PSRA-HGADMM: A Communication Efficient Distributed ADMM Algorithm

Yongwen Qiu
School of Computer Engineering and Science Shanghai University
Baoshan, Shanghai, China
wen_2021@shu.edu.cn

Yongmei Lei*
School of Computer Engineering and Science Shanghai University
Baoshan, Shanghai, China
Lei@shu.edu.cn

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

ABSTRACT

Among distributed machine learning algorithms, the global consensus alternating direction method of multipliers (ADMM) has attracted much attention because it can effectively solve large-scale optimization problems. However, the high communication cost slows its convergence and limits scalability. To solve the problem, we propose a hierarchical grouping ADMM algorithm (PSRA-HGADMM) with a novel Ring-Allreduce communication model in this paper. Firstly, we optimize the parameter exchange of the ADMM algorithm and implement the global consensus ADMM algorithm in the decentralized architecture. Secondly, to improve the communication efficiency of the distributed system, we propose a novel Ring-Allreduce communication model (PSR-Allreduce) based on the idea of parameter server architecture. Finally, a Worker-Leader-Group generator (WLG) framework is designed to solve the problem of inconsistency of cluster nodes. This framework combines hierarchical parameter aggregation and adopts the grouping strategy to improve the scalability of the distributed system. Experiments show that PSRA-HGADMM has better convergence performance and better scalability than ADMMLib and AD-ADMM. Compared with ADMMLib, the overall communication cost of PSRA-HGADMM is reduced by 32%.

CCS CONCEPTS

• **Computing algorithms** → **Parallel and Distributed Algorithms**; • **Architecture** → **Parallel Computer Architecture and Accelerator Designs**.

KEYWORDS

The global consensus ADMM algorithm, Ring Allreduce, Hierarchical grouping strategy, Worker-Leader-Group generator (WLG) framework, PSRA-HGADMM

*Corresponding author.

ICPP 2023, August 07–10, 2023, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0843-5/23/08…$15.00
https://doi.org/10.1145/3605573.3605610

ACM Reference Format:
Yongwen Qiu, Yongmei Lei, and Guozheng Wang. 2023. PSRA-HGADMM: A Communication Efficient Distributed ADMM Algorithm. In *52nd International Conference on Parallel Processing (ICPP 2023), August 07–10, 2023, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3605573.3605610

1 INTRODUCTION

In recent years, machine learning (ML) has achieved remarkable progress in various fields such as computer vision [18], game development [16], smart healthcare [24], machine translation [10], speech processing [1], etc. However, the reason for this success is the increasing size of the ML models [17] and the exponential explosion of training data [7] as well as breakthroughs in the performance of programmable highly parallel hardware [25]. When facing growing model parameters and training data, it is impractical to train a model with a large number of parameters on a single device. Therefore, it is necessary to optimally train ML models in clusters with new parallel and distributed training algorithms. Distributed optimization problems aim to solve the following consistency problems:

$$\min_{x} \sum_{i=1}^{N} f_i(x) + g(x) \tag{1}$$

This is a distributed machine learning model defined on N workers, where $x \in \mathbb{R}^d$ represents the model parameters, d is the number of features of samples, and $f_i : \mathbb{R}^d \to \mathbb{R}$ is a loss function that needs to be calculated by i-th worker, $g : \mathbb{R}^d \to \mathbb{R} \bigcup \{\infty\}$ is the regularization term.

The distributed optimization algorithm is an important research direction in distribution machine learning. The alternating direction method of multipliers (ADMM) [2] algorithm is an optimization algorithm that effectively solves large-scale machine learning problems. The ADMM algorithm decomposes a large global problem into several local sub-problems by decomposing the coordination process, and the solution to the global problem is obtained by coordinating the solutions of the sub-problems, so the distributed ADMM algorithm is also called the global consensus ADMM algorithm. Nonetheless, because distributed machine learning uses massive data to train high-dimensional and complex models, the efficiency of the ADMM algorithm and the accuracy of the final model are limited by unbalanced data load, limited cluster communication, and high model complexity.

In this paper, to reduce the communication overhead of the global consensus ADMM algorithm in a distributed system and improve the scalability of the global consensus ADMM algorithm,

we propose PSRA-HGADMM based on the global consensus ADMM algorithm. Furthermore, a novel Ring Allreduce communication model is designed to improve the efficiency of parameter exchange. The main contributions of this paper are as follows:

- By analyzing the parameter exchange process of the global consensus ADMM algorithm and the characteristics of the Ring Allreduce [5] algorithm, we combine the parameter exchange idea of the parameter server (PS) [12] and propose a parameter aggregation algorithm PSRA-ADMM for the global consensus ADMM algorithm based on data parallelism [11].
- In order to further speed up the synchronization of global variables, this paper proposes the Worker-Leader-Group generator (WLG) framework combined with the PSRA-ADMM algorithm. This framework combines the parameter hierarchical aggregation strategy with the dynamic group communication strategy to solve the problem of stragglers in the cluster.
- We evaluate the performance of PSRA-HGADMM with public data sets on the Tianhe-2 supercomputing platform. Experimental results prove that our PSRA-HGADMM algorithm has better convergence speed and accuracy and has obvious scalability.

The remainder of the paper is as follows. Section 2 expounds on the necessary background knowledge of the global consensus ADMM. Section 3 introduces the related research work on the distributed ADMM algorithm in recent years on communication optimization. We analyze the design ideas and implementation principles of the PSRA-HGADMM algorithm in Section 4. Section 5 is about the experimental analysis of the PSRA-HGADMM algorithm and verifies the superiority of the algorithm. We conclude this paper and discuss future work in Section 6.

