PR #27264 完整报告

sgl-project/sglang

[UnifiedTree]: Sync sidecar component hits across TP ranks and make SWA prefetch all-or-nothing

合并时间：2026-06-05 09:23

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27264

执行摘要

- 一句话：跨 TP 同步 sidecar 命中，SWA 预取全或无
- 推荐动作：值得精读 unified_radix_cache.py 中打包 all_reduce 的设计模式，可复用至其他 sidecar 组件。swa_component.py 的全或无逻辑可作为类似"原子预取"实现的参考。但需关注 review 中遗留的 pool_storage_result 保护问题和 sidecar 顺序假设，建议后续修复。

功能与动机

TP 多卡环境下，各 rank 的 sidecar 组件（如 SWA）预取进度可能不一致，导致 host tree 结构不同，影响后续 decode 一致性。本 PR 旨在通过 all-reduce 同步 sidecar 命中计数，并以全或无策略强制 SWA 预取原子性。

实现拆解

1. 修改 swa_component.py 的 build_hicache_transfers 中 PREFETCH 阶段：将滑动窗口页数 sw_pages 代替原有的 min 动态计算，当可用 token 不足一个完整窗口时返回 None（全或无）。
2. 同步修改同一文件的 _commit_prefetch：当 loaded_pages < window_require_pages 时释放整个 host 缓冲区而不分发，确保 TP 间树结构一致。
3. 在 unified_radix_cache.py 的 check_prefetch_progress 中，将 completed_tokens 与 sidecar 池的 hit_pages 打包为单一 tensor，通过 all_reduce MIN 一次性同步，取代原来仅同步 completed_tokens 的做法。
4. 在单元测试 test_unified_radix_cache_unittest.py 中添加 _patch_tp_all_reduce 等辅助函数模拟 TP 环境，验证全或无行为；在集成测试 test_unified_radix_cache_kl_dsv4.py 中新增 TestUnifiedDeepSeekV4FlashHiCacheL3 类，启动 file backend HiCache 服务并执行两遍精度比对。

关键文件：

- python/sglang/srt/mem_cache/unified_cache_components/swa_component.py（模块 缓存组件；类别 source；类型 core-logic；符号 build_hicache_transfers, _commit_prefetch）：核心逻辑：调整 SWA 预取为全或无，修改 build_hicache_transfers 和 _commit_prefetch。

- python/sglang/srt/mem_cache/unified_radix_cache.py（模块 缓存层；类别 source；类型 core-logic；符号 check_prefetch_progress）：核心逻辑：在 check_prefetch_progress 中将 completed tokens 与 sidecar 命中打包为单一 tensor 进行跨 rank MIN all_reduce。
- 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 _patch_tp_all_reduce, swa_packed_index, fake, _swa_host_on_path）：新增单元测试辅助函数，模拟 TP all_reduce 并验证 SWA 全或无行为。
- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_dsv4.py（模块 集成测试；类别 test；类型 test-coverage；符号 TestUnifiedDeepSeekV4FlashHiCacheL3, setUpClass, tearDownClass）：新增集成测试类 TestUnifiedDeepSeekV4FlashHiCacheL3，验证 HiCache L3 file 后端在 DeepSeek V4 模型上的精度。

关键符号：build_hicache_transfers, _commit_prefetch, check_prefetch_progress, _patch_tp_all_reduce, swa_packed_index

关键源码片段

python/sglang/srt/mem_cache/unified_cache_components/swa_component.py

核心逻辑：调整 SWA 预取为全或无，修改 build_hicache_transfers 和 _commit_prefetch。

```
# swa_component.py - PREFETCH 阶段的全或无实现
def build_hicache_transfers(self, node, phase, prefetch_tokens=0):
    # ... 前面处理 BACKUP_HOST, BACKUP_STORAGE 等阶段 ...
    if phase == CacheTransferPhase.PREFETCH:
        # 计算完整滑动窗口所需页数
        sw_pages = (
            self.sliding_window_size + self.cache.page_size - 1
        ) // self.cache.page_size
        # 可用 token 不足一个窗口时直接返回 None（全或无）
        if sw_pages == 0 or prefetch_tokens // self.cache.page_size < sw_pages:
            return None
        num_tokens = sw_pages * self.cache.page_size
        host_indices = self._swa_kv_pool_host.alloc(num_tokens)
        if host_indices is None:
            self.cache.evict_host(num_tokens, ComponentType.SWA)
            host_indices = self._swa_kv_pool_host.alloc(num_tokens)
        if host_indices is None:
            return []
        return [
            PoolTransfer(
                name=PoolName.SWA,
                host_indices=host_indices,
                keys=["__placeholder__"] * sw_pages,
                hit_policy=PoolHitPolicy.TRAILING_PAGES,
```

```
            )
        ]
    return None

# _commit_prefetch 中的 all-or-nothing 检查
def _commit_prefetch(self, anchor, transfers, *, insert_result=None, pool_storage_result=None):
    # ...
    page_size = self.cache.page_size
    host_indices = transfers[0].host_indices
    window_require_pages = (
        host_indices.numel() // page_size if host_indices is not None else 0
    )
    loaded_pages = pool_storage_result.extra_pool_hit_pages.get(PoolName.SWA, 0)
    target = insert_result.inserted_host_node if insert_result else None
    # 未达到完整窗口则释放整个 buffer，保持 TP 树结构一致
    if target is None or window_require_pages == 0 or loaded_pages < window_require_pages:
        self._release_swa_host(host_indices)
        return
    # 否则正常分发 buffer 到路径节点
    loaded_start = insert_result.total_len - window_require_pages * page_size
    # ...
```

