富岳を用いた大規模言語モデルの分散並列学習

Distributed Training of Large Language Models on Fugaku

Tokyo Institute of Technology
GSIC
Rio Yokota

2022年度第2回計算科学フォーラム

2023年3月28日(火)

Collaborators

GPT-Fugaku Team

Noriyuki Kojima

Kazuto Ando

Koji Nishiguchi

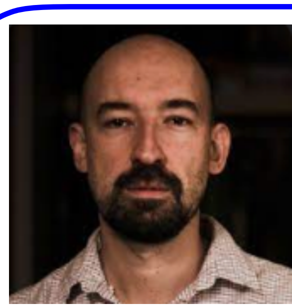
Jungo Kasai

Keisuke Sakaguchi

Shukai Nakamura

GPT FUGAKU
EST. 2021

DL4Fugaku Team @ R-CCS

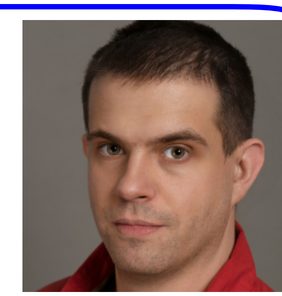


Aleksandr Drozd

Mohamed Wahib

Kento Sato

Jens Domke

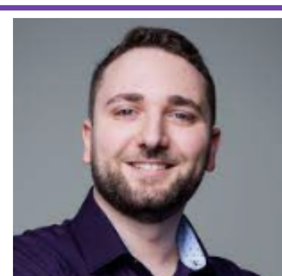


Emil Vatai

R-CCS

DL4Fugaku Team @ LLNL

Nikoli Dryden



Tal Ben Nun

Fujitsu

Koichi Shirahata

FUJITSU

Large Language Models

Fastest growing app in history

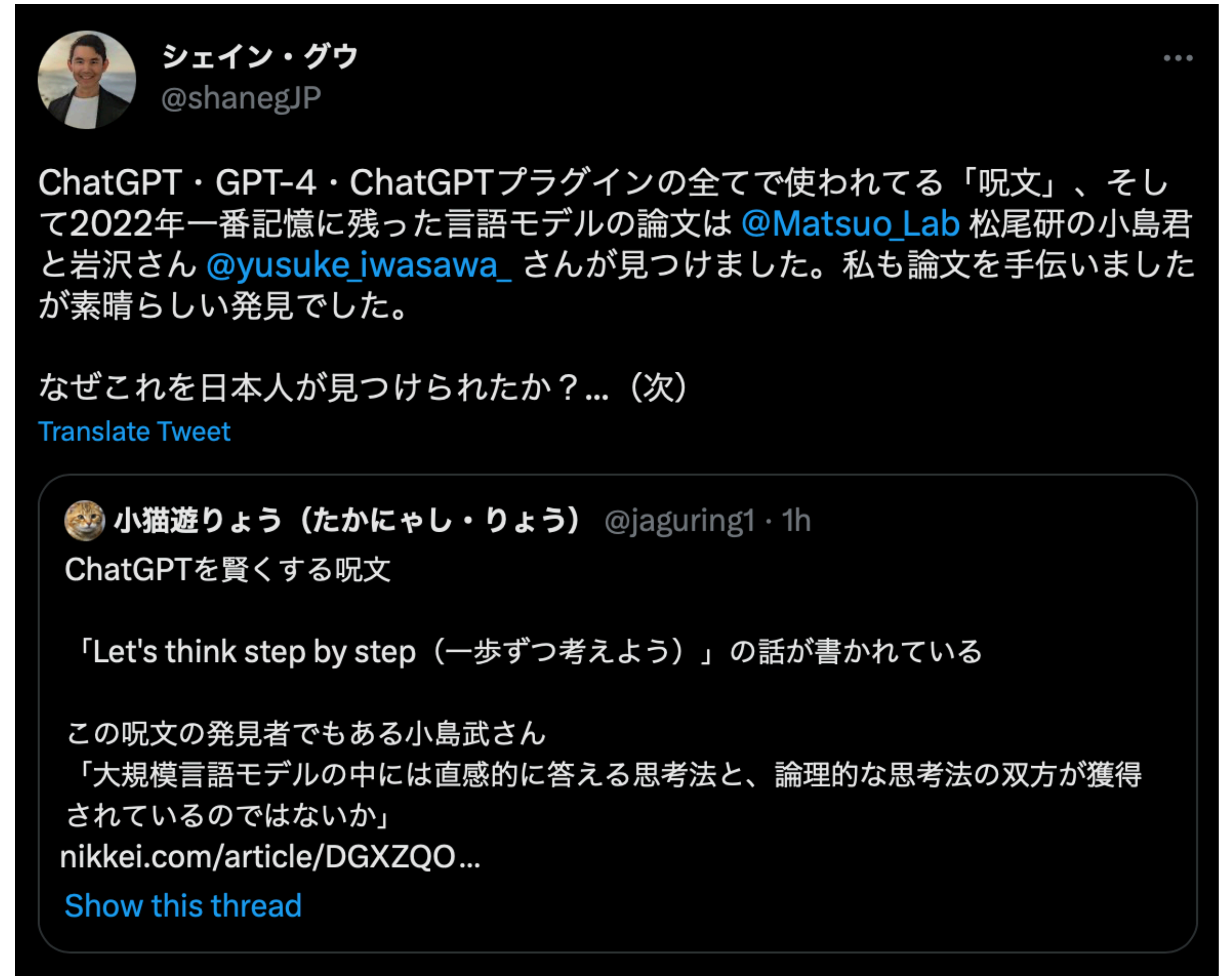
Platform	Time to first 1 million users
ChatGPT	5 days
Facebook	10 months
Instagram	2 months
Spotify	5 months
Netflix	3.5 years

https://nerdynav.com/chatgpt-statistics/

Will change the landscape of …

- Web search
- Machine translation & summarization
- Creative writing in research & education
- Coding & Debugging

Chain of Thought Prompting


シェイン・グウ
@shanegJP

ChatGPT・GPT-4・ChatGPTプラグインの全てで使われてる「呪文」、そして2022年一番記憶に残った言語モデルの論文は @Matsuo_Lab 松尾研の小島君と岩沢さん @yusuke_iwasawa_ さんが見つけました。私も論文を手伝いましたが素晴らしい発見でした。

なぜこれを日本人が見つけられたか？…（次）

Translate Tweet

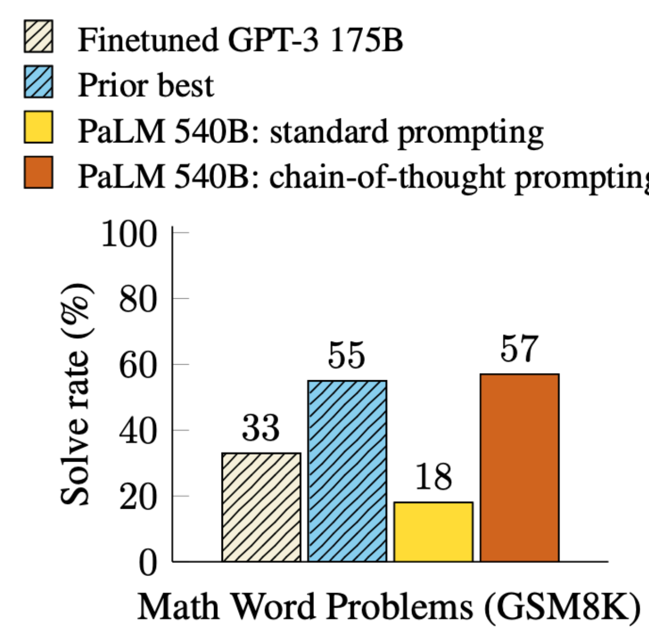
小猫遊りょう（たかにゃし・りょう） @jaguring1 · 1h

ChatGPTを賢くする呪文

「Let's think step by step（一歩ずつ考えよう）」の話が書かれている

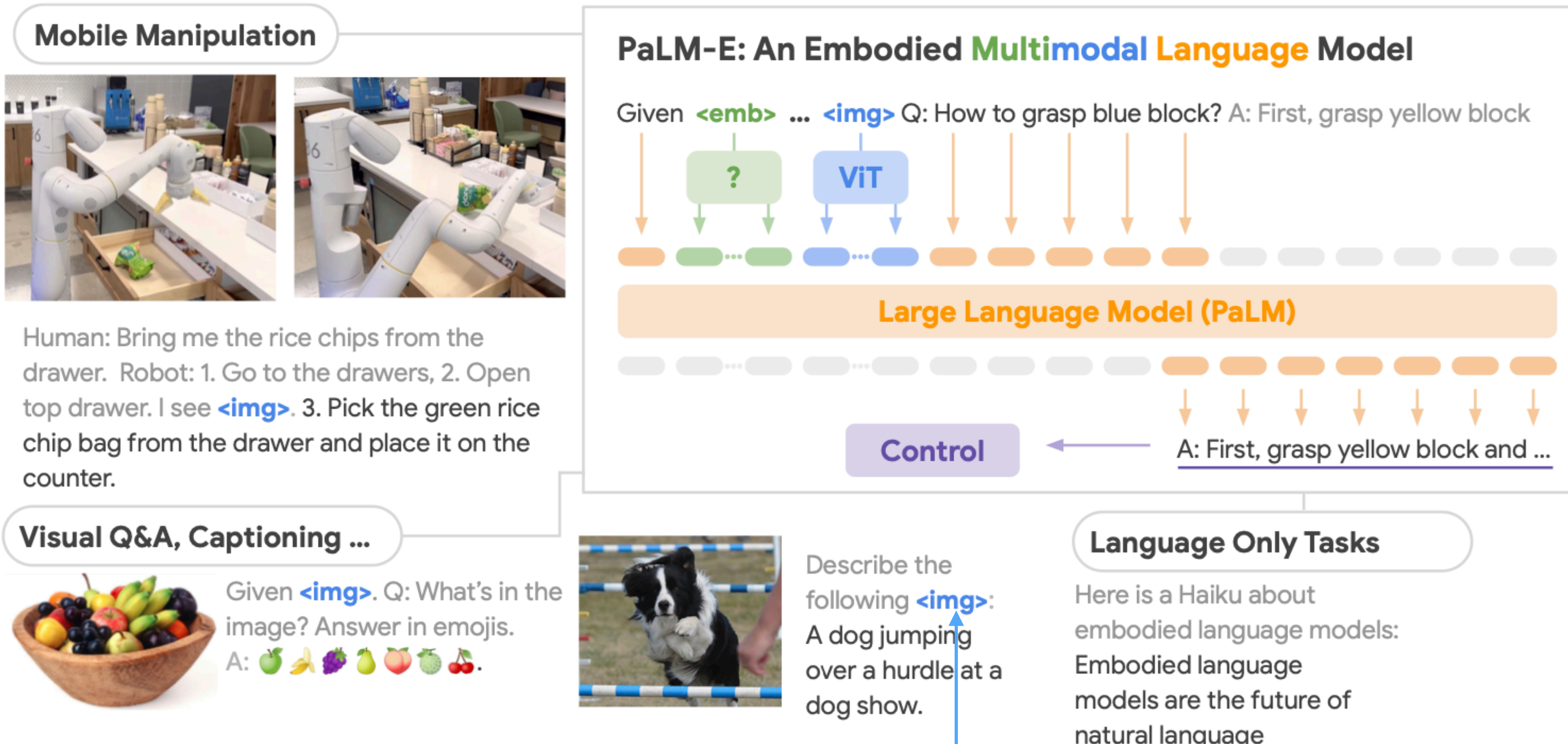
この呪文の発見者でもある小島武さん
「大規模言語モデルの中には直感的に答える思考法と、論理的な思考法の双方が獲得されているのではないか」
nikkei.com/article/DGXZQO…

