PR #26550 完整报告

sgl-project/sglang

[CI] FA3: ascending cuda-graph capture to avoid varlen workspace IMA (#26532)

合并时间：2026-05-28 15:50

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26550

执行摘要

- 一句话：修复 CI 中 FA3 的 CUDA Graph 捕获顺序导致的 IMA
- 推荐动作：建议精读。本 PR 展示了如何通过环境变量 + 后端检测条件性地调整 CUDA Graph 捕获策略，修复一个难以排查的内存越界问题。设计上避免了 test 依赖混入生产代码，值得参考。

功能与动机

修复 issue #26532 描述的 FA3 varlen workspace slot IMA（非法内存访问）。在 CI 中，降序捕获会导致 FA3 的 workspace slot 分配顺序与预期相反，从而触发内存越界。

实现拆解

1. 新增 gate 函数 _ci_use_ascending_capture_order（python/sglang/srt/model_executor/cuda_graph_runner.py） - 函数签名为 (server_args) -> bool，检查 envs.SGLANG_IS_IN_CI 环境变量是否为 True，再通过 server_args.get_attention_backends() 获取 prefill、decode 后端，同时检查 speculative_draft_attention_backend，任一为 "fa3" 则返回 True。 - 设计为模块级自由函数，避免污染 CudaGraphRunner 类，也避免了从 sglang.test.test_utils 导入的架构违规。
2. 修改 _capture_one_stream 内部逻辑（python/sglang/srt/model_executor/cuda_graph_runner.py） - 原代码固定使用 reversed(self.capture_bs) 实现降序捕获。 - 新增一个局部变量 bs_seq：若 _ci_use_ascending_capture_order 返回 True，则使用 list(self.capture_bs)（升序）；否则使用 list(reversed(self.capture_bs))（降序）。 - tqdm 和迭代器均基于 bs_seq 构建，tqdm 的 description 更新逻辑保持不变。
3. 测试验证 - 通过 /rerun-test 命令触发了 test_eagle_infer_b.py 测试（issue 中记录的原先 flaky 用例），运行成功。 - PR body 中详细记录了不同测试类的捕获顺序验证结果，确保 gate 只影响 FA3 + CI 场景，不干扰其他配置。

关键文件：

- python/sglang/srt/model_executor/cuda_graph_runner.py（模块 执行引擎；类别 source；类型 data-contract；符号 _ci_use_ascending_capture_order）：核心变更文件：新增 gate 函数 _ci_use_ascending_capture_order 并修改 _capture_one_stream 中的捕获顺序逻辑。

关键符号：_ci_use_ascending_capture_order

关键源码片段

python/sglang/srt/model_executor/cuda_graph_runner.py

核心变更文件：新增 gate 函数 _ci_use_ascending_capture_order 并修改 _capture_one_stream 中的捕获顺序逻辑。

```
def _ci_use_ascending_capture_order(server_args) -> bool:
    """Whether CI + FA3 forces ascending cuda-graph capture (FA3 varlen IMA workaround,
    #26532)."""
    if not envs.SGLANG_IS_IN_CI.get():
        return False
    # get_attention_backends 返回 (prefill_backend, decode_backend)
    prefill_backend, decode_backend = server_args.get_attention_backends()
    # 同时检查 speculative draft 的后端
    return "fa3" in (
        prefill_backend,
        decode_backend,
        server_args.speculative_draft_attention_backend,
    )

# 在 _capture_one_stream 中的使用：
def _capture_one_stream(stream_idx: Optional[int] = None):
    avail_mem = get_available_gpu_memory(...)
    # 根据条件选择升序或降序捕获顺序
    bs_seq = (
        list(self.capture_bs)
        if _ci_use_ascending_capture_order(self.model_runner.server_args)
        else list(reversed(self.capture_bs))
    )
    capture_range = (
        tqdm.tqdm(bs_seq) if get_tensor_model_parallel_rank() == 0 else iter(bs_seq)
    )
    for i, bs in enumerate(capture_range):
        # ... 原有捕获逻辑保持不变
```

评论区精华

reviewer（gemini-code-assist[bot]）指出初始实现从 sglang.test.test_utils 导入辅助函数是架构违规：生产环境中 test 包可能不存在，会导致 ImportError。建议将检查逻辑内联到 cuda_graph_runner.py。作者采纳建议，在第二版提交中将函数内联为模块级函数 _ci_use_ascending_capture_order，移除对 sglang.test 的依赖。

- 从 test_utils 导入的架构违规 (design): 作者采纳建议，在第二版提交中删除了 test_utils 中的辅助函数，直接在 cuda_graph_runner.py 中实现 _ci_use_ascending_capture_order。

风险与影响

- 风险：
  - 仅影响 CI + FA3 场景，生产环境捕获顺序不变，回归风险低。
  - 新增的 *gate* 依赖 *envs.SGLANG_IS_IN_CI* 和 *server_args.get_attention_backends*，这些接口稳定，但若未来私有 CI 未设置 SGLANG_IS_IN_CI 可能导致 workaround 不生效。
  - 内存成本评估显示 max_bs=5 时额外开销约 0.21 GB，小于 CUDA Graph footprint 的 1%，可接受。
- 影响：
  - 用户：无直接影响（仅 CI）。
  - 系统：修复了 CI 中 FA3 的随机 IMA 崩溃，使 *test_eagle_infer_b* 等测试稳定通过。
  - 团队：降低了 CI flakiness，提升开发效率。未来若确认无条件升序捕获无副作用，可移除此门控。
  - 风险标记：仅 CI 生效，env 依赖，测试覆盖有限，生产环境无影响

关联脉络

- PR #4195 [Memory] *better memory sharing across cuda graphs by reverse capture order*: 原降序捕获优化 PR，本 PR 在此基础上按条件保留降序或升序。