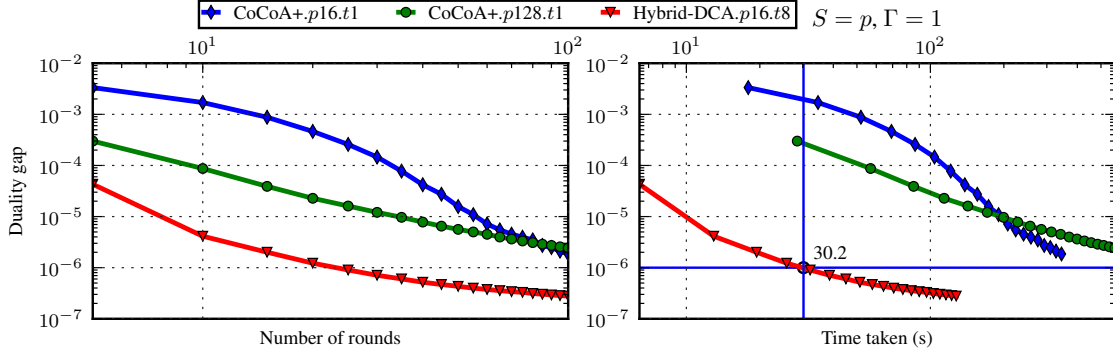
Figure 7: Performance of *Hybrid-DCA* on big dataset `splicesite`.

uses asynchronous updates at two cascading levels: inter-cores and inter-nodes. Experimental results show that our algorithm is faster than the state-of-the-art distributed algorithms and scales better than the state-of-the-art parallel algorithms.

References

Alekh Agarwal, Olivier Chapelle, Miroslav Dudík, and John Langford. A Reliable Effective Terascale Linear Learning System. *Journal of Machine Learning Research*, 15(1):1111–1133, 2014.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Christina Heinze, Brian McWilliams, and Nicolai Meinshausen. Dual-loco: Distributing statistical estimation using random projections. *arXiv preprint arXiv:1506.02554*, 2015.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the 25th international conference on Machine Learning (ICML)*, pages 408–415, 2008.

Cho-Jui Hsieh, Hsiang-Fu Yu, and Inderjit S Dhillon. PASSCoDe: Parallel ASynchronous Stochastic dual Co-ordinate Descent. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015.

Martin Jaggi, Virginia Smith, Martin Takác, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I Jordan. Communication-Efficient Distributed Dual Coordinate Ascent. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3068–3076, 2014.

Jason Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. Distributed stochastic variance reduced gradient methods and a lower bound for communication complexity. *CoRR*, arXiv:1507.07595, 2015.

Ji Liu, Stephen J Wright, Christopher Ré, Victor Bittorf, and Srikrishna Sridhar. An Asynchronous Parallel Stochastic Coordinate Descent Algorithm. *Journal of Machine Learning Research*, 16: 285–322, 2015.

Chenxin Ma, Jakub Konečnỳ, Martin Jaggi, Virginia Smith, Michael I Jordan, Peter Richtárik, and Martin Takáč. Distributed optimization with arbitrary local solvers. *arXiv preprint arXiv:1512.04039*, 2015a.

Chenxin Ma, Virginia Smith, Martin Jaggi, Michael I Jordan, Peter Richtárik, and Martin Takáč. Adding vs. Averaging in Distributed Primal-Dual Optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015b.

Ryan Mcdonald, Mehryar Mohri, Nathan Silberman, Dan Walker, and Gideon S Mann. Efficient large-scale distributed training of conditional maximum entropy models. In *Advances in Neural Information Processing Systems*, pages 1231–1239, 2009.

Brian McWilliams, Christina Heinze, Nicolai Meinshausen, Gabriel Krummenacher, and Hastagiri P Vanchinathan. Loco: Distributing ridge regression with random projections. *arXiv preprint arXiv:1406.3469*, 2014.

Eric Moulines and Francis R Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, pages 451–459, 2011.

Deanna Needell, Rachel Ward, and Nati Srebro. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Advances in Neural Information Processing Systems*, pages 1017–1025, 2014.

Zhimin Peng, Yangyang Xu, Ming Yan, and Wotao Yin. Arock: an algorithmic framework for asynchronous parallel coordinate updates. *arXiv preprint arXiv:1506.02396*, 2015.

Peter Richtárik and Martin Takáč. Distributed coordinate descent method for learning with big data. *arXiv preprint arXiv:1310.2059*, 2013.

Shai Shalev-Shwartz and Tong Zhang. Stochastic Dual Coordinate Ascent Methods for Regularized Loss Minimization. *Journal of Machine Learning Research*, 14(1):567–599, 2013.

Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, 155(1-2):105–145, 2016.

Ohad Shamir, Nathan Srebro, and Tong Zhang. Communication-Efficient Distributed Optimization using an Approximate Newton-type Method. In *Proceedings of the 31th International Conference on Machine Learning, (ICML), Beijing, China, 21-26 June 2014*, pages 1000–1008, 2014.

Martin Takac, Avleen Bijral, Peter Richtarik, and Nati Srebro. Mini-Batch Primal and Dual Methods for SVMs. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1022–1030, 2013.

Tianbao Yang. Trading Computation for Communication: Distributed Stochastic Dual Coordinate Ascent. In *Advances in Neural Information Processing Systems*, pages 629–637, 2013.

Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1-2):41–75, 2011.

Ruiliang Zhang and James Kwok. Asynchronous distributed ADMM for consensus optimization. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1701–1709, 2014.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 116. ACM, 2004.

Yuchen Zhang and Lin Xiao. Communication-efficient distributed optimization of self-concordant empirical loss. *CoRR*, abs/1501.00263, 2015.

Hybrid-DCA: A Double Asynchronous Approach for Stochastic Dual Coordinate Ascent

Appendix A. Proof of near optimality of local subproblem solution

Definition 8 *First, we define that:*

$$
\beta_t^{l+1} = \begin{cases} T_t\left(\hat{\mathbf{w}}^l, \beta_t^l\right) & \text{if } t = i(l), \\ \beta_t^l & \text{if } t \neq i(l) \end{cases}, \qquad\qquad \varepsilon^l = \beta_{i(l)}^{l+1} - \beta_{i(l)}^l,
$$

$$
\tilde{\beta}^{l+1} = T\left(\hat{\mathbf{w}}^l, \boldsymbol{\beta}^l\right), \qquad\qquad\qquad\qquad \bar{\beta}^{l+1} = T\left(\bar{\mathbf{w}}^l, \boldsymbol{\beta}^l\right),
$$

where $\boldsymbol{\beta}^l$ denotes the sequence generated by the local atomic solver *and $\hat{\mathbf{w}}^l$ denotes the actual values of w maintained at update l in the* local atomic solver. *Note that, $\tilde{\beta}_{i(l)}^{l+1} = \beta_{i(l)}^{l+1}$ and $\tilde{\beta}^{l+1} = \text{prox}\left(\boldsymbol{\beta}^l - \frac{\lambda n}{\sigma}\bar{\mathbf{X}}\hat{\mathbf{w}}^l\right).$*

Lemma 9 *Under Assumption 1, and let $\boldsymbol{\beta}_{[k]}^l = \boldsymbol{\alpha}_{[k]} + \nu\boldsymbol{\delta}_{[k]}^l$, $\rho = \left(1 + \frac{6(\gamma+1)eM}{\sqrt{n_k}}\right)^2$. Then, the local subproblem satisfy:*

$$
\mathbb{E}\left[\left\|\boldsymbol{\beta}_{[k]}^{l-1} - \tilde{\boldsymbol{\beta}}_{[k]}^l\right\|^2\right] \leq \rho\mathbb{E}\left[\left\|\boldsymbol{\beta}_{[k]}^l - \tilde{\boldsymbol{\beta}}_{[k]}^{l+1}\right\|^2\right], \tag{16}
$$

where $l \neq h$, represents the l-th update to ω in a local solver.

