PR #50133 完整报告

vllm-project/vllm

[CPU] Migrate unquantized MoE to the modular-kernel experts structure

合并时间：2026-08-03 14:55

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50133

执行摘要

- 一句话：CPU 未量化 MoE 迁移到模块化内核专家结构
- 推荐动作：值得精读。这是 vLLM 将 CPU MoE 完全纳入 modular-kernel/oracle 体系的关键一步，其中三个设计决策尤其值得借鉴：一是路由参数在 process_weights_after_loading 中从 layer 捕获以弥补 monolithic apply() 签名限制；二是不支持的形状"报错并提示调参"而非静默回退；三是评审过程中通过补齐向量原语将 grouped gemm 扩展到所有 ISA、直接删除 torch 回退的取舍过程。建议同时关注其与并行 PR 的合并顺序。

功能与动机

PR 正文明确指出：未量化 CPU MoE 是遗留 monolithic 路径上最后一个未量化后端，oracle/unquantized.py 在 # TODO: migrate to MK structure 处直接短路，UnquantizedFusedMoEMethod 为此携带了三个 CPU 逃生舱（is_monolithic、专门的 process_weights_after_loading 分支和 apply_monolithic 分支）。同时 CPUFusedMOE 把三种不相关的实现缠绕在 self.forward_method 属性和 check_grouped_gemm 的 if 阶梯里，SGLFusedMOE 仅支持 SILU 且已被 grouped-gemm kernel 取代。迁移目的是让 CPU 与其他后端一样由 oracle 统一选择专家实现，并消除重复维护负担。

实现拆解

实现可拆解为以下 5 步：

1. oracle 后端选择不再短路：vllm/model_executor/layers/fused_moe/oracle/unquantized.py 删除了 if current_platform.is_cpu(): return UnquantizedMoeBackend.CPU, None 的早期返回，并在 backend_to_kernel_cls 中为 CPU 注册 [X86CPUUnquantizedExperts, ArmCPUUnquantizedExperts, CPUUnquantizedExperts]，按架构优先级选择。现在 CPU 与 CUDA/TPU/OOT 等共享同一套 is_supported_config 筛选流程。
2. 专家类迁移到 experts/cpu_moe.py：将 grouped_topk、select_experts 从被删除的 cpu_fused_moe.py 移入，并新增 CPUUnquantizedExperts(FusedMoEExpertsMonolithic) 基类，集中实现三件事：_supports_grouped_gemm 的形状校验（hidden/intermediate 必须满足 32 对齐）、_pad_moe_intermediate 的零填充（如 moe_intermediate_size=704 在 TP=4 时 176→192）、process_weights_after_loading 中从 layer 捕获 use_grouped_topk、renormalize、scoring_func、custom_routing_function 等 monolithic apply() 无法携带的路由参数，然后执行 cpu_prepack_moe_weight 预打包。子类仅覆盖 ISA 相关类属性与 kernel 选择。

3. 移除 UnquantizedFusedMoEMethod 的 CPU 逃生舱：unquantized_fused_moe_method.py 删除了 is_monolithic 属性对 CPU 的特殊返回，删除了 process_weights_after_loading 中 SGL/CPUFusedMOE 实例化分支；相应在 _setup_kernel 构建 moe_kernel 后，若 backend 为 CPU，则额外调用 self.moe_kernel.fused_experts.process_weights_after_loading(layer)，因为 convert_to_unquantized_kernel_format 只能看到两个权重张量，无法表达填充、预打包与路由捕获。
4. C++ 向量原语补全，使 grouped gemm 覆盖所有 ISA：这是 fadara01 在 review 中引入的关键演进——cpu_fused_moe_activations.hpp 将 swigluoai_and_mul 的去交错路径从 __aarch64__ 特判改为 AVX512 特化 + 通用标量路径，并把 DEFINE_FAST_EXP 封装为可回退的 DEFINE_CPU_FUSED_MOE_EXP；cpu_types_vxe.hpp、cpu_types_x86.hpp、cpu_types_vsx.hpp、cpu_types_riscv_impl.hpp、cpu_types_scalar.hpp 补充了 exp、tanh、er、clamp、一元 operator-、INT8Vec64 等原语，最终删除 torch/oneDNN per-expert 回退循环，全平台走 grouped gemm。
5. 测试、文档与 CI 配套：test_cpu_fused_moe.py 改为从 experts/cpu_moe.py 导入新专家类并新增 _ref_moe_activation、test_cpu_vec_fused_moe_shape_support（含 swigluoai 不可填充断言）、test_cpu_fused_moe_unaligned_intermediate_size_swigluoai；test_moe.py 删除旧的 test_cpu_fused_moe_basic；test_unquantized_backend_selection.py 新增 x86/ARM kernel 选择测试，并将 default-backend 测试的 CPU 参数改为 32 对齐尺寸；docs/design/moe_kernel_features.md、docs/getting_started/installation/cpu.md 同步更新。