2 BACKGROUND

The global consensus ADMM algorithm can convert the distributed optimization problem in (1) into a global consensus optimization problem as follows:

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

where $x_i \in \mathbb{R}^d$ represents the local variable to be optimized, $z \in \mathbb{R}^d$ represents the global variable. The augmented Lagrangian function (L_ρ) for (2) can be deduced as shown in (3).

$$L_\rho(\tilde{x}, z, y) = \sum_{i=1}^{N} f_i(x_i) + \sum_{i=1}^{N} y_i^T(x_i - z) + \frac{\rho}{2} \sum_{i=1}^{N} \| x_i - z \|_2^2 + g(z) \tag{3}$$

where $y_i \in \mathbb{R}^d$is the dual variable , ρ is the penalty parameter, $\frac{\rho}{2}\sum_{i=1}^{N} \| x_i - z \|_2^2$ is the quadratic penalty term introduced in the augmented Lagrangian function, the purpose is to relax the restriction that the objective function must be strongly convex. The update formulas of the global consensus ADMM algorithm are shown in (4)-(6).

$$x_i^{k+1} = \min_{x_i}(f_i(x_i) + x_i^T y_i^k + \frac{\rho}{2} \| x_i - z^k \|_2^2) \tag{4}$$

$$z^{k+1} = \min_{z}(g(z) - z^T \sum_{i=1}^{N} y_i^k + \frac{\rho}{2} \sum_{i=1}^{N} \| x_i^{k+1} - z \|_2^2) \tag{5}$$

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

The sub-problem-solving process of the global consensus ADMM algorithm can be solved in parallel, and synchronization is required when calculating the global solution. The synchronization process requires synchronization parameters between nodes. When the nodes are out of step, the fast nodes need to wait for the slow nodes. The communication cost will seriously affect the convergence rate of the algorithm, making it difficult for the algorithm to expand to a larger cluster size. In order to solve this problem, many solutions have been proposed in recent years, such as threshold method, parameter quantization, and communication topology redesign to reduce communication costs. Unfortunately, while reducing communication costs, some parameters are often filtered or data accuracy is expressed with single precision, resulting in a decrease in the accuracy of the model trained by the global consensus ADMM algorithm. Therefore, when the cluster scale continues to expand, how to reduce the communication overhead of the global consensus ADMM algorithm and ensure that the accuracy of the algorithm does not decrease is the concern of this paper.

3 RELATED WORK

To achieve a balance between computation and communication, Weiyu Li et al. proposed COLA-ADMM [13]. They reduced communication costs by using communication thresholds to control parameter exchange between workers. Q-GADMM [4] is a communication-efficient ADMM algorithm based on the idea of communication quantization. Each worker in Q-GADMM only communicates with two neighbors and quantifies the parameters before communication, which effectively reduces the communication cost. Similarly, SCCD-ADMM [19] is based on the random distribution network of importance sampling, which can effectively reduce the total communication time and calculation time in the decentralized network. Due to the limited communication resources in the decentralized consistency model, inter-nodes compete for limited communication resources, resulting in slower convergence. GADMM [3] communicates with workers in groups to alleviate the problem of communication bottlenecks. Based on GADMM, GR-ADMM [9] uses the ring Allreduce communication strategy in inter-group communication to reduce communication costs in distributed systems.

Based on the SSP computing model [8], Ruiliang Zhang et al. [26] proposed an asynchronous ADMM (AD-ADMM) algorithm with partial barriers and bounded delays. However, AD-ADMM is not scalable due to the high communication overhead in the master-worker architecture. ADMMLib [22] utilizes a hierarchical communication architecture to integrate Ring AllReduce and mixed-precision training, which further effectively reduces the communication cost between nodes. On the other hand, the performance of ADMM is very sensitive to penalty parameters, and it is difficult for non-professional users to choose an appropriate penalty parameter. Zheng Xu [23] et al. proposed the Adaptive ADMM (AADMM) algorithm, which can adaptively adjust the penalty parameters to achieve fast convergence. HSAC-ALADMM [21] uses a delayed aggregation parameters communication strategy which reduces the

overhead of workers per iteration. Moreover, a sparse Allreduce communication mode is customized for sparse data, and the parameters that need to be transmitted are segmented and aggregated.

Although ADMMLib [22] uses hierarchical communication strategies under the decentralized communication topology, the application scenario of ADMMLib is to solve the problem of stragglers in the cluster through a limited asynchronous method. Unfortunately, ADMMLib does not achieve a good balance between workers' synchronization and algorithm accuracy. Therefore, this paper proposes an efficient parameter aggregation algorithm for the parameter aggregation characteristics of the global consensus ADMM algorithm. The parameter aggregation strategy combined with hierarchical grouping can effectively reduce communication overhead and speed up the convergence of the algorithm. It achieves a better balance between cluster node synchronization and algorithm accuracy, proving its effectiveness in high-performance clusters.

4 PROPOSED ALGORITHM: PSRA-HGADMM

In this section, by analyzing the communication parameter aggregation characteristics of the global consensus ADMM algorithm, a novel Ring Allreduce parameter aggregation model (PSR-Allreduce) based on the parameter server architecture is proposed. It is used to efficiently utilize the computing resources and bandwidth resources of the cluster, and on this basis, the global consensus ADMM algorithm of hierarchical grouping is implemented.

4.1 Problem formulation

The parameter exchange mode of the global consensus ADMM is the master-worker communication mode. In the master-worker communication mode, the worker sends x_i and y_i to the master and receives z from the master, so the communication load is mainly concentrated on the master, and the generated communication cost becomes the system performance bottleneck. In contrast, although the decentralized ADMM algorithm converges slowly, it can make full use of the network resources of the cluster.

To change the communication mode of the global consensus ADMM, it is necessary to transform the computing process of the ADMM algorithm, and change the update formula of the global variable z as follows:

$$\begin{aligned} z^{k+1} &= \min_z(g(z) - z^T \sum_{i=1}^{N} y_i^k + \frac{\rho}{2} \sum_{i=1}^{N} \| x_i^{k+1} - z \|_2^2) \\ &= \min_z(g(z) + \frac{\rho}{2} \| z \|_2^2 - z^T \sum_{i=1}^{N} (y_i^k + \rho x_i^{k+1})) \end{aligned} \tag{7}$$

Introducing new variables w_i and W, defined as follows:

$$w_i^{k+1} = y_i^k + \rho x_i^{k+1} \tag{8}$$

$$W^{k+1} = \sum_{i=1}^{N} w_i^{k+1} \tag{9}$$

The update formula for the global variable z of the global consensus ADMM algorithm can be transformed into:

$$z^{k+1} = \min_z(g(z) + \frac{\rho}{2} \| z \|_2^2 - z^T W^{k+1}) \tag{10}$$

The calculation of z^{k+1} depends on the cumulative sum of w_i, so each worker no longer sends x_i and y_i separately, but directly sends w_i^{k+1} and receives W^{k+1}. From the update formula of z, it can be seen that this is an Allreduce operation. An iterative process of the global consensus ADMM in the Allreduce communication mode is divided into five steps. The first step is that each worker updates x_i according to the formula (4). The second step is that each worker calculates w_i according to the formula (8). In the third step, all workers perform the Allreduce operation, to sum up w_i to obtain W. The fourth step is that each worker updates z according to the formula (10). The last step is that all workers update y_i according to the formula (6).

