PR #35221 完整报告

sgl-project/sglang

[HiCache] Support DCP with DSpark

合并时间：2026-08-19 16:42

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35221

执行摘要

- 一句话：支持 DCP 与 DSpark 结合，并修复 host pool 尺寸问题
- 推荐动作：值得精读，尤其是 hybrid_pool_assembler.py 中关于逻辑索引空间的处理，以及 server_args.py 中参数兼容性的设计方式。

功能与动机

根据 PR 描述，DCP+HiCache+DSpark 组合在 server 参数验证时被拒绝。移除守卫后暴露了 draft HiCache sidecar 的尺寸缺陷：draft 索引继承了 target 缓存的 DCP 全局逻辑索引空间，而 draft host pool 的尺寸只基于 target 的每 rank 物理行数，导致 L1 驱逐覆盖范围不够时，draft host 传输出错。此外，社区用户报告了相同的缺陷，并指出修复前会出现静默越界写入。

实现拆解

本 PR 主要包含三处逻辑修改：

1. 在 server_args.py 的 _resolve_hicache_dcp_compatibility 中，将 speculative_algorithm 的限制从 " 不能非 None" 改为 " 仅允许 DSPARK"，从而支持 DCP 与 DSpark 组合。
2. 在 hybrid_pool_assembler.py 的 build_full_draft_pools 中，将 host pool 的 host_to_device_ratio 从基于物理 size 改为基于逻辑 logical_size，以匹配 draft 索引的全局逻辑空间。
3. 在 kv_cache_builder.py 的 _register_legacy_hicache_draft 中做了相同修改，同步到 legacy 路径。
4. 新增测试 test_full_builder_sizes_sidecar_for_anchor_logical_space，验证了修改后的行为。

关键文件：

- python/sglang/srt/server_args.py（模块 参数解析；类别 source；类型 core-logic）：放宽了 DCP 与 DSpark 的限制，允许结合使用。
- python/sglang/srt/mem_cache/hybrid_cache/hybrid_pool_assembler.py（模块 资源池组装；类别 source；类型 core-logic）：核心修复：draft host pool 尺寸改用 logical_size。

- python/sglang/srt/mem_cache/kv_cache_builder.py（模块 缓存构建；类别 source；类型 core-logic）：同步 legacy 路径的尺寸修改，保持一致性。
- test/registered/unit/mem_cache/test_hybrid_pool_assembler.py（模块 资源池测试；类别 test；类型 test-coverage；符号 test_full_builder_sizes_sidecar_for_anchor_logical_space）：新增回归测试，确保修改生效。

关键符号：_resolve_hicache_dcp_compatibility, build_full_draft_pools, _register_legacy_hicache_draft, test_full_builder_sizes_sidecar_for_anchor_logical_space

关键源码片段

python/sglang/srt/server_args.py

放宽了 DCP 与 DSpark 的限制，允许结合使用。

```
# python/sglang/srt/server_args.py

def _resolve_hicache_dcp_compatibility(self):
    if self.dcp_size <= 1 or not self.enable_hierarchical_cache:
        return
    # L3 storage 在 DCP 下尚不支持，直接报错
    if self.hicache_storage_backend is not None:
        raise NotImplementedError(
            "--hicache-storage-backend (L3) with --dcp-size > 1 is not "
            "supported yet: under DCP each rank holds a distinct interleaved "
            "MLA KV shard..."
        )
    # 关键修改：从 " 不支持任何 speculative" 变为 " 仅支持 DSPARK"
    if self.speculative_algorithm not in (None, "DSPARK"):
        raise NotImplementedError(
            "HiCache with --dcp-size > 1 only supports DSPARK speculative "
            "decoding; other draft-model host pools have no DCP index "
            "translation."
        )
    # 其余检查保持不变 ...
```

python/sglang/srt/mem_cache/hybrid_cache/hybrid_pool_assembler.py

核心修复：draft host pool 尺寸改用 logical_size。

```
# python/sglang/srt/mem_cache/hybrid_cache/hybrid_pool_assembler.py

def build_full_draft_pools(
    *,
    draft_kv_pool: Any,
    tree_cache: Any,
    server_args: ServerArgs,
) -> tuple[list[SidecarPoolSpec], list[PoolEntry]]:
    """Build draft KV/DSA sidecars whose indices follow target full KV."""
    pool = draft_kv_pool
```

```
    if isinstance(pool, HybridLinearKVPool):
        pool = pool.full_kv_pool
    if pool.layer_num == 0:
        return [], []

    controller = tree_cache.cache_controller
    host_pool_group = controller.mem_pool_host

    # Note(kpham-sgl): DCP x DSpark draft KV is replicated and spans the virtual
    # loc space, so match the target host's logical_size instead of physical size.
    draft_host_pool = _build_mha_mla_host_pool(
        pool=pool,
        host_to_device_ratio=host_pool_group.logical_size / pool.size, # 改用 logical_size
        page_size=controller.page_size,
        layout=server_args.hicache_mem_layout,
        allocator_type=_get_allocator_type(server_args),
        pool_label="draft",
    )
    # 其余逻辑不变 ...
```

python/sglang/srt/mem_cache/kv_cache_builder.py

同步 legacy 路径的尺寸修改，保持一致性。

```
# python/sglang/srt/mem_cache/kv_cache_builder.py

def _register_legacy_hicache_draft(
    *,
    tree_cache,
    draft_pool,
    server_args: ServerArgs,
    page_size: int,
) -> None:
    # ...
    primary_host_pool = tree_cache.cache_controller.mem_pool_host
    host_pool_kwargs = dict(
        # 与 build_full_draft_pools 一致，使用 logical_size 保证容量覆盖逻辑索引空间
        host_to_device_ratio=primary_host_pool.logical_size / pool.size,
        host_size=0,
        page_size=page_size,
        layout=server_args.hicache_mem_layout,
        allocator_type=server_args.hicache_storage_backend,
        pool_label="draft",
    )
    # ...
```

评论区精华

Review 中 ispobock 建议在 hybrid_pool_assembler.py 中添加注释解释为什么使用 logical_size，作者已采纳并在第二 commit 中加入注释。无其他争议性问题。

- 使用 logical_size 的原因注释 (documentation): 作者采纳，在第二 commit 中添加了注释说明 DCP x DSpark draft KV 是复制的，跨越虚拟 loc space，因此匹配 logical_size。

风险与影响

- 风险：主要风险在于修改了 host pool 尺寸的推导方式，可能影响非 DCP 场景下的容量计算，但 logical_size 在非 DCP 下应等于物理 size，因此回归风险较低。此外，放宽了 server 参数校验，其他 speculative 算法仍被禁止，不会引入额外支持。需注意该修改依赖 logical_size 属性存在，若某些 pool 未定义可能报错，但从上下文看该属性是存在的。
- 影响：影响 HiCache + DCP + DSpark 组合的用户，使其能够正常启用该功能并避免潜在的内存越界错误。对非 DCP 或非 DSpark 用户无影响。团队内影响为主，因为该功能仍处于实验阶段。
- 风险标记：核心路径变更，容量计算改动，参数校验放宽

关联脉络

- PR #35840 Add PD test for inkling with mxfp8 KV: 同时修改了 kv_cache_builder.py 和 schedule_batch.py，涉及 HiCache 相关改动。
- PR #35957 Fix recurrent state loss on decode retraction: 修改了相同的 kv_cache_builder.py 和 mem_cache 相关文件，与 HiCache 路径有关。