A.1 Proof of Lemma 9

Proof We omit the subscript $_{[k]}$ of the notations, which specifies the k-th data partition, in the proof. For all $i \in \mathcal{I}_k$, we have following definitions:

$$
h_i(u) := \frac{\phi_i^*(-u)}{n\|\mathbf{x}_i\|^2} + \frac{\lambda}{2}\left(\frac{1}{S} - \frac{1}{\sigma}\right)\frac{\|\mathbf{w}\|^2}{\|\mathbf{x}_i\|^2}
$$

$$
\text{prox}_i(s) := \underset{u}{\text{argmin}}\ \frac{1}{2}(u-s)^2 + h_i(u)
$$

$$
\begin{aligned}
T_i(\mathbf{w}, s) :=& \underset{u}{\text{argmax}} -\frac{1}{\sigma}\frac{\lambda}{2}\|\mathbf{w}\|^2 - \frac{1}{n}\mathbf{w}^\top\mathbf{x}_i(u-s) - \frac{\lambda}{2}\sigma(\frac{1}{\lambda n}\mathbf{x}_i(u-s))^2 \\
& -\frac{1}{n}\phi_i^*(-u) - \frac{\lambda}{2}\left(\frac{1}{K}\|\mathbf{w}\|^2 - \frac{1}{\sigma}\|\mathbf{w}\|^2\right) \\
=& \underset{u}{\text{argmax}} -\frac{\lambda}{2}\left\|\frac{\mathbf{w}}{\sqrt{\sigma}} + \frac{\sqrt{\sigma}}{\lambda n}(u-s)\mathbf{x}_i\right\|^2 - \|\mathbf{x}_i\|^2 h_i(u) \\
=& \underset{u}{\text{argmin}}\ \frac{1}{2}\left(u - \left(s - \frac{\lambda n\mathbf{w}^\top\mathbf{x}_i}{\sigma\|\mathbf{x}_i\|^2}\right)\right)^2 + h_i(u),
\end{aligned}
$$

where $\mathbf{w} \in \mathbb{R}^d$ denotes any fixed vector and $s \in \mathbb{R}$. $\text{prox}(s)$ denotes the proximal operator. We can see the connection of above operator and proximal operator: $T_i(\mathbf{w}, s) =$

$\text{prox}_i \left(s - \frac{\mathbf{w}^\top \mathbf{x}_i}{\sigma \|\mathbf{x}_i\|^2} \right)$. Here both $h_i(u)$ and $T_i(\wedge, s)$ were revised from Hsieh et al. (2015) to satisfy the subproblem 4.

Proposition 10

$$\mathbb{E}_{i(l)} \left(\left\| \boldsymbol{\beta}^{l+1} - \boldsymbol{\beta}^l \right\|^2 \right) = \frac{1}{n} \left\| \tilde{\boldsymbol{\beta}}^{l+1} - \boldsymbol{\beta}^l \right\|^2, \tag{17}$$

Proposition 11

$$\left\| \bar{\mathbf{X}} \bar{\mathbf{w}}^j - \bar{\mathbf{X}} \hat{\mathbf{w}}^j \right\| \leq \frac{1}{\lambda n} M \sum_{t=j-\gamma}^{j-1} |\varepsilon^t|, \tag{18}$$

Proposition 12

$$\left| T_i \left(\mathbf{w}_1, s_1 \right) - T_i \left(\mathbf{w}_2, s_2 \right) \right| \leq \left| s_1 - s2 + \frac{(\mathbf{w}_1 - \mathbf{w}_2)^\top \mathbf{x}_i}{\|\mathbf{x}_i\|^2} \right|, \tag{19}$$

Proposition 13 *Let $M \geq 1$, $q = \frac{6(\gamma+1)eM}{\sqrt{n}}$, $\rho = (1+q)^2$, and $\theta = \sum_{t=1}^{\gamma} \rho^{t/2}$. If $q(\gamma+1) \leq 1$ and $\sigma \geq 1$, then $\rho^{(\gamma+1)/2} \leq e$, and*

$$\rho^{-1} \leq 1 - \frac{4}{\sqrt{n}} - \frac{4M + 4M\theta}{\sqrt{n}} \leq 1 - \frac{4}{\sqrt{n}} - \frac{4M + 4M\theta}{\sigma\sqrt{n}}, \tag{20}$$

Proposition 14 *For all $j > 0$, we have*

$$D \left(\boldsymbol{\alpha}^j \right) \leq D \left(\bar{\boldsymbol{\alpha}}^{j+1} \right) - \frac{\sigma \left\| \mathbf{x}_{i(j)} \right\|^2}{2} \left\| \boldsymbol{\alpha}^j - \bar{\boldsymbol{\alpha}}^{j+1} \right\|^2, \tag{21}$$

$$D \left(\boldsymbol{\alpha}^j \right) \geq D \left(\bar{\boldsymbol{\alpha}}^{j+1} \right) - \frac{L_{max}}{2} \left\| \boldsymbol{\alpha}^j - \bar{\boldsymbol{\alpha}}^{j+1} \right\|^2 \tag{22}$$

Above propositions are similar to Hsieh et al. (2015). We keep the conclusions of those propositions for the future use in our proof. We prove Eq (16) by induction. As shown in Hsieh et al. (2015), we have

$$\left\| \boldsymbol{\beta}^{l-1} - \tilde{\boldsymbol{\beta}}^l \right\|^2 - \left\| \boldsymbol{\beta}^l - \tilde{\boldsymbol{\beta}}^{l+1} \right\|^2 \leq 2 \left\| \boldsymbol{\beta}^{l-1} - \tilde{\boldsymbol{\beta}}^l \right\| \left\| \boldsymbol{\beta}^l - \tilde{\boldsymbol{\beta}}^{l+1} - \boldsymbol{\beta}^{l-1} + \tilde{\boldsymbol{\beta}}^l \right\|. \tag{23}$$

