2022 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)

Distributed ADMM Based on Sparse Computation and Allreduce Communication

1st Zeyu Zhang
School of Computer Engineering and Science
Shanghai University
Shanghai, China
zeyu_zhang@shu.edu.cn

2nd Yongmei Lei
School of Computer Engineering and Science
Shanghai University
Shanghai, China
lei@shu.edu.cn

3nd Dongxia Wang
School of Information Science and Engineering
Hunan Institute of Science and Technology
Yueyang, China
wangdongxia1983@126.com

4nd Guozheng Wang
School of Computer Engineering and Science
Shanghai University
Shanghai, China
gzh.wang@outlook.com

***Abstract*—The distributed alternating direction method of multipliers (ADMM) is an effective algorithm to solve large-scale optimization problems. However, there are still massive computation and communication cost in distributed ADMM when processing high-dimensional data. To solve this problem, we propose a distributed ADMM with sparse computation and Allreduce communication (SCAC-ADMM) which can process high-dimensional data effectively. In the algorithm, each node optimizes a sub-model of the target model in parallel. Then, the target model is obtained by aggregating all sub-models. The features in the sub-model are named associated features. In SCAC-ADMM, we first design a selecting method of associated features to determine the composition of each sub-model. This method can limit the dimension of the sub-model by setting appropriate parameters, so as to limit the computation cost. Secondly, to reduce the communication traffic caused by transmitting high-dimensional parameters, we propose a novel Allreduce communication model which can only aggregate associated parameters in sub-models. Experiments on high-dimensional datasets show that SCAC-ADMM has less computation cost and higher communication efficiency than traditional distributed ADMM. When solving large-scale logistic regression problem, SCAC-ADMM can reduce the system time by 73% compared with traditional distributed ADMM.**

***Index Terms*—distributed ADMM algorithm, sparse computation, allreduce communication, SCAC-ADMM**

I. Introduction

The distributed alternating direction method of multipliers (ADMM) [1], as an effective distributed optimization algorithm, is widely used to solve large-scale machine learning problems, such as Linear regression [2], Support vector machine [3], and many others. In the distributed ADMM algorithm, the original problem is decomposed into several sub-problems which can be solved in parallel, and then the solution of the original problem is obtained by coordinating the solutions of the sub-problems. In the scenario of data parallelism [4], the distributed ADMM algorithm aims to solve the following distributed optimization problem:

$$\min_{x} \sum_{i=1}^{P} f_i(x). \tag{1}$$

where the objective function is divided into P parts, $f_i : R^d \to R$ is local objective function, $x \in R^d$ represents the global variable to be optimized, d is the number of features.

With the rapid growth of data dimensions, the distributed ADMM algorithm encounters the following problems when dealing with high-dimensional data: (1) When optimizing sub-problems, other optimization algorithms is needed, such as L-BFGS [5], trust region Newton method [6]. In these iterative optimization processes, processing high-dimensional training data will cause massive computational overhead. (2) In distributed ADMM, communication between nodes is needed to aggregate the parameters at each iteration. Transmitting high-dimensional model will result in large amount of traffic in network. These two problems affect the efficiency of distributed ADMM. Therefore, in this paper, we try to reduce the computation and communication overhead of distributed ADMM for processing high-dimensional data.

At present, many studies have been carried out to improve the efficiency of distributed ADMM algorithms from the aspects of computation and communication. For computational optimization, linearized ADMM [7] and stochastic ADMM [8] reduced the computation overhead of a single iteration by reducing the computational complexity of solving the sub-problems. The convergence speed of the ADMM algorithm is affected by the penalty parameter, so the author of [9] designed a distributed ADMM algorithm with dynamic penalty parameter to accelerate the convergence speed of algorithm. In terms of communication optimization, the algorithms proposed in [10] and [11] reduced the communication cost based on quantized communication and communication censoring. In [12], an asynchronous distributed ADMM was proposed to reduce the synchronous waiting time.

However, the above studies improve computation and communication separately, and the optimization and the transmission of model are based on full model. To improve the efficiency of solving high-dimensional problems, the optimization on both computation and communication need to be considered. For high-dimensional problems, the author of [1] proposed a general form consensus optimization. In this optimization, each process only optimizes a sub-model in parallel, and the target model is obtained by aggregating these sub-models. The features in sub-model are named associated features, which are associated with some features in target model. Under this mode, each node only needs to optimize a smaller sub-model instead of full model, so the computation cost can be reduced. And aggregating only sub-models can also reduce the communication cost. Therefore, the general form consensus optimization is more suitable for processing high-dimensional data.

In this paper, in order to improve the optimization speed of distributed ADMM for processing high-dimensional data, based on general form consensus optimization, we propose a distributed ADMM based on sparse computation and Allreduce communication (SCAC-ADMM). In SCAC-ADMM, only associated features are optimized, which are determined according to selecting method. Moreover, a novel Allreduce communication model is designed to aggregate these associated features effectively. The contributions of this paper are as follows:

- We design a selecting method of associated features, which determines the composition of sub-models. This method is based on the number of non-zero elements in each dimension of dataset. By setting the hyperparameter in this method, the dimension of the sub-model can be limited, so as to limit the computation cost.
- To reduce communication traffic when transmitting high-dimensional data, an novel Allreduce communication model is proposed in this paper, which can only transmit the data on associated features instead of the full model.
- We evaluate the performance of SCAC-ADMM with high-dimensional datasets on the Tianhe-2 supercomputing platform. The experiments show that SCAC-ADMM has less computation and communication cost than traditional distributed ADMM when processing high-dimensional data.

