DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving

Yinmin Zhong Shengyu Liu Junda Chen Jianbo Hu
Yibo Zhu Xuanzhe Liu Xin Jin Hao Zhang

北京大学
PEKING UNIVERSITY


Various applications are powered by LLMs

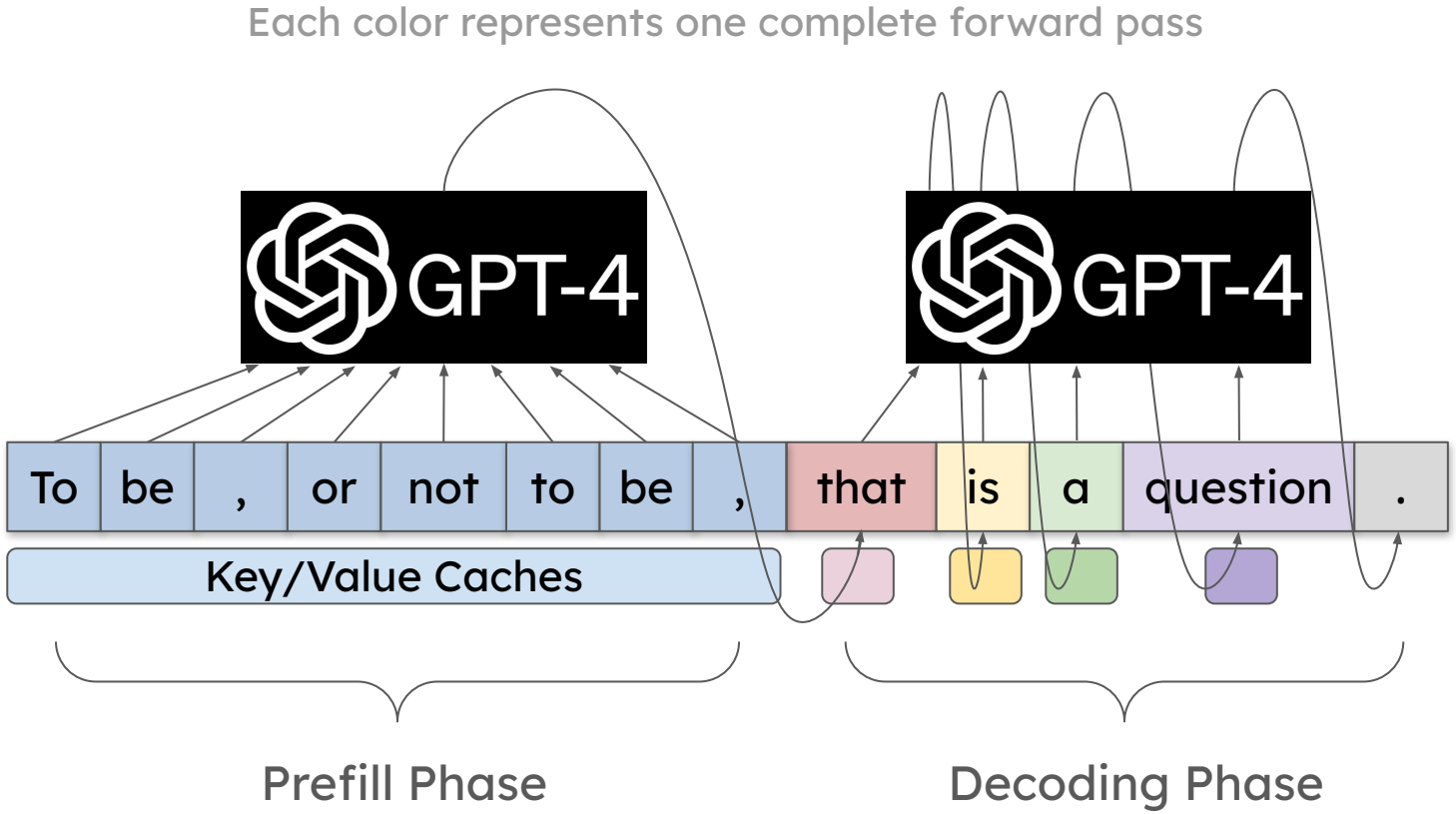
Search

OS

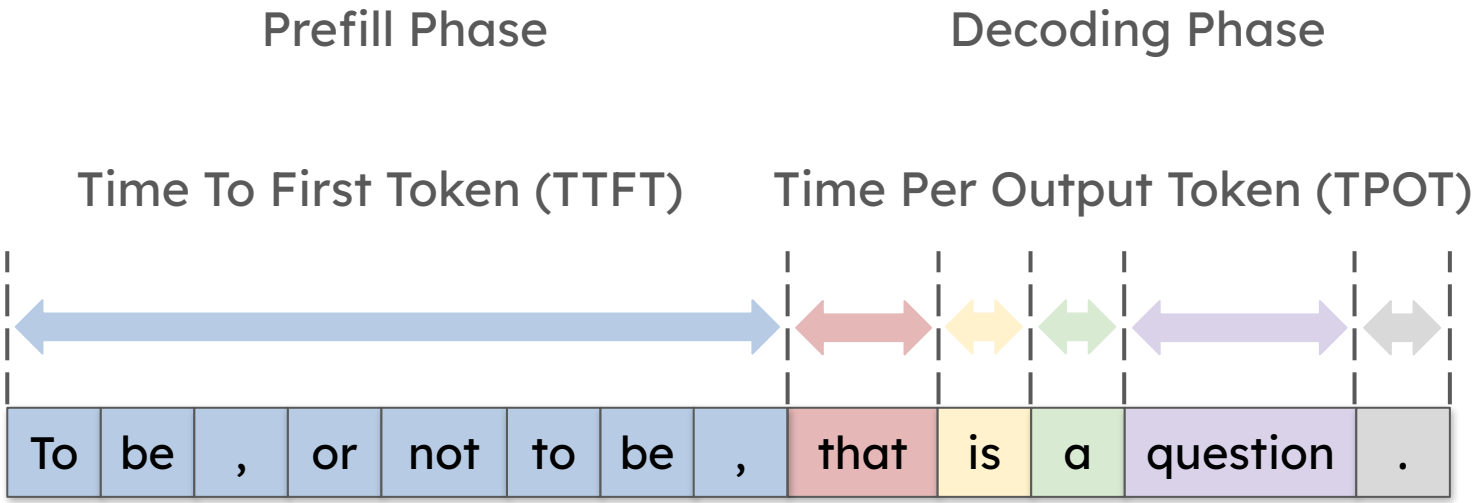
Chat

Program

LLM Inference



LLM Inference Latency



Latency = TTFT + TPOT * #Token

