PR #32447 完整报告

sgl-project/sglang

[MLX] Fix overlap-loop request bookkeeping and graceful shutdown

合并时间：2026-07-29 10:04

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32447

执行摘要

- 一句话：修复 MLX overlap 调度器首请求崩溃与优雅退出
- 推荐动作：该 PR 修复了 MLX 后端两个隐蔽但影响严重的问题，值得相关开发者精读，尤其是 _prepare_mlx_launch 的设计及其与 run_batch 的对齐方式，以及优雅退出的处理。Review 讨论中对 launch_ts 放置位置和 SWA 维护的权衡也值得参考。

功能与动机

引用 Issue #32445 描述：MLX overlap 调度循环绕过 Scheduler.run_batch，但 run_batch 会 stamp batch.launch_ts 并递增 forward_ct。process_batch_result -> _record_step_counters 无条件地使用 launch_ts，若为 None 则引发 TypeError，导致调度器崩溃。健康检查请求被排除在计数器外，因此 /health_generate 仍显示绿色，而真实请求全部失败。同时，event_loop_overlap_mlx 未检查 self.gracefully_exit，导致关闭信号无法使循环退出，最终超时后被强制杀进程，无法释放用户态资源。

实现拆解

1. 新增 _prepare_mlx_launch 方法：在文件 scheduler_mixin.py 的 SchedulerMlxOverlapMixin 中新增该方法，递增 forward_ct，为 batch 设置 forward_iter 和 launch_ts，并调用 _profile_batch_predicate。该方法在每次 MLX 发射前调用，等同于 run_batch 的启动会计。
2. 修改发射路径：在 _launch_fresh 和 _launch_chained 中，在调用 resolve_forward_inputs 或 async_chained_decode_mlx 之前调用 _prepare_mlx_launch，确保时间戳在异步图构建前记录。_launch_chained 中仅保留 forward_iter 更新（用于 SWA 维护），删除 launch_ts 重复赋值以与 run_batch 行为对齐。
3. 简化 _finalize_mlx_pending_job：移除原有的 forward_ct 递增和 profiler 调用（已迁移至发射前），使其仅专注于调用 finalize_mlx_result 并处理结果。这避免了重复计数。
4. 添加优雅退出支持：在 event_loop_overlap_mlx 的 while 循环顶部增加 if self.gracefully_exit: break，并在 break 前调用 mx.synchronize() 以确保所有已提交的异步 Metal 工作完成，避免在资源释放时出现异常。
5. 测试重构：将 test_scheduler_mixin.py 中的 TestFinalizeMlxPendingJob 类重写为 TestMlxLaunchBookkeeping，覆盖 _prepare_mlx_launch 的正确性（forward_ct 递增、forward_iter 和 launch_ts 类型、profiler 调用）。新增

TestOverlapLoopStampsLaunchTs 集成测试，使用真实 event_loop_overlap_mlx 循环验证新鲜发射和链式发射均能正确 stamp launch_ts。调整 test_attention_patching.py 中的测试，移除对 forward_ct 的严格断言（因计数已移至发射阶段），并添加 gracefully_exit 和 forward_ct 初始化。

关键文件：

- python/sglang/srt/hardware_backend/mlx/scheduler_mixin.py（模块 MLX 后端；类别 source；类型 core-logic；符号 _prepare_mlx_launch, _finalize_mlx_pending_job, event_loop_overlap_mlx）：核心源码文件，修复了 MLX overlap 调度器的记账和退出逻辑。新增 _prepare_mlx_launch 方法，重构 _finalize_mlx_pending_job，修改 _launch_fresh 和 _launch_chained，在 event_loop_overlap_mlx 中添加 graceful exit 检查和 mx.synchronize。
- test/registered/unit/hardware_backend/mlx/test_scheduler_mixin.py（模块 测试（MLX）；类别 test；类型 test-coverage；符号 TestMlxLaunchBookkeeping, TestOverlapLoopStampsLaunchTs, test_prepare_launch_advances_forward_ct_and_runs_predicate, test_forward_ct_advances_once_per_launch）：测试文件，几乎完全重写，提供对 _prepare_mlx_launch 的单元测试和集成测试，确保新记账行为正确，并验证优雅退出路径。
- test/registered/unit/hardware_backend/mlx/test_attention_patching.py（模块 测试（Attention）；类别 test；类型 test-coverage；符号 test_finalize_pending_job_updates_scheduler_last_batch, test_overlap_loop_materializes_prefill_input_ids, FakeOverlapScheduler）：辅助测试调整，删除已不再适用的 forward_ct 断言，添加 graceful_exit 和 forward_ct 初始化，以便与新的记账模型兼容。

