

PR #55522 完整报告

vllm-project/vllm

[Refactor][ROCm] Migrate the RDNA3 W4A16 MoE to the oracle/experts pa...

合并时间: 2026-09-12 21:20

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55522>

执行摘要

- 一句话: RDNA3 MoE 迁入 oracle 结构并修复加载崩溃
- 推荐动作: 值得精读, 尤其是正在做 MoE 后端接入、oracle 迁移或 ROCm 量化内核工作的工程师——它是一份完整的「把私有量化后端迁到 oracle/experts 模式」的参照实现, 也直观展示了这种架构如何把平台门禁、量化兼容性判定、权重预处理三件事收拢到统一契约。阅读时建议重点看三处设计决策: (1) workspace_shapes() 的内存声明, 它是替换旧 layer.rdna3_*_buf 预分配缓冲后的新契约; (2) finalize_weight_and_reduce_impl() 返回 TopKWeightAndReduceNoOP() 所依赖的内核 output_topk 语义; (3) _backend_incompatibility_reason() 中的四条拒绝条件, 它直接决定了哪些 checkpoint 会静默回退 Triton。如果你维护 gfx1100 回归, 建议把非对称、channel-wise、LoRA 三种回退场景补进手动回归清单, 并顺手处理 coderabbit 指出的三个空跑用例; 若你负责 MoE 加载路径, routed_experts.py 的类名白名单值得单独排期改造, 否则同类 KeyError 会再次出现。

功能与动机

动机是回应 #44075 落地 RDNA3 W4A16 MoE 内核时 @BowenBao 在 review 中提出的重构要求 (PR body 原文: this is the follow-up refactor @BowenBao asked for when the RDNA3 W4A16 MoE kernel landed in #44075)。原实现通过一个私有 quant-method 类 CompressedTensorsWNA16RDNA3MoEMethod 加 rocm_moe_rdna.py 这个 48 行的私有 dispatcher、再配合 compressed_tensors_moe.py 内联分支接入, 与仓库既有的「用类结构统一 MoE oracle」方向不一致 (#37753 明确要求把 select/convert/make_kernel 逻辑收敛到统一类结构, 本 PR 由 #44460 跟踪)。同时该路径阻塞在 #44570 (CompressedTensorsWNA16MoEMethod 合并 Marlin 变体) 之上, 后者合入后本 PR 才可落地。作者还在 body 中点出一个顺带修复: RoutedExperts 依据 quant method 类名决定行为, 子类不在名单上导致 gfx1100 上 W4A16 MoE 模型加载即 KeyError: 'intermediate_size_full', 因为没有 gfx1100 CI 而长期无人发现。

实现拆解

1. 新增标准 experts 实现 (vllm/model_executor/layers/fused_moe/experts/rdna3_moe.py, +235): 把原 _rdna3_fused_moe 的两次 moe_gptq_gemm_rdna3 调用重构进 Rdna3WNA16Experts.apply(), 并补齐 modular kernel 契约——moe_problem_size() 针对 K 维在前的打包布局 (w2.size(1) * 8 反推 hidden) 重写、workspace_shapes() 声明

workspace2 必须容纳 gate_up 输出与激活输出、finalize_weight_and_reduce_impl() 返回 TopKWeightAndReduceNoOP() (top-k 归约被内核的原子写回吸收)。能力声明拆成 _supports_current_device / _supports_quant_scheme / _supports_activation / _supports_parallel_config, 由 oracle 统一询问, 取代原先挂在 rocm_moe_rdna.is_supported() 上的单点判断。