The rest of this paper is organized as follows. Section II introduces the relevant background. In Section III, we describe the design of SCAC-ADMM. Experiments results are explained in Section IV. Section V concludes this paper and gives the future work.

II. Background

A. Global Consensus ADMM Algorithm

In [1], the distributed optimization problem in (1) was converted into a global consensus optimization problem as follow:

$$\min_{x_i} \sum_{i=1}^{P} f_i(x_i) + g(z) \text{ s.t. } x_i = z, i = 1, \ldots, P. \tag{2}$$

where $x_i \in R^d$ represents the local variable to be optimized, $z \in R^d$ represents the global variable, $g : R^d \to R \cup \{\infty\}$ is the regularization function. The equality constraint is used to control the difference of x_i through the global variable z, so that they can close to each other and finally achieve the global consensus.

Global consensus ADMM (GC-ADMM) starts from constructing the augmented Lagrangian L_ρ of (2). Then, the L_ρ is minimized by updating x_i and z alternately. The expression of L_ρ is shown in (3). And the iterative formulas of GC-ADMM are shown in (4)-(6).

$$L_\rho(\{x_i\}, z, \{y_i\}) = \sum_{i=1}^{P} \left(f_i(x_i) + \frac{\rho}{2} \left\| x_i + \frac{y_i}{\rho} - z \right\|_2^2 \right) + g(z). \tag{3}$$

$$x_i^{k+1} = \underset{x_i}{\operatorname{argmin}} \left(f_i(x_i) + \frac{\rho}{2} \left\| x_i + \frac{y_i^k}{\rho} - z^k \right\|_2^2 \right). \tag{4}$$

$$z^{k+1} = \underset{z}{\operatorname{argmin}} \left(g(z) + \frac{\rho}{2} \sum_{i=1}^{P} \left\| x_i^{k+1} + \frac{y_i^k}{\rho} - z \right\|_2^2 \right). \tag{5}$$

$$y_i^{k+1} = y_i^k + \rho \left(x_i^{k+1} - z^{k+1} \right). \tag{6}$$

where $y \in R^d$ represents dual variable, ρ represents the penalty parameter. The execution of GC-ADMM consists of following steps: first, local variable x_i is updated in each node in parallel, than x_i and dual variables y_i are aggregated to calculate global variable z, finally each node updates the y_i in parallel. The update of variables is performed iteratively until the stopping criterion is satisfied.

B. General Form Consensus Optimization

Since GC-ADMM does not consider the characteristics of the data, when processing high-dimensional sparse data in GC-ADMM, each node needs to optimize and transmit the whole high-dimensional model in each iteration, resulting in massive computation and communication overhead. In this case, the GC-ADMM cannot efficiently optimize the model with high-dimensional data. Therefore, the author of [1] convert problem (1) to general form consensus optimization shown in (7).

$$\min_{x_i} \sum_{i=1}^{P} f_i(x_i) + g(z) \text{ s.t. } x_i = z_{R_i}, i = 1, \ldots, P. \tag{7}$$

where $x_i \in R^{d_i}$ is local variable, and the features in x_i are named associated features, which are associated to some features in z. $z_{R_i} \in R^{d_i}$ is the corresponding part of the x_i in the z. The relationship of local variable and global variable is shown in Fig. 1.

In general form consensus optimization, only associated features need to be optimized in each node. But how to determine the associated features in each node is a problem. To further reduce the computation cost, we design a selecting method of associated features based on the number of non-zero elements in each dimension of dataset, which is introduced in Section III-A.

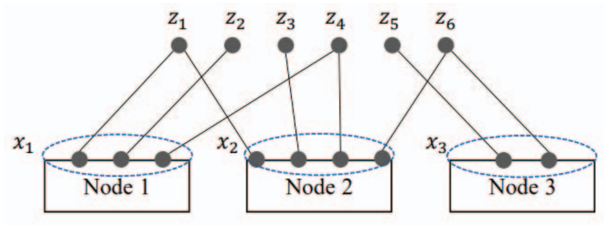


Fig. 1. The relationship between local variable and global variable. Edges represent the association of them.

C. Communication model in Distributed ADMM

In fully connected communication network, the distributed ADMM is mainly implemented based on two network topologies, the parameter server [13] and Allreduce model. The server nodes and worker nodes in parameter server make communication more flexible, but the server nodes are dedicated to aggregate parameters, which leads to a waste of computing resource. Besides, when aggregating high-dimensional parameters, saving these parameters will also result in huge memory overload in server nodes. In Allreduce model, the status of each node is same, and the communication load between nodes is balanced, which can make full use of computing resource and communication bandwidth.

