HALoS: Hierarchical Asynchronous Local SGD over Slow Networks for Geo-Distributed LLM Training

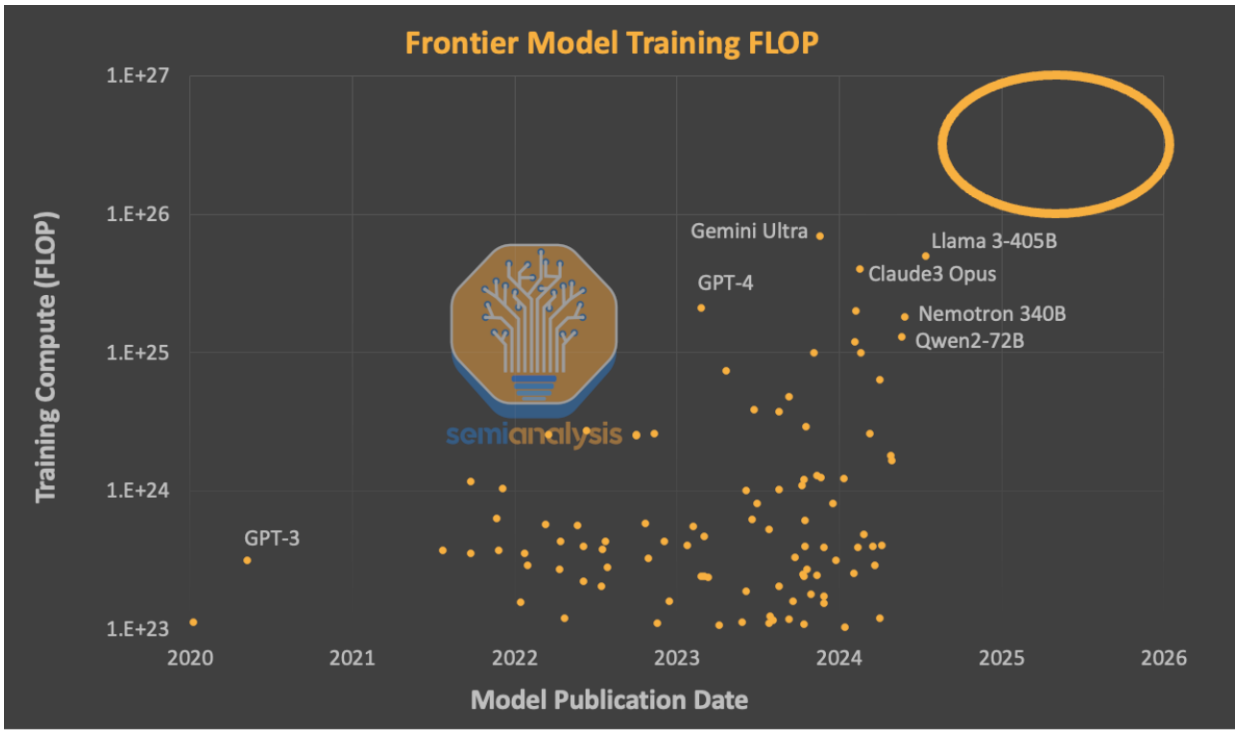
Geon-Woo Kim[1] Junbo Li[1] Shashidhar Gandham[2]
Omar Baldonado[2] Adithya Gangidi[2] Pavan Balaji [2]
Atlas Wang[1] Aditya Akella[1]

[1]UT Austin [2]Meta

Forty-Second International Conference on Machine Learning, July 2025

Massive Scaling Trends in LLMs

- Large Language Models (LLMs) continue to grow in size and training cost
  - GPT-4: 1.8T parameters
  - Gemini Ultra: 1.6T parameters
  - Llama 4: 2T parameters
- Training at exascale and far beyond is on the horizon



