Baton: Orchestrating GPU Memory for LLM Training on Heterogeneous Cluster

Yi Zhang
Zhejiang University

Shuibing He
Zhejiang University

Ping Chen
Zhejiang University

1 Background and Motivation

Large language models require substantial memory due to their growing size and sequence length, necessitating multiple GPUs for training [11]. As new GPUs are released annually, LLM training companies often deploy new GPUs in training clusters to boost the computing power, resulting in heterogeneous clusters composed of both high- and low-end devices [1]. Therefore, optimizing the use of heterogeneous GPUs is crucial for reducing costs [5, 8].

Existing memory optimization strategies include model parallelism [4], tensor parallelism [10], and activation recomputation [6]. However, existing training frameworks fail to achieve efficient hybrid strategies on heterogeneous clusters due to a lack of consideration for GPU diversity. There are several challenges. (1) How to efficiently partition models to balance the workload across GPUs with varying capabilities [7, 13]. Traditional distributed training assumes homogeneous GPUs and uses uniform load balancing [9, 10], which fails to address the performance and memory discrepancies in heterogeneous GPUs. This leads to load imbalance, where high-performance GPUs remain idle waiting for slower ones, causing resource underutilization and reduced system throughput [5]. (2) Existing homogeneous tensor parallelism (TP) strategies evenly distribute model weights and computation across GPUs. In heterogeneous TP groups, this strategy fails to match the varying computational and memory capabilities of different GPUs, leading to high-performance GPUs being idle and inefficient training. (3) Current recomputation strategies, which only retain activation boundaries and recompute others during backpropagation [3]. This coarse-grained approach does not account for the computational and memory requirements of heterogeneous GPUs, resulting in redundant recomputation time, exacerbating computational and memory imbalance, and failing to address the imbalance in activation memory consumption across pipeline stages, further reducing heterogeneous GPU utilization [11].

2 Our Approach

We propose Baton, a systematic approach to addressing core issues in LLM training on heterogeneous clusters. To solve challenges (1) and (2), Baton introduces the TP partition granularity to control the minimum partition size of Transformer layer weight tensors, enabling non-uniform tensor parallelism by allocating different numbers of minimum partition units to heterogeneous GPUs. This transforms the TP non-uniform partitioning problem into one similar to the layer partitioning problem, unifying the solutions for PP and TP partition strategies and achieving heterogeneous-aware load balancing. To address challenge (3), Baton uses operators as the minimum unit for recomputation, applying a fine-grained recomputation strategy that dynamically selects recomputations. It considers heterogeneous GPU performance and memory capacity to optimize and solve for the best strategy, achieving adaptive recomputation while reducing overhead and meeting memory requirements. Table 1 shows the support of different training systems for heterogeneous training features. Baton supports heterogeneous-aware tensor and pipeline parallelism and seamlessly integrates recomputation.

Table 1: Comparison of Different Training Systems' Support for Heterogeneous Training Features

	Fine-grained memory optimization	Adaptive layer partition	Adaptive tensor partition	Heterogeneous-aware
Megatron-LM [10]	✘	✘	✘	✘
AdaPipe [11]	✔	✔	✘	✘
Lynx [2]	✔	✔	✘	✘
Metis [12]	✘	✔	✘	✔
HAP [14]	✘	✘	✔	✔
Baton(Ours)	✔	✔	✔	✔

Baton is built on two key designs. First, we perform detailed performance profiling of heterogeneous GPUs, capturing key metrics to inform strategy development. This profiling analyzes GPU computing and memory capabilities, providing essential data for optimization. Second, we develop a three-level optimization algorithm that integrates recomputation, tensor, and pipeline parallelism, reducing search complexity through heuristics. This unified approach balances load and memory, optimizing GPU usage and throughput. Baton also customizes partitioning and recomputation strategies for each GPU, unlocking full resource potential and improving training efficiency in heterogeneous clusters. The overview of Baton design is illustrated in Figure 1.