Among the studies about general form consensus ADMM, the author of [14] implemented the algorithm based on master-slave mode in asynchronous manner. Although asynchronous communication can reduce the waiting time of master node, a single master node in network will limit the scalability of algorithm. In [15], general form consensus ADMM is implemented on parameter server with multiple server nodes, but it has the disadvantages described above.

To obtain balanced communication, we design a novel Allreduce model based on the characteristic that only associated features are optimized in general form consensus optimization. This Allreduce model is introduced in Section III-C.

III. Proposed Algorithm: SCAC-ADMM

In SCAC-ADMM, each node first fetches the composition of its sub-model according to the selecting method of associated features. Then, based on these features, the optimization is performed according to the update rules we give. And the communication in SCAC-ADMM based on the proposed Allreduce model.

A. Selecting Method of Associated Features

The set of associated features in the i-th node is defined as $\theta(i)$. The more features in $\theta(i)$ means the more computation cost, but losing too many features will affect the final accuracy. In the selecting method, $\theta(i)$ is determined by pre-analyzing the i-th dataset. We consider that if a dimension is non-zero in large number of training data, then this dimension is important, while it can be ignored if it is non-zero in only several pieces of data. Based on this idea, we proposed the selecting method. By analysing the local dataset, nodes can get the importance of each dimension of local dataset. And only the features whose importance is greater than threshold will be add to feature set. The process of selecting associated feature is shown in Fig. 2.

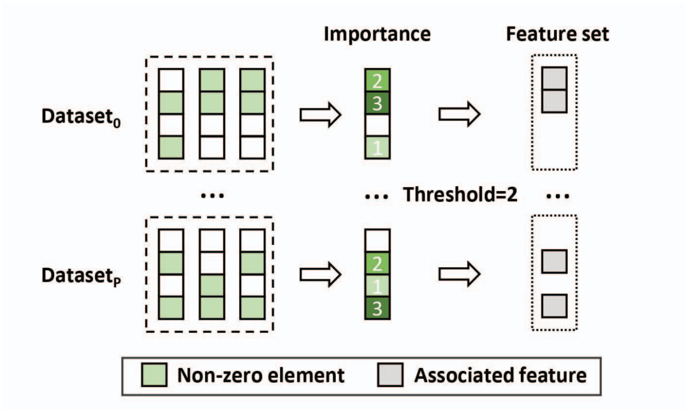


Fig. 2. The process of selecting associated features.

In this method, a matrix $M \in R^{P\times d}$ is defined to build the relationship between local variable (sub-model) and global variable (full model), and each node only stores a row of M. Assuming that D_i is the training dataset assigned to the i-th node, M_{ij} denotes the number of non-zero elements on the j-th dimension in D_i. Each node counts the number of non-zero elements in each dimension of local dataset and saves them in M. The filter threshold τ is defined to select associated features. If $M_{ij} >= \tau$, it indicates that j-th feature is important, and it will be selected as an associated feature in node i. At the same time, j will be added to $\theta(i)$, i will be added to $\phi(j)$ which is a set including the nodes associated with j-th dimension in z. The selecting method is summarized in Algorithm 1.

Algorithm 1 Selecting method of associated features

```
1:  for all nodes [in parallel] do
2:      Initialize $M_i = 0, \tau$
3:      for $data \in D_i$ do
4:          if $data_j ! = 0$ then
5:              $M_{ij} = M_{ij} + 1$
6:          end if
7:      end for
8:      if $M_{ij} >= \tau$ then
9:          add $j$ to $\theta(i)$
10:         add $i$ to $\phi(j)$
11:     end if
12: end for
```

The number of associated features in local variable is limited by τ, which is set artificially. When the global variable is high-dimensional, by setting appropriate filter threshold τ, we can limit the dimension of local variable, so as to limit the computation cost.

B. Update Rules

After determining the composition of each local variable, the optimization procedure begins. In SCAC-ADMM, the optimization consists of the update of local variable x_i, global variable z and dual variable y_i. Since the different dimensions of x_i and y_i in each node, it is hard to aggregate these parameters based on Allreduce model. Therefore, we expand the dimensions of x_i and y_i to the same as z. But each node only updates the associated features in x_i and y_i, and the calculation is performed in a sparse manner.Therefore, it is still a general form consensus optimization. We name this approach sparse computation.

According to definition of $\theta(i)$, z_{R_i} in (7) can be further defined as $z_{R_i} := \{z_j | j \in \theta(i)\}$. Like the derivation in Section II-A, we can get the iterative update rules of SCAC-ADMM by solving (7). And the rules are shown in (8) - (10).