Different apps have various latency requirements

	TTFT	TPOT
Chat	Low (< 1s)	Match read speed (~ 100ms)
Search	Very Low (~ 200ms)	Match read speed (~ 100ms)
Program	Very Low (~ 200ms)	Very Low (~ 50ms)

Our Goal

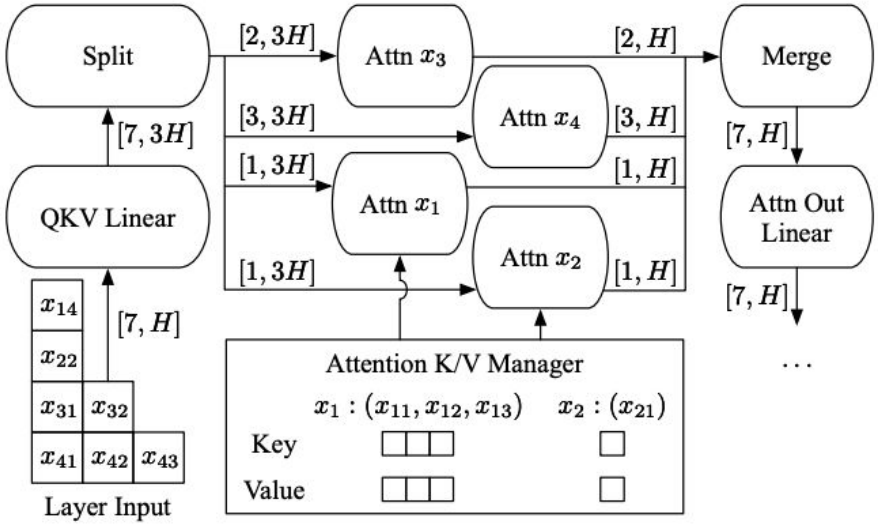
Use the fewest #GPUs to serve as many requests as possible while subject to various latency requirements of apps.

