PR #27947 完整报告

sgl-project/sglang

[AMD] Fix jit-kernel-unit-test-amd: activation.cuh ROCm build + per_token CUDA-only (R165)

合并时间：2026-06-16 14:44

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27947

PR 27947 分析报告

1. 执行摘要

本 PR 修复了 AMD 平台下 jit-kernel-unit-test-amd 套件中两个长期失败的测试：test_activation（编译错误）和 test_per_token_group_quant_8bit（运行时阻塞）。通过修改 activation.cuh 使用显式函数指针类型避免 clang-HIP 的 const 推导问题，并为 per_token_group_quant_8bit.cuh 添加 AMD 专用 reduce 实现，同时将 per_token 测试调整为仅 CUDA 注册。改动后 CUDA 路径完全不变，AMD activation 测试重新通过（73/73），CI 回归绿色。

2. 功能与动机

AMD 的 JIT 内核单元测试套件（R165）自 #27722 启用 test_activation 和 test_per_token_group_quant_8bit 后持续失败。根本原因：activation.cuh 中 unary kernel 的 select_unary_kernel 返回 decltype(auto) 导致 clang-HIP 无法初始化 const 限定的函数指针；而 per_token 测试依赖的 fp8 dtype trait 在 __HIPCC__ 下未实现，且其 warp reduce 使用 32 位掩码与 HIP 64 位 warp 不兼容。本 PR 旨在使该套件恢复绿色，并最小化对 CUDA 的影响。

3. 实现拆解

1. activation.cuh 编译修复：在 ActivationKernel 内新增 unary_kernel_fn_t 类型别名，将 select_unary_kernel 的返回类型从 trailing decltype 改为显式非 const 函数指针。该方案与二进制路径的现有模式一致，且无需 #ifdef，CUDA 编译不变。
2. per_token 内核 AMD 适配：在 per_token_group_quant_8bit.cuh 中，保持 CUDA GroupReduceMax 原样，新增 #ifdef USE_ROCM 分支，使用 device::warp::reduce_max<kThreadsPerGroup> 实现。HIP 的 64 位 warp 需要显式子组宽度，故不可复用原 32 位掩码逻辑。
3. 测试注册调整：在 test_per_token_group_quant_8bit.py 中移除 register_amd_ci，使该测试仅用于 CUDA。原因是框架级 fp8 支持缺失导致 TensorMatcher 无法识别输出类型。CUDA 全量 1872 个配置的矩阵测试保留并继续在 nightly 中运行。

python/sglang/jit_kernel/csrc/elementwise/activation.cuh

核心修复：解决 ROCm 下 unary kernel 编译错误，通过添加 unary_kernel_fn_t 类型别名并修改 select_unary_kernel 返回类型。

```
struct ActivationKernel {
    // ...
    using kernel_fn_t = decltype(&act_and_mul_kernel<T, ActivationKind::kSiLU, kUsePDL, false>)
    ;
    // ADDED: concrete typedef for unary kernel function pointer (mirrors binary path)
    using unary_kernel_fn_t = decltype(&act_kernel<T, ActivationKind::kReLU2, kUsePDL>);

    template <ActivationKind kAct>
    static constexpr auto unary_kernel = act_kernel<T, kAct, kUsePDL>;

    // BEFORE: static auto select_unary_kernel(const std::string& type)
    // -> decltype(ActivationKernel::template unary_kernel<ActivationKind::kReLU2>)
    // AFTER: explicit non-const fn-ptr type; works for both nvcc and clang-HIP
    static unary_kernel_fn_t select_unary_kernel(const std::string& type) {
        using namespace host;
        if (type == "relu2") {
            return ActivationKernel::template unary_kernel<ActivationKind::kReLU2>;
        }
        // ... other cases
        return nullptr;
    }
};
```

python/sglang/jit_kernel/csrc/gemm/per_token_group_quant_8bit.cuh

添加 AMD 专用的 GroupReduceMax 实现，保留 CUDA 路径不变。

```
namespace {
constexpr int kThreadsPerGroup = 16;

#ifdef USE_ROCM
// AMD implementation: HIP warps are 64-wide and require explicit sub-group width.
// Delegate to the portable warp-reduce primitive, which emits __shfl_xor with
// the correct kThreadsPerGroup sub-group width.
__device__ __forceinline__ float GroupReduceMax(float val, const int /*tid*/) {
    return device::warp::reduce_max<kThreadsPerGroup>(val);
}
#else
// CUDA implementation (unchanged from main): hand-rolled reduction using
// 32-bit shuffle masks; assumes warp width <= 32.
__device__ __forceinline__ float GroupReduceMax(float val, const int tid) {
    unsigned mask = threadIdx.x % 32 >= 16 ? 0xffff0000 : 0x0000ffff;
    val = fmaxf(val, __shfl_xor_sync(mask, val, 8));
    val = fmaxf(val, __shfl_xor_sync(mask, val, 4));
    val = fmaxf(val, __shfl_xor_sync(mask, val, 2));
    val = fmaxf(val, __shfl_xor_sync(mask, val, 1));
    return val;
}
#endif
} // namespace
```

5. 评论区精华

- BBuf和 DarkSharpness反对在 #ifdef USE_ROCM 中复制整个函数体，建议只保留一个非 trailing 返回类型的通用版本。最终作者移除 #ifdef，使用显式类型别名，使代码更简洁。
- BBuf建议保留 CUDA GroupReduceMax 不变，仅添加 AMD 独立分支，以避免影响 CUDA 代码。作者采纳，CUDA 路径字节级一致。
- HaiShaw询问测试移除原因，作者解释从未通过且是意外注册，社区认可该决策。

6. 风险与影响

- CUDA 路径：完全不受影响，activation 返回类型变更在 nvcc 下等价，per_token 代码保持字节一致。
- AMD 验证：新增的 warp reduce 实现未在 AMD CI 中直接测试（因 per_token 测试已移除注册）。若未来启用在 AMD 上运行，需额外验证。
- 测试覆盖：per_token 测试从 AMD 移除，但因其从未通过，实际覆盖未损失。后续 AMD fp8 框架补全后可重新注册。
- 维护成本：内核现在有两个 reduce 实现路径，未来修改需同步考虑两个分支。

7. 关联脉络

- 本 PR 是 #28323（同一问题的独立尝试）的演进版本，采纳了更多 Code Review 反馈（统一返回类型、分离 reduce 实现）。
- 27722 无意中在 AMD 上启用了这两个测试，暴露了底层编译和框架兼容性问题。
- 26733 引入了 activation unary kernel，其 trailing decltype 返回类型在 clang-HIP 下是已知陷阱。
- 随后的工作（如 #28293 NPU 回退）和本 PR 共同提升了 sglang 在非 NVIDIA 平台上的 JIT 内核兼容性。