$$x_i^{k+1} = \underset{x_i}{\operatorname{argmin}}(f_i(x_i) + \frac{\rho}{2} \sum_{j \in \theta(i)} \left\| x_{ij} + \frac{y_{ij}^k}{\rho} - z_j^k \right\|^2). \quad (8)$$

$$z^{k+1} = \underset{z}{\operatorname{argmin}}(g(z) + \sum_{j=1}^{d} \sum_{i \in \phi(j)} \frac{\rho}{2} \left\| x_{ij}^{k+1} + \frac{y_{ij}^k}{\rho} - z_j \right\|^2). \quad (9)$$

$$y_{ij}^{k+1} = y_{ij}^k + \rho\left(x_{ij}^{k+1} - z_j^{k+1}\right), \forall j \in \theta(i). \quad (10)$$

where the dimension of these variables is d, and the calculation in $f_i(x_i)$ is performed in a sparse manner. Note that the update of the global variable needs to calculate $x_{ij}^{k+1} + \frac{y_{ij}^k}{\rho}$, so we define it as w_{ij}^{k+1} and transmit it as a whole to reduce communication cost. The update rule of z is converted to (11).

$$z^{k+1} = \underset{z}{\operatorname{argmin}}(g(z) + \sum_{j=1}^{d} \sum_{i \in \Phi(j)} \frac{\rho}{2} \left\| w_{ij}^{k+1} - z_j \right\|^2). \quad (11)$$

The update steps of variables are the same as described in the GC-ADMM. In the next section, we introduce the proposed Allreduce model, which is used to aggregate x_i in each node when updating z.

C. Allreduce Communication in SCAC-ADMM

Among the distributed ADMM implemented based on the Allreduce model, most of them are implemented based on the Ring-Allreduce [16]. In [17] and [18], the Ring-Allreduce is used to implement asynchronous ADMM and decentralized consensus ADMM, respectively.

In Ring-Allreduce, the data is divided into P blocks for transmission, where P is the number of nodes in the network. All the nodes are organized in a logistic ring. Ring-Allreduce consists of two phases, Scatter-Reduce phase and Allgather phase. The former aggregates data blocks to one node, and the latter allows all nodes to own the aggregated data. The communication process of aggregating data blocks is shown in Fig. 3.

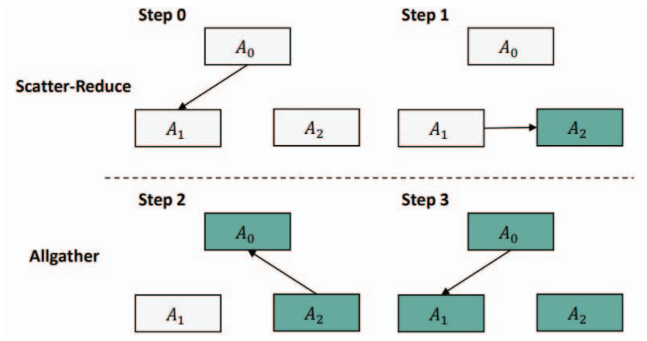


Fig. 3. The schematic diagram of aggregating data blocks based on Ring-Allreduce. Green blocks contained aggregated data. A_i represents the data block A in i-th node. $P = 3$.

To achieve balanced communication, we first consider implementing SCAC-ADMM based on Ring-Allreduce. However, since the communication in Ring-Allreduce is performed in one direction, it cannot flexibly transfer associated data to different nodes. If SCAC-ADMM is implemented based on Ring-Allreduce, each node will obtain z after each iteration. But according to (8) and (10), only the associated part in z is needed for updating. Transmitting whole z will incur additional communication overhead. To avoid this additional transmission, we design a new Allreduce model based on Ring-Allreduce.

1) Communication design of Allreduce model: The Allreduce operation we proposed also consists of the Scatter-reduce phase and the Allgather phase, each of which involves a number of communications. And the parameters are divided into P parts for transmission.

In the Scatter-reduce phase, parameters are transmitted in the same direction like Ring-Allreduce. In each communication, each node receives data block from its left neighboring node and sends data block to right neighboring node. After $P - 1$ communications, each node will get a part of the final result, and we name this part of data the complete block. At this moment, there is one complete block and $P - 1$ non-complete blocks in each node. The schematic diagram of Scatter-reduce phase is shown in Fig. 4. In fact, the update of z is finished after Scatter-reduce phase, and z consists of these complete blocks distributed in each node.

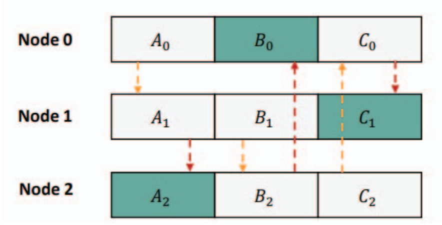


Fig. 4. The schematic diagram of Scatter-reduce phase in our Allreduce model. The communications in same color are performed simultaneously. Green data block represents complete block. $P = 3$. The sequence of communication: yellow, red.

In the Allgather phase, we have to let each node obtain its' associated part in z. Therefore, each node is responsible for managing its complete block and sending other nodes' associated parts in this complete block to these nodes. At the same time, each node receives its associated part in other complete blocks. After $P-1$ communications, each node gets its own associated parameters. The schematic diagram of Allgather phase is shown in Fig. 5. After Allgather phase, each node will obtain the associated part in z, which is z_{R_i}.

Since each node needs to manage its complete block, extra communication is necessary before the first Allreduce communication to obtain the indices of other nodes' associated parts in its complete block.

