PR #23874 完整报告

sgl-project/sglang

Fix failing `test_nvidia_nemotron_3_nano` by fixing `test_grouped_topk`

合并时间：2026-04-29 06:03

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23874

执行摘要

- 一句话：修复 grouped_topk 负分数排序和 Mamba 填充 bug，解锁 Nemotron-3-Nano
- 推荐动作：此 PR 值得精读，尤其是 pack_val_idx 的 IEEE 754 位转换技巧和 CUDA graph 下的零填充模式。对于维护其他 GPU kernel 的开发者有借鉴意义。

功能与动机

Nemotron-3-Nano 的 FP8 测试在 JIT grouped_topk 路径开启后持续失败。该模型的路由器配置（单专家组、topk_group=1、128 个路由专家、topk=6、带 correction_bias）恰好匹配 kernel 约束，从而触发了两个隐藏 bug：负 choice score 排序不正确，以及 Mamba 输出 padding 未初始化。这些 bug 需要修复以恢复模型可用性。

实现拆解

1. 修正 grouped_topk CUDA kernel 的浮点打包逻辑（python/sglang/jit_kernel/csrc/moe/grouped_topk.cuh）：
   - 修改 pack_val_idx 函数中的位转换策略，将 IEEE 754 位模式转换为全浮点范围单调的无符号排序，使负分数也能正确参与 warp-level 最大值归约。
   - 对应的 unpack_val_idx 做逆向还原。
   - 同时调整了 renormalize 阶段：让 warp 中所有 lane 都参与 warp_sum_f32，避免非对齐 topk 时部分 lane 不执行导致死锁。
2. 零填充 Mamba 分裂算子的输出 padding（python/sglang/srt/models/nemotron_h.py）：
   - 在 nemotron_mamba2_with_output 函数中，复制有效 token 结果后，对超出实际 token 数的部分显式调用 .zero_()，确保 graph replay 时 padding 区域不包含垃圾值。
3. 重新启用 Nemotron-3-Nano 集成测试（test/registered/models/test_nvidia_nemotron_3_nano.py）：
   - 移除 register_cuda_ci 中的 disabled 标志，让该测试重新加入 CI。
4. 新增 negative choice scores 专项测试（python/sglang/jit_kernel/tests/test_grouped_topk.py，新文件 210 行）：
   - 引入 test_grouped_topk_negative_choice_scores_match_reference，使用 correction_bias.fill_(-2.0) 强制产生负分数，验证 JIT 实现与参考实现一致。

- 同时添加 CORRECTNESS_CASES 参数化覆盖，包括最小的非 2 的幂 topk、Nemotron 形状（E=128, topk=6）等场景。

关键文件：

- python/sglang/jit_kernel/csrc/moe/grouped_topk.cuh（模块 路由内核；类别 other；类型 core-logic；符号 pack_val_idx, unpack_val_idx, grouped_topk_single_group_kernel）：核心修复：修正浮点打包使负分数正确排序，并调整 warp_sum 参与方式防止死锁。
- python/sglang/srt/models/nemotron_h.py（模块 模型层；类别 source；类型 data-contract；符号 nemotron_mamba2_with_output）：Mamba 输出零填充，防止下流 kernel 读到未初始化垃圾值。
- python/sglang/jit_kernel/tests/test_grouped_topk.py（模块 单元测试；类别 test；类型 test-coverage；符号 _make_inputs, _scatter_by_expert, test_grouped_topk_renormalize_matches_reference, test_grouped_topk_non_power_of_two_renormalize）：新增 210 行测试，包括负分数回归测试和多种形状的正确性验证。
- test/registered/models/test_nvidia_nemotron_3_nano.py（模块 集成测试；类别 test；类型 test-coverage）：移除了临时禁用标志，使 Nemotron-3-Nano 回归测试重新加入 CI。

关键符号：pack_val_idx, unpack_val_idx, grouped_topk_single_group_kernel, nemotron_mamba2_with_output, _make_inputs, _scatter_by_expert, test_grouped_topk_renormalize_matches_reference, test_grouped_topk_non_power_of_two_renormalize, test_grouped_topk_negative_choice_scores_match_reference, test_grouped_topk_without_renormalize_matches_reference