- *Service Level Objective (SLO)*
- *SLO Attainment*
- *Per-GPU Goodput*

Formally, our goal is to maximize the Per-GPU Goodput

Existing systems maximize the overall throughput

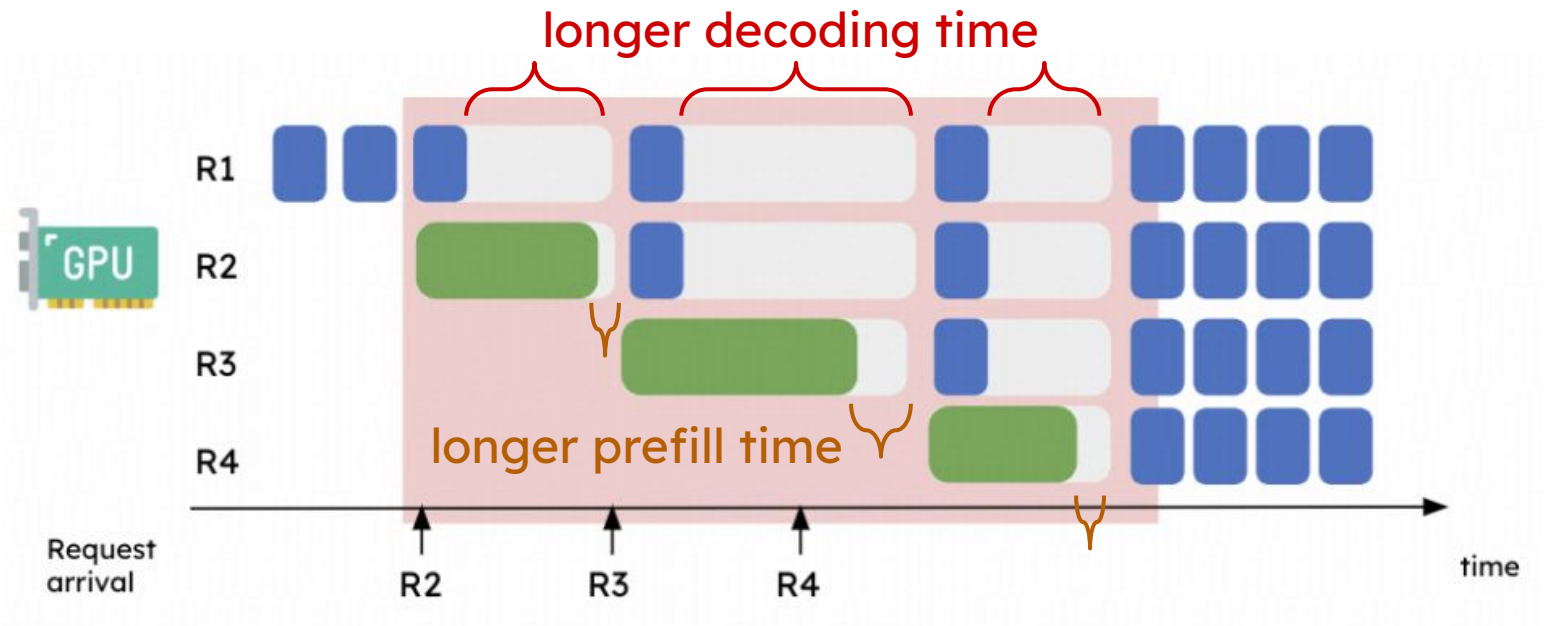
Existing approach: Batch requests in prefill and decoding phase together



Source: Gyeong-In Yu, et al. “Orca: A Distributed Serving System for Transformer-Based Generative Models”

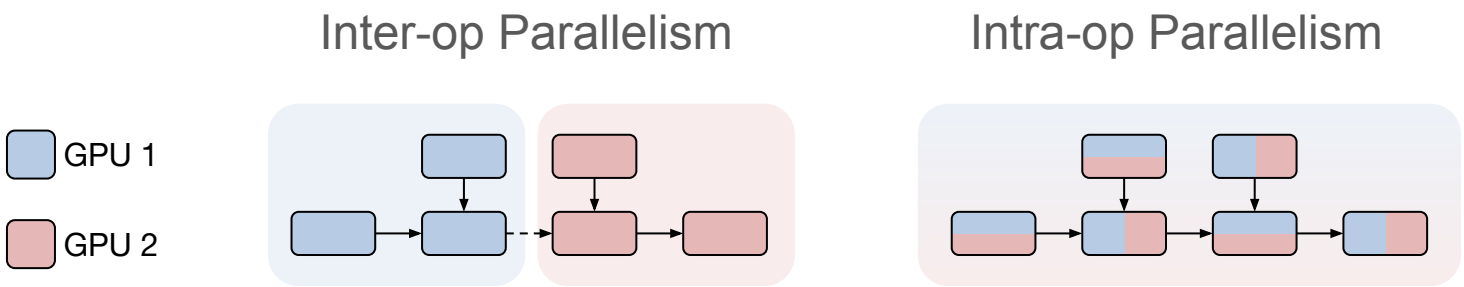
Problem 1: Prefill-Decoding Interference

