MegaScale: Scaling Large Language Model Training to More Than 10,000 GPUs

Ziheng Jiang[1,*] Haibin Lin[1,*] Yinmin Zhong[2,*]
Qi Huang[1] Yangrui Chen[1] Zhi Zhang[1] Yanghua Peng[1] Xiang Li[1] Cong Xie[1] Shibiao Nong[1]
Yulu Jia[1] Sun He[1] Hongmin Chen[1] Zhihao Bai[1] Qi Hou[1] Shipeng Yan[1] Ding Zhou[1] Yiyao
Sheng[1] Zhuo Jiang[1] Haohan Xu[1] Haoran Wei[1] Zhang Zhang[1] Pengfei Nie[1] Leqi Zou[1] Sida
Zhao[1] Liang Xiang[1] Zherui Liu[1] Zhe Li[1] Xiaoying Jia[1] Jianxi Ye[1] Xin Jin[2], Xin Liu[1]

[1]*ByteDance* [2]*Peking University*

ByteDance

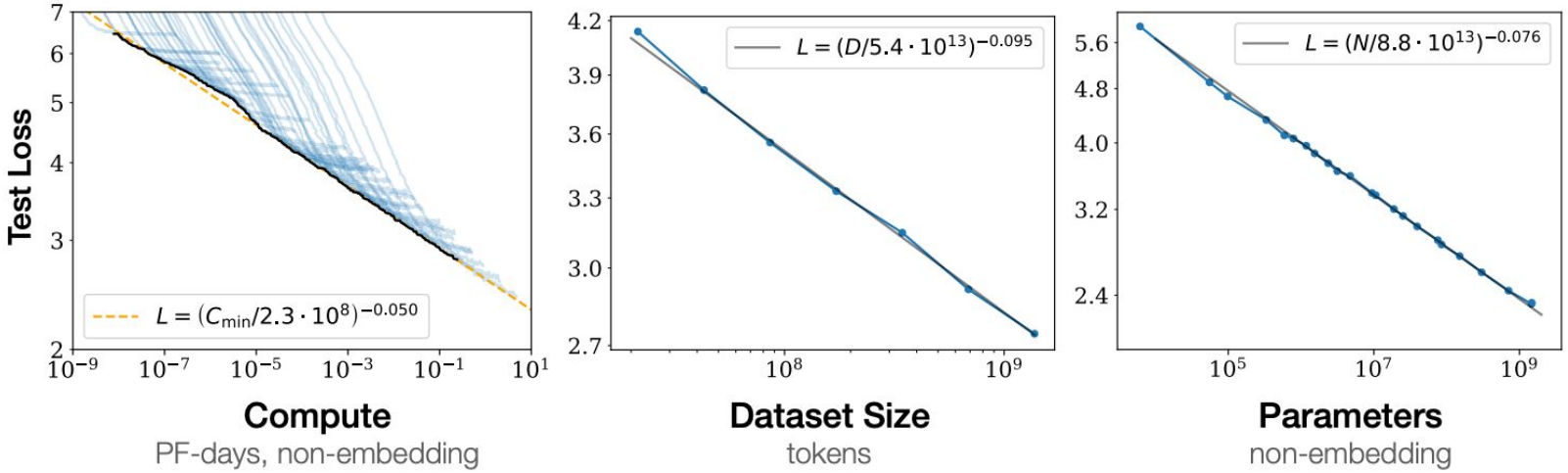
北京大学 PEKING UNIVERSITY

nsdi '24

Large Language Model (LLM) Applications

- conversational agents
- code development
- content creation
- …

The power behind LLMs is scaling



Source: Jared Kaplan, et al. "Scaling Laws for Neural Language Models". In preprint 2020.

Compute required to train an LLM

$$C = \tau T \approx 6ND$$

τ: cluster's effective throughput

T: training time

N: model size (#parameters)

D: dataset size (#tokens)

Source: Jared Kaplan, et al. "Scaling Laws for Neural Language Models". In preprint 2020.

