PR #29017 完整报告

sgl-project/sglang

[model-gateway] PD router: cancel paired decode when prefill fails

合并时间：2026-07-03 08:03

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29017

PR #29017 分析报告：PD 路由 prefill 失败时取消 decode 请求

执行摘要

本 PR 将 PD 分离部署中路由器的 prefill 和 decode 请求并发控制从 tokio::join! 改为 tokio::select!，实现了 prefill 请求失败时立即丢弃 decode 请求并关闭连接，使 decode 插槽释放时间从 300s 缩短至约 4-8s。这是引擎侧修复 #28651 在路由层的补充，覆盖了 prefill HTTP 失败场景，显著降低级联死锁风险。

功能与动机

在 PD 分离部署模式下，decode 节点必须等待 prefill 节点传输 KV cache 后才能开始生成。若 prefill 请求因超时、5xx、连接拒绝等原因失败，decode 请求将在 KVPoll.WaitingForInput 状态阻塞长达 300s（默认 SGLANG_DISAGGREGATION_WAITING_TIMEOUT），期间占用解码槽位，导致并发度下降。当多个 prefill 失败时，形成级联死锁，集群恢复时间可达 1 小时以上。

关联 Issue #28651 描述了类似的生产事故："The 300s decode hang per aborted request was the key amplifier that turned a transient prefill disruption into a prolonged cluster-wide outage."

实现拆解

1. 并发模型改造（pd_router.rs:execute_dual_dispatch_internal） - 将原来的 tokio::join!(prefill_fut, decode_fut) 替换为 tokio::select! 循环，优先等待 prefill 完成（biased 模式）。 - decode 如果先完成，暂存其结果但不中断循环，继续等待 prefill。 - 两个 future 仍然在同一个 handler 任务中运行（未使用 tokio::spawn），保留客户端断开时自动取消的行为。
2. 快速失败路径 - prefill 返回非 2xx 或传输错误时，记录警告日志，并调用 process_prefill_response 生成 HTTPS 响应：4xx 透传、5xx/ 传输错误转换为 502。 - 通过释放作用域隐式 drop decode future，关闭其 HTTP 连接。decode 引擎在 4-8s 内检测到连接断开并调用 abort_request()，释放解码槽位。
3. 断路器归属处理 - 仅根据 prefill 实际状态记录断路器结果：4xx 记为客户端错误（不影响断路器熔断），其余不记录。 - decode 端不记录（避免 prefill 故障风暴误伤正常 decode

断路器）。 - 响应中插入 BreakerOutcomesRecorded 标记，通知外层调度器跳过重复记录。

4. 正常路径完全不变 - prefill 成功后，从 decode_early 或再次 await decode future 获取结果，后续处理（状态检查、流式处理、断路器跟踪等）与原来一致。

sgl-model-gateway/src/routers/http/pd_router.rs

核心变更文件，PD 路由的并发控制从 tokio::join! 重构为 tokio::select!，实现 prefill 失败时快速关闭 decode 连接。

```
// sgl-model-gateway/src/routers/http/pd_router.rs

// 将 prefill 和 decode 的 future 固定，确保它们在同一个任务中运行
let prefill_fut = prefill_request.send();
let decode_fut = decode_request.send();
tokio::pin!(prefill_fut);
tokio::pin!(decode_fut);

let prefill_result;
let mut decode_early: Option<Result<reqwest::Response, reqwest::Error>> = None;
loop {
    tokio::select! {
        biased; // 优先 poll prefill，避免 decode 先完成时误判
        pr = &mut prefill_fut => {
            prefill_result = pr;
            break;
        }
        dr = &mut decode_fut, if decode_early.is_none() => {
            // decode 先完成则暂存结果，但不中断等待 prefill
            decode_early = Some(dr);
        }
    }
}

// 判断 prefill 是否失败（非 2xx 或传输错误）
let prefill_failed = match &prefill_result {
    Ok(resp) => !resp.status().is_success(),
    Err(_) => true,
};

if prefill_failed {
    warn!(
        "Prefill failed, aborting paired decode request decode_url={} prefill_url={}",
        decode.url(),
        prefill.url()
    );

    // 仅记录 prefill 断路器结果，decode 不记录避免误伤
    let prefill_ok = match &prefill_result {
```

```
            Ok(r) => r.status().is_client_error(),
            Err(_) => false,
        };
        prefill.record_outcome(prefill_ok);

        // 生成错误响应（4xx 透传，5xx/ 传输错误 -> 502）
        let mut response = match self
            .process_prefill_response(prefill_result, prefill.url(), false)
            .await
        {
            Err(error_response) => error_response,
            Ok(_) => error::bad_gateway(
                "prefill_server_error",
                "Prefill reported failure but returned a success response".to_string(),
            ),
        };
        response.extensions_mut().insert(BreakerOutcomesRecorded);
        return response;
    }

    // Prefill 成功：获取 decode 结果（若尚未完成则等待）
    let decode_result = match decode_early {
        Some(dr) => dr,
        None => (&mut decode_fut).await,
    };

    // 后续处理与之前一致（状态检查、流式处理等）
```

评论区精华

> Gemini Code Assist: " 优化建议：decode future 先于 preflight 完成时，可以立即 fail-fast 并返回，因为 PD 模式下 decode 不可能成功。"
>
> chenkaiyue: " 保留当前实现，原因：① 极罕见情况下 decode 可能返回 2xx（时序竞态）；② 早期返回会重复下游已有的 decode 错误处理逻辑；③ 性能收益微乎其微。"

此讨论体现了对边缘情况的严谨考虑：虽然理论上 decode 不能先成功，但实践中可能存在非常短的生成序列，时序竞态可能导致 decode 意外返回 2xx，直接 fail-fast 反而会误报。

风险与影响

- 回归风险：tokio::select! 的 biased 模式和行为与 join! 不同，若理解偏差可能导致任务泄漏或死锁。但代码通过 tokio::pin! 确保生命周期，且已通过生产环境验证。
- 兼容性风险：decode 端不再记录断路器结果，若其他组件依赖此计数可能需要同步更新。
- 测试覆盖不足：本次变更仅修改源码，未添加测试，建议 merge 前补充单元测试覆盖 prefill 失败、decode 先完成、客户端断开等场景。

关联脉络

- #28651（引擎侧修复）：在 mooncake 层面实现 prefill 中止时通知 decode 对端。本 PR 是路由层补充，覆盖 prefill HTTP 失败场景（如 4xx/5xx、连接拒绝）。两者结合后，PD 模式下 prefill 故障不再导致 decode 长时间阻塞。
- 近期历史：多 PR 围绕 PD 分离部署的稳定性改进（如 #29756 MiniMax M3 状态传输修复、#29982 AMD FlashMLA 稀疏 prefill 关闭），反映了该仓库对生产环境健壮性的持续投入。