关键文件：

- vllm/model_executor/layers/fused_moe/experts/cpu_moe.py（模块 专家内核；类别 source；类型 data-contract；符号 grouped_topk, select_experts, CPUUnquantizedExperts, X86CPUUnquantizedExperts）：核心迁移目标文件：未量化专家类、路由函数、形状校验、零填充与预打包逻辑全部集中于此，是全 PR 的枢纽。
- vllm/model_executor/layers/fused_moe/cpu_fused_moe.py（模块 旧实现；类别 source；类型 deletion；符号 CPUFusedMOE, SGLFusedMOE, grouped_topk, select_experts）：旧的整体路径实现被整体删除，是本次重构的主要清理对象，删除了 577 行。
- vllm/model_executor/layers/fused_moe/unquantized_fused_moe_method.py（模块 方法入口；类别 source；类型 data-contract；符号 is_monolithic, apply_monolithic, process_weights_after_loading, _setup_kernel）：移除 CPU 的三个逃生舱，CPU 路径并入与新架构一致的 kernel 初始化流程。
- vllm/model_executor/layers/fused_moe/oracle/unquantized.py（模块 后端选择；类别 source；类型 data-contract；符号 backend_to_kernel_cls, select_unquantized_moe_backend）：CPU 不再在 oracle 中短路，统一走 is_supported_config 选择链，是迁移的入口改动。
- csrc/cpu/cpu_fused_moe_activations.hpp（模块 C++ 内核；类别 source；类型 core-logic）：将 MoE 激活 kernel 从特定 ISA 特判改为全 ISA 可编译，是 torch 回退能被删除的编译基础。

- tests/kernels/moe/test_cpu_fused_moe.py（模块 测试；类别 test；类型 test-coverage；符号 _ref_moe_activation, _make_moe_config, test_cpu_vec_fused_moe_shape_support, test_cpu_fused_moe_unaligned_intermediate_size_swigluoai）：测试从旧 CPUFusedMOE 迁移到新专家类，新增形状支持与 swigluoai 非对齐断言，是本次行为变化最重要的回归保护。

关键符号：grouped_topk, select_experts, CPUUnquantizedExperts._supports_grouped_gemm, CPUUnquantizedExperts._padded_intermediate_size, CPUUnquantizedExperts.process_weights_after_loading, CPUUnquantizedExperts._pad_moe_intermediate, X86CPUUnquantizedExperts, ArmCPUUnquantizedExperts, UnquantizedFusedMoEMethod._setup_kernel, UnquantizedFusedMoEMethod.is_monolithic, backend_to_kernel_cls, CPUFusedMOE, SGLFusedMOE

