Hanayo: Harnessing Wave-like Pipeline Parallelism for Enhanced Large Model Training Efficiency

Ziming Liu Shenggan Cheng Haotian Zhou Yang You

Introduction

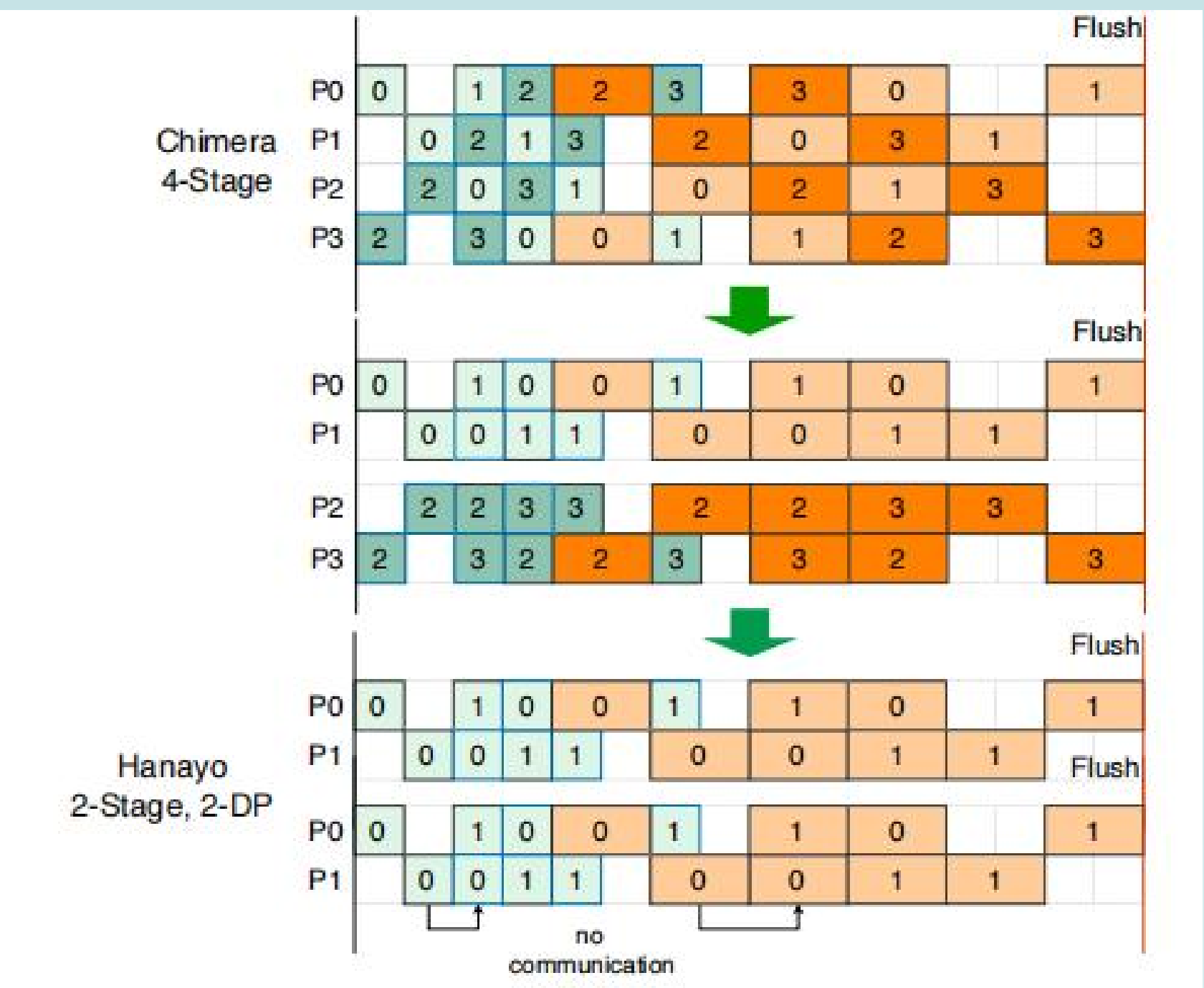
Large-scale language models have become increasingly challenging and expensive to train. Among various methods addressing this issue, Pipeline Parallelism has been widely employed to accom modate massive model weights within limited GPU memory. This paper introduces Hanayo, a wave-like pipeline parallelism strategy that boasts a concise structure and practical applicability, along side a high-performance pipeline execution runtime to tackle the challenges of pipeline strategy implementation. Hanayo mitigates the issues of pipeline bubbles and excessive memory consumption prevalent in existing schemes, without resorting to model dupli cates as in Chimera. Our evaluation, conducted on four distinct computing clusters and involving both GPT-like and BERT-like architectures with up to 32 GPUs, demonstrates up to a 30.4 % increase in throughput compared to the state-of-the-art approach.

