# PR #29217 完整报告

sgl-project/sglang

[MLX] Fix step-bounded profiling for bench tools on Apple Silicon

合并时间：2026-07-01 13:55

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29217

## 执行摘要

- 一句话：修复 MLX 后端步长限制分析无法自动停止的问题
- 推荐动作：值得精读。该 PR 展示了在非标准调度路径中准确插桩 profiling 钩子的模式，以及如何通过幂等设计提高 CLI 工具的健壮性。对于维护 MLX 后端或类似定制调度逻辑的开发者有直接参考价值。

## 功能与动机

PR #29217 是 Apple 设备支持路线图（#19137）Profiling 部分的后续。在 MLX 后端，bench_offline_throughput、bench_one_batch_server 和 bench_serving 无法产出可用的 Metal trace：step-bounded profiling 从不自动停止，导致 GPU 捕获整个生成过程，在 MTL_CAPTURE_ENABLED=1 下更慢，最终调度器 watchdog（300s）超时 SIGQUIT 杀掉服务器。根原因是 MLX overlap loop（event_loop_overlap_mlx）跳过了 Scheduler.run_batch()，而 forward_ct 递增和 profiler predicate 调用恰在 run_batch 中，导致 forward_ct 始终为 0，profiler 的 step 条件无法满足，watchdog 也因 liveness 计数器不更新而误判。

## 实现拆解

1. 在 MLX 重叠调度关键路径添加 profiler 支持：在 python/sglang/srt/hardware_backend/mlx/scheduler_mixin.py 的 _finalize_mlx_pending_job 方法开头增加 self.forward_ct += 1 和 self.profiler_manager._profile_batch_predicate(pending.schedule_batch)，模拟 run_batch() 中的行为。
2. 使 bench 工具 stop_profile 幂等：在 python/sglang/benchmark/offline_throughput.py 中，将直接调用 backend.stop_profile() 替换为 try: backend.stop_profile() except RuntimeError: pass，以兼容调度器自动停止后再次调用的情况。
3. 新增单元测试：新增 test/registered/unit/hardware_backend/mlx/test_scheduler_mixin.py，覆盖 forward_ct 递增和 profiler predicate 调用，验证每次 job finalize 恰好对应一次计数和一次 predicate，且 process_batch_result 仍正常执行。

关键文件：

- python/sglang/srt/hardware_backend/mlx/scheduler_mixin.py（模块 MLX 调度器；类别 source；类型 core-logic；符号 SchedulerMlxOverlapMixin._finalize_mlx_pending_job）：核心修复：在 _finalize_mlx_pending_job 中添加 forward_ct 递增和 profiler predicate

调用，使 step-bounded profiling 在 MLX 重叠循环中生效。

- python/sglang/benchmark/offline_throughput.py（模块 基准工具；类别 source；类型 core-logic）：为 stop_profile 添加幂等处理，防止调度器已自动停止后再次调用抛异常。
- test/registered/unit/hardware_backend/mlx/test_scheduler_mixin.py（模块 测试；类别 test；类型 test-coverage；符号 TestFinalizeMlxPendingJob, _make_scheduler, test_finalize_advances_forward_ct_and_runs_predicate, test_forward_ct_advances_once_per_step）：新增单元测试，验证 forward_ct 递增和 profiler predicate 调用，防止回归。

关键符号：SchedulerMlxOverlapMixin._finalize_mlx_pending_job

## 关键源码片段

### python/sglang/srt/hardware_backend/mlx/scheduler_mixin.py

核心修复：在 _finalize_mlx_pending_job 中添加 forward_ct 递增和 profiler predicate 调用，使 step-bounded profiling 在 MLX 重叠循环中生效。

```
def _finalize_mlx_pending_job(self: Scheduler, pending: MlxPendingJob):
    # 前进 forward_ct 并触发 profiler 批处理谓词；
    # 标准调度器在 run_batch 中做这些，但 MLX 重叠循环绕过 run_batch，
    # 因此在此处手动补齐，使得 step-bounded profiling 能自动停止。
    self.forward_ct += 1
    self.profiler_manager._profile_batch_predicate(pending.schedule_batch)

    # 原有 finalize 逻辑保持不变
    result = self.tp_worker.finalize_mlx_result(
        pending.prefills, pending.extends,
        pending.decode, pending.mode, pending.reqs,
    )
    if result.next_token_ids is not None:
        pending.batch_copy.input_ids = result.next_token_ids
        pending.schedule_batch.input_ids = result.next_token_ids
    self.last_batch = pending.schedule_batch
    self.process_batch_result(pending.batch_copy, result)
```

