PR #24943 完整报告

sgl-project/sglang

[UnifiedTree] fix: allow partial match on evicted+backuped nodes

合并时间：2026-05-13 11:43

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24943

执行摘要

- 一句话：修复被驱逐备份节点的部分匹配失败 bug
- 推荐动作：建议精读。该 PR 修复了一个涉及节点分裂与驱逐状态组合的边界逻辑，修改虽小但设计精准，测试覆盖了多种混合组件场景，体现了良好的工程实践。

功能与动机

修正前，partial match 会在遇到 host-only 节点（evicted + backuped）时提前终止，导致有效的 HiCache 命中被丢弃；修正后先 split，再让 host-only 前缀参与匹配和 load-back，使得当前请求可以利用该缓存。

实现拆解

1. 核心逻辑修改（unified_radix_cache.py）：在 _match_prefix_helper 中，当 prefix_len < len(child.key) 时，移除原先的 if child.evicted: break 条件，改为统一调用 _split_node 进行节点分裂；分裂后，仅当 not node.evicted 时才将 BASE_COMPONENT 的 value 附加到 value 列表中。
2. 新增单元测试（test_unified_radix_cache_unittest.py）：新增 test_hicache_partial_match_splits_evicted_backed_up_node 测试方法，构造一个被驱逐备份的节点，然后使用一个部分匹配的 query 触发匹配，验证 split 后的父节点是否保留了正确的前缀，以及在不同混合配置（Full、Full+Mamba、Full+SWA）下 host_hit_length 和 last_host_node 是否符合预期。

关键文件：

- python/sglang/srt/mem_cache/unified_radix_cache.py（模块 缓存层；类别 source；类型 core-logic；符号 _match_prefix_helper）：核心修改，移除 partial match 中 evicted 检查，允许对 evicted+backuped 节点进行 split
- test/registered/unit/mem_cache/test_unified_radix_cache_unittest.py（模块 测试；类别 test；类型 test-coverage；符号 test_hicache_partial_match_splits_evicted_backed_up_node）：新增单元测试，覆盖 evicted+backuped 节点 partial match 的多种混合配置场景

关键符号：_match_prefix_helper, test_hicache_partial_match_splits_evicted_backed_up_node

关键源码片段

python/sglang/srt/mem_cache/unified_radix_cache.py

核心修改，移除 partial match 中 evicted 检查，允许对 evicted+backuped 节点进行 split

```
# _match_prefix_helper 方法中的关键修改段
while len(key) > 0 and child_key in node.children:
    child = node.children[child_key]

    # HiCache: 完全死节点（evicted 且未 backuped）—— 停止遍历
    if child.evicted and not child.backuped:
        break

    prefix_len = child.key.match(key, page_size=self.page_size)
    if prefix_len < len(child.key):
        # [ 修复前 ]: 若 child.evicted 则直接 break，导致 host-only 前缀无法被利用
        # [ 修复后 ]: 直接进行 split，后续根据 node.evicted 判断是否获取 value
        node = self._split_node(child.key, child, prefix_len)
        # 分裂出的父节点若未被 evicted 才装载 BASE value
        #  （evicted 节点只有 host_value，device value 不可用）
        if not node.evicted:
            value.append(node.component_data[BASE_COMPONENT_TYPE].value)
        _update_best_if_valid(node)
        break

    if not child.evicted:
        value.append(child.component_data[BASE_COMPONENT_TYPE].value)
    node = child
    _update_best_if_valid(node)
    key = key[prefix_len:]
    if len(key):
        child_key = key.child_key(self.page_size)
return value, best_node, best_value_len
```