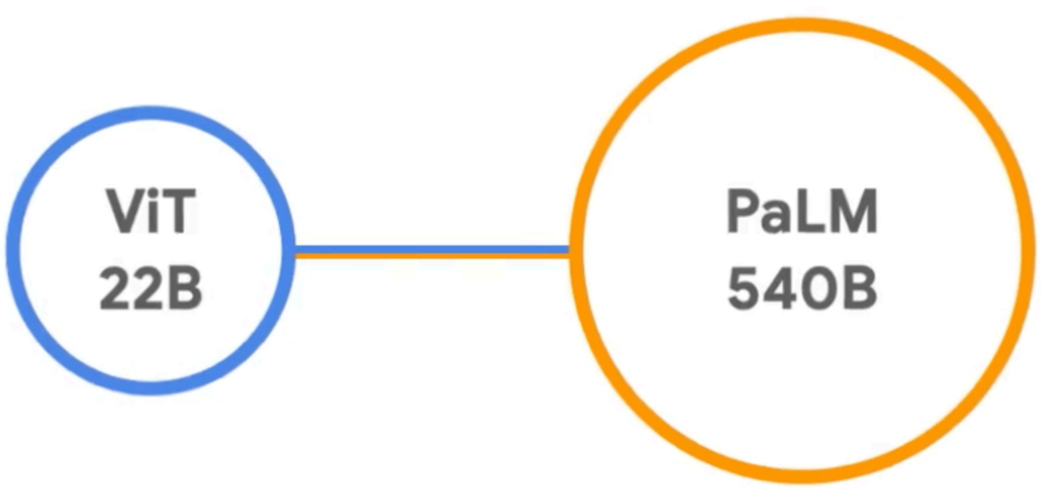
Show this thread



https://arxiv.org/abs/2201.11903

Multimodal Language Models



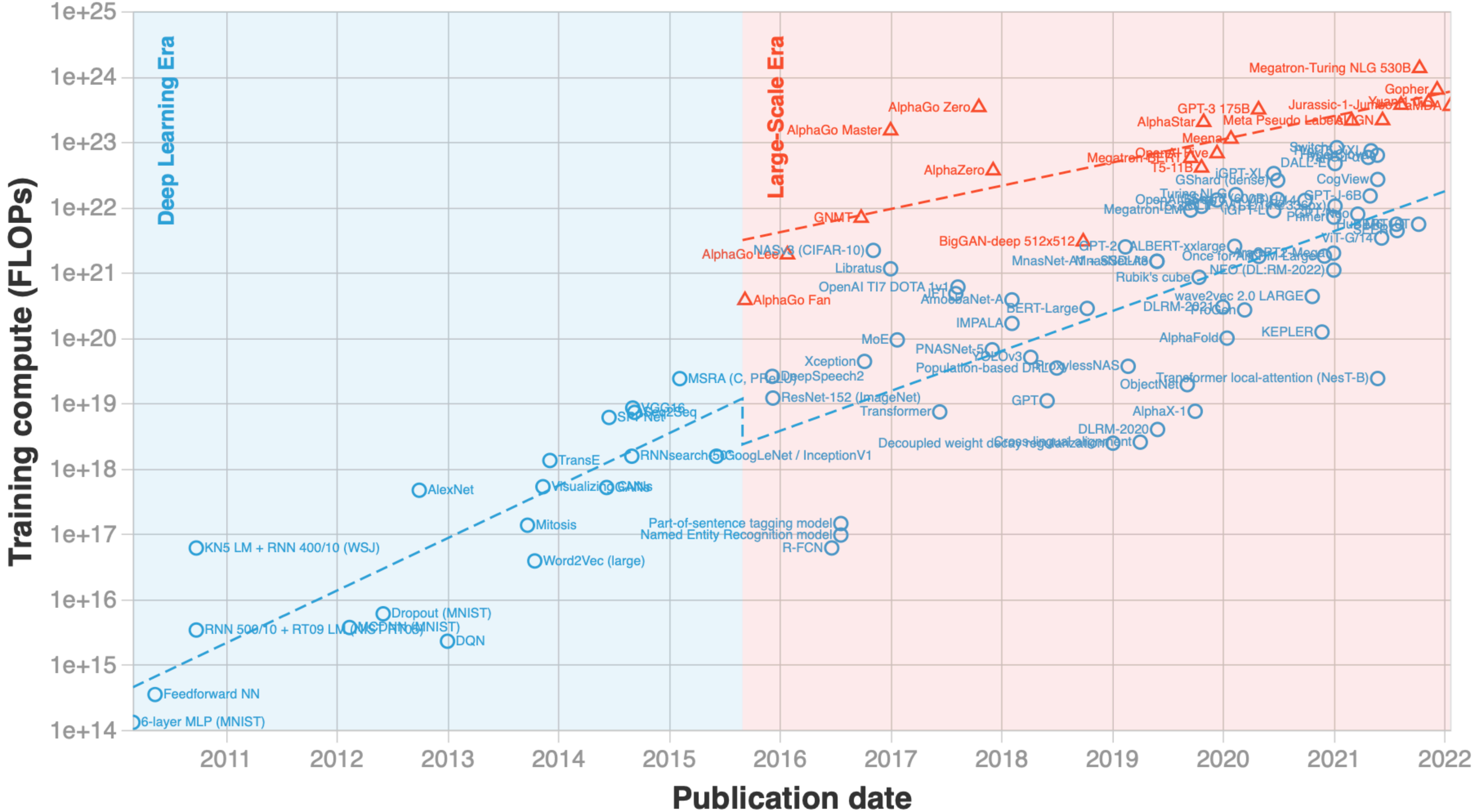


Will change the landscape of …

- Robotics & Autonomous driving
- Medicine & Science
- Explainable AI

PaLM-E: An Embodied Multimodal Language Model, https://arxiv.org/abs/2303.03378

Deep Learning Scaling Law

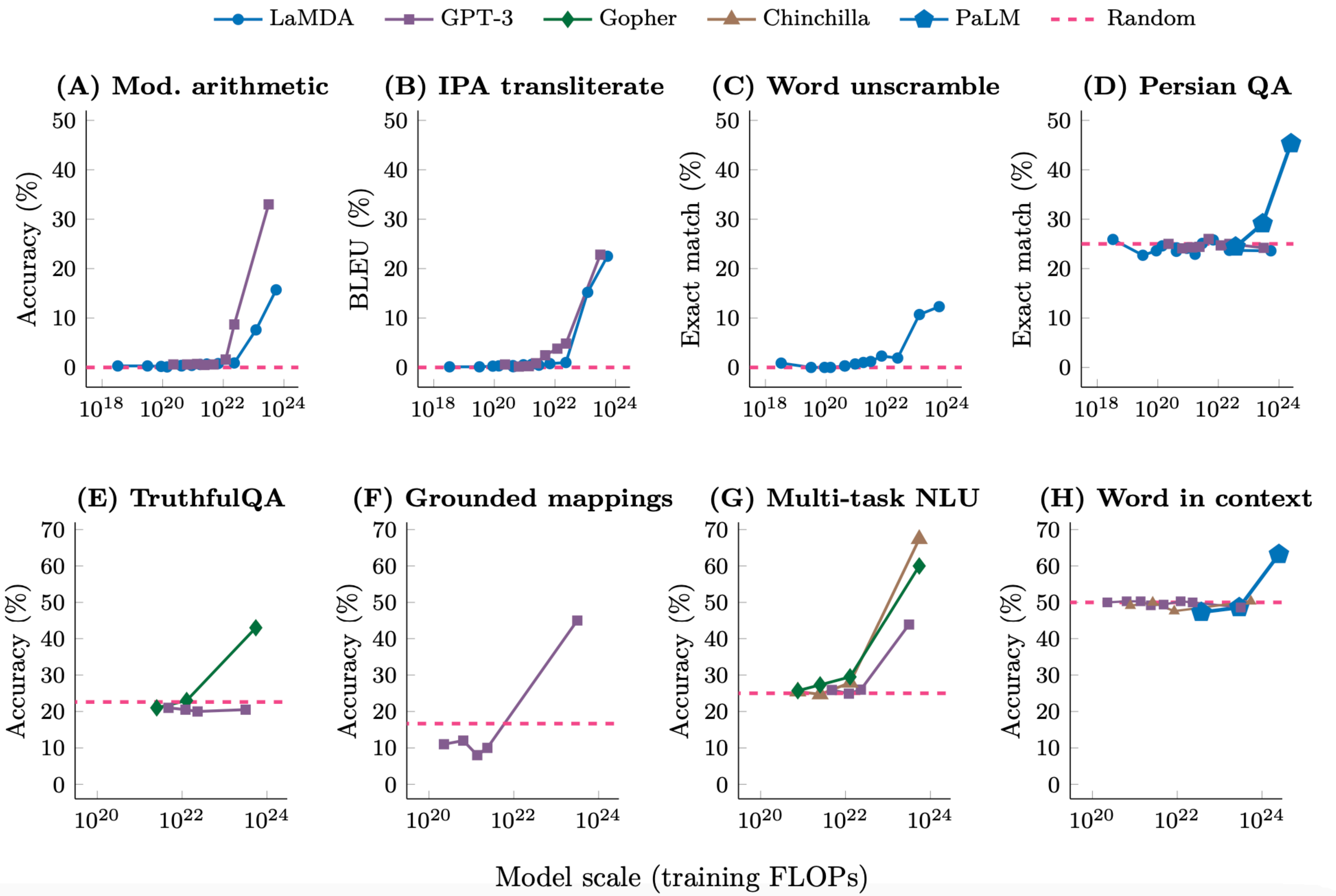


Compute Trends Across Three Eras of Machine Learning, https://arxiv.org/abs/2202.05924

