PR #34184 完整报告

sgl-project/sglang

Fix stale track rows corrupting conv checkpoints under the prefill graph

合并时间：2026-08-10 08:57

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34184

执行摘要

- 一句话：修复预填图重放残留 track 行导致 conv checkpoint 错乱
- 推荐动作：值得精读。改动虽小，但包含三类高价值内容：一是 captured graph 读全行、静态缓冲区需哨兵清理的通用模式；二是死代码成因分析（按 token 填充 vs 按 request 填充的语义差）；三是作者展示的系统化插桩定位方法（对比 slot 内容、A/B 验证）以及局部 vs 根治修复的取舍思路，可迁移到其他图捕获后端。

功能与动机

PR body 明确描述了触发链：prefix-cache 命中时，hybrid-SWA 模型可能从它从未产生的 conv state 上解码（"A prefix-cache hit on a hybrid-SWA model can decode off a conv state it never produced"）。重放的 prefill graph 读取 _capture_req_slots 行，但两个 track 缓冲区在 live batch 之后保留上次重放的值，残留的 mask 行仍携带有效目标 slot，导致 track scatter 把当前窗口落到先前请求的 checkpoint。该问题需要 prefill CUDA graph + chunked prefill + 并发 + prefix hit 同时出现，关闭 prefill graph 即可规避；它与 #34043 是相邻但不同的缺陷路径，#34043 在场时仍能复现。

实现拆解

1. 根因定位：PrefillInputBuffers.fill_from 中清理 mamba_track_* 的逻辑受 bs != raw_bs 条件保护，而 prefill graph 是按 token 填充（padded_bs == raw_bs），该分支从不执行，成为死代码；残留数据跨重放被 captured track scatter 消费。
2. 核心修改：在 python/sglang/srt/model_executor/runner/prefill_cuda_graph_runner.py 的静态缓冲区刷新段（_is_full_backend 且 bs < self._capture_req_slots 的 sentinel tail 分支）中，在既有 6 个字段清零之后，新增对 mamba_track_mask、mamba_track_indices 的 [bs:r] 区间清零；通过 self.buffer_registry.has_slot 做存在性保护，兼容没有 track 槽位的模型，与哨兵尾部既有语义保持一致。
3. 测试配套：test/registered/models_e2e/test_inkling_small_nvfp4.py 删除 --disable-prefill-cuda-graph 启动参数，使确定性 logprob 一致性用例（尤其 prefill cache hit 场景）重新运行在 prefill graph 路径上，成为本 bug 的回归守卫；此前该标志相当于在绕开问题路径。
4. 方案取舍：作者在 PR body 中对比了根治方案——让 padded_bs 报告捕获的 slot 数，从而激活 fill_from 的清理分支并消除死代码，但该改动会影响所有共享这些缓冲区的阶段的行为，故本 PR 选择局部修复，并明确表示可以切换（"Happy to switch"）。

关键文件：

- python/sglang/srt/model_executor/runner/prefill_cuda_graph_runner.py（模块 预填图；类别 source；类型 data-contract）：核心修复所在：captured graph 读取全部 req_slots 行，残留的 track 行会把当前窗口写入先前请求的 checkpoint；此处按 registry 槽位清零是唯一行为变更点，与既有 6 字段哨兵清理语义对齐。
- test/registered/models_e2e/test_inkling_small_nvfp4.py（模块 端到端；类别 test；类型 test-coverage）：该用例是此 bug 的回归守卫，此前用 --disable-prefill-cuda-graph 绕开问题路径，本 PR 移除该标志，使确定性 KL 测试真正覆盖 prefill graph。