Chanllages

- ➢Memory Wall
  - -The size of model parameters far exceeds the memory
- ➢Scaling Wall
  - -Complex parallel patterns and extensive communication lead to bottlenecks in scaling
- ➢Computational Wall
  - -Large models and massive data sets require huge computing power
- ➢Development Wall
  - -Parallel strategies and communication processes render the development of training difficult

Methods

We have found a simple way that leads us out of this dilemma. We know that the high efficiency of Chimera can be primarily attrib uted to its bidirectional pipeline structure, allowing pipelines in different directions to compensate for bubbles. The reason for em ploying model replication is that, in the current pipeline scheduling, the same micro-batch must continuously perform calculations and communication in the same direction. Therefore, when introducing calculations in another direction, another set of models must be stored on the GPU. To address this issue, we only need to enable a single pipeline to change direction during the computation process, transforming it into a wavy-shaped pipeline.



Conclusion

We introduce a wave-like pipeline scheme that achieves a low bubble ratio and high performance in large model training.

It can achieve increasingly higher throughput as the number of waves increases.

- Hanayo proposes a unified framework for pipeline paral lelism. Through theoretical analysis, we obtain a unified performance model for pipeline parallelism.
- In the design and implementation of the runtime system, we aim to decouple the runtime system from specific pipeline parallel algorithms. Utilizing the action list, Hanayo's run time system can support nearly all pipeline parallel algo rithms while optimizing performance through features such as asynchronous communication.
- We conduct experiments with mainstream GPT-style and BERT-style models, performing performance tests for vari ous model sizes on four different computing clusters. Exper imental results demonstrate that Hanayo achieves up to a 30.4% performance improvement over the current state-of the-art pipeline parallelism implementation, Chimera.