Moore's law: 1970-2020の50年で10^7倍

DL Scaling law: 2010-2022の12年で10^{10}倍

Emergent Abilities at 10^{22}-10^{24} FLOPs



Emergent Abilities of Large Language Models, https://arxiv.org/abs/2206.07682

How Long It Will Take to Train GPT

GPT-4: 3×10^{25}FLOPs (speculated)
GPT-3.5 (ChatGPT): 3×10^{24}FLOPs (speculated)
GPT-3: 3×10^{23}FLOPs

Fugaku:
FP32 6.76TFLOP/s x 158,976 = 1.07 EFLOP/s (theoretical peak)
GPT-4: 328 days x 10
GPT-3.5: 32 days x 10
GPT-3: 3.3 days x 10

OpenAI:
BF16 312 TFLOP/s x 25,000 = 7.8 EFLOP/s (theoretical peak)
GPT-4: 45 days x 2
GPT-3.5: 4.5 days x 2
GPT-3: 11 hours x 2

Actual Performance

https://lifearchitect.ai/gpt-4/#size

Cost of GPT vs Weather Simulation

	GPT-4	3.5Km Global Weather Simulation
Description	~ 1 Trillion parameters ~ 450 Billion Tokens	4.4 Trillion Grid Points w/ Data Assimilation
Goal	Training (Note: inference cost is also immense)	9 hours Simulation
Precision	16FP; Bfloat 32; FP32	Double precision Mixed precision (FP64+FP32)
Resources Time (end-to-end: inc. I/O) CPU/GPU-hours	25,000 Nvidia A100 GPUs 90 days 54,000,000 GPU-hours ~1625x	131,072 Fugaku Nodes (A64FX) 14,200 seconds 33,230 CPU-hours
Compute per token (LLM) or grid point (Gloal Weather Sim.)	Training: ~ 6x model parameters → 6 TFLOPS Inference: ~ 2x model parameters → 2 TFLOPS	~ 90 MFLOPS ~22,222x
Notes:	- 10s of attempts to get to production training - Full training might be needed to observe impovement	- Incrementaly increase resolution

A 1024-Member Ensemble Data Assimilation with 3.5-Km Mesh Global Weather Simulations, SC 2020 (Gordon Bell Finalist)

provided by Mohamed Wahib, R-CCS

A Gordon Bell submissions will use a few hours of the whole Fugaku

Training GPT-4 will take a year on the whole Fugaku even with its FP32 peak

Related Projects

ABCI GC FY2019: Second Order Optimization
Trained ImageNet in less than 2 minutes with 131k batch size (2048 GPUs)

HPCI FY2020: Minimizing Memory Footprint
Reduced memory consumption to allow training of 14B parameter model on a single node

HPCI FY2021: Minimizing I/O
Training directly from tar files, reduced I/O latency and inodes by 1/1000

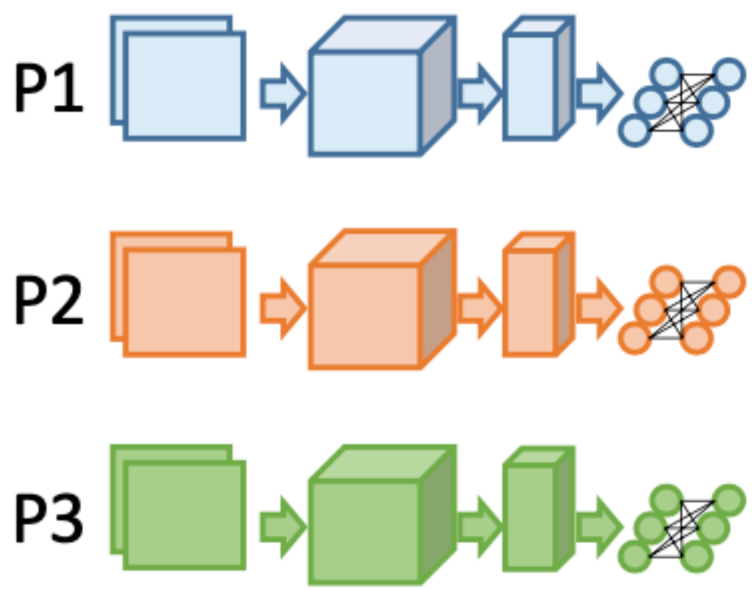
HPCI FY2022: Training Vision Transformers on Synthetic Datasets
Surpassed the accuracy of ImageNet-21k using a purely synthetic dataset

HPCI FY2023: Performance Optimization of Transformers on A64FX and Their Application to Vision & Language

INCITE FY2023: Large Vision+Language Models on Summit/Frontier
6M GPU hours allocated (PI: Irina Rish, U. Montreal)

Distributed Training

Data Parallel (DP)



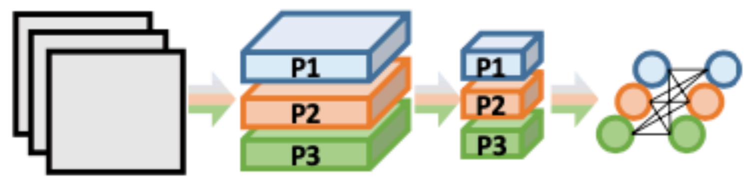
Distributed Data

Redundant Model

AllReduce Gradients

Large batch problem

Tensor Parallel (TP) or ZeRO/FSDP



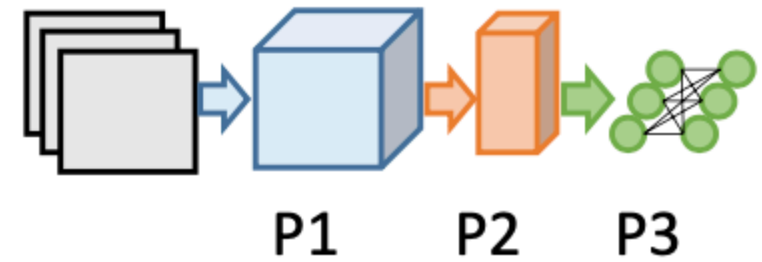
Redundant Data

Distributed Model

AllReduce Activations
or
AllGather Parameters

Frequent Communication

Pipeline Parallel (PP)



Distributed Data

Distributed Model

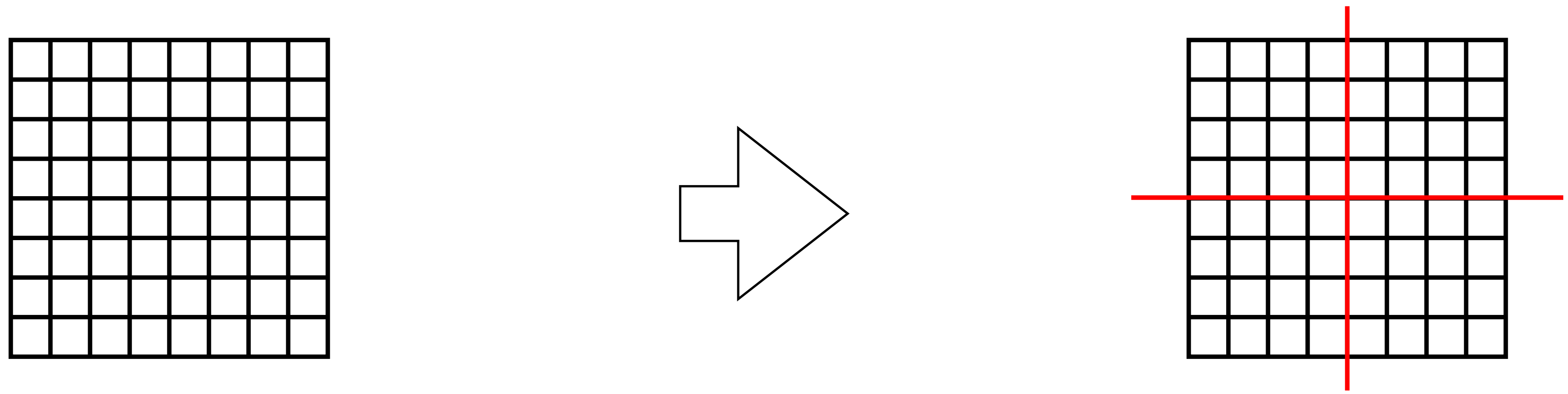
SendRecv Activations

Pipeline bubble

"Demystifying Parallel and Distributed Deep Learning: An In-Depth Concurrency Analysis", Ben-nun and Hoefler, ACM Computing Surveys, Article No.: 65

What is Strong Scaling in Deep Learning?

