PR #43615 完整报告

vllm-project/vllm

[ROCm] Enable AITER and FP8 inference on GFX120x

合并时间：2026-08-04 05:45

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/43615

执行摘要

- 一句话：GFX120x 启用 AITER 与 FP8 快速路径，MI3xx 行为保持不变
- 推荐动作：值得精读。重点学习三点：(1) 用「gfx9 CK 伞」与「gfx12 Triton 伞」分离 AITER 能力探测的手法，避免把不存在的内核暴露给新架构；(2) can_implement() 在 classmethod 阶段用 tuned 列表做 shape 校验，从而把调度决策前移的设计；(3) tuned shape 集合按架构拆分、避免交叉污染的做法。另建议 follow-up 补充针对 on_rdna4() 与 tuned shape 的自动化测试。

功能与动机

PR body 明确指出：Previously several AITER gates were effectively MI3xx-only, so gfx12 fell back to the generic ROCm/Triton paths even when AITER Triton support was available。即 gfx12 用户此前无法使用 AITER 为 RDNA 调优过的 Triton 内核，被迫回退到更慢的通用路径。本 PR 要解决的是「让 gfx12 使用 AITER 的 Triton 快速路径，同时绝不 dispatch 进 gfx12 上不存在的 CK 内核」，并在此前提下扩展 FP8 支持、调整 attention 后端顺序。

实现拆解

实现按以下 5 步展开：

1. 平台能力层（vllm/platforms/rocm.py）：新增 _ON_RDNA4（gfx1200/gfx1201）与 on_rdna4()；supports_fp8() 由 on_cdna() or on_gfx12x() 收紧为 on_cdna() or on_rdna4()，避免把纯 RDNA（gfx11）误判为 FP8 设备；_get_backend_priorities() 在 rdna4 上通过 backends.insert(0, ROCM_AITER_UNIFIED_ATTN) 把 AITER 注意力提到最前；get_default_ir_op_priority() 在 rdna4 上不再默认启用 aiter rms_norm。
2. AITER 能力探测（vllm/_aiter_ops.py）：新增 is_aiter_found_and_supported_on_rdna4()（只查平台 + 架构 + 库存在），以及 rocm_aiter_ops.is_rdna_aiter_enabled()、is_rdna_linear_enabled()、is_rdna_gdn_triton_kernels_available()——这些函数刻意不经过 gfx9 的 @if_aiter_supported 装饰器，只暴露 rdna4 可用的 Triton 路径；把 GDN 内核探测抽成静态方法 _gdn_triton_kernels_importable() 供 gfx9/rdna4 复用；register_ops_once() 的守卫改为 is_aiter_found_and_supported() or is_aiter_found_and_supported_on_rdna4()；is_triton_gemm_w8a8_tuned() 拆成 gfx950_tuned 与 rdna4_tuned 两个集合，rdna4 新增 15 个 (N, K) 调优 shape。

3. FP8 线性内核调度（scaled_mm/aiter.py、scaled_mm/pytorch.py）：AiterFp8BlockScaledMMKernel.is_supported() 增加 is_rdna_linear_enabled() 分支；can_implement() 在 rdna4 启用时对不在 tuned 列表的 (N, K) 返回 False，让调度器回退到 TritonFp8BlockScaledMMKernel；PyTorchScaledMMLinearKernel.is_supported() 使用 current_platform.supports_fp8() 替换 MI3xx 硬编码。
4. MoE 后端选择（oracle/fp8.py、oracle/unquantized.py、experts/fused_batched_moe.py）：显式请求 AITER MoE 时若处于 rdna4，将 Fp8MoeBackend.AITER 从 AVAILABLE_BACKENDS 移除，避免进入不存在的 CK 实现；_supports_quant_scheme() 把 RDNA4 纳入 FP8 设备支持条件，与 TritonExperts 保持一致。
5. 编译融合 pass（rocm_aiter_fusion.py、pass_manager.py、qwen_gdn_linear_attn.py）：rdna4 上 match_aiter_quant_op = False，所有 RMSNorm/quant 融合 pattern 改用 native quant 匹配（gfx12 无 AITER quant 自定义算子）；pass_manager 中 ROCm AITER 系列 pass 的启用条件补上 is_rdna_aiter_enabled()；GDN 线性注意力的 Triton kernel 可用性合并 rdna4 分支。

