HW3

Tguo67

Question(s):
Read the paper of FlashInfer, and answer the following questions:
1. How does the BSR format unify the data structures? Explain and compare it with PageAttention of vLLM.
2. How does the load-balanced scheduling work?

Related links:
Paper of FlashInfer: https://arxiv.org/pdf/2501.01005Links to an external site.
FlashInfer Git Repo: https://github.com/flashinfer-ai/flashinferLinks to an external site.
FlashInfer Website: https://flashinfer.ai/Links to an external site.

Answers:

1. **How does the BSR format unify the data structures?**
   FlashInfer stores the KV cache in a block-sparse row (BSR) matrix and treats each attention read as a block-sparse FlashAttention. KV layout (paged tables, radix trees, tree-masks, etc.) becomes just a different sparse indexing of the same BSR view. Queries/outputs are packed as ragged tensors; keys/values are inserted into a BSR KV with app-chosen block sizes. This enables one block-sparse FlashAttention kernel to serve many layouts.

   Explain and compare it with PageAttention of vLLM:
   PagedAttention (vLLM) is first and foremost a memory-management abstraction: split KV into fixed-size blocks ("pages"), store them non-contiguously, and dereference via a block table.

   FlashInfer's BSR is a compute-first unification: treat the page table (and other layouts) as a *sparse matrix structure* so the *same* block-sparse FlashAttention pipeline handles PageAttention, radix trees, tree decoding, importance masks and so on.

2. **How does the load-balanced scheduling work?**

FlashInfer separates compile-time tile-size selection from runtime scheduling and keeps kernels CUDAGraph-compatible. Based on the greedy algorithm, define a tile cost, compute a max KV chunk size, split KV into chunks and also sort chunks by length. Then use min heap based on the accumulated cost.

Attention kernel emits partial outputs per chunk; a contraction step composes them using the attention-state operator; plans are cacheable across similar ops and kept CUDA-Graph compatible via fixed workspace addresses.