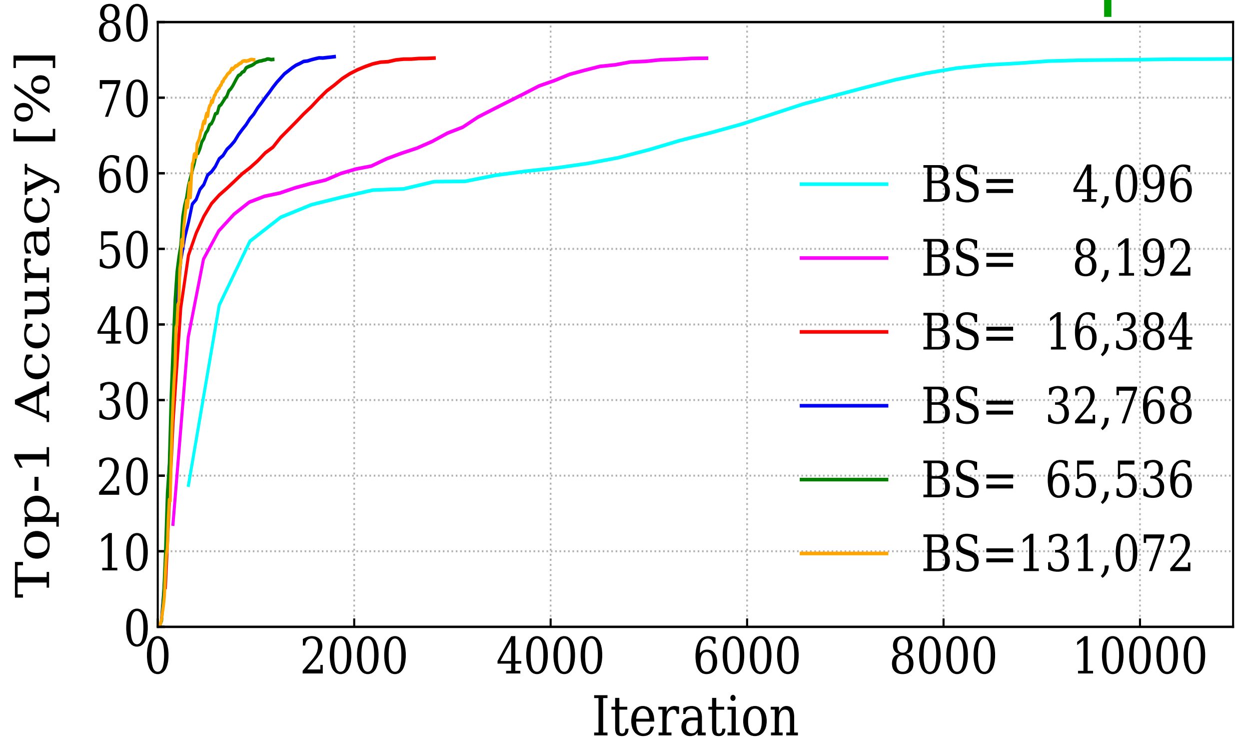
Scientific Computing

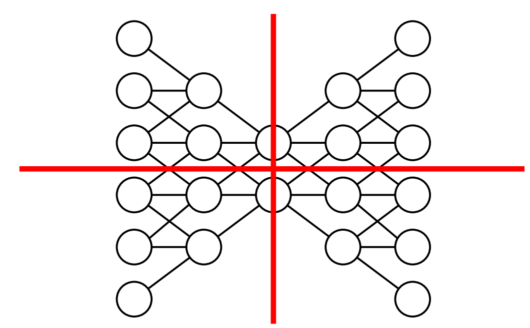


Given a certain mesh size

Reduce solution time by partitioning

Deep Learning

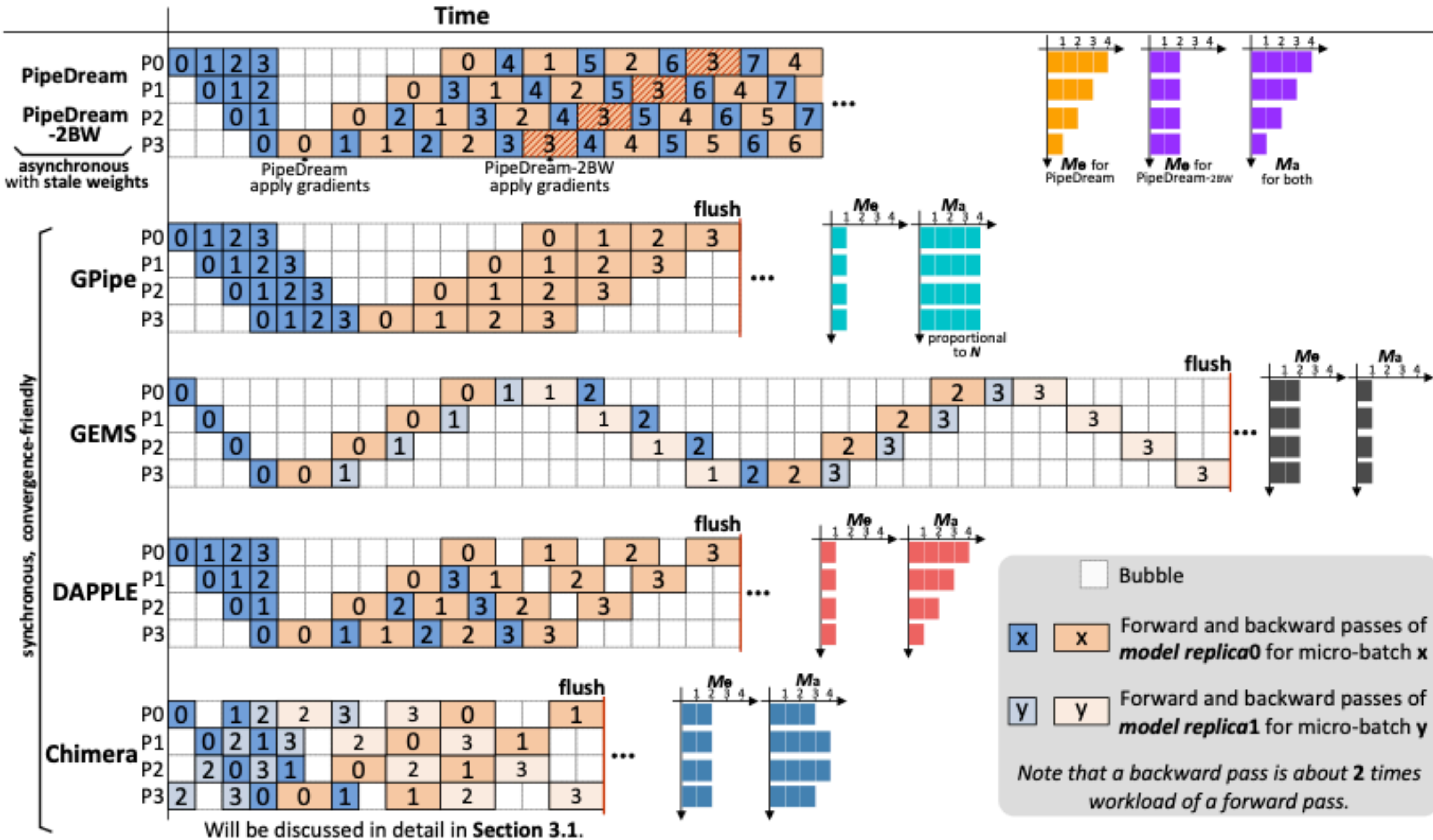




Reduce time per step by partitioning

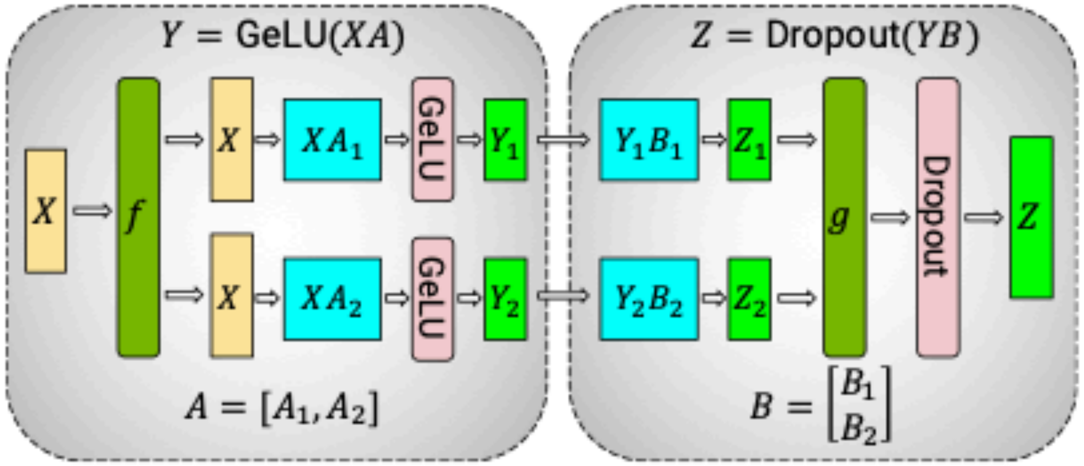
Reduce number of steps by partitioning

Pipeline Parallel

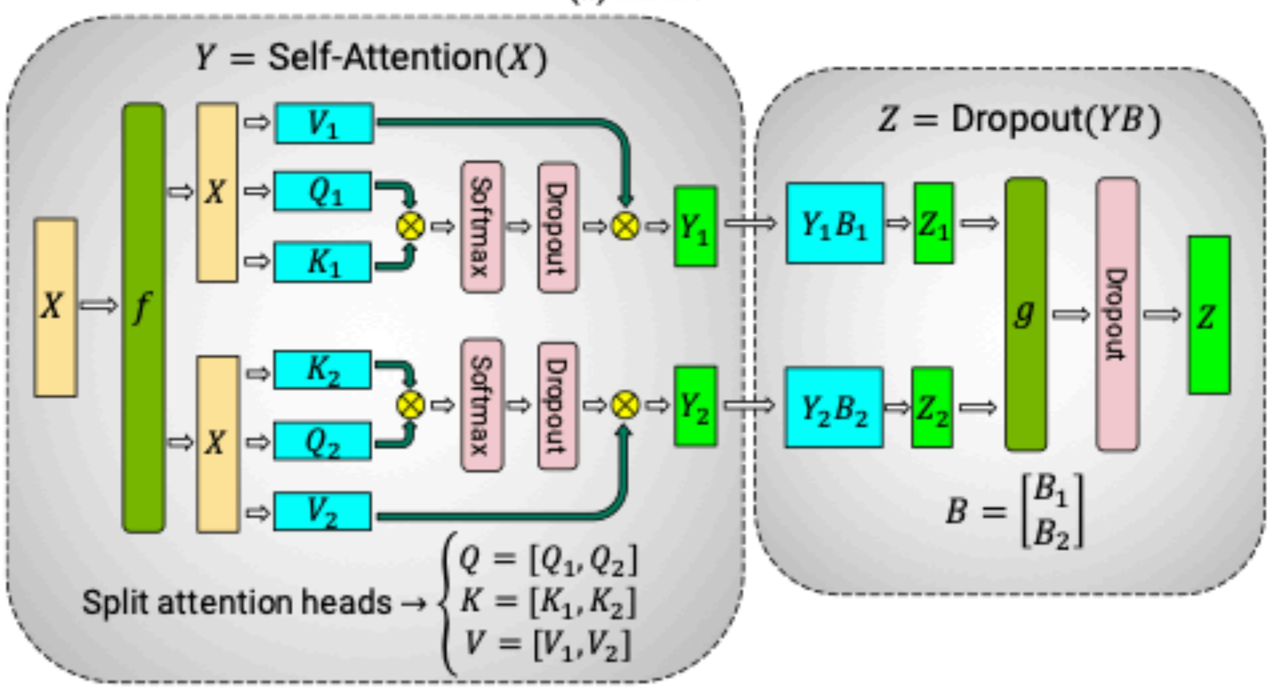


Chimera: Efficiently Training Large-Scale Neural Networks with Bidirectional Pipelines, https://arxiv.org/abs/2107.06925