测试配套：本 PR 未新增自动化测试文件。验证主要靠 benchmarks/attention_benchmarks/benchmark.py 的 attention 对比与 vllm bench serve 的端到端 serving；review 中 reviewer 曾追问 GFX120x 上的 AITER 测试覆盖，作者确认跑过 e2e。

关键文件：

- vllm/_aiter_ops.py（模块 算子层；类别 source；类型 core-logic；符号 is_aiter_found_and_supported_on_rdna4, is_rdna_aiter_enabled, is_rdna_linear_enabled, _gdn_triton_kernels_importable）：核心能力探测与开关所在，新增 rdna4 系列判定函数并拆分 gfx950/rdna4 tuned shape 集合，是本次功能开关的主控点。
- vllm/platforms/rocm.py（模块 平台层；类别 source；类型 core-logic；符号 on_rdna4, supports_fp8, _get_backend_priorities, get_default_ir_op_priority）：平台层新增 RDNA4 判定并调整 FP8 支持、attention 后端优先级与 RMSNorm 默认值，是架构分流的基石。
- vllm/model_executor/kernels/linear/scaled_mm/aiter.py（模块 线性算子；类别 source；类型 core-logic；符号 AiterFp8BlockScaledMMKernel.is_supported, AiterFp8BlockScaledMMKernel.can_implement）：FP8 block-scaled linear 内核的 is_supported/can_implement 是 gfx12 能否走 Triton 快速路径的关键阀门。
- vllm/compilation/passes/fusion/rocm_aiter_fusion.py（模块 编译融合；类别 source；类型 core-logic；符号 RocmAiterRMSNormQuantFusionPass.init, AiterSiluMulFp8GroupQuantPattern.init）：编译融合 pass 在 rdna4 上改用 native quant 匹配，避免将 gfx12 不支持的 AITER quant 自定义算子做为替换目标。
- vllm/compilation/passes/pass_manager.py（模块 编译调度；类别 source；类型 core-logic；符号 configure）：ROCm AITER 系列 fusion pass 的启用条件从 is_enabled() 扩展到包含 rdna4，否则 gfx12 上不会注册对应优化。
- vllm/model_executor/kernels/linear/scaled_mm/pytorch.py（模块 线性算子；类别 source；类型 data-contract；符号 is_supported）：通用 PyTorch scaled_mm 内核的

is_supported 放宽到 RDNA4，是 gfx12 上 untuned shape 的兜底路径。

- vllm/model_executor/layers/fused_moe/oracle/fp8.py（模块 MoE 路由；类别 source；类型 data-contract）：FP8 MoE 后端选择在 rdna4 上跳过 AITER，避免分发到 gfx12 不存在的 CK 实现。
- vllm/model_executor/layers/fused_moe/oracle/unquantized.py（模块 MoE 路由；类别 source；类型 data-contract）：与 fp8.py 同步的未量化 MoE 后端选择调整，保证统一行为。
- vllm/model_executor/layers/fused_moe/experts/fused_batched_moe.py（模块 MoE 专家；类别 source；类型 data-contract；符号 _supports_quant_scheme）：MoE batched experts 的 FP8 量化方案支持判定加入 RDNA4，与 TritonExperts 保持一致。
- vllm/model_executor/layers/mamba/gdn/qwen_gdn_linear_attn.py（模块 GDN 注意力；类别 source；类型 data-contract；符号 GDN_AITER_TRITON_AVAILABLE）：GDN 线性注意力的 AITER Triton 内核可用性检查合并 rdna4 分支，影响 Qwen 系列 GDN 模型的 decode fast-path。
- vllm/model_executor/layers/sparse_attn_indexer.py（模块 稀疏索引；类别 source；类型 data-contract）：AITER 稀疏注意力索引器的平台守卫微调，材料中未提供具体 diff，改动量小。
- vllm/v1/attention/ops/rocm_aiter_mla_sparse.py（模块 稀疏注意力；类别 infra；类型 infrastructure）：MLA 稀疏注意力算子的平台判定微调，影响面较小。