Batching prefill and decoding phase together hurt both TTFT and TPOT

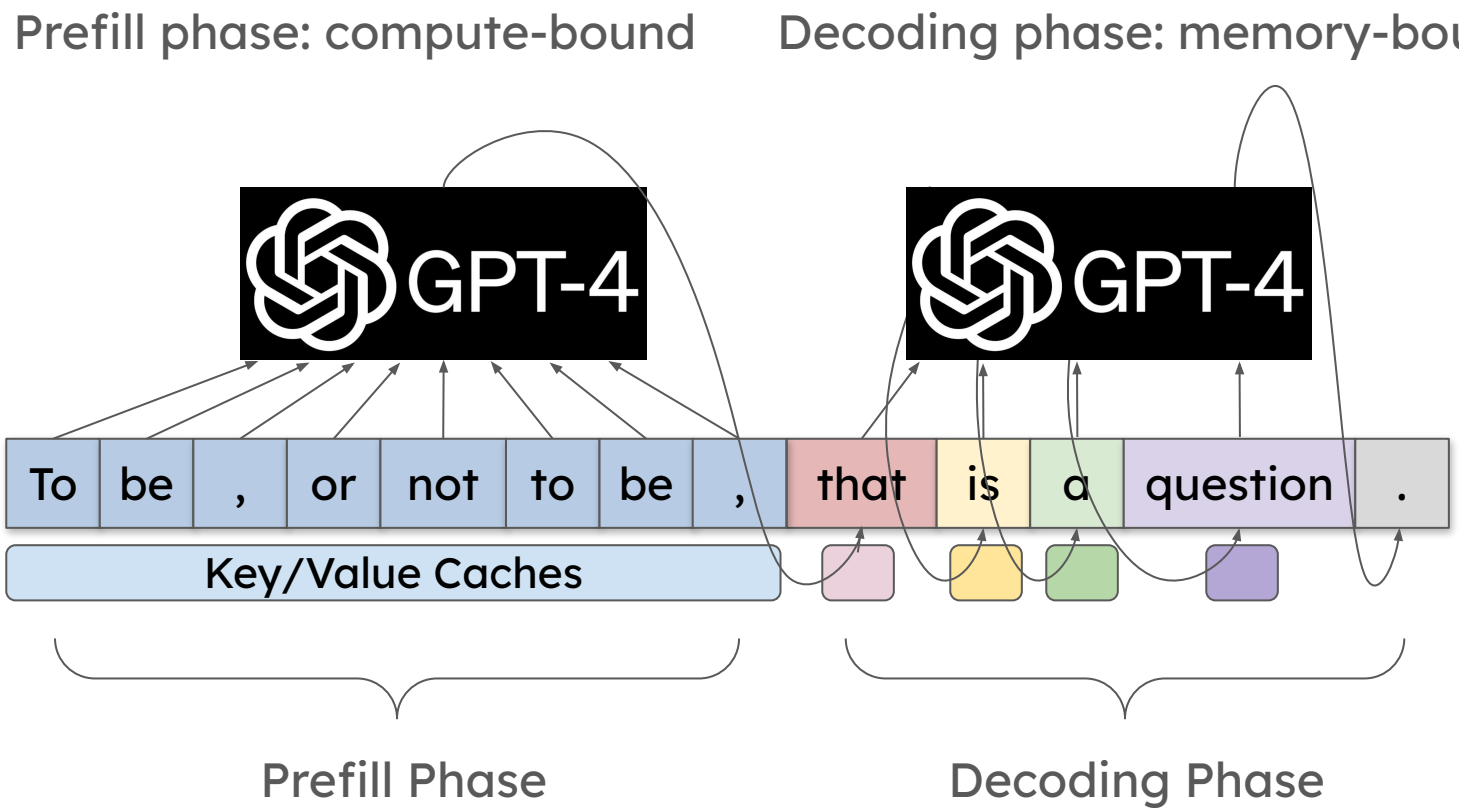


Problem 2: Resource and Parallelism Coupling

Batching the two phases make them share the same parallel strategy

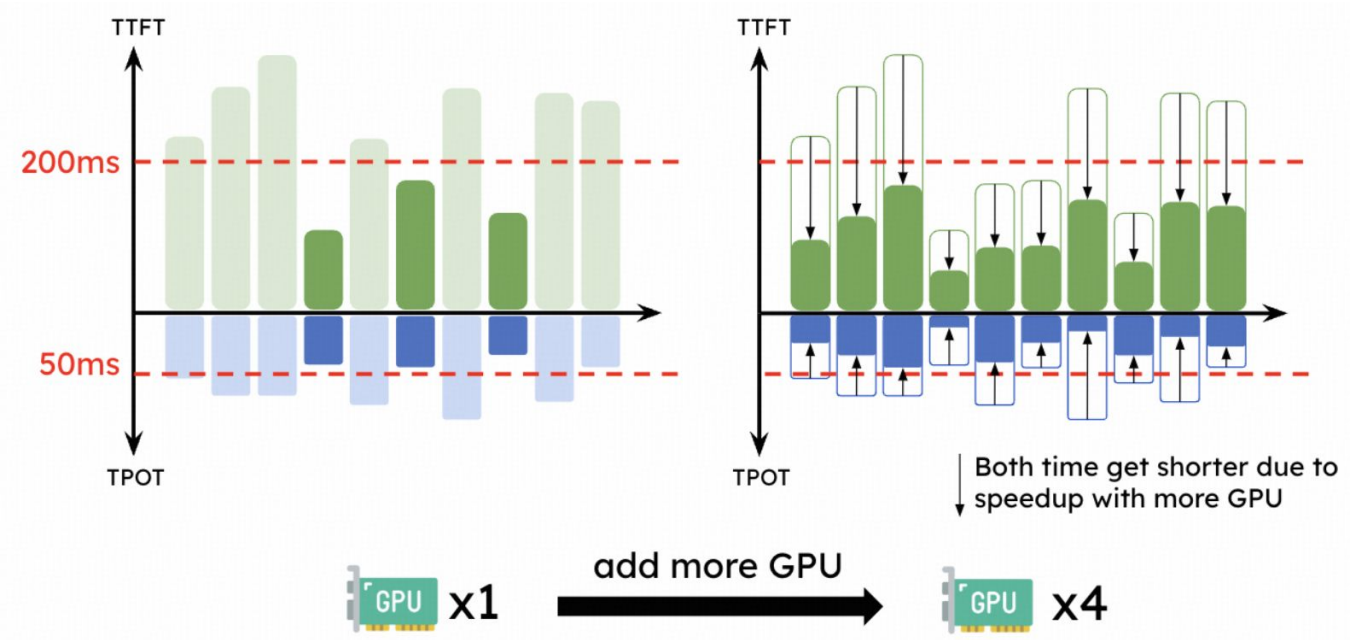


Problem 2: Resource and Parallelism Coupling