Tensor Parallel



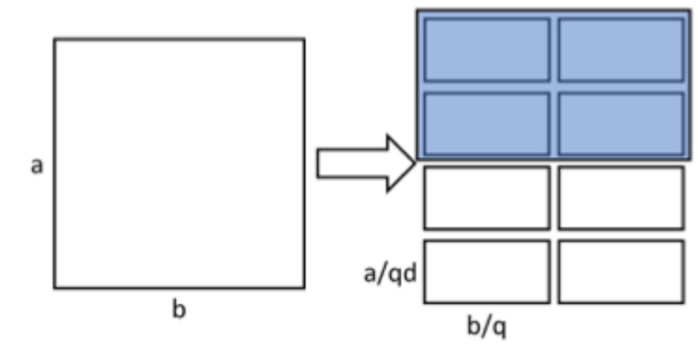
(a) MLP.



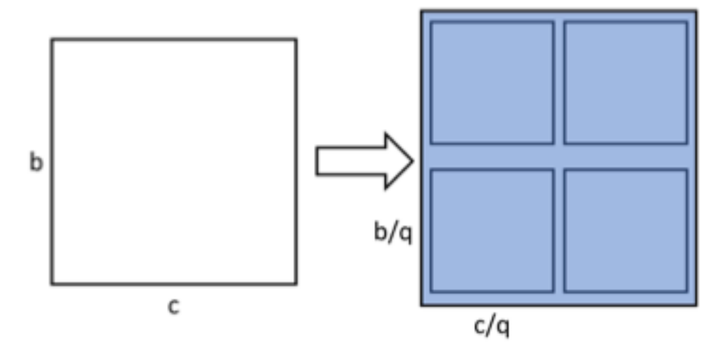
(b) Self-Attention.

Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM
https://arxiv.org/abs/2104.04473

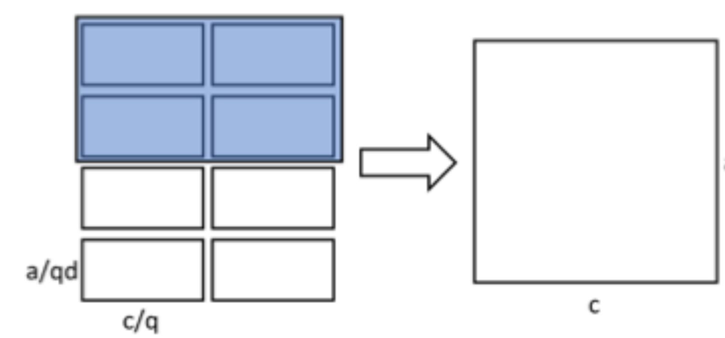
Apply SUMMA to Attention Layer



(a) Partition of matrix A



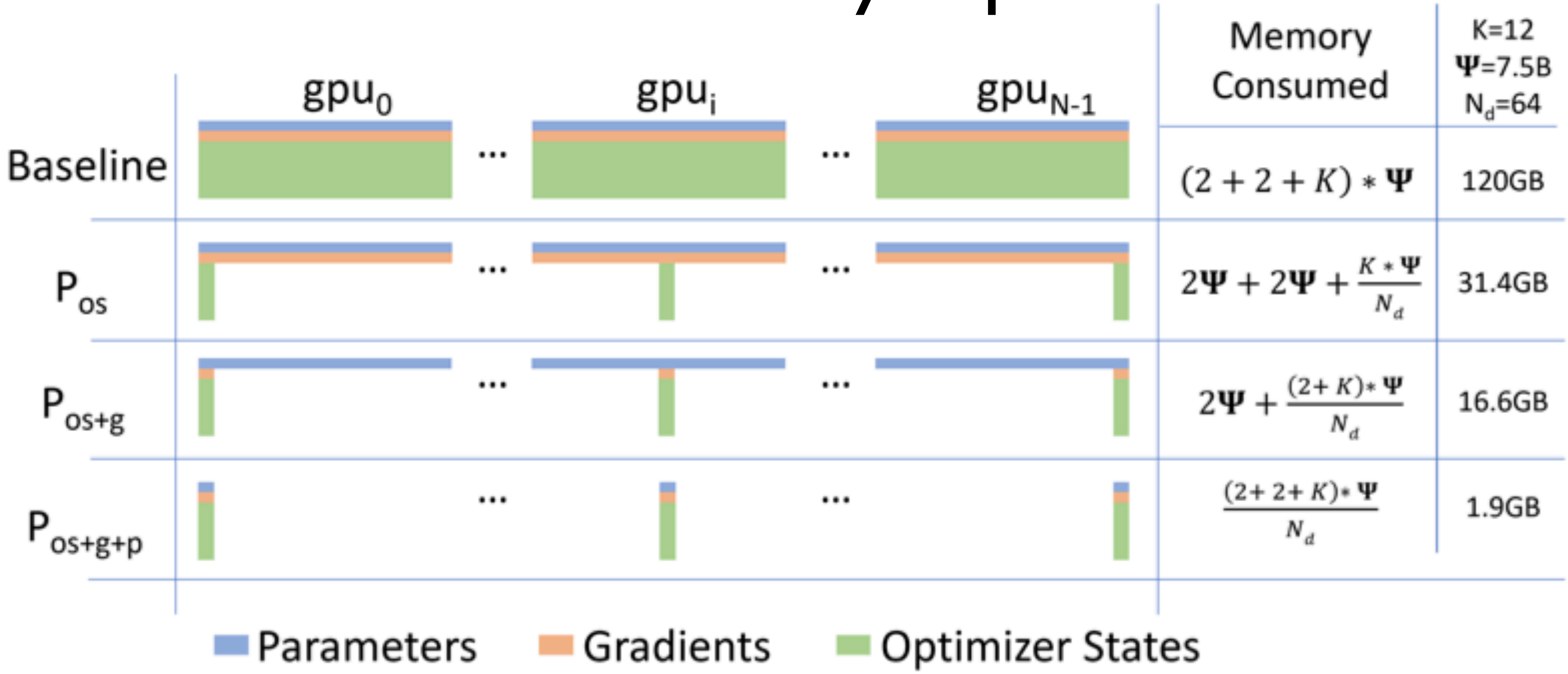
(b) Partition of matrix B

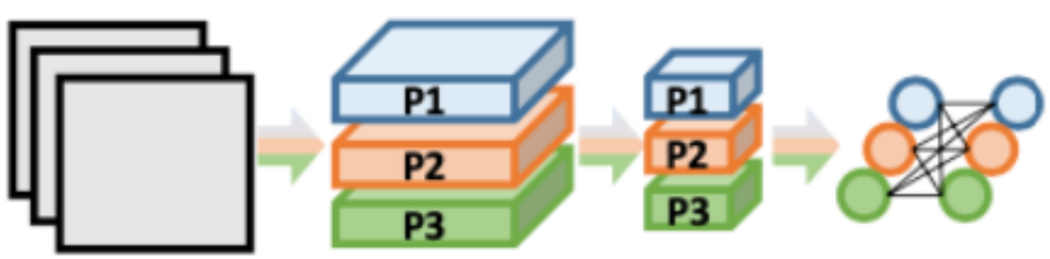


(c) Combination of matrix C

Tesseract: Parallelize the Tensor Parallelism Efficiently
https://arxiv.org/abs/2105.14500

Zero Redundancy Optimizer





AllGather Parameters and Optimizer States

ReduceScatter Gradients

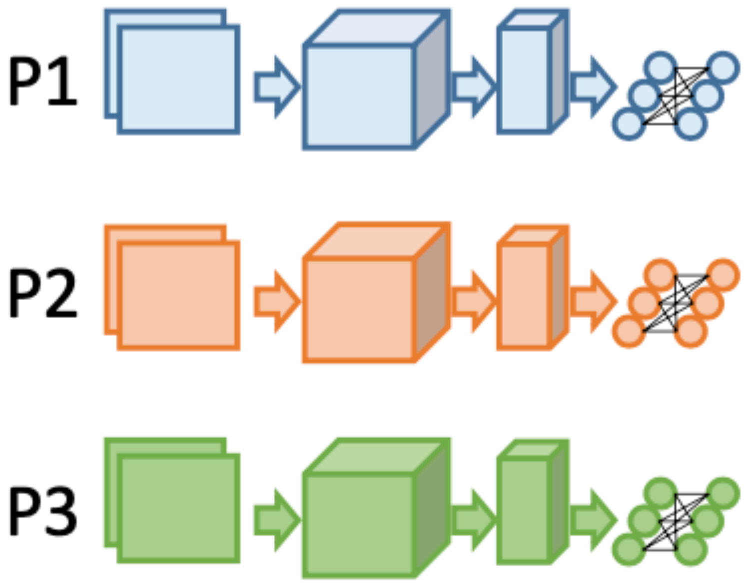
Name	Optimizer + Grad (devices/partitioned)	Parameters (devices/partitioned)
Data parallel	[GPU] / ✗	[GPU] / ✗
ZeRO 2	[GPU] / ✓	[GPU] / ✗
ZeRO-Offload	[CPU,GPU] / ✓	[GPU] / ✗
3D Parallelism	[GPU] / ✓	[GPU] / ✓
ZeRO 3	[GPU] / ✓	[GPU] / ✓
ZeRO-Inf-CPU	[CPU, GPU] / ✓	[CPU,GPU] / ✓
ZeRO-Inf-NVMe	[NVMe,CPU,GPU] / ✓	[NVMe,CPU,GPU] / ✓

