PR #22598 完整报告

sgl-project/sglang

[AMD] fix tbo runtime error when initializing metadata for cuda graph

合并时间：2026-04-20 03:42

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/22598

执行摘要

- 一句话：修复 AMD 平台上 aiter 后端 CUDA 图初始化时的运行时错误，通过重分配缓冲区适配填充批次大小。
- 推荐动作：建议精读 aiter_backend.py 中的 init_cuda_graph_state 方法变更，关注缓冲区动态重分配的设计决策和 dtype 统一策略，这反映了 CUDA 图优化中的内存管理权衡。

功能与动机

PR #20978 为了提升 CUDA 图覆盖性能，将 capture_bs 更新至超出 pool_size，但这导致 aiter 后端在初始化元数据时出现运行时错误，错误信息显示张量大小不匹配。此 PR 旨在修复此回归问题。

实现拆解

1. 核心逻辑修改：在 python/sglang/srt/layers/attention/aiter_backend.py 的 init_cuda_graph_state 方法中，添加条件检查，如果 max_bs + 1 大于现有缓冲区大小，则重分配 kv_indptr、qo_indptr、mask_indptr 和可选的 qo_indptr_ 缓冲区，并统一 cuda_graph_kv_last_page_len 的 dtype 为 torch.int32。
2. 测试配套调整：在 test/registered/amd/test_moriep_small.py 中新增 wait_all_ports_release 函数，通过检查所有派生端口是否可用，确保测试环境清理，并在多个测试类的 tearDownClass 中调用该函数。

关键文件：

- python/sglang/srt/layers/attention/aiter_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 init_cuda_graph_state）：这是修复运行时错误的核心文件，修改了 CUDA 图初始化逻辑，直接影响注意力后端的正确性。
- test/registered/amd/test_moriep_small.py（模块 AMD 测试；类别 test；类型 test-coverage；符号 wait_all_ports_release）：测试文件添加了端口释放等待逻辑，确保测试环境清理，支持修复的验证和稳定性。

关键符号：init_cuda_graph_state, wait_all_ports_release

关键源码片段

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

这是修复运行时错误的核心文件，修改了CUDA图初始化逻辑，直接影响注意力后端的正确性。

```
def init_cuda_graph_state(
    self,
    max_bs: int,
    max_num_tokens: int,
    kv_indices_buf: Optional[torch.Tensor] = None,
):
    # PR #20978 pads max_bs beyond pool_size for higher cuda-graph
    # coverage. Reallocate indptr buffers so they fit the padded max_bs.
    # See: https://github.com/sgl-project/sglang/pull/20978
    if max_bs + 1 > self.kv_indptr.shape[0]:
        self.kv_indptr = torch.zeros(
            (max_bs + 1,), dtype=torch.int32, device=self.device
        )
        self.qo_indptr = torch.zeros(
            (max_bs + 1,), dtype=torch.int32, device=self.device
        )
        self.mask_indptr = torch.zeros(
            (max_bs + 1,), dtype=torch.int64, device=self.device
        )
        if hasattr(self, "qo_indptr_"):
            self.qo_indptr_ = torch.zeros(
                (max_bs + 1,), dtype=torch.int32, device=self.device
            )

    self.cuda_graph_kv_last_page_len = torch.ones(
        max_bs, dtype=torch.int32, device=self.device # 统一 dtype 为 torch.int32，确保一致性
    )
    # 后续代码处理 kv_indices_buf 和自定义掩码，此处省略以聚焦核心变更
```

test/registered/amd/test_moriep_small.py

测试文件添加了端口释放等待逻辑，确保测试环境清理，支持修复的验证和稳定性。

```
def wait_all_ports_release(base_url, timeout_s=60):
    """等待直到所有派生端口完全释放。"""
    import time

    port = int(base_url.split(":")[-1])

    # 参考 server_args.py 中的端口偏移定义
    offsets = [
        0, # 基础端口无偏移
        ZMQ_TCP_PORT_DELTA, # dist_init_port
        ZMQ_TCP_PORT_DELTA + 1, # detokenizer_port
        ZMQ_TCP_PORT_DELTA + 2, # rpc_port
        ZMQ_TCP_PORT_DELTA + 3, # metrics_port
        ZMQ_TCP_PORT_DELTA + 4, # scheduler_input_port
    ]
    for _ in range(timeout_s):
```

```
            if all(is_port_available(port + off) for off in offsets):
                return # 所有端口可用则退出
            time.sleep(1)
        # 尽力而为：超时后仅打印警告，避免影响后续测试类
        print(f"Warning: some ports still occupied after {timeout_s}s")
```

评论区精华

review 中，gemini-code-assist[bot] 建议同时重分配 kv_last_page_len 缓冲区以确保足够大小，但作者 billishyahao 回应指出 cuda_graph_kv_last_page_len 是首次初始化，无需重分配，并确认 torch.int 等价于 torch.int32，但为统一代码风格同意修复 dtype。决策结论是只统一 dtype，未采纳 kv_last_page_len 的重分配建议。

- 缓冲区重分配和 dtype 统一 (correctness): 作者 billishyahao 回应 cuda_graph_kv_last_page_len 是首次初始化，无需重分配，但统一 dtype 为 torch.int32 以改善代码风格；kv_last_page_len 未被重分配。

风险与影响

- 风险：风险较低，因为变更主要涉及缓冲区大小检查和重分配，但需确保所有相关缓冲区（如 indptr）都被正确处理，避免遗漏导致未来类似错误。测试中添加的端口释放逻辑可能因端口占用导致测试超时，但采用了超时后仅打印警告的设计，降低了影响。
- 影响：影响范围限于使用 aiter 后端和 CUDA 图的 AMD 平台场景，修复了运行时错误，提升了系统稳定性和正确性。对用户透明，但确保了推理任务在填充批次大小下不会崩溃。测试改动改善了测试环境的可靠性，减少了因端口未释放导致的偶发失败。
- 风险标记：缓冲区大小管理，dtype 一致性

关联脉络

- PR #20978 未知（从 PR body 引用）：此 PR 引入了错误，将 capture_bs 更新至超出 pool_size，导致缓冲区大小不匹配，当前 PR 修复了该问题。