test/registered/unit/mem_cache/test_unified_radix_cache_unittest.py

新增单元测试，覆盖 evicted+backuped 节点 partial match 的多种混合配置场景

```
def test_hicache_partial_match_splits_evicted_backed_up_node(self):
    """Partial matches on host-only nodes must keep the host prefix usable."""
    tree, allocator, req_to_token_pool = build_fixture(self.cfg)
    ps = self.cfg.page_size
    seq = self._make_seq(1, 4) # 构造一个 4 页的序列
    expected_prefix = seq[: 2 * ps] # 预期 split 后的前缀
    expected_suffix = seq[len(expected_prefix) :] # 预期 split 后的后缀
    # query 包含预期的前缀和一个无关的后缀
    query = expected_prefix + self._make_seq(9000, 1)

    self._insert(tree, allocator, req_to_token_pool, seq)
    m = tree.match_prefix(MatchPrefixParams(key=RadixKey(seq)))
```

```
    node = m.last_device_node
    self._simulate_backup(tree, node) # 模拟节点被 backup 到 host

    tree.evict(EvictParams(num_tokens=len(seq)))
    self.assertTrue(node.evicted)
    self.assertTrue(node.backuped)

    # 用 query 进行匹配，此时应触发 partial match 并对 evicted+backuped 节点进行 split
    m = tree.match_prefix(MatchPrefixParams(key=RadixKey(query)))

    self.assertEqual(len(m.device_indices), 0)
    self.assertIs(m.last_device_node, tree.root_node)

    split_parent = node.parent
    self.assertIsNot(split_parent, tree.root_node)
    self.assertTrue(split_parent.evicted)
    self.assertTrue(split_parent.backuped)
    self.assertEqual(split_parent.key.token_ids, expected_prefix)
    self.assertEqual(node.key.token_ids, expected_suffix)

    if self.cfg.has_mamba:
        # Mamba 组件只存在于叶子节点，split 后的 host-only 父节点无 Mamba 数据
        self.assertEqual(m.host_hit_length, 0)
        self.assertIs(m.last_host_node, tree.root_node)
        self.assertIsNone(
            split_parent.component_data[ComponentType.MAMBA].host_value
        )
    else:
        # 非 Mamba 组件，host-only 前缀可以被识别
        self.assertEqual(m.host_hit_length, len(expected_prefix))
        self.assertIs(m.last_host_node, split_parent)
    tree.sanity_check()
```

评论区精华

审查者 hzh0425 要求添加单元测试覆盖 Full+Mamba、Full+SWA 的部分匹配场景。作者 alphabetc1 回应已补充测试，并列出不同配置下的匹配结果表格：Full 为 cache hit，Full+Mamba 为 cache miss（Mamba leaf-only 导致 split 后不符合 hybrid 边界），Full+SWA 为 cache hit，Full+SWA+Mamba 为 cache miss。测试通过 CI。

- 添加单元测试覆盖混合配置 (testing): 作者已添加并列出测试结果表格，覆盖 Full、Full+Mamba、Full+SWA、Full+SWA+Mamba 四种配置

风险与影响

- 风险：本次仅修改 5 行源码（2 行新增，3 行删除），逻辑清晰且被单元测试覆盖。风险较低，但需要注意：对于 Mamba 等仅叶子节点有效的组件，split 后的 host-only 节点可能无法被正确使用，测试验证了这种情况会正确返回 miss。没有性能或安全风险。

- 影响：影响范围限定在 HiCache 相关模块（unified_radix_cache.py），主要影响使用 HiCache 特性进行前缀匹配的场景。修正后，被驱逐备份节点中的主机端前缀可以被正确利用，提高缓存命中率。对不使用 HiCache 的场景无影响。
- 风险标记：核心路径变更，边界条件修复

关联脉络

- PR #25068 [UnifiedTree]: Fix the leaf determination logic in _cascade_evict.: 也涉及 unified_radix_cache.py 的叶子判定逻辑修复，与当前 PR 同属 UnifiedTree 的 bugfix 系列
- PR #25022 [Bugfix, NSA HiCache] Fix missing override_kv_cache_dim in attach_hybrid_nsa_pool_to_hiradix_cache: 同一代码区域（HiCache hybrid_cache）的 bugfix，涉及 overlapping 参数