PR #27238 完整报告

sgl-project/sglang

Add quiet mode for busy mem check (level 1: buffer + dump on leak)

合并时间：2026-06-04 16:31

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27238

执行摘要

- 一句话：为繁忙时内存检查添加安静模式，减少日志噪声
- 推荐动作：该 PR 设计简洁，改动集中，风险低，值得合并。建议在文档中备注级别 1 的缓冲区容量说明，以便用户了解极限场景下的日志覆盖能力。

功能与动机

繁忙时内存检查在每次事件循环迭代中都运行，之前当级别 > 1（含 2）时，每轮迭代都会记录 [Mem Check (BUSY)] 日志，在高负载场景下产生大量噪声，影响可观测性和排查效率。PR 希望在不牺牲泄漏检测能力的前提下，通过缓存日志的方式减少稳态日志量，仅在泄漏发生时输出完整上下文。

实现拆解

1. 新增环形缓冲区和配置常量：在文件顶部定义 BUSY_MEM_CHECK_LOG_RING_SIZE = 1000，在 SchedulerInvariantChecker 数据类中新增字段 recent_busy_msgs: Deque[str]，使用 deque(maxlen=1000) 初始化，自动维护最近 1000 条日志。
2. 重构 self_check_during_busy 的日志逻辑：原有逻辑仅在 level > 1 时逐条 logger.info。现改为先读取 level 值，构造完整的日志行 full_line 和 swa_line（如果存在），然后根据 level 分支处理：
   - level > 1（非安静模式）：直接 logger.info 输出每轮消息，保持原行为。
   - level == 1（安静模式）：将构造好的消息行追加到 self.recent_busy_msgs 中，不输出；但若检测到泄漏（full_leak 或 swa_leak 为真），则循环输出缓冲区中所有消息，用于泄漏时的上下文诊断。
3. 保留断言机制不变：无论哪种级别，泄漏时仍会触发 assert not full_leak 和 assert not swa_leak，确保泄漏被及时捕获。

关键文件：

- python/sglang/srt/managers/scheduler_components/invariant_checker.py（模块 调度器；类别 source；类型 core-logic；符号 BUSY_MEM_CHECK_LOG_RING_SIZE, SchedulerInvariantChecker.recent_busy_msgs, SchedulerInvariantChecker.self_check_during_busy）：唯一的变更文件，实现了所有逻辑：新增环形缓冲区字段、修改 self_check_during_busy 方法以支持三种日志级别，并在泄漏时 dump 历史消息后仍保持断言。

关键符号：SchedulerInvariantChecker.self_check_during_busy

关键源码片段

python/sglang/srt/managers/scheduler_components/invariant_checker.py

唯一的变更文件，实现了所有逻辑：新增环形缓冲区字段、修改 self_check_during_busy 方法以支持三种日志级别，并在泄漏时 dump 历史消息后仍保持断言。

```
# 文件顶部新增环形缓冲区大小常量 BUSY_MEM_CHECK_LOG_RING_SIZE = 1000 @dataclass(kw_only=True, slots=True) classSchedulerInvariantChecker:     # … 其他字段不变 …     recent_busy_msgs: Deque[str] = field(         default_factory=lambda: deque(maxlen=BUSY_MEM_CHECK_LOG_RING_SIZE)     ) defself_check_during_busy(self):         # … 前置检查代码不变 …         # 读取当前级别     level = envs.SGLANG_ENABLE_STRICT_MEM_CHECK_DURING_BUSY.get()         # 构造日志行（基于池状态和 uncached 计算，略）         full_line = f"[Mem Check (BUSY)]{full_msg}"         swa_line = f"[Mem Check (BUSY)]{swa_msg}" if swa_msg else None         if level > 1:             # Verbose: 每轮直接输出，原行为             logger.info(full_line)             if swa_line:                 logger.info(swa_line)         elif level == 1:             # Quiet: 缓冲日志，仅泄漏时 dump             self.recent_busy_msgs.append(full_line)             if swa_line:                 self.recent_busy_msgs.append(swa_line)             if full_leak or swa_leak:                 # 泄漏时打印缓冲区内所有历史消息，用于定位趋势                 for msg in self.recent_busy_msgs:                     logger.info(msg)         # 断言机制保持与原来一致，不因级别改变而跳过         assert not full_leak, f"Full Pool Mem Leak Detected!{full_msg}"         assert not swa_leak, f"SWA Pool Mem Leak Detected!{swa_msg}" （注：因篇幅省略前置检查和 pool check 方法，其逻辑未变更。）
```

评论区精华

PR 没有产生 review 评论，所有决策由作者 hnyls2002 通过多个提交自行迭代完成：第一版实现基本 buffer 逻辑，第二版明确分化为 level 1 和 level 2，后续提交清理注释和合并主分支。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 环形缓冲区内存占用：1000 条字符串消息的内存占用在数百 KB 级别，属于可接受范围；但若消息较长或未来频繁改动格式，需关注内存增长。
  2. 日志丢失风险：安静模式下，当泄漏持续发生时，缓冲区可能被后续消息覆盖旧消息，导致泄漏前早期趋势丢失。当前 1000 条容量对于典型调度循环场景已足够，但仍需监控是否够用。
  3. 行为一致性：级别 1 在无泄漏时完全不输出日志，可能被运维人员误以为检查未运行。需要文档或注释明确说明。 - 影响：影响范围仅限 SchedulerInvariantChecker 类的自检方法，该模块本身是 debug/assert 用途，不构成关键业务路径。用户需要显式设置环

境变量 SGLANG_ENABLE_STRICT_MEM_CHECK_DURING_BUSY=1 才会启用安静模式。对于生产环境，启用级别 1 可以在保持泄漏检测的同时显著减少日志泛洪，提升可观测性。 - 风险标记：缺少测试覆盖

关联脉络

- PR #26676 [mem_cache][2/N] refactor: move SWATokenToKVPoolAllocator to allocator/swa.py: 涉及同一模块 SchedulerInvariantChecker 和内存池相关组件，属于 mem_cache 重构系列的一部分。
- PR #25000 Reduce mamba prefill allocation overhead: 涉及调度器模块的内存检查优化，与本次减少日志开销方向一致，都属于调度器性能 / 可观测性改进。