关键源码片段

vllm/model_executor/layers/fused_moe/experts/cpu_moe.py

核心迁移目标文件：未量化专家类、路由函数、形状校验、零填充与预打包逻辑全部集中于此，是全 PR 的枢纽。

```
# experts/cpu_moe.py —— 未量化 CPU MoE 专家类的核心单元
# 三个子类共享同一套 形状校验 + 零填充 + 预打包 流程，
# 行为与旧 CPUFusedMOE 保持一致，但已按模块化结构组织。

class CPUUnquantizedExperts(mk.FusedMoEExpertsMonolithic):
    # 便携 vector grouped-gemm 路径：非 x86 / 非 ARM 架构的兜底
    isa = 'vec'
    output_alignment = 32 # grouped gemm kernel 按 32 分块，无尾处理
    reduction_alignment = 1

    @classmethod
    def _padded_intermediate_size(cls, moe_config: FusedMoEConfig) -> int:
        # swigluoai 的 gate/up 交错布局无法用零填充对齐，只能原样保留
        intermediate_size = moe_config.intermediate_size_per_partition
        if moe_config.activation == MoEActivation.SWIGLUOAI:
            return intermediate_size
        return round_up(intermediate_size, cls._intermediate_alignment())

    @classmethod
    def _supports_grouped_gemm(cls, moe_config: FusedMoEConfig):
        # oracle 通过 is_supported_config 调用，决定形状能否走 grouped gemm
        intermediate_size = cls._padded_intermediate_size(moe_config)
        if (moe_config.hidden_dim % cls.output_alignment != 0
                or intermediate_size % cls.output_alignment != 0):
            return False, (
                f'kernel requires hidden and intermediate dimensions '
                f'divisible by {cls.output_alignment}')
        return True, None
```

```
    def __init__(self, moe_config, quant_config):
        super().__init__(moe_config, quant_config)
        # apply() 的签名带不了路由参数，这里从 layer 抓取，
        # 保证 grouped-topk / sigmoid scoring / custom routing 行为不变
        self.use_grouped_topk = False
        self.renormalize = False
        self.scoring_func = 'softmax'
        self.custom_routing_function = None

    @property
    def expects_unquantized_inputs(self) -> bool:
        return True

    def process_weights_after_loading(self, layer: torch.nn.Module) -> None:
        self._pad_moe_intermediate(layer)
        self.use_grouped_topk = layer.use_grouped_topk
        self.renormalize = layer.renormalize
        self.scoring_func = layer.scoring_func
        self.custom_routing_function = layer.custom_routing_function
        # 预打包成 grouped gemm 的运行时布局（含 VNNI/NEON/VSX 格式）
        replace_parameter(layer, 'w13_weight',
                          cpu_prepack_moe_weight(layer.w13_weight, self.isa))
        replace_parameter(layer, 'w2_weight',
                          cpu_prepack_moe_weight(layer.w2_weight, self.isa))

    def _pad_moe_intermediate(self, layer: torch.nn.Module) -> None:
        # 例：moe_intermediate_size=704、TP=4 时每分区 176，不是 32 的倍数；
        # 零填充后 w13 / w2（及 bias）同步扩维，才能命中 grouped gemm 快速路径
        intermediate_size = self.moe_config.intermediate_size_per_partition
        padded_size = self._padded_intermediate_size(self.moe_config)
        if padded_size == intermediate_size:
            return
        num_experts, _, hidden_size = layer.w13_weight.shape
        new_w13 = layer.w13_weight.new_zeros(
            num_experts, 2 * padded_size, hidden_size)
        new_w13[:, :intermediate_size] = layer.w13_weight[:, :intermediate_size]
        new_w13[:, padded_size:padded_size + intermediate_size] = \
            layer.w13_weight[:, intermediate_size:]
        replace_parameter(layer, 'w13_weight', new_w13)
        # w2 为 [num_experts, hidden_size, padded_size]，同样零填充并替换
```

评论区精华

评审讨论围绕“是否保留 torch 回退”展开，最终方向在协作中发生变化：

- PowerPC prefill 性能回退（performance）：IBM 的 Akashcodes732 在 Power 平台用 google/gemma-4-26B-A4B-it（TP=4）实测后反馈：模型输出与重构前完全一致，但移除 TorchCPUUnquantizedExperts 后 PowerPC 走通用 VSX loop，替代了原先 OpenBLAS 支撑的重 prefill torch 路径，prefill 出现明显性能下降，建议保留 Torch 回退。