The global consensus ADMM algorithm based on the Allreduce communication mode requires an Allreduce operation for each iteration, and the communication cost brought by the Allreduce operation determines the efficiency of the global consensus ADMM algorithm. Therefore, designing an efficient Allreduce communication algorithm that fully utilizes network bandwidth resources is the key to reducing communication costs.

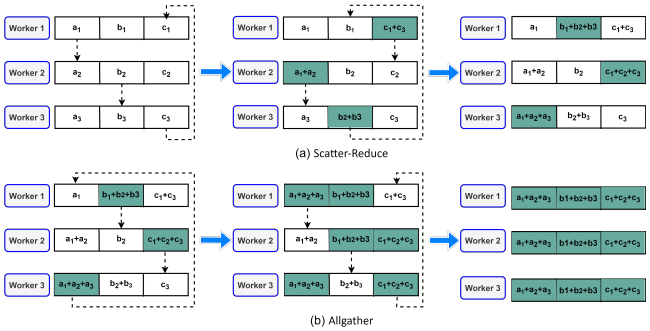

Figure 1: Schematic diagram of data block aggregation based on Ring Allreduce, including the Scatter-Reduce stage and the Allgather stage, a_i, b_i, c_i represents the data block split by worker i, and the green block represents the aggregated data block after one iteration, where $N = 3$.

4.2 PSR-Allreduce design

The Ring Allreduce communication model is based on decentralized network topology. All workers in the cluster are connected in a logical ring. As shown in Figure. 1, Ring Allreduce divides Allreduce into two stages: the Scatter-Reduce stage and the Allgather stage. Each stage requires $N - 1$ iterations, where N is the number of workers in the cluster, and the communication cost of each iteration is given by the slowest worker determination in a unidirectional loop.

In the sparse data storage format, it is assumed that the number of non-zero elements in the data block owned by each worker is c, the time required to transmit a data element is $\theta_s = (value + index)/B$, where $value$ is the byte size required to store the data element, $index$ is the byte size required to store the index, and B is the communication bandwidth. Since the communication content of each iteration of the Ring Allreduce communication model is a data block instead of the entire vector, the communication capability of

Ring Allreduce will not be affected when the size of the transmitted data blocks is the same, but it will be affected when the size of the data blocks is different under sparse communication. Therefore, the minimum communication cost of Ring Allreduce is that all non-zero elements are evenly distributed in each data block, and the maximum communication cost is that all non-zero elements are concentrated in the same data block. The communication cost of Ring Allreduce in the Scatter-Reduce stage is:

$$\frac{c\theta_s(N-1)}{N} \leq T_{ring-sr} \leq \frac{cN\theta_s(N-1)}{2} \tag{11}$$

The communication cost of Ring Allreduce in the Allgather stage is:

$$\frac{c\theta_s(N-1)}{N} \leq T_{ring-ag} \leq cN\theta_s(N-1) \tag{12}$$

Then a Ring Allreduce communication cost is:

$$\frac{2c\theta_s(N-1)}{N} \leq T_{ring-allreduce} \leq \frac{3cN\theta_s(N-1)}{2} \tag{13}$$

From (13), it can be seen that the difference between the best-case and worst-case communication cost is almost N^2. Therefore, it is necessary to design a new communication model that can reduce the performance impact caused by the sparseness and density of the data set and improve the robustness of cluster communication.

Inspired by the parameter server architecture, a communication model PSR-Allreduce is designed based on Ring Allreduce for aggregation parameters in a decentralized topology architecture. Like Ring Allreduce, PSR-Allreduce also includes the Scatter-Reduce stage and the Allgather stage as shown in Figure. 2. The

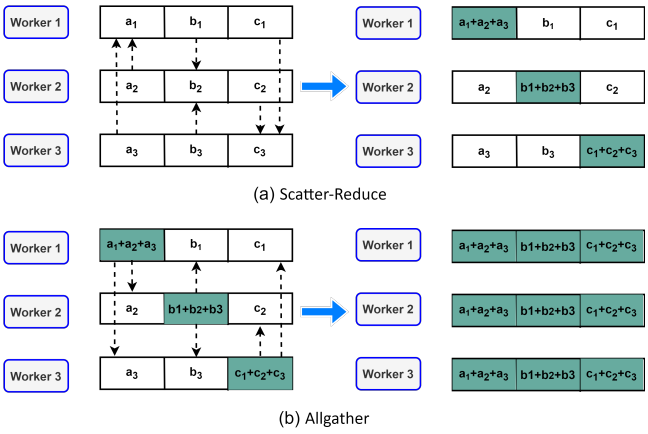

Figure 2: Schematic diagram of data block aggregation based on PSR-Allreduce, including the Scatter-Reduce stage and the Allgather stage, a_i, b_i, c_i represents the data block split by worker i, and the green block represents the aggregated data block after one iteration, where $N = 3$.

PSR-Allreduce divides the data block into N blocks evenly according to the number of workers, each block has a mark corresponding to the worker rank, and each worker is responsible for maintaining the corresponding data block. Different from Ring Allreduce, PSR-Allreduce sends the divided data blocks to the corresponding numbered workers in the Scatter-Reduce stage. After the Reduce-Scatter stage, each worker has the final result of the corresponding numbered data blocks. In the Allgather stage, the block of the final result obtained by each worker is sent to other workers, and finally, each worker has the final model parameter result.

Performance analysis of PSR-Allreduce under sparse communication. In the Scatter-Reduce stage, the best case is that the non-zero elements of all workers are only distributed in the blocks managed by themselves. At this time, the communication cost is the smallest. The worst case is that the non-zero elements of all workers are outside the blocks managed by themselves. At this time, the communication cost is the largest. The communication cost of the Scatter-Reduce stage is:

$$0 \leq T_{psr-sr} \leq c\theta_s \tag{14}$$

For the Allgather stage, since the number of non-zero elements in the result of the Scatter-Reduce stage is $c \leq C \leq Nc$. Therefore, the best case is that there are $\frac{c}{N}$ non-zero elements in the data block managed by each worker, then $C = c$, and the worst case is that there are c non-zero elements in the data block managed by each worker, then $C = Nc$. The communication cost of the Allgather stage is:

$$\frac{c\theta_s(N-1)}{N} \leq T_{psr-ag} \leq c\theta_s(N-1) \tag{15}$$

In the PSR-Allreduce model, the lowest communication overhead in the Scatter-Reduce phase is exactly the highest communication cost in the Allgather stage. At this time, all the non-zero elements of each worker happen to be in the blocks managed by each worker. If it is desired to minimize the total communication cost, the best case is that the non-zero elements on each worker are evenly distributed in each data block. In summary, the communication cost of the PSR-Allreduce is as follows:

$$\frac{2c\theta_s(N-1)}{N} \leq T_{psr-allreduce} \leq cN\theta_s \tag{16}$$

From the above calculation formula, it can be seen that the communication cost of PSR-Allreduce is N times higher than that in the worst case. It can be said that the performance of the PSR-Allreduce model is better than that of the Ring Allreduce model. Therefore, based on the global consensus ADMM algorithm described in Section 4.1, this paper proposes a PSR-Allreduce-based ADMM algorithm (PSRA-ADMM), which aggregates the parameter w_i during each iteration, and introduces the parameter update rules in Section 4.1.

