PR #25976 完整报告

sgl-project/sglang

[DeepSeek-V4] Add mhc_fused_post_pre kernel

合并时间：2026-05-30 17:04

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25976

执行摘要

- 一句话：融合 DeepSeek-V4 mHC 后 / 前步长 kernel，解码性能 +3.35%
- 推荐动作：建议精读，重点关注 mhc_fused_post_pre_fma_tilelang 的 TileLang 实现和融合调度策略，对 LLM 推理 kernel fusion 有参考价值。同时注意其与现有 TileLang mHC 路径的依赖关系。

功能与动机

优化 DeepSeek-V4 mHC 延迟敏感解码路径，避免启动分离内核，借鉴 TRTLLM 融合策略。

实现拆解

1. 环境变量注册：在 python/sglang/srt/environ.py 添加 SGLANG_OPT_FUSE_MHC_POST_PRE = EnvBool(False)，默认关闭。
2. 核心融合内核：在 python/sglang/srt/layers/mhc.py 新增 mhc_fused_post_pre_fma_tilelang（TileLang JIT 内核），以及调度函数 mhc_fused_post_pre，小批次走融合 FMA，大批次走非融合 mhc_post + mhc_pre。
3. 模型集成：在 python/sglang/srt/models/deepseek_v4.py 中添加 _is_fused_mhc_post_pre_enabled、refresh_mhc_norm_weight_cache、prewarm_mhc_token_counts 预热扩展；修改 forward 方法实现跨层与层内融合。
4. NextN 适配：在 python/sglang/srt/models/deepseek_v4_nextn.py 中，解码器返回四元组，NextN 层执行最终 hc_post。
5. 单元测试：新增 tests/kernels/test_mhc_kernels.py，参数化验证融合路径与分离路径数值等价。

关键文件：

- python/sglang/srt/models/deepseek_v4.py（模块 模型适配；类别 source；类型 data-contract；符号 _is_fused_mhc_post_pre_enabled, refresh_mhc_norm_weight_cache, prewarm_mhc_token_counts, prewarm_mhc_token_count_buckets）：集成融合逻辑的核心文件，包含使能检查、归一化权重缓存、预热扩展和 forward 修改，是整个 PR 的主控点。
- python/sglang/srt/layers/mhc.py（模块 mHC 内核；类别 source；类型 core-logic；符号 mhc_fused_post_pre_fma_tilelang, mhc_fused_post_pre）：新增融合内核和调度函数的正确位置，是性能优化的核心实现。

- tests/kernels/test_mhc_kernels.py（模块 测试用例；类别 test；类型 test-coverage；符号 test_mhc_fused_post_pre_matches_unfused）：新测试文件，参数化验证融合路径与分离路径数值等价，覆盖多种隐藏维度、token 数量和归一化配置。
- python/sglang/srt/models/deepseek_v4_nextn.py（模块 NextN 适配；类别 source；类型 data-contract）：适配 NextN 编码器使用融合后的四元组返回，并在单层网络后补全 hc_post。
- python/sglang/srt/environ.py（模块 环境配置；类别 source；类型 core-logic）：注册环境变量开关 SGLANG_OPT_FUSE_MHC_POST_PRE，控制融合优化是否生效。

关键符号：_is_fused_mhc_post_pre_enabled, refresh_mhc_norm_weight_cache, prewarm_mhc_token_counts, prewarm_mhc_token_count_buckets, mhc_fused_post_pre_fma_tilelang, mhc_fused_post_pre, test_mhc_fused_post_pre_matches_unfused

关键源码片段

python/sglang/srt/models/deepseek_v4.py

集成融合逻辑的核心文件，包含使能检查、归一化权重缓存、预热扩展和 forward 修改，是整个 PR 的主控点。

```
# 全局函数：检查融合 mHC post/pre 是否启用
def _is_fused_mhc_post_pre_enabled() -> bool:
    return (
        envs.SGLANG_OPT_FUSE_MHC_POST_PRE.get()
        and envs.SGLANG_OPT_USE_TILELANG_MHC_PRE.get()
        and envs.SGLANG_OPT_USE_TILELANG_MHC_POST.get()
    )


# DeepseekV4DecoderLayer 类新增的方法：缓存 bf16 归一化权重
# 避免每轮 forward 重复 cast 和 contiguous 操作
def refresh_mhc_norm_weight_cache(self):
    self._input_layernorm_weight_bf16 = (
        self.input_layernorm.weight.data.bfloat16().contiguous()
    )
    self._post_attention_layernorm_weight_bf16 = (
        self.post_attention_layernorm.weight.data.bfloat16().contiguous()
    )
```

