PR #27391 完整报告

sgl-project/sglang

[UnifiedTree]: Fix SWA admission budget under-counts HiCache load-back consumption

合并时间：2026-06-07 10:47

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27391

执行摘要

- 一句话：修复 SWA 准入预算少算 HiCache load-back 消耗
- 推荐动作：建议精读此 PR，尤其是 _swa_budget_for_req 的预算逻辑和 MatchResult 拆分的设计。layering violation 可作为后续重构的引导。

功能与动机

在混合 SWA + HiCache 场景中，调度器可能在 prefill 期间 OOM，尽管准入预算刚刚通过。故障路径：ScheduleBatch.add_one_req 检查 rem_swa_tokens 与 _swa_budget_for_req(...)（在树锁内外都检查）。通过后，init_load_back -> HybridCacheController.load -> _resolve_pool_transfers_allocation 同步地从 SWA 设备池为每个匹配窗口中仅主机的 SWA tombstone 分配 SWA page。刚加载的 SWA page 立即 inc_lock_ref，因此它们离开了 swa_available_size 但未出现在 swa_evictable_size 中。prepare_for_extend -> alloc_paged_token_slots_extend 然后尝试分配实际的 prefill page 并返回 None，引发 'Prefill out of memory. Try to allocate N tokens.'

实现拆解

1. MatchResult 扩展：在 base_prefix_cache.py 中为 MatchResult 增加 swa_host_hit_length 和 mamba_host_hit_length 字段，并在 zero_match_result 中重置它们。
2. Req 属性扩展：在 schedule_batch.py 的 Req 类中增加 swa_host_hit_length 和 mamba_host_hit_length 属性，并新增 needs_host_load_back 方法聚合判断三类 host hit。
3. 匹配结果传播：在 schedule_policy.py 的 match_prefix_for_req 和 schedule_batch.py 的 init_next_round_input 中，从 MatchResult 解包新字段到 req 对应属性。
4. SWA 组件填充：在 swa_component.py 的 finalize_match_result 中，遍历 best_match_node 的父节点，累计 SWA 主机 tombstone 长度，并写入 result.swa_host_hit_length。
5. 预算修复：在 schedule_policy.py 的 _swa_budget_for_req 中添加可选参数 swa_host_hit_length，预算增加 ceil_paged_tokens(swa_host_hit_length)。在 add_one_req 的两处准入检查（树锁外和树锁内）传入 req.swa_host_hit_length；其他调用点（_update_prefill_budget、回缩路径）保持默认 0，避免重复扣除。

6. load_back 触发条件扩展：在 unified_radix_cache.py 的 init_load_back 中，将触发条件从 host_hit_length > 0 扩展为也检查 swa_host_hit_length > 0 或 mamba_host_hit_length > 0。
7. Mamba 相关清理：调整 mamba_component.py 和 hi_mamba_radix_cache.py 以正确设置 host_hit_length 相关字段。
8. 测试更新：更新 test_unified_radix_cache_unittest.py，新增 _apply_match_to_req 辅助方法将整个 MatchResult 应用到 Req；修改相关测试用例验证拆分字段；在 test_prefill_adder.py 中增加一行属性以便测试通过。

关键文件：

- python/sglang/srt/managers/schedule_policy.py（模块 调度器；类别 source；类型 core-logic；符号 _swa_budget_for_req）：核心预算修复：_swa_budget_for_req 增加 swa_host_hit_length 参数，并在准入检查中传入该值。
- python/sglang/srt/managers/schedule_batch.py（模块 请求状态；类别 source；类型 core-logic；符号 needs_host_load_back）：新增 swa_host_hit_length 和 mamba_host_hit_length 属性和 needs_host_load_back 方法，支撑预算与 load_back 逻辑。
- test/registered/unit/mem_cache/test_unified_radix_cache_unittest.py（模块 单元测试；类别 test；类型 test-coverage；符号 _apply_match_to_req）：测试覆盖新字段，新增 _apply_match_to_req 辅助方法，修改测试用例验证拆分字段。
- python/sglang/srt/mem_cache/base_prefix_cache.py（模块 缓存层；类别 source；类型 core-logic；符号 MatchResult, zero_match_result）：MatchResult 新增 swa_host_hit_length 和 mamba_host_hit_length 字段，zero_match_result 重置它们。
- python/sglang/srt/mem_cache/unified_cache_components/swa_component.py（模块 SWA 组件；类别 source；类型 core-logic；符号 finalize_match_result）：finalize_match_result 中累计 SWA 主机 tombstone 并设置 swa_host_hit_length。
- python/sglang/srt/mem_cache/unified_radix_cache.py（模块 缓存层；类别 source；类型 core-logic；符号 init_load_back）：init_load_back 触发条件扩展到包含 SWA 和 Mamba 主机命中。