Even so, the basic communication operation of the PSR-Allreduce is still the Allreduce communication mode, so the PSR-Allreduce requires each worker to communicate. During the calculation process, due to the different loads of each node in the cluster and the speed difference between the bus bandwidth and the network bandwidth, the nodes are out of step, resulting in a waste of computing resources. Therefore, designing a communication model that ensures the nodes in the cluster can quickly synchronize parameters is the key to efficiently utilizing cluster computing resources.

4.3 Hierarchical grouping mode

4.3.1 Basic idea: Modern computers are usually configured with multiple computing cores that share memory and bus bandwidth. High-performance computing clusters are formed by connecting a large number of multi-core computers through the network infrastructure. Since bus bandwidth is much higher than network

bandwidth, communication within a node is faster than communication between nodes. Therefore, to fully utilize the bandwidth of network devices at different levels during communication among workers, this paper performs hierarchical communication on the global variable W according to the communication mechanism of the multi-core cluster and the characteristics of the PSRA-ADMM algorithm.

In other words, all workers are grouped according to whether they are in the same physical node. Workers in the same group form a communication domain and elect a worker responsible for communication between communication domains, which is called the *Leader*. On the other hand, in order to coordinate the communication pace between nodes and improve the utilization rate of cluster computing resources, different nodes are further grouped for communication.

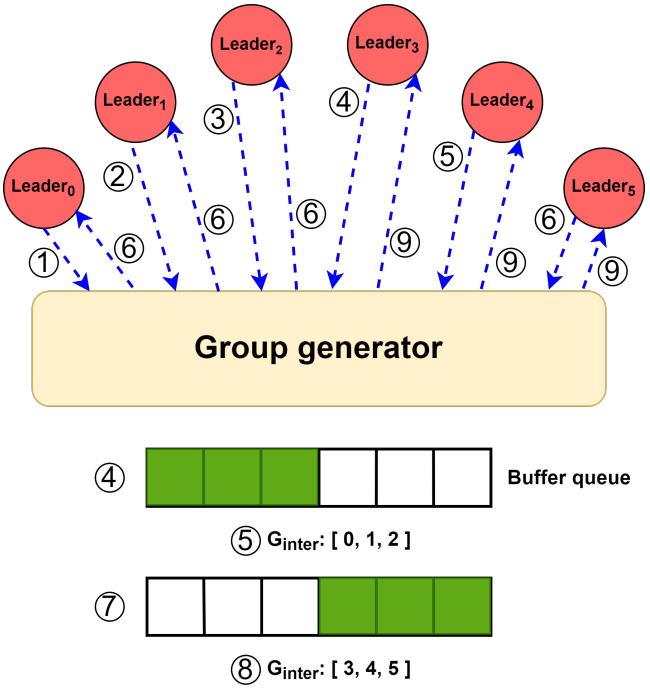

Figure 3: Generates Groups on Behalf of *Leaders*,the threshold is set to 3.

4.3.2 Grouping strategy: In order to group physical nodes according to the actual operation of the cluster, this paper uses a dynamic grouping strategy and proposes a Group Generator (GG) for group management. GG dynamically adjusts the grouping of cluster nodes by setting the threshold value, and users can also set the threshold value by themselves. GG has a group buffer queue GQ. Whenever a *Leader* reports to GG, the rank of the *Leader* is pushed into GQ. Within a given grouping threshold (set to 3 here), the *Leader* reporting to GG will form a communication group, notify the *Leader* in the group to synchronize, and then enter the next grouping cycle.

As shown in Figure. 3, we consider 6 nodes, each node contains 3 workers, and each node elects a *Leader* responsible for inter-node communication. In the beginning, *Leader*0, *Leader*1, and *Leader*2 need to perform inter-node synchronization after completing local iterations, and *Leader* needs to send a request to GG, indicated in ①, ② and ③. GQ saves the synchronization request of the *Leader*, when the group threshold is reached (④), GG generates a group G_{inter}: [0,1,2] (⑤), and broadcasts the group information to *Leader*0, *Leader*1 and *Leader*2 (⑥). GG is receiving synchronization requests from *Leader*3, *Leader*4, and *Leader*5 while performing operations ④, ⑤ and ⑥ simultaneously. When the grouping threshold is reached (⑦), GG generates group G_{inter}: [3,4,5] (⑧), and broadcasts grouping information to *Leader*3, *Leader*4, and *Leader*5 (⑨).

