PR #33137 完整报告

sgl-project/sglang

[CP] Fuse zigzag attention into a single call

合并时间：2026-08-03 11:46

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33137

执行摘要

- 一句话：融合 zigzag CP 两次 TRT-LLM 注意力调用为一次
- 推荐动作：值得精读。它展示了把两次 CUDA 内核调用融合为一次的标准手法：构造合成 batch 几何（combined cu_seqlens）加行复制 page table，并在策略层用后端类型分支保护原路径。对后续 CP 性能优化和大型 PR 拆分都有参考价值。重点关注 zigzag.py 的 run_attention 分流与 trtllm_mha_backend.py 的 _trtllm_context_attn 参数扩展。

功能与动机

PR body 明确这是 #32714 的 Split PR 3，目标是把 zigzag context parallelism 中两次 TRT-LLM attention 调用融合为一次独立优化。profile 显示该方法可将每 rank 的 TRT-LLM context-attention 内核数从 72 降到 36（四 rank 288→144），即在 prefill 关键路径上减少约一半内核启动开销。同时作者强调保留原始非 TRT-LLM zigzag 路径不动，以控制回归风险。

实现拆解

1. 扩展 TRTLLMMHAMetadata 与 page table 准备逻辑（python/sglang/srt/layers/attention/trtllm_mha_backend.py）：新增 zigzag_page_table / zigzag_swa_page_table 字段；新增 _maybe_build_cp_zigzag_page_tables，在 init_forward_metadata 填充常规 page table 后通过 torch.cat 把 page table 按行复制一份，形成 2 * batch_size 行合成 batch 所需表。
2. 给闭包 _trtllm_context_attn 增加 cu_seqlens_kv 关键字参数与 use_zigzag_page_table 开关：开关打开时改用 zigzag（或 zigzag SWA）page table，但仍复用同一份 kv_cache 与 workspace。
3. 组合几何张量（python/sglang/srt/layers/cp/zigzag.py）：ZigzagContextParallelMetadata 新增 actual_seq_q_combined_tensor / kv_len_combined_tensor / cu_seqlens_q_combined_tensor / cu_seqlens_kv_combined_tensor / max_seqlen_q_combined；build_metadata 在原来分别构造 prev/next 张量的基础上，把两组列表直接拼接并做前缀和，得到合成 batch 的 cu_seqlens，推导结果直接落到 GPU 张量。
4. run_attention 分流（zigzag.py）：当 attention_backend == TRTLLM_MHA 时，用 q[:logical_tokens] 一次性调用 attn_fn，传入 combined 张量与 use_zigzag_page_table=True；其他后端走原有两次调用分支，prev_kwargs / next_kwargs 逻辑原样保留。

5. 测试配套：test/registered/cp/test_cp_strategy_unit.py 新增 test_zigzag_combined_attention_matches_two_half_reference，用 CPU 参考注意力对比“两半分别计算”与“组合张量单次计算”在 4 个 rank 上输出一致；按 review 要求删除 test_trtllm_mha_zigzag.py 及其他新增单测，收敛为一个聚焦用例。无配置、schema 或部署改动。

关键文件：

- python/sglang/srt/layers/cp/zigzag.py（模块 CP 策略；类别 source；类型 core-logic；符号 run_attention, build_metadata, ZigzagContextParallelMetadata）：核心策略改动：build_metadata 新增 combined 几何张量，run_attention 对 TRTLLM_MHA 后端改走单次融合调用，并保留非 TRTLLM 原始路径。
- python/sglang/srt/layers/attention/trtllm_mha_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 _maybe_build_cp_zigzag_page_tables, TRTLLMMHAMetadata, _trtllm_context_attn）：TRT-LLM 后端适配：metadata 新增 zigzag page table 字段，新增 _maybe_build_cp_zigzag_page_tables, _trtllm_context_attn 支持 use_zigzag_page_table 切换。
- test/registered/cp/test_cp_strategy_unit.py（模块 CP 单测；类别 test；类型 test-coverage；符号 test_zigzag_combined_attention_matches_two_half_reference, reference_attention）：聚焦单测：用 CPU 参考注意力验证组合几何与两半分别计算的结果一致，覆盖全部 4 个 rank。

关键符号：run_attention, build_metadata, _maybe_build_cp_zigzag_page_tables, _trtllm_context_attn, test_zigzag_combined_attention_matches_two_half_reference, reference_attention