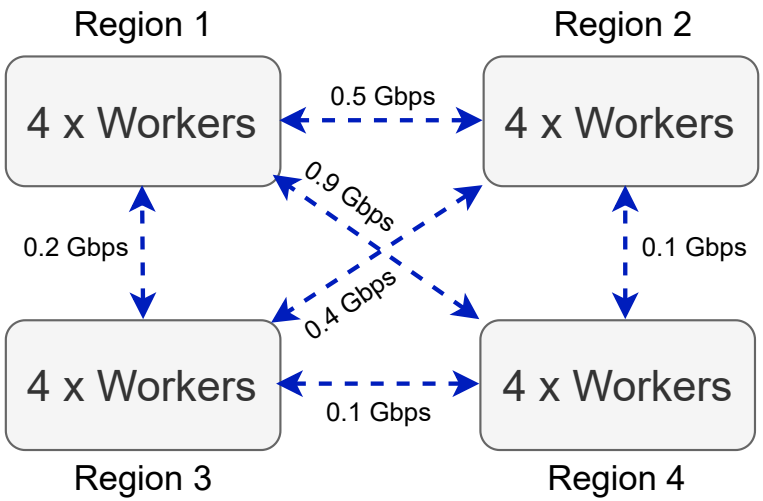
Source: SemiAnalysis

Fully Synchronous Training: The Standard Today

- Fully synchronous SGD and variants remain dominant.
- **All model parameters & gradients** (trillions) are synchronized for every iteration.
- Enabled in a single homogeneous datacenter[1]:
  - Ultra-fast intra-datacenter interconnects (e.g., NVLink, InfiniBand)
  - Highly homogeneous hardware setup (e.g., 16K H100 GPUs)
- Becoming impractical:
  - Enormous **power demands** for large clusters
  - **Limited availability** of cloud resources in the same region
  - Rapid evolution of **hardware heterogeneity** (Nvidia Blackwell, Nvidia Hopper, .. AMD MI350, etc.)

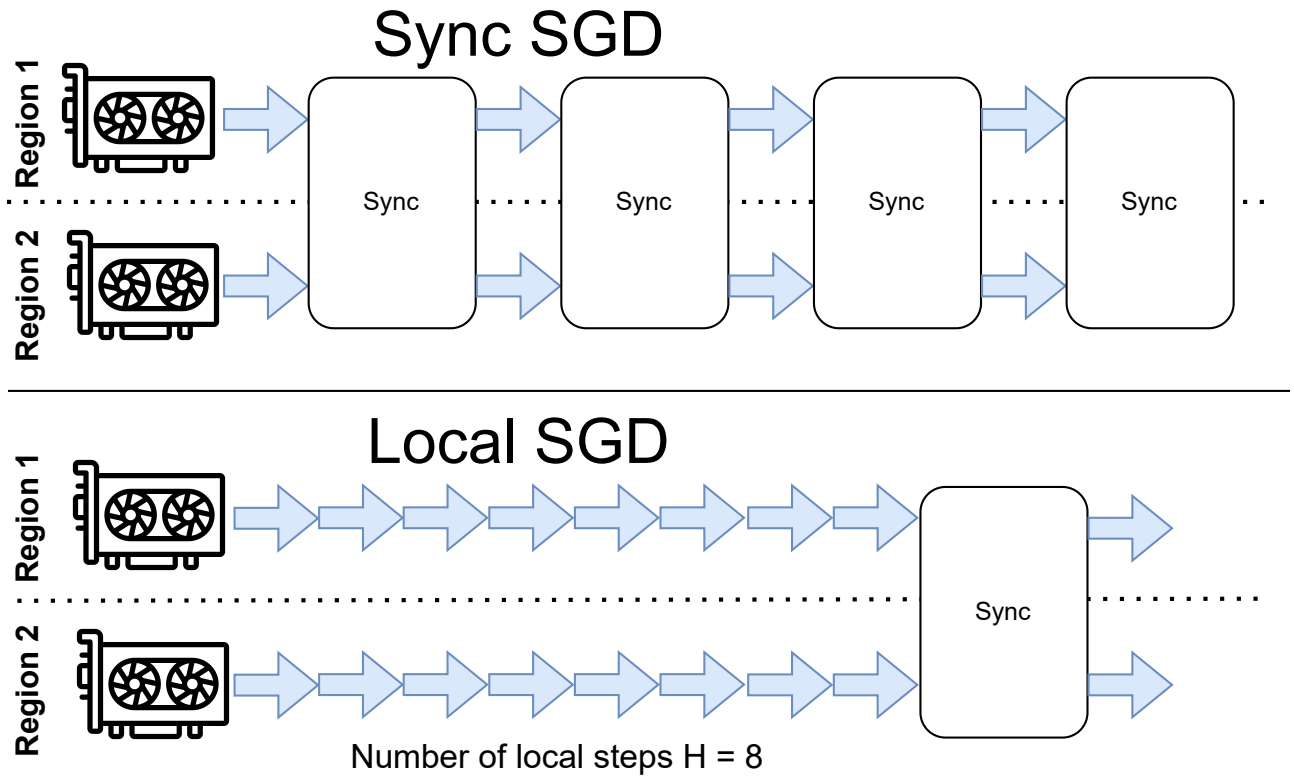
[1] Llama Team, AI @ Meta, The Llama 3 Herd of Models.