关键源码片段

python/sglang/jit_kernel/csrc/moe/grouped_topk.cuh

核心修复：修正浮点打包使负分数正确排序，并调整 warp_sum 参与方式防止死锁。

```
// sglang/jit_kernel/csrc/moe/grouped_topk.cuh

// 将 (value, index) 打包为 uint64_t，用于 warp-level max 归约。
// 将 IEEE 754 位模式转换为全浮点范围单调的无符号排序，
// 使得负分数也能正确比较（因为 correction_bias 可使 sigmoid(score)+bias 为负）。
__device__ __forceinline__ uint64_t pack_val_idx(float val, int32_t idx) {
    uint32_t val_bits = __float_as_uint(val);
    // 若最高位（符号位）为 1，将所有位取反；否则仅翻转符号位
    val_bits ^= (val_bits & 0x80000000u) ? 0xffffffffu : 0x80000000u;
    // 使用 (65535 - idx) 使较小 index 在 tie 时获胜
    uint32_t idx_bits = static_cast<uint32_t>(65535 - idx);
    return (static_cast<uint64_t>(val_bits) << 32) | idx_bits;
}

__device__ __forceinline__ void unpack_val_idx(uint64_t packed, float& val, int32_t& idx) {
    uint32_t idx_bits = static_cast<uint32_t>(packed & 0xFFFFFFFF);
```

```
    idx = static_cast<int32_t>(65535 - idx_bits);
    uint32_t val_bits = static_cast<uint32_t>(packed >> 32);
    // 逆向还原：若最高位为 1，翻转符号位；否则全取反
    val_bits ^= (val_bits & 0x80000000u) ? 0x80000000u : 0xffffffffu;
    val = __uint_as_float(val_bits);
}

// (kernel 中 renormalize 阶段变更 )
// 让所有 lane 参与 warp_sum_f32，然后仅前 topk 个 lane 写输出
float weight = (lane_id < topk) ? selected_weights[lane_id] : 0.0f;
float divisor = renormalize ? warp_sum_f32(weight) + 1e-20f : 1.0f;
if (lane_id < topk) {
    out_ids[lane_id] = selected_ids[lane_id];
    out_vals[lane_id] = weight * scaling_factor / divisor;
}
```

python/sglang/srt/models/nemotron_h.py

Mamba 输出零填充，防止下流 kernel 读到未初始化垃圾值。

```
# sglang/srt/models/nemotron_h.py

@register_custom_op(mutates_args=["output"])
@register_split_op()
def nemotron_mamba2_with_output(
    hidden_states: torch.Tensor,
    output: torch.Tensor,
    layer_id: int,
) -> None:
    """Split op for Mamba2 forward in piecewise CUDA graph mode."""
    context = get_forward_context()
    forward_batch = context.forward_batch
    attention_layers = context.attention_layers
    mamba_layer = attention_layers[layer_id]

    # 在 CUDA graph 模式下，hidden_states 可能被 padding 到捕获图的大小
    attn_backend = forward_batch.attn_backend
    metadata = attn_backend.linear_attn_backend.forward_metadata
    num_actual_tokens = metadata.num_prefill_tokens + (
        metadata.num_decodes * metadata.draft_token_num
        if metadata.is_target_verify
        else metadata.num_decodes
    )
    if hidden_states.shape[0] != num_actual_tokens:
        hidden_states = hidden_states[:num_actual_tokens]

    ret = mamba_layer._forward_mamba(hidden_states, forward_batch)

    # 仅复制有效 token 的结果；output 可能比实际大（padding）
    output[:num_actual_tokens].view(ret.shape).copy_(ret)
```

```
    # 新增：若输出有 padding，则将其零填充，防止下流 kernel（如 grouped_topk/FP8 MoE）
    # 在 graph replay 时读到垃圾值
    if output.shape[0] != num_actual_tokens:
        output[num_actual_tokens:].zero_()
```

