PR #34613 完整报告

sgl-project/sglang

feat(unified-memory): read unified pool from attention backends fa3/flashinfer/trtllm_mha/flashmla

合并时间：2026-08-31 14:58

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34613

执行摘要

- 一句话：统一内存池读路径迁移至全部注意力后端，删除钩子并放宽后端白名单
- 推荐动作：值得精读。这是 unified-memory 存储系列中架构收敛最彻底的一步：1）canonical read table + 生产点翻译的所有权模型，是理解 SGLang 未来 KV 索引架构的必修课；2）enforcement 扫描测试（test_kv_translate_ownership.py）展示了如何用源码级测试固化架构边界，防止设计被静默回退；3）prefix-only 填充纪律（保留后端 -1 尾哨兵）与 CUDA graph 捕获稳定 buffer 的处理是 kernel/backend 开发的实战范例。建议阅读顺序：先看 PR body 的 stack 说明和表格，再读 flashattention_backend.py 的 eager/capture 双路径接线，最后看 test_kv_translate_ownership.py 与 test_unified_mla_block_table.py 如何验证身份等价与字节级等价。

功能与动机

PR body 明确说明：在读路径收敛到单一 choke point 后，剩余注意力后端应能直接读取统一内存池，而不必各自携带 id-space 逻辑。此前 unified pool 只支持 Triton，而用户实际主要运行 fa3、FlashInfer、trtllm 等后端，因此本次迁移是“把统一池从 Triton-only 提升到用户真正运行的后端”的必要步骤。PR 还强调设计动机：虚拟 id 与物理 id 值域重叠，后端忘记翻译或重复翻译都不会崩溃，只会读错行，因此必须把翻译所有权收敛到恰好两处（读：KVIndexTranslator；写：ForwardBatch rebind）并用扫描强制。

实现拆解

1. 读路径迁移到 KVIndexTranslator canonical table：
   - flashattention_backend.py：构造函数改用 model_runner.kv_index_translator，eager 路径通过 index_table_for_batch(forward_batch) 取得每批次的 KVIndexTable，CUDA graph 捕获路径通过 build_index_table(... into=self.kv_read_tables) 写入捕获稳定的 buffer，normal_decode_set_metadata 通过 src_is_read_table=True 消费 canonical 行；对 unified + local attention、prefill-aware SWA 两种无法支持的组合主动 assert 拒绝。
   - flashinfer_backend.py / flashinfer_mla_backend.py：decode/prefill 各 updater 改为从 kv_view.ids + kv_view.row_ids 构建 KV indices（create_flashinfer_kv_indices_triton 传 ENTRY_PAGE_SIZE=kv_view.entry_page_size），删除了原先 post-gather 的 translate_kv_loc_for_kernel in-place 翻译；SWA 写路径

统一走 kv_index_translator.sliding_window_write_loc_for。

- trtllm_mla_backend.py：_create_block_kv_indices 与 _apply_cuda_graph_metadata 在 kv_index_translator.is_translating 时改调 fill_read_table 填充 padded block table 的 live prefix（prefix-only，保留后端自己的 -1 尾哨兵）；非 unified 路径保持原 create_flashmla_kv_indices_triton 调用。
- trtllm_mha_backend.py：CUDA graph 捕获时把 page table 绑定到 translator 的 capture-stable read-table buffer，graph 内 builder 跳过 page-table 与 SWA write-loc 工作。

2. MLA 模型门读索引在生产点翻译：forward_batch_deepseek_mha_mixin.py 中 fetch_mha_one_shot_kv_indices 等 req_to_token 派生索引在产生处通过 translator 翻译并缓存，memory_pool.py 的 get_mla_kv_buffer 门变为透传（不再翻译），删除 HybridLinearKVPool._full_translate 钩子。
3. 删除 unified_mem_hooks.py 并用 enforcement 扫描替代：删除 UnifiedMLAHooks / unified_mla_hooks 探测 shim，新增 test_kv_translate_ownership.py 源码扫描测试，用正则禁止 layers/attention/ 下出现 .translate_kv_loc 调用、getattr(...translate_kv_loc...) 探测或对已删除模块的 import。
4. 放宽后端白名单：server_args.py 的 _handle_page_major_kv_layout 从单臂扩为双臂：unified MLA 允许 {triton, fa3, trtllm_mla, flashinfer, cutedsl_mla, tokenspeed_mla, flashmla}，unified MHA/SWA（uniform rows）允许 {triton, fa3, fa4, flashinfer, trtllm_mha}；新增 flashmla 接线（flashmla_backend.py 支持 ps=64 snap）。同时新增对非对称 K/V（如 MiMoV2）统一池的启动拒绝，以及不带 unified memory 的 page-major 直接拒绝。
5. 测试与配套：test_unified_mla_block_table.py 改为覆盖 canonical build_kv_read_table 路线与静态 stripped kernel 的字节级等价；新增 TestReadRailTranslatesAtProduction（生产点翻译恰好一次并缓存）、TestUnifiedTranslateBanned（所有权扫描）、page-major 后端矩阵单测与 test_page_major_gpt_oss.py / test_page_major_qwen_hybrid.py 的真实启动 e2e 单元格；文档注释统一术语（choke point → translator，canonical → read table，rail → write loc）。