Challenges

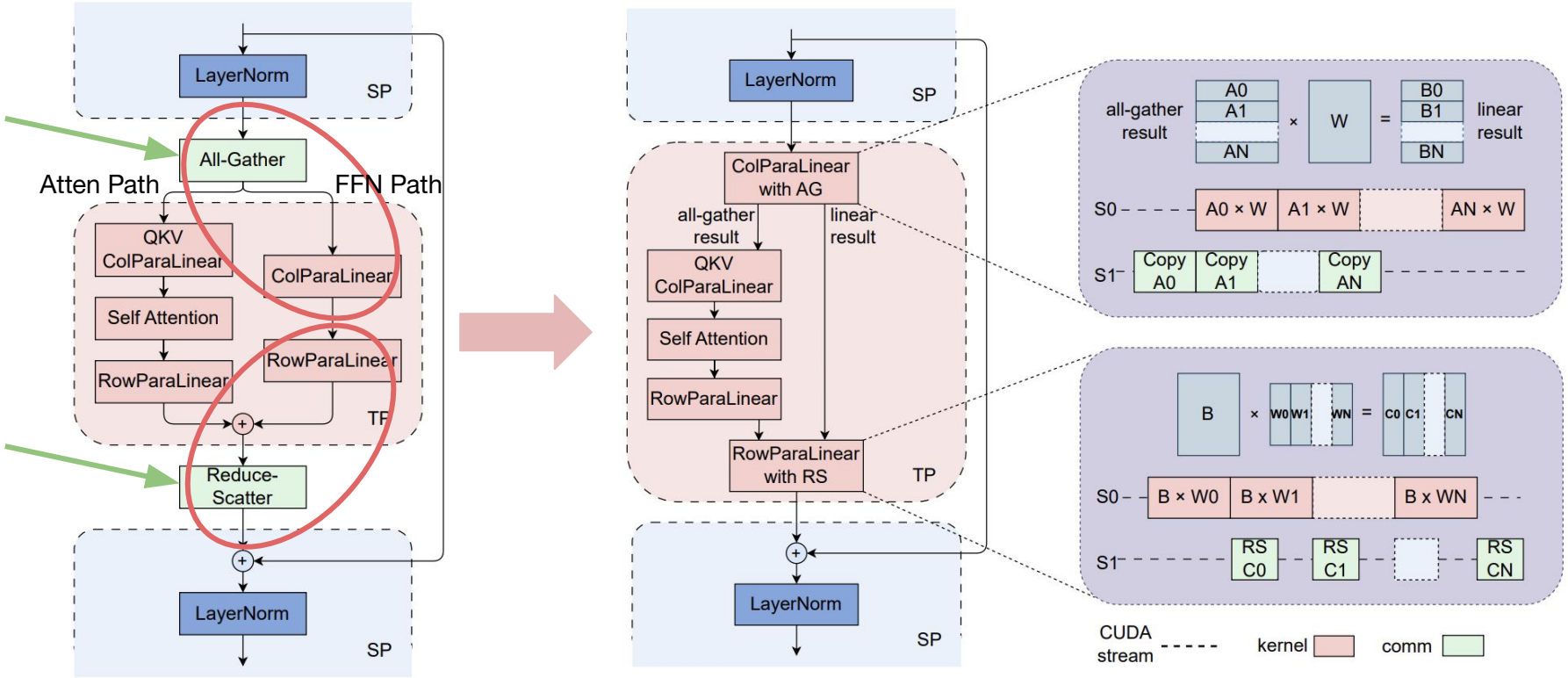
- Achieve high training efficiency at scale
  - LLM training is not embarrassingly parallel
  - communication and many other factors contribute significantly to the efficiency

- Achieve high training stability at scale
  - the trainining duration can extend beyond months
  - failures and stragglers are the norm for LLM training

