ZipLine: An Optimized Algorithm for the Elastic Bulk Synchronous Parallel Model

Xing Zhao*, Manos Papagelis*, Aijun An*, Bao Xin Chen*, Junfeng Liu†, Yonggang Hu†
**Department of Electrical Engineering and Computer Science, York University,* Toronto, Canada
†*Platform Computing, IBM Canada,* Markham, Canada
{xingzhao, papaggel, aan, baoxchen}@eecs.yorku.ca;{jfliu, yhu}@ca.ibm.com

***Abstract*—The *bulk synchronous parallel* (BSP) is a celebrated *synchronization model* for distributed training of deep learning models. A shortcoming of the BSP is that it requires workers to wait for the straggler at every iteration. Therefore, employing BSP increases the waiting time of the faster workers of a cluster and results in an overall prolonged training time. To ameliorate this shortcoming of BSP, we proposed ElasticBSP [1], a model that aims to relax its strict synchronization requirement with an *elastic synchronization* by allowing delayed synchronization to minimize the waiting time. ElasticBSP is realized by the algorithm named ZipLine. In this work, we show the theoretical proof of ZipLine and further propose algorithmic and implementation optimizations of ZipLine, namely ZipLineOpt and ZipLineOptBS, which reduce the time complexity of ZipLine to linearithmic time. The experiments show that ZipLineOpt and ZipLineOptBS enable the scalability of ElasticBSP. Further experimental evaluation on large deep neural networks on large ImageNet dataset demonstrate that our proposed ElasticBSP model, materialized by the proposed optimized ZipLine variants, converges faster and to a higher accuracy than the predominant BSP.**

***Index Terms*—distributed deep learning, parameter server framework, data parallelism, bulk synchronous parallel, stale synchronous parallel, asynchronous parallel**

I. Introduction

The *parameter server framework* [2] has been widely adopted to distribute the training of large deep neural networks (DNNs). The framework consists of multiple *workers* and a logical *server* that maintains globally shared parameters. Due to its importance, a number of *synchronization models* have been proposed [2], the most important of which are the *asynchronous parallel* (ASP) [3], the *bulk synchronous parallel* (BSP) [4], and the *stale synchronous parallel* (SSP) [5] models. Nonetheless, these models exhibit certain limitations. In ASP there is no need for synchronization, so the waiting time of the workers is eliminated. However, the convergence in the training might be dramatically affected due to inconsistent weight updates to the model. On the other hand, a prevalent shortcoming of the BSP is the strict synchronization requirement it imposes. In BSP, all workers are waiting for each other by a synchronization barrier. In SSP, while the strict synchronization requirement of BSP is removed, a user-specified threshold (fixed throughout the training period) is needed to control the maximum iteration difference among workers. Further, SSP offers a shortsighted solution to the problem, as it does not consider the computational capacity

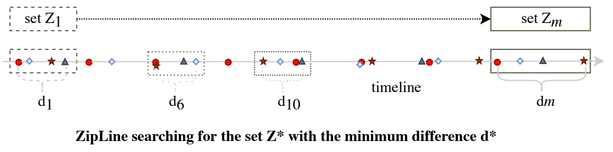
set Z_1
set Z_m
d_1
d_6
d_{10}
timeline
d_m
ZipLine searching for the set Z* with the minimum difference d*

Fig. 1. ZipLine scans all elements on the timeline, from left to right, one element at a time. When a solution Z of n distinct elements is formed, d_Z is computed. At the end of the process the optimal solution Z^* is found that yields the minimum d^*_Z. If multiple solutions exhibit the same d^*_Zs, then the solution Z that occurred first (chronologically) is selected. In this example, d_6 and d_{10} have the same minimum value — Z_6 associated with d_6 is chosen as the optimal solution.

of each worker (i.e., how fast it is), merely relying on the number of iterations that each worker has completed.