Geo-Distributed Environment for Training LLMs



- **Geo-distributed training would be inevitable**, but:
  - Inter-region network bandwidth can be 10×-100× slower than intra-region bandwidth
  - Heterogeneous accelerators performance can be 3×-10× different
- **Synchronous training will suffer**:
  - Inter-region communication bottlenecks
  - Straggler effect (slowest worker stalls all)
- **Need both comm efficient & straggler tolerant training algorithm**

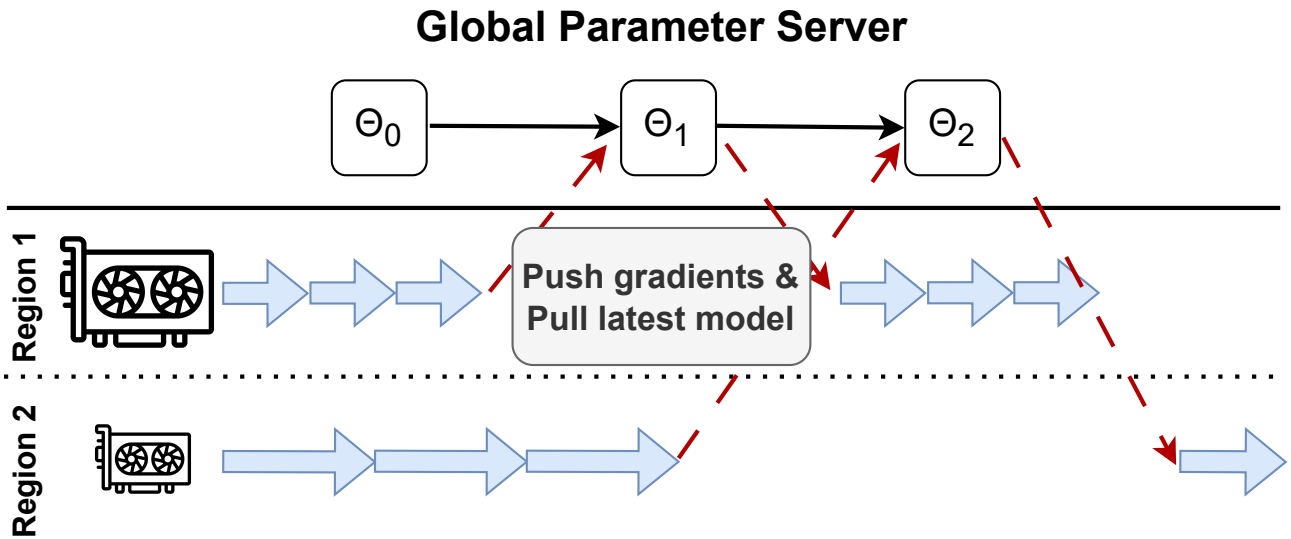
One Approach: Local SGD for Mitigating Communication



- Recent work (DiLoCo)[1] adapts Local SGD for geo-distributed LLM training
  - Workers do multiple *local* updates before a global sync
  - Amortizes expensive cross-region communication
- However, a slowest worker (straggler) can stall entire training due to the strict synchronization

[1] Arthur Douillard *et al.*, "DiLoCo: Distributed Low-Communication Training of Language Models." WANT@ICML 2024.