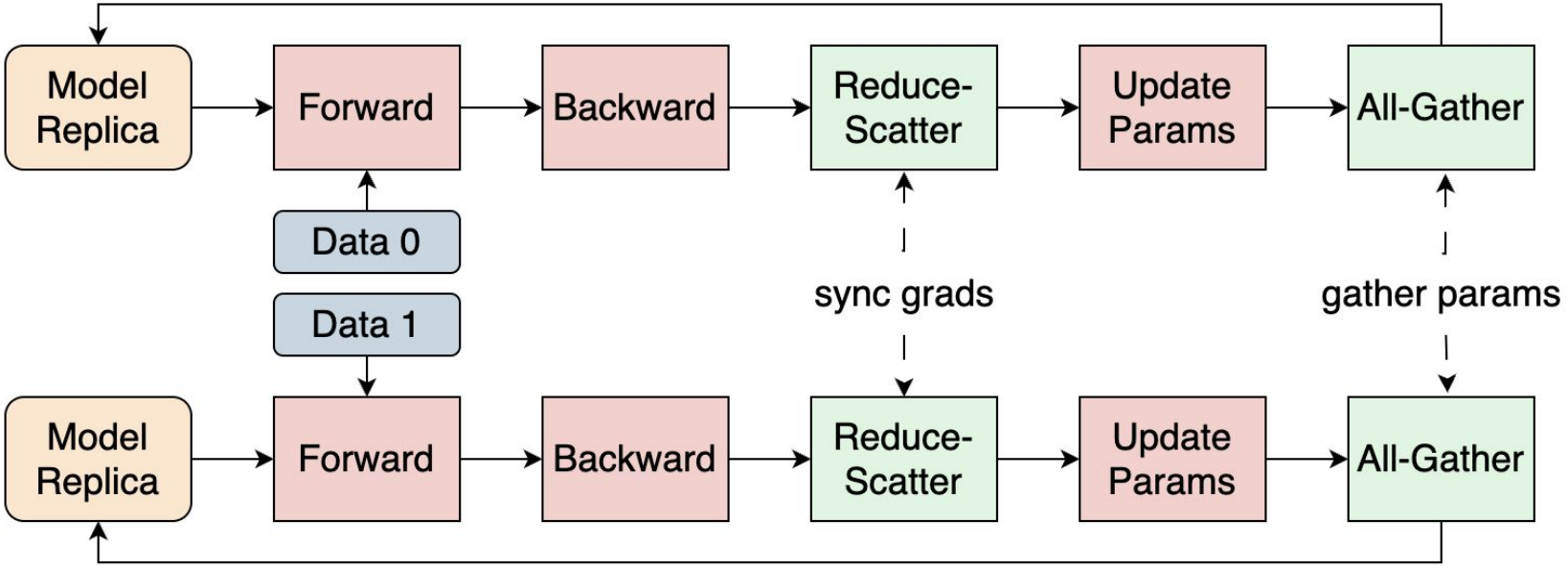
Our solution — MegaScale

- Achieve high training efficiency at scale
  - Algorithmic optimizations
  - Communication overlapping in 3D parallelism
  - Training data preprocessing and loading
  - Collective communication group initialization
  - Network performance tuning
- Achieve high training stability at scale

