PR #32162 完整报告

sgl-project/sglang

[HiSparse] Support hisparse multi-step swap io kernel

合并时间：2026-08-25 08:29

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32162

执行摘要

- 一句话：新增多步 HiSparse swap 内核，一次 gather/commit 完成 MTP 交换
- 推荐动作：值得精读。本 PR 展示了如何通过双内核 + PDL 设计将多步交换合并为一次启动，并包含跨步骤去重与近似缓存接纳的权衡，对推理引擎开发者有参考价值。同时需关注其后续集成进展与边界测试补强。

功能与动机

PR body 明确指出：现有 HiSparse swap 路径按顺序处理 speculative decoding 步骤，对四步 MTP 请求需要四次独立的缓存查找、swap 传递和缓存状态更新，且无法自然去重多个步骤共享的 miss。本 PR 的目标是“treats all MTP top-k tensors as one working set and resolves their device locations in one gather/commit launch pair”，同时保持当前调用的数据精确性，仅在超容量时让缓存接纳策略变为近似。

实现拆解

1. 新增多步 swap CUDA 内核（python/sglang/kernels/jit/csrc/kvcacheio/hisparse_spec.cuh）：实现 gather 与 commit 两个内核。gather 内核扁平化 [batch, steps, top_k]，执行 extra-page 查找、hot-cache 哈希查找、跨步骤 miss 去重，并把每个唯一 miss 复制到 scratch 设备位置；commit 内核按请求更新 CLOCK 缓存、轮转 scratch 与 hot 位置、更新 token-to-slot 哈希表并解析溢出 miss。双内核结构提供 grid 级同步，使 commit 能安全消费最终 miss 计数与 scratch 哈希表；并在 Hopper 上启用 Programmatic Dependent Launch（PDL）减少启动开销。
2. Python 封装与 JIT 接入（python/sglang/kernels/ops/kvcache/hisparse.py）：新增 HiSparseSpecState NamedTuple 持有 cache_index、cache_policy、scratch_locs、scratch_state 等持久状态；_jit_spec_module 带 cache_once 装饰，将 block_size、num_top_k、hot_buffer_size、item_size_bytes、num_steps、record_miss_plan 及 is_arch_support_pdl() 编译为模板参数；入口函数 load_cache_to_device_buffer_spec_mla 校验步数范围（2-4），并支持可选的 miss 计划输出，以便与 copy_cache_planned_mla 的预取重放路径兼容。
3. 目录整理：将原 python/sglang/kernels/jit/csrc/hisparse.cuh 移动到 kvcacheio/hisparse.cuh，与新增的 hisparse_spec.cuh 隔离，保持单步内核的既有引用不变。

4. 测试与基准：新增 test/registered/jit/test_hisparse_spec.py 覆盖跨步骤重复 miss 去重与完整条目复制、782 个跨步骤唯一 miss 复制正确性；新增 test/registered/jit/benchmark/bench_hisparse_spec.py 用 CUDA Graph 对比四步多步 swap 与顺序 LRU，并注册到 CI（base-b-kernel-unit、base-b-kernel-benchmark）。
5. 演进与重构：commit 历史显示经过状态张量打包、JIT 内核精简迁移、与 IO 预取内核的兼容调整等多次迭代，最终形成当前接口。

关键文件：

- python/sglang/kernels/ops/kvcache/hisparse.py（模块 内核封装；类别 infra；类型 core-logic；符号 HiSparseSpecState, _jit_spec_module, load_cache_to_device_buffer_spec_mla）：新增 HiSparseSpecState 状态容器与 load_cache_to_device_buffer_spec_mla 入口函数，负责参数校验、JIT 模块缓存与多步 swap 内核的调用，是 Python 侧的核心封装。
- python/sglang/kernels/jit/csrc/kvcacheio/hisparse_spec.cuh（模块 内核实现；类别 source；类型 core-logic；符号 SpecCacheState, SpecMissWorkspace, PackedRingState）：新增的多步 swap 内核实现，包含 gather 与 commit 两个内核以及 CLOCK 缓存状态机，是本 PR 性能提升的核心。
- test/registered/jit/test_hisparse_spec.py（模块 单元测试；类别 test；类型 test-coverage；符号 _SwapState, _make_cache_index, _make_state, _run_swap）：正确性测试，覆盖跨步骤去重与完整条目复制场景，验证多步 swap 内核的行为。
- test/registered/jit/benchmark/bench_hisparse_spec.py（模块 基准测试；类别 test；类型 test-coverage；符号 _BenchmarkState, _make_top_k_tokens, _make_cache_index, _build_state）：基准测试，用 CUDA Graph 对比四步多步 swap 与顺序 LRU，量化性能提升并注册到 CI。
- python/sglang/kernels/jit/csrc/kvcacheio/hisparse.cuh（模块 内核实现；类别 other；类型 rename-or-move）：文件移动，将原 hisparse.cuh 从 csrc 移至 kvcacheio，与新增的多步内核文件隔离，保持单步内核引用不变。

