PR #30793 完整报告

sgl-project/sglang

[Kernel] Migrate linear-attention, MiniMax-sparse and diffusion kernels to sglang.kernels (RFC #29630, Phase 2.5, 6/7)

合并时间：2026-07-15 11:21

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30793

执行摘要

- 一句话：线性注意力 /MiniMax 稀疏 / 扩散内核搬迁至 sglang.kernels
- 推荐动作：值得仔细阅读。该 PR 展示了大规模代码组织重构的最佳实践：使用逐字节搬迁保持历史可追溯性、统一注册入口、系统更新所有导入引用。对于关注代码架构和内核管理的工程师有参考价值。

功能与动机

Phase 2.5 of RFC #29630 旨在将散落在 sglang 包各处的约 280 个 Triton 内核及 CuTe DSL / TileLang 内核迁移到 sglang.kernels 包下的三个规范位置。本 PR 作为第 6/7 轮，覆盖线性注意力家族、MiniMax 稀疏操作和 multimodal_gen 中的遗留内核，包括唯一一个位于规范内核树之外的 .cu 源文件。

实现拆解

1. 线性注意力内核搬迁：将 lightning_attn.py、seg_la.py 以及 gdn_blackwell/、kda_blackwell/ 子包（CuTe DSL）从 sglang.srt.layers.attention.linear 整体移到 sglang.kernels.ops.attention.linear，保持逐字节相同。
2. MiniMax 稀疏内核搬迁：将 minimax_sparse_ops/（common/decode/prefill 共 12 个 Triton 内核）从 sglang.srt.layers.attention 移到 sglang.kernels.ops.attention.minimax_sparse。
3. 扩散 / 3D 渲染内核搬迁：将 multimodal_gen/csrc/render/ 下的 .cu 文件（hunyuan3d_rasterizer）和 cpp_extension 加载器移到 sglang.kernels.ops.diffusion.render。
4. 稀疏线性注意力内核提取：将 sparse_linear_attn.py 中的 get_block_map、mean_pool、compress_kernel、_attn_fwd 四个函数提取到新文件 sglang/kernels/ops/diffusion/sparse_linear_attn_kernels.py，原文件改为从新位置导入。
5. 导入引用重写：全仓 40+ 个文件（包括 benchmark、测试、SRT 后端）的导入路径统一指向新的 sglang.kernels 路径。
6. 注册表更新：在 ops/attention/__init__.py 中添加 linear 和 minimax_sparse 子包的导入，使新迁移的内核可通过 KernelSpec 清单发现。

关键文件：

- python/sglang/multimodal_gen/runtime/layers/attention/backends/sparse_linear_attn.py（模块 扩散注意力；类别 source；类型 core-logic；符号 get_block_map, mean_pool, compress_kernel, _attn_fwd）：核心变更：移除了 4 个 Triton 内核定义（get_block_map, mean_pool, compress_kernel, _attn_fwd），改为从新的 sglang.kernels 子包导入。这是多模态扩散模型中 Sparse Linear Attention 的后端文件，也是本 PR 中唯一有实质内容删除的文件。
- python/sglang/kernels/ops/diffusion/sparse_linear_attn_kernels.py（模块 注意力内核；类别 infra；类型 infrastructure；符号 get_block_map, mean_pool, compress_kernel, _attn_fwd）：新增目标文件，包含所有从原 sparse_linear_attn.py 提取的内核函数。作为迁移的目的地，定义了 get_block_map、mean_pool、compress_kernel 和 _attn_fwd。
- python/sglang/srt/layers/attention/linear/lightning_backend.py（模块 线性注意力；类别 source；类型 dependency-wiring）：修改导入路径：lightning_attn 和 seg_la 的导入从相对路径改为指向 sglang.kernels 下的新位置。反映了线性注意力内核的搬迁。
- benchmark/bench_linear_attention/bench_kda_prefill_cutedsl.py（模块 KDA 基准；类别 source；类型 dependency-wiring）：KDA Blackwell 基准脚本的导入路径更新，反映 kda_blackwell 子包搬迁。
- python/sglang/multimodal_gen/runtime/utils/mesh3d_utils.py（模块 3D 渲染；类别 source；类型 dependency-wiring）：唯一的 .cu 文件所在的渲染模块导入路径更新，从 multimodal_gen/csrc/render 迁移至 sglang.kernels/ops/diffusion/render。
- python/sglang/kernels/ops/attention/__init__.py（模块 内核注册；类别 infra；类型 infrastructure）：作为注意力内核的总注册表，新增了对 linear 和 minimax_sparse 子模块的导入注册，使新的内核路径可通过 KernelSpec 发现。

关键符号：get_block_map, mean_pool, compress_kernel, _attn_fwd

关键源码片段

python/sglang/multimodal_gen/runtime/layers/attention/backends/sparse_linear_attn.py