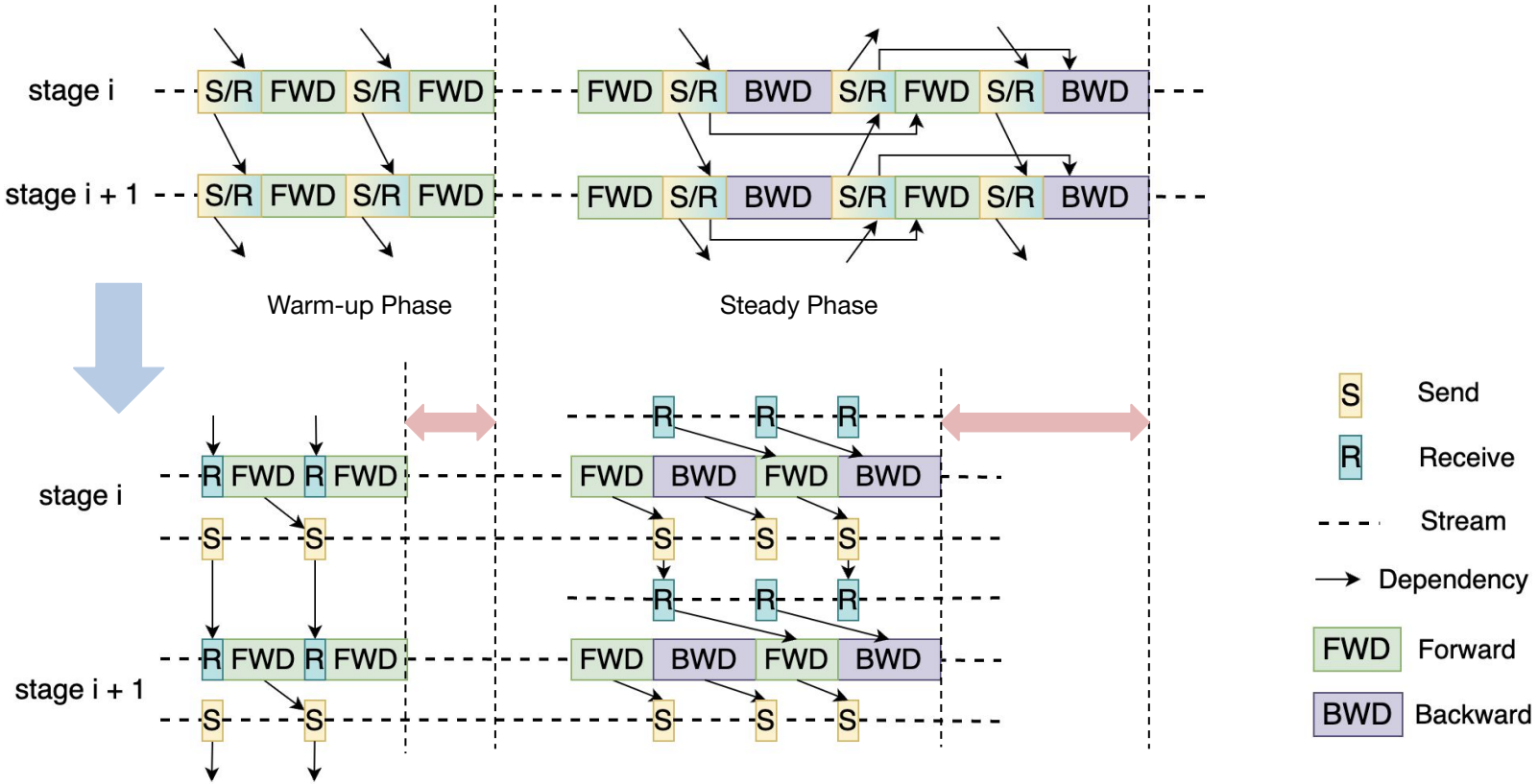
Overlapping in tensor/sequence parallelism



Overlapping in data parallelism (with ZeRO)