The second of factor in the r.h.s of Eq 23 is bounded as follows with the revisions:

$$\left\| \boldsymbol{\beta}^l - \tilde{\boldsymbol{\beta}}^{l+1} - \boldsymbol{\beta}^{l-1} + \tilde{\boldsymbol{\beta}}^l \right\|$$

$$\leq \left\| \boldsymbol{\beta}^l - \boldsymbol{\beta}^{l-1} \right\| + \left\| \text{prox} \left(\boldsymbol{\beta}^l - \frac{\lambda n}{\sigma} \bar{\mathbf{X}} \hat{\mathbf{w}}^l \right) - \text{prox} \left(\boldsymbol{\beta}^{l-1} - \frac{\lambda n}{\sigma} \bar{\mathbf{X}} \hat{\mathbf{w}}^{l-1} \right) \right\|$$

$$\leq \left\| \boldsymbol{\beta}^l - \boldsymbol{\beta}^{l-1} \right\| + \left\| \left(\boldsymbol{\beta}^l - \frac{\lambda n}{\sigma} \bar{\mathbf{X}} \hat{\mathbf{w}}^l \right) - \left(\boldsymbol{\beta}^{l-1} - \frac{\lambda n}{\sigma} \bar{\mathbf{X}} \hat{\mathbf{w}}^{l-1} \right) \right\|$$

$$\leq 2 \left\| \boldsymbol{\beta}^l - \boldsymbol{\beta}^{l-1} \right\| + \frac{\lambda n}{\sigma} \left\| \bar{\mathbf{X}} \hat{\mathbf{w}}^l - \bar{\mathbf{X}} \hat{\mathbf{w}}^{l-1} \right\|$$

$$= 2 \left\| \boldsymbol{\beta}^l - \boldsymbol{\beta}^{l-1} \right\| + \frac{\lambda n}{\sigma} \left\| \bar{\mathbf{X}} \hat{\mathbf{w}}^l - \bar{\mathbf{X}} \bar{\mathbf{w}}^l + \bar{\mathbf{X}} \bar{\mathbf{w}}^l - \bar{\mathbf{X}} \bar{\mathbf{w}}^{l-1} + \bar{\mathbf{X}} \bar{\mathbf{w}}^{l-1} - \bar{\mathbf{X}} \hat{\mathbf{w}}^{l-1} \right\|$$

$$\leq 2 \left\| \boldsymbol{\beta}^l - \boldsymbol{\beta}^{l-1} \right\| + \frac{\lambda n}{\sigma} \left(\left\| \bar{\mathbf{X}} \bar{\mathbf{w}}^l - \bar{\mathbf{X}} \bar{\mathbf{w}}^{l-1} \right\| + \left\| \bar{\mathbf{X}} \hat{\mathbf{w}}^l - \bar{\mathbf{X}} \bar{\mathbf{w}}^l \right\| + \left\| \bar{\mathbf{X}} \bar{\mathbf{w}}^{l-1} - \bar{\mathbf{X}} \hat{\mathbf{w}}^{l-1} \right\| \right)$$

$$\leq \left(2 + 2 \frac{\lambda n}{\sigma} \frac{M}{\lambda n} \right) \left\| \boldsymbol{\beta}^l - \boldsymbol{\beta}^{l-1} \right\| + 2 \frac{\lambda n}{\sigma} \frac{M}{\lambda n} \sum_{t=l-\gamma-1}^{l-2} |\varepsilon^t| \quad \text{(Proposition 11)} \tag{24}$$

$$\leq \left(2 + 2 \frac{M}{\sigma} \right) \left\| \boldsymbol{\beta}^l - \boldsymbol{\beta}^{l-1} \right\| + 2 \frac{M}{\sigma} \sum_{t=l-\gamma-1}^{l-2} |\varepsilon^t| \tag{25}$$

No we start the induction. Although some steps may be the same as the steps in Hsieh et al. (2015), we still keep them here to make the proof self-contained.

Induction Hypothesis. We prove the following equivalent statement. For all j,

$$\mathbb{E} \left(\left\| \boldsymbol{\beta}^{l-1} - \tilde{\boldsymbol{\beta}}^l \right\|^2 \right) \leq \rho \mathbb{E} \left(\left\| \boldsymbol{\beta}^l - \tilde{\boldsymbol{\beta}}^{l+1} \right\|^2 \right),$$

Induction Basis. When $l = 1$,

$$\mathbb{E} \left(\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 \right) - \mathbb{E} \left(\left\| \boldsymbol{\beta}^1 - \tilde{\boldsymbol{\beta}}^2 \right\|^2 \right) \leq 2E \left(\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\| \left\| \boldsymbol{\beta}^1 - \tilde{\boldsymbol{\beta}}^2 - \boldsymbol{\beta}^0 + \tilde{\boldsymbol{\beta}}^1 \right\| \right)$$

$$\leq \left(4 + 4 \frac{M}{2} \right) \mathbb{E} (\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\| \left\| \boldsymbol{\beta}^0 - \boldsymbol{\beta}^1 \right\|).$$

By Proposition 10 and AM-GM inequality, which for any $b_1, b_2 > 0$ and any $c > 0$, we have

$$b_1 b_2 \leq \frac{1}{2} \left(c b_1^2 + c^{-1} b_2^2 \right) \tag{26}$$

Therefore, we have

$$\mathbb{E} \left(\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\| \left\| \boldsymbol{\beta}^0 - \boldsymbol{\beta}^1 \right\| \right) \leq \frac{1}{2} \mathbb{E} \left(\sqrt{n} \left\| \boldsymbol{\beta}^0 - \boldsymbol{\beta}^1 \right\|^2 + \frac{1}{\sqrt{n}} \left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 \right)$$

$$= \frac{1}{2} \mathbb{E} \left(\frac{1}{\sqrt{n}} \left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 + \frac{1}{\sqrt{n}} \left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 \right) \quad \text{(Proposition 10)}$$

$$= \frac{1}{\sqrt{n}} \mathbb{E} \left(\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 \right)$$

Therefore,

$$\mathbb{E} \left(\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 \right) - \mathbb{E} \left(\left\| \boldsymbol{\beta}^1 - \tilde{\boldsymbol{\beta}}^2 \right\|^2 \right) \leq \left(\frac{4}{\sqrt{n}} + \frac{4M}{\sigma \sqrt{n}} \right) \mathbb{E} \left(\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 \right),$$

which implies

$$\mathbb{E} \left(\left\| \boldsymbol{\beta}^0 - \tilde{\boldsymbol{\beta}}^1 \right\|^2 \right) \leq \left(1 - \frac{4}{\sqrt{n}} - \frac{4M}{\sigma \sqrt{n}} \right)^{-1} \mathbb{E} \left(\left\| \boldsymbol{\beta}^1 - \tilde{\boldsymbol{\beta}}^2 \right\|^2 \right) \leq \rho \mathbb{E} \left(\left\| \boldsymbol{\beta}^1 - \tilde{\boldsymbol{\beta}}^2 \right\|^2 \right),$$

where the last inequality is base on Proposition 13 and the fact $\theta M \geq 1$.

