PR #36317 完整报告

sgl-project/sglang

[HiCache] Keep auxiliary load-back out of Full KV pending ownership

合并时间：2026-08-27 01:30

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36317

执行摘要

- 一句话：HiCache 辅助加载不再复用 Full KV 固定标记
- 推荐动作：值得精读，原因有三：一是它是所有权分离的典型设计案例，核心数据结构只固定真正共享的 Full KV 资源，辅助资源由各自锁保护；二是展示了大 PR 拆分聚焦修复的工程实践；三是测试构造了清晰的跨组件重叠场景，可当作 HiCache 并发语义的入门样例。

功能与动机

写回模式下，load_back_pending_id 用于防止 Full KV 主存重复回收在 H2D 传输期间释放主机槽；但 commit_load_back 把该 Full 专属的标量属主也赋给了 SWA 与 Mamba 传输，使安全的跨组件重叠加载失败。这是 #34975 报告的 Full+Mamba 场景与 #35931 报告的 Full+SWA 场景。本 PR 从 #34515 中聚焦抽取该修复，独立评审验证。

实现拆解

1. 收紧固定范围： 在 python/sglang/srt/mem_cache/unified_cache/unified_tree_core.py 的 commit_load_back 中，原先遍历 kv_xfer 与全部 comp_xfers 的节点统一设置 load_back_pending_id，现改为只遍历 kv_xfer.nodes_to_load。
2. 保留安全断言： 对不同 anchor 再次固定同一节点仍触发 AssertionError，确保真实重叠的 Full KV 加载不会被静默覆盖。
3. 移交保护责任： SWA 与 Mamba 辅助传输不再使用 Full 专属 pin，其源节点生命周期继续由各自的 window / state host 锁保证，避免跨组件固定冲突。
4. 补充回归测试： 在 test/registered/unit/mem_cache/test_unified_radix_cache_unittest.py 新增 test_auxiliary_load_does_not_reuse_full_pending_pin，构造 anchor_a Full 加载固定 shared、anchor_b SWA 加载同一 shared 的场景，断言 shared 的 pin 保持 anchor_a 且 anchor_b 自身被 pin。
5. 验证： 96 个聚焦 HiCache load-back/write-back 测试与完整 pre-commit 套件通过；CI 组测试有偶发失败重跑记录，最终全绿。

关键文件：

- python/sglang/srt/mem_cache/unified_cache/unified_tree_core.py（模块 缓存核心；类别 source；类型 core-logic；符号 commit_load_back）：核心修复点：将 commit_load_back 中写回模式的 pin 范围从所有传输收紧为仅 Full KV 源节点，避免 SWA/Mamba 辅助加载占用 load_back_pending_id 引发的跨组件重叠冲突。

- test/registered/unit/mem_cache/test_unified_radix_cache_unittest.py（模块 缓存测试；类别 test；类型 test-coverage；符号 test_auxiliary_load_does_not_reuse_full_pending_pin）：新增回归测试，模拟 anchor_a Full KV 加载固定 shared 节点后，anchor_b 的 SWA 传输再次加载同一节点，验证不会覆盖 Full pin。

关键符号：commit_load_back, test_auxiliary_load_does_not_reuse_full_pending_pin

关键源码片段

python/sglang/srt/mem_cache/unified_cache/unified_tree_core.py

核心修复点：将 commit_load_back 中写回模式的 pin 范围从所有传输收紧为仅 Full KV 源节点，避免 SWA/Mamba 辅助加载占用 load_back_pending_id 引发的跨组件重叠冲突。

```
def commit_load_back(
    self,
    node_id: NodeId,
    device_indices: torch.Tensor,
    kv_xfer: PoolTransfer,
    comp_xfers: dict[ComponentType, list[PoolTransfer]],
) -> list[CacheAction | ComponentAction]:
    """提交一次成功的 H->D load-back；SWA full->swa 映射重建推迟到编排层。"""
    node = self.node_by_id(node_id)
    cache_actions: list[CacheAction | ComponentAction] = []
    if self.is_write_back:
        # 只 Pin Full KV 主存 host 槽位，防止写回重复回收在 H2D DMA 期间释放。
        # 辅助池（SWA、Mamba）有独立的 host 锁，可以合法地以不同 anchor 加载同一节点。
        for nid in kv_xfer.nodes_to_load or ():
            pinned = self.node_by_id(nid)
            # 只允许同一个 anchor 重复固定；不同 anchor 的真实 Full KV 重叠仍会触发断言。
            assert pinned.load_back_pending_id in (None, node_id)
            pinned.load_back_pending_id = node_id
    kv_xfer.device_indices = device_indices
    self.components_by_type[BASE_COMPONENT_TYPE].commit_hicache_transfer(
        node, CacheTransferPhase.LOAD_BACK, [kv_xfer], cache_actions
    )
    for nid in kv_xfer.nodes_to_load or ():
        self.kv_events.record_store(self.node_by_id(nid), medium=StorageMedium.GPU)
    for ct, xfers in comp_xfers.items():
        self.components_by_type[ct].commit_hicache_transfer(
            node, CacheTransferPhase.LOAD_BACK, xfers, cache_actions
        )
    self._update_evictable_leaf_sets(node)
    return cache_actions
```