python/sglang/srt/mem_cache/unified_radix_cache.py

核心逻辑：在 check_prefetch_progress 中将 completed tokens 与 sidecar 命中打包为单一 tensor 进行跨 rank MIN all_reduce。

```
# unified_radix_cache.py - check_prefetch_progress 中的打包 all_reduce
def check_prefetch_progress(self, req_id: str) -> bool:
    # ...
    completed_tokens, hash_value = self.cache_controller.terminate_prefetch(operation)
    min_completed_tokens = completed_tokens
    hit_pages = operation.pool_storage_result.extra_pool_hit_pages
    if self.tp_world_size > 1:
        # 收集当前 prefetch 用到的 sidecar 池名称（如 SWA）
        sidecar_pools = [t.name for xfers in comp_xfers.values() for t in xfers]
        # 将 completed_tokens 和各 sidecar 的命中页数打包成一个 tensor
        # 一次 all_reduce MIN 保证所有 rank 取到相同的最小值
        packed = torch.tensor(
            [completed_tokens] + [hit_pages.get(p, 0) for p in sidecar_pools],
            dtype=torch.int,
        )
        torch.distributed.all_reduce(packed, op=torch.distributed.ReduceOp.MIN, group=self.tp_
        group)
        min_completed_tokens = int(packed[0].item())
        # 将同步后的命中页数写回 hit_pages，供后续 commit 使用
        for i, p in enumerate(sidecar_pools, start=1):
            hit_pages[p] = int(packed[i].item())
    # ... 后续根据 min_completed_tokens 插入 host 节点并调用 component commit
```

评论区精华

- gemini-code-assist[bot]指出在 unified_radix_cache.py 中直接访问 operation.pool_storage_result.extra_pool_hit_pages 有风险：若 pool_storage_result 为 None 会引发 AttributeError，建议加上默认空字典保护。该意见未被修复即合并，属于遗留风险。
- ispobock询问测试代码 _patch_tp_all_reduce 中的 swa_packed_index 假设 SWA 是最后一个 sidecar，但若存在多个 sidecar 则顺序不确定。作者回复"good idea"但未进一步修改，存在潜在顺序依赖风险。
- pool_storage_result 可能为 None 的风险 (correctness): 未修复即合并；需要后续补丁处理。
- 测试中 sidecar 顺序假设 (design): 作者认同但未修改，存在潜在顺序依赖。

风险与影响

- 风险：
  1. unified_radix_cache.py 第 1893 行 (hit_pages = operation.pool_storage_result.extra_pool_hit_pages) 缺少 None 保护，若 pool_storage_result 为 None 时直接崩溃。
  2. 测试辅助函数 swa_packed_index 硬编码 SWA 位于 sidecar 列表的最后，若未来新增其他 sidecar 类型，打包索引可能错位，导致错误的 all_reduce 结果。
  3. SWA 预取的全或无策略可能降低高负载下小请求的预取频率，影响缓存命中率。
  4. 新集成测试仅覆盖 DeepSeek V4 Flash 模型且使用 file 后端，对 kernel 后端或其他模型尚未验证。 - 影响：影响范围限于启用 HiCache 且使用 TP 多卡的用户群体（如 DeepSeek V4）。全或无策略使 SWA 预取行为更保守，减少了因部分预取导致的 TP 间不一致风险，但可能增加 host 内存占用峰值和预取延迟。测包新增 2 个用例，有助于防止回归。 - 风险标记：潜在属性错误风险，侧边池顺序假设，全或无影响命中率

关联脉络

- 暂无明显关联 PR