ZeRO-Infinity: Breaking the GPU Memory Wall for Extreme Scale Deep Learning, https://arxiv.org/abs/2104.07857

Megatron-DeepSpeed

https://github.com/microsoft/Megatron-DeepSpeed

Data Parallel (DP)



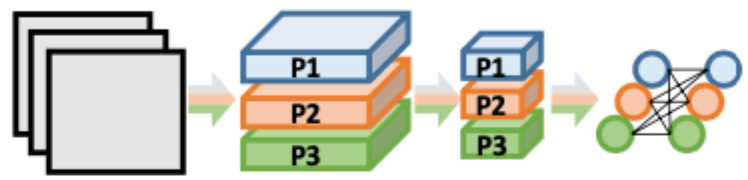
Distributed Data

Redundant Model

AllReduce Gradients

Large batch problem

Tensor Parallel (TP)
or
ZeRO/FSDP



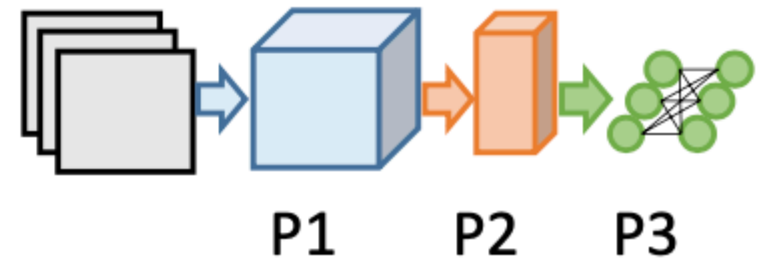
Redundant Data

Distributed Model

AllReduce Activations
or
AllGather Parameters

Frequent Communication

Pipeline Parallel (PP)



Redundant Data

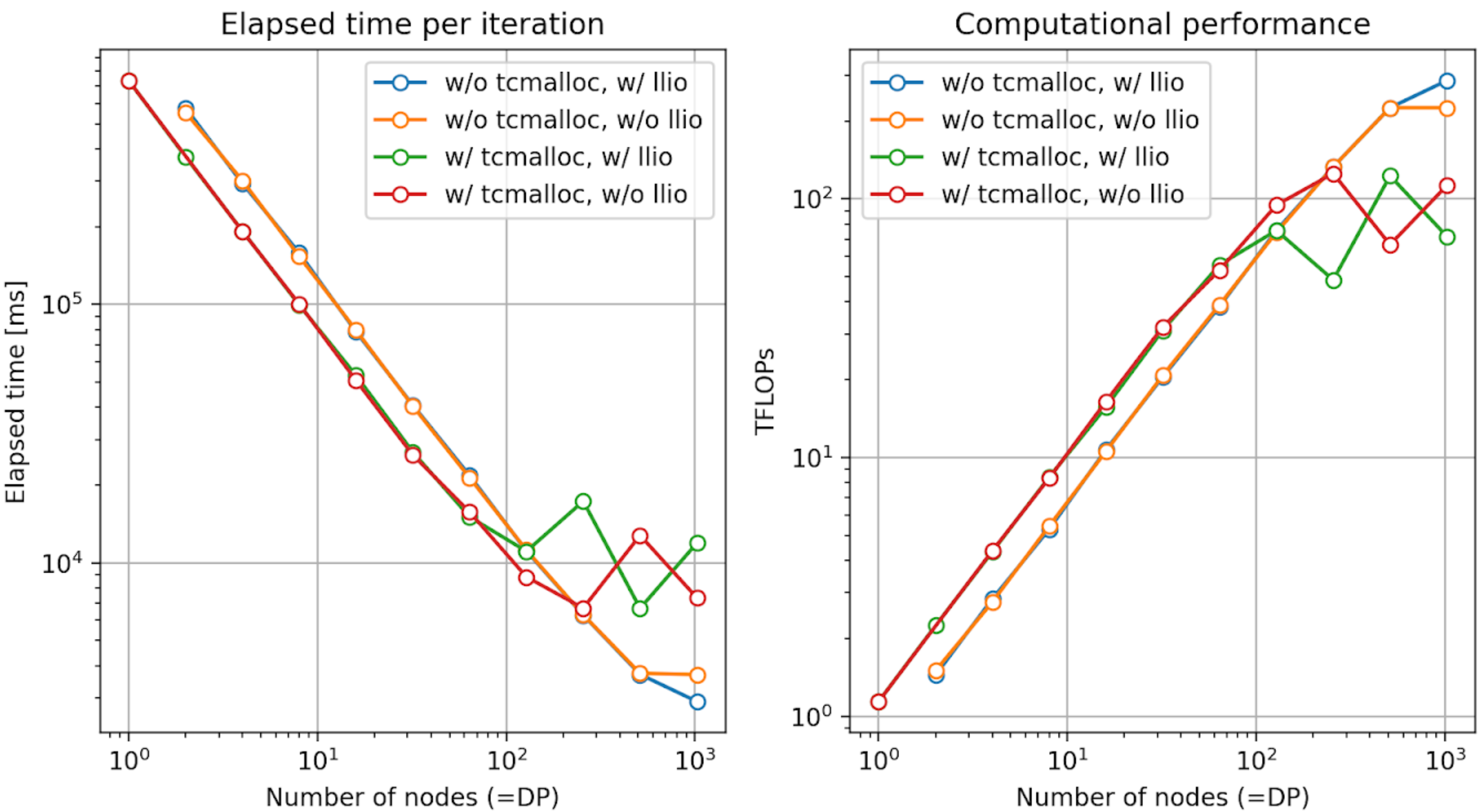
Distributed Model

SendRecv Activations

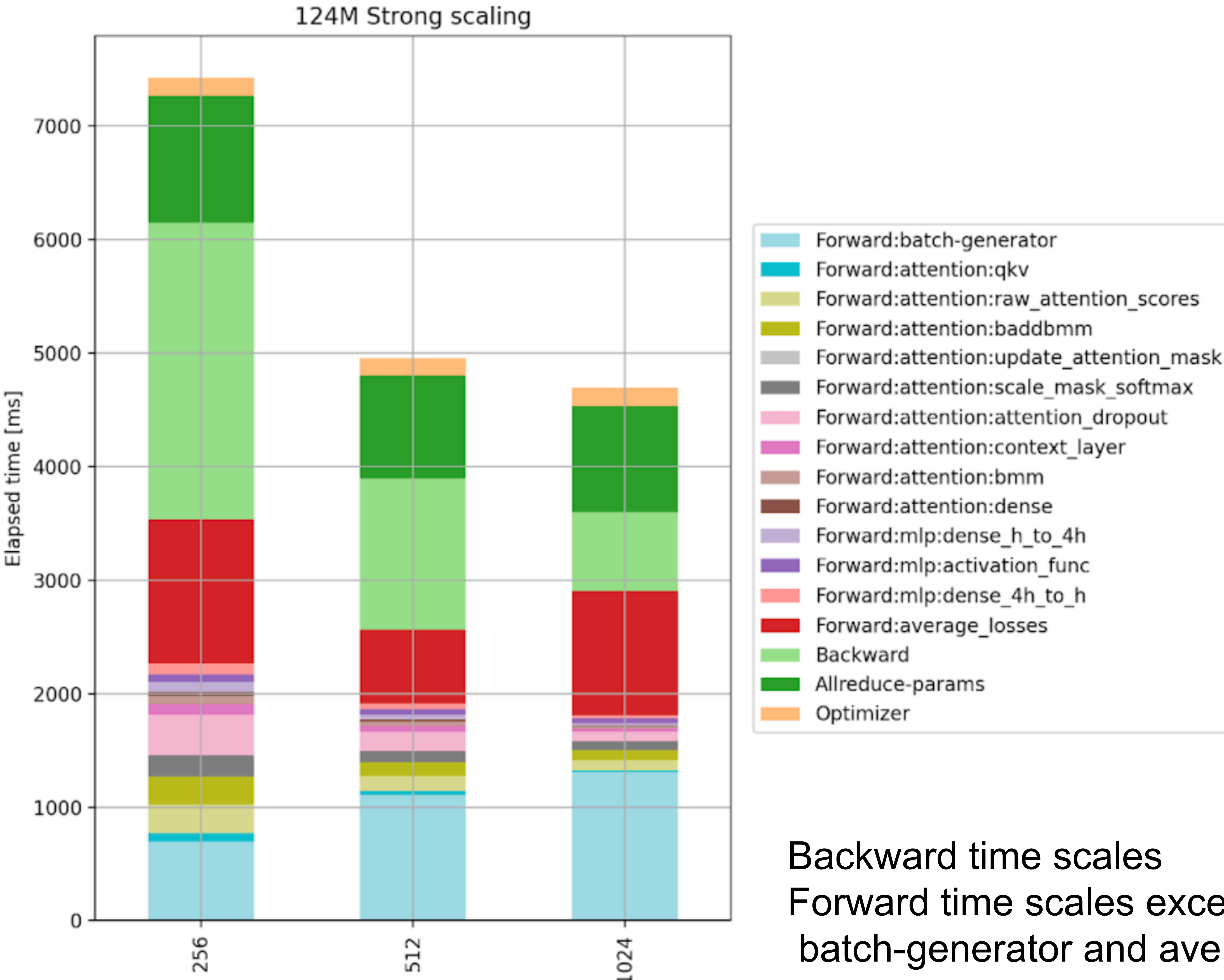
Pipeline bubble

Strong Scalability of Data Parallel

sequence-length=1024
per-cpu-batchsize=1, global-batch-size=1024
gradient-accumulation-steps=1024/#nodes
#parameters=**124M**