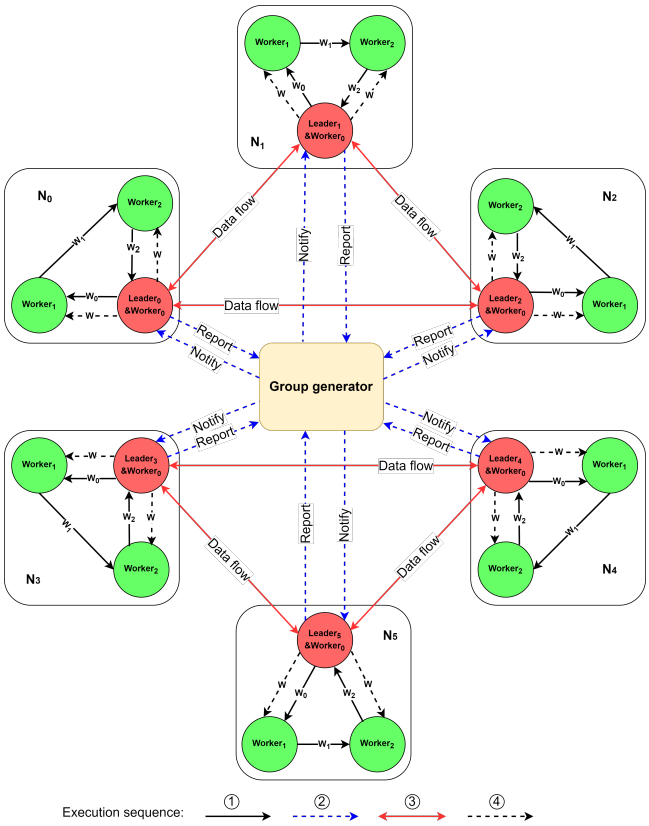

Figure 4: The Worker-Leader-Group generator framework

4.3.3 Implement PSRA-ADMM algorithm based on hierarchical grouping model: In this part, a Worker-Leader-Group generator (WLG) framework is designed based on the hierarchical grouping communication to implement the PSRA-ADMM algorithm. The underlying communication of the WLG communication framework is implemented based on MPICH [6]. The WLG framework is shown in Figure. 4: the framework includes a GG and N nodes. Each node contains several workers to form a communication domain and a *Leader* is elected from the workers to be responsible for inter-group communication. All *Leaders* generate groups between different communication domains through GG. After the local iteration is completed, the *Leader* sends an inter-group synchronization request to the GG, and the GG regenerates the different ones. Since the

inner *Leader* is implemented based on Batch Synchronous Parallel (BSP) [20], the inner communication uses blocking communication. The PSRA-ADMM algorithm based on the hierarchical grouping communication model (PSRA-HGADMM) includes four steps: (1) After the cluster is initialized, workers in the same node form a communication domain G_{intra}, and a *leader* in the communication domain is selected. (2) Workers in the same physical node calculate x_i and w_i, and reduce to the *Leader* in the same communication domain to generate W. At the same time, the *Leader* sends a grouping request to the GG, and the GG groups all physical nodes according to the dynamic grouping strategy. (3) After the *Leader* receives the group information G_{inter}, it executes the PSR-Allreduce algorithm to exchange the parameter in the group according to the G_{inter}. (4) The *Leader* obtains the updated parameter and broadcasts it to all workers in the G_{intra}, and starts the next round of computing iterations. The above four steps are repeated until the stop condition is reached. The whole algorithm is described in Algorithms 1-3.

Algorithm 1 The PSRA-HGADMM algorithm: the process of the worker

```
for all workers [in parallel] do do
    initialize: x_i = 0, y_i = 0, k = 0
    generating the G_intra
    select a worker as the Leader in G_intra
    while k < max_Iteration do
        k = k + 1
        update x_i by (4)
        update w_i by (8)
        do Allreduce operation to get w in the G_intra
        notify Leader of finishing updating W
        wait until receiving global W from Leader
        update z by (10)
        update y_i by (6);
    end while
end for
```

Algorithm 2 The PSRA-HGADMM algorithm: the process of the group generator

```
initialize: GQ= N,G_intra=null, G_inter=null, k=0, threshold=3
repeat
    for k < threshold do
        k = k + 1
        wait until receiving the report from Leader i
        add Leader i to GQ
    end for
    generate the G_inter from GQ
    send G_inter to all Leaders in G_inter
    clear GQ
until terminal
```

Table 1: Summary of datasets

Datasets	Dimension	Training set	Test set
news20	1355191	16000	3996
webspam	16609143	300000	50000
url	3231961	2000000	396130

Algorithm 3 The PSRA-HGADMM algorithm: the process of the *Leader i*

```
repeat
    sends a report to the group generator
    get G_inter from group generator
    wait until receiving notice from the worker
    do PSR-Allreduce to update W in G_inter
    broadcast W to all worker in G_intra
until terminal
```

5 EXPERIMENTAL

5.1 Environment settings

The experimental platform in this paper is the Tianhe-2 supercomputer. The maximum number of physical nodes used in the experiment is 32 nodes, and each node runs a maximum of 16 processes. Each computing node contains 2 Intel Xeon E5-2692 v2 (12 core/ 2.2GHz) CPUs and 64GB of memory. The operating system that each node runs is Red Hat Enterprise Linux Server release 6.5 (Santiago) with GCC/4.8.4 and MPICH/3.2 environments. The nodes are connected through the self-developed high-speed internet network TH2 Express-2+14 Gbps × 8lane. The above algorithms are all implemented using C++ programming language, and the communication between distributed cluster nodes is implemented using the MPICH/3.2 communication library.

5.2 Baseline and benchmarks

In this section, we solve the logistic regression problem with L1 regularization using PSRA-HGADMM. The convergence, system time, and accuracy of the PSRA-HGADMM algorithm are tested and compared with ADMMLib and AD-ADMM introduced in Section 3. The public data sets used in this experiment are news20[1], webspam[2], and url[3]. The detailed information on the data sets is shown in Table 1.

In the experiments, we solve the logistic regression problem with L1 regularization as follows:

$$\min_{x} \sum_{i=1}^{N} \log(1 + e^{-b_i D_i^T x}) + \lambda \|x\|_1 \tag{17}$$

Where $x \in R^d$ is model parameters, $D_i \in R^d$ is the *i-th* training sample, $b_i \in \{-1, 1\}$ represents the label of the *i-th* sample, λ is the regularization parameter and we set it to 1 in the experiment. We use the trust region Newton methods (Tron) [14] to solve subproblems in the global consensus ADMM. Since the computing model used by

[1] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html#news20
[2] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html#webspam
[3] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html#url