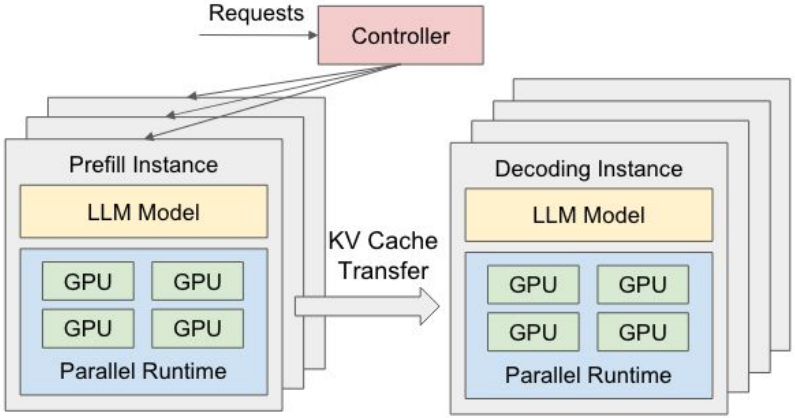
Problem 2: Resource and Parallelism Coupling

Coupling leads to overprovision resources to meet the more demanding SLO

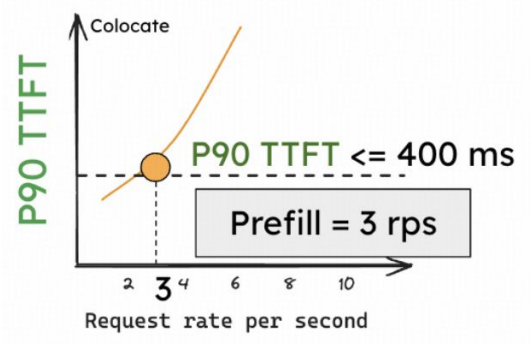


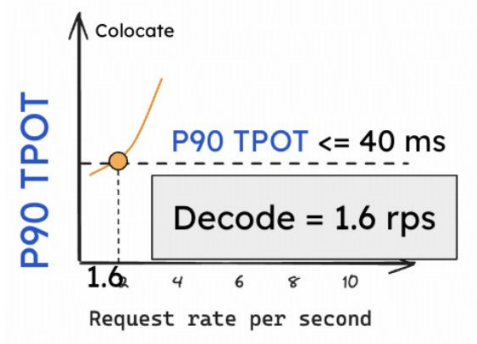
Opportunity: Disaggregating Prefill and Decoding

