PR #24984 完整报告

sgl-project/sglang

[HiCache] feat: support draft offload for mooncake

合并时间：2026-06-03 12:42

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24984

执行摘要

- 一句话：为 Mooncake 等后端添加 draft KV 卸载支持
- 推荐动作：该 PR 实现了关键的 draft KV 卸载功能，设计较稳健，建议精读 cache_controller.py 中的 _maybe_register_draft_with_storage 和 _draft_page_set_v2 函数以理解零拷贝集成模式。测试重构思路也值得参考。

功能与动机

关联 Issue #24956 报告在使用 GLM5 / DeepSeek V3.2 + HiCache + MTP 时触发 assert target_locations is not None 崩溃，根本原因在于 draft KV 无法正确卸载到 Mooncake 存储。PR 旨在通过支持 draft KV 卸载来修复该问题，并在重启后保持推测解码性能。

实现拆解

实现步骤

1. 引入 draft I/O 函数指针：在 CacheController.__init__中添加 draft_page_get_func 和 draft_page_set_func，初始均为 None。
2. 实现专用 I/O 路径：新增 _draft_page_set_v2 / _draft_page_get_v2（零拷贝多池路径）和 _draft_page_set_generic / _draft_page_get_generic（通用路径）。
3. 条件注册：_maybe_register_draft_with_storage 根据 storage_backend_type 选择路径：mooncake 注册 v2 并要求 should_split_heads 为 False；hf3fs 等暂不支持，走通用路径。
4. 存储后端适配：mooncake_store.py 中扩展 register_mem_host_pool_v2 以处理 PoolName.DRAFT，并修改 _get_hybrid_page_component_keys 按 draft 池的 MLA/MHA 类型生成独立后缀。
5. 枚举扩展：hicache_storage.py 新增 PoolName.DRAFT，确保键空间隔离。
6. 测试重构：提取 HiCacheSpecStorageMixin 公共基类，File 和 Mooncake 测试继承它，大幅减少复制代码。
7. 配套调整：scheduler.py 中跳过 HybridLinearKVPool 的 draft 注册（暂不支持）。

关键文件：

- python/sglang/srt/managers/cache_controller.py（模块 缓存控制；类别 source；类型 core-logic；符号 _maybe_register_draft_with_storage, _draft_page_set_v2,

_draft_page_get_v2, _draft_page_set_generic）：核心变更文件，新增 draft I/O 函数指针、v2 零拷贝路径及后端选择逻辑。

- python/sglang/srt/mem_cache/storage/mooncake_store/mooncake_store.py（模块 存储后端；类别 source；类型 dependency-wiring；符号 register_mem_host_pool_v2, _get_hybrid_page_component_keys）：修改了月亮蛋糕存储后端的 pool 注册和键生成，以支持 draft 池。
- python/sglang/srt/mem_cache/hicache_storage.py（模块 存储定义；类别 source；类型 data-contract；符号 PoolName）：新增 PoolName.DRAFT 枚举。
- python/sglang/test/hicache_spec_storage_common.py（模块 测试公用；类别 test；类型 test-coverage；符号 HiCacheSpecStorageMixin, _get_spec_server_args, _launch_spec_server, _restart_spec_server）：新增公共测试基类，消除了两个测试文件的大量重复。
- test/registered/hicache/test_hicache_spec_mooncake_storage.py（模块 Mooncake 测试；类别 test；类型 test-coverage；符号 TestHiCacheSpecMooncakeStorage, setUpClass, tearDownClass, _start_mooncake_store_service）：新的 Mooncake 端点集成测试。
- test/registered/hicache/test_hicache_spec_file_storage.py（模块 文件存储测试；类别 test；类型 test-coverage；符号 TestHiCacheSpecFileStorage, _get_spec_server_env, _count_file_storage_pages, _wait_for_file_storage_pages）：重构为使用公共基类，大幅缩减代码。

关键符号：_maybe_register_draft_with_storage, _draft_page_set_v2, _draft_page_get_v2, _draft_page_set_generic, _draft_page_get_generic, register_mem_host_pool_v2, _get_hybrid_page_component_keys, HiCacheSpecStorageMixin._get_spec_server_args, HiCacheSpecStorageMixin._launch_spec_server

