**Efficiently Scaling Transformer Inference**

**Problem to Tackle**

Efficient inference for large language models (LLMs) with tight latency constraints and long sequence lengths.

**Impact**

LLMs have broad practical utility across many applications, but their efficient deployment remains challenging due to the sequential nature of generative inference. Developing engineering principles to serve large-scale Transformer models efficiently is crucial to unlocking their full potential and making them more practical for real-world use.

**Proposed Idea**

1. Introduce an abstract partitioning framework that analytically determines the optimal multi-dimensional partitioning strategy across multiple TPUs, based on the model's size and specific application requirements.
2. Reduces memory overhead by allowing multiple query heads to share the same key/value heads, significantly lowering memory consumption for the KV cache.

**Summary of Proposed Technique**

1. Feedforward layer partition
2. 1D weight-stationary: Partitions weights only across the feedforward dimension (d\_ff). This baseline approach results in constant communication time regardless of chip count. Used as reference.
3. 2D weight-stationary: Partitions weights across both the feedforward (d\_ff) and model dimensions (d\_model), which reduces the amount of data shared between chips. Communication time scales down as the number of chips increases, improving efficiency.
4. Weight-gathered: Instead of gathering activations only, this method also gathers weights while keeping most activations stationary. This approach becomes more efficient when either the batch size or sequence length is large, reducing communication costs.
5. Multiquery attention: Instead of assigning each query head its own key and value heads, multiple query heads share a single key and value head. This significantly reduces the memory required to store the KV cache and the time to load it into the TPU, improving both memory efficiency and speed during inference.

**Strengths and Weaknesses**

Strengths:

1. Significantly improves latency and FLOPs utilization with the proposed partitioning strategy.
2. Reduces KV cache size, enabling the system to handle much longer context lengths and generate longer output.

Weakness:

1. Bandwidth modeling may be overly simplified, as TPU’s 3D torus interconnect has non-uniform bandwidth between different node pairs.
2. To achieve optimal communication times, the XYZ dimensions of the TPU interconnect need to be adjusted based on feedforward and model dimensions. However, this is impractical in a real-world cluster, as it would require physically reconfiguring the TPU interconnects.
3. The evaluation focuses solely on a 3D torus interconnect topology.

**Room for Improvement**

1. In addition to weight quantization, applying quantization to activations could further enhance performance and resource utilization.
2. Extending the method to other hardware platforms with different interconnect topologies.