PR #43905 完整报告

vllm-project/vllm

[DSv4] Move mHC tilelang kernels & Don't use CustomOP in dsv4/nvidia

合并时间：2026-05-29 10:25

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/43905

执行摘要

- 一句话：重构 DSv4 的 mHC tilelang 内核路径，移除 CustomOp 包装
- 推荐动作：值得精读，特别是如何逐步移除 CustomOp 包装、将内核文件组织到统一位置的清理模式。设计者可以借鉴这种降低抽象层、提升可读性的重构手法。

功能与动机

清理 tilelang 内核的组织，减少对 CustomOp 包装层的依赖，使 NVIDIA 路径更直接。PR body 指出：

1) Move vllm/_tilelang_ops.py to vllm/model_executor/kernels/mhc/tilelang_kernels.py 2) Remove custom ops in deepseek_v4/nvidia/ and directly call the tilelang kernels.

实现拆解

1. 重命名并移动 tilelang ops 文件：将 vllm/_tilelang_ops.py 重命名为 vllm/model_executor/kernels/mhc/tilelang_kernels.py，保持在 mhc 子目录下，使内核路径更一致。
2. 更新 tilelang.py 中的导入与 hc_head 签名：tilelang.py 中所有从 vllm._tilelang_ops 的导入改为指向新的位置；同时将 _hc_head_fused_kernel_tilelang 函数改为 hc_head_fused_kernel_tilelang，不再接受预分配的 out 参数，而是内部创建并返回 torch.Tensor，并新增 _hc_head_fused_kernel_tilelang_fake 用于 meta device 的 shape 推理。
3. 移除 model.py 中的 CustomOp 使用：在 DeepseekV4DecoderLayer 中移除 HCHeadOp、MHCPreOp、MHCPostOp、MHCFusedPostPreOp 的导入和实例化，改为直接导入并调用 mhc_pre_tilelang、mhc_post_tilelang、mhc_fused_post_pre_tilelang 等函数；hc_pre、hc_post 等方法被内联。
4. 更新 mhc.py 中的 CustomOp 包装：HCHeadOp.forward_cuda 和 forward_hip 中不再预分配 out，而是直接调用 torch.ops.vllm.hc_head_fused_kernel_tilelang 并使用返回值。
5. 更新 mtp.py：与 model.py 类似，移除 HCHeadOp 导入，直接使用 hc_head_fused_kernel_tilelang 和 mhc_post_tilelang。
6. 测试适配：调整 test_mhc_kernels.py 中的测试用例，不再检查返回值为 None，而是验证输出 tensor 的 shape 和 dtype。

关键文件：

- vllm/model_executor/kernels/mhc/tilelang.py（模块 内核 (Kernels)；类别 source；类型 data-contract；符号 _hc_head_fused_kernel_tilelang, hc_head_fused_kernel_tilelang, _hc_head_fused_kernel_tilelang_fake）：核心内核封装文件，修改了导入路径和 hc_head 函数的签名，新增 fake impl。
- vllm/models/deepseek_v4/nvidia/model.py（模块 模型 (Models)；类别 source；类型 data-contract；符号 hc_pre, hc_post）：模型主文件，移除了对 CustomOp 类的依赖，直接调用 tilelang 函数，简化了 DecoderLayer。
- vllm/model_executor/layers/mhc.py（模块 算子 (Ops)；类别 source；类型 data-contract）：CustomOp 包装层，调整 forward_cuda/forward_hip 以使用新的 kernel 签名。
- vllm/models/deepseek_v4/nvidia/mtp.py（模块 模型 (Models)；类别 source；类型 data-contract）：MTP 模型文件，同步移除 CustomOp 使用，直接调用 tilelang 函数。
- vllm/model_executor/kernels/mhc/tilelang_kernels.py（模块 内核 (Kernels)；类别 source；类型 rename-or-move）：重命名文件，从 vllm/_tilelang_ops.py 移动过来，内容未变。
- tests/kernels/test_mhc_kernels.py（模块 测试 (Tests)；类别 test；类型 test-coverage）：测试适配新的 kernel 签名，验证返回值而非副作用。

