PR #24401 完整报告

sgl-project/sglang

[Fix] Reset positions tensor in CUDA graph runner when batch size differs from captured size

合并时间：2026-06-10 06:03

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24401

执行摘要

- 一句话：修复 CUDA Graph 运行器中缺少 positions 重置导致的非法内存访问
- 推荐动作：该 PR 值得精读，尤其是了解 CUDA graph replay 中填充策略如何导致非法内存访问。关键设计决策是：对于被 flashinfer 等后端读取的尾部填充区域，必须使用 ZERO 填充策略而非 FOREACH_COPY，因为 FOREACH_COPY 不会重置尾部。建议阅读关联 Issue #24361 和之前类似的修复 PR #10892 以系统理解此类问题的模式。

功能与动机

关联 Issue #24361 报告了 NGRAM speculative decoding + flashinfer attention + CUDA graph replay 在 Blackwell sm_120 上运行 2-5 小时后间歇性崩溃，错误为 cudaErrorIllegalAddress 内部调用 flashinfer.prefill.plan。PR body 说明根因在于 cuda_graph_runner.py 的 populate_from_forward_batch 函数中，当 bs != raw_bs 时重置了 seq_lens、out_cache_loc、mamba_track_* 等张量，但遗漏了 positions，导致填充 token 的 positions 保留前一次 replay 的垃圾值，进而 flashinfer 计算了无效的 qo_indptr 并访问非法内存。

实现拆解

1. 修改构建注册表时的填充策略：在 python/sglang/srt/model_executor/cuda_graph_buffer_registry.py 的 build_decode_registry 函数中，将 positions 和 mrope_positions 两个 GraphSlot 的 padding_policy 从默认行为（不指定时相当于 FOREACH_COPY）修改为 PaddingPolicy.ZERO。
2. 更新文档注释：同步更新了函数文档字符串，明确说明 positions 和 mrope_positions 现在使用 ZERO 填充策略，因为 flashinfer verify-path plan 会读取填充尾部，保留过期值会触发非法内存访问（关联 Issue #24361）。同时将 input_ids 的说明改为 FOREACH_COPY，因为其头部被覆盖且尾部不被读取。
3. 涉及文件：仅修改了 python/sglang/srt/model_executor/cuda_graph_buffer_registry.py 一个文件，改动量 +15/-7 行。没有添加测试、配置或部署配套。

关键文件：

- python/sglang/srt/model_executor/cuda_graph_buffer_registry.py（模块 模型执行器；类别 source；类型 data-contract；符号 build_decode_registry）：核心修复文件，修改了 build_decode_registry 函数中 positions 和 mrope_positions 的填充策略，从无（默认

为 FOREACH_COPY）改为 ZERO。

关键符号：build_decode_registry

关键源码片段

python/sglang/srt/model_executor/cuda_graph_buffer_registry.py

核心修复文件，修改了 build_decode_registry 函数中 positions 和 mrope_positions 的填充策略，从无（默认为 FOREACH_COPY）改为 ZERO。

```
# 文件：python/sglang/srt/model_executor/cuda_graph_buffer_registry.py
# 函数：build_decode_registry
# 关键变更：positions 和 mrope_positions 增加 padding_policy=PaddingPolicy.ZERO

slots = [
    GraphSlot("input_ids", _tokens, torch.int64, axis="tokens"),
    # 修复前：无 padding_policy，默认为 FOREACH_COPY（不重置尾部）
    # 修复后：使用 ZERO 填充策略，每次 replay 时尾部置零
    GraphSlot(
        "positions",
        _tokens,
        torch.int64,
        axis="tokens",
        padding_policy=PaddingPolicy.ZERO, # <-- 新增
    ),
    GraphSlot(
        "out_cache_loc",
        _tokens,
        cache_loc_dtype,
        axis="tokens",
        padding_policy=PaddingPolicy.ZERO,
    ),
    GraphSlot(
        "req_pool_indices",
        _bs,
        torch.int64,
        axis="bs",
        padding_policy=PaddingPolicy.ZERO,
    ),
    GraphSlot(
        "seq_lens",
        _bs,
        torch.int32,
        axis="bs",
        padding_policy=PaddingPolicy.FILL_SENTINEL,
        pad_value=seq_len_fill_value,
    ),
    GraphSlot(
        "seq_lens_cpu",
```

```
        _bs,
        torch.int32,
        axis="bs",
        device=torch.device("cpu"),
        padding_policy=PaddingPolicy.FILL_SENTINEL,
        pad_value=seq_len_fill_value,
    ),
    GraphSlot(
        "mrope_positions",
        lambda _bs2, mt: (3, mt),
        torch.int64,
        axis="tokens",
        slice_fn=lambda buf, n: buf[:, :n],
        padding_policy=PaddingPolicy.ZERO, # <-- 新增
    ),
]
```

评论区精华

该 PR 的 review 评论较少且无争议。审核者 kpham-sgl 直接批准了 PR。提交者 weizhoublue 在 PR body 中详细描述了根因分析。整个讨论聚焦于技术正确性，没有设计权衡或未解决的疑虑。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 回归风险较低：变更仅涉及为一个已存在的张量添加填充策略（ZERO），该策略与其他张量（如 out_cache_loc、req_pool_indices）使用的策略一致，且该张量在每次 replay 时会被正确覆盖头部。没有改变数据流或计算逻辑。
  2. 性能影响可忽略：ZERO 填充仅对尾部少量元素赋值，与现有的 tail-reset 路径开销相同。
  3. 缺少单元测试：PR 没有添加单元测试来验证修复后的行为。虽然 Issue 报告的场景难以在短时间内复现，但可以添加模拟不同 batch size 下的 padding 重置测试。 - 影响：影响范围：直接影响使用 NGRAM speculative decoding + flashinfer attention + CUDA graph replay 的部署，尤其是 Blackwell（sm_120）等需要 CUDA graph 的场景。该修复消除了一个间歇性崩溃，提升了系统稳定性。影响程度：中等。修复了一个影响推理稳定性的误报问题，对于受到影响的用户是关键的。
- 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- PR #10892 [Fix] Reset positions tensor in EAGLE draft runner when batch size differs from captured size: 之前修复了 EAGLE draft runner 中相同的 positions 重置遗漏问题，本 PR 将其扩展到主 runner。