introduction

- context lengths are really large now.
- It might help to show that a 1.5x training speed reduction can result in x$ of savings.
- quick primer on 4D parallelism
- use a long sequence to highlight heterogeneity.
- there is inter-batch imbalance and intra-batch imbalance.
- shuffling may resolve the former, but not the latter.

introduction

- you want to train LLMs
- you get a lot of GPUs, to train in parallel
- it takes you 40hours to train a model at a whopping $2million. (or you could use aws costs)
- if it had taken you 30 hours instead, a 1.3x reduction you would have saved $xx in cloud costs.

background

let's see how these models are trained,

- firstly, quick model architecture in discussion (dense GPT-2 style architecture)
  - big model.
  - self-attention
    - builds up understanding of the sequence of tokens.
    - it is important to only look at your own past history, so when packing documents into "one long sequence", you use masking.
    - attention has a lot of interesting properties, the important one for our discussion today is that later tokens have a longer history to look at.
  - linear layers
    - they don't have any odd properties, just good old matrix multiplication (TM).
- because these models are so big, you want to parallelize training over a lot of GPUs

baseline approach: attention aware packing.

algorithm

- use causal mask as a proxy for the attention workload.
- minimize the maximum mini-batch workload.
- solve via ILP

tradeoff analysis

- increasing this to across micro-batches should even further reduce imbalance, of course you have more to balance across.
- however, leads to loss of model quality since you have messed with the randomness of the training process.
- fixed-4d in evaluations compares against this baseline, it is not that much better.

Section 4: improved algorithm.

this is the main contribution, they propose a packing algorithm that combines:

1. variable-length micro-batch packing.
2. outlier document detection method.

1. variable-length micro-batches

- earlier, they had claimed that 4D parallelism training libraries enforced each micro-batch to be of context-length size.
- they relax this constraint for their approach (but don't hint at what underlying library changes this might need)
- they observe that it is impossible to balance micro-batches, if your global batch contains a really long document (because attention is quadratic)
- so, you could instead combine a lot more smaller documents, such that the length

general presentation notes

- include piazza questions naturally in the flow of the presentation if highly aligned.
- include results naturally in the flow if it helps.

Section 5: Improved CP sharding

algorithm

- The problem with standard CP sharding is that it only guarantees equal number of tokens assigned to each CP worker, which might not result in balanced attention computation.
- Instead, apply this approach for each document within your sequence. CP_0 contains the first and last index across all documents.
- This should yield a more balanced workload across multiple CP workers.
- They also implement an optimization to avoid padding tokens, by distributing the remainder across multiple workers.

```
A concrete example (CP=2)

Say a packed sequence has two docs:
Doc0 length = 4096 (long)
Doc1 length = 256 (short)
```

Section 6. Evaluation

- their contributions are in PP and CP balancing. Let's think, when would PP and CP imbalance hurt most?
    - CP: longer context, more chance of long document being assigned to imbalanced worker.
    - PP: larger model, micro-batch imbalance adds more to the latency.
- figure 12.
    - long context model benefits more from wlb for the same model size compared to baseline at the same context.
    - but, larger models see lesser improvement over baseline (why?)
        - This is because larger models involve more GPUs for training, which increases the ratio of communication latency to computation latency
        - making the impact of workload imbalance in the attention layer less