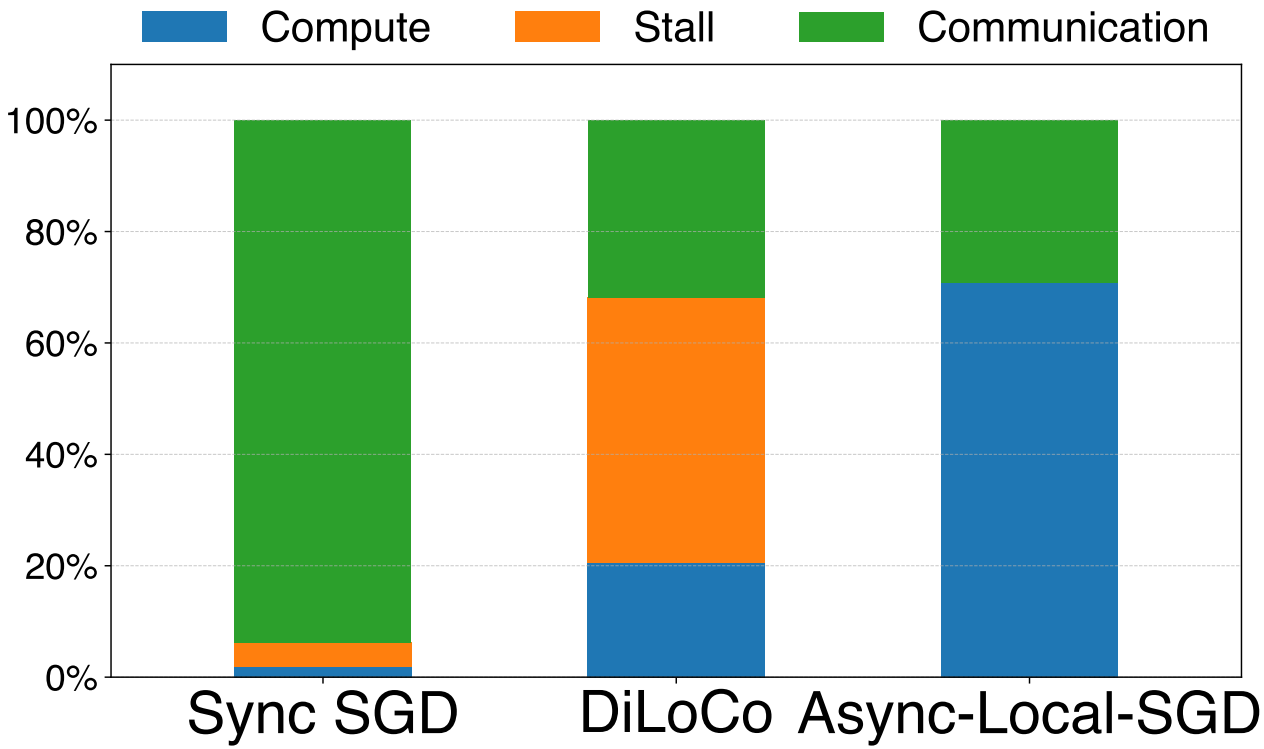
Asynchronous Training to Overcome Stragglers



- Async-Local-SGD[1] addresses straggler problem with asynchronous training
  - Push (pseudo-)gradients to a parameter server after multiple local steps
  - Without global sync, workers continue training with the latest global model
- Limitations:
  - Slow inter-region comm is still used to communicate with the parameter server
  - All updates are handled by a single (global) parameter server
  - Lack of theoretical analysis for guaranteed convergence