python/sglang/jit_kernel/tests/test_grouped_topk.py

新增 210 行测试，包括负分数回归测试和多种形状的正确性验证。

```
# sglang/jit_kernel/tests/test_grouped_topk.py

import itertools
import pytest
import torch
from sglang.jit_kernel.grouped_topk import grouped_topk as jit_grouped_topk
from sglang.jit_kernel.utils import get_ci_test_range
from sglang.srt.layers.moe.topk import biased_grouped_topk_impl
from sglang.test.ci.ci_register import register_cuda_ci

# 注册为 CI 测试
register_cuda_ci(est_time=30, suite="stage-b-kernel-unit-1-gpu-large")
register_cuda_ci(est_time=120, suite="nightly-kernel-1-gpu", nightly=True)

# 正确性测试形状组合：包括 Nemotron-3-Nano 暴露的 bug 形状 (17,128,6)
CORRECTNESS_CASES = get_ci_test_range(
    full_range=list(itertools.product(
        [1, 17, 128], # num_tokens
        [16, 32, 64, 128, 192, 256, 384, 512], # num_experts
        [1, 2, 3, 4, 5, 6, 7, 8], # topk
    )),
    ci_range=[
        (1, 16, 3), # 最小非 2 的幂 topk
        (17, 128, 6), # Nemotron-3-Nano bug 形状
        (128, 192, 8), # Hunyuan-3 power-of-two 基准
        (33, 512, 7), # 大专家数非 2 的幂 topk
    ],
)

def test_grouped_topk_negative_choice_scores_match_reference() -> None:
    """强制 correction_bias 为 -2.0，使 sigmoid(score) + bias 一半为负，
    验证 JIT 实现与参考实现一致。"""
    torch.manual_seed(23758)
    hidden_states = torch.empty((64, 1), dtype=torch.float32, device="cuda")
    gating_output = torch.randn((64, 128), dtype=torch.float32, device="cuda")
    correction_bias = torch.full((128,), -2.0, dtype=torch.float32, device="cuda")

    topk_weights, topk_ids = jit_grouped_topk(
        gating_output, correction_bias, 1, 1, 6, True, 1.0)
    ref_weights, ref_ids = biased_grouped_topk_impl(
        hidden_states, gating_output, correction_bias, 6, True, 1, 1,
```

```
    routed_scaling_factor=1.0, apply_routed_scaling_factor_on_output=True)

# 通过 scatter 将稀疏结果转为稠密矩阵比较
def scatter(weights, ids, num_exp):
    dense = torch.zeros((weights.shape[0], num_exp),
                        dtype=torch.float32, device=weights.device)
    dense.scatter_(1, ids.long(), weights)
    return dense

torch.testing.assert_close(
    scatter(topk_weights, topk_ids, 128),
    scatter(ref_weights, ref_ids, 128),
    rtol=1e-5, atol=1e-6)
```

评论区精华

本 PR 的 review 评论主要为 CI 触发和执行结果报告，未出现技术争议。作者 kpham-sgl 多次通过 /rerun-test 和 /rerun-failed-ci 指令重跑失败测试，最终所有测试通过后由 Qiaolin-Yu 批准合并。

- 暂无高价值评论线程

风险与影响

- 风险：CUDA kernel 位操作风险：pack_val_idx 中的符号位处理（条件异或）容易出错，若未来引入其他浮点打包方式需特别测试。 Mamba 零填充风险：零填充仅依赖于 output.shape[0] != num_actual_tokens 的判断，若 output 的存储布局在后续版本中变更（如变为非连续视图），.zero_() 可能产生意外影响。当前仅在 piecewise CUDA graph 模式下生效，但代码路径是通用的，安全。 回归风险：改动影响所有使用 grouped_topk JIT kernel 的模型（如 Hunyuan-3），但测试已覆盖典型形状。新增的 negative score 测试直接针对边界条件。
- 影响：用户：Nemotron-3-Nano 模型可正常使用，FP8 模式不再因路由崩溃。 系统：JIT grouped_topk kernel 的数值正确性提升，适用于任何带 correction_bias 且可能产生负 choice score 的路由场景。 团队：模型测试已重新激活，CI 覆盖更全面。
- 风险标记：CUDA kernel 位操作，Mamba 零填充依赖 shape 判断

关联脉络

- 暂无明显关联 PR