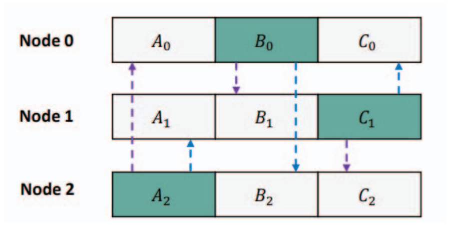


Fig. 5. The schematic diagram of Allgather phase in our Allreduce model. The communications in same color are performed simultaneously. Green data block represents complete block. $P=3$. The sequence of communication: blue, purple.

2) Support of sparse communication: Because of the selecting method in Section III-A, the local variable become sparse. Therefore, we use sparse communication to further reduce the communication traffic.

In sparse communication, two data formats are defined, the sparse data format and the vector data format. The sparse data format is a sequence of key-value pairs of non-zero elements in the parameters to be transmitted. The vector data format is the original data, represented as a vector. We use m to represent the number of elements in parameters, nnz to represent the number of non-zero elements, and b_k and b_v to represent the number of bytes to store the value and index of parameter, respectively. When $nnz(b_v+b_k) < mb_v$, the sparse data format is used for transmission, otherwise the vector data format is used.

Proposed Allreduce model is based on the above two parts. In the Scatter-reduce phase, all the local variables are sparse, so using sparse communication can reduce the communication traffic when aggregating them. In the Allgather phase, since only the associated feature needs to be transmitted, communication traffic can be further reduced by using sparse data format. Experiment in Section IV-D shows that using the proposed Allreduce model to implement SCAC-ADMM can greatly reduce the communication traffic compared with sparse Ring-Allreduce.

SCAC-ADMM consists of the above procedures. And it is summarized it in Algorithm 2. In the next section, we demonstrate the advantages of SCAC-ADMM through several experiments.

Algorithm 2 SCAC-ADMM

```
1:  for all nodes [in parallel] do
2:      fetch associated features according to Algorithm 1
3:      Initialize $x_i^{(0)} = 0, y_i^{(0)} = 0, z^{(0)} = 0, k = 0$
4:      while $k < max\_iteration$ do
5:          $k = k + 1$
6:          update $x_i^{k+1}$ according to (8)
7:          $w_{ij}^{k+1} = x_{ij}^{k+1} + \frac{y_{ij}^k}{\rho}$
8:          Aggregate $w_{ij}^{k+1}$ using proposed Allreduce model
9:          update $z^{k+1}$ according to (11)
10:         update $y_i^{k+1}$ according to (10)
11:     end while
12: end for
```

IV. Experiment

In this section, we test the convergence, system time, communication traffic of SCAC-ADMM. SCAC-ADMM is compared with the GC-ADMM introduced in Section II-A, which is implemented based on the Ring-Allreduce and with the AD-ADMM [12]. These algorithms are implemented in C++ and the communication is implemented based on MPICH.

The experiments are carried out on the Tianhe-2 supercomputer platform. At most 8 physical nodes are used in the following experiments. Each physical node runs 16 processes, and each process represents a worker. In order to test the ability of SCAC-ADMM for processing high-dimensional data, experiments are conducted on three public high-dimensional datasets, kddb[1], url[2] and avazu[3], respectively. And the training data is partitioned by data parallelism. The specific information of the datasets is shown in Table I.

TABLE I
Summary of Datsets

Dataset	Number of training samples	Number of testing samples	Number of features
kddb	19,264,097	748,401	1,163,024
url	2,000,000	396,130	3,231,961
avazu	12,642,186	1,719,304	1,000,000

A. Parameters setting of experiments

In the experiments, we solve the logistic regression problem as follow

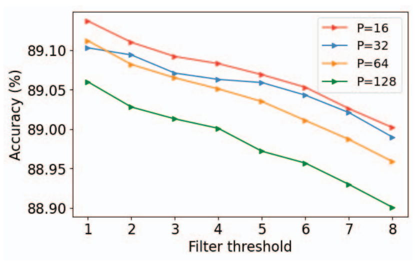
$$\min_x \sum_{i=1}^{P} \log\left(1+\exp\left(-b_i D_i^T x\right)\right) + \lambda\|x\|_1. \tag{12}$$

where $x \in R^d$ is model parameter, D_i is training samples, $b_i \in \{-1, 1\}$ is the label of sample, λ is regularization parameter. L-BFGS [5] algorithm is used to solve sub-problems. The regularization parameter λ is set to 0.5, and the penalty

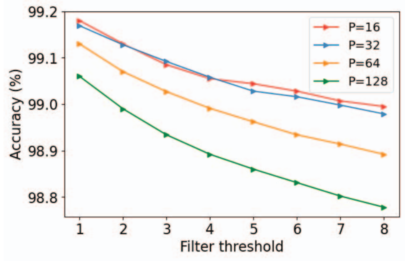
[1]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html#kdd2010 raw version (bridge to algebra).