关键源码片段

python/sglang/srt/layers/cp/zigzag.py

核心策略改动：build_metadata 新增 combined 几何张量，run_attention 对 TRTLLM_MHA 后端改走单次融合调用，并保留非 TRTLLM 原始路径。

```
# build_metadata 中新增的组合几何：在原有 prev/next 张量之外，
# 把两组列表“首尾相接”，得到一个 2 * bs 行的逻辑 batch。
# 这样一次 TRT-LLM context attention 调用就能同时处理两半。
actual_seq_q_combined_list = actual_seq_q_prev_list + actual_seq_q_next_list
kv_len_combined_list = kv_len_prev_list + kv_len_next_list
cu_q_combined = [0] + list(accumulate(actual_seq_q_combined_list))
cu_kv_combined = [0] + list(accumulate(kv_len_combined_list))

# run_attention 分流：只有 TRTLLM_MHA 后端走融合单调用；
# 其他后端保持原有两次调用，避免引入回归。
if attention_backend == CPAttentionBackendKind.TRTLLM_MHA:
    result = attn_fn(
        q[:logical_tokens],
        meta.cu_seqlens_q_combined_tensor,
        meta.kv_len_combined_tensor,
```

```
            meta.max_seqlen_q_combined,
            cu_seqlens_kv=meta.cu_seqlens_kv_combined_tensor,
            use_zigzag_page_table=True,
        )
    else:
        result_prev = attn_fn(
            q_prev,
            meta.cu_seqlens_q_prev_tensor,
            meta.kv_len_prev_tensor,
            meta.max_seqlen_q_prev,
            **prev_kwargs,
        )
        result_next = attn_fn(
            q_next,
            meta.cu_seqlens_q_next_tensor,
            meta.kv_len_next_tensor,
            meta.max_seqlen_q_next,
            **next_kwargs,
        )
        result = torch.cat([result_prev, result_next], dim=0)
```

python/sglang/srt/layers/attention/trtllm_mha_backend.py

TRT-LLM 后端适配：metadata 新增 zigzag page table 字段，新增 _maybe_build_cp_zigzag_page_tables，_trtllm_context_attn 支持 use_zigzag_page_table 切换。

```
# TRTLLMMHAMetadata 新增的 zigzag 专用字段：
# CP-v2 的 zigzag 策略把 prev/next 两半当作一个 2 * batch_size 的合成 batch，
# 因此需要行数翻倍的 page table（SWA 表同理）。
zigzag_page_table: torch.Tensor = None
zigzag_swa_page_table: torch.Tensor = None

# 仅在 CP-v2 zigzag 激活时为合成 batch 准备行数翻倍的 page table。
def _maybe_build_cp_zigzag_page_tables(
    self,
    metadata: TRTLLMMHAMetadata,
    forward_batch: ForwardBatch,
) -> None:
    """为组合的 prev-then-next CP 单次调用复制请求行。"""
    if not is_cp_v2_active(forward_batch):
        return

    # TODO: 避免物化重复 page table，降低 zigzag CP 的 page table 显存占用。
    metadata.zigzag_page_table = torch.cat(
        (metadata.page_table, metadata.page_table), dim=0
    )
    if metadata.swa_page_table is not None:
        metadata.zigzag_swa_page_table = torch.cat(
            (metadata.swa_page_table, metadata.swa_page_table), dim=0
```

```
        )

    # _trtllm_context_attn 闭包新增 use_zigzag_page_table 开关：
    # 合成 batch 场景下切换到行数翻倍的表，SWA 层则使用对应的 zigzag SWA 表。
    def _trtllm_context_attn(
        q_chunk,
        cu_seqlens_q,
        cache_seqlens,
        max_seqlen_q,
        *,
        cu_seqlens_kv,
        use_zigzag_page_table=False,
    ):
        block_tables = page_table
        if use_zigzag_page_table:
            block_tables = self.forward_metadata.zigzag_page_table
            zigzag_swa_pt = self.forward_metadata.zigzag_swa_page_table
            if zigzag_swa_pt is not None:
                _, is_swa = self._swa_kv_pool.layers_mapping[layer.layer_id]
                if is_swa:
                    block_tables = zigzag_swa_pt
        return flashinfer.prefill.trtllm_batch_context_with_kv_cache(
            query=q_chunk,
            kv_cache=kv_cache,
            workspace_buffer=self.workspace_buffer,
            block_tables=block_tables,
            seq_lens=cache_seqlens,
            max_q_len=max_seqlen_q,
            max_kv_len=self.max_context_len,
            batch_size=cu_seqlens_q.shape[0] - 1,
            cum_seq_lens_q=cu_seqlens_q,
            cum_seq_lens_kv=cu_seqlens_kv,
            window_left=layer.sliding_window_size,
            sinks=attention_sink,
            skip_softmax_threshold_scale_factor=envs.SGLANG_SKIP_SOFTMAX_PREFILL_
            THRESHOLD_SCALE_FACTOR.get(),
            out_dtype=self.q_data_type,
        )
```