### python/sglang/benchmark/offline_throughput.py

为 stop_profile 添加幂等处理，防止调度器已自动停止后再次调用抛异常。

```
    if profile:
        dir = os.getenv("SGLANG_TORCH_PROFILER_DIR")
        if not profile_steps:
            known_files = set(os.listdir(dir))
        # 当 --profile-steps 指定步数时，调度器会在 N 步后自动停止；
        # 此时再调 stop_profile 会抛 "not in progress" RuntimeError。
        # 短于 N 步的 runs 未达目标，仍需显式停止。
        # 无论如何，stop 必须在 monitor_trace_file 之前完成。
        try:
            backend.stop_profile()
```

```
        except RuntimeError:
            pass
        monitor_trace_file(known_files, dir)
```

### test/registered/unit/hardware_backend/mlx/test_scheduler_mixin.py

新增单元测试，验证 forward_ct 递增和 profiler predicate 调用，防止回归。

```
    def test_finalize_advances_forward_ct_and_runs_predicate(self):
        from sglang.srt.hardware_backend.mlx.scheduler_mixin import (
            SchedulerMlxOverlapMixin,
        )
        scheduler = self._make_scheduler()
        pending = MagicMock()
        SchedulerMlxOverlapMixin._finalize_mlx_pending_job(scheduler, pending)
        # forward_ct 应从 0 变为 1
        self.assertEqual(scheduler.forward_ct, 1)
        # profiler 批处理谓词应被调用一次
        scheduler.profiler_manager._profile_batch_predicate.assert_called_once_with(
            pending.schedule_batch
        )
        # 原有 finalize 流程仍正常执行
        scheduler.process_batch_result.assert_called_once()

    def test_forward_ct_advances_once_per_step(self):
        # 连续调用三次，验证 forward_ct 每次递增 1，数量匹配
```

## 评论区精华

Reviewer jlee5814 指出：当 --profile-steps N 指定的步数超过实际生成步数时，monitor_trace_file 会无限循环等待，建议将 stop_profile 调用包裹为幂等。作者随后修改，添加 try/except RuntimeError。该提议被接纳并合并。

- stop_profile 幂等性需求 (correctness): 作者添加 try/except RuntimeError 实现幂等，被接受。

## 风险与影响

- 风险：变更仅影响 MLX 后端且 profiling 启用时，正常推理不受影响。新代码引入的整数递增和方法调用开销可忽略（O(1)）。风险点在于 self.profiler_manager 在 mixin 中是否确保初始化；单元测试通过 MagicMock 验证了调用，但在实际调度器中可能因缺失属性导致 AttributeError。不过 profiler_manager 在 Scheduler 基类中已有默认初始化，因此风险低。
- 影响：用户影响：Apple Silicon 上使用 MLX 后端的开发者，现在可以通过 --profile-steps 正常进行 step-bounded profiling，获得可用的 .gputrace 文件，且不再因 watchdog 超时而崩溃。 系统影响：无性能退化，因 forward_ct 递增和空 predicate 不在标准路径上。 团队影响：为后续 MLX 性能分析工作流铺平道路，减少调试 profiling 工具的时间。
- 风险标记：MLX 后端变更，步长分析修复，幂等设计

## 关联脉络

- PR #28122 [MLX] Add Metal profiling hooks to server profiler: 为本 PR 的前置工作，添加了 Metal profiling 后端钩子。本 PR 在其基础上修复 step-bounded 自动停止。
- PR #19137 [Roadmap] Apple Device Support (2026 Q2): PR #29217 属于该路线图 Profiling 部分的实现，目标是在 Apple Silicon 上完整支持 sglang。