关键符号：_swa_budget_for_req, needs_host_load_back, _apply_match_to_req, finalize_match_result

关键源码片段

python/sglang/srt/managers/schedule_policy.py

核心预算修复：_swa_budget_for_req 增加 swa_host_hit_length 参数，并在准入检查中传入该值。

```
def _swa_budget_for_req(
    self, extend_input_len: int, swa_host_hit_length: int = 0
) -> int:
    # SWA pool budget per request. Only valid when is_hybrid_swa is True.
    #
```

```
        # With chunked prefill + overlap scheduler, the peak SWA occupancy is:
        # chunk N (running, not yet in tree) + sliding window (locked in tree)
        # + chunk N+1 (new allocation)
        # Since chunk N and locked tokens are already excluded from
        # swa_available + swa_evictable, the budget only needs to cover the
        # chunk N+1 allocation. We floor at sliding_window_size to reserve
        # room for the decode phase.
        if self.rem_chunk_tokens is not None:
            alloc = min(extend_input_len, self.rem_chunk_tokens)
        else:
            alloc = extend_input_len
        # 基础预算：本 chunk 分配与滑动窗口保留取最大，加一页安全量
        budget = max(alloc, self.tree_cache.sliding_window_size) + self.page_size
        # 如果本次 prefill 需要从主机加载 SWA 页面，额外补充对齐后的页面预算
        if swa_host_hit_length > 0:
            budget += self.ceil_paged_tokens(swa_host_hit_length)
        return budget

    # 在 add_one_req 方法（树锁外准入检查）中调用：
    # swa_needed = self._swa_budget_for_req(
    # req.extend_input_len, swa_host_hit_length=req.swa_host_hit_length
    # )
    # if swa_needed >= self.rem_swa_tokens:
    # return AddReqResult.NO_TOKEN
```

python/sglang/srt/managers/schedule_batch.py

新增 swa_host_hit_length 和 mamba_host_hit_length 属性和 needs_host_load_back 方法，支撑预算与 load_back 逻辑。

```
    # 在 Req.__init__ 中，约 line 818-821
    # 将原有的单字段 host_hit_length 拆分为三个 per-component 字段
    self.host_hit_length = 0 # Full-KV 主机命中长度
    self.swa_host_hit_length = 0 # SWA 窗口内主机命中长度
    self.mamba_host_hit_length = 0 # Mamba 状态主机命中长度

    # 在 Req 类中新增的方法
    def needs_host_load_back(self) -> bool:
        # 判断请求是否需要执行 L2 主机到设备的 load_back 操作
        return (
            self.host_hit_length > 0
            or self.swa_host_hit_length > 0
            or self.mamba_host_hit_length > 0
        )
```

评论区精华

- 设计分层违规讨论：gemini-code-assist[bot] 指出在 unified_radix_cache.py 的 init_load_back 中直接检查 req.swa_host_hit_length 是分层违规，建议通过 InitLoadBackParams 传递。当前 PR 未修改，作为已知设计权衡合并。

- 预算正确性确认：ispobock 确认仅在第一个 chunk 的预算中加上 load-back 消耗是正确的，与作者意图一致。
  - Layering violation in init_load_back: checking req directly (design): 当前 PR 未修改，合并后可能遗留该设计问题；但现有调用路径均传入 req，风险可控。
  - Budget addition only for first chunk confirmed correct (correctness): 逻辑正确，已合并。

风险与影响

- 风险：
  - 分层违规隐患：unified_radix_cache.py 直接引用 req 对象，若未来其他调用路径传入 req=None 则 SWA/Mamba load_back 不会触发。当前所有调用点均传入 req，风险较低。
  - 预算 double-counting 风险：若 _swa_budget_for_req 错误地在非首次 chunk 也传入 swa_host_hit_length，会导致预算偏高。作者通过只在准入 gate 传入、其他情形默认 0 来避免，逻辑正确。
  - 回归风险：拆分 host_hit_length 可能影响其他依赖原字段的代码（如 retraction、disagg），但单元测试覆盖了主要路径。
- 影响：
  - 用户：修复 OOM 问题，提高 SWA+HiCache 场景下的系统稳定性。
  - 系统：调度预算计算更精确，避免因 load_back 导致的临时资源不足。
  - 团队：需注意新增字段的命名和历史区别，未来可能通过 InitLoadBackParams 进一步治理分层问题。
  - 风险标记：分层违规隐患，预算重复计算风险，回归风险

关联脉络

- PR #27285 [HiCache] Fix crash when using PP + HiCache L2: 同为 HiCache 缓存系统修复，涉及相似的 cache controller 和 radix cache 模块，可能需统一理解。