关键符号：hc_head_fused_kernel_tilelang, mhc_pre_tilelang, mhc_post_tilelang, mhc_fused_post_pre_tilelang, HCHeadOp.forward_cuda, HCHeadOp.forward_hip, DeepseekV4DecoderLayer.forward

关键源码片段

vllm/model_executor/kernels/mhc/tilelang.py

核心内核封装文件，修改了导入路径和 hc_head 函数的签名，新增 fake impl。

```
# vllm/model_executor/kernels/mhc/tilelang.py (head version)

import torch
from vllm.utils.torch_utils import direct_register_custom_op


def hc_head_fused_kernel_tilelang(
    hs_flat: torch.Tensor,
    fn: torch.Tensor,
    hc_scale: torch.Tensor,
    hc_base: torch.Tensor,
    rms_eps: float,
    hc_eps: float,
) -> torch.Tensor:
    """
    Apply the fused hc_head kernel and return the (T, H) bf16 result.
    Previously this function received a pre-allocated `out` tensor and
    filled it in-place; now it allocates the output internally and
    returns it, eliminating the need for `mutates_args` in the custom op.
```

```
    """
    num_tokens, hc_mult, hidden_size = hs_flat.shape
    out = torch.empty(
        num_tokens, hidden_size, dtype=torch.bfloat16, device=hs_flat.device
    )
    if num_tokens == 0:
        return out
    # Import tilelang kernel from the new location
    from vllm.model_executor.kernels.mhc.tilelang_kernels import hc_head_fuse_tilelang
    hc_head_fuse_tilelang(
        hs_flat,
        fn,
        hc_scale,
        hc_base,
        out,
        hidden_size,
        rms_eps,
        hc_eps,
        hc_mult,
    )
    return out


def _hc_head_fused_kernel_tilelang_fake(
    hs_flat: torch.Tensor,
    fn: torch.Tensor,
    hc_scale: torch.Tensor,
    hc_base: torch.Tensor,
    rms_eps: float,
    hc_eps: float,
) -> torch.Tensor:
    """Fake implementation for meta device shape inference."""
    num_tokens, _, hidden_size = hs_flat.shape
    return torch.empty(
        num_tokens, hidden_size, dtype=torch.bfloat16, device=hs_flat.device
    )

# Register the custom op without mutates_args because hc_head_fused_kernel_tilelang
# returns a new tensor rather than writing to a pre-allocated one.
direct_register_custom_op(
    op_name="hc_head_fused_kernel_tilelang",
    op_func=hc_head_fused_kernel_tilelang,
    mutates_args=[],
    fake_impl=_hc_head_fused_kernel_tilelang_fake,
)
```

vllm/models/deepseek_v4/nvidia/model.py

模型主文件，移除了对 CustomOp 类的依赖，直接调用 tilelang 函数，简化了 DecoderLayer。

```
# vllm/models/deepseek_v4/nvidia/model.py (head version, relevant portion)

# Imports changed: instead of from vllm.model_executor.layers.mhc import ...
from vllm.model_executor.kernels.mhc.tilelang import (
    hc_head_fused_kernel_tilelang,
    mhc_fused_post_pre_tilelang,
    mhc_post_tilelang,
    mhc_pre_tilelang,
)

class DeepseekV4DecoderLayer(nn.Module):
    def __init__(self, vllm_config, prefix, ...):
        super().__init__()
        # Removed: import vllm.model_executor.layers.mhc # noqa: F401
        # Removed: self.mhc_pre = MHCPreOp() / self.mhc_post / self.mhc_fused_post_pre
        ...

    def forward(self, positions, x, residual=None, post_mix=None, res_mix=None):
        attn_norm_weight = self.attn_norm.weight.data
        attn_norm_eps = self.attn_norm.variance_epsilon
        if residual is None:
            # First layer: use mhc_pre_tilelang directly instead of self.hc_pre
            residual = x
            post_mix, res_mix, x = mhc_pre_tilelang(
                x,
                self.hc_attn_fn,
                self.hc_attn_scale,
                self.hc_attn_base,
                self.rms_norm_eps,
                self.hc_eps,
                self.hc_eps,
                self.hc_post_alpha,
                self.hc_sinkhorn_iters,
                norm_weight=attn_norm_weight,
                norm_eps=attn_norm_eps,
            )
        else:
            # Subsequent layers: use mhc_fused_post_pre_tilelang instead of self.mhc_fused_post_
            pre
            residual, post_mix, res_mix, x = mhc_fused_post_pre_tilelang(
                x,
                residual,
                post_mix,
                res_mix,
                ...
            )
        ...
```

