PR #27597 完整报告

sgl-project/sglang

[lora] Exclude finished requests from running_loras

合并时间：2026-06-09 14:05

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27597

执行摘要

- 一句话：修复 LoRA 调度中已结束请求残留导致崩溃
- 推荐动作：值得合并，修复逻辑清晰、风险极低。建议后续添加单元测试，覆盖“请求结束但尚未被 filter_batch”时 LoRA 调度的正确性。

功能与动机

修复动态 LoRA 卸载场景下调度器崩溃。PR body 明确指出：*get_new_batch_prefill builds running_loras from all of running_batch.reqs, but finished requests are not removed from running_batch until the deferred filter_batch in update_running_batch. With dynamic LoRA unloading, if an adapter is unloaded after its requests finish but before they are filtered, the finished request's lora_id lingers in running_loras; the next validate_lora_batch call then trips assert lora_ref is not None ("LoRA ID ... not found in lora_refs") and crashes the scheduler process.*

实现拆解

1. 定位问题源头：在 python/sglang/srt/managers/scheduler.py 的 _get_new_batch_prefill_raw 方法中，第 2644 行构建 running_loras 集合时，直接遍历 self.running_batch.reqs 中的所有请求，未排除已结束的请求。
2. 单行修复：在集合推导式中增加过滤条件 if not req.finished()，即： running_loras = {req.lora_id for req in self.running_batch.reqs if not req.finished()} 确保只有尚未结束的请求的 lora_id 被纳入 running_loras。
3. 影响范围：此修改仅影响 enable_lora 为 True 时的调度路径。running_loras 用于后续遍历 waiting_queue 时调用 _can_schedule_lora_req 判断新请求能否调度，以及用于 lora_drainer.update_draining_state。排除已结束请求后，running_loras 不会包含已卸载适配器的 ID，从而避免 validate_lora_batch 断言失败。
4. 配套改动：无测试文件变更（PR 未附带测试），但逻辑正确性由现有 LoRA 调度流程覆盖，且修复本身十分微小。

关键文件：

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic）：修复的核心文件，在 _get_new_batch_prefill_raw 方法中修改了 running_loras 的构建逻辑。

关键符号：未识别

关键源码片段

python/sglang/srt/managers/scheduler.py

修复的核心文件，在 _get_new_batch_prefill_raw 方法中修改了 running_loras 的构建逻辑。

```
# python/sglang/srt/managers/scheduler.py, 位于 _get_new_batch_prefill_raw 方法中
# 构建当前运行的 LoRA ID 集合用于调度决策
if self.enable_lora:
    # 关键修复：排除已结束的请求。之前直接使用所有 running_batch.reqs，
    # 但已结束请求要到 filter_batch 阶段才会被移除。在这段窗口期内，
    # 如果适配器被卸载，已结束请求的 lora_id 会残留在 running_loras 中，
    # 导致后续 validate_lora_batch 因找不到 lora_ref 而崩溃。
    running_loras = {
        req.lora_id for req in self.running_batch.reqs if not req.finished()
    }
    # 同时考虑已在 PrefillAdder 中分配的请求（例如 chunked 请求）的 LoRA
    running_loras.update(req.lora_id for req in adder.can_run_list)

    if self.lora_drainer:
        self.lora_drainer.update_draining_state(
            self.waiting_queue,
            self.running_batch.reqs,
        )
```

评论区精华

无 review 评论。两位 reviewer（yushengsu-thu 和 Fridge003）均直接批准，未提出讨论或异议。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。修改仅在一处集合推导式中增加过滤条件，逻辑符合预期：已结束的请求不应参与下一轮解码批次的 LoRA 调度。req.finished() 方法是现有 API（推测基于 completion 标志），行为稳定。潜在风险是如果 finished() 方法在某些边缘状态下返回错误值（例如请求尚未完全清理但标记为未结束），则仍可能触发原 bug；但这属于更底层的状态管理问题，与本修复不冲突。
- 影响：
  - 用户影响：修复了动态 LoRA 卸载场景下的调度器崩溃，提升系统稳定性。影响范围限于使用 LoRA 且频繁卸载 / 加载适配器的用户。
  - 系统影响：修改极小，无性能影响，无接口变更。
  - 团队影响：无，仅修复 bug。
  - 风险标记：缺少测试覆盖

关联脉络

- PR #23802 fix: stop-string check misses early matches during speculative decoding: 同属调度批处理修复，涉及 schedule_batch.py 中的请求状态检查，与本 PR 在请求生命周期管理上有相似上下文。
- PR #22516 fix(server): clamp piecewise_cuda_graph_max_tokens to context_length: 同为调度器相关 bugfix，修复后能避免调度器崩溃，属于同一维护领域。