PR #26227 完整报告

sgl-project/sglang

[PD]: Support HiCache prefetching and pd-incremental transfer on decode side

合并时间：2026-06-02 15:40

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26227

执行摘要

- 一句话：decode 端支持 HiCache 三层缓存预取与 PD 增量传输
- 推荐动作：该 PR 是 PD 项目的重要演进，值得所有关注长上下文延迟和缓存效率的读者精读。重点关注 query_storage_hit_length 中的 all_reduce 同步设计、_process_hicache_local_restore 的状态机调度方式，以及如何通过继承 Mixin 实现功能扩展。建议后续跟进批量 all_reduce 优化和并发加载支持。

功能与动机

作为 PD Disaggregation Roadmap (#21703) 的重要组成部分，该 PR 旨在填补 decode 侧无法利用分层缓存的空白。在现有 PD 模式中，decode 端只能使用设备端 L1 缓存，当缓存未命中时需从 prefill 端全量重传 KV cache，导致 TTFT 较高。通过引入 HiCache 的 L2(主机) 和 L3(存储) 缓存层次，decode 端可以直接从本地恢复部分 KV cache，减少跨节点传输依赖。Roadmap Issue #21846 明确将 'Precisely control token budget' 和 'Improve parallelization between HiCache transfer and PD transfer' 作为前置步骤。

实现拆解

1. 定义核心数据结构：在新增文件 decode_hicache_mixin.py 中定义 DecodePrefixMatch dataclass（包含 l1_prefix_len、decode_prefix_len 等属性）和 HiCacheRestoreResult 枚举（PENDING/READY/FAILED），为三层缓存匹配和本地恢复提供类型化表示。
2. 扩展DecodeRequest：在decode.py的DecodeRequest dataclass中增加prefix_match、hicache_restored_kv_indices、hicache_restore_status 等字段，用于跟踪请求在整个缓存恢复流程中的状态。
3. 重构 DecodePreallocQueue：让 DecodePreallocQueue 继承新混入类 DecodeHiCachePreallocMixin；修改 _match_prefix_and_lock 方法，使其返回 DecodePrefixMatch 而不是简单的 (indices, length)，并在启用 decode 端 HiCache 时调用 query_storage_hit_length 查询 L3 存储匹配长度。
4. 触发预取和本地恢复：在 pop_preallocated 中调用 _start_hicache_prefetch 发送 L3 存储预取请求；在 pop_transferred 中新增状态机逻辑，驱动 L2→L1 本地加载（通过 init_load_back），并在 _process_hicache_local_restore 中轮询加载完成事件和检查所有挂起事件。

5. 底层缓存支持：在 hiradix_cache.py 中新增 query_storage_hit_length 方法，用于计算 L3 存储命中长度（包含跨 rank all_reduce 同步）；新增 is_load_back_event_done 方法用于轮询异步加载完成。在 scheduler.py 中新增 enable_decode_hicache 开关，仅在同时启用 disaggregation_decode_radix_cache 和 hierarchical_cache 时激活。
6. 测试与配置：在测试文件 test_disaggregation_decode_radix_cache.py 中新增 TestDisaggregationDecodeRadixHiCacheFileBackend 类，端到端验证三层缓存恢复行为（包括 flush cache 后的 L3 重用场景）；调整单元测试中的 mock 函数签名以支持新增的 total_prefix_len 参数。