关键源码片段

python/sglang/srt/managers/cache_controller.py

核心变更文件，新增 draft I/O 函数指针、v2 零拷贝路径及后端选择逻辑。

```
def _maybe_register_draft_with_storage(self) -> None:
    """Wire draft host pool to the storage backend.

    # 根据后端类型选择 draft I/O 实现路径
    # - mooncake: 使用 v2 零拷贝多池路径（需注册 draft 主机池）
    # - hf3fs / eic / nixl / simm: 尚不支持，跳过
    # - 其他后端: 使用通用路径（_draft_page_get/set_generic）
    """
    # 先重置函数指针，确保后续调用不会误用旧路径
    self.draft_page_get_func = None
    self.draft_page_set_func = None

    # 只有同时启用了 draft 和 storage 才进行注册
```

```
    if not self.has_draft or not self.enable_storage:
        return

    backend = self.storage_backend_type

    # ---- mooncake 零拷贝 v2 路径 ----
    if backend == "mooncake":
        # should_split_heads 下 suffix 生成尚不兼容，暂时禁用
        if self.storage_config.should_split_heads:
            logger.warning(
                "HiCache draft L3 disabled: should_split_heads not yet "
                "supported on the mooncake v2 path."
            )
            return
        # 注册 draft 主机池到 mooncake store，使用 PoolName.DRAFT 区分键前缀
        self.storage_backend.register_mem_host_pool_v2(
            self.mem_pool_host_draft, PoolName.DRAFT
        )
        self.draft_page_get_func = self._draft_page_get_v2
        self.draft_page_set_func = self._draft_page_set_v2
        return

    # ---- 其他零拷贝后端暂不支持 ----
    if backend in {"hf3fs", "eic", "nixl", "simm"}:
        logger.warning(
            "HiCache draft L3 disabled: backend %s does not yet support "
            "draft pool registration.",
            backend,
        )
        return

    # ---- 通用后端路径（非零拷贝） ----
    self.draft_page_get_func = self._draft_page_get_generic
    self.draft_page_set_func = self._draft_page_set_generic
```

评论区精华

Review 中主要讨论了以下设计权衡：

- MooncakeStore DRAFT 注册的特殊处理：Gemini 建议统一混合池处理，避免直接调用 kv_buffer，但目前仅支持标准 KV 池且未来可优化。
- should_split_heads 检查应泛化：Gemini 建议将 mooncake 专有检查扩展到所有 v2 后端，作者在最终实现中已为 hf3fs 等后端添加了警告跳过。
- HF3FS v2 路径不兼容：Codex 指出 HF3FS v2 的页面大小与 draft 池的数据布局不匹配，作者已将其排除在 v2 之外。
- HybridLinear 池的 draft 注册：Codex 警告跳过 HybridLinear 会导致 draft 缓存丢失，当前策略是跳过并打日志，等待后续支持。

- draft 键前缀 'd:' 的冲突风险：stmatengss 建议使用更强标识的魔数，作者认为 d: 配合 PoolName.DRAFT 后缀已足够区分。
- draft_io_mode 命名改进：huangtingwei 建议枚举合理化，作者改为函数指针方案，获得 reviewer 认可。
  - MooncakeStore DRAFT 注册对混合架构的兼容性 (design): 当前仅支持标准 KV 池，混合架构尚不投入生产，暂用直接访问。未来可重构。
  - should_split_heads 检查应泛化到所有 v2 后端 (correctness): 作者在最终实现中为 hf3fs 等后端添加了警告并跳过 v2，统一了行为。
  - HF3FS v2 路径与 MHA draft 池不兼容 (correctness): 将 hf3fs 排除在 v2 路径之外，回退到通用路径。
  - HybridLinear draft 池注册被跳过导致缓存丢失 (correctness): 当前策略是跳过并记录告警，等待后续版本支持混合架构的 draft 存储。
  - draft 键前缀 'd:' 的冲突风险 (design): 作者回复 'd:' 加上 PoolName.DRAFT 后缀已足够区分目标键，无需变更。
  - draft_io_mode 命名改进为函数指针 (style): 作者重构为 draft_page_get_func / draft_page_set_func 函数指针，获 reviewer 认可。

风险与影响

- 风险：
  1. (cache_controller.py) draft I/O 路径选择依赖 storage_backend_type，若新增后端未在 _maybe_register_draft_with_storage 中显式处理，会走通用路径，但通用路径未使用零拷贝，可能性能降级。
  2. (mooncake_store.py) register_mem_host_pool_v2 对 DRAFT 的直接 kv_buffer 假设可能对混合架构（如 Mamba）不兼容，但当前没有此类 draft 模型投入生产。
  3. (scheduler.py) 跳过 HybridLinearKVPool 的 draft 注册意味着某些模型（如 DeepSeek V3.2 MLA）的 draft cache 完全无法卸载，影响范围有限但需文档说明。
  4. 缺乏对大 batch 或多并发请求的场景测试，可能暴露竞态条件。 - 影响：用户影响：使用 HiCache + 推测解码的用户在服务器重启后可保持 draft KV 缓存，避免重新生成，减少首次响应延迟。Mooncake 后端用户受益最大。系统影响：新增约 600 行代码，核心路径复杂度增加，但由于使用函数指针，零拷贝路径与通用路径隔离良好。团队影响：降低了后续添加新存储后端（如 hf3fs、nixl）时支持 draft 的门槛，只需在 _maybe_register_draft_with_storage 中添加分支即可。
- 风险标记：核心路径变更，缺少覆盖性测试，兼容性限制，后端扩展性

关联脉络

- 暂无明显关联 PR