To ameliorate these shortcomings, we proposed ElasticBSP [1], a model that relaxes the strict synchronization requirement of the predominant BSP, reduces worker waiting time, and therefore increases the iteration throughput. At the same time, the model limits the staled gradients and their staleness values in the iterative stochastic gradient descent convergent process. ElasticBSP is realized by the ZipLine algorithm, originally proposed in [1]. The current extended abstract provides a summary of the journal archival version of our research [6], including a theoretical proof of ZipLine's optimality, and two optimizations of the general ZipLine algorithm, namely ZipLineOpt and ZipLineOptBS, which reduce its time complexity to linearithmic time and improve the scalability of ElasticBSP. A thorough experimental evaluation on large deep neural networks using the large ImageNet dataset demonstrates that the ElasticBSP model, realized by the proposed optimized ZipLine variants, converges faster and to a higher accuracy than the predominant BSP.

II. ZipLine Optimality and Optimization

ZipLine consists of 2 phases: Phase I initializes the scanning set Z, and Phase II iteratively searches for the optimal set Z^*. Figure 1 provides an intuitive demonstration of how ZipLine works in Phase II. As evidence of ZipLine's correctness, we provide a formal proof of its optimality (see Theorem 1 below and its proof in our journal paper [6]).

Theorem 1: [**ZipLine optimality**] ZipLine leads to an optimal solution Z^*.
Further, we proposed algorithmic and implementation optimizations of ZipLine, namely ZipLineOpt and ZipLineOptBS, which reduce its time complexity to linearithmic time $\mathcal{O}(Rn \log n)$ (see Figure 2). ZipLineOptBS opti-

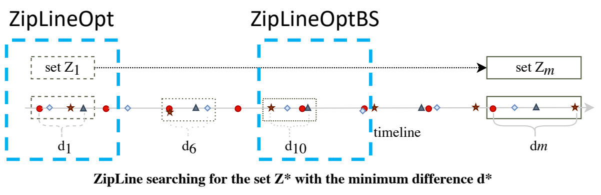

Fig. 2. Two possible cases of the `add` operation in searching for the same color in the scanning set Z: does the leftmost element of Z have the same color as the new adding point? ZipLineOpt checks the leftmost element to skip the search if it has the same color as the new adding point. Otherwise, ZipLineOptBS performs a binary search in Z with an auxiliary matrix. It retrieves the timestamp value of the element of Z with the same color as the new adding point from the auxiliary matrix.

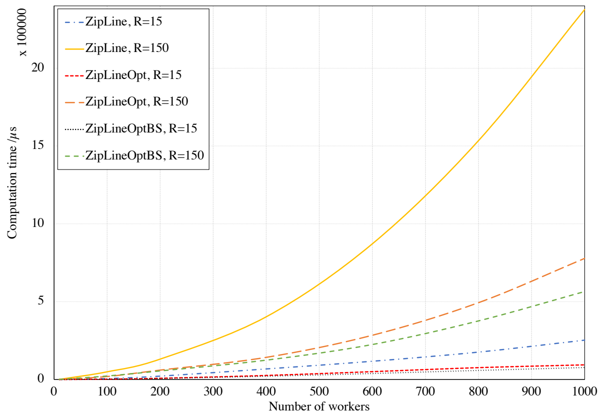

Fig. 3. Computation time cost comparison of ZipLine and its variants. The cost of ZipLine and its variants increases as the number of workers n and the value of parameter R increases. Both *ZipLineOpt* and *ZipLineOptBS* outperform the basic *ZipLine*. For larger values of R ($R \geq 100$), *ZipLineOptBS* outperforms *ZipLineOpt*.

mizes the '`add`' operation of the scanning set Z. It uses a pruning technique to accelerate the addition of new elements to set Z. Specifically, it skips the search if a new element has the same color as the leftmost element of Z. ZipLineOptBS further optimizes the '`add`' operation of the scanning set Z. It uses an auxiliary matrix M containing information (e.g., timestamps and colors) of data points and performs a binary search in set Z for finding the element with the same color as the newly added element. The matrix M: n (workers) $\times$ R (future iterations) is constructed in the *lookalead* step of ElasticBSP (see details in our journal paper [6]).