- Prefill-Decoding interference is immediately eliminated
- Naturally divide the SLO satisfaction problem into two optimizations:
  - Prefill instance optimizes for TTFT.
  - Decoding instance optimizes for TPOT.
  - Choose the most suitable parallelism and resource allocation for each phase.

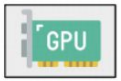


Opportunity: Disaggregting Prefill and Decoding





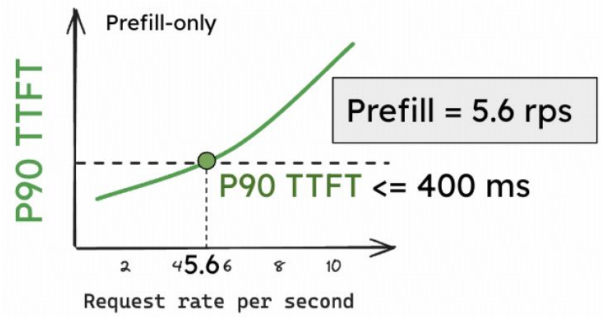
Colocation

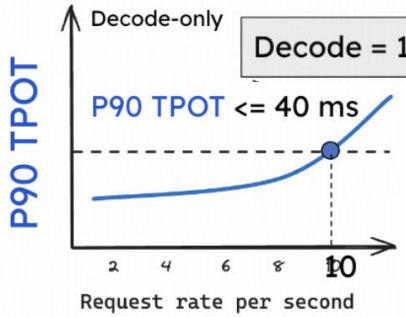


Max System rps

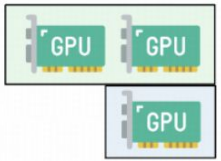
= Min(Prefill, Decode)

= 1.6 rps / GPU





Disaggregation (2P1D)



Max System rps

= Min (5.6 x **2**, 10) rps / 3 GPU

= 3.3 rps / GPU

Challenges of Disaggregation

- Communication overhead for KV-Cache transmission
- The optimization target — ***per-GPU goodput***, is difficult to optimize:
  - the workload pattern
  - SLO requirements
  - parallelism strategies
  - resource allocation
  - network bandwidth

DistServe Overview

Definition of Placement:

(1) parallelism strategy for prefill/decoding instance
(2) the number of each instance to deploy
(3) how to place them onto the physical cluster

Featured algorithms:

- Placement for High Node-Affinity Cluster
- Placement for Low Node-Affinity Cluster
- Online Scheduling Optimization

Placement for High Node-Affinity Cluster

Assumption:

- Nodes are connected with high bandwidth network, e.g., InfiniBand.

Observation:

- We can optimize prefill and decoding instances separately.

Algorithm Sketch:

- Use simulation to measure the goodput for a specific parallelism config.
- Obtain the optimal parallelism config for each phase.
- Use replication to match the overall traffic.

Placement for Low Node-Affinity Cluster

Assumption:

- GPUs inside one node are connected with NVLINK.

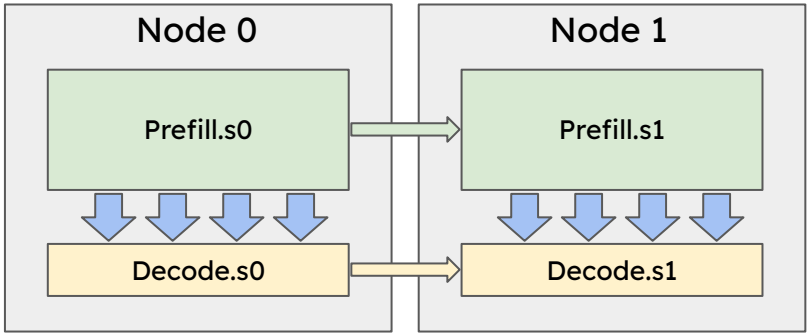
Observation:

- KV-Cache transmission only happens between ***the same layer.***

Algorithm Sketch:

- Similar to the previous one but add the constraint to require the same stage of prefill/decoding instances to be on the same node.