2. oracle 侧接线 (vllm/model_executor/layers/fused_moe/oracle/int_wna16.py, +110)
: 新增 WNA16MoEBackend.RDNA3 枚举与 backend_to_kernel_cls() 的延迟导入分支;
把 RDNA3 置于 _get_priority_backends() 列表首位, 靠设备门自门控;
map_wna16_backend() 增加 'rdna3' 映射从而打通 --moe-backend rdna3;
make_wna16_moe_kernel() 把 Rdna3WNA16Experts 加入 allowed experts, 使内核只能经 oracle 抵达。
3. 不兼容判定集中化: _backend_incompatibility_reason() 新增 RDNA3 分支, 逐条拒绝非 compressed-tensors 配置、非对称 (may_have_zp)、expert bias、group actorder、非 QuantizationStrategy.GROUP 的 scale 策略, 并以字符串原因返回给 oracle, 由其静默回退到下一个可用后端 (Triton / Marlin)。这一层把「哪些 checkpoint 能进 HIP 内核」从散落的隐式条件变成显式契约。
4. 权重预处理下沉: 新增 _synthesize_rdna3_qzeros() (复刻 GPTQv1 的 +1 怪癖, 编码 bias - 1 = 7) 与 _process_weights_rdna3() (逐 expert 做 ops.gptq_shuffle、合成对称零点并 expand 到全部 expert), 在 convert_to_wna16_moe_kernel_format() 中按 backend 分派, 与 _process_weights_cpu / marlin 系列并列; 权重、scale、零点的一次性准备工作由量化类搬到 oracle 后, compressed_tensors_moe_wna16.py 只需把 RDNA3 与 CPU 一起纳入「对称量化也要 ZP」的断言。
5. 删除旧接线: rocm_moe_rdna.py (is_supported / make_method, -48) 与 compressed_tensors_moe_wna16_rdna3.py (CompressedTensorsWNA16RDNA3MoEMethod、_synthesize_qzeros、_rdna3_fused_moe, -260) 整体删除, compressed_tensors_moe.py 移除内联 ROCm 分支 (-7)。删除后 ROCm 上的 WNA16 MoE 只剩 oracle 一条选路, 重复的权重准备逻辑不再存在。
6. 崩溃修复与配置配套: routed_experts.py 的 weight_loader 类名白名单删除已不存在的 CompressedTensorsWNA16RDNA3MoEMethod; 新后端复用父类名后走通用路径, 作者描述的 gfx1100 KeyError: 'intermediate_size_full' 随之消失。vllm/config/kernel.py 把 'rdna3' 加入合法 MoE backend 列表并补充说明 (ROCm gfx1100 only), 否则 --moe-backend rdna3 会被配置校验拒绝。
7. 测试配套: tests/kernels/quantization/test_rdna3_compile_guards.py 把断言对象从已删除的 rocm_moe_rdna.is_supported 换成 rdna3_moe_kernel_available(), Rdna3WNA16Experts._supports_current_device() 与 _supports_quant_scheme(), 并新增 TestWNA16OracleWiring 验证 backend↔experts 映射及 RDNA3 优先级早于 Triton; tests/quantization/test_moe_wna16.py 新增三条 RDNA3 拒绝用例 (AutoGPTQ 布局、asymmetric、channel-wise)。HIP 内核与 tests/kernels/quantization/test_rdna3_moe_w4a16.py 未改动。

行为	迁移前	迁移后
非对称 checkpoint	内核合成对称零点，静默忽略 checkpoint 零点	oracle 判为不兼容，回退 Triton
LoRA 适配	直接丢弃 LoRA delta	回退 Triton，保留 delta
swiglu clamp / alpha / beta	未生效	经共享激活实现生效
channel-wise / 非 GROUP 策略	可能被 RDNA3 接管	明确回退 Triton