Induction Step. By the induction hypothesis, we assume

$$\mathbb{E} \left(\left\| \boldsymbol{\beta}^{t-1} - \tilde{\boldsymbol{\beta}}^t \right\|^2 \right) \leq \rho \mathbb{E} \left(\left\| \boldsymbol{\beta}^t - \tilde{\boldsymbol{\beta}}^{t+1} \right\|^2 \right) \quad \forall t \leq l - 1. \tag{27}$$

To show

$$\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right) \leq \rho \mathbb{E}\left(\left\|\boldsymbol{\beta}^{l}-\tilde{\boldsymbol{\beta}}^{l+1}\right\|^{2}\right),$$

we firstly show that for all $t < j$,

$$\begin{aligned}
&\mathbb{E}\left(\left\|\boldsymbol{\beta}^{t}-\boldsymbol{\beta}^{t+1}\right\|\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|\right)\\
\leq&\frac{1}{2}\mathbb{E}\left(\sqrt{n}\rho^{(t+1-l)/2}\left\|\boldsymbol{\beta}^{t}-\boldsymbol{\beta}^{t+1}\right\|^{2}+\frac{1}{\sqrt{n}}\rho^{(l-1-t)/2}\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right) \text{ (Eq. 26)}\\
=&\frac{1}{2}\mathbb{E}\left(\sqrt{n}\rho^{(t+1-l)/2}\mathbb{E}\left(\left\|\boldsymbol{\beta}^{t}-\boldsymbol{\beta}^{t+1}\right\|^{2}\right)+\frac{1}{\sqrt{n}}\rho^{(l-1-t)/2}\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right)\\
=&\frac{1}{2}\mathbb{E}\left(\frac{1}{\sqrt{n}}\rho^{(t+1-l)/2}\left\|\boldsymbol{\beta}^{t}-\tilde{\boldsymbol{\beta}}^{t+1}\right\|^{2}+\frac{1}{\sqrt{n}}\rho^{(l-1-t)/2}\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right) \text{ (Proposition 10)}\\
\leq&\frac{1}{2}\mathbb{E}\left(\frac{1}{\sqrt{n}}\rho^{(t+1-l)/2}\rho^{l-t-1}\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}+\frac{1}{\sqrt{n}}\rho^{(l-1-t)/2}\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right) \text{ (Eq. 27)}\\
\leq&\frac{\rho^{(l-1-t)/2}}{\sqrt{n}}\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right). \hspace{4cm} (28)
\end{aligned}$$

Let $\theta = \sum_{t=1}^{\gamma}\rho^{t/2}$. We have

$$\begin{aligned}
&\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right)-\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l}-\tilde{\boldsymbol{\beta}}^{l+1}\right\|\right)\\
\leq&\mathbb{E}\left(2\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|\left(\left(2+2\frac{M}{\sigma}\right)\left\|\boldsymbol{\beta}^{l-1}-\boldsymbol{\beta}^{l}\right\|+2\frac{M}{\sigma}\left\|\boldsymbol{\beta}^{t-1}-\boldsymbol{\beta}^{t}\right\|\right)\right) \text{ (Eq. 23, Eq. 24)}\\
=&\left(4+4\frac{M}{\sigma}\right)\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|\left\|\boldsymbol{\beta}^{l-1}-\boldsymbol{\beta}^{l}\right\|\right)+4\frac{M}{\sigma}\sum_{t=l-\gamma-1}^{l-1}\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|\left\|\boldsymbol{\beta}^{t-1}-\boldsymbol{\beta}^{t}\right\|\right)\\
\leq&\frac{4\sigma+4M}{\sigma\sqrt{n}}\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right)+\frac{4M}{\sigma\sqrt{n}}\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|\right)\sum_{t=l-\gamma-1}^{l-2}\rho^{(l-1-t)/2} \text{ (Eq. 28)}\\
\leq&\frac{4\sigma+4M}{\sigma\sqrt{n}}\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right)+\frac{4M}{\sigma\sqrt{n}}\theta\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|\right)\\
\leq&\left(\frac{4}{\sqrt{n}}+\frac{4M+4M\theta}{\sigma\sqrt{n}}\right)\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right)
\end{aligned}$$

which implies that

$$\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right) \leq \frac{1}{1-\frac{4}{\sqrt{n}}-\frac{4M+4M\theta}{\sigma\sqrt{n}}}\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l}-\tilde{\boldsymbol{\beta}}^{l+1}\right\|^{2}\right) \leq \rho\mathbb{E}\left(\left\|\boldsymbol{\beta}^{l-1}-\tilde{\boldsymbol{\beta}}^{l}\right\|^{2}\right)$$

by Proposition 13. $\blacksquare$

A.2 Proof of Lemma 3

Proof We also omit the subscript $_{[k]}$ of the notations in the proof. We can bound the expected distance $\mathbb{E}\left(\left\|\bar{\boldsymbol{\beta}}^{j+1} - \tilde{\boldsymbol{\beta}}^{l+1}\right\|^2\right)$ by the following derivation.

$$
\begin{aligned}
\mathbb{E}\left(\left\|\bar{\boldsymbol{\beta}}^{l+1} - \tilde{\boldsymbol{\beta}}^{l+1}\right\|^2\right) &= \mathbb{E}\left(\sum_{t=1}^{n}\left(T_t(\bar{\mathbf{w}}^l, \beta_t^l) - T_t(\hat{\mathbf{w}}^l, \beta_t^l)\right)^2\right) \\
&\leq \mathbb{E}\left(\sum_{t=1}^{n}\left(\frac{\lambda n\left(\bar{\mathbf{w}}^l - \hat{\mathbf{w}}^l\right)^\top \mathbf{x}_t}{\sigma\left\|\mathbf{x}_t\right\|^2}\right)^2\right) && \text{(Proposition 12)} \\
&= \frac{\lambda^2 n^2}{\sigma^2}\mathbb{E}\left(\left\|\bar{\mathbf{X}}\left(\bar{\mathbf{w}}^l - \hat{\mathbf{w}}^l\right)\right\|^2\right) \\
&\leq \frac{M^2}{\lambda^2 n^2}\frac{\lambda^2 n^2}{\sigma^2}\mathbb{E}\left(\left(\sum_{t=l-\gamma}^{l-1}\left\|\boldsymbol{\beta}^t - \boldsymbol{\beta}^{t+1}\right\|\right)^2\right) && \text{(Proposition 11)} \\
&\leq \frac{M^2}{\sigma^2}\mathbb{E}\left(\gamma\left(\sum_{t=l-\gamma}^{l-1}\left\|\boldsymbol{\beta}^t - \boldsymbol{\beta}^{t+1}\right\|^2\right)\right) && \text{(Cauchy Schwarz Inequality)} \\
&\leq \frac{\gamma M^2}{\sigma^2}\mathbb{E}\left(\gamma\left(\sum_{t=1}^{\gamma}\rho^t\left\|\boldsymbol{\beta}^l - \boldsymbol{\beta}^{l+1}\right\|^2\right)\right) && \text{(Lemma 9)} \\
&\leq \frac{\gamma M^2}{\sigma^2 n}\left(\sum_{t=1}^{\gamma}\rho^t\right)\mathbb{E}\left(\left\|\boldsymbol{\beta}^l - \tilde{\boldsymbol{\beta}}^{l+1}\right\|^2\right) && \text{(Proposition 10)} \\
&\leq \frac{\gamma^2 M^2}{\sigma^2 n}\rho^\gamma\mathbb{E}\left(\left\|\boldsymbol{\beta}^l - \tilde{\boldsymbol{\beta}}^{l+1}\right\|^2\right) \\
&\leq \frac{\gamma^2 M^2 e^2}{\sigma^2 n}\mathbb{E}\left(\left\|\boldsymbol{\beta}^l - \tilde{\boldsymbol{\beta}}^{l+1}\right\|^2\right). && \text{(Proposition 13)} && (29)
\end{aligned}
$$

