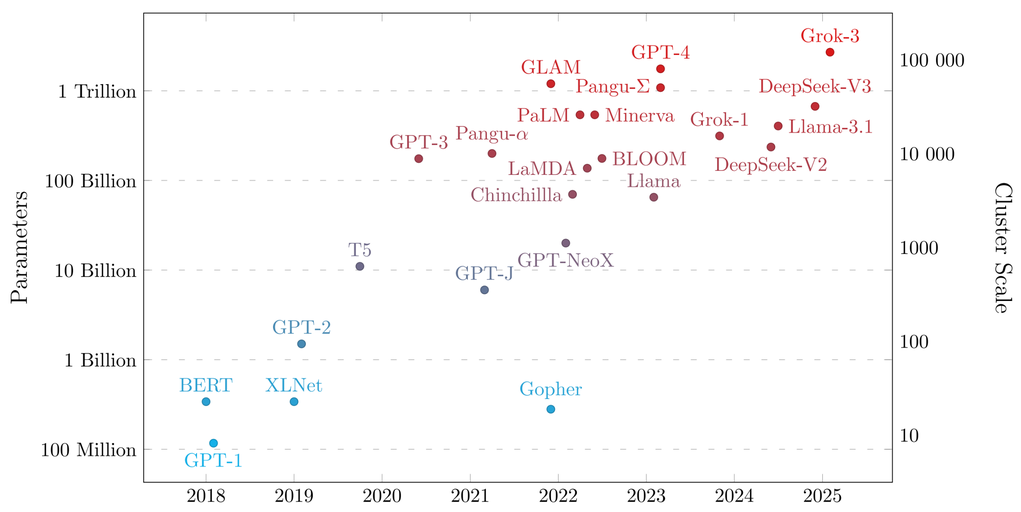
Improving Giant Neural Network Performance with Symbolic Analytical Formulas

Shijie Shen[1,4], Walid Astaoui[2,3,4], Thibaut Tachon[4], Chong Li[4]

[1]École Polytechnique [2]CentraleSupélec [3]LISITE Lab, ISEP [4]Huawei DPSL-Paris

Background

In recent years, **Large Language Models (LLMs)** have gained high interest and show remarkable capabilities in various fields. However, along with their increasing performance, their computational and memory demands grow exponentially. Whereas the first generation of LLMs in 2018 contained only a few million parameters, today's state-of-the-art models have bypassed the size of 1 trillion parameters.



Challenge

Various distributed training techniques have been proposed to improve training efficiency and to reduce memory consumption (data, tensor, pipeline, expert, recompute, offloading, etc.). In practice, the optimal strategy is often a carefully tuned combination of these approaches.

It has several drawbacks:

- Demanding engineers with deep expertise in both **LLM architecture** and **distributed training**
- Requiring heavy time-consuming **profiling**
- Bounded with **model architecture, AI frameworks** and underlying **hardware**

State of the Art

Existing automatic parallel training system fall into two broad categories:

- **Black-Box and Gray-Box Approaches**

  Rely on extensive profiling or trial runs to explore the optimization space, which can be prohibitively time-consuming.

- **Symbolic Analytical Methods**

  Use formula-based cost models to predict performance, but remain tightly coupled to specific AI frameworks and hardware, limiting their portability and robustness.

	Method	DP + TP + PP	Other dims	Memory
vTrain	Profiling	✓	PP scheduler + All Reduce fusion	✓
Galvatron	**Gray-Box**	✓	Zero + Batch	✓
InternEvo	**Gray-Box**	✓	State sharding + SP/UP	✓
Calculon	**Analytical**	✓	Many	✓
AMPeD	**Analytical**	✓	Expert Parallelism	✗
Mist	**Analytical**	✓	Recompute Zero + SWAP	✓

* Notation: DP (Data Parallelism), TP (Tensor Parallelism), PP (Pipeline Parallelism)

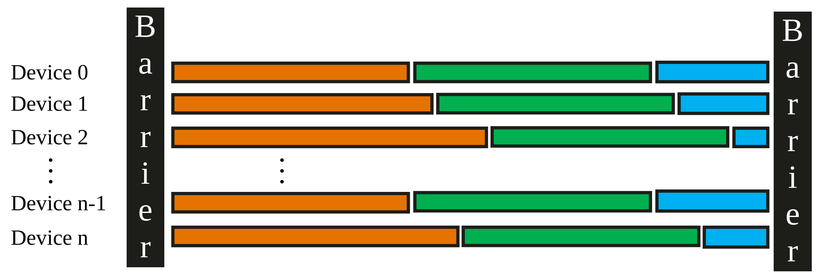
* Publication year :

vTrain: MICRO 2024 Galvatron: VLDB 2022 InternEvo: arXiv 2024
Calculon: SC 2023 AMPeD: ISPASS 2023 Mist: EuroSys 2025

Our Methodology

Our Methodology is as follows:

- Model the distributed training process and abstract the components that are independent from the model architecture, framework and hardware.
- Optimizing the independent aspects, which guarantee robustness and portability across different framework and hardware.
- Deducing symbolic analytical formulas that predict computation cost, communication overhead, and synchronization time for each parallelism strategy.