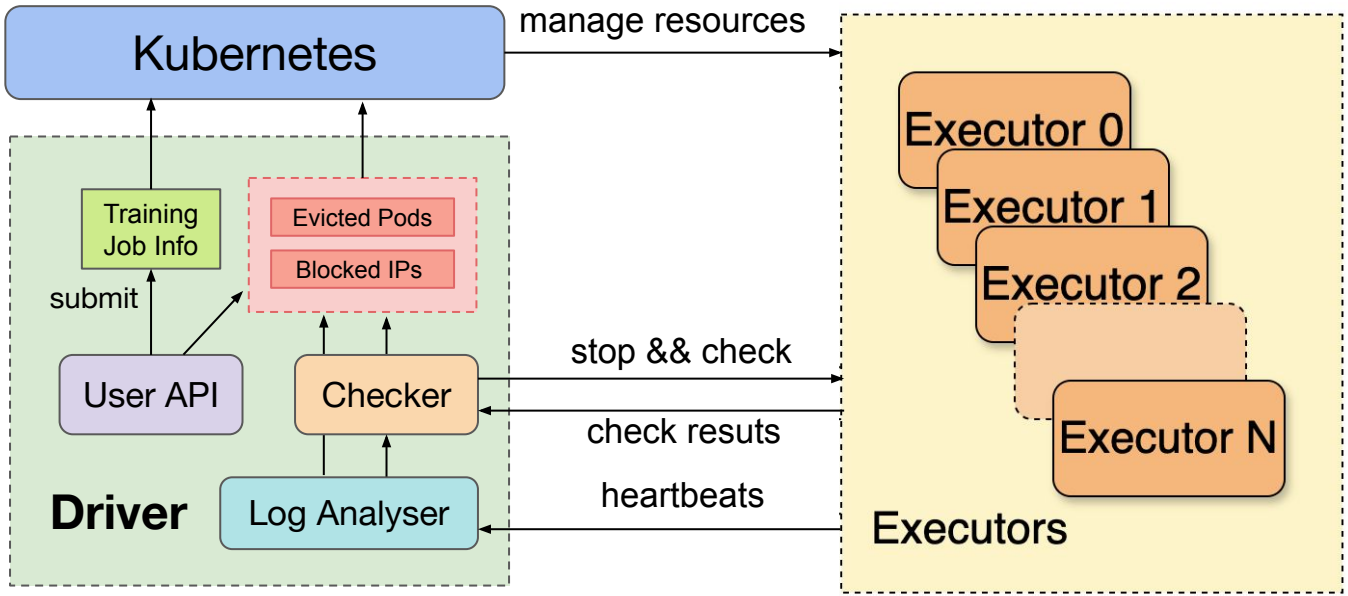
Overlapping in pipeline parallelism



Our solution — MegaScale

- Achieve high training efficiency at scale
  - Algorithmic optimizations
  - Communication overlapping in 3D parallelism
  - Training data preprocessing and loading
  - Collective communication group initialization
  - Network performance tuning
- Achieve high training stability at scale
  - Robust training framework
  - Fast checkpointing and recovery
  - Monitoring and analysis tools

Robust Training Workflow



Deployment Experience

- Training performance
- Training stability
- Problems discovered and fixed

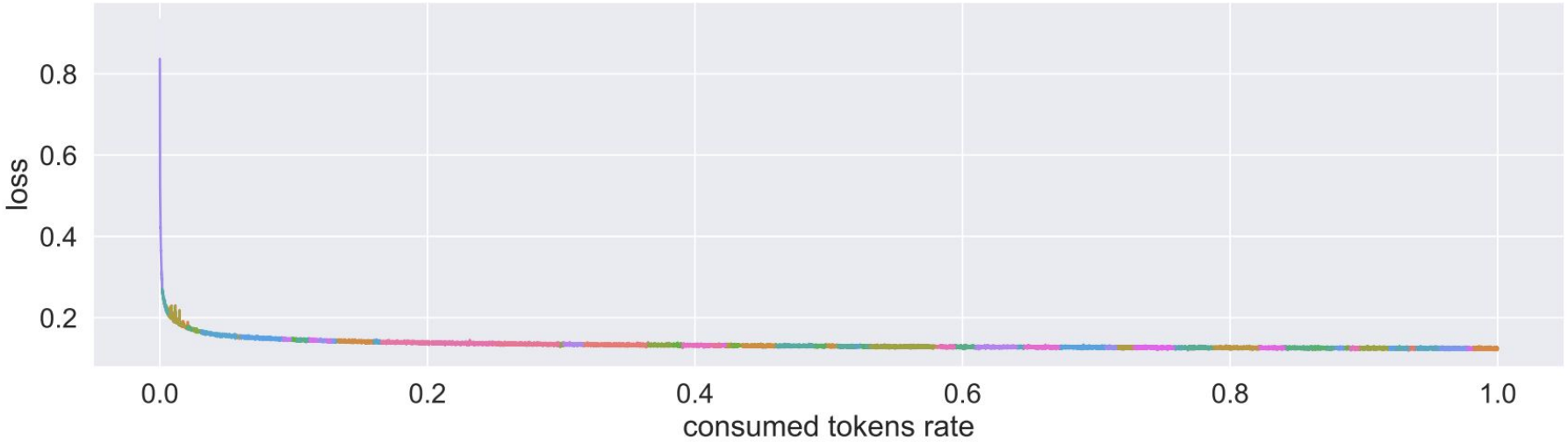
Training Performance

Strong-scaling training performance for the 175B model over 300B tokens compared to Megatron-LM.