评论区精华

Fridge003 在 self-review 中对代码提出 6 条意见，最终版本全部落实：

> “prev_kwargs and next_kwargs shouldn't be removed when attn backend is not trtllm_mha. Don't modify the logic of original path.” 结论：run_attention 采用 if/else 分支隔离，TRTLLM_MHA 走融合调用，其余后端保留两次调用与原 kwargs。“We only need this unit test. Please delete any other unit tests added in this PR.” 结论：删除整个 test_trtllm_mha_zigzag.py 及其他新增单测，只保留 test_zigzag_combined_attention_matches_two_half_reference。“Rename this

function to _maybe_build_cp_zigzag_page_tables." / "No need to create thisfunction. The logics can be put under _trtllm_context_attn inline." 结论：函数保留但更名为 _maybe_build_cp_zigzag_page_tables，内联建议未完全采纳。"Add a TODO onwe plan to save the memory usage for zigzag page table." 结论：已添加 TODO，注释说明避免物化重复 page table 以降低显存。 另外，PR Test CI 状态为失败，/rerun-test补充了 4 个 CP 用例，其中 test_mimo_cp.py（8-gpu-b200）仍失败，其余 3 个通过。

- 保留非 TRT-LLM 原始路径逻辑 (design): 最终 run_attention 用 if/else 分支：TRTLLM_MHA 走融合单调用，其余后端保留两次调用与原 kwargs。
- 精简测试范围，只保留一个聚焦单测 (testing): 已删除额外测试文件与多余单测，最终只保留一个 CPU 参考对比用例，PR 收敛为 3 个文件。
- 函数命名与内联取舍 (style): 函数保留但更名为 _maybe_build_cp_zigzag_page_tables，内联建议未完全采纳，独立函数在 init_forward_metadata 中调用。
- zigzag page table 显存占用 (performance): 已添加 TODO 注释，说明避免物化重复 page table 以降低显存占用。

风险与影响

- 风险：回归风险：run_attention 是 zigzag CP 各类注意力后端的公共入口，虽然非 TRTLLM 分支保持原逻辑，但 metadata 新增了 combined 张量字段，任何初始化遗漏都会在张量构建时报错而非静默错误，风险集中在 TRTLLM 分支。 性能与资源：每次 prefill forward 都会用 torch.cat 复制 page table，行数翻倍，带来一次设备端分配与拷贝；作者已留 TODO。对 page 数很大的长序列，显存开销值得关注。 正确性覆盖：新增单测在 CPU 参考上验证组合几何与两半一致，但没有直接验证 flashinfer 内核调用与 zigzag_page_table 的实际绑定；GPU 端正确性依赖既有 CP 用例，且 CI 中 test_mimo_cp.py（8-gpu-b200）失败，需要确认是否与本 PR 相关。 兼容性：仅当 is_cp_v2_active 且 TRTLLM_MHA 后端时走新路径，其他组合不受影响。
- 影响：性能影响：Blackwell 上 zigzag CP prefill 的 TRT-LLM context attention 内核启动次数减半，长序列预填充场景受益。 代码结构影响：zigzag 策略与 TRTLLM 后端新增耦合（use_zigzag_page_table 参数穿过 attn_fn 接口），但通过注意力后端类型分支隔离，保持非 TRTLLM 路径不变。 团队与流程影响：这是 #32714 系列拆分的第 3 个 PR，确立了"每个拆分独立优化 + 保留原路径 + 单一聚焦测试"的模式，对后续拆分有示范作用。
- 风险标记：核心路径变更（CP prefill 注意力），page table 显存翻倍（有 TODO），CI 部分失败（test_mimo_cp 8-gpu-b200），GPU 端正确性依赖既有测试

关联脉络

- PR #32714 Upstream parent PR: 本 PR 是 #32714 的第 3 个拆分 PR，原始性能 profile 与端到端测量数据记录在 #32714 中。