PR #27291 完整报告

sgl-project/sglang

[HiCache] Fix SWA L3 cache miss due to a prefetch/hit len mismatch

合并时间：2026-06-18 14:55

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27291

执行摘要

- 一句话：修复 SWA 模型 L3 缓存预取与匹配长度不一致问题
- 推荐动作：建议阅读本 PR 以理解 HiCache 预取与 SWA correction 的交互，以及 _compute_max_prefix_len 在缓存一致性中的关键作用。相关测试配置方式也可作为后续架构适配的参考。

功能与动机

在 HiCache L3 缓存与 SWA 模型结合的场景下，预取与匹配阶段 token 长度不一致，导致即使相同 prompt 重复请求也无法命中缓存。PR body 详细描述了该现象，并指出当前测试无法通过。

实现拆解

1. 问题定位：在 sglang/srt/managers/scheduler.py 的 _prefetch_kvcache 方法中，new_input_tokens = req.full_untruncated_fill_ids[matched_len:] 使用全长 token，未考虑 SWA 的 correction。
2. 核心修复：引入 match_end = req._compute_max_prefix_len(len(req.full_untruncated_fill_ids))，并将切片终点从末尾改为 match_end，确保预取长度与后续 match 逻辑一致。
3. 测试框架增强：在 AccuracyTwoPassMixin（位于 test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_nightly.py）中添加 test_l3_prefetch_full_prefix_hit_after_flush 方法，使用随机 token 序列，经两次 generate 和中间 flush_cache，断言 cached_tokens >= input_len - max_uncached。
4. 架构差异适配：引入类属性 l3_prefetch_page_size、l3_prefetch_prompt_pages 和 l3_prefetch_max_uncached_tokens，允许不同模型架构自定义预期缓存边界（例如 Mamba 的状态只存储在 chunk 边界，因此 max_uncached 需设为 chunked_prefill_size）。
5. 测试用例配置：更新 test_unified_radix_cache_kl_mamba.py 和 test_unified_radix_cache_kl_dsv4.py，为相关测试类设置适当的属性值以适配新测试。

关键文件：

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 _prefetch_kvcache）：核心修复文件，修正预取 token 长度与 match 时长度不一致的问题，是解决 cache miss 的关键变更。

- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_nightly.py（模块 缓存测试；类别 test；类型 test-coverage；符号 test_l3_prefetch_full_prefix_hit_after_flush, AccuracyTwoPassMixin）：新增了验证 L3 缓存完全前缀命中率的核心测试方法，并引入架构适配属性。
- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_mamba.py（模块 Mamba 测试；类别 test；类型 test-coverage；符号 l3_prefetch_page_size, l3_prefetch_prompt_pages, l3_prefetch_max_uncached_tokens）：为 Mamba 混合模型适配缓存命中测试参数，反映其状态缓存边界。
- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_dsv4.py（模块 DSv4 测试；类别 test；类型 test-coverage；符号 l3_prefetch_page_size, l3_prefetch_prompt_pages）：为 DeepSeek V4 Flash 及其 EAGLE 变体配置缓存命中测试参数。

关键符号：_prefetch_kvcache, test_l3_prefetch_full_prefix_hit_after_flush, _compute_max_prefix_len

关键源码片段

python/sglang/srt/managers/scheduler.py

核心修复文件，修正预取 token 长度与 match 时长度不一致的问题，是解决 cache miss 的关键变更。

```
# sglang/srt/managers/scheduler.py (modified _prefetch_kvcache)
def _prefetch_kvcache(self, req: Req):
    if self.enable_hicache_storage:
        req.init_next_round_input(self.tree_cache, cow_mamba=False)
        last_host_node = req.last_host_node
        if last_host_node.backuped or last_host_node is self.tree_cache.root_node:
            last_hash = last_host_node.get_last_hash_value()
            matched_len = len(req.prefix_indices) + req.host_hit_length
            # 使用 _compute_max_prefix_len 计算匹配终点，与后续 match 逻辑对齐
            match_end = req._compute_max_prefix_len(
                len(req.full_untruncated_fill_ids)
            )
            new_input_tokens = req.full_untruncated_fill_ids[matched_len:match_end]

            prefix_keys = (
                last_host_node.get_prefix_hash_values(last_host_node.parent)
                if self.tree_cache.hicache_storage_pass_prefix_keys
                else None
            )
            self.tree_cache.prefetch_from_storage(
                req.rid,
                last_host_node,
                new_input_tokens,
                last_hash,
                prefix_keys,
```

```
        )
```