关键符号：_prepare_mlx_launch, _finalize_mlx_pending_job, _launch_fresh, _launch_chained, event_loop_overlap_mlx, TestMlxLaunchBookkeeping._make_scheduler, TestMlxLaunchBookkeeping.test_prepare_launch_advances_forward_ct_and_runs_predicate, TestMlxLaunchBookkeeping.test_forward_ct_advances_once_per_launch, TestMlxLaunchBookkeeping.test_finalize_does_not_double_count_launch, TestOverlapLoopStampsLaunchTs.test_fresh_launch_stamps_launch_ts_before_input_materialization, TestOverlapLoopStampsLaunchTs.test_chained_launch_re_stamps_forward_iter_and_profiler

关键源码片段

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

核心源码文件，修复了 MLX overlap 调度器的记账和退出逻辑。新增 _prepare_mlx_launch 方法，重构 _finalize_mlx_pending_job，修改 _launch_fresh 和 _launch_chained，在 event_loop_overlap_mlx 中添加 graceful exit 检查和 mx.synchronize。

```
def _prepare_mlx_launch(self: Scheduler, batch: ScheduleBatch):
    """在 MLX 发射前 stamp 调度器记账字段。"""
    # 与 run_batch 的发射边界对齐。特别是 profiler 谓词
    # 必须在图构建 / mx.async_eval 之前运行；如果在 finalize
```

```
    # 阶段运行，会导致至少多 profile 一个已排队的 decode 步骤。
    self.forward_ct += 1
    batch.forward_iter = self.forward_ct
    batch.launch_ts = time.monotonic()
    self.profiler_manager._profile_batch_predicate(batch)

def _finalize_mlx_pending_job(self: Scheduler, pending: MlxPendingJob):
    # 移除了 forward_ct 递增和 profiler 调用，这些已移至发射前。
    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)

def event_loop_overlap_mlx(self: Scheduler):
    ...
    while True:
        # 优雅退出检查：必须排在任何 pause 逻辑之前
        if self.gracefully_exit:
            # synchronize 确保所有异步 Metal 工作已完成
            mx.synchronize()
            break
        if self._engine_paused:
            ...
        # 其余循环逻辑 ...
```

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

测试文件，几乎完全重写，提供对 _prepare_mlx_launch 的单元测试和集成测试，确保新记账行为正确，并验证优雅退出路径。

```
class TestMlxLaunchBookkeeping(unittest.TestCase):
    """run_batch 风格的记账检测：MLX overlap 循环必须正确 stamp 发射边界。"""

    def _make_scheduler(self):
        scheduler = MagicMock()
        scheduler.forward_ct = 0
        result = MagicMock()
        result.next_token_ids = None
        scheduler.tp_worker.finalize_mlx_result.return_value = result
        return scheduler

    def test_prepare_launch_advances_forward_ct_and_runs_predicate(self):
```

```
        from sglang.srt.hardware_backend.mlx.scheduler_mixin import (
            SchedulerMlxOverlapMixin,
        )
        scheduler = self._make_scheduler()
        batch = MagicMock()

        SchedulerMlxOverlapMixin._prepare_mlx_launch(scheduler, batch)

        # forward_ct 递增
        self.assertEqual(scheduler.forward_ct, 1)
        # batch 上 stamp forward_iter
        self.assertEqual(batch.forward_iter, 1)
        # launch_ts 是 float（时间戳）
        self.assertIsInstance(batch.launch_ts, float)
        # profiler 谓词已调用
        scheduler.profiler_manager._profile_batch_predicate.assert_called_once_with(batch)

    def test_forward_ct_advances_once_per_launch(self):
        # 连续调用三次 forward_ct 依次为 1,2,3
        ...

    def test_finalize_does_not_double_count_launch(self):
        # 先 launch 后 finalize，forward_ct 仍为 1
        ...
```