test/registered/unit/mem_cache/test_unified_radix_cache_unittest.py

新增回归测试，模拟 anchor_a Full KV 加载固定 shared 节点后，anchor_b 的 SWA 传输再次加载同一节点，验证不会覆盖 Full pin。

```
def test_auxiliary_load_does_not_reuse_full_pending_pin(self):
```

```
    # 验证辅助传输不会覆盖 Full KV 专属的 load_back_pending_id
    core, shared, anchor_a, anchor_b = self._build_core(is_write_back=True)
    core.components_by_type[ComponentType.SWA] = mock.Mock()

    # anchor_a 的 Full KV 加载先固定 shared 节点
    self._commit_load_back(core, anchor_a, shared)
    self.assertEqual(shared.load_back_pending_id, anchor_a.id)

    # anchor_b 的 KV 传输指向自己，SWA 传输指向同一 shared 节点
    kv_transfer = PoolTransfer(
        name=PoolName.KV,
        host_indices=torch.tensor([1], dtype=torch.int64),
        nodes_to_load=[anchor_b.id],
    )
    swa_transfer = PoolTransfer(
        name=PoolName.SWA,
        host_indices=torch.tensor([2], dtype=torch.int64),
        nodes_to_load=[shared.id],
    )
    UnifiedTreeCore.commit_load_back(
        core,
        anchor_b.id,
        torch.tensor([3], dtype=torch.int64),
        kv_transfer,
        {ComponentType.SWA: [swa_transfer]},
    )

    # shared 的 pin 仍属于 anchor_a，anchor_b 只 pin 自己
    self.assertEqual(shared.load_back_pending_id, anchor_a.id)
    self.assertEqual(anchor_b.load_back_pending_id, anchor_b.id)
```

评论区精华

评审仅有一条 APPROVED，hzh0425 评论 Looks good，说明方案获得认可，没有留下实质性的技术异议。PR 期间的公开讨论主要在 CI 侧：radix_cache/unified_radix_tree 组在 4-gpu-h100 上被多次重跑，其中一次失败，重跑通过后以 /rerun-failed-ci 收尾，未暴露代码问题。

- 辅助池是否应复用 Full KV 的 load_back_pending_id (correctness): 辅助池依赖各自组件 host 锁保护，不再占用 Full 专属 pin；保留不同 anchor 断言防止真正重叠的 Full KV 加载。
- radix_cache/unified_radix_tree 测试组在 H100 上的稳定性 (testing): 最终 CI 全绿，未发现与本次改动相关的回归；偶发失败可能与测试组资源或时序敏感有关。

风险与影响

- 风险：主要风险来自保护责任转移：辅助池源节点的安全性现在完全依赖 SWA window host 锁与 Mamba state host 锁的持有范围和 H2D 传输生命周期匹配。单元测试用

mock.Mock() 替代 SWA 组件，无法端到端验证这些锁的真实时序；若未来更改锁的释放时机，可能重现同类竞态且更难定位。另外，不同 anchor 的 Full 重叠加载断言保持不变，但新增辅助组件类型时若再次误加 pin，会回到此前的失败模式。CI 上该测试组有偶发失败记录，虽然最终通过，仍需留意时序敏感性。

- 影响：对使用 HiCache 写回模式的推理服务，该修复解除跨组件重叠加载的限制，直接支撑 #34515 中更大范围的重叠调度优化；对 Full+SWA、Full+Mamba 场景从报错或拒绝转为可安全并发。非 HiCache 用户不受影响。对团队而言，这是从大 PR #34515 中拆分出的高价值小修，便于独立评审、验证与回溯。
- 风险标记：核心缓存并发路径变更，依赖组件 host 锁生命周期，CI 测试组偶发不稳定

关联脉络

- PR #34515 Remove many scheduler synchronization to enable overlap scheduler on agent workload: 本 PR 由 #34515 拆出，聚焦其中的 load-back 所有权修复；#34515 仍是集成与端到端性能参考。
- PR #35944 Pin scheduler metadata before asynchronous H2D copies: 同为 #34515 的拆分，关注 H2D 异步拷贝与调度并发，与本 PR 在并发固定模式上互补。