关键文件：

- `vllm/model_executor/layers/fused_moe/experts/rdna3_moe.py`（模块 专家内核；类别 source；类型 core-logic；符号 `rdna3_moe_kernel_available`, `Rdna3WNA16Experts`, `init`, `activation_format`）：本 PR 新增的核心文件：`Rdna3WNA16Experts` 承接原 `_rdna3_fused_moe` 的全部计算逻辑与模块化内核契约（`moe_problem_size / workspace_shapes / finalize_weight_and_reduce_impl`），并提供 `rdna3_moe_kernel_available()` 作为统一设备门禁；后续 RDNA4/CDNA 内核的接入模板。
- `vllm/model_executor/layers/fused_moe/oracle/int_wna16.py`（模块 内核选路；类别 source；类型 data-contract；符号 `WNA16MoEBackend.RDNA3`, `backend_to_kernel_cls`, `_get_priority_backends`, `_backend_incompatibility_reason`）：`WNA16 oracle` 是本次迁移的接收端：新增 RDNA3 枚举、kernel 类映射、优先级列表、四条不兼容判定，以及 `_synthesize_rdna3_qzeros / _process_weights_rdna3` 权重预处理，是「选路 + 权重格式契约」的集中落点。
- `vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/compressed_tensors_moe_wna16_rdna3.py`（模块 量化方法；类别 source；类型 deletion；符号 `synthesize_qzeros`, `CompressedTensorsWNA16RDNA3MoEMethod`, `process_weights_after_loading`, `apply`）：被删除的私有实现（-260）：`CompressedTensorsWNA16RDNA3MoEMethod`、`_synthesize_qzeros`、`_rdna3_fused_moe` 以及挂在 layer 上的 `rdna3*` 预分配缓冲全部移除，是理解「迁移走了什么」的对照面。
- `vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/compressed_tensors_moe_wna16.py`（模块 量化方法；类别 source；类型 data-contract；符号 `CompressedTensorsWNA16MoEMethod.process_weights_after_loading`）：零点断言条件从「仅 CPU」扩展为「CPU 与 RDNA3」，是 RDNA3 在对称量化下依赖 oracle 合成 ZP 的关键耦合点，配合 `_process_weights_rdna3` 才能正确写回权重。
- `tests/kernels/quantization/test_rdna3_compile_guards.py`（模块 编译守卫；类别 test；类型 test-coverage；符号 `TestMoEDispatchMocked.test_kernel_unavailable_when_mocked_cdna`, `test_quant_scheme_rejects_non_symmetric_int4`, `test_quant_scheme_accepts_symmetric_int4`, `test_backend_maps_to_rdna3_experts`）：门禁测试的断言对象整体切换到新的 `experts` 类与 `oracle` 选路，并新增

TestWNA16OracleWiring 锁定 backend↔experts 映射与优先级，是本次重构的行为回归防线。

- vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/rocm_moe_rdna.py (模块 内核调度; 类别 source; 类型 deletion; 符号 is_supported, make_method) : 被删除的私有 dispatcher (-48) : is_supported / make_method 是原先 RDNA3 内核接入的唯一入口，删除后 ROCm 平台专属分支从 compressed_tensors 层彻底消失。
- vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/compressed_tensors_moe.py (模块 分发入口; 类别 source; 类型 core-logic; 符号 get_moe_method) : 移除内联的 ROCm/RDNA3 分支，使 get_moe_method 回到纯平台无关的量化选型逻辑，是「分发收敛到 oracle」的直接证据。
- vllm/model_executor/layers/fused_moe/routed_experts.py (模块 权重加载; 类别 source; 类型 bugfix; 符号 RoutedExperts.weight_loader) : 类名白名单删除 CompressedTensorsWNA16RDNA3MoEMethod，是作者所述 KeyError: 'intermediate_size_full' 崩溃修复在 diff 中的落点，也暴露了基于类名字符串分派的脆弱性。
- tests/quantization/test_moe_wna16.py (模块 量化测试; 类别 test; 类型 test-coverage; 符号 test_wna16_oracle_rejects_incompatible_quant_structures) : 新增三条 RDNA3 不兼容用例 (AutoGPTQ 布局、asymmetric、channel-wise)，把 _backend_incompatibility_reason 的新契约固化成测试。
- vllm/config/kernel.py (模块 后端配置; 类别 source; 类型 configuration; 符号 KernelConfig.moe_backend) : 把 'rdna3' 加入合法 MoE backend 列表并补充文档说明，使用户侧 --moe-backend rdna3 可以真正生效。

关键符号: rdna3_moe_kernel_available, Rdna3WNA16Experts.apply, Rdna3WNA16Experts.workspace_shapes, Rdna3WNA16Experts.moe_problem_size, Rdna3WNA16Experts._supports_quant_scheme, Rdna3WNA16Experts.finalize_weight_and_reduce_impl, _get_priority_backends, _backend_incompatibility_reason, backend_to_kernel_cls, map_wna16_backend, make_wna16_moe_kernel, _synthesize_rdna3_qzeros, _process_weights_rdna3, convert_to_wna16_moe_kernel_format, CompressedTensorsWNA16MoEMethod.process_weights_after_loading, RoutedExperts.weight_loader

关键源码片段

vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/compressed_tensors_moe_wna16.py