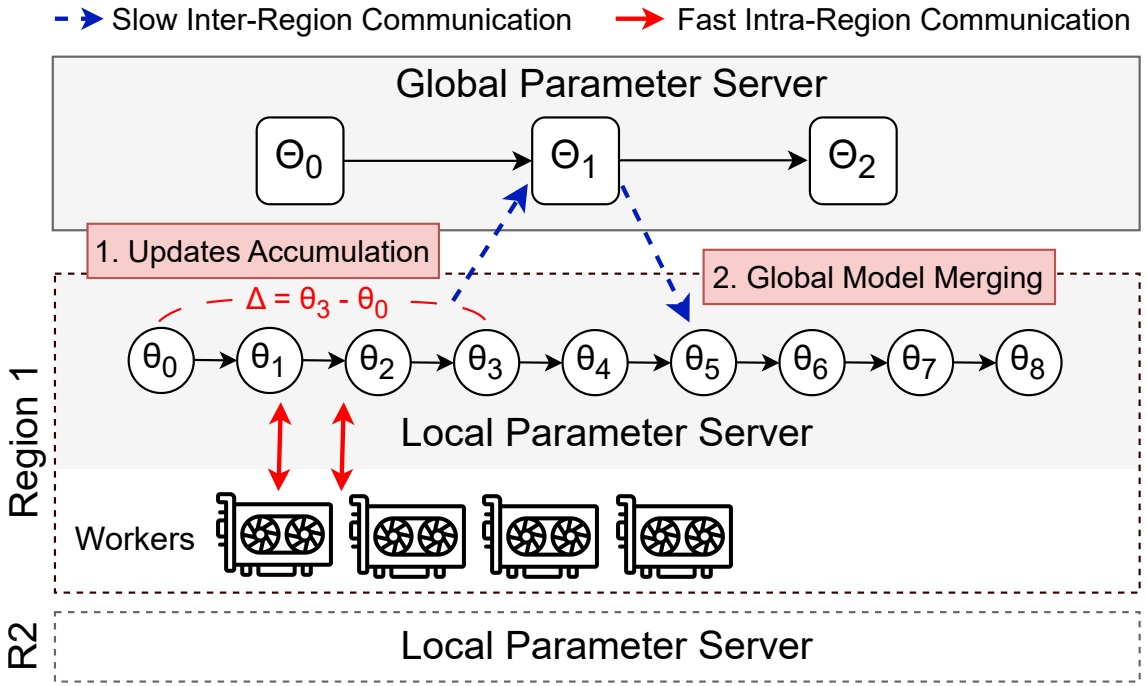
[1] Bo Liu *et al.*, "Asynchronous Local-SGD Training for Language Modeling." WANT@ICML 2024.

Motivating Experiment



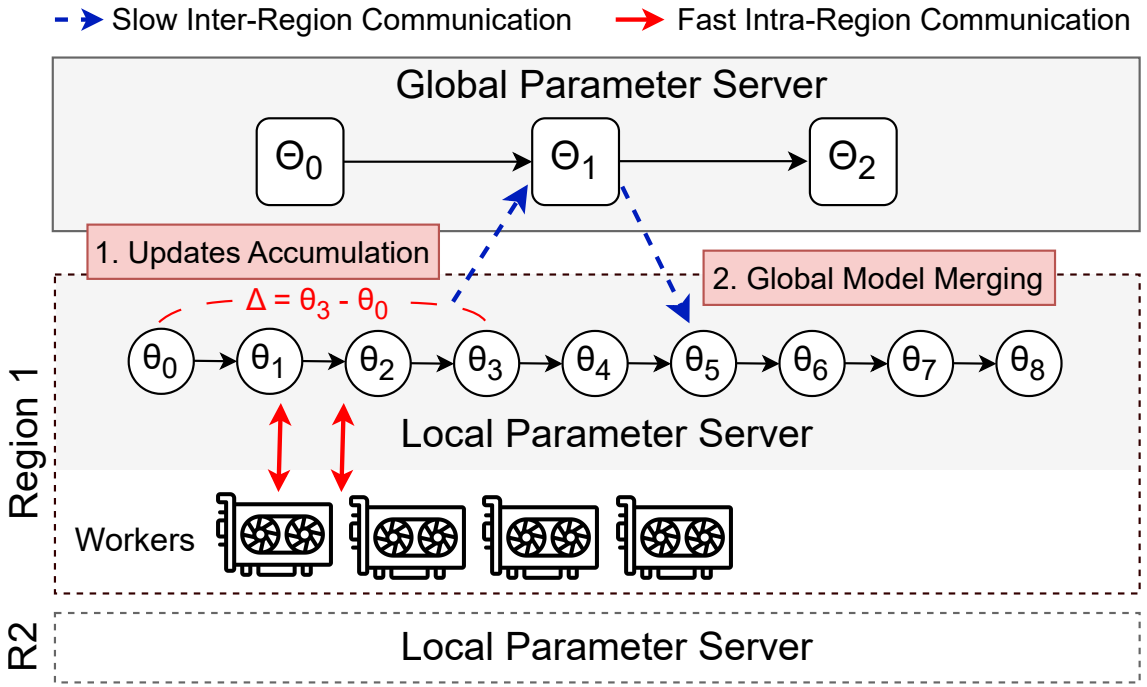
- Worker runtime breakdown
  - Sync methods (Sync SGD and DiLoCo) suffer from communication bottlenecks and stragglers ($> 80\%$)
  - Async-Local-SGD reduces stalling but still incurs high inter-region communication cost ($\sim 30\%$)