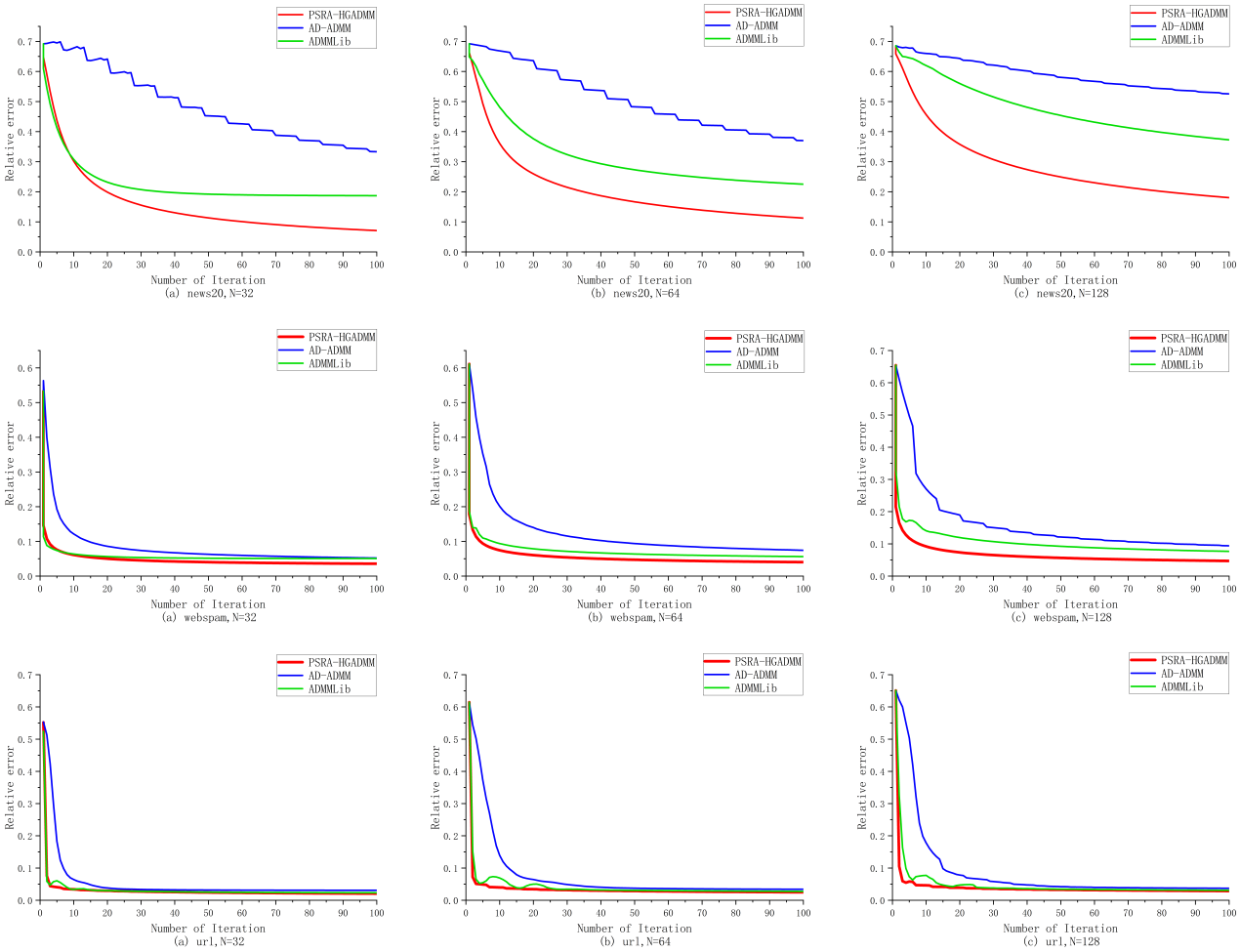

Figure 5: Relative error of PSRA-HGADMM, ADMMLib, and AD-ADMM

ADMMLib and AD-ADMM are all SSP, it is necessary to set the two hyperparameters of these algorithms, *Min_barrier* and *Max_delay*. In the following experiments, we set the *Min_barrier* to half the number of workers and the *Max_delay* to 5.

5.3 Convergence

In this part, we test the convergence of PSRA-HGADMM, ADMMLib, and AD-ADMM. The *max_iteration* in this experiment is set to 100 times, and the group generator cache queue GQ of PSRA-HGADMM is initialized to half the number of nodes, the number of nodes used in the experiment is 8, and the number of workers changes to 32, 64, 128, that is, the number of the workers in each node is 4, 8, 16. The algorithm convergence is measured by the relative target error expressed by equation (18).

$$f_{relative\_error} = |f^* - f|/f \tag{18}$$

Where f represents the minimum value that the loss function in the algorithm can take, and f^* represents the value of the loss function in the current state. The experimental results are shown in Figure. 5.

As shown in the convergence curve in Figure. 5, these three different ADMM algorithms gradually tend to be flat after the number of iterations increases, but there are differences in the convergence of these three different ADMM algorithms from the perspective of convergence degree. It can be seen from Figure. 5 that the convergence of PSRA-HGADMM has obvious advantages over the other two ADMM algorithms, and this advantage becomes more obvious as the number of workers continues to increase. The reason is that although PSRA-HGADMM and ADMMLib both use a hierarchical communication strategy, PSRA-HGADMM's computational model is a BSP computing model, while ADMMLib's computing model is SSP. The stragglers of the SSP computing model use stale values as update parameters, so PSRA-HGADMM can achieve a better convergence effect in fewer iterations. On the other hand,

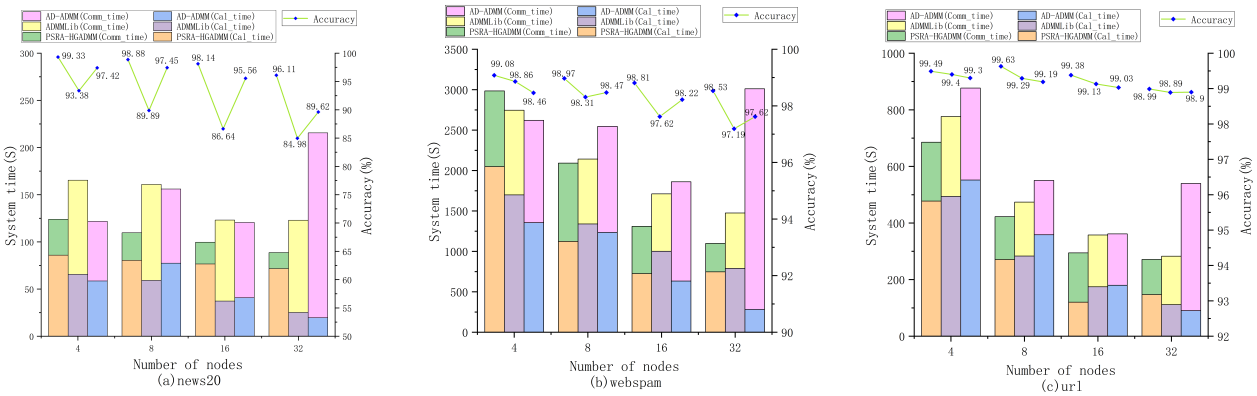

Figure 6: System time and accuracy of various algorithms for different numbers of nodes

PSRA-HGADMM uses the proposed PSR-Allreduce communication model in parameter exchange, while ADMMLib uses the Ring Allreduce communication model. Theoretically, the communication overhead of the PSR-Allreduce communication model is less than that of Ring Allreduce, so this experiment also proves that the PSR-Allreduce communication model has more advantages in distributed ADMM. In terms of implementation, PSRA-HGADMM uses a decentralized communication topology, while AD-ADMM uses a master-worker structure, and the computing model is limited by the communication bandwidth of the master node. Therefore, it can be seen from Figure. 5 that the scalability and convergence rate of PSRA-HGADMM is better than AD-ADMM.