[2]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html#url

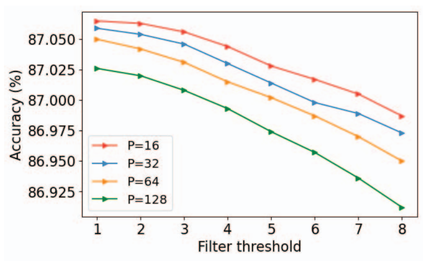
[3]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html#avazu



(a) kddb

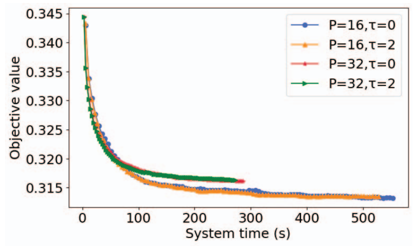


(b) url

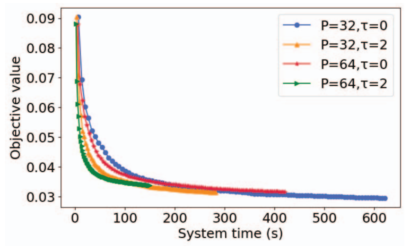


(c) avazu

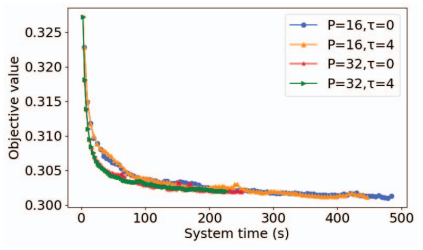
Fig. 6. Accuracy of SCAC-ADMM with different filter thresholds.



(a) kddb



(b) url



(c) avazu

Fig. 7. Convergence of SCAC-ADMM on kddb, url and avazu datasets.

parameter ρ is set to 1. $Min\_barrier$ and max_delay are two important hyperparameters in AD-ADMM. In AD-ADMM, the master worker will not start the next iteration until it receives $min\_barrier$ local variables. And the difference between the number of iteration of the fastest node and the slowest node at cannot exceed max_delay. In the following experiments, $min\_barrier$ is set to the half of the workers used in AD-ADMM, and max_delay is set to 5.

The filter threshold τ is a crucial parameter which affect the accuracy and total running time of the algorithm. The larger the filter threshold is, the more features will be ignored when optimizing model, and this may finally result in a large loss of accuracy. In order to evaluate how filter threshold affect accuracy, we test the filter threshold vs. accuracy of SCAC-ADMM with different filter thresholds. The result is shown in Fig. 6.

From Fig. 6, it can be found that as the filter threshold increases, the accuracy of the algorithm decreases gradually, which is consistent with the previous analysis. Besides, it can be found that the more workers used in SCAC-ADMM, the lower the accuracy is when the filter threshold keeps constant. The reason is that in this case, more features will be filtered according to the selecting method of associated feature. Therefore, in order not to cause too much impact on the accuracy, the filter threshold is set to 2, 2, 4 when testing SCAC-ADMM with kddb, url and avazu dataset, respectively.

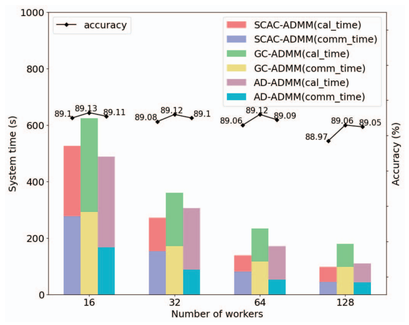
B. Convergence Test

The system time vs. objective value is used to measure the convergence speed of SCAC-ADMM. The $max\_iteration$ is set to 100. When testing with the kddb and avazu dataset, experiments are run on 16 and 32 workers, respectively. When testing with the url dataset, experiments are run on 32 and 64 workers, respectively. To test the impact of filter threshold on convergence, we compare the convergence curves when τ is 0 and the set value. Fig. 7 shows the convergence curves of SCAC-ADMM on three datasets.

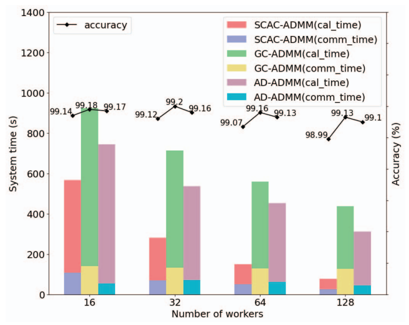
Result of experiment shows that with different parameter settings, the convergence curves tend to be flat, and the algorithm gradually converges. When testing the algorithm with the set filter threshold, the curve almost converge to the same value comparing with the curves when $\tau = 0$, and the system time of SCAC-ADMM is less. The reason is that a larger filter threshold means smaller sub-model in each node, the optimization will take less time. In addition, comparing the convergence curves on three datasets, it can be found that the convergence acceleration on the higher-dimensional dataset url is more obvious. The experiment shows that SCAC-ADMM has good convergence speed and it can be accelerated by setting appropriate parameters.