Our Proposal: Hierarchical Asynchronous Training (1/2)



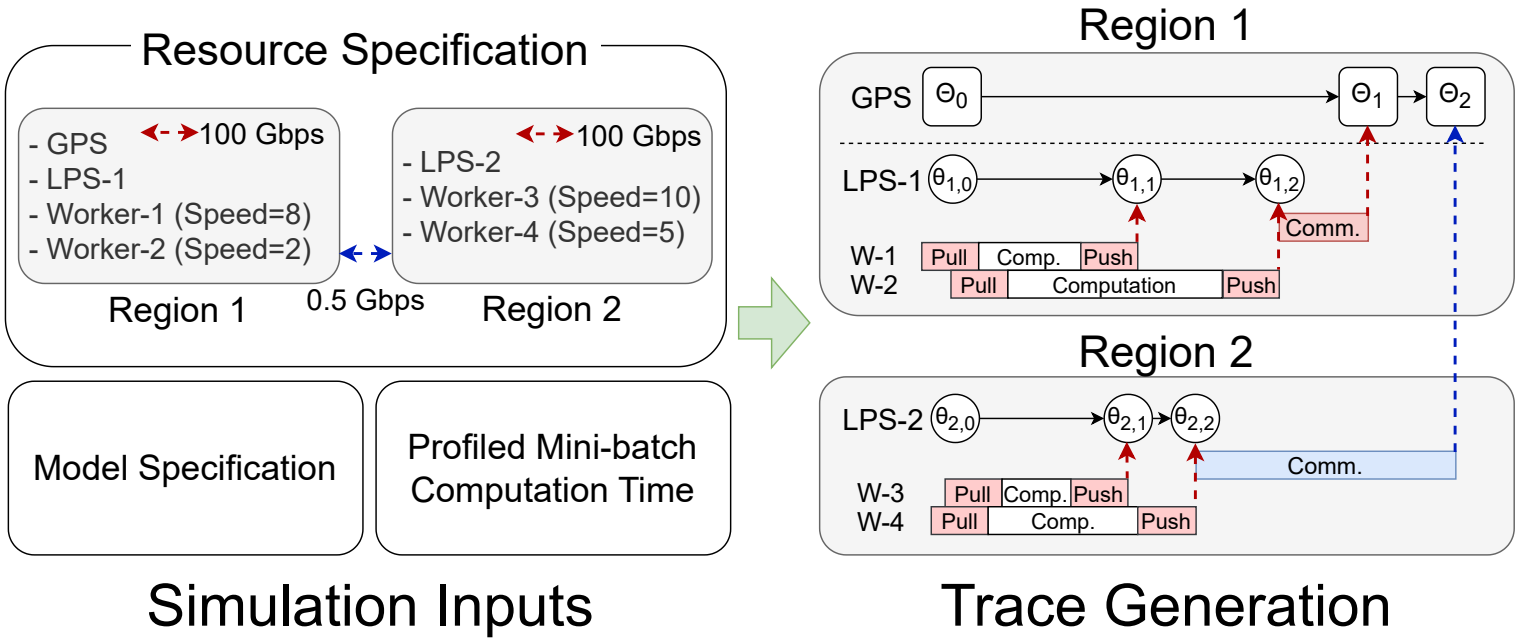
- Our hierarchical design:
  - Introduce **local models** communicating with workers in the same region
  - Fully **asynchronous** at local and global levels
  - Comes with **formal convergence analysis**
- Local Parameter Servers (LPSs) leverage fast intra-region communication and updates a local model with co-located workers
- Global Parameter Server (GPS) orchestrates learning across different LPSs, asynchronously decoupling slow inter-region communications from workers

Our Proposal: Hierarchical Asynchronous Training (2/2)