Evaluation

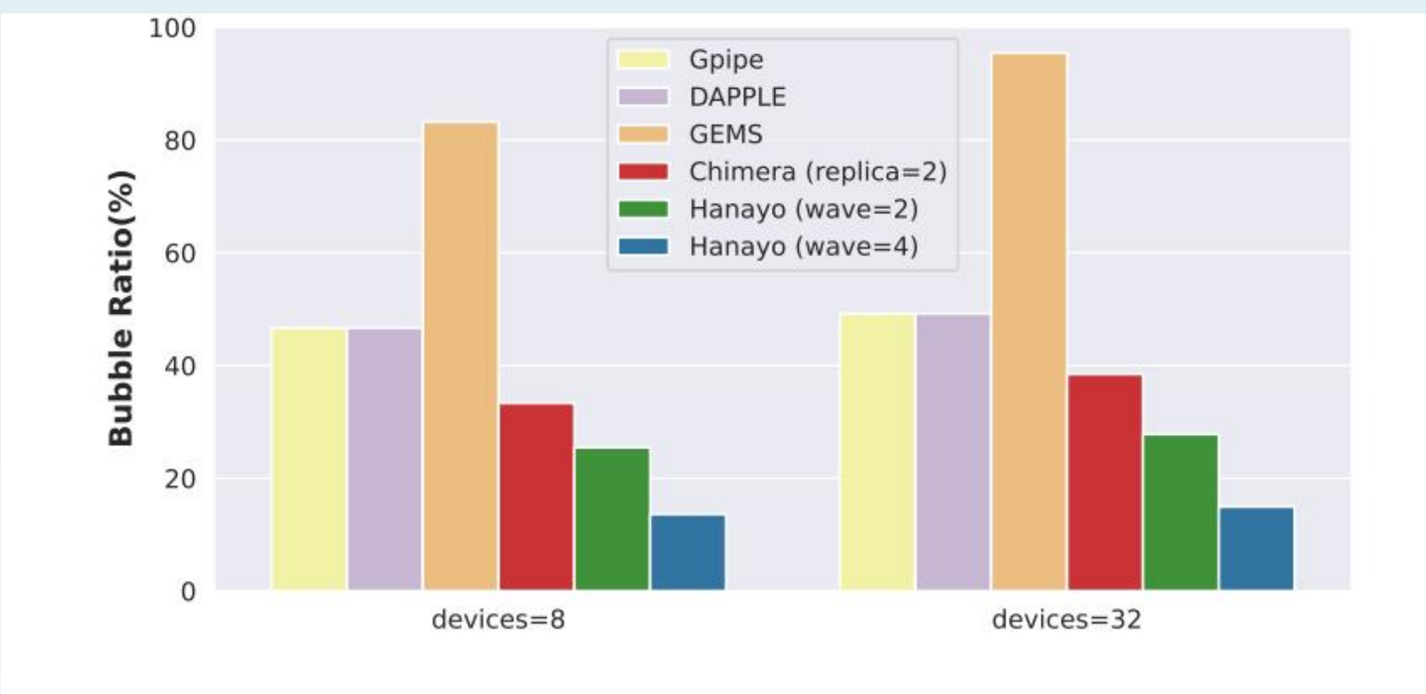


Figure 1: The theoretical bubble ratio of synchronous pipeline schemes

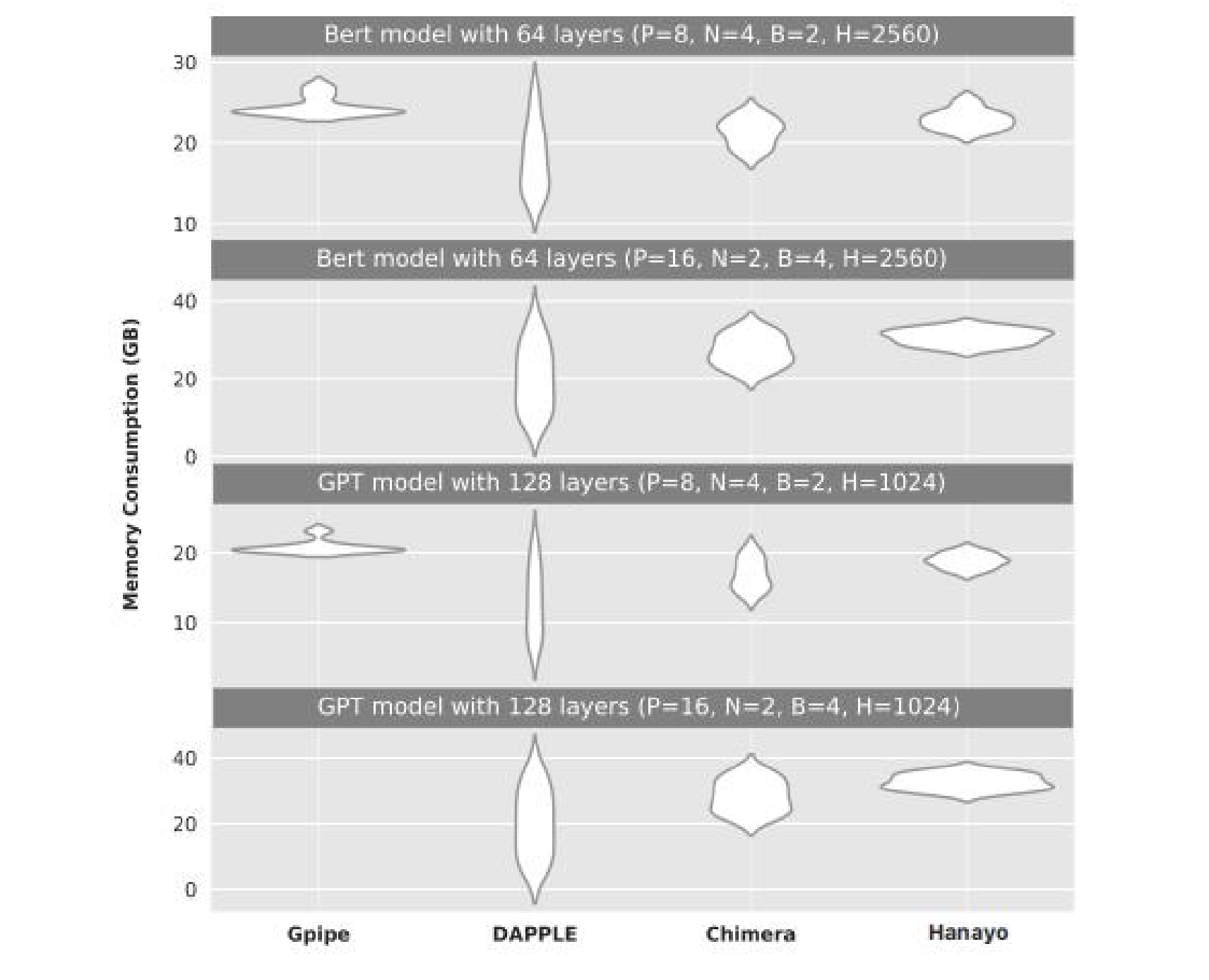


Figure 8: The distribution of peak memory consumption for GPipe, DAPPLE, Chimera, and Hanayo during the training of Bert and GPT model on 32 GPUs of the TACC Lonestar6 cluster