Example:



Online Scheduling Optimization

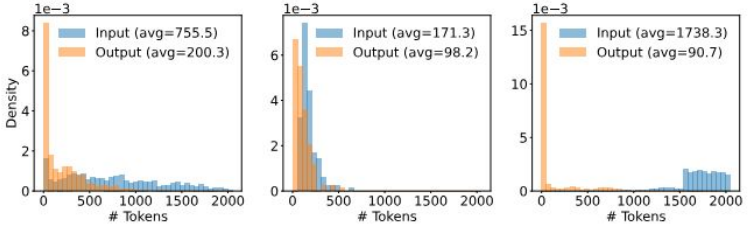
- Scheduling to reduce pipeline bubbles
- Combat workload burstiness
- Periodic Replaning

Evaluations

- Setup:
  - 4 x DGX-A100, each with 8 x NVIDIA A100-80GB
  - intra-node network bandwidth: 600GB/s
  - cross-node network bandwidth 25Gbps
- Workloads:

Application	Model Size	TTFT	TPOT	Dataset
Chatbot OPT-13B	26GB	0.25s	0.1s	ShareGPT [8]
Chatbot OPT-66B	132GB	2.5s	0.15s	ShareGPT [8]
Chatbot OPT-175B	350GB	4.0s	0.2s	ShareGPT [8]
Code Completion OPT-66B	132GB	0.125s	0.2s	HumanEval [14]
Summarization OPT-66B	132GB	15s	0.15s	LongBench [13]

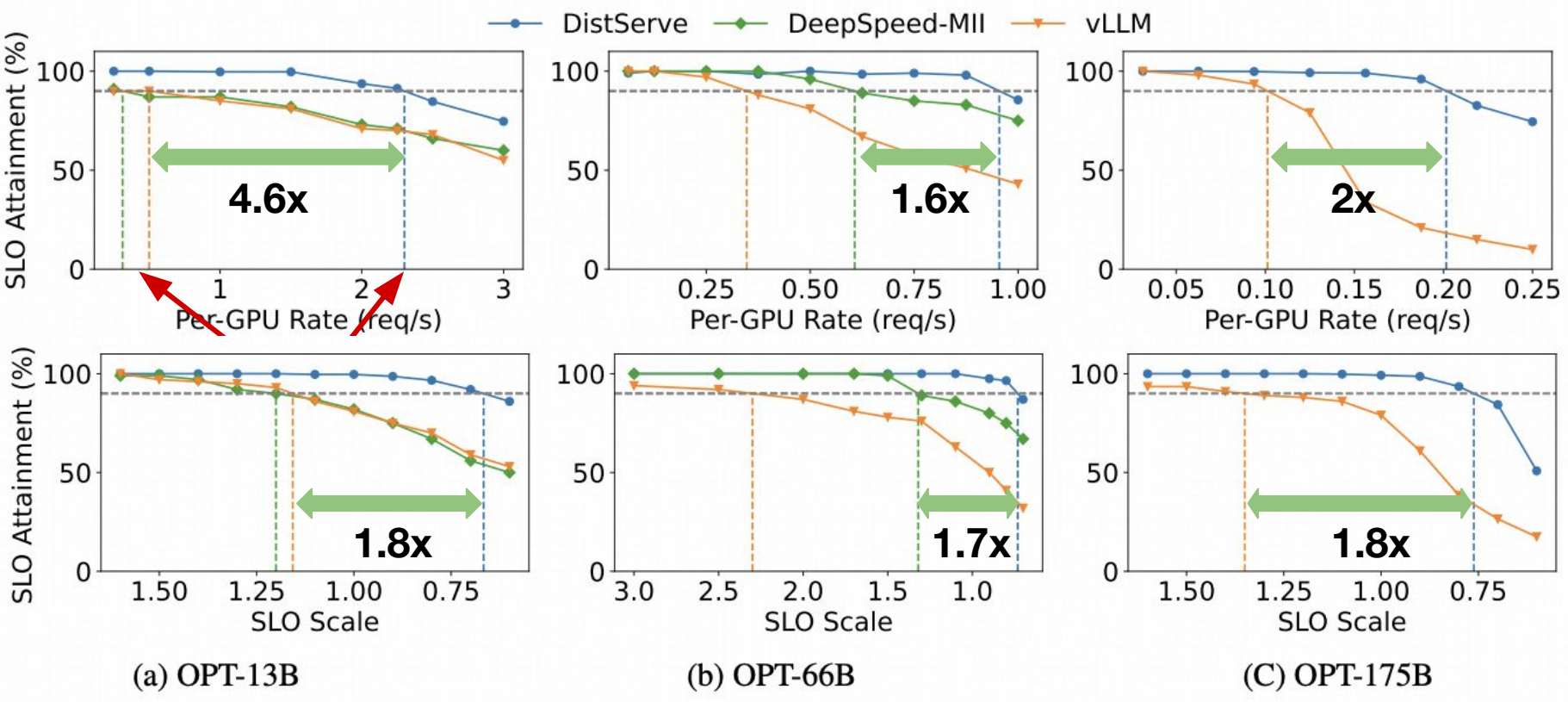
Table 1: Workloads in evaluation and latency requirements.