关键文件：

- python/sglang/srt/disaggregation/decode_hicache_mixin.py（模块 缓存层；类别 source；类型 core-logic；符号 DecodePrefixMatch, l1_prefix_len, decode_prefix_len, needs_local_restore）：新增文件，定义三层缓存匹配数据结构、恢复结果枚举和 Prealloc Mixin 类，是本次变更的核心逻辑枢纽。
- python/sglang/srt/disaggregation/decode.py（模块 调度器；类别 source；类型 core-logic；符号 DecodePreallocQueue, _match_prefix_and_lock, DecodeTransferQueue, DecodeRequest）：主调度文件，修改了 DecodeRequest、DecodePreallocQueue 和 DecodeTransferQueue 的导入与转发逻辑，整合 HiCache 状态机。
- python/sglang/srt/mem_cache/hiradix_cache.py（模块 缓存层；类别 source；类型 core-logic；符号 is_load_back_event_done, query_storage_hit_length）：底层缓存实现，新增 query_storage_hit_length 和 is_load_back_event_done 方法，为 decode 侧 L3 存储查询和加载完成检测提供基础。
- test/registered/disaggregation/test_disaggregation_decode_radix_cache.py（模块 集成测试；类别 test；类型 test-coverage；符号 TestDisaggregationDecodeRadixHiCacheFileBackend, setUpClass, tearDownClass, _post_ok）：新增端到端测试类 TestDisaggregationDecodeRadixHiCacheFileBackend，验证 flush cache 后 L3 重用 decode 输出，确保三层缓存在 PD 场景下正确工作。
- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 configuration）：新增 enable_decode_hicache 开关，控制是否在 decode 端启用 HiCache。
- test/registered/unit/managers/test_priority_scheduling_disaggregation.py（模块 单元测试；类别 test；类型 test-coverage；符号 pre_alloc_mock）：调整 mock 函数以适配新的 _pre_alloc 签名增加 total_prefix_len 参数。
- test/registered/unit/mem_cache/test_decode_radix_lock_ref.py（模块 单元测试；类别 test；类型 test-coverage）：配合 decode 侧缓存锁引用测试调整，但改动量小。

关键符号：DecodeHiCachePreallocMixin._build_decode_prefix_match, DecodeHiCachePreallocMixin._start_hicache_prefetch, DecodePreallocQueue._match_prefix_and_lock, DecodePreallocQueue.pop_preallocated, DecodeTransferQueue._process_hicache_local_restore, HiRadixCache.query_storage_hit_length, HiRadixCache.is_load_back_event_done

关键源码片段

python/sglang/srt/mem_cache/hiradix_cache.py

底层缓存实现，新增 query_storage_hit_length 和 is_load_back_event_done 方法，为 decode 侧 L3 存储查询和加载完成检测提供基础。

```
def query_storage_hit_length(
    self,
    last_host_node: TreeNode,
    new_input_tokens: List[int],
    last_hash: Optional[str] = None,
    prefix_keys: Optional[List[str]] = None,
) -> int:
    """
    查询 L3 存储中的前缀命中长度。
    若存储不可用或受速率限制，返回 0。否则构造 PrefetchOperation
    并调用 _storage_hit_query，再通过 all_reduce 取所有 rank 的最小值。
    """
    # 存储未启用或预取被限速则直接返回 0
    if not self.enable_storage or self.cache_controller.prefetch_rate_limited():
        return 0

    # 构造 radix key，page 对齐
    prefetch_key = RadixKey(
        new_input_tokens,
        extra_key=last_host_node.key.extra_key,
        is_bigram=self.is_eagle,
    ).page_aligned(self.page_size)
    if len(prefetch_key) < self.prefetch_threshold:
        return 0

    operation = PrefetchOperation(
        "__storage_hit_query__",
        self.cache_controller.mem_pool_host.get_dummy_flat_data_page()[:0],
        prefetch_key,
        last_hash,
        prefix_keys,
    )
    _, storage_hit_count = self.cache_controller._storage_hit_query(operation)
    # 所有 rank 取最小值以保证一致性
    storage_hit_count_tensor = torch.tensor(storage_hit_count, dtype=torch.int)
    self._all_reduce_attn_groups(
        storage_hit_count_tensor, torch.distributed.ReduceOp.MIN
    )
    storage_hit_count = storage_hit_count_tensor.item()
    storage_hit_count = storage_hit_count - (storage_hit_count % self.page_size)
    return storage_hit_count

def is_load_back_event_done(self, consumer_index: int) -> bool:
```

```
    """检查指定 consumer 的本地加载事件是否完成"""
    if consumer_index < 0:
        return True
    finish_event = self.cache_controller.layer_done_counter.events[
        consumer_index
    ].finish_event
    if not finish_event.query():
        return False
    self.loading_check()
    return True
```

