PR #47538 完整报告

vllm-project/vllm

[Performance][Hardware][RISC-V] Reduce LMUL pressure in INT4 LUT dequant

合并时间：2026-07-06 13:58

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47538

执行摘要

该 PR 针对 RISC-V 平台的 INT4 LUT 反量化构造函数（FP32Vec16(int64_t, const FP32Vec16&)），通过将向量操作从 u64@LMUL_1024 降级为 u32@LMUL_256 并拆分合并，降低了 75% 的寄存器压力。实测在 Spacemit X100（VLEN=256）上获得 2.08 倍加速，且 VLEN=128 同样受益。变更仅涉及单个头文件，回归风险低。

功能与动机

RISC-V 向量扩展中，LMUL（向量长度乘数）决定了每个向量变量占用的物理寄存器数量。原始实现使用 u64@LMUL_1024 处理 64 位整数的 nibble 提取：

- 在 VLEN=128 上映射为 m8（占用全部 32 个向量寄存器）
- 在 VLEN=256 上映射为 m4（占用 16 个向量寄存器）

这导致严重的寄存器溢出（spilling），拖累性能。优化方案将 64 位值拆分为两个 32 位半字，各自使用 u32@LMUL_256 处理，再通过 vcreate 合并为 LMUL_512 用于最终查表。每变量的 LMUL 需求降至原来的 1/4（m1 或 m2），寄存器压力降低 75%。

实现拆解

1. 拆分 64 位输入：将 int64_t value 转换为 uint64_t，再分别提取低 32 位（lo）和高 32 位（hi）。
2. 半字规模处理：对每个半字，生成 8 个 lane ID（vid），左移 2 位得到 nibble 偏移，用 vsrl 和 vand 提取 4-bit 索引。所有操作均使用 u32@LMUL_256。
3. 合并与查表：通过 vcreate 将两个 vuint32m1_t 合并为 vuint32m2_t（即 LMUL_256 到 LMUL_512），最后用 vrgather 从 LUT 中 gather 16 个 float32 结果。
4. 测试验证：PR body 中附带了完整的正确性测试（与标量参考对比）和性能基准（百万次迭代计时），结果 PASS 且加速比 2.08x。

csrc/cpu/cpu_types_riscv_impl.hpp

核心逻辑变更：FP32Vec16 构造函数中 INT4 LUT 反量化路径的 LMUL 优化。

```
// FP32Vec16(int64_t, const FP32Vec16&) — INT4 LUT 反量化构造函数
// 原始实现使用 u64 @ LMUL_1024，在 VLEN=128 上注册压力高达 m8（所有 32 个向量寄存器）
// 优化后：先将 64 位值拆为两个 32 位半字，各用 u32 @ LMUL_256 处理，
// 最终通过 vcreate 合并为 u32 @ LMUL_512 用于 vrgather。寄存器压力降至 m1/m2。
explicit FP32Vec16(int64_t value, const FP32Vec16& lut) {
```

```
    constexpr int HALF = VEC_ELEM_NUM / 2; // HALF = 8
    const auto q = static_cast<uint64_t>(value);
    const uint32_t lo = static_cast<uint32_t>(q);
    const uint32_t hi = static_cast<uint32_t>(q >> 32);

    // 生成 0..7 的 lane ID（u32 @ LMUL_256）
    auto lane_ids = RVVI(__riscv_vid_v_u32, LMUL_256)(HALF);
    // 每个 nibble 偏移 4 bits，故 left shift by 2（乘 4）
    auto shifts = RVVI(__riscv_vsll_vx_u32, LMUL_256)(lane_ids, 2, HALF);

    // 提取低 8 个 nibble 索引
    auto packed_lo = RVVI(__riscv_vmv_v_x_u32, LMUL_256)(lo, HALF);
    auto idx_lo = RVVI(__riscv_vand_vx_u32, LMUL_256)(
        RVVI(__riscv_vsrl_vv_u32, LMUL_256)(packed_lo, shifts, HALF),
        0xF, HALF);

    // 提取高 8 个 nibble 索引
    auto packed_hi = RVVI(__riscv_vmv_v_x_u32, LMUL_256)(hi, HALF);
    auto idx_hi = RVVI(__riscv_vand_vx_u32, LMUL_256)(
        RVVI(__riscv_vsrl_vv_u32, LMUL_256)(packed_hi, shifts, HALF),
        0xF, HALF);

    // 合并为 16 个 u32 索引（LMUL_256 -> LMUL_512），用于 vrgather
    auto idx = RVVI4(__riscv_vcreate_v_u32, LMUL_256, _u32,
                     LMUL_512)(idx_lo, idx_hi);
    reg = RVVI(__riscv_vrgather_vv_f32, LMUL_512)(lut.reg, idx, VEC_ELEM_NUM);
}
```

评论区精华

无 review 技术讨论。PR 由维护者 bigPYJ1151 直接批准，仅有一条 bot 自动评论和一条无关 CI 问题的评论。

风险与影响

- 风险：变更仅影响 RISC-V 平台下的 FP32Vec16 构造函数，不涉及其他硬件或逻辑路径，回归风险极低。
- 影响：显著提升 RISC-V 设备上 INT4 量化模型的推理性能（约 2 倍加速），对 VLEN=128 和 VLEN=256 均有效。

关联脉络

无直接关联的 PR 或 Issue。本 PR 可视为 RISC-V 向量化优化系列的一部分，与近期 #45243（RISC-V BF16 支持）、#47532（VLEN 检测修复）共同完善了 vLLM 对 RISC-V 平台的适配。