关键文件：

- python/sglang/srt/layers/attention/unified_mem_hooks.py（模块 注意力后端；类别 source；类型 deletion；符号 UnifiedMLAHooks, unified_mla_hooks）：整个文件被删除（-72 行）：这是此前 fa3/flashinfer_mla/trtllm_mla 三族后端共享的 allocator 探测 shim，其 v2p_page_table / translate_kv_loc_for_kernel / kernel_page_multiplier 语义被 KVIndexTranslator 的 isinstance 探测取代。删除它是本次架构收敛的标志性动作，也是 enforcement 扫描的核心保护对象。
- python/sglang/srt/layers/attention/trtllm_mla_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 _resolve_fused_write_loc, _create_block_kv_indices, _apply_cuda_graph_metadata）：trtllm_mla 及其子类（cutedsl_mla、tokenspeed_mla）是 MLA 主力后端。本文件将 block table 构建从自带 v2p_ptr/PAGE_MULT 的 create_flashmla_kv_indices_triton 切换到 translator 的 fill_read_table（unified 时），并新增 _resolve_fused_write_loc 处理 fp8 融合写路径，是后端迁移的代表性实现。

- test/registered/unit/layers/attention/test_kv_translate_ownership.py（模块 所有权扫描；类别 test；类型 test-coverage；符号 _iter_sources, TestUnifiedTranslateBanned, test_no_unified_translate_calls, test_no_translate_capability_probing）：新增的架构边界 enforcement 测试（+83 行）：用源码扫描禁止 layers/attention/ 下任何后端调用 unified translate 面、探测 translate 能力或 import 已删除的 hooks 模块。它把"翻译所有权只属于 KVIndexTranslator（读）与 ForwardBatch rebind（写）"变成可自动验证的约束，是防止设计回退的关键守门测试。

关键符号：unified_mla_hooks, _fill_block_table, _fill_block_table_static, _resolve_fused_write_loc, _create_block_kv_indices, fill_read_table, build_index_table, index_table_for_batch, make_capture_tables, sliding_window_write_loc_for, rebind_write_loc, fetch_mha_one_shot_kv_indices, normal_decode_set_metadata, _handle_page_major_kv_layout, translate_full_attn_ids, get_mla_kv_buffer

关键源码片段

python/sglang/srt/layers/attention/unified_mem_hooks.py

整个文件被删除（-72 行）：这是此前 fa3/flashinfer_mla/trtllm_mla 三族后端共享的 allocator 探测 shim，其 v2p_page_table / translate_kv_loc_for_kernel / kernel_page_multiplier 语义被 KVIndexTranslator 的 isinstance 探测取代。删除它是本次架构收敛的标志性动作，也是 enforcement 扫描的核心保护对象。

```
# python/sglang/srt/layers/attention/unified_mem_hooks.py （本 PR 删除，base 版本节选）

class UnifiedMLAHooks(msgspec.Struct, frozen=True):
    """Dense-view hooks for one KV allocator.

    All-``None``/1/``False`` for the statically-partitioned pool, where
    ``req_to_token`` already holds physical ids and no translation is needed.
    """

    # Page-level virtual->physical table, gathered through by block-table kernels.
    v2p_page_table: Optional[torch.Tensor]
    # Virtual token id -> DENSE kernel-facing id (tombstones clamped to the sink).
    translate_kv_loc_for_kernel: Optional[Callable[..., torch.Tensor]]
    # Dense page stride scale (= number of full-attention MLA layers).
    kernel_page_multiplier: int
    enabled: bool


_STATIC_POOL = UnifiedMLAHooks(
    v2p_page_table=None,
    translate_kv_loc_for_kernel=None,
    kernel_page_multiplier=1,
    enabled=False,
)

def unified_mla_hooks(allocator) -> UnifiedMLAHooks:
```

```
    """Probe ``allocator`` for the unified-pool per-layer-view hooks.

    Detection keys on the v2p table, NOT on ``kernel_page_multiplier > 1``: a
    rank owning exactly ONE full-attention layer has multiplier 1 while its
    ``req_to_token`` is still virtual. There the kernel-facing id collapses onto the
    physical id, so the v2p gather alone is the whole translation.
    """
    v2p = getattr(allocator, "full_v2p_page_table", None)
    if v2p is None:
        return _STATIC_POOL
    return UnifiedMLAHooks(
        v2p_page_table=v2p,
        translate_kv_loc_for_kernel=getattr(
            allocator, "translate_kv_loc_for_kernel", None
        ),
        kernel_page_multiplier=getattr(allocator, "kernel_page_multiplier", 1),
        enabled=True,
    )
```