关键符号：load_cache_to_device_buffer_spec_mla, _jit_spec_module, HiSparseSpecState

关键源码片段

python/sglang/kernels/ops/kvcache/hisparse.py

新增 HiSparseSpecState 状态容器与 load_cache_to_device_buffer_spec_mla 入口函数，负责参数校验、JIT 模块缓存与多步 swap 内核的调用，是 Python 侧的核心封装。

```
class HiSparseSpecState(NamedTuple):
    # 持久缓存状态与可复用的 miss 工作区：
    # cache_index 保存两个 int64 哈希 bank，形状为 [num_requests, 2, hash_size]；
    # cache_policy 使用控制平面行保存打包的 CLOCK 状态，随后是每个请求的引用 epoch 行：
    # [1 + num_requests, hot_buffer_size]；
    # scratch_locs 和 scratch_state 保存所有层共享的可复用 miss 位置、计数器和元数据。
    cache_index: torch.Tensor
    cache_policy: torch.Tensor
    scratch_locs: torch.Tensor
```

```
    scratch_state: torch.Tensor


@cache_once
def _jit_spec_module(
    item_size_bytes: int,
    block_size: int,
    num_top_k: int,
    hot_buffer_size: int,
    num_steps: int,
    record_miss_plan: bool,
) -> Module:
    # 将内核参数全部固化进模板参数，JIT 编译一次后按缓存复用
    template_args = make_cpp_args(
        block_size,
        num_top_k,
        hot_buffer_size,
        item_size_bytes,
        num_steps,
        record_miss_plan,
        is_arch_support_pdl(), # Hopper 上启用 Programmatic Dependent Launch
    )
    return load_jit(
        "hisparse_spec",
        *template_args,
        cuda_files=["kvcacheio/hisparse_spec.cuh"],
        cuda_wrappers=[
            (
                "load_cache_to_device_buffer_spec",
                f"load_cache_to_device_buffer_spec<{template_args}>",
            )
        ],
    )
```

评论区精华

本 PR 没有内联 review 评论，审核直接通过。唯一的实质性评论来自合并者 xiezhq-hermann，他在合并前要求“Let's merge this after this PR so we have a clean line of changes: https://github.com/sgl-project/sglang/pull/34329. sorry for the delay rebasing”，即先合入 PR#34329 再合并本 PR，以避免变更线交错。作者随后进行了 rebase 并触发 CI，最终合并。

- 合并顺序与 rebase 协调 (other): 作者按建议进行了 rebase 并触发 CI，最终由 xiezhq-hermann 合入。

风险与影响

- 风险：新内核尚未接入 HiSparse 调度器与 IO 预取路径，目前是独立可用内核；后续集成时若接口不一致可能引入回归，需要保持 miss_src/miss_dst/miss_count 协议兼容。当唯一 miss 并集超过 hot-cache 容量时，缓存接纳策略变为近似（优先保留最近投机步骤），可能影响后续投机步骤的命中率与性能，但不影响当前调用的数据正确性。内核使用了 PDL，该特性依赖 Hopper 架构，虽然代码中有 is_arch_support_pdl() 判断，但非 Hopper 回退路径的性能与正确性缺乏专项测试。新增约 1113 行 CUDA 代码，测试目前只覆盖 782 个唯一 miss 和 batch 为 1 的场景，多请求并发、hot_buffer_size 边界（如 32768 上限）、步数 2/3 等情况需要更多验证。文件移动 hisparse.cuh 可能影响依赖该路径的构建脚本，需确认编译产物及包含关系。
- 影响：对使用 HiSparse 的 MTP 推理场景有显著性能提升（H20 batch-64 延迟降低 48.2%），用户可预期更低的投机解码延迟。系统层面，该 PR 只新增内核和测试，未改动调度器或协调器，因此对现有主路径影响有限。团队方面，为未来 IO 预取和协调器集成奠定了基础，但端到端收益仍需后续接入才能兑现。
- 风险标记：核心内核新增未集成主路径，近似缓存接纳策略，PDL 架构依赖，测试边界覆盖不足

关联脉络

- PR #34329 Predecessor PR required for clean merge (title not provided): 合并者明确要求先合入 PR#34329 再合并本 PR，以避免变更线交错；两个 PR 在文件或功能上存在先后依赖。