In summary, the experimental results show that PSRA-HGADMM based on the WLG framework is better than ADMMLib and AD-ADMM in convergence in theory and experiment, and shows better scalability.

5.4 System time

To prove the effectiveness of our proposed WLG framework, in this part, we test the system time of PSRA-HGADMM and the accuracy of the algorithm on the data set news20, webspam, and url. In the experiment, the number of nodes in the cluster is 4, 8, 16, and 32 respectively, and the corresponding number of workers is 16, 32, 64, and 128. The number of iterations in the experiment is 100.

The system time is defined here as the sum of the calculation time (*Cal_time*) and the communication time (*Comm_time*). The calculation time includes the time to update parameters x, y, and W, and the communication time includes the grouping request time and the time to exchange parameter W between nodes. Accuracy is the ratio between the number of samples and the total number of test samples of the given test data classified correctly by the model trained by the algorithm after each iteration. Experimental results of system time and algorithm accuracy are shown in Figure. 6.

As can be seen from Figure. 6, with the increase in the number of nodes, the communication time of PSRA-HGADMM decreases linearly, while that of ADMMLib remains basically unchanged, while that of AD-ADMM increases continuously. Because the communication cost of the master node in the AD-ADMM increases with the increase of the number of nodes, the communication cost of the nodes in the ADMMLib is not affected by the cluster scale, and PSRA-HGADMM based on the WLG framework can effectively reduce the communication cost through the dynamic grouping strategy. On the other hand, although PSRA-HGADMM and ADMMLib use hierarchical communication, PSRA-HGADMM requires less system time than ADMMLib as the number of nodes increases. The system time required by PSRA-HGADMM is reduced by 28.3% on news20, 63.18% on webspam, and 60.4% on url, which indicates that the PSR-Allreduce communication model on distributed ADMM algorithm has better performance than the normal Ring Allreduce.

Figure. 5 and Figure. 6 show that PSRA-HGADMM has higher accuracy and lower objective error after 100 iterations, which indicates that PSRA-HGADMM has significant global convergence. With the expansion of the cluster scale, the number of data samples processed by each worker node will become smaller, so it can be found from Figure. 6 that the accuracy of the three algorithms decreases, but the accuracy of PSRA-HGADMM decreases the least. When the number of nodes increases from 4 to 32, the accuracy of PSRA-HGADMM decreases by 3.23% on news20, 0.57% on webspam, and 0.59% on url. This shows that the accuracy of PSRA-HGADMM does not decrease with the expansion of cluster size, because the BSP computing model used by PSRA-HGADMM can ensure that the parameters can be updated synchronously, so the scalability of PSRA-HGADMM is better than that of ADMMLib and AD-ADMM.

Therefore, based on the above analysis, the system time of PSRA-HGADMM decreases with the expansion of cluster scale and maintains a high precision. It is proved that PSRA-HGADMM has better advantages in system time and precision, and has better scalability in solving large-scale computing problems.

5.5 Performance of dynamic grouping strategy

To solve the problem of stragglers in a cluster, PSRA-HGADMM uses a strategy of dynamic grouping. In order to be able to test the advantages of the dynamic grouping strategy, inspired by [15], we randomly select nodes and prolong their computation time during computing to simulate the phenomenon that nodes are out of step in a cluster. In the experiment, the number of nodes in the cluster is 4, 8, 16, and 32 respectively, the corresponding number of workers is 16, 32, 64, and 128, and the number of iterations of the experiment

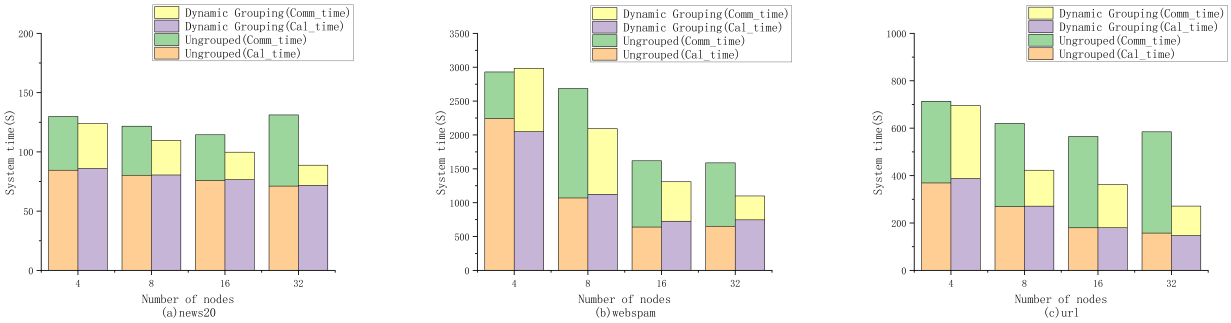

Figure 7: Performance of Dynamic Grouping Strategy

is 100. Here, the system time is defined as the sum of the calculation time (*Cal_time*) and the communication time (*Comm_time*). The experimental results are shown in Figure. 7.

As can be seen from Figure. 7, when the number of nodes is 4, there is little difference between the system time of the dynamic grouping strategy and that of the ungrouped. On high dimensional data sets such as webspam and url, the ungrouped system time is even less than the dynamic grouping strategy time. This is because the number of nodes is small and the synchronization waits time of nodes is small. On the other hand, the PSRA-HGADMM which uses the dynamic grouping strategy consumes time on node grouping and increases the system time. Therefore, the advantage of PSRA-HGADMM which uses the dynamic grouping strategy is not obvious. Despite this, when the number of nodes increases from 4 to 32, the communication time required by PSRA-HGADMM with dynamic grouping strategy decreases significantly, while the communication time required by PSRA-HGADMM without dynamic grouping increases. For example, the communication time of PSRA-HGADMM with dynamic grouping strategy is reduced by 62% on webspam, while the communication time of PSRA-HGADMM without dynamic grouping strategy is increased by 36% on webspam. The reason for our analysis is that PSRA-HGADMM avoids too long communication time by dynamically modifying the grouping, and the advantage of the dynamic grouping strategy becomes more obvious with the increasing number of nodes.