- fadara01 的主张（design）：他认为保留 torch 实现会让代码超过必要复杂度，且基于已有的 grouped gemm 结构，为 PowerPC 增加专用 MMA grouped GEMM 是低成本工作，并说明已咨询 IBM 的 R3hankhan123，对方确认 s390x 侧无异议。
- 结论：合并采用 grouped-gemm-only 方案，PowerPC 专用优化由 Akashcodes732 以独立后续 PR 推进，避免"先加回退、再删回退"的重复重构。最终评审中 fadara01 与 jikunshang 均 APPROVED。
  - 移除 Torch 回退后 PowerPC 的 prefill 性能退化 (performance): 保持 grouped-gemm-only 结构，PowerPC 专用 MMA grouped GEMM 优化由 Akashcodes732 以独立后续 PR 推进；R3hankhan123 已确认 s390x 侧无异议。
  - 为所有 ISA 启用 grouped gemm 并删除 torch fallback (design): 合并采用 grouped-gemm-only 方案；PR 描述中的 TorchCPUUnquantizedExperts 随之不再存在。
  - 合并冲突与 CI 重跑 (other): 冲突解决后 CI 通过并合并。

风险与影响

- 风险：
  - PowerPC prefill 性能回退：Akashcodes732 实测证实通用 VSX loop 不如 OpenBLAS torch 路径，对以 prefill 为主的 Power 平台负载有明显影响，需等待后续 MMA 优化补回。
  - x86 行为从"静默降级"变为"报错"：swigluoai 交错的 gate/up 布局在 per-partition intermediate 非 32 倍数时无法零填充，x86 上现在会直接报错并提示调整 --tensor-parallel-size，而非像以前在非 AMX 机器上静默走慢速 torch loop。
  - 非 x86 平台验证不足：Apple Silicon、RISC-V、POWERPC、s390x 等路径主要靠代码审查与个别厂家验证，统一 grouped gemm 后原语替换（exp、tanh、er、clamp 等）的数值一致性风险仍需更多实机回归。
  - 多 PR 文本重叠：PR 正文列出 #43653、#45480、#48430、#47778、#50116 与被删除 / 修改文件存在文本重叠，先后合并会产生 rebase 成本。
  - 核心权重加载路径重构：权重填充、预打包与路由参数捕获整体迁移到 process_weights_after_loading，任何对 layer 属性读取顺序的疏忽都可能影响 grouped-topk（DeepSeek）、sigmoid scoring 或 custom routing 类模型。
  - 影响：对用户：CPU 上 MoE 推理的 kernel 选择更统一，x86 上 FP16/FP32 在 AMX 机器从报错变为可用 vector kernel，性能与鲁棒性都有收益；但 swigluoai 非对齐形状在 x86 上会以明确报错替代静默回退，部分用户需调整 TP。对系统：cpu_fused_moe.py 删除后 MoE 激活与路由逻辑只保留一份实现，C++ 侧 vector 原语补全使 grouped gemm 可跨所有 CPU ISA 编译，长期降低维护成本。对团队：需要协调与 #43653、#45480、#48430 等并行 PR 的 rebase，PowerPC 后续 MMA 优化是明确的跟进项。
  - 风险标记：PowerPC prefill 性能回退，x86 swigluoai 非对齐形状由回退改为报错，非 x86 平台实机验证不足，多 PR 文本重叠需协调 rebase，核心权重加载路径重构

关联脉络

- PR #46901 [oracle refactor series]（标题未在上下文中提供）：PR 正文指出 #37753 oracle 重构系列（#44231、#44562、#37776、#46901）是本次迁移的前置工作，本 PR 是其中 CPU 部分的收尾。
- PR #43653 [PR 正文提及的文本重叠 PR]（标题未在上下文中提供）：为被删除的 cpu_fused_moe.py 添加 SWIGLUSTEP/RELU2 激活，后续需将功能落到 experts/cpu_moe.py。
- PR #45480 [PR 正文提及的文本重叠 PR]（标题未在上下文中提供）：同样 patch 了被删除的 cpu_fused_moe.py，与本 PR 存在文本冲突。
- PR #48430 [PR 正文提及的文本重叠 PR]（标题未在上下文中提供）：向 tests/kernels/moe/test_cpu_fused_moe.py 添加回归测试，需将 import 指向 experts.cpu_moe。
- PR #50116 [PR 正文提及的文本重叠 PR]（标题未在上下文中提供）：INT8 prepack cleanup 触及 experts/cpu_moe.py，与本次未量化路径改动需要协调 rebase。