test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_nightly.py

新增了验证 L3 缓存完全前缀命中率的核心测试方法，并引入架构适配属性。

```
# test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_nightly.py
class AccuracyTwoPassMixin:
    # ... existing code ...
    l3_prefetch_page_size: int = 64 # 默认 page_size
    l3_prefetch_prompt_pages: int = 16 # 构造 prompt 所用的 page 数
    # 架构可覆盖该值以指示最多可能未命中的 token 数
    l3_prefetch_max_uncached_tokens: int = None

    def test_l3_prefetch_full_prefix_hit_after_flush(self):
        from sglang.test.kl_test_utils import _flush_cache, _generate

        page = int(self.l3_prefetch_page_size)
        n_tokens = page * int(self.l3_prefetch_prompt_pages)
        max_uncached = int(
            self.l3_prefetch_max_uncached_tokens
            if self.l3_prefetch_max_uncached_tokens is not None
            else page
        )

        rng = random.Random(987)
        input_ids = [rng.randint(1, 30000) for _ in range(n_tokens)]

        # 第一次 generate，填充缓存
        _generate(self.base_url, [input_ids], max_new_tokens=4)
        # flush 缓存
        _flush_cache(self.base_url)
        # 第二次 generate，应命中大多数 token
        results = _generate(self.base_url, [input_ids], max_new_tokens=4)
        cached = int(results[0]["meta_info"]["cached_tokens"])

        expected_min = n_tokens - max_uncached
        self.assertGreaterEqual(
            cached,
            expected_min,
            f"cached_tokens={cached} < {expected_min} (= input_len - {max_uncached})",
        )
```

评论区精华

核心讨论集中在预取长度是否需要与 _compute_max_prefix_len 对齐。审核者 hzh0425 提出这一问题，作者 vladnosiv 确认并修复。此外还讨论了不同模型架构（SWA vs Mamba）的缓存状态策略差异：Mamba 状态仅在 chunk 边界持久化，因此缓存命中上限不同，测试中通过

l3_prefetch_max_uncached_tokens 加以区分。

- 预取长度应与 _compute_max_prefix_len 对齐 (correctness): 作者 vladnosiv 同意并修复，将 new_input_tokens 切片终点改为 match_end。
- Mamba 模型缓存测试为何不设置更大 prompt (question): vladnosiv 解释 prompt 长度已设为 chunked_prefill_size + 16，足以验证那部分 token 的命中，并通过 l3_prefetch_max_uncached_tokens 反映 Mamba 的状态缓存间隔。
- Mamba 与 SWA 的缓存状态策略差异 (design): 同意当前测试通过 l3_prefetch_max_uncached_tokens 区分，未来可考虑更通用的方案。

风险与影响

- 风险：主要风险在于 _compute_max_prefix_len 的实现：如果该函数对非 SWA 模型返回了非预期值，可能导致预取范围变小甚至无法预取必要 token。但由于预取是尽力而为的优化，即使预取不足也不会影响正确性，仅影响缓存命中率。另外新增测试仅覆盖 L3 文件后端场景，L2 场景未覆盖。此外 Mamba 模型的缓存行为依赖 chunk 边界，测试断言较宽松，可能掩盖其他 bug。
- 影响：直接影响使用 HiCache L3 缓存的 SWA 模型（如 DeepSeek V4 Flash、GLM-5）在 flush 缓存后重新请求相同 prompt 时的缓存命中率，从几乎 0 提升至接近 prompt 长度减去 page_size/ 窗口大小。对其他模型和缓存层级无影响。团队需注意不同模型架构的 l3_prefetch_max_uncached_tokens 配置。测试扩展使缓存验证更严谨。
- 风险标记：核心路径变更，架构差异适配，测试覆盖边界

关联脉络

- 暂无明显关联 PR