vllm/model_executor/layers/mhc.py

CustomOp 包装层，调整 forward_cuda/forward_hip 以使用新的 kernel 签名。

```
# vllm/model_executor/layers/mhc.py (head version, HCHeadOp.forward_cuda)

class HCHeadOp(CustomOp):
    ...
    def forward_cuda(self, hidden_states, hc_fn, hc_scale, hc_base,
                     rms_norm_eps, hc_eps):
        hc_mult, hidden_size = hidden_states.shape[-2:]
        outer_shape = hidden_states.shape[:-2]
        hs_flat = hidden_states.view(-1, hc_mult, hidden_size)
        # Previously: pre-allocated out and passed to op
        # Now: op returns tensor directly
        out = torch.ops.vllm.hc_head_fused_kernel_tilelang(
            hs_flat,
            hc_fn,
            hc_scale,
            hc_base,
            rms_norm_eps,
            hc_eps,
        )
        return out.view(*outer_shape, hidden_size)

    def forward_hip(self, hidden_states, hc_fn, hc_scale, hc_base,
                    rms_norm_eps, hc_eps):
        hc_mult, hidden_size = hidden_states.shape[-2:]
        outer_shape = hidden_states.shape[:-2]
        hs_flat = hidden_states.view(-1, hc_mult, hidden_size)
        if HAS_TILELANG:
            out = torch.ops.vllm.hc_head_fused_kernel_tilelang(
                hs_flat, hc_fn, hc_scale, hc_base, rms_norm_eps, hc_eps)
        else:
            # Triton fallback: still pre-allocate out because triton op signature unchanged
            num_tokens = hs_flat.shape[0]
            out = torch.empty(num_tokens, hidden_size, dtype=torch.bfloat16,
                              device=hidden_states.device)
            torch.ops.vllm.hc_head_triton(hs_flat, hc_fn, hc_scale, hc_base,
                                          out, hidden_size, rms_norm_eps,
                                          hc_eps, hc_mult)
        return out.view(*outer_shape, hidden_size)
```

评论区精华

@tjtanaa 表示需要修复 AMD 的 import 语句，并会在当天处理。@WoosukKwon 解释故意不在此 PR 处理 AMD 路径，因为 AMD tilelang 路径尚未稳定，且此 PR 不对 AMD 产生任何变更。@tjtanaa 批准时确认：“Yes, it indeed doesn't affect amd as the custom op class is still there.”

- AMD 兼容性 (design): AMD 路径由 tjtanaa 在后续 PR 中修复；当前 PR 仅变更 NVIDIA 路径。

风险与影响

- 风险：主要风险在于 NVIDIA 路径的功能等价性：移除 CustomOp 包装后调用方式改变，但通过测试和 review 保证了正确性。hc_head_fused_kernel_tilelang 返回值的修改可能影响调用方（已全面修改）。AMD 路径未被触及，但未来 AMD 也可能需要类似清理。另外，部分 MTP 代码涉及推测解码，需确保行为一致。
- 影响：对用户透明，DeepSeek V4 模型推理行为无变化。对系统而言，减少了 CustomOp 注册的依赖，使得 tilelang 内核引用更直观，便于内核层面的独立迭代。AMD 平台无影响。
- 风险标记：核心路径改动，自定义算子移除，AMD 路径未触及

关联脉络

- PR #43891 [Model Refactoring] Remove unncessary torch op registration for DSv4: 同为 DeepSeek V4 的清理重构，移除不必要的 CustomOp 注册，方向一致。