评论区精华

- launch_ts 放置顺序：reviewer yeahdongcn 指出 _launch_fresh 中 launch_ts 在 resolve_forward_inputs 之后赋值，而 run_batch 在之前，导致 total_prefill_busy_us 含义不一致。作者在后续 commit 中已调整顺序至输入物化之前。
- 链式发射中是否需要重新 stamp：LarrySimingDeng 提问链式发射中是否不需要更新 forward_iter/launch_ts，因为 process_batch_result 使用的是 batch_copy。作者解释保留 forward_iter 的目的是为了维持 SWA 维护的 cadence（maybe_evict_swa 依赖 forward_iter），因此保留 forward_iter 赋值，删除 launch_ts 赋值。
- 优雅退出时 drain 异步工作：LarrySimingDeng 提出退出时可能存在尚未 finalize 的 mx.async_eval 作业，建议添加 mx.synchronize()。作者采纳，在 graceful exit 路径中加入同步调用。
  - launch_ts 放置顺序及其对性能指标的影响 (correctness): 作者在后续 commit 中将 _prepare_mlx_launch 调用提前至 resolve_forward_inputs 之前，使顺序与 run_batch 一致。
  - 链式发射中 forward_iter/launch_ts 更新的必要性 (design): 作者同意保留 forward_iter 以维持 SWA 维护 cadence（maybe_evict_swa 依赖 forward_iter），删除 launch_ts 赋值以避免重复。
  - 优雅退出时未同步异步 Metal 工作 (correctness): 作者在 graceful exit 路径添加 mx.synchronize() 并注释说明理由。

风险与影响

- 风险：
  - 性能指标精度：调整 launch_ts 记录时间点后，total_prefill_busy_us 等指标的含义与标准调度器完全一致，消除了之前可能存在的偏差，但若用户依赖旧有的 MLX 特有指标，可能需要重新校准。
  - Graceful exit 延迟：添加 mx.synchronize() 可能增加 shutdown 等待时间，但这是释放 GPU 资源的必要步骤，且仅发生在退出时，不影响在线服务。
  - SWA 维护频次变化：之前因 forward_iter 为 None，SWA eviction 每步都触发（(None or 0) % eviction_interval == 0），修复后回到正常的每 N 步执行，可能影响长序列推理的内存行为。
  - 覆盖范围：变更仅作用于 MLX 后端，通过 unit test 验证，但缺少端到端 integration 测试覆盖 shutdown 场景。
- 影响：
  - 用户影响：MLX（Apple Silicon）后端用户将不再遭遇首请求崩溃，且能正常通过 /shutdown 或 ShutdownReq 停止服务，释放资源。之前需要手动 kill 进程。性能指标更准确。
  - 系统影响：不涉及其他后端或模块。调度器记账逻辑的变更严格限定在 SchedulerMlxOverlapMixin 类中，不改变基类行为。
  - 团队影响：维持了与标准调度器一致的设计模式，降低后续维护者理解成本。需要注意 MLX overlap 循环的特殊性已在文档中注明。
  - 风险标记：MLX-specific，核心路径变更，性能指标含义改变，Graceful exit 延迟增加，SWA 维护频次变化

关联脉络

- PR #32445 [MLX] Stamp launch_ts/forward_iter in the overlap loop; fix scheduler crash on first request: 该 PR 直接关闭的 Issue，详细描述了崩溃和动机。