C. Performance Test

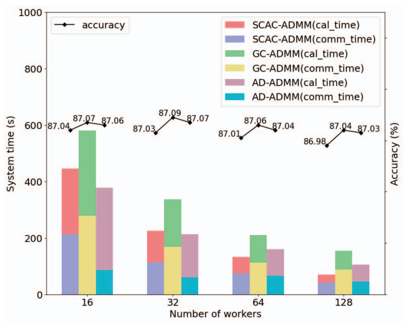
In this section, we test the system time and accuracy of SCAC-ADMM, GC-ADMM and AD-ADMM by running them to 100 iterations. The system time consists of two parts: the calculation time (t_{cal}) and the communication time (t_{comm}). The calculation time is defined as the optimization time of x_i and y_i, and the communication time includes the synchronization waiting time and the time of transmitting parameters. Considering the differences between the computation time and communication time of each worker, we record the



(a) kddb

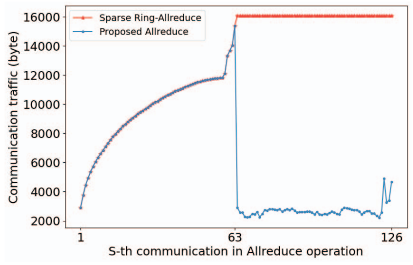


(b) url

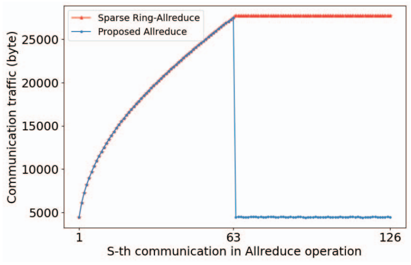


(c) avazu

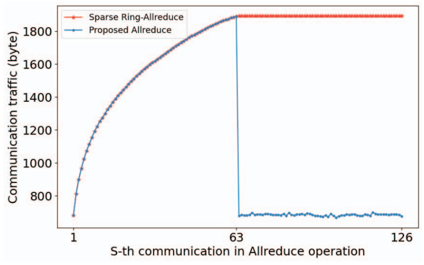
Fig. 8. Performance comparisons among SCAC-ADMM, GC-ADMM and AD-ADMM.



(a) kddb; P=64



(b) url; P=64



(c) avazu; P=64

Fig. 9. Average communication traffic of aggregating a data block in sparse Ring-Allreduce and proposed Allreduce. The Scatter-reduce phase is from 1 to $P-1$ and Allgather phase is from P to $2(P-1)$.

average value of them. The system time and accuracy of three algorithms are shown in Fig. 8.

When compared with GC-ADMM, it can be found that the computation time and communication time of SCAC-ADMM are shorter. In terms of computation time, SCAC-ADMM adopts a sparse calculation method to optimize only the associated features in the local variable, while GC-ADMM optimizes the entire local variable. Therefore, when the number of iterations is same, SCAC-ADMM takes less computation time. In terms of communication time, on the one hand, the reduction of the computation time of SCAC-ADMM leads to a reduction of the synchronization waiting time between workers. On the other hand, SCAC-ADMM is implemented based on Allreduce model we proposed, in which only associated features are transmitted, so the data transmission time is also reduced. However, due to the filtering of some features, the accuracy of SCAC-ADMM decreases slightly compared with GC-ADMM and AD-ADMM.

In Fig. 8, as the number of workers in the system increases, the system time of SCAC-ADMM and GC-ADMM both decreases, but SCAC-ADMM has a greater reduction. The reason is that in SCAC-ADMM, the selection of associated features is based on the number of non-zero elements in training data, when the number of workers increases, the amount of training data in each worker decreases, so fewer features will be selected to compose the sub-model. Therefore, the system time of SCAC-ADMM will decrease due to both less training data and smaller sub-model in each worker, while the system time of the GC-ADMM will decrease only due to less training data in each worker.

Comparing AD-ADMM with GC-ADMM, it can be found that AD-ADMM can also reduce the communication time, since the asynchronous communication is used. However, the computational time in AD-ADMM is still high. But SCAC-ADMM can reduce the communication time and computational time simultaneously, which means SCAC-ADMM can better process high-dimensional data.

Fig. 8 indicates that, under a loss of accuracy within 0.1%, the system time of SCAC-ADMM is reduced by 59% (128 workers) on kddb, by 73% (64 workers) on url and by 54% (128 workers) on avazu compared with GC-ADMM. By setting appropriate parameters, SCAC-ADMM can achieve a large reduction in system time with a slight loss of accuracy compared with GC-ADMM.

D. *Communication Traffic of SCAC-ADMM*

To demonstrate the advantages of the Allreduce model we proposed, we implement SCAC-ADMM based on the sparse Ring-Allreduce, and compare the communication traffic of SCAC-ADMM under these two communication models. Since both communication models transmit data blocks, we track the aggregation process of each data block, and record the traffic of each communication in Scatter-reduce phase and Allgather phase. We run SCAC-ADMM to 100 iterations and count the average communication traffic. The experiments were carried out with 64 workers. And the result is shown in Fig. 9.

In the Scatter-reduce phase, since the number of non-zero elements increases when aggregating data blocks and the use of sparse communication, it can be found from Fig. 9 that the traffic increases gradually. Besides, because the communication of both Allreduce models is performed in the same way in this phase, their traffic is the same.