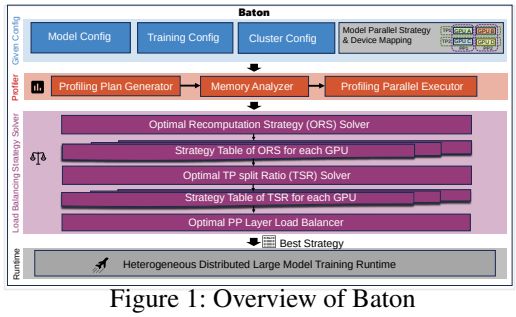

Figure 1: Overview of Baton

We conduct experiments on a heterogeneous cluster of 2 A100 and 2 T4 GPUs, training mini GPT-3 1B model.

Specifically, Baton improves the training throughput by 7.12× compared to LLMs training system Megatron [10]. Baton improves GPU utilization and training efficiency by leveraging heterogeneous-aware recomputation and model partitioning, effectively balancing computation and memory.

References

[1] Amazon Web Services. Configure a training job with a heterogeneous cluster in Amazon SageMaker. https://docs.aws.amazon.com/sagemaker/latest/dg/train-heterogeneous-cluster-configure.html.

[2] Ping Chen, Wenjie Zhang, Shuibing He, Yingjie Gu, Zhuwei Peng, Kexin Huang, Xuan Zhan, Weijian Chen, Yi Zheng, Zhefeng Wang, et al. Optimizing Large Model Training through Overlapped Activation Recomputation. *arXiv preprint arXiv:2406.08756*, 2024.

[3] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training Deep Nets with Sublinear Memory Cost.(2016). *arXiv preprint arXiv:1604.06174*, 2016.

[4] Aaron Harlap, Deepak Narayanan, Amar Phanishayee, Vivek Seshadri, Nikhil Devanur, Greg Ganger, and Phil Gibbons. Pipedream: Fast and Efficient Pipeline Parallel DNN Training. *arXiv preprint arXiv:1806.03377*, 2018.

[5] Xianyan Jia, Le Jiang, Ang Wang, Wencong Xiao, Ziji Shi, Jie Zhang, Xinyuan Li, Langshi Chen, Yong Li, Zhen Zheng, et al. Whale: Efficient Giant Model Training over Heterogeneous {GPUs}. In *Proceedings of the USENIX Annual Technical Conference*, 2022.

[6] Vijay Anand Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeybi, and Bryan Catanzaro. Reducing Activation Recomputation in Large Transformer Models. *Proceedings of Machine Learning and Systems*, 2023.

[7] Dacheng Li, Hongyi Wang, Eric Xing, and Hao Zhang. Amp: Automatically Finding Model Parallel Strategies with Heterogeneity Awareness. *Proceedings of the Advances in Neural Information Processing Systems*, 2022.

[8] Jay H Park, Gyeongchan Yun, M Yi Chang, Nguyen T Nguyen, Seungmin Lee, Jaesik Choi, Sam H Noh, and Young-ri Choi. HetPipe: Enabling Large DNN Training on (whimpy) Heterogeneous GPU Clusters Through Integration of Pipelined Model Parallelism and Data Parallelism. In *Proceedings of the USENIX Annual Technical Conference*, 2020.

[9] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory Optimizations toward Training Trillion Parameter Models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020.

[10] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training Multi-billion Parameter Language Models Using Model Parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[11] Zhenbo Sun, Huanqi Cao, Yuanwei Wang, Guanyu Feng, Shengqi Chen, Haojie Wang, and Wenguang Chen. AdaPipe: Optimizing Pipeline Parallelism with Adaptive Recomputation and Partitioning. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, 2024.

[12] Taegeon Um, Byungsoo Oh, Minyoung Kang, Woo-Yeon Lee, Goeun Kim, Dongseob Kim, Youngtaek Kim, Mohd Muzzammil, and Myeongjae Jeon. Metis: Fast Automatic Distributed Training on Heterogeneous {GPUs}. In *Proceedings of the USENIX Annual Technical Conference*, 2024.

[13] Ran Yan, Youhe Jiang, Wangcheng Tao, Xiaonan Nie, Bin Cui, and Binhang Yuan. FlashFlex: Accommodating Large Language Model Training over Heterogeneous Environment. *arXiv preprint arXiv:2409.01143*, 2024.

[14] Shiwei Zhang, Lansong Diao, Chuan Wu, Zongyan Cao, Siyu Wang, and Wei Lin. HAP: SPMD DNN Training on Heterogeneous GPU Clusters with Automated Program Synthesis. In *Proceedings of the Nineteenth European Conference on Computer Systems*, 2024.