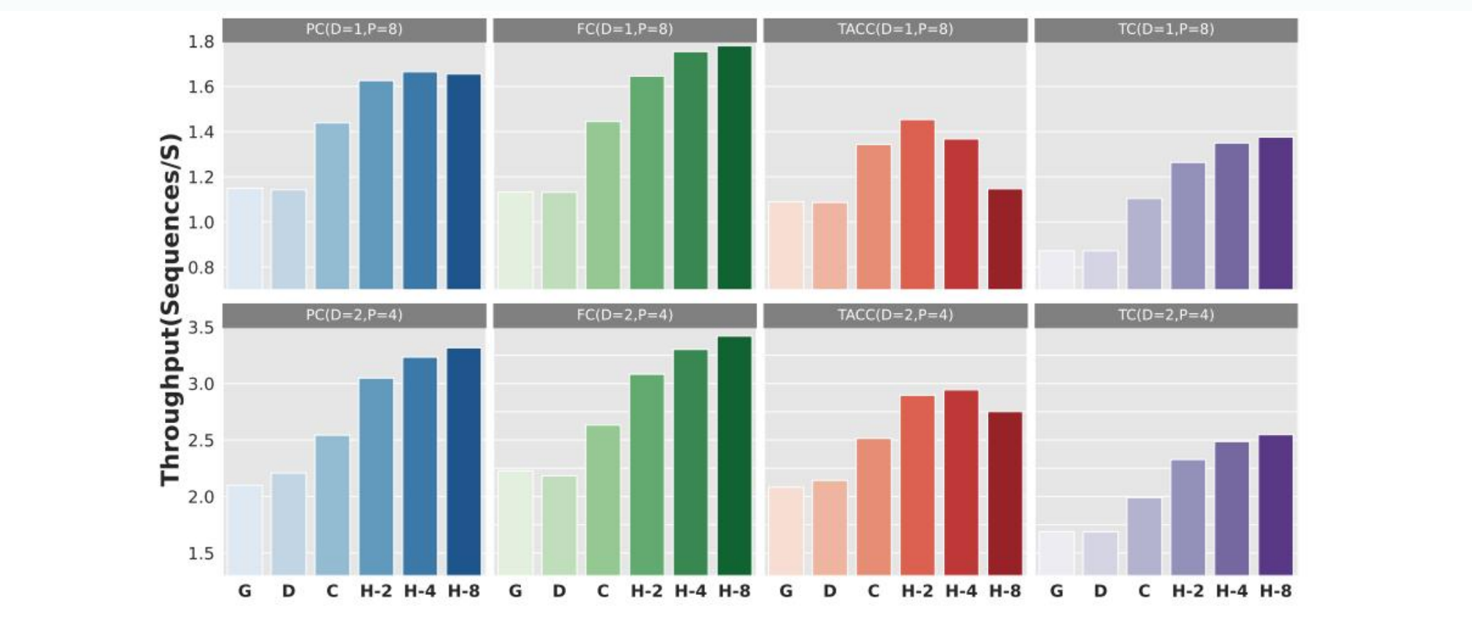


Figure 9: Throughput of training the Bert-style model on totally 32 GPUs from 4 different clusters. PC and FC refer to the two local clusters where the NVIDIA A100 GPUs are partially and fully connected with NVLink. TACC refers to the Lonestar6 cluster from TACC and TC refers to the cloud server of Tencent. As for the methods, G stands for GPipe, D stands for DAPPLE, C stands for Chimera-wave, and H-X stands for Hanayo with X waves.