tests/kernels/test_mhc_kernels.py

新测试文件，参数化验证融合路径与分离路径数值等价，覆盖多种隐藏维度、token 数量和归一化配置。

```
import pytest
import torch

import sglang.srt.layers.mhc as mhc
from sglang.srt.layers.mhc import mhc_fused_post_pre, mhc_post, mhc_pre
```

```
@pytest.mark.parametrize("hidden_size", [4096, 7168])
@pytest.mark.parametrize("num_tokens", [0, 1, 8, 17, 32, 64])
@pytest.mark.parametrize("use_norm", [False, True])
def test_mhc_fused_post_pre_matches_unfused(
    monkeypatch, hidden_size, num_tokens, use_norm
):
    if not torch.cuda.is_available():
        pytest.skip("CUDA is required for TileLang mHC kernels")

    monkeypatch.setattr(mhc, "is_dsa_prefill_cp_round_robin_split", lambda: False)
    torch.manual_seed(0)
    device = torch.device("cuda")
    hc_mult = 4
    hc_mult3 = hc_mult * 2 + hc_mult * hc_mult
    hc_hidden_size = hc_mult * hidden_size

    x = torch.randn(num_tokens, hidden_size, device=device, dtype=torch.bfloat16) * 0.1
    residual = (
        torch.randn(
            num_tokens, hc_mult, hidden_size, device=device, dtype=torch.bfloat16
        )
        * 0.1
    )
    post_prev = torch.rand(num_tokens, hc_mult, 1, device=device, dtype=torch.float32)
    comb_prev = (
        torch.rand(num_tokens, hc_mult, hc_mult, device=device, dtype=torch.float32)
        * 0.25
    )
    fn = (
        torch.randn(hc_mult3, hc_hidden_size, device=device, dtype=torch.float32) * 0.01
    )
    hc_scale = torch.tensor([0.5, 0.25, 0.25], device=device, dtype=torch.float32)
    hc_base = torch.zeros(hc_mult3, device=device, dtype=torch.float32)
    norm_weight = (
        torch.ones(hidden_size, device=device, dtype=torch.bfloat16)
        if use_norm
        else None
    )
    norm_eps = 1e-6 if use_norm else None

    rms_eps = 1e-6
    hc_eps = 1e-6
    sinkhorn_repeat = 2

    residual_ref = post_ref = comb_ref = layer_ref = None
    if num_tokens > 0:
        residual_ref = mhc_post(x, residual, post_prev, comb_prev)
        post_ref, comb_ref, layer_ref = mhc_pre(
            residual_ref, fn, hc_scale, hc_base, rms_eps, hc_eps, hc_eps, 2.0,
```

```
            sinkhorn_repeat, norm_weight=norm_weight, norm_eps=norm_eps
        )
    residual_out, post_out, comb_out, layer_out = mhc_fused_post_pre(
        x, residual, post_prev, comb_prev, fn, hc_scale, hc_base, rms_eps,
        hc_eps, hc_eps, 2.0, sinkhorn_repeat, norm_weight=norm_weight,
        norm_eps=norm_eps
    )

    torch.cuda.synchronize()
    if num_tokens == 0:
        assert residual_out.shape == residual.shape
        assert post_out.shape == (0, hc_mult, 1)
        assert comb_out.shape == (0, hc_mult, hc_mult)
        assert layer_out.shape == (0, hidden_size)
        return

    assert residual_ref is not None and post_ref is not None
    assert comb_ref is not None and layer_ref is not None
    torch.testing.assert_close(residual_out, residual_ref, atol=0, rtol=0)
    torch.testing.assert_close(post_out, post_ref, atol=1e-3, rtol=1e-3)
    torch.testing.assert_close(comb_out, comb_ref, atol=1e-3, rtol=1e-3)
    layer_atol = 2e-2 if use_norm else 2e-3
    layer_rtol = 2e-2 if use_norm else 2e-3
    torch.testing.assert_close(layer_out, layer_ref, atol=layer_atol, rtol=layer_rtol)
```

评论区精华

1. 测试覆盖率扩展：审查者 yhyang201 建议增加 hidden_size=7168 和 batch size 64，以覆盖 V4 Pro 和大路径；作者已采纳。
2. 融合内核预热：yhyang201 提出在 prewarm_mhc_token_counts 中添加融合内核预热，避免首次调用冷启动延迟；已实现。
3. 代码质量：gemini-code-assist 指出未使用变量 block_k/block_m、重复导入及魔数 2.0；后续添加了 _MHC_POST_MULT_VALUE 常量，部分问题未明确回应。

- 增加 hidden_size 参数化和 batch size 64 (testing): 作者已采纳并补充。
- 融合内核预热 (performance): 作者已实现。

风险与影响

- 风险：
  1. 数值精度：融合内核计算顺序不同，测试允许 2e-2 误差，需持续监控端到端精度。
  2. 大批次回退：依赖现有 DeepGEMM 路径，无回归风险。
  3. TileLang 依赖：仅 CUDA 平台支持，其他硬件自动禁用。
  4. 配置复杂性：需同时启用 SGLANG_OPT_FUSE_MHC_POST_PRE 和 TileLang 开关才能激活。
- 影响：

1. 性能收益：小批次解码吞吐提升约 3.35%，对交互式部署有利。
2. 用户影响：新增环境变量默认关闭，不影响现有行为。
3. 维护负担：融合路径需与分离路径保持语义一致，后续 mHC 修改需同步。 - 风险标记：核心路径变更，TileLang 依赖，仅 CUDA 支持，精度兼容风险

关联脉络

- 暂无明显关联 PR