A APPENDIX

A.1 Block Assignment Algorithm

Given a set of blocks B from different sequences, each block i may have a different computation cost c_i. This difference arises because, although Zig-Zag packing balances workloads among blocks from the same sequence, blocks from different sequences exhibit distinct arithmetic intensity, leading to varying compute usages.

Thus, FCP adopts a variant of the Longest Processing Time (LPT) scheduling algorithm (Chandran et al., 2024), to reduce workload imbalance. The problem is modeled as a multi-dimensional bin-packing problem, where the objective is to minimize the maximum computation usage among all workers, subject to a per-worker memory constraint M.

As shown in Algorithm 1, the blocks are first sorted by their compute and memory usages (line 2), and then greedily assigned to workers. At each iteration, the worker with the smallest current load is selected. The load is defined as a weighted sum of memory and compute usage, with α and β controlling the preference. Only workers that satisfy the memory constraint are eligible for selection (line 5).

Algorithm 1 Greedy Load-Balanced Assignment

input List of blocks B with memory usages m_i and compute amount c_i, number of workers N, weights α, β, and memory usage limit M, tolerant factor δ

1: Compute desired (average) memory and compute loads per worker: $\hat{m} \leftarrow \sum_i m_i/N$, $\hat{c} \leftarrow \sum_i c_i/N$
2: Sort blocks in descending order by $\max(m_i/\hat{m},\ c_i/\hat{c})$
3: Initialize empty workers $w_1, \ldots, w_N$ with zero workload $(m_w, c_w) = (0, 0)$
4: **for** each block B_i in descending order **do**
5: Find worker w^* with minimal load and below memory limit ($(m_{w*} + m_i) \leq (M * (1 + \delta))$):

$$w^* \leftarrow \arg\min_w \max\left(\alpha \cdot \frac{m_w + m_i}{\hat{m}},\ \beta \cdot \frac{c_w + c_i}{\hat{c}}\right)$$

6: Assign B_i to w^* and update (m_{w^*}, c_{w^*})
7: **end for**

A.2 Proof of Congestion-Free Solver

Given an undirected bipartite graph G with a maximum degree Δ, edge set E, and $2N$ vertices, the minimum number of disjoint graph matchings from a decomposition of E is denoted as Δ^*. As shown in Lemma 2, the lower bound of Δ^* is Δ. Thus, by constructing a decomposition of E into Δ disjoint matchings M, we can prove that $\Delta^* = \Delta$.

We first convert G into a Δ-regular bipartite graph $\hat{G}$ by greedily adding edges into E. Because the sum of degrees on the first N send nodes equals the sum of degrees on the N receive nodes, it is apparent that an edge $\hat{e}$ from the i-th send node to the j-th receive node can be inserted where both $\texttt{deg}(i) < \Delta$ and $\texttt{deg}(j) < \Delta$. We keep adding such edges, denoted as $\hat{E}$, until all nodes reach degree Δ. Therefore, the augmented graph $\hat{G}$ with edge set $E \cup \hat{E}$ becomes a Δ-regular bipartite graph.

By Hall's theorem (Cameron, 2025), every regular bipartite graph has a perfect matching. Hence, $\hat{G}$ can be decomposed into Δ perfect matchings $\hat{M}$ by recursively removing a perfect matching from $\hat{G}$. Finally, based on the perfect matchings $\hat{M}$, the original matching set M can be obtained by deleting all newly added edges $\hat{E}$. Proof completes.

A.3 Detailed Configurations of Baselines

Ring Attention. We implement Ring Attention on top of TransformerEngine (Sivamani et al., 2025), which follows a peer-to-peer communication schema. To enable computation and communication overlap, we adopt double buffering, which allows concurrent operations on two CUDA streams. Zig-Zag packing is enabled by default to maintain intra-sequence workload balance.

ByteScale. We reproduce the CP algorithm described in Algorithm 2 of their paper (Ge et al., 2025), also known as HDP-balanced, by referring to their private codebases. Excluding pipeline parallelism, HDP-balanced assigns sequences to workers in proportion to their context lengths. For instance, a sequence of length $k \cdot L$ is assigned to $k\times$ more workers than a sequence of length L. Within each partition, Ring Attention is applied recursively.

WLB-LLM. Since the official implementation (Wang, 2025) uses FlashAttention2, which performs suboptimally on latest Hopper and Blackwell GPUs, we reimplement WLB-LLM ourselves. We use the maximum performance of Ring Attention and ByteScale as an oracle version, where the online estimator is replaced with an oracle one. Note that per-sequence and per-document sharding correspond to ByteScale and Ring Attention, respectively.