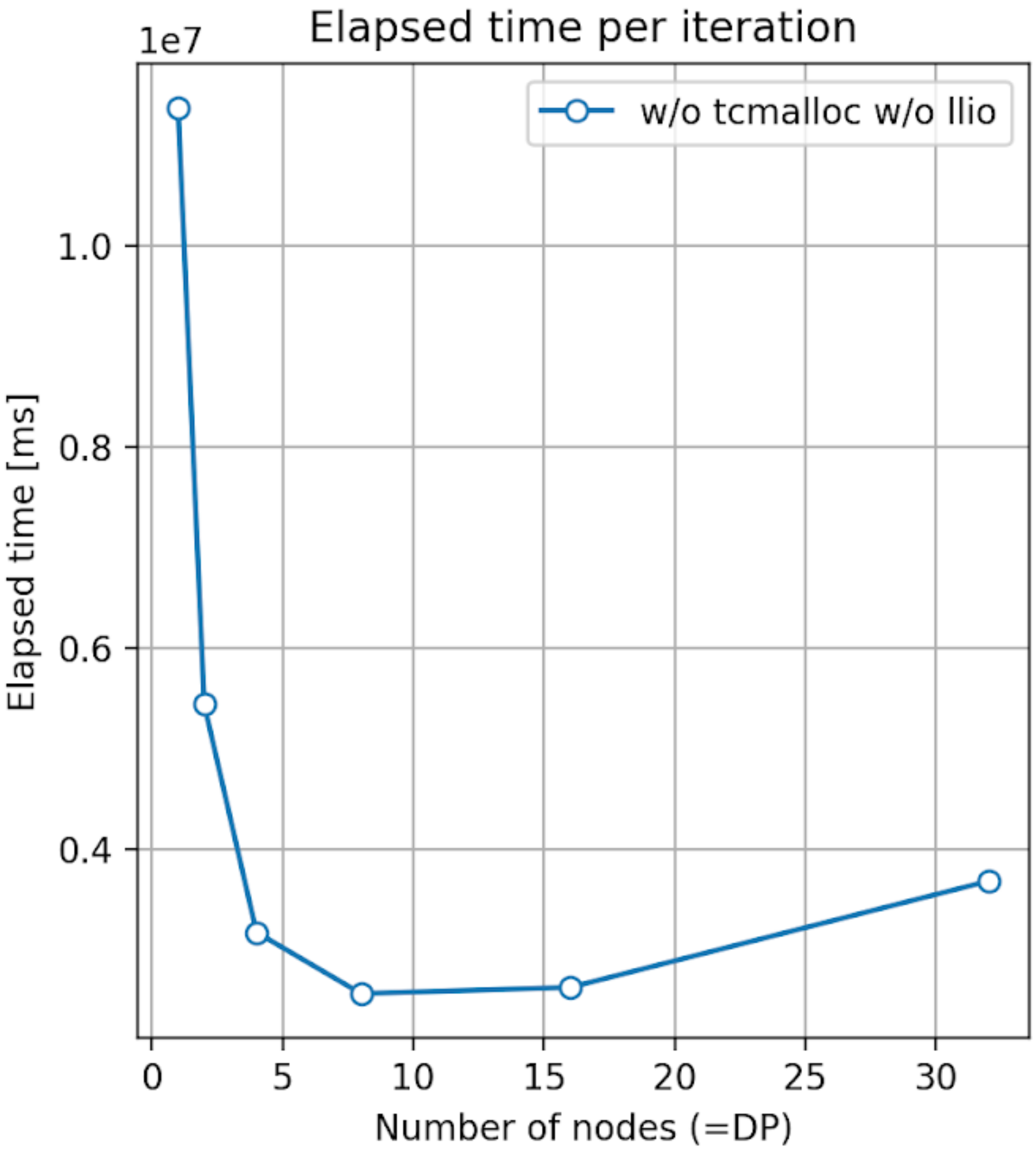
Breakdown of Data Parallel

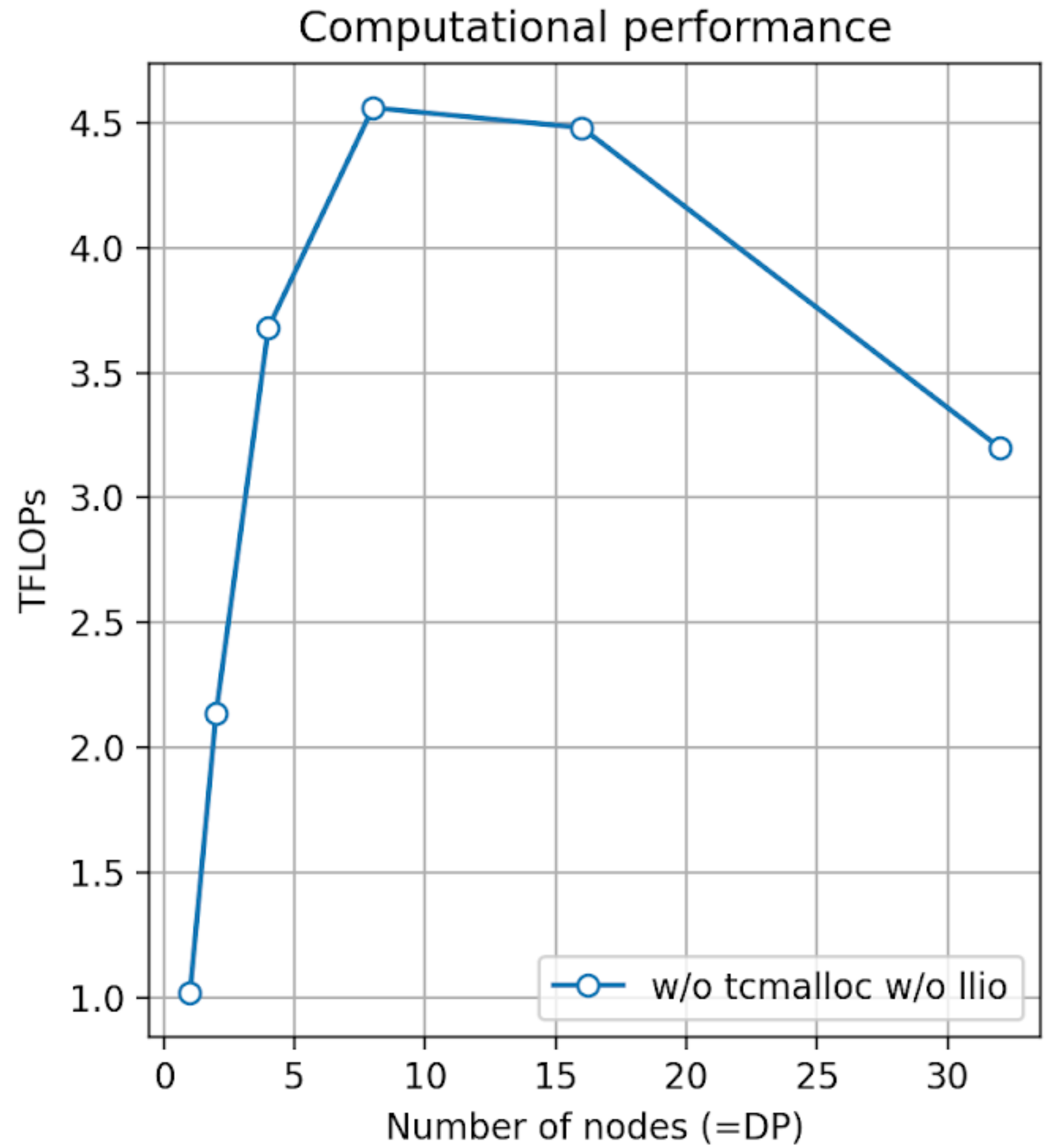


Backward time scales
Forward time scales except for…
batch-generator and average_losses
AllReduce time is constant

Model Parallel

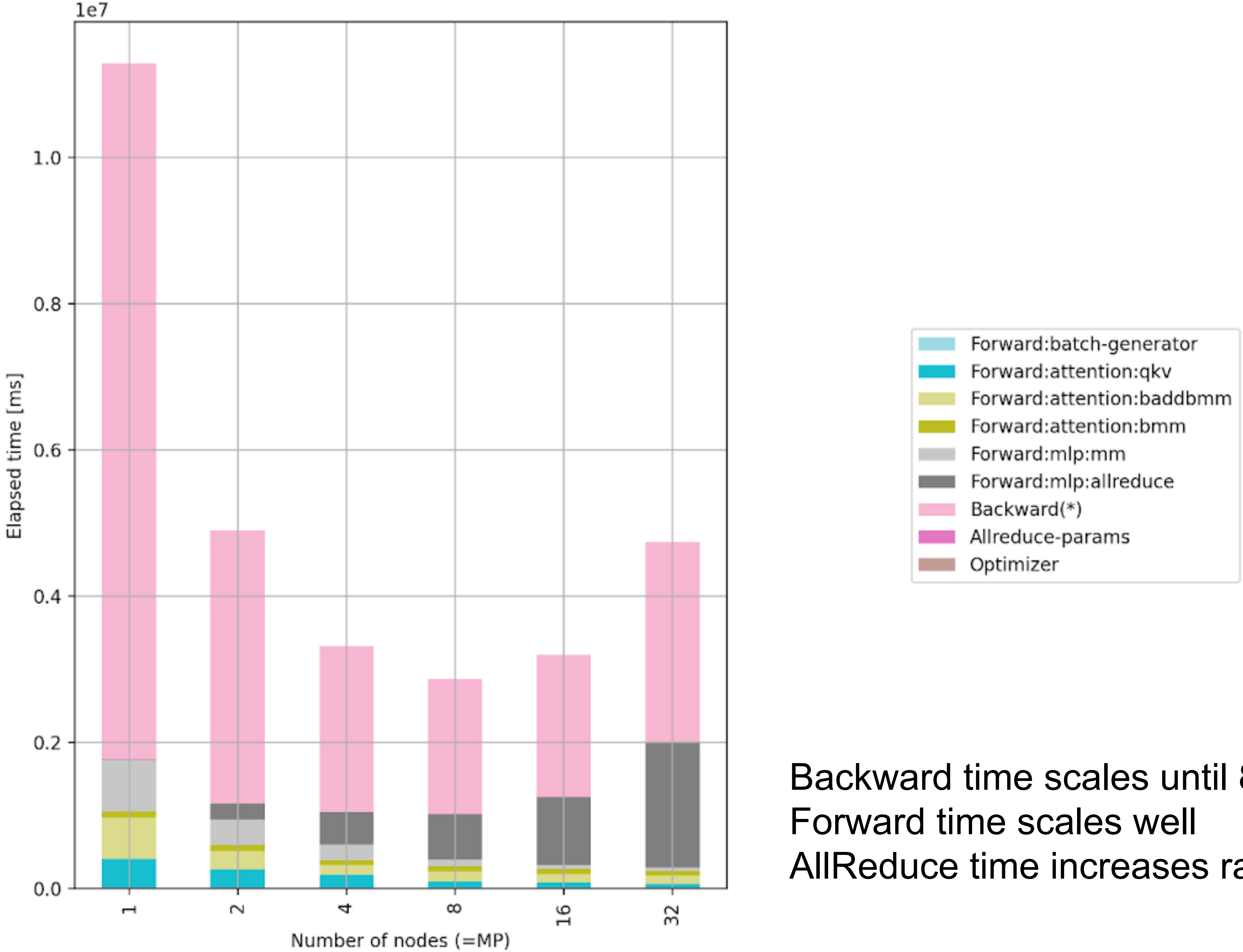
sequence-length=1024
per-cpu-batchsize=1024, global-batch-size=1024
gradient-accumulation-steps=1
#parameters=**1.3B**