$$(30)$$

Moreover,

$$
\begin{aligned}
\mathbb{E}\left(\left\|\bar{\boldsymbol{\beta}}^l - \boldsymbol{\beta}^{l+1}\right\|^2\right) &= \mathbb{E}\left(\left\|\bar{\boldsymbol{\beta}}^{l+1} - \tilde{\boldsymbol{\beta}}^{l+1} + \tilde{\boldsymbol{\beta}}^{l+1} - \boldsymbol{\beta}^l\right\|^2\right) \\
&\leq \mathbb{E}\left(2\left(\left\|\bar{\boldsymbol{\beta}}^{l+1} - \tilde{\boldsymbol{\beta}}^{l+1}\right\|^2 + \left\|\tilde{\boldsymbol{\beta}}^{l+1} - \boldsymbol{\beta}^l\right\|^2\right)\right) && \text{(Cauchy} - \text{Schwarz)} \\
&\leq 2\left(1 + \frac{\gamma^2 M^2 e^2}{\sigma^2 n}\right)\mathbb{E}\left(\left\|\tilde{\boldsymbol{\beta}}^{l+1} - \boldsymbol{\beta}^l\right\|^2\right) && (31)
\end{aligned}
$$

The bound of the increase of local objective function value by

$$
\mathbb{E}\left(D\left(\boldsymbol{\beta}^{l+1}\right)\right) - \mathbb{E}\left(D\left(\boldsymbol{\beta}^l\right)\right)
$$

$$=\mathbb{E}\left(-\left(D\left(\boldsymbol{\beta}^l\right)-D\left(\bar{\boldsymbol{\beta}}^{l+1}\right)\right)\right)-\mathbb{E}\left(\left(D\left(\bar{\boldsymbol{\beta}}^{l+1}\right)-D\left(\boldsymbol{\beta}^{l+1}\right)\right)\right)$$

$$\geq\mathbb{E}\left(\frac{\sigma\left\|\mathbf{x}_{i(l)}\right\|^2}{2}\left\|\boldsymbol{\beta}^l-\bar{\boldsymbol{\beta}}^{l+1}\right\|^2\right)-\mathbb{E}\left(\frac{L_{max}}{2}\left\|\boldsymbol{\beta}^{l+1}-\bar{\boldsymbol{\beta}}^{l+1}\right\|^2\right) \quad \text{(Proposition 14)}$$

$$\geq\frac{R_{min}}{2n}\mathbb{E}\left(\left\|\boldsymbol{\beta}^l-\bar{\boldsymbol{\beta}}^{l+1}\right\|^2\right)-\frac{L_{max}}{2n}\mathbb{E}\left(\left\|\tilde{\boldsymbol{\beta}}^{l+1}-\bar{\boldsymbol{\beta}}^{l+1}\right\|^2\right)$$

$$\geq\frac{R_{min}}{2n}\mathbb{E}\left(\left\|\boldsymbol{\beta}^l-\bar{\boldsymbol{\beta}}^{l+1}\right\|^2\right)-\frac{L_{max}}{2n}\frac{\gamma^2M^2e^2}{\sigma^2n}\mathbb{E}\left(\left\|\tilde{\boldsymbol{\beta}}^{l+1}-\boldsymbol{\beta}^l\right\|^2\right) \quad \text{(Eq. 29)}$$

$$\geq\frac{R_{min}}{2n}\mathbb{E}\left(\left\|\boldsymbol{\beta}^l-\bar{\boldsymbol{\beta}}^{l+1}\right\|^2\right)-\frac{2L_{max}}{2n}\frac{\gamma^2M^2e^2}{\sigma^2n}\left(1+\frac{\gamma^2M^2e^2}{\sigma^2n}\right)\mathbb{E}\left(\left\|\bar{\boldsymbol{\beta}}^{l+1}-\boldsymbol{\beta}^l\right\|^2\right) \quad \text{(Eq. 31)}$$

$$\geq\frac{R_{min}}{2n}\left(1-\frac{2L_{max}}{2n}\left(1+\frac{\gamma^2M^2e^2}{\sigma^2n}\right)\left(\frac{\gamma^2M^2e^2}{\sigma^2n}\right)\right)\mathbb{E}\left(\left\|\bar{\boldsymbol{\beta}}^{l+1}-\boldsymbol{\beta}^l\right\|^2\right)$$

$$\geq\frac{\kappa R_{min}}{2n}\left(1-\frac{2L_{max}}{2n}\left(1+\frac{\gamma^2M^2e^2}{\sigma^2n}\right)\left(\frac{\gamma^2M^2e^2}{\sigma^2n}\right)\right)\mathbb{E}\left(\left\|P_S\left(\boldsymbol{\beta}^l\right)-\boldsymbol{\beta}^l\right\|^2\right)$$

$$\geq\frac{\kappa R_{min}}{2nL_{max}}\left(1-\frac{2L_{max}}{2n}\left(1+\frac{\gamma^2M^2e^2}{\sigma^2n}\right)\left(\frac{\gamma^2M^2e^2}{\sigma^2n}\right)\right)\mathbb{E}\left(D^*-D\left(\boldsymbol{\beta}^l\right)\right)$$

Therefore,

$$D^*-\mathbb{E}\left(D\left(\boldsymbol{\beta}^{l+1}\right)\right)=D^*-\mathbb{E}\left(D\left(\boldsymbol{\beta}^l\right)\right)-\left(\mathbb{E}\left(D\left(\boldsymbol{\beta}^{l+1}\right)-\mathbb{E}\left(D\left(\boldsymbol{\beta}^l\right)\right)\right)\right)$$
$$\leq\eta\left(D^*-\mathbb{E}\left(D\left(\boldsymbol{\beta}^l\right)\right)\right)$$

