PR #7129 完整报告

PaddlePaddle/FastDeploy

[Feature] Fix mixed cache-aware

合并时间：2026-04-01 19:29

原文链接：http://prhub.com.cn/PaddlePaddle/FastDeploy/pull/7129

执行摘要

本 PR 修复了 FastDeploy 中 mixed 模式下 cache-aware 调度策略的两个关键问题：SelectWorker 调用时传递空参数导致调度决策失效，以及资源释放时缺少 ReleasePrefillTokens 调用导致 token 计数器泄漏。变更仅涉及一个文件（completions.go）的 4 行代码，使 mixed 模式与 PD 模式的实现逻辑保持一致，确保调度策略的正确性和资源管理的完整性。

功能与动机

根据 PR 描述，需要修复 mixed cache-aware 策略中的释放和 selectworker 逻辑。具体来说：

- 问题 1：在非 PD（mixed）模式下，原代码调用 manager.SelectWorker(ctx, "") 时传递空字符串，导致 cache-aware 调度策略无法根据请求内容（message）做出正确决策。
- 问题 2：在 defer 函数中释放资源时，只调用了 scheduler_handler.Release，缺少 scheduler_handler.ReleasePrefillTokens 调用，导致 prefill tokens 计数器未正确递减，可能引发资源泄漏。

这些修复旨在确保 mixed 模式下的调度行为与 PD 模式保持一致，提升系统稳定性和资源利用率。

实现拆解

变更集中在 fastdeploy/golang_router/internal/gateway/completions.go 文件的 CommonCompletions 函数中：

关键改动点

1. SelectWorker 参数修复（第 415 行）： `go // 原代码 dest, err := manager.SelectWorker(ctx, "") // 新代码 message = extractor(rawReq) dest, err := manager.SelectWorker(ctx, message)` 通过提取 message 并传递给 SelectWorker，使调度器能基于请求内容进行 cache-aware 决策。
2. ReleasePrefillTokens 调用补充（第 431 行）： `go defer func() { for _, url := range releaseTargets { scheduler_handler.Release(ctx, url) scheduler_handler.ReleasePrefillTokens(ctx, url, message) // 新增 } }()` 在资源释放循环中新增 ReleasePrefillTokens 调用，确保 token 计数器正确递减。

模块关联

- 所属模块：APIServer（gateway 层）
- 影响组件：调度器（Scheduler）、缓存管理器（KVCache）
- 一致性目标：使 mixed 模式实现与 PD 模式（如 PR #7107 中的逻辑）对齐

评论区精华

Review 中仅包含 AI Code Review 的自动分析，没有人工讨论。AI 分析要点如下：

> 总体评价：本次变更正确修复了非 PD（mixed）模式下 cache-aware 策略的两个问题：1. SelectWorker 调用修复：原代码传递空字符串 ""，导致 cache-aware 调度策略无法根据请求内容做出正确决策。现在正确提取 message 并传递给 SelectWorker。 2. ReleasePrefillTokens 调用补充：在 defer 释放资源时新增 ReleasePrefillTokens 调用，确保 token 计数器正确递减，与 PD 模式（Splitwise）的实现保持一致。
>
> 代码实现清晰，与已有的 PD 模式逻辑对齐，建议合入。

由于缺乏人工讨论，未发现设计权衡或争议点。

风险与影响

技术风险

- 低风险：变更范围小，逻辑清晰，主要风险已通过 AI 分析识别：
  - message 提取可能失败或返回空值，但原代码已有错误处理机制。
  - ReleasePrefillTokens 调用可能引入微小性能开销，但属于必要的资源清理。
- 回归风险：由于修复了资源泄漏问题，可能影响现有测试中对资源使用量的断言，但这是正向改进。

影响范围

- 用户影响：修复后 mixed 模式下的 cache-aware 调度能正确工作，提升请求处理效率和资源利用率。
- 系统影响：确保 token 计数器正确递减，避免 KVCache 资源泄漏，提升系统稳定性。
- 团队影响：代码更一致（mixed 与 PD 模式对齐），减少维护复杂度和潜在 bug。

关联脉络

与历史 PR 的关联

1. PR #7125（Config eviction_duration）：修改了相同目录下的 completions_test.go 文件，且都涉及 cache-aware 和调度相关功能，表明该模块近期在持续优化。
2. PR #7107（PD Disaggregation）：都涉及 KVCache 和调度逻辑优化，本 PR 的 ReleasePrefillTokens 补充与 PD 模式资源管理（如 cache 写入 storage）有协同关系。

3. PR #6992（/v1/abort_requests 端点）：都修改了 APIServer 相关代码，涉及请求处理和资源管理，反映 APIServer 层在完善生命周期管理。

演进趋势

从近期 PR 分析可见：

- 调度策略精细化：PR #7125、#7107、#6992 和本 PR 都围绕调度器（Scheduler）和 KVCache 进行优化，表明团队在持续提升调度效率和资源管理能力。
- 模式统一化：本 PR 将 mixed 模式逻辑向 PD 模式对齐，是减少代码分支、提升一致性的典型实践。
- APIServer 增强：多个 PR（#7054、#6992、#7082）都在扩展 APIServer 功能，本 PR 修复的 CommonCompletions 函数是请求处理的核心路径之一。

本 PR 虽小，但填补了 mixed 模式下 cache-aware 策略的关键缺口，是调度系统演进中的重要一环。