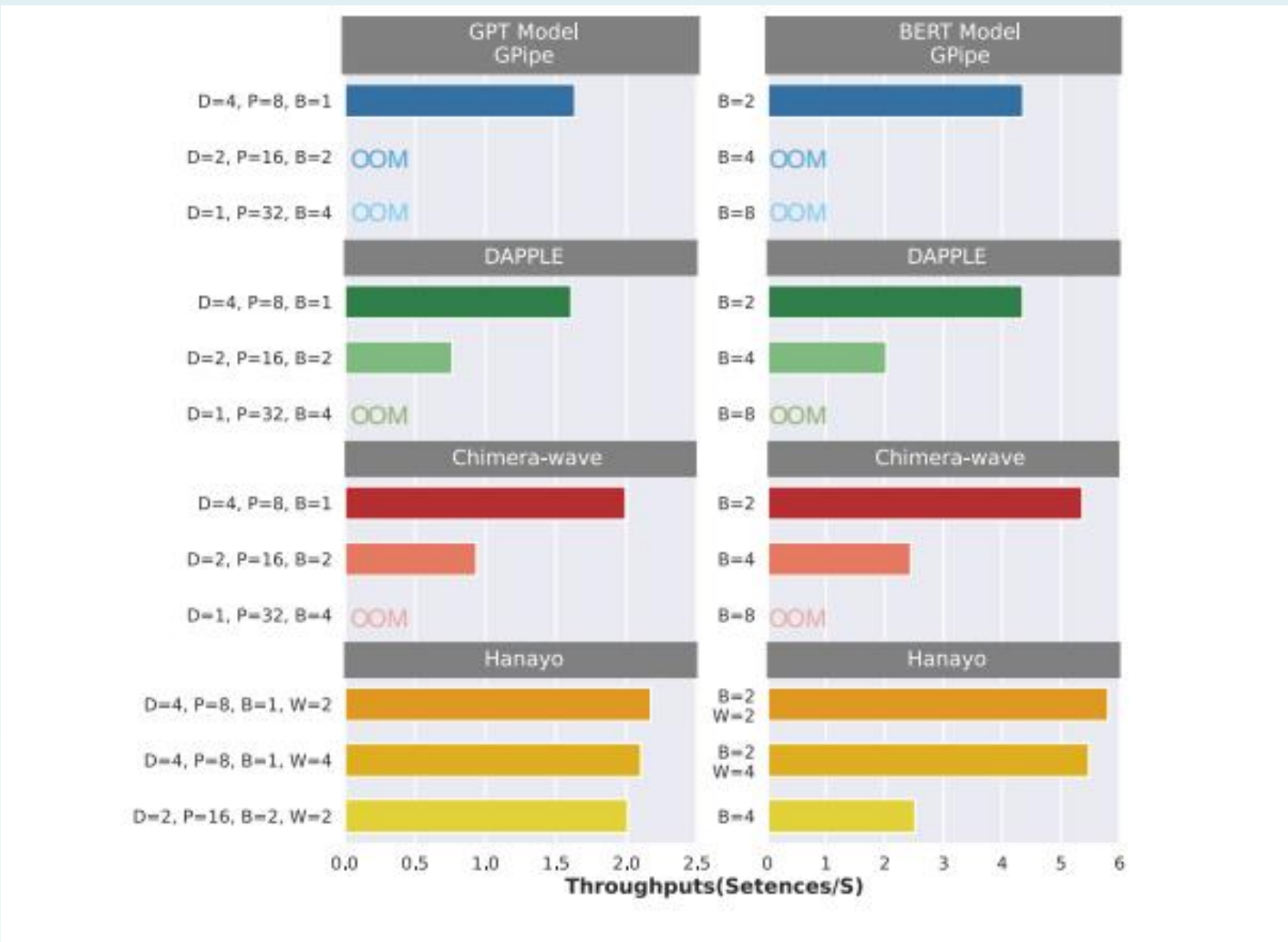


Figure 10: Part of the performance search for the four methods of training the Bert-style model on 32 V100 GPUs from TACC. The configurations with the highest throughput are chosen as targets to be used for further comparison.