∎

A.3 Proof of Θ-approximate solution of Lemma 3

Proof Let us assume that $\beta_{[k]}^*$ is the optimal solution of the subproblem 4 denotes as:

$$\beta_{[k]}^*=\arg\max_{\beta_{[k]}\in\mathbb{R}^{n_k}}D(\beta_{[k]};\bar{\mathbf{w}}). \tag{32}$$

According to above proof of Lemma 3, the local atomic solver has a linear convergence rate in expectation, that is,

$$D(\beta_{[k]}^*;\bar{\mathbf{w}})-\mathbb{E}\left(D(\beta_{[k]}^{j+1};\bar{\mathbf{w}})\right)\leq\eta\left(\mathbb{E}\left(D(\beta_{[k]}^*;\bar{\mathbf{w}})-D(\beta_{[k]}^j;\bar{\mathbf{w}})\right)\right)$$

It is obvious that $\Theta=\eta^H$. Thus, we can easily get the induction as

$$D(\beta_{[k]}^*;\bar{\mathbf{w}})-\mathbb{E}\left(D(\beta_{[k]}^H;\bar{\mathbf{w}})\right)\leq\eta\left(D(\beta_{[k]}^*;\bar{\mathbf{w}})-\mathbb{E}\left(D(\beta_{[k]}^{H-1};\bar{\mathbf{w}})\right)\right)$$
$$\leq\eta^2\left(\mathbb{E}\left(D(\beta_{[k]}^*;\bar{\mathbf{w}})-D(\beta_{[k]}^{H-2};\bar{\mathbf{w}})\right)\right)\leq\cdots\leq\Theta\left(D(\beta_{[k]}^*;\bar{\mathbf{w}})-\mathbb{E}\left(D(\beta_{[k]}^0;\bar{\mathbf{w}})\right)\right).$$

Notice that $\beta_{[k]}^0$ are the start points of the local atomic solver and $\beta_{[k]}^H$ are the final results of $\beta_{[k]}$ of the local atomic solver. So the following equations hold for the global problem:

$$\beta_{[k]}^0=\alpha_{[k]}$$

$$\beta_{[k]}^H - \beta_{[k]}^0 = \boldsymbol{\delta}_{[k]}$$
$$\beta_{[k]}^* - \beta_{[k]}^0 = \boldsymbol{\delta}_{[k]}^*$$

Therefore, we have:

$$\mathbb{E}\left[Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \mathbf{v}, \boldsymbol{\alpha}_{[k]}) - Q_k^\sigma(\boldsymbol{\delta}_{[k]}; \mathbf{v}, \boldsymbol{\alpha}_{[k]})\right] \le \Theta\left[Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \mathbf{v}, \boldsymbol{\alpha}_{[k]}) - Q_k^\sigma(\mathbf{0}; \mathbf{v}, \boldsymbol{\alpha}_{[k]})\right]$$

with $\Theta = \eta^H$. ∎

Appendix B. Proof of convergence of global solution

B.1 Proof of Lemma 4

Proof Assume that $\mathcal{I} = \bigcup_{k \in \mathcal{P}_S} \mathcal{I}_k$. Then, we have

$$
\begin{aligned}
D\left(\boldsymbol{\alpha} + \nu \sum_{k \in \mathcal{P}_S} \boldsymbol{\delta}_{[k]}\right) &= -\frac{1}{n}\sum_{i=1}^n \phi_i^*\left(-\alpha_i - \nu\left(\sum_{k \in \mathcal{P}_S} \boldsymbol{\delta}_{[k]}\right)_i\right) \\
&\quad - \frac{\lambda}{2}\left\|\frac{1}{\lambda n}\mathbf{X}\left(\boldsymbol{\alpha} + \nu \sum_{k \in \mathcal{P}_S} \boldsymbol{\delta}_{[k]}\right)\right\|^2 \\
&= -\frac{1}{n}\sum_{i \notin \mathcal{I}} \phi_i^*(-\alpha_i) - \frac{1}{n}\sum_{k \in \mathcal{P}_S}\left(\sum_{i \in \mathcal{I}_k} \phi_i^*(-(1-\nu)\alpha_i - \nu(\boldsymbol{\alpha} + \boldsymbol{\delta}_{[k]})_i)\right) \\
&\quad - \frac{\lambda}{2}\left(\|\mathbf{w}(\boldsymbol{\alpha})\|^2 + \frac{2\nu}{\lambda n}\sum_{k \in \mathcal{P}_S} \mathbf{w}(\boldsymbol{\alpha})^\top \mathbf{X}\boldsymbol{\delta}_{[k]} + \left(\frac{\nu}{\lambda n}\right)^2\left\|\sum_{k \in \mathcal{P}_S} \mathbf{X}\boldsymbol{\delta}_{[k]}\right\|^2\right) \\
&\ge -\frac{1}{n}\sum_{k \in \mathcal{P}_S}\left(\sum_{i \in \mathcal{I}_k}((1-\nu)\phi_i^*(-\alpha_i) + \nu\phi_i^*(-(\boldsymbol{\alpha} + \boldsymbol{\delta}_{[k]})_i))\right) \\
&\quad - \frac{\lambda}{2}\left(\|\hat{\mathbf{w}}\|^2 + \frac{2\nu}{\lambda n}\sum_{k \in \mathcal{P}_S} \hat{\mathbf{w}}^\top \mathbf{X}\boldsymbol{\delta}_{[k]} + \left(\frac{\nu}{\lambda n}\right)^2\left\|\sum_{k \in \mathcal{P}_S} \mathbf{X}\boldsymbol{\delta}_{[k]}\right\|^2\right) \\
&\quad - \frac{\nu}{n}\sum_{k \in \mathcal{P}_S}(\mathbf{w} - \hat{\mathbf{w}})^\top \mathbf{X}\boldsymbol{\delta}_{[k]} - \frac{\lambda}{2}(\|\mathbf{w}\|^2 - \|\hat{\mathbf{w}}\|^2) - \frac{1}{n}\sum_{i \notin \mathcal{I}} \phi_i^*(-\alpha_i) \\
&= \underbrace{-\frac{1}{n}\sum_{k \in \mathcal{P}_S}\left(\sum_{i \in \mathcal{I}_k}(1-\nu)\phi_i^*(-\alpha_i)\right) - (1-\nu)\frac{\lambda}{2}\|\mathbf{w}(\hat{\boldsymbol{\alpha}})\|^2}_{(1-\nu)D(\hat{\boldsymbol{\alpha}})} \\
&\quad + \nu \sum_{k \in \mathcal{P}_k}\left(-\frac{1}{n}\sum_{i \in \mathcal{I}_k} \phi_i^*(-(\boldsymbol{\alpha} + \boldsymbol{\delta}_{[k]})_i) - \frac{1}{K}\frac{\lambda}{2}\|\mathbf{w}(\hat{\boldsymbol{\alpha}})\|^2\right.
\end{aligned}
$$

$$-\frac{1}{n}\mathbf{w}(\hat{\boldsymbol{\alpha}})^{\top}\mathbf{X}\boldsymbol{\delta}_{[k]} - \frac{\lambda}{2}\sigma\left\|\frac{1}{\lambda n}\mathbf{X}\boldsymbol{\delta}_{[k]}\right\|^2\right)$$