Only scales up to 8 nodes at the moment

Breakdown of Model Parallel



Backward time scales until 8 nodes
Forward time scales well
AllReduce time increases rapidly

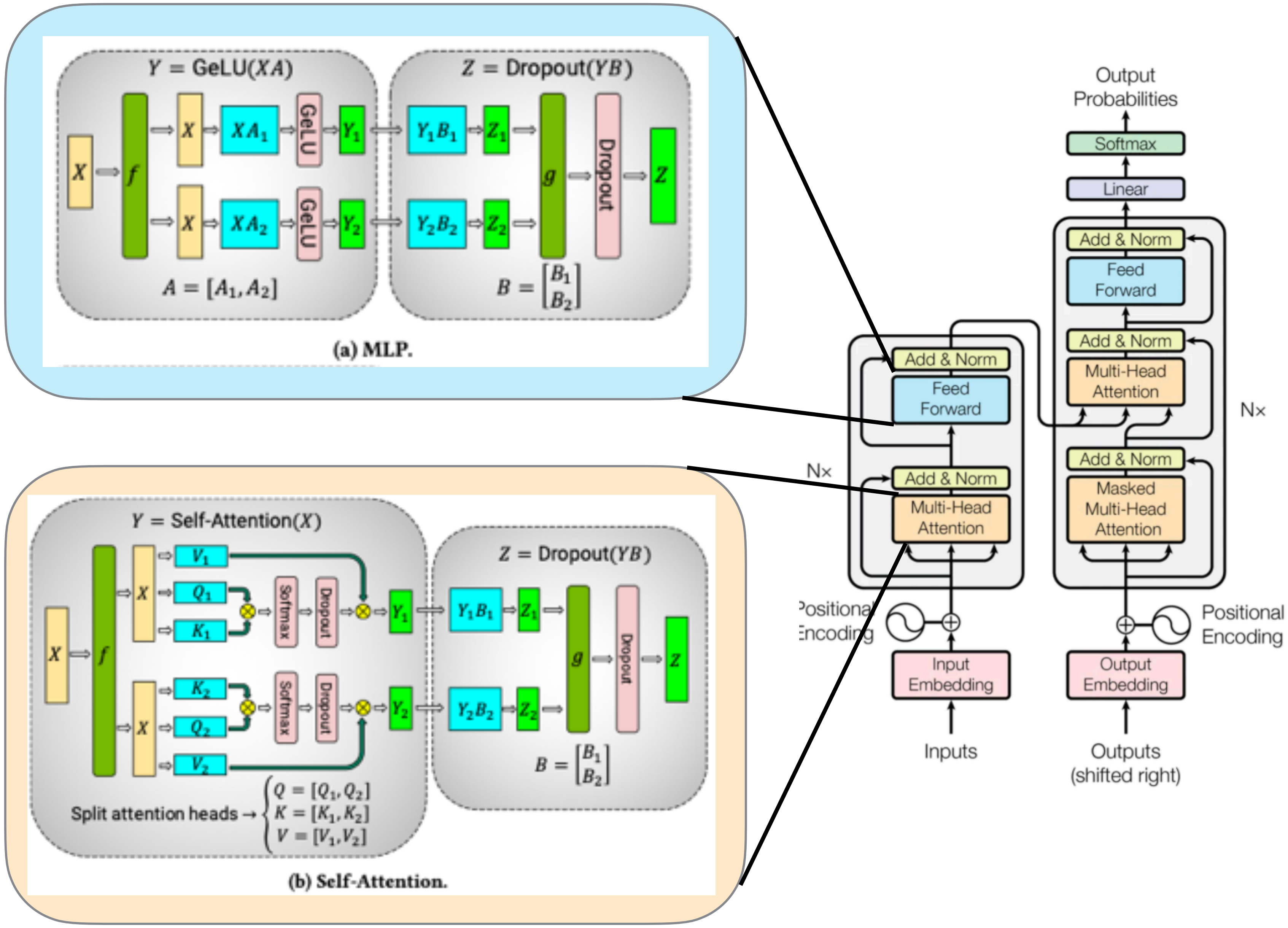
FLOPs Achieved on 1.3B Model

sequence-length=1024
per-cpu-batchsize=1, global-batch-size=1024
gradient-accumulation-steps=1024/#DP
#parameters=**1.3B**

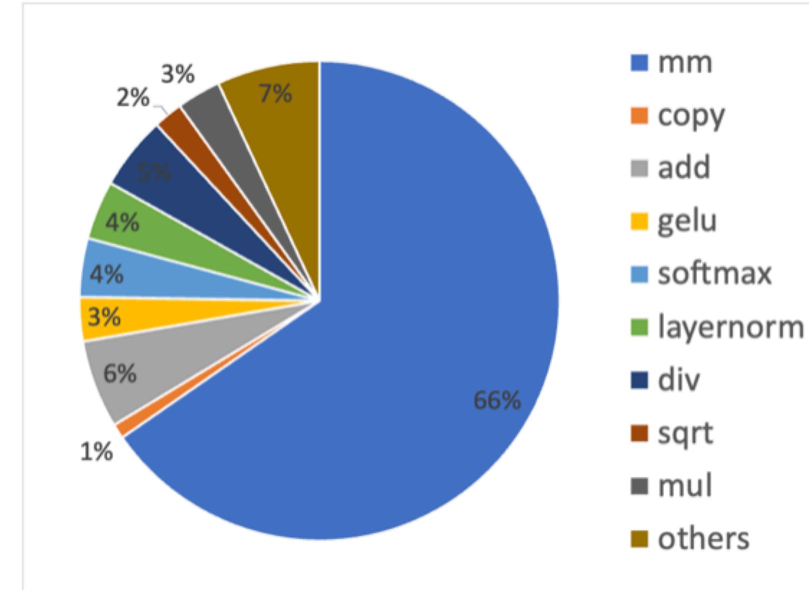
# CPUs	# DP	# MP	Achieved teraFLOPs per CPU	Percentage of Theoretical Peak FLOPS	Aggregated petaFLOPs per System	Equivalence to # of A100s (compared to 1.7B set-up)
1	1	1	0.99	16%	0.001	0.01
4	1	4	0.86	14%	0.003	0.02
64	16	4	0.84	14%	0.053	0.38
256	64	4	0.79	13%	0.198	1.44
1024	256	4	0.59	10%	0.590	4.31
2048	512	4	0.49	8%	0.980	7.15
4096	1024	4	0.41	**7%**	**1.640**	**11.97**

We are only getting around 10% of the theoretical peak of A64fx at the moment

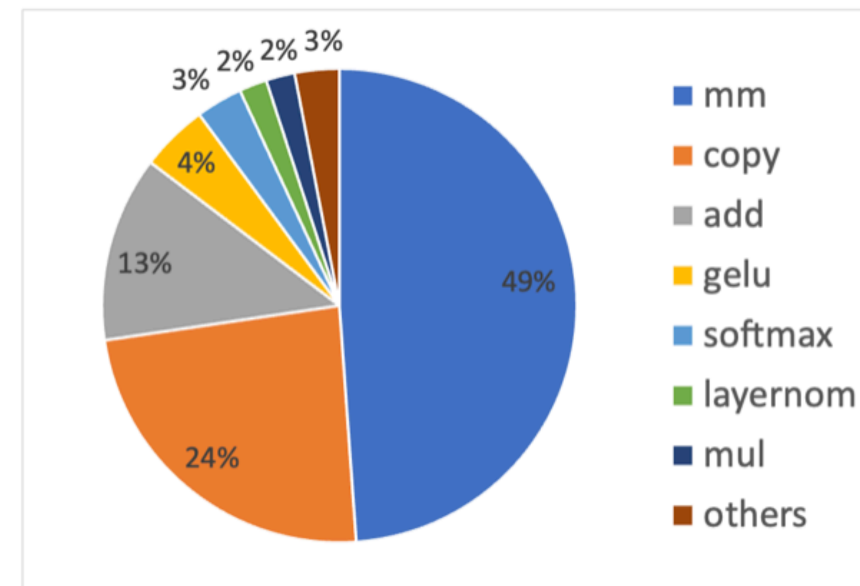
99% of the FLOPs is GEMM



A64FX



A100



Currently uses batched GEMM implementation by Daichi Mukunoki
→ Achieves 2 TFLOPs (FP32) on a single A64FX

https://www.r-ccs.riken.jp/labs/lpnctrt/projects/batchedblas/index.html

Du Wu and Mohammed Wahib are also working on a faster version

Summary and Outlook

Fugaku:
FP32 6.76TFLOP/s x 158,976 = 1.07 EFLOP/s (theoretical peak)
GPT-4: 328 days x 10
GPT-3.5: 32 days x 10
GPT-3: 3.3 days x 10

Actual Performance

<u>HPC tasks</u>

- Optimizing batched GEMM to scale across CMGs
- Develop techniques to enable FP16 training
- Optimize non-GEMM operations on A64FX
- Reduce communication overhead

<u>NLP tasks</u>

- Collecting, downloading, and cleaning large multilingual corpa
- Discuss legal issues with lawyers https://storialaw.jp/blog/9239
- New models appearing every week: Alpaca, LLaMA, RWKV https://github.com/Hannibal046/Awesome-LLM
- Reinforcement learning with human feedback (RLHF)

Reference on the NLP side : Slides by Naoaki Okazaki https://speakerdeck.com/chokkan/20230327_riken_llm