# PR #25621 完整报告

sgl-project/sglang

Move pool-stats sampling to SchedulerPoolStatsObserver

合并时间：2026-05-18 18:37

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25621

## 执行摘要

- 一句话：将 pool-stats 采样从 Mixin 移至独立组件
- 推荐动作：值得精读，作为 SRT 调度器逐步重构的范例。可以学习如何将静态辅助方法安全地迁移为组件方法，并系统性更新调用点。虽改动机械，但 review 中发现的两个优化点（冗余调用、缺失类型导入）值得关注。建议审阅者重点关注 scheduler_runtime_checker_mixin.py 的剩余类型导入修复。

## 功能与动机

PR body 说明："Mechanical cut + paste for the introduce-pool-stats-observer mech move." 目的是将池统计采样方法从 SchedulerRuntimeCheckerMixin 抽离到专用的 SchedulerPoolStatsObserver 组件，实现关注点分离，为后续进一步重构（如删除 mixin）做准备。

## 实现拆解

1. 从 SchedulerRuntimeCheckerMixin 中删除 pool-stats 方法：删除了 streaming_session_count、active_pool_idxs、session_held_tokens、session_held_full_tokens、session_held_swa_tokens、session_held_req_count、session_held_mamba_slots、get_pool_stats 及其辅助方法（_get_token_info、_get_hisparse_token_info、_get_mamba_token_info、_get_swa_token_info），同时移除了不再需要的 dataclasses 和 PoolStats 类型的 import（后者有潜在问题，详见评论区精华）。
2. 添加到 SchedulerPoolStatsObserver：在 python/sglang/srt/managers/scheduler_components/pool_stats_observer.py 中，将这些方法以实例方法形式添加到 @dataclass 装饰的 SchedulerPoolStatsObserver 类中，去掉 @staticmethod 装饰器，将 self 类型注解从显式 "SchedulerPoolStatsObserver" 简化为隐式 self，内部对同一类方法的调用从 SchedulerRuntimeCheckerMixin.method(self) 改为 self.method()。
3. 更新调用点：在 scheduler.py（get_new_batch_prefill、on_idle）、scheduler_runtime_checker_mixin.py 的剩余方法（_check_full_pool、_check_swa_pool 等中的调用）以及 observability/scheduler_metrics_mixin.py（report_prefill_stats、report_decode_stats、get_loads 等）中，将形如 self.get_pool_stats(self.pool_stats_observer, ...) 的调用全部改为 self.pool_stats_observer.get_pool_stats(...)。

4. 清理 import：在 scheduler_runtime_checker_mixin.py 中，移除了对 SchedulerPoolStatsObserver 和 PoolStats 的导入（但 reviewer 指出 PoolStats 仍被剩余类型提示使用，需要补回）。

关键文件：

- python/sglang/srt/managers/scheduler_runtime_checker_mixin.py（模块 调度器；类别 source；类型 core-logic；符号 streaming_session_count, active_pool_idxs, session_held_tokens, session_held_full_tokens）：从该文件删除了 8 个 pool-stats 方法和相关导入，剩余方法继续使用这些统计但改为委托调用。是本次重构的主要输出端。
- python/sglang/srt/managers/scheduler_components/pool_stats_observer.py（模块 池统计；类别 source；类型 core-logic；符号 streaming_session_count, active_pool_idxs, session_held_tokens, session_held_full_tokens）：新方法被添加到此文件的 SchedulerPoolStatsObserver 类，成为池统计采样的新家。是本次重构的核心接收端。
- python/sglang/srt/observability/scheduler_metrics_mixin.py（模块 观测；类别 source；类型 core-logic）：多处调用点被修改，从使用旧委托方式切换到新方式，体现了重构对外部使用者的影响。
- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic）：核心调度器中的两个调用点更新为委托给 pool_stats_observer。

关键符号：streaming_session_count, active_pool_idxs, session_held_tokens, session_held_full_tokens, session_held_swa_tokens, session_held_req_count, session_held_mamba_slots, get_pool_stats, _get_token_info, _get_hisparse_token_info, _get_mamba_token_info, _get_swa_token_info

