PR #23006 完整报告

sgl-project/sglang

[Pipeline Parallelism][Bug] Fix scheduler hang in pipeline parallelism setup

合并时间：2026-04-17 14:50

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23006

执行摘要

- 一句话：修复流水线并行模式下调度器因条件判断错误导致的预填充请求挂起问题。
- 推荐动作：该 PR 值得精读，尤其是 PR body 中详细的根因分析和计算示例，展示了如何定位和修复调度器中的条件竞争问题。关注点在于理解流水线并行下分块请求的调度策略，以及为何必须允许其在槽位不足时继续执行。

功能与动机

PR body 详细描述了在 PP=2 配置下，发送单个大提示词时服务器会挂起。根本原因是 _get_new_batch_prefill_raw 中一个条件判断被错误反转：当 self.chunked_req is not None（即存在分块请求）且无可用请求槽位时，原逻辑会错误地返回 None，导致调度器循环无法推进任何请求。作者追溯了问题引入于 PR #14883，并提供了具体的计算示例说明挂起场景。

实现拆解

1. 定位问题入口：修改文件 python/sglang/srt/managers/scheduler.py 中的 _get_new_batch_prefill_raw 方法。这是调度器核心逻辑，负责决定是否返回新的预填充批次。
2. 修正条件判断：将第 2462 行的条件从 self.chunked_req is not None 改为 self.chunked_req is None。这样，只有当不存在分块请求且无可用槽位时，才阻止新批次生成；而存在分块请求时，即使槽位不足，也允许其继续执行，避免内存泄漏和挂起。
3. 影响分析：此变更恢复了 PR #5724 引入流水线并行时的原始行为，确保分块请求能在微批次间正常流转。注释已明确说明在 PP 场景下，分块请求不应受每微批次最大运行请求数的严格限制。
4. 测试配套：本次变更未直接包含测试文件，但 PR 评论中触发了 CI 测试（如 test_pp_single_node.py），以验证修复效果。

关键文件：

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 _get_new_batch_prefill_raw）：这是调度器的核心文件，修改了预填充批次生成的关键条件逻辑，直接修复了 PP 下的挂起问题。

关键符号：_get_new_batch_prefill_raw

关键源码片段

python/sglang/srt/managers/scheduler.py

这是调度器的核心文件，修改了预填充批次生成的关键条件逻辑，直接修复了 PP 下的挂起问题。

```
def _get_new_batch_prefill_raw(self):
    # ... 其他逻辑 ...
    running_bs = len(self.running_batch.reqs)

    # 忽略检查如果 self.chunked_req 不为 None。
    # 在非 PP 情况下，当 self.chunked_req 不为 None 时，num_allocatable_reqs 应始终大于 0，
    # 因为分块请求的空间刚刚被释放。
    # 在 PP 情况下，分块请求（或 dllm
    请求）可能在一个微批次开始，在另一个微批次结束，因此每微批次的最大运行请求数不应严格限制
    。
    # 相反，我们应始终允许分块请求被添加，否则会导致内存泄漏。
    if (
        self.get_num_allocatable_reqs(running_bs) <= 0
        and self.chunked_req is None # 修复：仅当没有分块请求且槽位不足时才阻止新批次
        and not self.enable_priority_preemption
    ):
        self.running_batch.batch_is_full = True
        return None
    # ... 后续逻辑 ...
```

评论区精华

Review 中仅有一名审核者（ShangmingCai）批准，评论指出逻辑正确并感谢深入挖掘。未出现争议或未解决疑虑。

- 逻辑正确性确认 (correctness)：变更被接受，未提出修改意见。

风险与影响

- 风险：回归风险：变更仅涉及单行条件反转，影响范围局限在调度器的预填充批次生成逻辑。若反转错误，可能导致在非 PP 场景下，当存在分块请求且槽位不足时，错误地允许新批次生成，可能引发资源竞争或内存问题。但根据注释，非 PP 场景下 num_allocatable_reqs 在分块请求释放后应大于 0，因此风险较低。 性能风险：无直接影响，修复的是功能正确性问题。 兼容性风险：与所有支持流水线并行的 SGLang 服务器版本兼容，恢复至原始设计行为。
- 影响：用户影响：使用流水线并行的用户将不再遇到服务器挂起问题，特别是处理长提示词时。影响范围限于 PP 配置下的预填充请求。 系统影响：修复了调度器核心路径中的一个阻塞缺陷，确保请求能正常推进，提升系统可靠性。 团队影响：揭示了历史 PR 中细微条件反转引入的 bug，提醒团队在修改核心调度逻辑时需谨慎验证边界条件。
- 风险标记：核心路径变更 , 条件逻辑风险

关联脉络

- PR #5724 初始流水线并行支持 PR: PR body 提及此 PR 最初引入了流水线并行，其原始逻辑允许分块请求在条件 not self.chunked_req 下推进。
- PR #14883 引入条件反转的 PR: PR body 指出 bug 源于此 PR 将条件从 not self.chunked_req 错误地改为 self.chunked_req is not None。