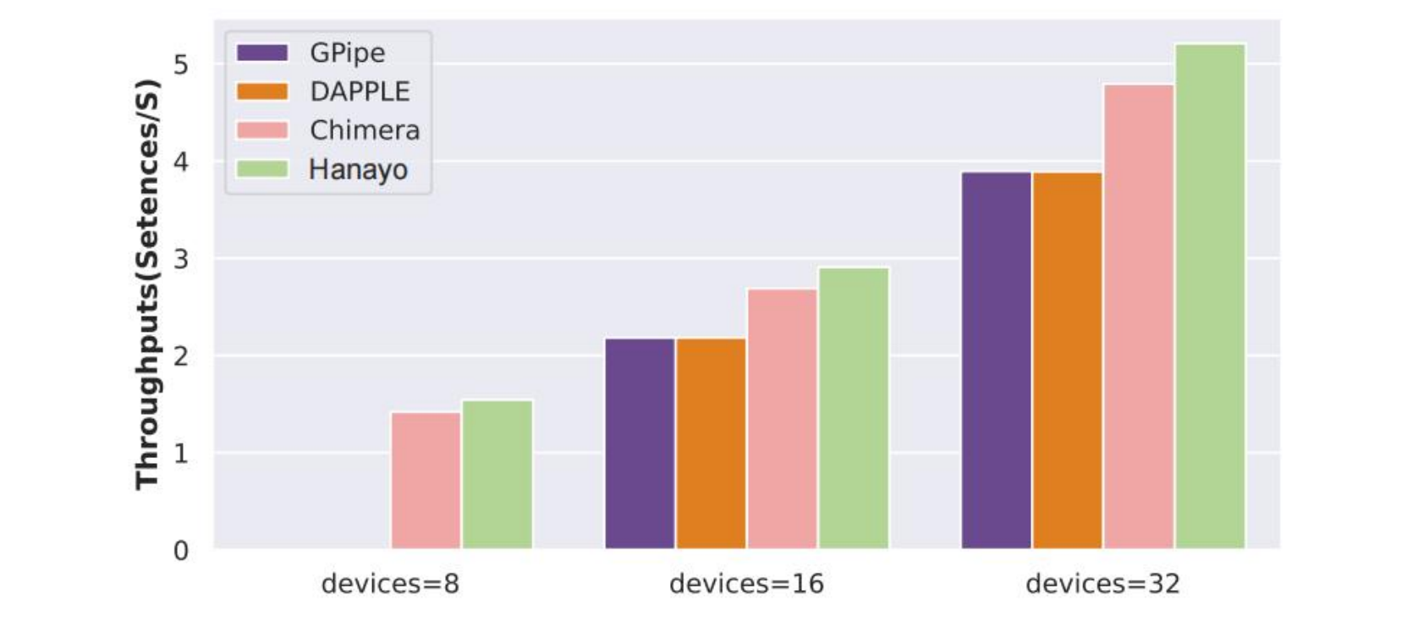


Figure 12: Strong scaling for Bert-style model. We speed up a fixed batch of training with more devices, from 8 to 32.

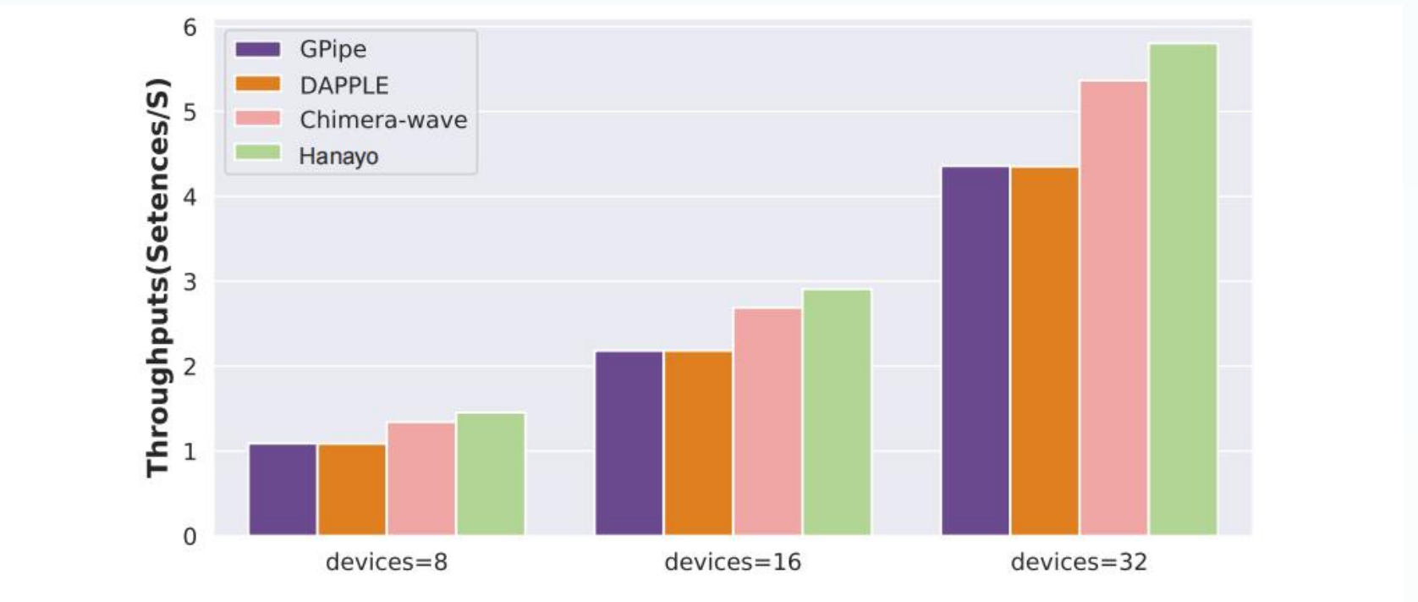


Figure 11: Weak scaling for Bert-style model. The number of devices scales from 8 to 32 while the batch size increases proportionally