A Artifact Appendix

A.1 Abstract

This artifact provides the source code, evaluation scripts, and instructions necessary to reproduce the end-to-end throughput evaluation (Figures 9, 10, 11, and 12) of DynaFlow.

A.2 Artifact check-list (meta-information)

- **Program:** DynaFlow, vLLM, SGLang, and HuggingFace Transformers.
- **Run-time environment:** Linux, Python 3.10+, PyTorch 2.8+, CUDA 12.8+.
- **Hardware:** Multi-GPU setups, specifically evaluated on a DGX B200 system with 8 NVIDIA B200 GPUs and 2 Intel Xeon 8570 CPUs.
- **Metrics:** End-to-end throughput (tokens/s, seq/s).
- **Output:** JSON files of performance metrics and generated visualization charts.
- **Experiments:** Offline inference and training throughput evaluations across varying batch sizes and sequence lengths.
- **How much disk space required (approximately)?:** 400 GB.
- **How much time is needed to prepare workflow (approximately)?:** 1-2 hours.
- **How much time is needed to complete experiments (approximately)?:** 12 hours.
- **Publicly available?:** Yes.
- **Code licenses (if publicly available)?:** MIT.
- **Archived (provide DOI)?:** We will provide this as soon as the artifact evaluation finishes.

A.3 Description

A.3.1 How delivered

The source code is provided via a public GitHub repository at `https://github.com/uw-syfi/DynaFlow`. This repository contains the DynaFlow core package, patches of the evaluated ML frameworks (vLLM, SGLang), and automated benchmarking scripts.

A.3.2 Hardware dependencies

An NVIDIA GPU environment is required. Replicating the exact evaluation result requires an NVIDIA DGX B200 node.

A.3.3 Software dependencies

The environment requires PyTorch 2.8+, CUDA 12.8+, and NVIDIA driver version 580+. Notice that the API `mark` in Figure 5 requires PyTorch 2.10+.

A.3.4 Data sets

The ShareGPT dataset is used to test diverse input and output lengths. Randomly generated datasets are used for fixed-size inputs and outputs.

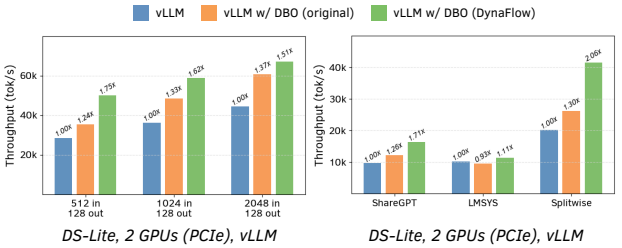

Figure 15. Serving throughput of DynaFlow-based DBO integration in vLLM under PCIe interconnect.

A.4 Installation

Please follow the instructions under `examples/ae` in the GitHub repository under branch `ae` to install DynaFlow and the modified systems.

A.5 Evaluation and expected result

The evaluation scripts will generate JSON files containing the throughput metrics in either token/s or seq/s under `examples/ae/results`. Then, users can execute the Python script under `examples/ae/plot` to visualize the results. Each script will save a PDF-format figure under the current directory.

B Low-Bandwidth DBO Performance

To further evaluate the practical benefits of DBO in communication-bottlenecked scenarios, which are common in multi-node deployments, we conducted an additional experiment. Due to hardware constraints preventing a direct multi-node evaluation, we simulated a low-bandwidth interconnect on our two-GPU setup by forcing all peer-to-peer communication over the PCIe bus instead of the high-bandwidth NVLink interconnect. This was achieved by configuring the relevant NCCL environment variables (`NCCL_P2P_DISABLE=1`).

The results are shown in Figure 15. With communication time significantly increased, our DynaFlow-enabled DBO integration achieved a throughput improvement of up to 2.06x over the baseline vLLM. vLLM's native DBO implementation also demonstrated significant gains, with up to a 1.37x speedup, validating the general utility of the DBO strategy in communication-constrained settings. However, its performance was still constrained by its static splitting policy. Finally, we note that the speedup on the LMSYS-Chat dataset remained low for both implementations, as its characteristically small batch sizes are not large enough for batch splitting to be beneficial.