$$-\frac{\nu}{n}\sum_{k\in\mathcal{P}_S}(\mathbf{w}-\hat{\mathbf{w}})^{\top}\mathbf{X}\boldsymbol{\delta}_{[k]} - \frac{\lambda}{2}(\|\mathbf{w}\|^2 - \|\hat{\mathbf{w}}\|^2) - \frac{1}{n}\sum_{i\notin\mathcal{I}}\phi_i^*(-\alpha_i)$$

$$=(1-\nu)D(\hat{\boldsymbol{\alpha}}) + \nu\sum_{k\in\mathcal{P}_S}Q_k^\sigma(\boldsymbol{\delta}_{[k]};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})$$

$$-\frac{\nu}{n}\sum_{k\in\mathcal{P}_S}(\mathbf{w}-\hat{\mathbf{w}})^{\top}\mathbf{X}\boldsymbol{\delta}_{[k]} - \frac{\lambda}{2}(\|\mathbf{w}\|^2 - \|\hat{\mathbf{w}}\|^2) - \frac{1}{n}\sum_{i\notin\mathcal{I}}\phi_i^*(-\alpha_i).$$

∎

B.2 Proof of Lemma 5

Proof For sake of notation, we will write $\boldsymbol{\alpha}$ instead of $\boldsymbol{\alpha}^t$, $\mathbf{w}$ instead of $\mathbf{w}(\boldsymbol{\alpha}^t)$, $\hat{\mathbf{w}}$ instead of $\mathbf{w}(\hat{\boldsymbol{\alpha}})$, and $\boldsymbol{\delta}$ instead of $\boldsymbol{\delta}^t$.

Now, the expected change of the dual objective is

$$\mathbb{E}[D(\boldsymbol{\alpha}^t) - D(\boldsymbol{\alpha}^{(t+1)})] = \mathbb{E}[D(\boldsymbol{\alpha}^t) - D(\hat{\boldsymbol{\alpha}}) + D(\hat{\boldsymbol{\alpha}}) - D(\boldsymbol{\alpha}^{(t+1)})]$$
$$= \mathbb{E}[D(\boldsymbol{\alpha}^t) - D(\hat{\boldsymbol{\alpha}})] + \mathbb{E}[D(\hat{\boldsymbol{\alpha}}) - D(\boldsymbol{\alpha}^{(t+1)})]$$

Thus, it is a summation of two parts. Let us estimate both parts as following,

$$\mathbb{E}[D(\boldsymbol{\alpha}^t) - D(\hat{\boldsymbol{\alpha}})] = \mathbb{E}\left[-\frac{1}{n}\sum_{i\notin\mathcal{I}}\phi_i^*(-\alpha_i) - \frac{1}{n}\sum_{k\in\mathcal{P}_S}\left(\sum_{i\in\mathcal{I}_k}\phi_i^*(-\alpha_i)\right) - \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{k\in\mathcal{P}_S}\left(\sum_{i\in\mathcal{I}_k}\phi_i^*(-\alpha_i)\right) + \frac{\lambda}{2}\|\hat{\mathbf{w}}\right.$$
$$= \mathbb{E}\left[-\frac{\lambda}{2}\left(\|\mathbf{w}\|^2 - \|\hat{\mathbf{w}}\|^2\right) - \frac{1}{n}\sum_{i\notin\mathcal{I}}\phi_i^*(-\alpha_i)\right]$$

$$\mathbb{E}[D(\hat{\boldsymbol{\alpha}}) - D(\boldsymbol{\alpha}^{(t+1)})] = \mathbb{E}\left[D(\hat{\boldsymbol{\alpha}}) - D\left(\boldsymbol{\alpha} + \nu\sum_{k\in\mathcal{P}_S}\boldsymbol{\delta}_{[k]}\right)\right]$$
$$\leq \mathbb{E}\left[D(\hat{\boldsymbol{\alpha}}) - (1-\nu)D(\hat{\boldsymbol{\alpha}}) - \nu\sum_{k\in\mathcal{P}_S}Q_k^\sigma(\boldsymbol{\delta}_{[k]};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right]$$
$$+ \mathbb{E}\left[\frac{\nu}{n}\sum_{k\in\mathcal{P}_S}(\mathbf{w}-\hat{\mathbf{w}})^{\top}\mathbf{X}\boldsymbol{\delta}_{[k]} + \frac{\lambda}{2}(\|\mathbf{w}\|^2 - \|\hat{\mathbf{w}}\|^2) + \frac{1}{n}\sum_{i\notin\mathcal{I}}\phi_i^*(-\alpha_i)\right] \quad \text{(Lemma 4)}$$

Therefore,

$$\mathbb{E}[D(\boldsymbol{\alpha}^t) - D(\boldsymbol{\alpha}^{(t+1)})] \leq \mathbb{E}\left[-\frac{\lambda}{2}\left(\|\mathbf{w}\|^2 - \|\hat{\mathbf{w}}\|^2\right) - \frac{1}{n}\sum_{i\notin\mathcal{I}}\phi_i^*(-\alpha_i)\right]$$

26

$$+ \mathbb{E}\left[D(\hat{\boldsymbol{\alpha}}) - (1 - \nu)D(\hat{\boldsymbol{\alpha}}) - \nu \sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}})\right]$$

$$+ \mathbb{E}\left[\frac{\nu}{n} \sum_{k \in \mathcal{P}_S} (\mathbf{w} - \hat{\mathbf{w}})^\top \mathbf{X} \boldsymbol{\delta}_{[k]} + \frac{\lambda}{2}(\|\mathbf{w}\|^2 - \|\hat{\mathbf{w}}\|^2) + \frac{1}{n} \sum_{i \notin \mathcal{I}} \phi_i^*(-\alpha_i)\right]$$

$$= \nu \mathbb{E}\left[D(\hat{\boldsymbol{\alpha}}) - \sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}})\right.$$

$$+ \sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}}) - \sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}})\Bigg]$$

$$+ \mathbb{E}\left[\frac{\nu}{n} \sum_{k \in \mathcal{P}_S} (\mathbf{w} - \hat{\mathbf{w}})^\top \mathbf{X} \boldsymbol{\delta}_{[k]}\right]$$

$$\leq \nu \left(D(\hat{\boldsymbol{\alpha}}) - \sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}})\right.$$

$$+ \Theta \left(\sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}}) - \underbrace{\sum_{k \in \mathcal{P}_S} Q_k^\sigma(\mathbf{0}; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}})}_{D(\hat{\boldsymbol{\alpha}})}\right)\right) \quad \text{(Lemma ??)}$$

$$+ \mathbb{E}\left[\frac{\nu}{n} \sum_{k \in \mathcal{P}_S} (\mathbf{w} - \hat{\mathbf{w}})^\top \mathbf{X} \boldsymbol{\delta}_{[k]}\right]$$

