PR #34481 完整报告

sgl-project/sglang

[AMD] Keep the PTX-inline-asm diffusion norm fusions off on ROCm (fix FLUX warmup crash)

合并时间：2026-08-20 08:55

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34481

执行摘要

- 一句话：在 ROCm 上禁用 NV PTX 位精确 norm 融合，修复 FLUX 预热崩溃。
- 推荐动作：此 PR 值得精读。它揭示了在跨平台代码中处理不可编译的 PTX 时应使用平台守卫而非依赖 try/except，并展示了如何通过测试正确标记平台相关行为。

功能与动机

FLUX.1-dev 在 ROCm 上预热时崩溃，服务器无法就绪，CI 测试 multimodal-gen-test-1-gpu-amd* 失败。tl.inline_asm_elementwise PTX 中的 float 寄存器约束在 AMDGPU 后端不受支持，LLVM 将其视为致命错误直接终止进程，绕过了各站点的 try/except 回退。

实现拆解

1. 修改融合内核守卫

在 layernorm_modulate_triton.py 的 _is_bf16_cuda 和 rmsnorm_scale_shift_bitexact.py 的 can_use_fused_rmsnorm_scale_shift 中前置 is_cuda() 判断，使守卫在 ROCm 上返回 False，从而不执行 PTX 内核而走 eager 路径。

2. 添加测试跳过标记

在 test_model_fast_paths.py 中定义 requires_inline_ptx skip 标记，为断言融合结果的子测试添加此标记，使它们在 ROCm 上跳过，而其余测试可正常运行。

3. 新增平台一致性测试

新增 test_bitexact_norm_guards_follow_platform 测试，验证守卫在不同平台上的行为符合预期（CUDA 上启用，ROCm 上拒绝）。

4. 配套改动

未修改 common/numerics.py，复用已有的 is_cuda() 约定。

关键文件：

- python/sglang/kernels/ops/diffusion/norm/layernorm_modulate_triton.py（模块 内核层；类别 infra；类型 infrastructure）：修改 _is_bf16_cuda 守卫，前置 is_cuda()，使 PTX 融合在 ROCm 上禁用。

- python/sglang/kernels/ops/diffusion/norm/rmsnorm_scale_shift_bitexact.py（模块 内核层；类别 infra；类型 infrastructure）：修改 can_use_fused_rmsnorm_scale_shift 守卫，前置 is_cuda()，使 RMSNorm 融合在 ROCm 上禁用。
- test/registered/kernels/ops/diffusion/test_model_fast_paths.py（模块 测试；类别 test；类型 test-coverage；符号 test_bitexact_norm_guards_follow_platform）：添加 requires_inline_ptx skip 标记和新的平台一致性测试，使 AMD nightly 测试可正常运行。

关键符号：_is_bf16_cuda, can_use_fused_rmsnorm_scale_shift, test_bitexact_norm_guards_follow_platform

关键源码片段

python/sglang/kernels/ops/diffusion/norm/layernorm_modulate_triton.py

修改 _is_bf16_cuda 守卫，前置 is_cuda()，使 PTX 融合在 ROCm 上禁用。

```
# python/sglang/kernels/ops/diffusion/norm/layernorm_modulate_triton.py
# 此函数原本仅检查张量是否在 CUDA 且为 bf16。
# 现在前置了平台判断 `is_cuda()`，确保在 ROCm 上直接返回 False，
# 从而跳过后续的 PTX inline asm 内核，避免 LLVM 因不支持的 `=f`
# 寄存器约束而触发致命错误终止进程。
def _is_bf16_cuda(t: torch.Tensor) -> bool:
    # `is_cuda` 也适用于 ROCm，但下面的 inline PTX 无法在 ROCm 上编译：
    # LLVM 将不可用的 `=f` 约束视为致命错误，直接杀死进程，
    # 因此必须在首次 launch 之前拒绝，而不是依赖调用方的 try/except。
    return is_cuda() and t.is_cuda and t.dtype is torch.bfloat16
```

