PR #25442 完整报告

sgl-project/sglang

Lift forward_ct/cur_batch and use direct access in watchdog

合并时间：2026-05-16 09:22

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25442

执行摘要

- 一句话：提升 watchdog 属性初始化顺序，移除防御性 getattr
- 推荐动作：值得快速合并。这是一个典型的机械重构，提升了代码可读性并消除不必要的防御性模式，可作为后续类似清理的参考。

功能与动机

watchdog 守护线程在首次 tick 时读取 forward_ct 和 cur_batch，如果 __init__中尚未设置这些属性，则必须回退到 getattr 防御式访问。PR 将属性提升到 init_soft_watchdog 之前，使属性存在性成为 __init__的不变式，而不是分散在多个文件中的防御性约定。

实现拆解

1. 在 scheduler.py 中提前初始化属性：在 Scheduler.__init__ 中，将 self.forward_ct: int = 0 和 self.cur_batch: Optional[ScheduleBatch] = None 的赋值移动到 self.is_initializing = True 之后、self.init_soft_watchdog(server_args) 之前。
2. 在 scheduler_runtime_checker_mixin.py 中移除 getattr：在 create_scheduler_watchdog 函数中，将 getattr(scheduler, "forward_ct", 0) 改为 scheduler.forward_ct，将 getattr(scheduler, "cur_batch", None) is not None 改为 scheduler.cur_batch is not None。
3. 保持功能不变：不引入行为变更，仅重构代码以消除防御性模式。

关键文件：

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 Scheduler.init）：将 forward_ct 和 cur_batch 初始化提前到 init_soft_watchdog 调用之前，确保 watchdog 线程首次读取时属性已定义。
- python/sglang/srt/managers/scheduler_runtime_checker_mixin.py（模块 调度器；类别 source；类型 core-logic；符号 create_scheduler_watchdog）：移除 watchdog 创建函数中针对 forward_ct 和 cur_batch 的 getattr 防御性访问，改为直接属性访问。

关键符号：Scheduler.init, create_scheduler_watchdog

关键源码片段

python/sglang/srt/managers/scheduler.py

将 forward_ct 和 cur_batch 初始化提前到 init_soft_watchdog 调用之前，确保 watchdog 线程首次读取时属性已定义。

```
# python/sglang/srt/managers/scheduler.py - Scheduler.__init__ (partial)
# 变更前：forward_ct 和 cur_batch 在 init_soft_watchdog 之后才被初始化（如果存在）
# 变更后：在 init_soft_watchdog 之前就明确初始化为默认值

def __init__(self, ...):
    self.is_initializing = True
    # init_soft_watchdog 启动一个守护线程，首次 tick 时会读取这些属性
    self.forward_ct: int = 0
    self.cur_batch: Optional[ScheduleBatch] = None
    self.init_soft_watchdog(server_args)
    # ... 其余初始化
```

python/sglang/srt/managers/scheduler_runtime_checker_mixin.py

移除 watchdog 创建函数中针对 forward_ct 和 cur_batch 的 getattr 防御性访问，改为直接属性访问。

```
# python/sglang/srt/managers/scheduler_runtime_checker_mixin.py - create_scheduler_
watchdog (partial)
# 变更后：直接访问 scheduler.forward_ct 和 scheduler.cur_batch，
# 因为现在这些属性保证在调度器构造完成时已初始化。

def create_scheduler_watchdog(
    scheduler: Scheduler, watchdog_timeout: float, soft: bool = False
) -> WatchdogRaw:
    def dump_info() -> str:
        if scheduler.is_initializing:
            return ""
        _, messages = scheduler._check_all_pools(scheduler.get_pool_stats())
        return (
            f"{scheduler.cur_batch.batch_size()=}\n"
            f"{scheduler.cur_batch.reqs=}\n" + "\n".join(messages)
        )

    return WatchdogRaw(
        debug_name="Scheduler",
        get_counter=lambda: scheduler.forward_ct, # 原为 getattr(scheduler, "forward_ct", 0)
        is_active=lambda: scheduler.is_initializing or scheduler.cur_batch is not None, # 原为
        getattr(scheduler, "cur_batch", None) is not None
        watchdog_timeout=watchdog_timeout,
        soft=soft,
        dump_info=dump_info,
    )
```

评论区精华

无 review 评论或讨论。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。变更仅涉及属性初始化顺序调整和 getattr 替换，不改变任何运行时行为。但需确保 watchdog 线程在读取属性时调度器尚未完成初始化（例如在并发构造中）——当前架构下 init_soft_watchdog 立即启动守护线程，但属性在调用前已赋值，因此安全。
- 影响：影响范围仅限于 Scheduler 初始化流程和 watchdog 创建函数。无外部行为变化，不涉及测试、配置或 API 变更。
- 风险标记：暂无

关联脉络

- PR #25430 Convert local-only self.X attributes to locals: 同属属性初始化清理系列，移除未使用的 self 属性。
- PR #25447 Replace defensive getattr in pool_configurator with direct access: 类似地移除 getattr 防御性访问，同一重构主题。