Batch Size	Method	GPUs	Iteration Time (s)	Throughput (tokens/s)	Training Time (days)	MFU	Aggregate PFlops/s
768	Megatron-LM	256	40.0	39.3k	88.35	53.0%	43.3
		512	21.2	74.1k	46.86	49.9%	77.6
		768	15.2	103.8k	33.45	46.7%	111.9
		1024	11.9	132.7k	26.17	44.7%	131.9
	MegaScale	256	32.0	49.0k	70.86	65.3%(**1.23**×)	52.2
		512	16.5	95.1k	36.51	63.5%(**1.27**×)	101.4
		768	11.5	136.7k	25.40	61.3%(**1.31**×)	146.9
		1024	8.9	176.9k	19.62	59.0%(**1.32**×)	188.5
6144	Megatron-LM	3072	29.02	433.6k	8.01	48.7%	466.8
		6144	14.78	851.6k	4.08	47.8%	916.3
		8192	12.24	1027.9k	3.38	43.3%	1106.7
		12288	8.57	1466.8k	2.37	41.2%	1579.5
	MegaScale	3072	23.66	531.9k	6.53	59.1%(**1.21**×)	566.5
		6144	12.21	1030.9k	3.37	57.3%(**1.19**×)	1098.4
		8192	9.56	1315.6k	2.64	54.9%(**1.26**×)	1400.6
		12288	6.34	1984.0k	**1.75**	55.2%(**1.34**×)	2166.3

Training stability

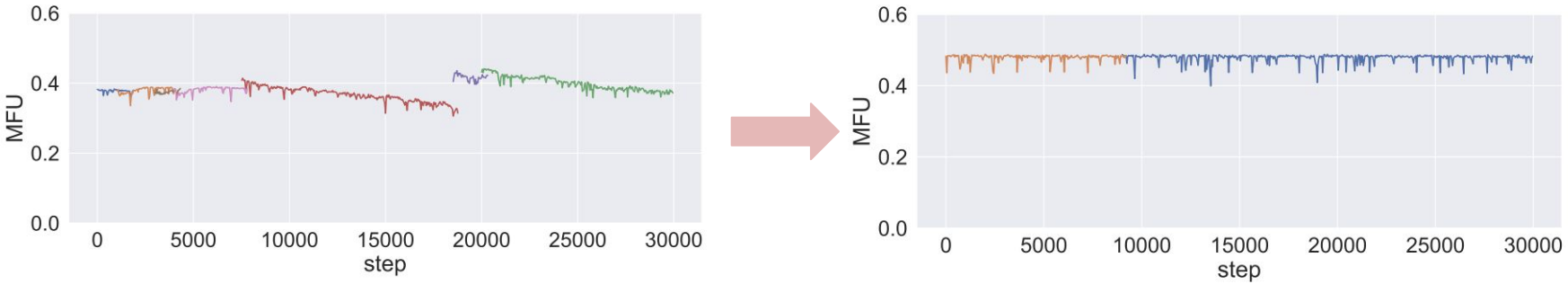
The loss curve of a real production run that trains a proprietary model with hundreds of billions of parameters on multi-trillion tokens.



Problems discovered and fixed

- Computational stragglers

- MFU decreasing



Summary of MegaScale

- Achieve high training efficiency at scale
  - Algorithmic optimizations
  - Communication overlapping in 3D parallelism
  - Training data preprocessing and loading
  - Collective communication group initialization
  - Network performance tuning
- Achieve high training stability at scale
  - Robust training framework
  - Fast checkpointing and recovery
  - Monitoring and analysis tools

veScale

github.com/volcengine/veScale

Large Language Models (LLMs)