## 关键源码片段

### python/sglang/srt/managers/scheduler_components/pool_stats_observer.py

新方法被添加到此文件的 SchedulerPoolStatsObserver 类，成为池统计采样的新家。是本次重构的核心接收端。

```
    def streaming_session_count(self) -> int:
        """返回当前处于 streaming 状态的 session 数量。"""
        return sum(
            1
            for session in self.session_controller.sessions.values()
            if session.streaming
        )

    def active_pool_idxs(self) -> set:
        """Pool idxs currently owned by reqs in last_batch / running_batch.

        Used to decide which session slots' KV is owned by batch reqs
        (and thus counted via uncached_size, not session_held).
        """
        idxs = set()
```

```
        for batch in [self.get_last_batch(), self.get_running_batch()]:
            if batch is None or batch.is_empty():
                continue
            for req in batch.reqs:
                if req.req_pool_idx is not None:
                    idxs.add(req.req_pool_idx)
        return idxs

    def session_held_tokens(self) -> int:
        return self.tree_cache.session_held_tokens(self.active_pool_idxs())

    # ... 其他 session_held_* 方法类似 ...

    def get_pool_stats(self) -> PoolStats:
        """汇总所有 token 池的统计信息。"""
        if self.is_hybrid_swa:
            pool_stats = self._get_swa_token_info()
        elif self.is_hybrid_ssm:
            pool_stats = self._get_mamba_token_info()
        else:
            pool_stats = self._get_token_info()

        if self.enable_hisparse:
            pool_stats = self._get_hisparse_token_info(pool_stats)

        # swa + ssm can coexist: overlay mamba fields onto swa stats
        if self.is_hybrid_ssm:
            mamba_stats = self._get_mamba_token_info()
            pool_stats.is_hybrid_ssm = True
            pool_stats.mamba_num_used = mamba_stats.mamba_num_used
            pool_stats.mamba_usage = mamba_stats.mamba_usage
            pool_stats.mamba_available_size = mamba_stats.mamba_available_size
            pool_stats.mamba_evictable_size = mamba_stats.mamba_evictable_size

        return pool_stats
```

## 评论区精华

- 缺失 PoolStats 导入：reviewer (gemini-code-assist) 指出，scheduler_runtime_checker_mixin.py 在删除对 PoolStats 的导入后，仍在其方法（如 _check_full_pool）中使用了 PoolStats 类型，这会导致静态分析错误，建议在 TYPE_CHECKING 块中恢复导入。
- 冗余调用优化：在 get_pool_stats 方法中，当 is_hybrid_ssm 为 True 且 is_hybrid_swa 为 False 时，_get_mamba_token_info 被调用了两次（一次构造 pool_stats，一次叠加 mamba 字段），该分支可以优化为仅在同时启用两种模式时执行叠加逻辑。
- 缺少返回类型注解：_get_mamba_token_info 方法缺少返回类型 PoolStats 的注解。

- 缺失 PoolStats 导入 (correctness): 需要添加 from ... pool_stats_observer import PoolStats 到 TYPE_CHECKING 块中，以修复静态分析。
- 冗余调用 _get_mamba_token_info (performance): 应优化为仅在同时启用两种模式时执行叠加，避免重复计算。
- 缺少返回类型注解 (style): 添加返回类型注解以符合项目惯例。

# 风险与影响

- 风险：
  1. 导入缺失：如上所述，scheduler_runtime_checker_mixin.py 中 PoolStats 类型导入被移除，可能导致类型检查失败或潜在运行错误（若该类型在运行时被使用）。目前该文件剩余方法仅在类型提示中使用 PoolStats，且被 TYPE_CHECKING 条件包裹，不会在运行时出错，但仍需修复以保持静态分析健康。
  2. 冗余调用：在 PoolStatsObserver.get_pool_stats() 中，当 is_hybrid_ssm 且非 is_hybrid_swa 时重复计算 token stats，虽不影响正确性，会带来可忽略的性能开销（实际可能微不足道）。
  3. 测试覆盖：本次重构未包含测试变更，但方法体是字节等价的，风险较低。但仍应确保回归测试覆盖池统计逻辑。
- 影响：
  1. 对用户：无行为变化，所有统计输出和调度决策与重构前一致。
  2. 对系统：池统计逻辑更集中，便于后续维护和扩展。SchedulerRuntimeCheckerMixin 的删除为后续该 mixin 退休铺平道路。
  3. 对团队：需注意在合并后续 PR 时，若同时修改这些方法可能产生冲突。建议 CI 覆盖池统计的 E2E 测试。 - 风险标记：导入缺失，重复调用，缺少类型注解

# 关联脉络

- PR #25622 Move create_scheduler_watchdog from runtime_checker mixin to scheduler.py: 同一重构系列，将 SchedulerRuntimeCheckerMixin 中的其他功能逐步迁出。
- PR #25623 Introduce SchedulerInvariantChecker to own invariant-check state: 接续引入不变量检查组件，继续分解 mixin。
- PR #25624 Move invariant checks to SchedulerInvariantChecker and retire runtime_checker mixin: 退休 runtime_checker mixin 的后续 PR。