核心变更：移除了 4 个 Triton 内核定义（get_block_map, mean_pool, compress_kernel, _attn_fwd），改为从新的 sglang.kernels 子包导入。这是多模态扩散模型中 Sparse Linear Attention 的后端文件，也是本 PR 中唯一有实质内容删除的文件。

```
"""Sparse Linear Attention backend - 内核已搬迁至 sglang.kernels."""

import torch
import triton
# 注意：不再直接 import triton.language as tl，因为内核已移出

from sglang.kernels.ops.diffusion.sparse_linear_attn_kernels import (
    _attn_fwd,
    get_block_map,
)
# 原文件中定义的 get_block_map、mean_pool、compress_kernel、_attn_fwd
# 已整块搬迁至 sparse_linear_attn_kernels.py，此处仅保留导入
```

```
# 后端类保持不变，注意力计算逻辑通过导入的内核函数完成
class SparseLinearAttentionBackend(AttentionBackend):
    # ... 类实现细节未改动 ...
    pass
```

python/sglang/kernels/ops/diffusion/sparse_linear_attn_kernels.py

新增目标文件，包含所有从原 sparse_linear_attn.py 提取的内核函数。作为迁移的目的地，定义了 get_block_map、mean_pool、compress_kernel 和 _attn_fwd。

```
"""扩散模型稀疏线性注意力内核（从 sglang.multimodal_gen 迁移而来）。"""

import torch
import triton
import triton.language as tl


def get_block_map(q, k, topk_ratio, BLKQ=64, BLKK=64):
    # smooth-k 技术（来自 SageAttention）：对 k 减去均值
    arg_k = k - torch.mean(k, dim=-2, keepdim=True)
    # 对 Q 和 K 进行块均衡池化
    pooled_qblocks = mean_pool(q, BLKQ)
    pooled_kblocks = mean_pool(arg_k, BLKK)
    pooled_score = pooled_qblocks @ pooled_kblocks.transpose(-1, -2)

    K = pooled_score.shape[-1]
    topk = min(K, int(topk_ratio * K))
    lut = torch.topk(pooled_score, topk, dim=-1, sorted=False).indices
    sparse_map = torch.zeros_like(pooled_score, dtype=torch.int8)
    sparse_map.scatter_(-1, lut, 1)
    return sparse_map, lut, topk


def mean_pool(x, BLK):
    x = x.contiguous()
    B, H, L, D = x.shape
    L_BLOCKS = (L + BLK - 1) // BLK
    x_mean = torch.empty((B, H, L_BLOCKS, D), device=x.device, dtype=x.dtype)
    grid = (L_BLOCKS, B * H)
    compress_kernel[grid](x, x_mean, L, D, BLK)
    return x_mean


@triton.jit
def compress_kernel(
    X, XM, L: tl.constexpr, D: tl.constexpr, BLOCK_L: tl.constexpr,
):
    idx_l = tl.program_id(0)
    idx_bh = tl.program_id(1)
    offs_l = idx_l * BLOCK_L + tl.arange(0, BLOCK_L)
```

```
    offs_d = tl.arange(0, D)
    x_offset = idx_bh * L * D
    xm_offset = idx_bh * ((L + BLOCK_L - 1) // BLOCK_L) * D
    x = tl.load(
        X + x_offset + offs_l[:, None] * D + offs_d[None, :],
        mask=offs_l[:, None] < L,
    )
    nx = min(BLOCK_L, L - idx_l * BLOCK_L)
    x_mean = tl.sum(x, axis=0, dtype=tl.float32) / nx
    tl.store(
        XM + xm_offset + idx_l * D + offs_d,
        x_mean.to(XM.dtype.element_ty),
    )
# _attn_fwd 等其他内核在此文件后续定义，与原实现逐字节相同
```

评论区精华

无实质性讨论。PR 由 BBuf 自动化生成并直接合并，仅触发 CI 重跑。

- 暂无高价值评论线程

风险与影响

- 风险：风险较低。所有内核移动均为逐字节搬迁（git rename R100 或仅导入路径差异），不改变任何执行逻辑。主要风险包括：(1) 导入路径重写遗漏可能导致运行时错误，但已通过仓库范围的 py_compile 和测试套件验证；(2) 与同系列其他 PR 存在 init.py 追加冲突，通过多次合并 main 解决。整体回归风险很小。
- 影响：影响 44 个文件的导入路径，涉及 sglang.srt、sglang.multimodal_gen 等模块。对最终用户透明，所有功能保持不变。为后续内核版本管理奠定基础。
- 风险标记：导入路径变更，多 PR 合并冲突，零功能变更

关联脉络

- PR #30784 [Kernel] Migrate scattered quantization kernels to sglang.kernels (RFC #29630, Phase 2.5, 1/7): 同系列 Phase 2.5 的第 1 个 PR，负责量化内核迁移，与本 PR 共享迁移目标包和注册表结构。
- PR #30786 [Kernel] Migrate scattered MoE kernels to sglang.kernels (RFC #29630, Phase 2.5, 2/7): 同系列第 2 个 PR，负责 MoE 内核迁移，与本 PR 共享类似的搬迁模式和导入重写策略。
- PR #30787 [Kernel] Migrate top-level srt/layers stray kernels to sglang.kernels (RFC #29630, Phase 2.5, 3/7): 同系列第 3 个 PR，负责 srt/layers 下零散内核迁移，与本 PR 有重叠的 ops/attention/init.py 注册表变更。
- PR #30789 [Kernel] Migrate generic attention kernels to sglang.kernels (RFC #29630, Phase 2.5, 4/7): 同系列第 4 个 PR，负责通用注意力内核迁移，与本 PR 共享 ops/attention 目标路径。

- PR #30792 [Kernel] Migrate DSA + DSV4 attention kernels to sglang.kernels (RFC #29630, Phase 2.5, 5/7): 同系列第 5 个 PR，负责 DSA 和 DSV4 注意力内核迁移，与本 PR 为连续步骤，共同完成注意力内核的全面集中。