关键符号：未识别

关键源码片段

python/sglang/srt/model_executor/runner/prefill_cuda_graph_runner.py

核心修复所在：captured graph 读取全部 req_slots 行，残留的 track 行会把当前窗口写入先前请求的 checkpoint；此处按 registry 槽位清零是唯一行为变更点，与既有 6 字段哨兵清理语义对齐。

```
# prefill_cuda_graph_runner.py 中重放路径的静态缓冲区刷新逻辑（节选）
if self._is_full_backend and bs < self._capture_req_slots:
    # Sentinel tail：captured graph 会读取全部 req_slots 行（例如
    # logits processor 的 cumsum），所以上一次重放留下的旧值必须清零。
    # 清零后这些行的长度为零，sentinel 不会产生实际 token 效果；
    # extend_start_loc 的 sentinel 位于真实 token 的平坦末端。
    r = self._capture_req_slots
    s["seq_lens"][bs:r].zero_()
    s["extend_seq_lens"][bs:r].zero_()
    s["extend_prefix_lens"][bs:r].zero_()
    s["extend_start_loc"][bs:r].fill_(self.raw_num_tokens)
    s["req_pool_indices"][bs:r].zero_()
    s["orig_seq_lens"][bs:r].zero_()

    # mamba_track_mask / mamba_track_indices 由 buffer registry 管理，
    # 其清理本应在 PrefillInputBuffers.fill_from 的 bs != raw_bs 分支完成；
    # 但 prefill graph 以 padded_bs == raw_bs 调用（图按 token 而非
    # request 填充），该分支永不执行。残留的 mask 行仍携带有效的目标
    # slot，会导致本次重放的 track scatter 把当前窗口写入先前请求的
    # checkpoint。这里与上面 6 个字段一起就地清零。
    registry = self.buffer_registry
    for name in ("mamba_track_mask", "mamba_track_indices"):
        if registry.has_slot(name):
            registry.get_slot(name).buffer[bs:r].zero_()
```

评论区精华

本 PR 没有 review 评论（4 条 issue 评论均为 /rerun-test、/rerun-failed-ci 等 CI 命令式操作）。核心设计权衡沉淀在 PR body 中，作者对局部修复与根治方案做了明确取舍：让

padded_bs 报告捕获的 slot 数量是 " 更诚实的形态 "，但会改变每个共享这些缓冲区的阶段的行为，因此采取局部修复并愿意按 reviewer 意见切换。作者也坦诚说明了验证边界：CI 场景跑真实 NVFP4 权重、tp=4，batch 打包方式不同、受害请求不同，未在该配置上精确复现，若 CI 仍红，说明同一形状的 gap 存在于别处，可用同样的插桩方法定位。

- padded_bs 语义与局部修复 / 根治方案取舍 (design): 采取局部修复，作者表示愿意切换为根治方案；无 reviewer 提出异议。

风险与影响

- 风险：影响面受控：修改仅作用于 _is_full_backend 且 bs < _capture_req_slots 的 prefill graph 重放路径；满批（bs == r）时切片为空、无副作用，无 track 槽位的模型因 has_slot 保护不受影响，eager、投机、decode 路径行为不变。残余风险有三点：一是新增清零依赖人工在 sentinel tail 中维护，未来若再有按行捕获、由 registry 管理的缓冲区而未被纳入此处，会复发同类静默损坏；二是 fill_from 中 bs != raw_bs 分支仍是死代码，根因未消除；三是 CI 场景（tp=4 真实 NVFP4 权重）未精确复现 victim，验证主要依赖 tp=1 收缩 checkpoint 的插桩结果，存在 CI 仍红的可能。
- 影响：修复 Inkling / Inkling-Small 等 hybrid-SWA 模型在 prefill graph + chunked prefill + 并发 + prefix cache 命中组合下的静默生成错误，此前表现为长上下文生成 " 失忆 "、logprob 发散、确定性不一致。测试侧，确定性 KL 用例移除 --disable-prefill-cuda-graph 后，每次 PR CI 都会以 prefill graph 路径运行，回归覆盖面显著提升。对团队而言，确立了 sentinel tail 的覆盖范围应扩展至 registry 管理缓冲区的约定，也为后续把 padded_bs 语义改为 " 捕获 slot 数 " 的根治重构留下了入口。
- 风险标记：核心路径变更，静默数据损坏，死代码未根除，清理逻辑需人工同步

关联脉络

- PR #34043 [srt] Fix sconv state memory corruption on specdec: 投机路径上的 sconv 状态损坏修复，与本 PR 构成同一 "conv 状态来源错误 " 问题的相邻缺陷路径；作者明确验证本 PR 在 #34043 在场时仍能复现（checkpoint 写入了 stale 目的地）。
- PR #34168 Add deterministic logprob-consistency test for inkling-small nvfp4: 引入 Inkling-Small NVFP4 确定性 logprob 一致性测试的 PR，本 PR 修改的 test_inkling_small_nvfp4.py 正是该测试文件；移除 --disable-prefill-cuda-graph 后该测试回归 guard 真正生效。
- PR #34189 [DSV4] Fix silent KV corruption when speculative draft tokens > 4: 同属 " 静默状态损坏 " 修复脉络（DSV4 投机 draft 场景 KV 写坏），说明该时间段多个后端并发暴露同类 captured-buffer 残留问题。