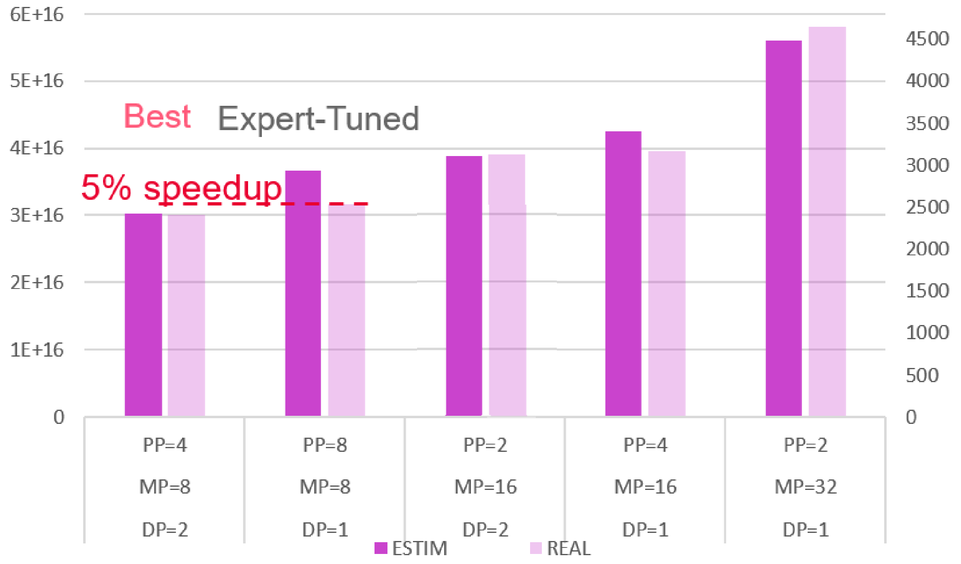
Computation + Communication + Synchronization = Overhead

Current Test Result

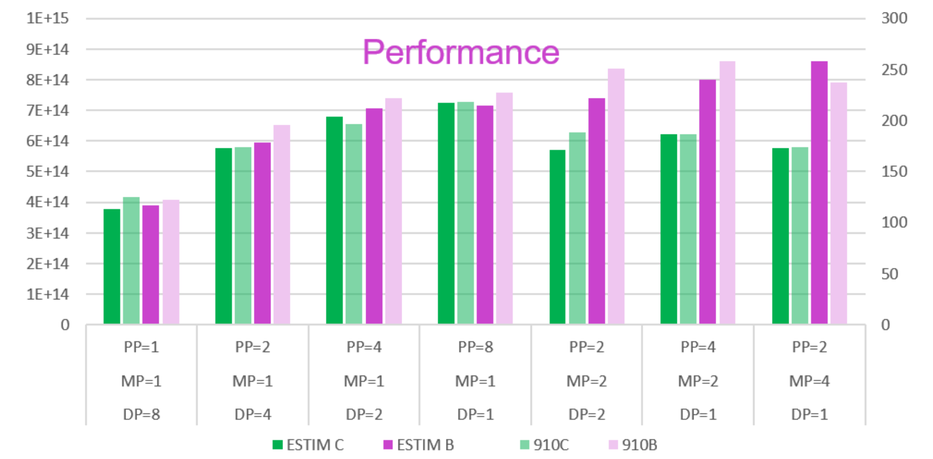
On Qwen, our proposal finds a better strategy than the expert-tuned one.

On Llama2, our proposal remains a good performance estimation on different hardware.

Qwen1.5-72B; 64 devices, vary: DP, MP, PP



Llama2-7B; 8 devices, vary: DP, MP, PP, on Ascend 910B and 910C



Insights

Current work aims to quantify the **computation**, **communication** and **synchronization** overhead of different parallelism strategies without introducing the dependency on AI framework and hardware. We observe a strong correlation between the model architecture and the parallelism strategies. **By joint analysis of neural architectures and parallelization techniques**—and extracting their shared, structural commonalities—we can co-optimize models inherently tailored for distributed training. This co-design approach yields neural networks that are both computation-efficient and scalable across diverse hardware and frameworks, this is our next step.