python/sglang/srt/layers/attention/trtllm_mla_backend.py

trtllm_mla 及其子类（cutedsl_mla、tokenspeed_mla）是 MLA 主力后端。本文件将 block table 构建从自带 v2p_ptr/PAGE_MULT 的 create_flashmla_kv_indices_triton 切换到 translator 的 fill_read_table（unified 时），并新增 _resolve_fused_write_loc 处理 fp8 融合写路径，是后端迁移的代表性实现。

```
# python/sglang/srt/layers/attention/trtllm_mla_backend.py （head 版本节选）

def _create_block_kv_indices(
    self,
    batch_size: int,
    max_blocks: int,
    req_pool_indices: torch.Tensor,
    seq_lens: torch.Tensor,
    device: torch.device,
) -> torch.Tensor:
    """Create block KV indices tensor using Triton kernel."""
    block_kv_indices = torch.full(
        (batch_size, max_blocks), -1, dtype=torch.int32, device=device
    )

    if self.kv_index_translator.is_translating:
        # Unified pool：canonical read table 按行填充 live prefix，
        # 后端自己的 -1 / stale 尾哨兵得以保留（prefix-only 纪律）。
        self.kv_index_translator.fill_read_table(
            out=block_kv_indices,
            req_pool_indices=req_pool_indices,
            seq_lens=seq_lens,
        )
    else:
```

```
        # 静态池：沿用原始 flashmla 内核，v2p 参数已剥离（id-space-free）。
        create_flashmla_kv_indices_triton[
            (
                batch_size,
                get_num_kv_index_blocks_flashmla(max_blocks, self.page_size),
            )
        ](
            self.req_to_token,
            req_pool_indices,
            seq_lens,
            None,
            block_kv_indices,
            self.req_to_token.stride(0),
            max_blocks,
            PAGED_SIZE=self.page_size,
        )

    return block_kv_indices


def _resolve_fused_write_loc(
    self, forward_batch: ForwardBatch
) -> Optional[torch.Tensor]:
    """Write loc for the fused fp8 KV scatter, or None when this batch is
    not covered by it.

    Captured decode refills ``_decode_kernel_loc`` out of the graph, and the
    captured kernel must read that buffer. Eager decode on a unified pool
    has no such buffer, and the caller falls back to the unfused path.
    """
    if self._decode_kernel_loc is not None:
        return self._decode_kernel_loc
    return (
        None
        if self.kv_index_translator.is_translating
        else forward_batch.out_cache_loc
    )
```

评论区精华

本 PR 的 review 评论为 0 条，issue 评论也仅有作者触发的 CI 重跑命令 /tag-and-rerun-ci extra。PR body 中自述了关键设计权衡：

- PR 曾被拆分：原先同时包含 read-path refactor 与 backend 迁移，现按 #34602 → #35245 → #35247 → 本 PR 的 stack 顺序拆分，便于独立 review 机制与后端工作。
- 准确性表格注明“基于上一版 stack 测得，系列已按 review 重构并 rebase，正在重新验证”——说明 review 曾推动结构性返工（12 个 commit 中后 8 个为本 PR，前 4 个来自 stacked PR）。

- enforcement 扫描的设计意图：虚拟与物理 id 值域重叠导致“忘记翻译”与“重复翻译”都是静默错误，扫描使两种失败模式不可表示（unrepresentable）。扫描明确排除 allocator 内部 v2p 实现、PD 传输面 translate_kv_indices_for_transfer 与静态 SWA 的 full→swa 映射。
- PR 拆分与 stack 结构 (design): 采用 stack 拆分，机制（translator）与 per-backend 迁移可独立审查；本 PR 的 body 与 commit 均按此结构组织。
- 准确性与时延数据的时效性 (question): 数据暂为上一版测量结果，合并前需以重新验证后的数据为准；评论区无后续更新。
- enforcement 扫描的边界与排除项 (design): 扫描范围明确写入测试 docstring，排除项被显式陈述，未来新增调用点须通过测试审查。
- CUDA graph 捕获时序下的零填充 sink 约定 (correctness): 以零填充捕获稳定 buffer，重放前在图外 refill；该约定被多个测试用例固定。

