PR #17913 完整报告

sgl-project/sglang

[Feature] add LoRADrainer to address high P99 TTFT

合并时间：2026-05-03 07:13

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/17913

执行摘要

- 一句话：新增 LoRA 排空器（LoRADrainer）降低尾部延迟
- 推荐动作：建议精读 LoRADrainer 的设计：通过 starvation 检测和 greedy draining 选择（优先排空剩余 token 最少的适配器）是一种经典的公平调度启发式，值得在类似场景复用。同时注意其默认关闭的设计体现了对主流性能的谨慎。

功能与动机

根据 PR body 提供的性能数据，当前 LoRA 实现在 6 个适配器、请求速率 4 的压力下，P99 TTFT 接近 40 秒（45870ms），P99 E2E 同样惊人（45870ms），严重影响服务质量。本 PR 旨在通过主动排空策略，防止少数适配器长时间霸占 batch 槽位，大幅降低尾延迟。

实现拆解

1. 新增 LoRADrainer 类（python/sglang/srt/lora/lora_drainer.py）：包含 AdapterStats 数据类，跟踪每个适配器的等待请求数、最大等待时间和运行中剩余 token；核心逻辑在 update_draining_state 中，每次调度前调用，先更新统计，再检测饥饿适配器，然后选择剩余 token 最少的运行中适配器标记为 draining。
2. 调度器集成（python/sglang/srt/managers/scheduler.py）：在 _get_new_batch_prefill_raw 中插入 lora_drainer.update_draining_state 调用；提取 _can_schedule_lora_req 方法，其中首先检查 drainer 的 can_schedule 接口，若请求的适配器正在被 draining 则暂不调度。
3. 新增服务器参数--lora-drain-wait-threshold（python/sglang/srt/server_args.py）：float 类型，默认 0.0（关闭）。仅当该参数 >0 时才创建 LoRADrainer 实例。
4. 单元测试（test/registered/lora/test_lora_drainer.py）：验证 draining 标记逻辑、can_schedule 的容错（DRAIN_SCHEDULE_TOLERANCE = 1.2）以及 batch splitting 集成测试。
5. 辅助测试工具（python/sglang/test/lora_utils.py、runners.py）：透传 lora_drain_wait_threshold 参数支持集成测试。

关键文件：

- python/sglang/srt/lora/lora_drainer.py（模块 排空策略；类别 source；类型 core-logic；符号 AdapterStats, _reset_stats, is_starving, LoRADrainer）：新增 LoRADrainer 类，核心排空逻辑实现

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 _can_schedule_lora_req）：调度器集成 LoRADrainer，在 batch 构建前调用排空逻辑并过滤不可调度请求
- python/sglang/srt/server_args.py（模块 配置参数；类别 source；类型 configuration）：新增 --lora-drain-wait-threshold 参数并校验
- test/registered/lora/test_lora_drainer.py（模块 单元测试；类别 test；类型 test-coverage；符号 make_req, TestLoRADrainer, test_update_draining_marks_adapter, test_can_schedule_respects_draining_tolerance）：单元测试覆盖 draining 标记、can_schedule 容错和 batch splitting 集成

关键符号：LoRADrainer, AdapterStats, update_draining_state, _update_adapter_stats, _update_draining_loras, can_schedule, _can_schedule_lora_req, make_req, TestLoRADrainer, test_update_draining_marks_adapter, test_can_schedule_respects_draining_tolerance, test_batch_splitting_with_drainer

关键源码片段

python/sglang/srt/managers/scheduler.py

调度器集成 LoRADrainer，在 batch 构建前调用排空逻辑并过滤不可调度请求

调度器中的 LoRA 请求许可检查

```
def _can_schedule_lora_req(
    self, req: Req, running_loras: set[Optional[str]]
) -> bool:
    # 如果启用了 LoRADrainer 且该请求的 adapter 正在排空状态，拒绝调度
    if self.lora_drainer and not self.lora_drainer.can_schedule(req):
        return False

    # 如果请求的 adapter 已经在运行中，可以直接调度
    if req.lora_id in running_loras:
        return True

    # 尝试加载新的 adapter
    if self.enable_lora_overlap_loading:
        # 重叠加载模式：逐个加载 adapter
        return self.lora_overlap_loader.try_overlap_load_lora(
            req.lora_id, running_loras
        )
    else:
        # 常规模式：检查新 adatper 集合是否在 batch 容量内
        new_lora_set = {req.lora_id} | running_loras
        return self.tp_worker.model_runner.lora_manager.validate_lora_batch(
            new_lora_set
        )
```