评论区精华

Review 重点讨论了以下设计权衡和潜在风险：

1. per-request all_reduce 性能风险：gemini-code-assist[bot] 指出 query_storage_hit_length 在 request 循环中调用 all_reduce 可能导致性能瓶颈。ShangmingCai 回应可后续优化，hzh0425 说明已通过 prefetch_rate_limited 做限流。
2. check_hicache_events 在循环中：同样由 bot 指出，应在 batch 级别调用一次而非 per-request。
3. 冗余前缀匹配：bot 指出 _process_hicache_local_restore 中重新调用 match_prefix_for_req 是冗余的，可能导致不一致。最终决定复用 DecodeRequest 中存储的 prefix_match。
4. _start_hicache_prefetch 异常处理：ShangmingCai 询问是否可能失败，hzh0425 随后添加了 fallback 逻辑（捕获异常后降级为 L2-only 恢复）。
5. TP rank 发散风险：ShangmingCai 关注对 prefetch_rate_limited 等分支的跨 rank 一致性，hzh0425 确认依赖变量已全局同步。

- query_storage_hit_length 中的 all_reduce 性能风险 (performance): 作者 hzh0425 说明已通过 prefetch_rate_limited 做限流，且该方法本身包含 all_reduce 保证一致性；ShangmingCai 同意可后续优化。
- check_hicache_events 在循环中被调用 (performance): 未在 comments 中明确是否修改，但从最终代码看可能已优化（因为最终提交修复了 comment）。
- 冗余的前缀匹配调用 (correctness): 作者 hzh0425 确认已改为复用 DecodeRequest.prefix_match，避免重复匹配。
- _start_hicache_prefetch 异常处理 (correctness): hzh0425 添加了 fallback 逻辑，捕获异常后降级为 L2-only 恢复。
- TP rank 节点变异的并发安全 (correctness): hzh0425 回应 init_load_back 触发 L2→L1 加载，通常不会失败；若异常，清除逻辑应已验证。

风险与影响

- 风险：
  1. 性能风险（高）：query_storage_hit_length 中的 all_reduce 在 per-request 粒度执行，当 batch size 增大时可能成为瓶颈。虽然当前有 prefetch_rate_limited 限制，但高频

同步仍可能增加调度延迟。

2. 冗余匹配不一致风险（中）：原实现中 _process_hicache_local_restore 重新调用 match_prefix_for_req，若 radix 树在此期间变更，可能导致分配与恢复参数不一致。Review 后改为复用已存储的 prefix_match，已缓解。
3. 序列化加载瓶颈（中）：_process_hicache_local_restore 中检查 len(self.tree_cache.ongoing_load_back) 来控制并发加载，可能导致请求排队，影响吞吐。
4. TP rank 状态发散（低）：在 query_storage_hit_length 中如果因为 prefetch_rate_limited 提前返回 0，各 rank 可能分歧。但作者称依赖变量已同步，且该方法本身就包含 all_reduce，因此风险较低。
5. 事件泄漏风险（低）：若请求在恢复中途被 abort，_clean_hicache_prefetch_resources 能否正确清理所有状态需要测试验证。 - 影响：对用户：启用 --disaggregation-decode-enable-radix-cache 和 --enable-hierarchical-cache 后，PD 部署的 TTFT 显著降低（平均 -39%，P99-52%），TPOT 略有上升（~2.5%），适合 TTFT 敏感型场景。

对系统：增加 decode 端的内存占用（L2/L3 缓存），并新增存储后端（文件 / 分布式）的读写压力。需监控磁盘 IO 和内存使用。

对团队：新增一个源文件（decode_hicache_mixin.py 约 300 行）和测试框架，维护复杂度提升。引入的状态机增加了调度逻辑的调试难度，需配套详细的日志和 metrics。

兼容性：仅在同时启用 --disaggregation-decode-enable-radix-cache 和 --enable-hierarchical-cache 时激活，默认行为不变，向后兼容。

- 风险标记：per-request all_reduce 同步，冗余匹配不一致，序列化加载瓶颈，TP rank 发散

关联脉络

- PR #21703 [Roadmap] Prefill-Decode Disaggregation Roadmap (2026 Q2): 该 PR 是 roadmap 的具体实现项，旨在补充 decode 侧 HiCache 支持。
- PR #26939 [Bug Fix][HiCache] Drop @lru_cache on UnifiedTreeNode.get_prefix_hash_values: 修复了 HiCache 统一 radix 树缓存突变 bug，与 decode 侧缓存正确性相关。
- PR #21591 [PD] Compatibility with Hisparse: 允许在 PD 模式下从 prefill HBM 向 decode host 发送 KV cache，是本次 decode 侧恢复传输的基础。