PR #27948 完整报告

sgl-project/sglang

Skip custom all-reduce v2 CUDA graph capture with torch memory saver.

合并时间：2026-07-01 05:53

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27948

执行摘要

- 一句话：修复 custom all-reduce v2 在 torch memory saver 下 CUDA graph 捕获失败
- 推荐动作：小而关键的 bugfix，值得精读其讨论以理解 CUDA graph 捕获和 TMS 交互的细节。建议 reviewer 关注 C++ 侧 m_is_graph_capturing 语义的理解。

功能与动机

当 --colocate 启用 torch_memory_saver 时，rollout engine 的 CUDA graph 捕获因 custom all-reduce v2 试图注册已被 TMS 劫持的 IPC handles 而失败，抛出 Runtime check failed at custom_all_reduce.cuh:37: CUDA error: invalid argument。本 PR 旨在使 v2 在 TMS 下正确降级到内部 buffer 路径。

实现拆解

1. 新增环境变量检查：在 CustomAllReduceV2.__init__ 中，新增 self.tms_cudagraph = envs.SGLANG_MEMORY_SAVER_CUDA_GRAPH.get()，用于在初始化时判断是否启用 TMS CUDA graph 模式。
2. 修改 capture 方法：将 self.obj.set_cuda_graph_capture(True) 改为 self.obj.set_cuda_graph_capture(not self.tms_cudagraph)，使得在 TMS 启用时 CUDA graph 捕获仍然进行，但 graph zero-copy 优化被禁用（C++ 侧 m_is_graph_capturing 为 false），因此不会尝试注册 IPC handles。
3. 修改 custom_all_reduce 方法：将同样将捕获结束时恢复 set_cuda_graph_capture(True) 改为 set_cuda_graph_capture(not self.tms_cudagraph)，以与 capture 方法的行为一致。

关键文件：

- python/sglang/srt/distributed/device_communicators/custom_all_reduce_v2.py（模块 分布式通信；类别 source；类型 core-logic；符号 CustomAllReduceV2.init, CustomAllReduceV2.capture, CustomAllReduceV2.custom_all_reduce）：核心变更文件，修改了 CUDA graph 捕获时是否启用 graph zero-copy 的逻辑。

关键符号：CustomAllReduceV2.init, CustomAllReduceV2.capture, CustomAllReduceV2.custom_all_reduce

关键源码片段

python/sglang/srt/distributed/device_communicators/custom_all_reduce_v2.py

核心变更文件，修改了 CUDA graph 捕获时是否启用 graph zero-copy 的逻辑。

```
# python/sglang/srt/distributed/device_communicators/custom_all_reduce_v2.py

from sglang.srt.environ import envs # 新增导入

class CustomAllReduceV2:
    def __init__(self, ...):
        # ... 初始化其他成员 ...
        self.tms_cudagraph = envs.SGLANG_MEMORY_SAVER_CUDA_GRAPH.get() # 检查环境变量
        # ... 后续初始化 ...

    @contextmanager
    def capture(self):
        if self.disabled:
            yield
            return
        try:
            # 当 tms_cudagraph 为 True 时，不启用 graph zero-copy（C++ 侧 m_is_graph_
            capturing=False）
            self.obj.set_cuda_graph_capture(not self.tms_cudagraph)
            yield
        finally:
            self.obj.set_cuda_graph_capture(False)
        # ... 后续注册逻辑 ...

    def custom_all_reduce(self, input: torch.Tensor) -> torch.Tensor:
        # ...
        try:
            self.obj.set_cuda_graph_capture(False)
            return self._all_reduce(input)
        finally:
            # 恢复时也保持与 capture 一致
            self.obj.set_cuda_graph_capture(not self.tms_cudagraph)
        return self._all_reduce(input)
```

评论区精华

Reviewer DarkSharpness 最初建议直接用 self.obj.set_cuda_graph_register_inputs(not self.tms_cudagraph) 来禁用 graph input 注册，同时保持 set_cuda_graph_capture(True)。但作者 zyzshishui 指出，CustomAllReduceV1 在 TMS 下仅切换 all-reduce 到未注册路径，而 v2 的 set_cuda_graph_capture 也控制其他捕获时行为，禁用过多。DarkSharpness 随后澄清，C++ 侧 m_is_graph_capturing 完全等价于是否启用 graph zero-copy 优化（包括捕获和分配分发），因此复用旧接口 set_cuda_graph_capture(not self.tms_cudagraph) 即可，无需引入新接口。作者接受并更新了代码。最终 DarkSharpness 批准 PR。

- 是否复用 set_cuda_graph_capture 接口 (design): 采纳复用 set_cuda_graph_capture(not self.tms_cudagraph) 的方案，移除新接口。

风险与影响

- 风险：低风险。改动仅涉及一个文件，共 6 行，且逻辑与 CustomAllReduceV1 已在 #19162 中的处理方式一致。在 TMS 下禁用了 graph zero-copy 优化，可能对性能略有影响，但这是在 TMS 场景下的正确行为。
- 影响：直接影响使用 --colocate 和 torch_memory_saver 的 rollout engine 场景。修复了 CUDA graph 捕获崩溃，使得 8-GPU Miles Megatron colocate 端到端测试通过。不启用 TMS 时无行为改变。
- 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- PR #19162 Skip custom all-reduce v1 CUDA graph capture with torch memory saver.: 本 PR 是 #19162 的 follow-up，将相同的修复逻辑应用到 custom all-reduce v2。