In the Allgather phase, Fig. 9 shows that the traffic in sparse Ring-Allreduce remains constant, while the traffic in the proposed Allreduce model is reduced greatly. This is because in sparse Ring-Allreduce, each communication in Allgather phase transmits the same complete block, while only the associated features in data block are transmitted through sparse communication in the proposed Allreduce model. Experiment shows that compared with sparse Ring-Allreduce, our Allreduce model has less communication traffic.

We further analyse the communication traffic of SCAC-ADMM. d is used to denote the dimension of local variable and P is used to denote the number of workers in SCAC-ADMM. In a Ring-Allreduce operation, the number of communications is $2\times(P-1)$. If all the parameters are transmitted, the traffic of an Allreduce operation can be represented as $2\times(P-1)\times d/P$. However, in proposed Allreduce model, only associated parameters are transmitted, so the average amount of parameters actually transmitted is less than d, which means there is less communication traffic in it than in Ring Allreduce.

V. Conclusion

In order to improve the efficiency of distributed ADMM for processing high-dimensional data, based on general form consensus optimization, we propose a distributed ADMM based on sparse computation and Allreduce communication (SCAC-ADMM). SCAC-ADMM improves the efficiency of processing high-dimensional data through two approaches: (1) A selecting method of associated features based on the number of non-zero elements in each dimension of dataset is designed, which can limit the dimension of sub-models by setting appropriate filter threshold, so as to limit the computation cost. (2) Based on the characteristic that only associated features are optimized in general form consensus optimization, a novel Allreduce model is proposed to reduce communication traffic by aggregating these associated parameters instead of all the model parameters. Experiments show that SCAC-ADMM has less computational and communication cost when processing high-dimensional data than traditional distributed ADMM. However, due to ignoring some data features, the accuracy of SCAC-ADMM is slightly affected. In future work, we will focus on the how to choose a better filter threshold and the trade-off between accuracy and system time.

Acknowledgment

The research is supported by the National Natural Foundation of China under grant NO. U1811461.

References

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

[2] Y. Li, X. Wang, W. Fang, F. Xue, H. Jin, Y. Zhang, and X. Li, "A distributed admm approach for collaborative regression learning in edge computing," *Comput. Mater. Contin*, vol. 59, pp. 493–508, 2019.

[3] L. Guan, L. Qiao, D. Li, T. Sun, K. Ge, and X. Lu, "An efficient admm-based algorithm to nonconvex penalized support vector machines," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2018, pp. 1209–1216.

[4] X. Jia, S. Song, W. He, Y. Wang, H. Rong, F. Zhou, L. Xie, Z. Guo, Y. Yang, L. Yu *et al.*, "Highly scalable deep learning training system with mixed-precision: Training imagenet in four minutes," *arXiv preprint arXiv:1807.11205*, 2018.

[5] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1, pp. 503–528, 1989.

[6] C.-J. Lin, R. C. Weng, and S. S. Keerthi, "Trust region newton method for large-scale logistic regression." *Journal of Machine Learning Research*, vol. 9, no. 4, 2008.

[7] Q. Liu, X. Shen, and Y. Gu, "Linearized admm for nonconvex nonsmooth optimization with convergence analysis," *IEEE Access*, vol. 7, pp. 76 131–76 144, 2019.

[8] Y. Liu, F. Shang, and J. Cheng, "Accelerated variance reduced stochastic admm," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[9] Y. Xu, M. Liu, Q. Lin, and T. Yang, "Admm without a fixed penalty parameter: Faster convergence with new adaptive penalization," *Advances in neural information processing systems*, vol. 30, 2017.

[10] Y. Liu, K. Yuan, G. Wu, Z. Tian, and Q. Ling, "Decentralized dynamic admm with quantized and censored communications," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019, pp. 1496–1500.

[11] C. B. Issaid, A. Elgabli, J. Park, M. Bennis, and M. Debbah, "Communication efficient distributed learning with censored, quantized, and generalized group admm," *arXiv preprint arXiv:2009.06459*, 2020.

[12] R. Zhang and J. Kwok, "Asynchronous distributed admm for consensus optimization," in *International conference on machine learning*. PMLR, 2014, pp. 1701–1709.

[13] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[14] D. Wang and Y. Lei, "Asynchronous distributed admm for learning with large-scale and high-dimensional sparse data set," in *International Conference on Advanced Hybrid Information Processing*. Springer, 2019, pp. 259–274.

[15] R. Zhu, D. Niu, and Z. Li, "A block-wise, asynchronous and distributed admm algorithm for general form consensus optimization," *arXiv preprint arXiv:1802.08882*, 2018.

[16] P. Patarasuk and X. Yuan, "Bandwidth optimal all-reduce algorithms for clusters of workstations," *Journal of Parallel and Distributed Computing*, vol. 69, no. 2, pp. 117–124, 2009.

[17] J. Xie and Y. Lei, "Admmlib: A library of communication-efficient ad-admm for distributed machine learning," in *IFIP International Conference on Network and Parallel Computing*. Springer, 2019, pp. 322–326.

[18] X. Huang, G. Wang, and Y. Lei, "Gr-admm: A communication efficient algorithm based on admm," in *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*. IEEE, 2021, pp. 220–227.