python/sglang/kernels/ops/diffusion/norm/rmsnorm_scale_shift_bitexact.py

修改 can_use_fused_rmsnorm_scale_shift 守卫，前置 is_cuda()，使 RMSNorm 融合在 ROCm 上禁用。

```
# python/sglang/kernels/ops/diffusion/norm/rmsnorm_scale_shift_bitexact.py
# 此守卫同样前置 `is_cuda()`，确保 ROCm 上不执行 inline PTX。
# 与 LayerNorm 版不同，这里的条件原来是 `x.dtype is torch.bfloat16`，
# 现在将其与平台判断组合，保证在 ROCm 上返回 False。
def can_use_fused_rmsnorm_scale_shift(
    x: torch.Tensor,
    weight: torch.Tensor,
    scale: torch.Tensor,
    shift: torch.Tensor,
) -> bool:
    # ROCm 无法编译上面的 inline PTX：LLVM 将不可用的 `=f` 约束
    # 视为致命错误杀死进程，所以要在首次 launch 前拒绝，
    # 而不是依赖调用方的回退。
    return (
        is_cuda()
        and x.dtype is torch.bfloat16
        and x.is_cuda
        and x.dim() == 3
```

```
        and x.is_contiguous()
    )
```

test/registered/kernels/ops/diffusion/test_model_fast_paths.py

添加 requires_inline_ptx skip 标记和新的平台一致性测试，使 AMD nightly 测试可正常运行。

```
# test/registered/kernels/ops/diffusion/test_model_fast_paths.py
# 定义平台相关的 skip 标记，并新增测试验证守卫在两种平台上行为一致。
# 由于位精确的 LayerNorm/RMSNorm 融合是 NVIDIA inline PTX，
# 其在 ROCm 上无法编译，因此守卫在 ROCm 上返回 False，
# 这些站点在 ROCm 上使用 eager 路径；只有断言融合结果的子测试是 CUDA 专用的。
requires_inline_ptx = pytest.mark.skipif(
    not is_cuda(), reason="bit-exact norm fusions are NVIDIA PTX"
)


def test_bitexact_norm_guards_follow_platform():
    # 在两条 CI lane 上运行，形状在守卫的契约内，
    # 因此只有平台决定结果：CUDA 上启用，ROCm 上拒绝。
    # 在 ROCm 上，LLVM 错误会杀死进程，因此站点的 try/except 无法捕捉，
    # 守卫必须提前返回 False。
    x = torch.randn(1, 256, 4096, device="cuda", dtype=torch.bfloat16)
    row = torch.randn(1, 4096, device="cuda", dtype=torch.bfloat16)
    vec = torch.randn(1, 1, 4096, device="cuda", dtype=torch.bfloat16)
    weight = torch.randn(4096, device="cuda", dtype=torch.bfloat16)
    q = torch.randn(1, 256, 32, 128, device="cuda", dtype=torch.bfloat16)
    assert can_use_fused_layernorm_modulate(x, row, row) is is_cuda()
    assert can_use_fused_qk_head_layernorm(q, q) is is_cuda()
    assert can_use_fused_rmsnorm_scale_shift(x, weight, vec, vec) is is_cuda()
```

评论区精华

Review 中 sushildubey171 对 test_model_fast_paths.py 和 common/numerics.py 提出了 'cleanup' / 'cleanup comments' 的评论，但未展开具体内容；PR 已获得 APPROVED。issue 中 kangwangamd 评论表示此 PR 与 #34352 解决相同问题，并关闭了自己的 PR 以支持此方案。

- 测试清理评论 (style)：已处理，PR 获得批准。

风险与影响

- 风险：该改动仅影响 ROCm 平台，CUDA 行为不变。风险较低，因为守卫在 ROCm 上仅回退到 eager 路径，而 eager 路径原本就是形状超出契约时的回退。但需注意，测试文件 test_model_fast_paths.py 在 AMD nightly 上因第一种子测试崩溃而无法运行，因此此改动实际上恢复了该文件的其余测试覆盖。
- 影响：影响范围限于 AMD/ROCm 上的扩散模型（FLUX.1、LTX-2 等）。修复了 FLUX.1-dev 预热崩溃，使 CI 测试通过；对 CUDA 用户无影响。
- 风险标记：ROCm 专属修复，测试恢复覆盖，平台守卫而非 try/except

关联脉络

- PR #34352 [AMD] Gate the fused Diffusion-LayerNorm modulate on CUDA: 同一 FLUX 预热崩溃根因，仅门控了 LayerNorm 模块；本 PR 扩展至 RMSNorm 侧。
- PR #34351 [AMD] Gate the fused Diffusion-LayerNorm modulate on CUDA: 相关 issue ，与本 PR 涉及相同问题。
- PR #34485 [AMD] Let the diffusion AITer backend take grouped-query K/V (fix Cosmos3-Nano startup): 修复同一个 1-GPU CI 作业中的 Cosmos3 错误，与本 PR 共同解决 AMD CI 失败。