PR #33595 完整报告

sgl-project/sglang

Measure prefill busy time between launches

合并时间：2026-08-06 02:26

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33595

执行摘要

- 一句话：改为 launch 间隔统计 prefill/decode 忙时，统一负载指标
- 推荐动作：值得精读，虽然改动量不大，但它是调度器负载统计口径的一次重要统一设计。可以重点关注 _record_step_counters 中样本过滤条件的组合，以及 after_idle_gap 标记在重叠事件循环中的传递路径。建议后续补充针对边界条件的单元测试，以保护该统计逻辑的稳定性。

功能与动机

PR body 明确指出：Prefill busy time currently spans from batch launch until result processing. Under scheduler overlap, that includes unrelated CPU scheduling work and makes prefill accounting inconsistent with the existing launch-to-launch decode metric. 因此需要将 prefill 统计改为与 decode 相同的 launch 间隔口径，保证 autoscale 等负载判断一致性。

实现拆解

1. 数据模型扩展：在 ScheduleBatch 类中新增 after_idle_gap: bool = False 字段，并在 copy() 方法中透传该字段，用于标记当前批次是否为调度器空闲后的首个批次。
2. 调度器状态追踪：在 Scheduler 中，将原有的 _prev_decode_launch_ts 替换为 _prev_step: Optional[Tuple[int, float, bool]]，保存上一次的前向迭代号、launch 时间戳和 prefill 标志；同时在 init_load_inquirer 中新增 _sched_idled = False 状态，并在 event_loop_normal 与 event_loop_overlap 中，当批次为空时（batch is None）置位 _sched_idled = True。
3. launch 入口标记：run_batch 中在记录 launch_ts 后，将 _sched_idled 写入 batch.after_idle_gap，然后立即复位 _sched_idled = False，从而让后续的统计逻辑感知到 idle 空档。
4. 统计逻辑重写：_record_step_counters 改为基于 _prev_step 与当前批次的 launch 时间戳计算间隔 step_us，并依次丢弃四类无效样本：无前置记录、after_idle_gap 为真、前后迭代号不连续、prefill/decode 模式切换；在合法区间 0 < step_us < STEP_MAX_US 内，prefill 累加 total_prefill_busy_us，decode 调用 _accumulate_decode_moment。
5. 常量和测试配套：将 DECODE_STEP_MAX_US 重命名为 STEP_MAX_US 以覆盖两种统计；第二个提交删除了原来针对 step counter 的单元测试，因此该改动目前没有新增自动化

测试覆盖。

关键文件：

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 _record_step_counters, _prev_step, _sched_idled, run_batch）：核心统计逻辑所在文件，包含 _record_step_counters 重写、_prev_step 状态引入、_sched_idled 标记以及常量重命名。
- python/sglang/srt/managers/schedule_batch.py（模块 调度器；类别 source；类型 core-logic；符号 ScheduleBatch.after_idle_gap, ScheduleBatch.copy）：为 ScheduleBatch 新增 after_idle_gap 字段并在 copy() 中透传，是统计逻辑传递 idle 状态的关键数据通路。

关键符号：_record_step_counters, run_batch, init_load_inquirer, ScheduleBatch.copy

关键源码片段

python/sglang/srt/managers/scheduler.py

核心统计逻辑所在文件，包含 _record_step_counters 重写、_prev_step 状态引入、_sched_idled 标记以及常量重命名。

```
# _record_step_counters 核心统计逻辑（重新整理）
def _record_step_counters(self, batch, result):
    # 健康检查请求不参与负载统计，直接返回
    if all(is_health_check_generate_req(req) for req in batch.reqs):
        return

    # 根据 forward_mode 判断当前批次是否为 prefill
    is_prefill = batch.forward_mode.is_prefill()
    prev = self._prev_step
    # 先记录当前步，之后再用旧值计算间隔
    self._prev_step = (batch.forward_iter, batch.launch_ts, is_prefill)

    # 丢弃无前置记录、idle 空档后的首个批次
    if prev is None or batch.after_idle_gap:
        return

    prev_iter, prev_ts, prev_is_prefill = prev
    # 丢弃 forward 迭代不连续（如插入其他 batch 或跳步）以及模式切换的样本
    if prev_iter + 1 != batch.forward_iter or prev_is_prefill != is_prefill:
        return

    # 相邻两次 launch 的时间间隔（微秒），这里才是真正的忙时口径
    step_us = int((batch.launch_ts - prev_ts) * 1e6)
    # 丢弃非法时长，避免时钟异常或过大间隔污染统计
    if not 0 < step_us < STEP_MAX_US:
        return

    if is_prefill:
```

```
            self.total_prefill_busy_us += step_us
            self.total_prefill_uncached_tokens += batch.extend_num_tokens
        else:
            # decode 沿用原有动量累积，但间隔也改为 launch-to-launch
            _accumulate_decode_moment(
                self.decode_moment_totals,
                len(batch.reqs),
                step_us,
                len(batch.reqs) + result.num_correct_drafts,
            )
```

python/sglang/srt/managers/schedule_batch.py

为 ScheduleBatch 新增 after_idle_gap 字段并在 copy() 中透传，是统计逻辑传递 idle 状态的关键数据通路。

```
# ScheduleBatch 中新增的字段定义（节选）
class ScheduleBatch(...):
    # ... 其他字段 ...
    # Metrics
    dp_cooperation_info: Optional[DPCooperationInfo] = None
    prefill_stats: Optional[PrefillStats] = None
    forward_iter: Optional[int] = None
    launch_ts: Optional[float] = None
    # 标记当前批次是否紧跟在调度器空闲之后，用于统计时丢弃跨 idle 的样本
    after_idle_gap: bool = False

    def copy(self):
        # ... 其他参数 ...
        return ScheduleBatch(
            # ... 其余构造参数 ...
            forward_iter=self.forward_iter,
            launch_ts=self.launch_ts,
            after_idle_gap=self.after_idle_gap,
            extend_num_tokens=self.extend_num_tokens,
        )
```

评论区精华

该 PR 没有公开的 review 评论或讨论线程，仅有两次 /tag-and-rerun-ci 触发 CI 的指令。作者在 PR body 中说明此变更只影响调度器指标记账，不影响模型输出，因此未提供测试和性能数据。

- 暂无高价值评论线程

风险与影响

- 风险：风险主要在于统计口径变更可能影响 autoscale 等依赖负载指标的功能：
  1. _prev_step 只在连续同模式批次间累积，若长期 idle 或频繁模式切换，样本量可能减少，导致忙时估算偏低；

2. after_idle_gap 标记依赖 run_batch 入口，prebuilt batch 等提前返回路径是否同样设置尚未验证；
3. 该 PR 明确没有单元测试覆盖，_record_step_counters 的边界条件（如 forward_iter 不连续、时长阈值）缺少回归保护。 但这些风险仅影响指标统计精度，不直接影响模型推理正确性。

- 影响：影响范围集中在调度器内部的负载统计模块，涉及 scheduler.py 和 schedule_batch.py 两个文件。用户可见行为无变化，但 autoscale 和负载监控相关的下游逻辑可能观察到不同的 prefill busy 时间数值。对团队而言，后续在 overlap 调度下需要重新校准基于该指标设置的阈值。
- 风险标记：核心调度指标变更，缺少单元测试覆盖

关联脉络

- PR #33403 [Scheduler] Honor explicit min-free-slots thresholds: 同为调度器负载控制相关改动，修改了 min-free-slots 延迟器的阈值逻辑，与本次 load metrics 统计口径调整共同影响调度器的负载判断和 autoscale 行为。