风险与影响

- 风险：
  1. 核心前向路径大面积改动：fa3/fa4、FlashInfer、trtllm_mla/mha 均为线上主力后端，读路径全部改走 translator 的 canonical table。虽然 PR 声称非 unified 池字节级等价，但改动面达 31 个文件、±2000 行，且 CI 最终状态为红色（PR Test、Extra、AMD ROCm 三栏均为 ）），合并前需确认失败用例与本次改动无关或已修复。
  2. CUDA graph 捕获 / 重放语义风险：trtllm_mha 与 flashinfer 的 captured 路径改为读取图外刷新的 kv_read_tables 捕获稳定 buffer；fa3 的 normal_decode_set_metadata 新增 src_is_read_table=True 分支，若 capture/replay 时序错误（如 replay 前未刷新表）会静默读旧 KV。trtllm_mha 新增的 test_skip_page_table_updates_seqlens_only 正是针对此类回归。
  3. enforcement 扫描的维护成本：test_kv_translate_ownership.py 用正则扫描源码，任何后端新增合法的 translate 调用都会被测试阻断；虽然 docstring 声明了排除项，但正则 \.translate_kv_loc(_kernel_id)?\(可能误伤未来合法命名（如注释、字符串）。
  4. SWA 写路径语义变化：sliding_window_write_loc_for 取代了静态 translate_loc_from_full_to_swa，capture 批次（未经过 init_new）统一以零填充（page-0 sink），依赖该约定在重放时正确 refill；gpt-oss 这类 SWA 模型（测试中特地未加 flashinfer 单元格，因其使用 attention sinks）需要额外关注。
  5. allowlist 收紧的启动回归：不带 unified memory 的 --enable-page-major-kv-layout 现在被直接拒绝（包括 Triton），以及非对称 K/V 模型禁止 unified pool——对依赖旧行为的部署是 breaking change。 - 影响：用户影响：启用 --enable-unified-memory 的 MLA 与 MHA/SWA 模型不再被锁定在 Triton 后端，可在 fa3/fa4、FlashInfer、trtllm_mha、flashmla 等后端上运行，这对 Blackwell（fa4）、DeepSeek MLA 系（Kimi-Linear、DeepSeek V4）用户是直接收益；同时 page-major 布局的启动约束收紧，不带 unified memory 的 page-major 用户会收到拒绝而非静默降级。 系统影响：KV id 翻译所有权收敛到 translator 与 ForwardBatch rebind 两处，后端不再持有 v2p 知识，降低后续新增后端接入 unified pool 的成本，也消除了“重复 / 遗漏翻译”这类静默错误类别。 团队影响：本 PR 与 #35245 等构成一个 4-PR stack，设计决策（canonical

read table、所有权扫描、prefix-only 填充纪律）为后续 HiCache 外部缓存链路（#37151 的 unified cache linker）提供了统一的索引抽象基础。影响程度为高——这是 unified-memory 从 Triton-only 走向生产后端的里程碑步骤。

- 风险标记：核心前向路径变更，CUDA graph 捕获语义，CI 最终状态未绿，allowlist 收紧属 breaking change，基准数据基于旧版本

关联脉络

- PR #35245 refactor(unified-memory): translate the KV write location once, at ForwardBatch construction: 本 PR 直接 stacking 其上的前置 refactor：写 loc 在 ForwardBatch 构造时统一 rebind，本 PR 才能让各后端读路径接入 canonical read table 并删除剩余的翻译钩子。
- PR #35247 refactor(unified-memory): canonical read-table build: PR body 声明的 stack 成员（#34602 → #35245 → #35247 → 本 PR），提供 build_kv_read_table / KVIndexTable 这一 canonical 读表机制，本 PR 的所有后端 dispatch 到它。
- PR #37170 [unified-memory] Drop the vacated 'dense' qualifier and the restating comments: 同一功能线上的后续清理 PR：本 PR 引入的术语（dense/canonical/choke point）在该 PR 中被统一为 kernel-facing，多个测试文件（test_unified_mla_block_table、test_page_major_backend_allowlist）重叠，说明该系列仍在持续演进。
- PR #37151 [Unified Cache Linker][3/N]: Add backend-independent linker core: 同一架构方向（unified memory / HiCache）的后续 PR：translator 的 canonical read table 为外部缓存链路提供了统一的索引抽象，两个 PR 同属 unified-radix-cache 演进主线。