PR #23037 完整报告

sgl-project/sglang

[Bug Fix] Resolve EAGLE cuda graph IMA under PD + DP + MTP with GLM-5.1

合并时间：2026-05-02 04:53

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23037

执行摘要

- 一句话：修复 EAGLE cuda graph 因 padding 残留和视图别名导致的 OOB
- 推荐动作：建议快速合并，该修复解决了实际运行中频繁崩溃的已知问题。值得关注的设计决策： 1) 清零填充区域 vs 让下游 gather 感知 padding 的权衡； 2) 使用 maybe_detect_oob 代替 clamp/assert 以避免 GPU-CPU 同步。

功能与动机

修复 PD + DP + MTP 长运行时 EAGLE 解码崩溃，详见 PR body 和关联 issue #15457 及 #22096。根因为 cuda graph 输出缓冲区视图别名和填充区域陈旧数据。

实现拆解

1. 在 eagle_draft_cuda_graph_runner.py 导入 maybe_detect_nan 和 maybe_detect_oob。
2. 在 replay 方法的 padding 分支（bs != raw_bs）中，新增对 buffers.topk_p、topk_index、hidden_states、req_pool_indices 的 .zero_() 清零操作。
3. 在复制 topk_p 和 topk_index 之前，分别调用 maybe_detect_nan 和 maybe_detect_oob 进行范围检测，替代原有 clamp 逻辑，避免 GPU-CPU 同步。 未涉及测试文件变更，eagle_worker.py 中的 clone 修改在后续讨论中被作者撤销。

关键文件：

- python/sglang/srt/speculative/eagle_draft_cuda_graph_runner.py（模块 推测解码；类别 source；类型 core-logic；符号 replay, EAGLEDraftCudaGraphRunner）：核心修复文件；修改 replay 方法中 padding 分支的清零逻辑和输入范围检测。

关键符号：EAGLEDraftCudaGraphRunner.replay

关键源码片段

python/sglang/srt/speculative/eagle_draft_cuda_graph_runner.py

核心修复文件；修改 replay 方法中 padding 分支的清零逻辑和输入范围检测。

```
# 文件：python/sglang/srt/speculative/eagle_draft_cuda_graph_runner.py
# 核心修改：replay 方法中 padding 分支清零 + 范围检测

from sglang.srt.speculative.spec_utils import maybe_detect_nan, maybe_detect_oob
```

```
class EAGLEDraftCudaGraphRunner:
    def replay(self, forward_batch: ForwardBatch):
        # ... 其他逻辑 ...
        bs = self.capture_bs[index]
        if bs != raw_bs:
            # 清零所有可能被后续 kernel 读取的 padding 区域，
            # 防止陈旧值导致 OOB 或 NaN 污染
            buffers.seq_lens.fill_(self.seq_len_fill_value)
            buffers.out_cache_loc.zero_()
            buffers.positions.zero_()
            buffers.topk_p.zero_()
            buffers.topk_index.zero_()
            buffers.hidden_states.zero_()
            buffers.req_pool_indices.zero_()

        # 复制有效数据前，检测输入是否合法
        maybe_detect_nan(
            forward_batch.spec_info.topk_p,
            "EagleDraftCudaGraphRunner.replay: topk_p",
        )
        maybe_detect_oob(
            forward_batch.spec_info.topk_index,
            0,
            self.model_runner.model_config.vocab_size,
            "EagleDraftCudaGraphRunner.replay: topk_index vs vocab_size="
            f"{self.model_runner.model_config.vocab_size}",
        )
        buffers.topk_p[:raw_bs].copy_(forward_batch.spec_info.topk_p)
        buffers.topk_index[:raw_bs].copy_(forward_batch.spec_info.topk_index)
        buffers.hidden_states[:raw_bs].copy_(forward_batch.spec_info.hidden_states)
        buffers.req_pool_indices[:raw_bs].copy_(forward_batch.req_pool_indices)
        # ... 后续 replay 逻辑 ...
```

评论区精华

核心讨论在于是否使用 clamp 还是 assert/oob 检测：JustinTong0323 建议放弃 clamp 改用 assert 避免掩盖真正问题；kpham-sgl 建议使用 maybe_detect_oob 避免 GPU-CPU 同步；Qiaolin-Yu 询问为何清零 hidden_states，作者解释 NaN 污染可能导致后续计算异常。最终采用 maybe_detect_oob 并维持清零。

- clamp vs assert 的取舍 (design): 作者接受建议，但最终采用了 maybe_detect_oob 以避免 GPU-CPU 同步
- hidden_states 清零的必要性 (correctness): 作者坚持认为清零更安全，且开销在 padding 分支可接受
- eagle_worker.py 中的 clone 修改 (other): 作者撤销了 eagle_worker.py 的 clone 修改，只保留 dart graph runner 的修复

风险与影响

- 风险：性能风险：新增 zero_ 仅在 padding 分支执行，开销可控；但若 padding 频繁触发（如批量大小波动大），hidden_states 清零可能成为瓶颈。正确性风险：maybe_detect_nan/oob 在发现异常时会直接抛出，可能使原可容忍的轻微异常变为硬失败，但作者认为应及早暴露问题。
- 影响：影响范围：仅 EAGLE 推测解码路径，特别影响 GLM-5.1 模型在 PD+DP+MTP 配置下的稳定性。不影响其它模型或非推测解码模式。缓解了偶发但严重的 cuda graph IMA 崩溃。
- 风险标记：核心路径变更，仅 padding 分支清零

关联脉络

- PR #15457 疑似提及该 bug: PR body 引用 issue #15457 提到同一 crash
- PR #22096 也可能被修复的罕见并发 bug: reviewer kpham-sgl 指出这个修复可能也解决了 #22096（极罕见发生）