PR #6680 完整报告

PaddlePaddle/FastDeploy

[Optimization] Optimize ttft for prefill pd

合并时间：2026-03-30 20:36

原文链接：http://prhub.com.cn/PaddlePaddle/FastDeploy/pull/6680

执行摘要

- 一句话：优化 PD 预填充场景下的调度逻辑，减少排队并提升批处理效率。
- 推荐动作：面向技术管理者和工程师，建议：
- 精读重点：该 PR 值得精读，特别是 engine_forward_signal 的设计和调度时机变化，这些是性能优化的关键决策点。
- 关注设计：留意讨论中的并发优化建议和接口语义问题，可应用于其他调度优化场景。
- 跟进风险：建议后续补充测试覆盖，并监控生产环境中的性能表现和并发问题。

功能与动机

根据 PR body 描述，动机是“之前 scheduler 和 worker 是完全异步分开进行的，schedule() 会在引擎 forward 开始前发生并立即结束，容易造成每次调度的请求量不满，从而让请求易发生排队”。优化目的是在引擎 forward 结束后再进行 schedule()，以达到最佳组 batch 效果。

实现拆解

实现方案拆解为以下模块：

1. 引擎层（Engine）：在 fastdeploy/engine/common_engine.py 中新增 engine_forward_signal（IPCSignal），用于标记 forward 状态；修改 _schedule_request_to_worker_v1 函数，在 forward 期间累积请求（通过检查 engine_forward_signal.value[0] != 0），结束后统一调度；为 EP 并行添加空任务推送逻辑以同步 worker。
2. Worker 层：在 fastdeploy/worker/worker_process.py 中修改 event_loop_normal 函数，在检测到新请求时设置 engine_forward_signal 为 1，forward 结束时置 0；调整任务处理逻辑，支持空任务 barrier；简化 init_health_status 以初始化信号。
3. 调度器层（Scheduler）：在 fastdeploy/scheduler/dp_scheduler.py 中简化 get_requests 函数，移除资源检查逻辑，改为只取一个请求，以适应 V1 资源管理。
4. 文档和环境变量：移除 FD_EP_BATCHED_TOKEN_TIMEOUT 环境变量及其相关文档（docs/usage/environment_variables.md, fastdeploy/envs.py）。
5. 测试更新：调整测试文件（如 tests/engine/test_common_engine.py, tests/scheduler/test_dp_scheduler.py）以适应新逻辑，并注释不稳定测试。

关键文件：

- fastdeploy/engine/common_engine.py（模块 Engine）：核心调度逻辑修改，新增 engine_forward_signal 并调整 _schedule_request_to_worker_v1 函数，决定调度时机和累积机制。
- fastdeploy/worker/worker_process.py（模块 Worker）：worker 处理逻辑调整，更新 engine_forward_signal 状态、处理空任务同步，影响 forward 期间的请求接收和调度。
- fastdeploy/scheduler/dp_scheduler.py（模块 Scheduler）：调度器简化，get_requests 函数移除资源检查逻辑，接口语义变化，需关注与 V1 资源管理的兼容性。

关键符号：_schedule_request_to_worker_v1, event_loop_normal, init_health_status, get_requests

评论区精华

review 讨论的核心要点包括：

- 锁竞争优化：Copilot 建议在 common_engine.py 中使用 exist_tasks() 代替 num_tasks() 以减少锁竞争，rainyfly 已采纳修改。
- 空任务推送开销：Copilot 指出 EP 空闲时频繁推送空任务可能增加 CPU/IPC 开销，rainyfly 解释为同步需要，保持原逻辑。
- 信号同步问题：Copilot 担忧 engine_forward_signal 在多 TP rank 下的并发写入可能导致竞态，建议只让 tp_rank 0 更新，但 rainyfly 认为无影响，未采纳。
- 接口语义模糊：Copilot 批评 dp_scheduler.py 中 get_requests 函数参数未使用，接口语义不一致，此问题未明确解决，可能需后续重构。
- 逻辑细节与注释：Jiang-Jia-Jun 强调添加注释和警告日志，rainyfly 已更新注释并改用 assert 处理空任务。
  - engine_forward_signal 锁竞争优化 (performance): rainyfly 采纳建议并修改为使用 exist_tasks()，以减少开销。
  - 空任务推送频率开销 (performance): rainyfly 解释为同步需要，保持原逻辑，认为开销可接受。
  - engine_forward_signal 多 TP 同步 (correctness): rainyfly 表示多 TP 下无影响，未采纳建议，认为当前逻辑正确。
  - DP scheduler 接口语义 (design): 未明确解决，rainyfly 未回复此点，可能需要后续重构或文档更新。
  - worker 处理逻辑注释和细节 (correctness): rainyfly 添加注释并改用 assert 处理空任务，部分细节已澄清。

风险与影响

- 风险：技术风险具体如下：
  1. 并发风险：engine_forward_signal 在多 TP rank 下的写入可能引入竞态，影响调度时机正确性（fastdeploy/worker/worker_process.py）。
  2. 性能开销：在 EP 空闲时频繁推送空任务（fastdeploy/engine/common_engine.py）可能增加不必要的 IPC 和 CPU 开销，影响系统功耗和延迟抖动。

3. 接口兼容性：dp_scheduler.py 的 get_requests 函数参数未使用，接口语义模糊，可能误导后续维护者或调用方。
4. 测试覆盖不足：Codecov 报告显示 50.9% 的补丁覆盖率，新增调度逻辑（如 engine_forward_signal gating）缺少单元测试，增加回归风险。
5. 回归风险：移除 FD_EP_BATCHED_TOKEN_TIMEOUT 环境变量可能影响依赖此超时的旧配置或系统。

- 影响：影响范围和程度：
- 用户影响：对使用 PD disaggregation 的用户，此优化有望减少 TTFT，提升请求响应速度和吞吐量，改善用户体验。
- 系统影响：修改核心调度路径（引擎和 worker 交互），可能影响所有 PD 相关推理任务；性能提升但需监控并发开销。
- 团队影响：工程师需理解新调度时机和信号机制，以避免后续修改引入回归；技术管理者可借鉴设计决策进行类似优化。
- 风险标记：核心路径变更，并发风险，接口语义模糊，测试覆盖不足

关联脉络

- 暂无明显关联 PR