III. Experiment

We compare the run time of ZipLine and its optimized versions as a function of n workers on varying values of the lookahead parameter R in Figure 3. It can be observed that when the worker number n is small (e.g., below 150), ZipLineOpt is faster than ZipLineOptBS. But, as n increases (e.g., above 200), ZipLineOptBS outperforms ZipLineOpt. We observe in Figure 3 that for R=150, ZipLineOptBS grows significantly slower than ZipLineOpt. The same trend is depicted for R=15 as well – as n increases, ZipLineOptBS outperforms ZipLineOpt.

We compare ElasticBSP with BSP, ASP, SSP and DSSP [7] on training large DNNs on ImageNet1k. As depicted in Figure 4, ElasticBSP outperforms BSP. Particularly, at later

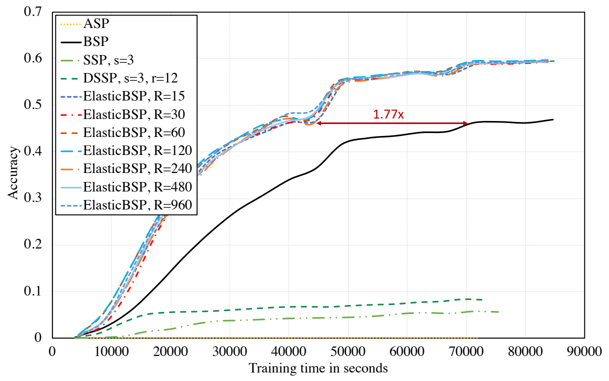

Fig. 4. VGG-16 on ImageNet 1K dataset ($n = 4$)

training time, it converges 1.77× faster and achieves 12.6% higher test accuracy than BSP. We also observe that while SSP, ASP and DSSP complete 19 epochs faster than ElasticBSP, they fail to learn. This is due to staled gradients that are constantly present in the training process.

IV. Conclusions

In this extended abstract, we discussed the optimality of ZipLine, and presented ZipLineOpt and ZipLineOptBS that improve on the scalability of ElasticBSP. ZipLineOptBS has a linearithmic time complexity which supports large-scale scalability for ElasticBSP. The experimental results show that ElasticBSP provides faster convergence than BSP for large-sized DNNs on the high dimensional dataset while achieving higher (or comparable) accuracy than other most related state-of-the-art synchronization models on large datasets in the parameter server setting.

Acknowledgment

This work is funded by the Natural Sciences and Engineering Research Council of Canada (NSERC), IBM Canada and the Big Data Research Analytics and Information Network (BRAIN) Alliance established by Ontario Research Fund - Research Excellence Program (ORF-RE). The experiments were performed on the GPU cluster of SOSCIP.

References

[1] X. Zhao, M. Papagelis, A. An, B. X. Chen, J. Liu, and Y. Hu, "Elastic bulk synchronous parallel model for distributed deep learning," in *2019 IEEE International Conference on Data Mining*, 2019, pp. 1504–1509.

[2] M. Langer, Z. He, W. Rahayu, and Y. Xue, "Distributed training of deep learning models: A taxonomic perspective," *IEEE Trans Parallel Distrib Syst*, vol. 31, no. 12, pp. 2802–2818, 2020.

[3] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. a. Ranzato, A. Senior, P. Tucker, K. Yang, Q. Le, and A. Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012, pp. 1223–1231.

[4] A. V. Gerbessiotis and L. G. Valiant, "Direct bulk-synchronous parallel algorithms," *Journal of parallel and distributed computing*, vol. 22, no. 2, pp. 251–267, 1994.

[5] Q. Ho, J. Cipar, H. Cui, S. Lee, J. K. Kim, P. B. Gibbons, G. A. Gibson, G. Ganger, and E. P. Xing, "More effective distributed ml via a stale synchronous parallel parameter server," in *NeurIPS*, 2013, pp. 1223–1231.

[6] X. Zhao, M. Papagelis, A. An, B. X. Chen, J. Liu, and Y. Hu, "Zipline: an optimized algorithm for the elastic bulk synchronous parallel model," *Machine Learning. In Press. Accepted: Sep 3*, 2021.

[7] X. Zhao, A. An, J. Liu, and B. X. Chen, "Dynamic stale synchronous parallel distributed training for deep learning," in *IEEE 39th International Conference on Distributed Computing Systems*, 2019, pp. 1507–1517.