调度器中的 LoRA 请求许可检查

```
def _can_schedule_lora_req(
    self, req: Req, running_loras: set[Optional[str]]
) -> bool:
    # 如果启用了 LoRADrainer 且该请求的 adapter 正在排空状态，拒绝调度
    if self.lora_drainer and not self.lora_drainer.can_schedule(req):
        return False

    # 如果请求的 adapter 已经在运行中，可以直接调度
    if req.lora_id in running_loras:
        return True

    # 尝试加载新的 adapter
    if self.enable_lora_overlap_loading:
        # 重叠加载模式：逐个加载 adapter
        return self.lora_overlap_loader.try_overlap_load_lora(
            req.lora_id, running_loras
        )
    else:
        # 常规模式：检查新 adatper 集合是否在 batch 容量内
        new_lora_set = {req.lora_id} | running_loras
        return self.tp_worker.model_runner.lora_manager.validate_lora_batch(
            new_lora_set
        )
```

test/registered/lora/test_lora_drainer.py

单元测试覆盖 draining 标记、can_schedule 容错和 batch splitting 集成

测试辅助函数和 Draining 标记测试

```
def make_req(lora_id, wait_queue_entry_time, max_new_tokens, output_len=0):
    time_stats = SimpleNamespace(wait_queue_entry_time=wait_queue_entry_time)
    sampling_params = SimpleNamespace(max_new_tokens=max_new_tokens)
    req_ns = SimpleNamespace(
        lora_id=lora_id,
        time_stats=time_stats,
        sampling_params=sampling_params,
        output_ids=[0] * output_len,
    )
    return cast(Req, req_ns)


class TestLoRADrainer(unittest.TestCase):
    def test_update_draining_marks_adapter(self):
        if is_in_ci():
            return

        with mock.patch('time.monotonic', return_value=MOCK_START_TIME):
            drainer = LoRADrainer(
                max_loras_per_batch=1,
```

```
        max_wait_time_secs=LORA_DRAIN_WAIT_THRESHOLD
    )

    wait_entry = MOCK_START_TIME - (LORA_DRAIN_WAIT_THRESHOLD + 0.01)
    waiting_req = make_req('A', wait_entry, max_new_tokens=10)
    running_req = make_req('B', wait_entry, max_new_tokens=100, output_len=0)

    drainer.update_draining_state(
        waiting_queue=[waiting_req],
        running_reqs=[running_req],
    )

    # 运行中的 adapter B 应被标记为 drain 给 A
    self.assertEqual(drainer.adapter_to_stats['B'].is_draining_for, 'A')

    # B 完成后标记应清除
    drainer.update_draining_state(
        waiting_queue=[waiting_req],
        running_reqs=[]
    )
    self.assertIsNone(drainer.adapter_to_stats['B'].is_draining_for)
```

评论区精华

- Fridge003指出 drain 策略会损害 median latency/TTFT（从 83.30ms 升至 3728.36ms），因此作者新增 server 参数并默认关闭，按需启用。
- gemini-code-assist[bot]建议无条件初始化 running_loras 变量以避免 NameError，但该建议未在后续 commit 中体现（最终代码仍放在 if enable_lora 块内，但实际安全）。
- Drainer 对 median latency 的影响 (design): 作者新增 --lora-drain-wait-threshold 参数，默认 0.0（关闭），让用户按需启用。
- running_loras 变量初始化问题 (correctness): 作者未直接采纳，但实际 _can_schedule_lora_req 仅在 enable_lora 为真时调用，因此安全。

风险与影响

- 风险：
  - 启用后 median 延迟显著上升：这是设计上的 trade-off，默认关闭可避免对主流场景的影响。
  - 核心调度路径变更：_get_new_batch_prefill_raw 是调度的核心，新增的 draining 逻辑若存在 bug 可能导致所有 LoRA 请求调度阻塞。单元测试覆盖了主要分支，但仍需集成测试验证。
  - DRAIN_SCHEDULE_TOLERANCE 硬编码：值为 1.2，若与剩余 token 估算偏差大可能造成不公平。
- 影响：

- 用户：默认无变化；启用 drainer 后，P99 TTFT 可降低 70%，但 median 显著升高，适合尾部延迟敏感场景（如 SLO 对 P99 有严格要求的服务）。影响范围可选且可控。
- 系统：新增一个调度前 O(n) 的统计更新，对性能影响可忽略。LoRADrainer 内部使用 defaultdict，内存占用低。
- 团队：新增一个独立模块，维护成本低；测试覆盖了单元和集成。
- 风险标记：核心调度路径变更，默认关闭，Median 损害风险，单元测试覆盖

关联脉络

- PR #22125 throw ValueError for DoRA adapters: 同属于 SGLang LoRA 子系统，本 PR 新增的 LoRADrainer 与 LoRA 配置管理器协同工作。