PR #36583 完整报告

sgl-project/sglang

Fix KV cache pool sized far too small when weight-loading memory is still referenced

合并时间：2026-08-29 00:47

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36583

执行摘要

- 一句话：修复 KV 缓存池因未回收引用内存而过度缩小
- 推荐动作：该 PR 值得精读，它揭示了显存测量中的引用计数陷阱。建议后续添加针对该场景的回归测试，确保 gc.collect() 调用在关键路径上保留。

功能与动机

PR body 指出：KV 池预算在权重加载后立即测量，而 get_available_gpu_memory() 调用 empty_cache() 只能回收未被引用的块，加载器临时对象仍被引用，导致预算被低估数数量级。实测 4×H200 上，修复前 3000 词提示被拒绝报错 “Input length (3034 tokens) exceeds the maximum allowed length (1735 tokens)”，修复后正常。

实现拆解

1. 在 _profile_available_bytes() 方法中，在调用 get_available_gpu_memory() 之前添加 gc.collect() 调用（对 python/sglang/srt/mem_cache/kv_cache_configurator.py 文件）。
2. 添加解释性注释，说明 empty_cache() 无法回收被引用块，以及收集先行的必要性。
3. 在文件顶部导入 gc 模块（+1 行）。
4. 无测试、配置或部署改动。

关键文件：

- python/sglang/srt/mem_cache/kv_cache_configurator.py（模块 缓存池；类别 source；类型 core-logic；符号 _profile_available_bytes）：核心修复文件，在 KV 缓存池预算测量前插入 gc.collect()，是唯一变更文件。

关键符号：_profile_available_bytes

关键源码片段

python/sglang/srt/mem_cache/kv_cache_configurator.py

核心修复文件，在 KV 缓存池预算测量前插入 gc.collect()，是唯一变更文件。

```
# python/sglang/srt/mem_cache/kv_cache_configurator.py

def _profile_available_bytes(self, pre_model_load_memory: int) -> int:
    # KV pool budget = currently-free GPU memory minus the non-static runtime
```

```
    # slack (pre_model_load_memory * (1 - mem_fraction_static)). Whatever is
    # already resident (model weights, etc.) is thus charged against it.
    # Weight-loading temporaries can still be referenced at this point, and
    # empty_cache() (which get_available_gpu_memory already calls) cannot
    # reclaim referenced blocks. Without collecting first, the KV budget is
    # measured against an understated free-memory figure and the pool can be
    # sized orders of magnitude too small while GPU memory sits idle.
    gc.collect() # 关键修复：先回收引用，避免预算低估
    available_gpu_memory = get_available_gpu_memory(
        self.device,
        self.gpu_id,
        distributed=get_world_group().world_size > 1,
        cpu_group=get_world_group().cpu_group,
    )
    # 后续计算 slack 和 rest_memory 的逻辑不变
    slack_gb = pre_model_load_memory * (1 - get_schedule().mem_fraction_static)
    # ...
```

评论区精华

仅有一条批准评论“LGTM”，无实质性讨论。

- 暂无高价值评论线程

风险与影响

- 风险：gc.collect() 是轻量操作，在模型加载阶段调用一次，性能影响可忽略。但该修复基于“权重加载临时对象仍被引用”的假设，若加载流程变化可能导致回收不完全，仍存在预算偏低风险。此外，未添加回归测试，后续若未来改动 get_available_gpu_memory 或加载流程，可能重新引入问题。
- 影响：影响所有使用 KV 缓存池的推理部署，特别是长上下文场景。修复后，在相同显存配置下，可处理的 token 数大幅提升，避免了因池过小而拒绝合理请求的问题。团队无需额外操作，但建议关注长上下文场景的显存占用变化。
- 风险标记：缺少测试覆盖，依赖引用回收假设

关联脉络

- PR #35931 [HiCache] Reject load-back specs that claim nodes pinned by an in-flight load-back: 同为 KV 缓存与调度相关修复，涉及显存管理，可参考其测试策略。
- PR #36425 HiCache: avoid unnecessary all-reduce in check_prefetch_progress: 同为 KV 缓存性能优化，体现该模块持续演进。
- PR #36227 [HiCache] Retry L3 storage prefetch after a missed attempt: 同为 KV 缓存与调度改进，涉及内存管理。