Project Proposal

TITLE: MPI-Based Multi-Node Tensor Parallelism
Team: Ruiqi Yang, Zijia Liu

URL: https://artificialricky.github.io/15618-final-project/

SUMMARY:
We will implement a multi-node tensor parallelism and focus on communication optimization with MPI. By replacing built-in collective operations with optimized AllReduce and Scatter-Reduce algorithms, we aim to gain fine-grained control over communication and explore strategies that overlap GPU computation with inter-node data exchange. This allows us to evaluate the correctness, memory efficiency, and performance of parallel attention mechanisms without requiring model training.

BACKGROUND:
In modern Transformer models, one of the biggest challenges during multi-node distributed inference is the communication overhead, which slows down system performance. This is especially true when tensor parallelism is used to split the model across multiple devices, as the devices need to share and combine data. For example, different GPUs might store pieces of the weights for a certain layer of the model, and they need to communicate to merge the results at each computation step. This communication can cause delays and waste resources in large-scale setups. This project aims to improve the way data is shared between devices during Transformer inference. We're developing custom methods, like AllReduce and Scatter-Reduce, based on MPI, to replace the standard communication methods used in existing libraries (like MPI_Allreduce). We plan to implement strategies like Ring AllReduce and Recursive Doubling and explore ways to overlap communication with GPU computation. By doing this, we hope to reduce the impact of communication delays. Our goal is to perform communication tasks, such as sending and receiving data, at the same time as the computation, so that they don't slow each other down.

THE CHALLENGE:
Attention mechanisms require global interaction between queries, keys, and values (three different matrices), which means that even when computation is distributed, nodes must frequently exchange data—creating potential communication bottlenecks. Unlike operations such as convolution, attention cannot be easily split into fully independent parts because each position in the sequence will depend on all others.

In addition, varying sequence lengths may lead to imbalanced workloads across processes, which reduces overall efficiency. To address this, we will explore optimization strategies to mitigate the resulting bottlenecks.

We hope to gain deeper insight into the factors that impact communication bottlenecks and bandwidth usage, helping us build intuition for optimizing communication. At the same time, we want to enhance our understanding of tensor parallelism and explore more efficient model partitioning strategies to reduce the performance impact of communication bottlenecks like AllReduce during inference.

RESOURCES:
We will use GPU resources provided by GHC and PSC for this project, and we will also attempt to request access to university-provided GPU nodes or use the GPU on a personal laptop when necessary. All computing environments should support MPI-based communication. For code, we will modify existing open-source language models, such as Meta's LLaMA model, and replace the original attention mechanism with our custom parallel attention variants, such as Ring Attention. Since our project focuses on inference only and does not require training, our demand for computational resources is relatively low.

References:
Vaswani et al., 2017. Attention Is All You Need.
Shoeybi et al., 2019. Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism
Dao et al., 2022. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness
Touvron et al., 2023. LLaMA: Open and Efficient Foundation Language Models.

GOALS AND DELIVERABLES:
PLAN TO ACHIEVE：
Our primary goal is to implement two custom collective communication strategies—Ring AllReduce and Recursive Doubling—to replace built-in MPI collectives in the context of transformer inference with tensor parallelism. We aim to integrate these into the attention computation pipeline and compare their performance (latency, bandwidth usage, and memory footprint) against standard MPI AllReduce. A successful baseline would demonstrate that our custom implementations match or outperform built-in collectives in controlled test cases

HOPE TO ACHIEVE:
we hope to extend our implementation to support multiple input sequences in parallel, simulating pipeline parallelism at the inference level. We also hope to explore simple dynamic workload balancing techniques to address sequence-length variability across nodes.

PLATFORM CHOICE:
We chose to implement our system on GPU- and MPI-enabled cluster platforms (e.g., GHC and PSC resources), as it involves both GPU-intensive computation and inter-node communication.

We will use either C++ or Python to implement the communication and model inference tasks, depending on which LLM framework provides better abstraction. This will allow us to implement AllReduce and Scatter-Reduce communication strategies while more conveniently distributing and loading model parameters across the cluster and managing tensors efficiently.

SCHEDULE:

- **Mar 25 – Mar 31**
  Study communication mechanisms in Megatron-LM, FlashAttention, and Ring Attention; set up the development environment.
- **Apr 1 – Apr 7**
  Complete baseline tests for AllReduce.

- **Apr 8 – Apr 14**
  Implement a minimal MPI communication framework and verify both Ring AllReduce and Recursive Doubling;
  Integrate Ring AllReduce into the attention inference pipeline;
  Compare MPI_Allreduce with the custom Ring implementation;
  Prepare materials for the midpoint presentation.

- **Apr 15 (Midpoint Milestone)**
  Submit baseline implementation and preliminary evaluation (correctness + throughput comparison).

- **Apr 15 – Apr 21**
  Extend support for multiple input sequences in parallel;
  Analyze load imbalance across sequences and explore basic dynamic scheduling strategies.

- **Apr 22 – Apr 28**
  Complete performance testing of all communication strategies;
  Write the final report and prepare the project poster.