零点断言条件从「仅 CPU」扩展为「CPU 与 RDNA3」，是 RDNA3 在对称量化下依赖 oracle 合成 ZP 的关键耦合点，配合 _process_weights_rdna3 才能正确写回权重。

```
# vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/
compressed_tensors_moe_wna16.py
# process_weights_after_loading() 中的零点断言条件变化
```

```
# CPU 的 fused_experts_cpu 与 RDNA3 HIP 内核即使面对对称量化也需要零点
```

```
# (RDNA3 的零点由 oracle 的 _process_weights_rdna3 合成)，而 EMULATION
# 会把零点烘进反量化后的 bf16 权重，所以要从断言里排除。
if (
    not self.symmetric
    or self.wna16_backend in (WNA16MoEBackend.CPU, WNA16MoEBackend.RDNA3)
) and self.wna16_backend != WNA16MoEBackend.EMULATION:
    assert w13_qzeros is not None and w2_qzeros is not None
    # 随后把 w13 / w2 的零点 replace_parameter 写回 layer
```

评论区精华

本 PR 的 review 讨论非常轻 (`review_comments_count = 0`，仅 4 条 issue 评论)，结论性意见集中在批准与自测说明上：

BowenBao (#44075 reviewer、本次重构的提出者)：Thanks @JartX! LGTM

fxmarty-amd：LGTM

tjtanaa：批准并执行合入 (merged_by)

唯一的实质性技术反馈来自 coderabbitai 的 nitpick：tests/kernels/quantization/test_rdna3_compile_guards.py 中 TestMoEDispatchMocked 的三个用例在非 ROCm 机器上「空跑」——rdna3_moe_kernel_available() 在 current_platform.is_rocm() 处就返回 False，根本不会读到被 mock 的 on_gfx1100 与 torch.ops，断言退化为 False is False；建议给该类补上文件里已有的 gfx1100_only 或 ROCm skip marker。该建议未见修复记录，PR 直接合入。

流程侧：claude[bot] 因 fork PR 自动审查被禁用 (This pull request is from a fork — automated review is disabled)，mergify 提示过合并冲突 (作者随后 merge main 产生第二个 commit)，AndreasKaratzas 触发 /ci run，Buildkite CI #88379 在 commit 8b31bef89a20 上运行。整体上，对一个改动内核分发路径、含三处行为变化的 PR，审查强度主要依赖作者自测与三位维护者的 LGTM。

- 非 ROCm 平台上 TestMoEDispatchMocked 用例空跑 (testing)：建议被标记为 Trivial nitpick，PR 合入时未见修复记录；对新 experts 类的门禁测试仍存在「纸面覆盖」。
- 维护者对 oracle/experts 迁移方向的确认 (design)：迁移方案获得原始提出者与两位维护者认可，被视为 #37753 统一 oracle 结构的落地样板。
- fork PR 自动审查缺位与合并流程摩擦 (other)：审查主要依赖三位维护者的 LGTM 与作者本地 gfx1100 验证，过程性风险 (冲突 rebase、CI 手动触发) 已解决但审查深度有限。

风险与影响

- 风险：

1. 缺少 gfx1100 自动化覆盖 (最高风险)：本 PR 修复的 KeyError: 'intermediate_size_full' 正是因为没有 gfx1100 CI 才长期潜伏；迁移后的 Rdna3WNA16Experts 路径同样只靠作者本地手跑 tests/kernels/quantization/test_rdna3_moe_w4a16.py 与 2xRX 7900 XTX 服务验证，test_rdna3_compile_guards.py 只覆盖门禁与选路，不覆盖端到端数值。