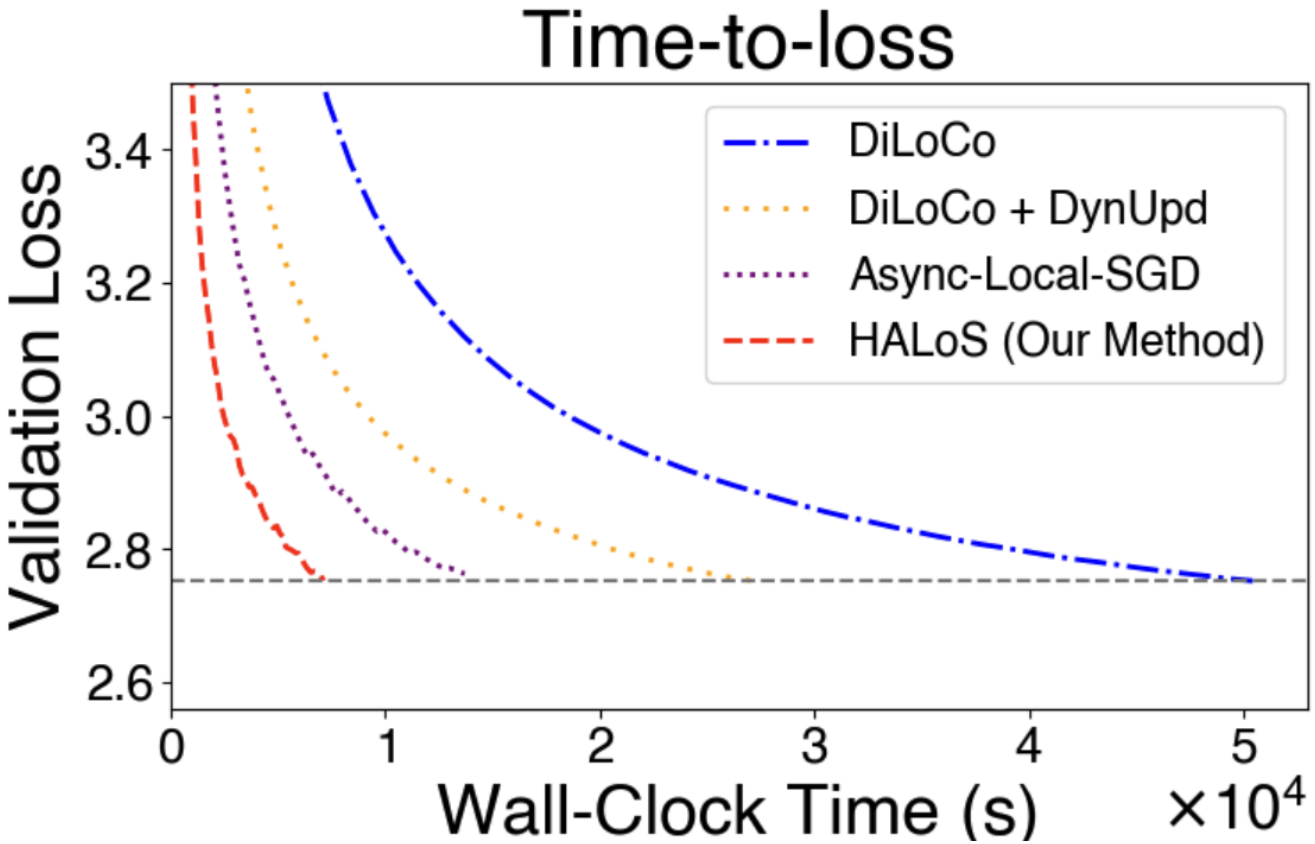
- **Local server-side updates accumulation**
  - LPSs accumulate updates and communicate with the global server at optimized intervals
  - Significantly reducing the global server overhead and ensuring scalability
- **Global model merging**
  - LPSs merge the updated local model with the model pulled from GPS
  - Effectively applying learning progress during LPS-GPS communication

