PR #26292 完整报告

sgl-project/sglang

Zero `req_pool_indices` padding in cuda-graph populate

合并时间：2026-05-25 18:29

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26292

执行摘要

- 一句话：修复 CUDA Graph 填充行 req_pool 索引脏数据问题
- 推荐动作：推荐合并。这是一个简洁且正确的 bugfix，与已有的 Eagle draft 做法保持一致。值得关注的是，设计上保留 slot 0 作为全零占位的约定，后续开发中应注意维护此约定。

功能与动机

PR body 指出：在 CUDA Graph populate 时只对 seq_lens 和 out_cache_loc 进行了清零 / 填充，但遗漏了 req_pool_indices。当批次发生 padding 时，padding 行的 req_pool_indices 残留在之前 replay 写入的值，导致注意力计算的 dummy read 可能读到 stale 的 req_to_token 行，产生错误。Eagle draft runner 已经做了同样的配对处理，这次将修复扩展到其余 4 个 runner。

实现拆解

1. cuda_graph_runner.py：在 populate_from_forward_batch 方法的 padding 分支（if bs != raw_bs）中，在 seq_lens.fill_(seq_len_fill_value) 和 out_cache_loc.zero_() 之后添加 self.req_pool_indices.zero_()。
2. cpu_graph_runner.py：在 prepare_replay 方法的 padding 分支中，在 captured_forward_batch.seq_lens.fill_(self.seq_len_fill_value) 和 captured_forward_batch.out_cache_loc.zero_() 之后添加 captured_forward_batch.req_pool_indices.zero_()。
3. eagle_draft_extend_cuda_graph_runner.py：在 replay 方法的 padding 分支（if bs * self.num_tokens_per_bs != num_tokens）中，在已有的 seq_lens.fill_、out_cache_loc.zero_、positions.zero_ 之后添加 buffers.req_pool_indices.zero_()。
4. frozen_kv_mtp_cuda_graph_runner.py：在 replay 方法的 padding 分支（if bs != raw_bs）中，在已有的 seq_lens.fill_、positions.zero_ 之后添加 buffers.req_pool_indices.zero_()。
5. 所有变更均紧跟在对应的 padding 条件块内，与已有的清零操作并列，且新增了注释说明意图。

关键文件：

- python/sglang/srt/model_executor/cuda_graph_runner.py（模块 调度器；类别 source；类型 data-contract；符号 populate_from_forward_batch）：核心 CUDA Graph runner

，主 populate 路径。本次修复在此新增 self.req_pool_indices.zero_()，与已有的 seq_lens fill 和 out_cache_loc zero 配对。

- python/sglang/srt/model_executor/cpu_graph_runner.py（模块 调度器；类别 source；类型 data-contract；符号 prepare_replay）：CPU Graph runner，prepare_replay 中遗漏了 req_pool_indices 清零，本 PR 补上。
- python/sglang/srt/speculative/eagle_draft_extend_cuda_graph_runner.py（模块 推测解码；类别 source；类型 core-logic；符号 replay）：speculative decoding EAGLE draft extend CUDA Graph runner，补齐 padding 分支中 req_pool_indices 清零。
- python/sglang/srt/speculative/frozen_kv_mtp_cuda_graph_runner.py（模块 推测解码；类别 source；类型 core-logic；符号 replay）：Frozen KV MTP CUDA Graph runner，补齐 padding 分支中 req_pool_indices 清零。

关键符号：populate_from_forward_batch, prepare_replay, replay

关键源码片段

python/sglang/srt/model_executor/cuda_graph_runner.py

核心 CUDA Graph runner，主 populate 路径。本次修复在此新增 self.req_pool_indices.zero_()，与已有的 seq_lens fill 和 out_cache_loc zero 配对。

```
def populate_from_forward_batch(self, *, forward_batch, raw_bs, raw_num_token, bs, seq_len_
fill_value, ...):
    if bs != raw_bs:
        self.seq_lens.fill_(seq_len_fill_value)
        self.out_cache_loc.zero_()
        # Pair with seq_lens fill: padded rows must point at reserved
        # req_pool slot 0 (req_to_token[0, :] is all zeros from init),
        # so dummy attention reads land on slot 0 instead of a stale
        # req_to_token row left by an earlier replay.
        self.req_pool_indices.zero_() # <-- 新增清零
        if self.mamba_track_indices is not None:
            self.mamba_track_indices.zero_()
        if self.mamba_track_mask is not None:
            self.mamba_track_mask.fill_(False)
```

python/sglang/srt/model_executor/cpu_graph_runner.py

CPU Graph runner，prepare_replay 中遗漏了 req_pool_indices 清零，本 PR 补上。

```
def prepare_replay(self, forward_batch):
    ...
    captured_forward_batch.seq_lens.fill_(self.seq_len_fill_value)
    captured_forward_batch.out_cache_loc.zero_()
    # Pair with seq_lens fill: padded rows must point at reserved
    # req_pool slot 0 (req_to_token[0, :] is all zeros from init).
    captured_forward_batch.req_pool_indices.zero_() # <-- 新增清零
    captured_forward_batch.input_ids[:raw_num_token].copy_(forward_batch.input_ids)
    captured_forward_batch.req_pool_indices[:raw_bs].copy_(forward_batch.req_pool_indices)
    ...
```

python/sglang/srt/speculative/eagle_draft_extend_cuda_graph_runner.py

speculative decoding EAGLE draft extend CUDA Graph runner，补齐 padding 分支中 req_pool_indices 清零。

```
def replay(self, forward_batch):
    ...
    if bs * self.num_tokens_per_bs != num_tokens:
        buffers.seq_lens.fill_(self.seq_len_fill_value)
        buffers.out_cache_loc.zero_()
        buffers.positions.zero_()
        # Pair with seq_lens fill: padded rows must point at reserved
        # req_pool slot 0 (req_to_token[0, :] is all zeros from init).
        buffers.req_pool_indices.zero_() # <-- 新增清零
        buffers.num_correct_drafts.fill_(self.num_tokens_per_bs)
        ...
```

评论区精华

Review 中无实质性讨论，仅由 gemini-code-assist bot 自动评论，确认变更正确且无反馈。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。变更仅增加了一行清零操作，且仅在 padding 路径上执行，不影响正常路径。清零操作已存在于同类 runner（Eagle draft）中，本次是补齐遗漏。但注意：如果 req_pool_indices 在后续逻辑中有特殊值约定，强制清零可能改变行为，但目前代码预设 slot 0 为保留的全零行，因此安全。建议在测试中覆盖 padding 场景，验证 padding 行是否确实指向 slot 0。
- 影响：影响范围包括 CUDA Graph 和 CPU Graph 的 populate/replay 流程，以及两个 speculative decoding runner（Eagle draft extend 和 Frozen KV MTP）。对用户而言，修复了因残留在 padding 行上的 stale req_pool 指针导致的潜在错误结果，提升了推理的正确性。对系统而言，性能影响可忽略（一次额外的 GPU memset 操作）。
- 风险标记：缺少测试覆盖

关联脉络

- 暂无明显关联 PR