MagiAttention. We directly adopt the official codebase implementation[4] (Zewei & Yunpeng, 2025). For benchmarking, we use the scripts provided by the authors (Tao & Huang, 2025). To enable communication and computation overlap, we manually set `CUDA_DEVICE_MAX_CONNECTIONS` to 8. We also mannuly disable `MAGI_ATTENTION_HIERARCHICAL_COMM` as it causes NCCL errors in our clusters.

[4]Commit hash: 3a96ab7

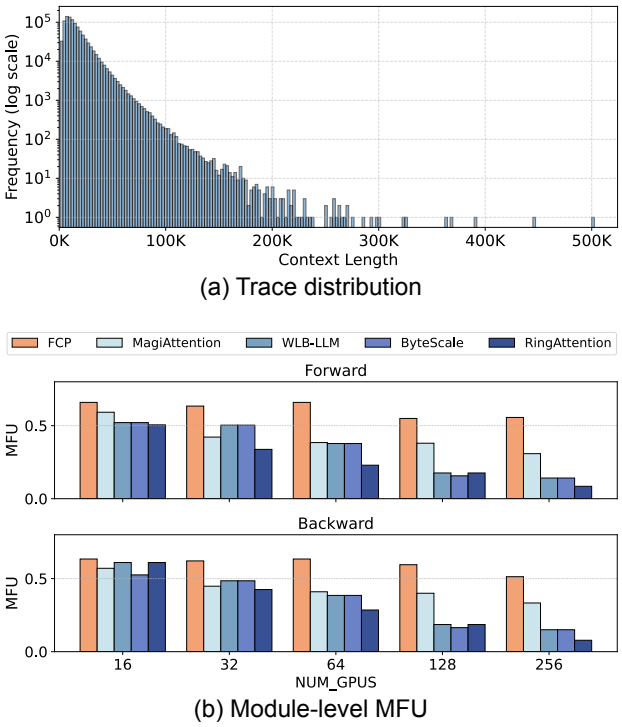

Figure 15. (a) Trace distribution of the *lognormal* distribution and (b) weak-scaling of module-level attention MFU. The number of tokens per GPU is fixed at 32K.

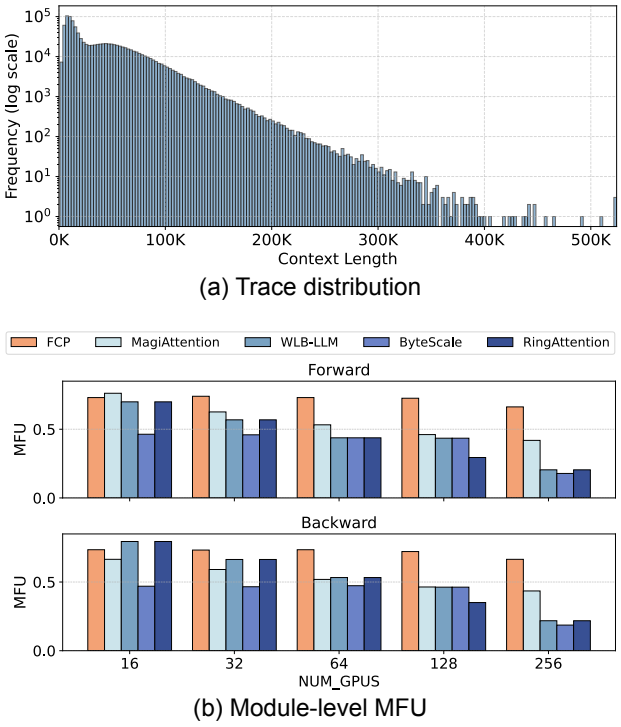

Figure 16. (a) Trace distribution of the *bimodal* distribution and (b) weak-scaling of module-level attention MFU. The number of tokens per GPU is fixed at 32K.

A.4 Evaluation on Additional Workloads

Besides the real-world distribution derived from our pre-training tasks Figure 2, we also construct two synthetic workloads to demonstrate the generality of FCP. One exhibits a less long-tailed distribution, and the other follows a bimodal pattern. Both are generated by sampling from lognormal distributions. The minimum and maximum sequence lengths are set between 1K and 512K. The remaining evaluation setup is identical to that in § 6.1.

Less long-tailed. We generate this workload using a lognormal distribution with a standard deviation of $s = 0.7$ and an expected sequence length of 16K. As shown in Figure 15 (a), the resulting trace is much more concentrated compared to Figure 2. As the extremely long context is rare in these cases, ByteScale performs better than Ring Attention due to less workload imbalance. All remaining methods perform worse as the total computation amount is smaller.

Bimodal. We synthesize this workload by combining two lognormal distributions with different mean sequence lengths to form a bimodal pattern. The two components are configured as $s = 0.5$ with an average of 16K, and $s = 0.5$ with an average of 64K, as illustrated in Figure 16 (a).