Through the above experiments, we can see that the dynamic grouping strategy can effectively solve the problem of the inconsistent step of the nodes in the cluster and reduce the waiting time required for global synchronization of the BSP computing model. PSRA-HGADMM not only guarantees the accuracy of the algorithm but also has better scalability, which proves that it has great advantages in solving large-scale computing problems.

6 CONCLUSION AND FUTURE WORK

In order to reduce the communication cost and improve the scalability of the global consensus ADMM algorithm, this paper proposes a hierarchical grouping ADMM algorithm (PSRA-HGADMM) based on a novel Ring Allreduce communication model (PSR-Allreduce). PSRA-HGADMM reduces the system time and ensures the scalability of the algorithm through three methods: The first is to introduce new parameter variables W in the global consensus ADMM algorithm so that the global consensus ADMM algorithm can be implemented on a decentralized architecture. The second is to optimize the parameter exchange process of Ring Allreduce based on the parameter server architecture and propose the PS Ring Allreduce (PSR-Allreduce) communication model to improve the utilization rate of cluster bandwidth under the condition of sparse data. The third is to design the Worker-Leader-Group generator (WLG) framework, which solves the problem of inconsistent computing nodes in the cluster and speeds up the convergence rate of PSRA-HGADMM. Experiments for logistic regression with L1 regularization show that PSRA-HGADMM can converge faster with less communication time than AMMLib and AD-ADMM, and the dynamic grouping strategy shows better advantages. A limitation lies in that the dynamic grouping strategy consumes a lot of system time when the number of nodes is relatively small. In future work, we will focus on optimizing the dynamic grouping strategy to reduce the cost of system time.

ACKNOWLEDGMENTS

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

REFERENCES

[1] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*. PMLR, 173–182.

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

[3] Anis Elgabli, Jihong Park, Amrit S Bedi, Mehdi Bennis, and Vaneet Aggarwal. 2020. GADMM: Fast and communication efficient framework for distributed machine learning. *J. Mach. Learn. Res.* 21, 76 (2020), 1–39.

[4] Anis Elgabli, Jihong Park, Amrit Singh Bedi, Chaouki Ben Issaid, Mehdi Bennis, and Vaneet Aggarwal. 2020. Q-GADMM: Quantized group ADMM for communication efficient decentralized machine learning. *IEEE Transactions on Communications* 69, 1 (2020), 164–181.

[5] Andrew Gibiansky. 2017. Bringing HPC techniques to deep learning. *Baidu Research, Tech. Rep.* (2017).

[6] William Gropp, William D Gropp, Ewing Lusk, Anthony Skjellum, and Argonne Distinguished Fellow Emeritus Ewing Lusk. 1999. *Using MPI: portable parallel programming with the message-passing interface*. Vol. 1. MIT press.

[7] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Patwary, Mostofa Ali, Yang Yang, and Yanqi Zhou. 2017.

Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409* (2017).

[8] Qirong Ho, James Cipar, Henggang Cui, Jin Kyu Kim, and Eric P Xing. 2013. More Effective Distributed ML via a Stale Synchronous Parallel Parameter Server. *Advances in Neural Information Processing Systems* 2013, 2013 (2013), 1223.

[9] Xin Huang, Guozheng Wang, and Yongmei Lei. 2021. 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, 220–227.

[10] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics* 5 (2017), 339–351.

[11] Arun Kumar, Matthias Boehm, and Jun Yang. 2017. Data management in machine learning: Challenges, techniques, and systems. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 1717–1722.

[12] Mu Li, David G Andersen, Alexander J Smola, and Kai Yu. 2014. Communication efficient distributed machine learning with the parameter server. *Advances in Neural Information Processing Systems* 27 (2014).

[13] Weiyu Li, Yaohua Liu, Zhi Tian, and Qing Ling. 2019. Communication-censored linearized ADMM for decentralized consensus optimization. *IEEE Transactions on Signal and Information Processing over Networks* 6 (2019), 18–34.

[14] Chih-Jen Lin, Ruby C Weng, and S Sathiya Keerthi. 2007. Trust region newton methods for large-scale logistic regression. In *Proceedings of the 24th international conference on Machine learning*. 561–568.

[15] Qinyi Luo, Jinkun Lin, Youwei Zhuo, and Xuehai Qian. 2019. Hop: Heterogeneity-aware decentralized training. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. 893–907.

[16] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529, 7587 (2016), 484–489.

[17] K Sohn, H Lee, X Yan, C Cortes, N Lawrence, and D Lee. 2015. Advances in neural information processing systems. *Neural Information Processing Systems Foundation, Curran Associates, Inc* (2015), 3483–3491.

[18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[19] Zhuojun Tian, Zhaoyang Zhang, Jue Wang, Xiaoming Chen, Wei Wang, and Huaiyu Dai. 2020. Distributed ADMM with synergetic communication and computation. *IEEE Transactions on Communications* 69, 1 (2020), 501–517.

[20] Leslie G Valiant. 1990. A bridging model for parallel computation. *Commun. ACM* 33, 8 (1990), 103–111.

[21] Dongxia Wang, Yongmei Lei, Jinyang Xie, and Guozheng Wang. 2021. HSAC-ALADMM: an asynchronous lazy ADMM algorithm based on hierarchical sparse allreduce communication. *The Journal of Supercomputing* 77, 8 (2021), 8111–8134.

[22] Jinyang Xie and Yongmei Lei. 2019. ADMMLIB: A library of communication-efficient AD-ADMM for distributed machine learning. In *IFIP International Conference on Network and Parallel Computing*. Springer, 322–326.

[23] Zheng Xu, Mario Figueiredo, and Tom Goldstein. 2017. Adaptive ADMM with spectral penalty parameter selection. In *Artificial Intelligence and Statistics*. PMLR, 718–727.

[24] Kun-Hsing Yu, Andrew L Beam, and Isaac S Kohane. 2018. Artificial intelligence in healthcare. *Nature biomedical engineering* 2, 10 (2018), 719–731.

[25] Hao Zhang, Zeyu Zheng, Shizhen Xu, Wei Dai, Qirong Ho, Xiaodan Liang, Zhiting Hu, Jinliang Wei, Pengtao Xie, and Eric P Xing. 2017. Poseidon: An efficient communication architecture for distributed deep learning on {GPU} clusters. In *2017 USENIX Annual Technical Conference (USENIX ATC 17)*. 181–193.

[26] Ruiliang Zhang and James Kwok. 2014. Asynchronous distributed ADMM for consensus optimization. In *International conference on machine learning*. PMLR, 1701–1709.