$$= \nu(1 - \Theta)\left(D(\hat{\boldsymbol{\alpha}}) - \sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}})\right) + \mathbb{E}\left[\frac{\nu}{n} \sum_{k \in \mathcal{P}_S} (\mathbf{w} - \hat{\mathbf{w}})^\top \mathbf{X} \boldsymbol{\delta}_{[k]}\right]$$

$$\leq \nu(1 - \Theta)\left(D(\hat{\boldsymbol{\alpha}}) - \sum_{k \in \mathcal{P}_S} Q_k^\sigma(\boldsymbol{\delta}_{[k]}^*; \boldsymbol{\alpha}_{[k]}, \hat{\mathbf{w}})\right)$$

$$+ \frac{\nu}{2n}\underbrace{\left(\mathbb{E}\left[\|\mathbf{w} - \hat{\mathbf{w}}\|^2\right] + \mathbb{E}\left[\left\|\sum_{k \in \mathcal{P}_S} \mathbf{X} \boldsymbol{\delta}_{[k]}\right\|^2\right]\right)}_{A}$$

Note that $\mathbf{w} = \hat{\mathbf{w}} + \frac{1}{\lambda n} \sum_{j=t-\Gamma}^{t-1} \mathbf{X} \boldsymbol{\delta}^j$. Now, let us bound the term A. We have

$$A = \mathbb{E}\left[\frac{1}{\lambda n}\left\|\sum_{j=t-\Gamma}^{t-1} \mathbf{X} \boldsymbol{\delta}^j\right\|^2\right] + \mathbb{E}\left[\left\|\sum_{k \in \mathcal{P}_S} \mathbf{X} \boldsymbol{\delta}_{[k]}\right\|^2\right]$$

$$\leq \mathbb{E}\left[\frac{\Gamma}{\lambda n} \sum_{j=t-\Gamma}^{t-1} \|\mathbf{X} \boldsymbol{\delta}^j\|^2\right] + \mathbb{E}\left[S \sum_{k \in \mathcal{P}_S} \|\mathbf{X} \boldsymbol{\delta}_{[k]}\|^2\right] \quad \text{(Cauchy Schwarz Inequality)}$$

$$\leq \mathbb{E}\left[\frac{\Gamma M}{\lambda n}\sum_{j=t-\Gamma}^{t-1}\left\|\boldsymbol{\delta}^{j}\right\|^{2}\right] + \mathbb{E}\left[SM\sum_{k\in\mathcal{P}_{S}}\left\|\boldsymbol{\delta}_{[k]}\right\|^{2}\right] \quad \text{(Proposition 11)}$$

$$\leq \mathbb{E}\left[\frac{\Gamma M}{\lambda n}\left(\sum_{j=t-\Gamma}^{t-1}\varrho^{j}\right)\left\|\boldsymbol{\delta}^{t-1}\right\|^{2}\right] + \mathbb{E}\left[SM\sum_{k\in\mathcal{P}_{S}}\left\|\boldsymbol{\delta}_{[k]}\right\|^{2}\right] \quad \text{(By (12))}$$

$$\leq \frac{\Gamma M L_{max}}{2\lambda n}\sum_{j=t-\Gamma}^{t-1}\varrho^{j}\left(D(\hat{\boldsymbol{\alpha}}) - \sum_{k\in\mathcal{P}_{S}}Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right)$$

$$+ \frac{SM L_{max}}{2}\sum_{k\in\mathcal{P}_{S}}\left(D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right) \quad \text{(Proposition 14)}$$

Here $D(\hat{\boldsymbol{\alpha}}) = Q_{k}^{\sigma}(\mathbf{0};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})$. Thus, Eq. 7 can be rewritten as,

$$\mathbb{E}\left[Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right] \leq \Theta\left(Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}}) - D(\hat{\boldsymbol{\alpha}})\right) + D(\hat{\boldsymbol{\alpha}}) - D(\hat{\boldsymbol{\alpha}})$$

$$D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}}) \leq (1-\Theta)D(\hat{\boldsymbol{\alpha}}) - (1-\Theta)Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})$$

$$D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}}) \leq -(1-\Theta)\left(Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}}) - D(\hat{\boldsymbol{\alpha}})\right) \quad (33)$$

Then, A can be bounded as,

$$A \leq -\frac{\Gamma^{2}M L_{max}}{2\lambda n}\varrho^{\Gamma}(1-\Theta)\sum_{k\in\mathcal{P}_{S}}\left(D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right)$$

$$- \frac{SM L_{max}}{2}(1-\Theta)\sum_{k\in\mathcal{P}_{S}}\left(D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right) \quad \text{(By (33))}$$

$$\leq -\frac{\Gamma^{2}e M L_{max}}{2\lambda n}(1-\Theta)\sum_{k\in\mathcal{P}_{S}}\left(D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right)$$

$$- \frac{SM L_{max}}{2}(1-\Theta)\sum_{k\in\mathcal{P}_{S}}\left(D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*};\boldsymbol{\alpha}_{[k]},\hat{\mathbf{w}})\right) \quad \text{(Assumption (5))}$$

By substituting A, we have

$$\mathbb{E}[D(\boldsymbol{\alpha}^{t}) - D(\boldsymbol{\alpha}^{(t+1)})] \leq \nu\left(1 - \frac{\Gamma^{2}e M L_{max}}{4\lambda n^{2}} - \frac{SM L_{max}}{4n}\right)(1-\Theta)\sum_{k\in\mathcal{P}_{S}}\left(D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*})\right)$$

$$\mathbb{E}[D(\boldsymbol{\alpha}^{t}) - D(\boldsymbol{\alpha}^{(t+1)})] \leq \Psi(1-\Theta)\sum_{k\in\mathcal{P}_{S}}\left(D(\hat{\boldsymbol{\alpha}}) - Q_{k}^{\sigma}(\boldsymbol{\delta}_{[k]}^{*})\right)$$

Using the Eq. C in the proof of Lemma 5 in Ma et al. (2015b), we can show that

$$\mathbb{E}[D(\boldsymbol{\alpha}^{t}) - D(\boldsymbol{\alpha}^{(t+1)})] \leq \Psi(1-\Theta)\left(-sG(\hat{\boldsymbol{\alpha}}) - \frac{1}{2\mu}(1-s)s\frac{1}{n}\|\hat{\mathbf{u}} - \hat{\boldsymbol{\alpha}}\|^{2} + \frac{\sigma}{2\lambda}\left(\frac{s}{n}\right)^{2}\sum_{k\in\mathcal{P}_{S}}\left\|\mathbf{X}(\hat{\mathbf{u}} - \hat{\boldsymbol{\alpha}})_{[k]}\right\|\right)$$

$$= \Psi(1 - \Theta) \left(-sG(\hat{\boldsymbol{\alpha}}) + \frac{\sigma}{2\lambda} \left(\frac{s}{n} \right)^2 \hat{R} \right)$$

$$\blacksquare$$