Execution Trace-Driven Simulation



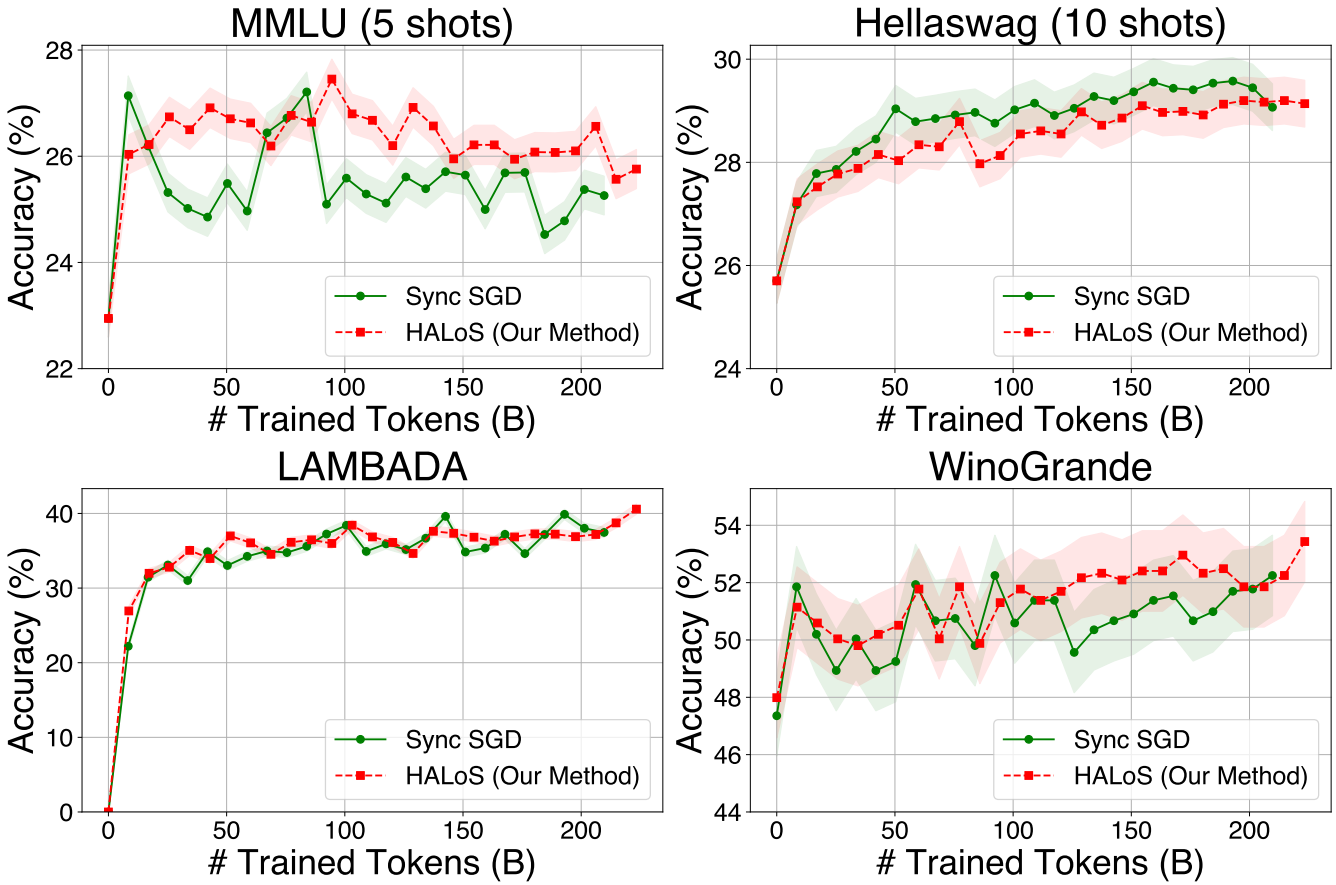
- **Inputs**:
  - Inter- and intra-region communication bandwidths
  - Placements of GPS, LPSs, and workers
  - Heterogeneous worker speeds
  - Profiled training iteration time on H100
- Calculate computation and communication times
- Generate an execution trace that contains the orders and dependency of local/global models and execute the trace on real GPUs
- Open-sourced at https://github.com/utnslab/halos

End-to-end Performance



- **HALoS converges faster than**:
  - DiLoCo **7.1**× and DiLoCo+DynUpd **3.8**×
  - Async-Local-SGD **1.9**×
- Trained Pythia-160M until each method matches the validation loss of training 12.9B tokens by sync SGD

Benchmark Performance



- HALoS matches or exceeds the test accuracy of fully synchronous SGD, while converging **68.6**× faster
- Fully train Pythia-160M model on one epoch (209B) of Pile dataset
- Run standard benchmarks: MMLU, Hellaswag, LAMBADA, and WinoGrande

Summary

- Upcoming geo-distributed LLM training faces communication bottlenecks and stragglers, limiting synchronous methods.
- HALoS utilizes hierarchical async updates via local and global parameter servers, with proven convergence.
- It significantly outperforms local SGD-based methods and matches the benchmark performance of synchronous SGD.