2. 类名白名单耦合未根除: `vllm/model_executor/layers/fused_moe/routed_experts.py` 仍以 `quant method` 类名字符串判断是否走 `intermediate_size_full` 路径, 本次只是让 RDNA3 退回父类名而避开; 未来任何新的 MoE 量化子类都可能重演同类加载失败。
3. 回退面变化带来的可感知影响: 非对称、channel-wise (非 GROUP strategy)、LoRA 三类配置现在一律落到 Triton。语义上更正确 (不再静默忽略零点或丢弃 delta), 但用户在 `gfx1100` 上的性能特征会改变; 作者只验证了对称 `int4 / group 32` 场景吞吐不变 (~385 tok/s), 回退场景没有基准数据。
4. 优先级首位带来的默认路径切换: `_get_priority_backends()` 把 RDNA3 排在 `FLASHINFER_TRITLLM / MARLIN` 之前, `gfx1100` 上所有满足 `_supports_*` 的 W4A16 MoE 默认走 HIP 内核; 若门禁条件放宽而内核未同步演进, 可能出现静默的数值错误而非报错。
5. 隐式的 workspace 契约: `workspace_shapes()` 返回值必须精确覆盖 `M * topk * (gate_up + N)`, 因为输出缓冲区由 modular kernel 从 `workspace13` 切出、GEMM-1 的两个缓冲区复用 `workspace2`; 尺寸声明与实际切分不一致会直接越界或写坏相邻缓冲, 且没有任何断言兜底。
6. TopKWeightAndReduceNoOP 的正确性前提: 归约完全依赖 `output_topk=top_k` 在内核内完成, 并在 `apply()` 中对 `output` 先做 `zero_()`; 一旦内核的 `output_topk` 语义或调用侧 `apply_router_weight_on_input` 分支被改动, 错误会静默传播。
7. 测试空跑问题未处理: `coderabbit` 指出的三个 mock 用例在非 ROCm 上不真正覆盖逻辑, 使门禁回归的「纸面覆盖」高于实际覆盖。- 影响: 用户侧: `gfx1100` (RX 7900 XTX 等 RDNA3 消费卡) 用户从「W4A16 MoE 模型加载即崩」恢复到可正常加载运行, 并首次可以用 `--moe-backend rdna3` 强制指定该内核; 非对称、channel-wise、LoRA 用户得到的是 Triton 回退 (结果更正确, 但性能曲线可能与旧行为不同)。系统侧: ROCm 上 WNA16 MoE 的选路收敛到 `oracle/int_wna16.py` 单一入口, `compressed_tensors_moe.py` 不再持有平台专属内联分支, 权重准备逻辑从两份 (量化类内 + `oracle` 内) 收敛为一份, 后续新增 RDNA4/CDNA 内核只需补 `backend + experts` 类。日志格式随之变化: 旧的 `Using CompressedTensorsWNA16RDNA3MoE Method (native RDNA3 HIP kernel)` 变为 `oracle` 的 `Using 'RDNA3' WNA16 MoE backend` 与 `Using Rdna3WNA16Experts`, 依赖旧日志的排障脚本或文档需要同步。团队侧: 这是 #37753 「统一 MoE oracle 类结构」在 ROCm 后端的落地样板, BowenBao 的批准意味着该模式获得原始提出者确认; 代价是 `gfx1100` 的验证责任继续落在个别贡献者的本地硬件上, 缺少 CI 兜底。- 风险标记: 缺少 `gfx1100` CI 覆盖, 类名白名单耦合未根除, 三处行为变化 (非对称 /LoRA/ 激活), 默认后端优先级变更, `workspace` 尺寸契约无断言, 测试用例空跑

关联脉络

- PR #44570 [MoE Refactor] Combine CompressedTensorsWNA16MarlinMoEMethod with CompressedTensorsWNA16MoEMethod: 本 PR 被它阻塞 (body 原文 blocked on #44570, which is merged), 两者同属 WNA16 oracle 的后端选型整合线: 它先把 Marlin 变体并入 CompressedTensorsWNA16MoEMethod, 本 PR 才能把 RDNA3 也退化为父类名并交由 `oracle` 调度。

- PR #44075 RDNA3 (gfx1100) 融合 W4A16 MoE 内核落地 PR (PR body 引用, 非精确标题): 本 PR 是该内核合入时 @BowenBao 要求的后续重构; body 明确 the RDNA3 W4A16 MoE kernel added in #44075 was wired in as a bespoke quant-method subclass, 本 PR 只改分发管线、HIP 内核与其测试保持不变。
- PR #54985 [Elastic EP] Reuse CUDA graphs across reconfiguration: 同属 vllm/model_executor/layers/fused_moe 子系统的专家 / 后端与并行配置改造 (涉及 all2all_utils.py、prepare_finalize/nixl_ep.py、config/parallel.py), 与本 PR 一样通过能力声明约束内核可用性; 关联为同包同模式的工程脉络, 而非共改同一文件。