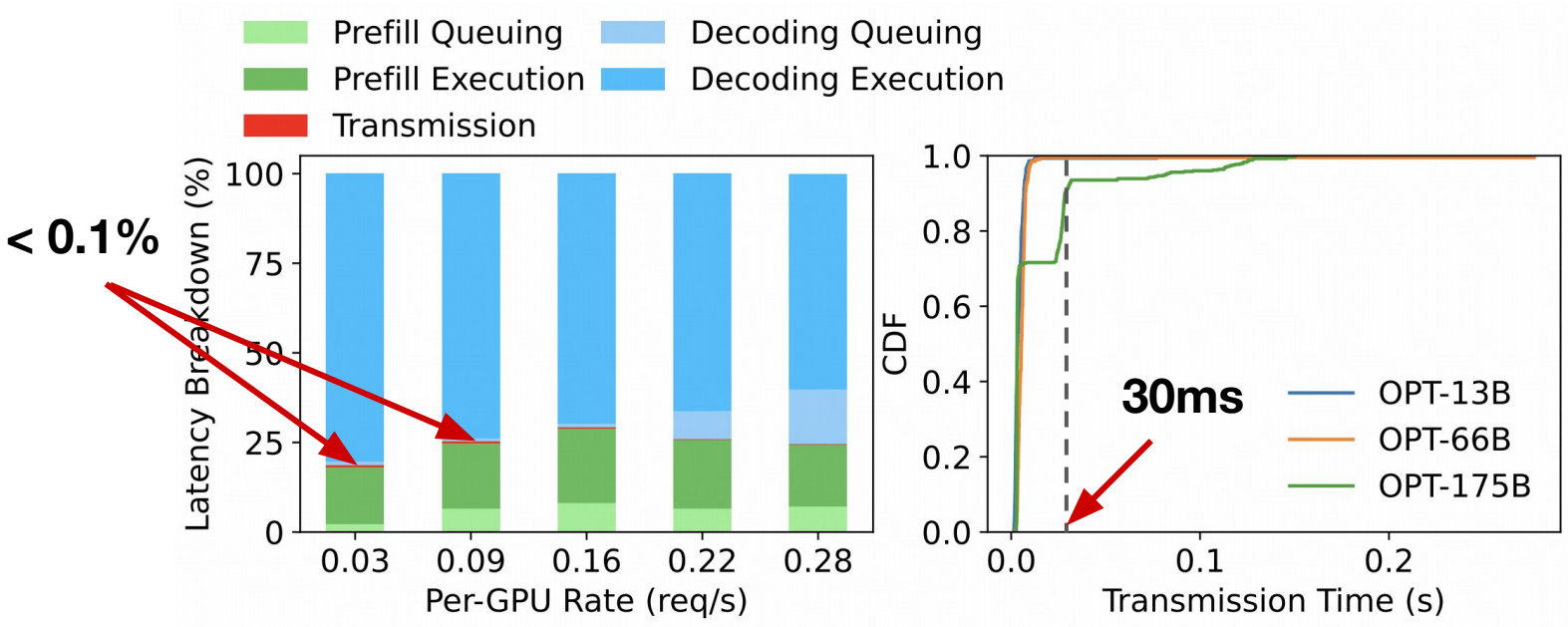
(a) ShareGPT (b) HumanEval (c) LongBench

- Metric: SLO Attainment
- Baseline: vLLM, DeepSpeed-MII

End-to-end Experiment – Chatbot



Latency Breakdown



Summary

- DistServe disaggregates prefill from decoding to serve LLM inference:
  - Bandwidth-aware placement algorithm to minimize communication overhead
  - Co-optimize the parallelism strategies for prefill/decoding instances
  - Online scheduling to handle real-world workload
- Experiments show that DistServe can serve 7.4x more requests or 12.6x tighter SLO, compared to SOTA systems, while staying within latency requirements for > 90% of requests.


https://github.com/LLMServe/DistServe


zhongyinmin@pku.edu.cn