关键符号：is_aiter_found_and_supported_on_rdna4, is_rdna_aiter_enabled, is_rdna_linear_enabled, _gdn_triton_kernels_importable, is_rdna_gdn_triton_kernels_available, is_triton_gemm_w8a8_tuned, register_ops_once, on_rdna4, supports_fp8, _get_backend_priorities, get_default_ir_op_priority, AiterFp8BlockScaledMMKernel.is_supported, AiterFp8BlockScaledMMKernel.can_implement, RocmAiterRMSNormQuantFusionPass.init, AiterSiluMulFp8GroupQuantPattern.init

关键源码片段

vllm/_aiter_ops.py

核心能力探测与开关所在，新增 rdna4 系列判定函数并拆分 gfx950/rdna4 tuned shape 集合，是本次功能开关的主控点。

```
# vllm/_aiter_ops.py
def is_aiter_found_and_supported_on_rdna4() -> bool:
    """RDNA4 (gfx12) 的 AITER 可用性判定。

    gfx12 没有 AITER CK 构建，所以这里刻意不挂在 gfx9 的
    `@if_aiter_supported` 伞下，只报告 AITER Triton 内核是否可用。
    与 gfx9 对应函数一致：只查平台 + 架构 + 库存在，不查环境变量。
    """
    if current_platform.is_rocm() and IS_AITER_FOUND:
        from vllm.platforms.rocm import on_rdna4
```

```
        return on_rdna4()
    return False


class rocm_aiter_ops:
    # gfx9 上 is_enabled() 走 @if_aiter_supported；rdna4 需要独立的开关入口。
    @classmethod
    def is_rdna_aiter_enabled(cls) -> bool:
        """RDNA4 上 AITER 是否启用：库存在 + 架构为 rdna4 + 用户开启 AITER。

        只控制 rdna4 使用的 Triton 路径，是 is_enabled() 的 gfx12 analog。
        """
        if not current_platform.is_rocm() or not IS_AITER_FOUND:
            return False
        from vllm.platforms.rocm import on_rdna4

        return on_rdna4() and cls._AITER_ENABLED

    @classmethod
    def is_rdna_linear_enabled(cls) -> bool:
        """RDNA4 上 AITER Triton blockscale 线性层是否启用。"""
        return cls.is_rdna_aiter_enabled() and cls._LINEAR_ENABLED
```

vllm/platforms/rocm.py

平台层新增 RDNA4 判定并调整 FP8 支持、attention 后端优先级与 RMSNorm 默认值，是架构分流的基石。

```
# vllm/platforms/rocm.py
# 模块加载时一次性解析 GCN arch（amdsmi 优先，避免 CUDA 初始化）
_ON_RDNA4 = any(arch in _GCN_ARCH for arch in ["gfx1200", "gfx1201"])


def on_rdna4() -> bool:
    """当前 GPU 是否 RDNA4（gfx1200 / gfx1201）。"""
    return _ON_RDNA4


# gfx12 上 AITER UNIFIED 注意力要优先于原生 ROCM_ATTN，因此用 insert(0) 提到最前
def _get_backend_priorities(use_mla: bool, use_sparse: bool,
                            use_kv_connector: bool = False) -> list[AttentionBackendEnum]:
    from vllm._aiter_ops import is_aiter_found_and_supported, rocm_aiter_ops

    # ... sparse / MLA 分支略 ...
    backends = []
    if not use_kv_connector:
        backends.append(AttentionBackendEnum.ROCM_ATTN)
    if rocm_aiter_ops.is_mha_enabled():
        backends.append(AttentionBackendEnum.ROCM_AITER_FA)
    if is_aiter_found_and_supported():
```

```
        # gfx9：追加到 ROCM_ATTN 之后
        backends.append(AttentionBackendEnum.ROCM_AITER_UNIFIED_ATTN)
    elif rocm_aiter_ops.is_rdna_aiter_enabled():
        # gfx12：AITER 支持时放到列表最前，优先于 ROCM_ATTN
        backends.insert(0, AttentionBackendEnum.ROCM_AITER_UNIFIED_ATTN)
    backends.append(AttentionBackendEnum.TRITON_ATTN)
    backends.append(AttentionBackendEnum.TURBOQUANT)
    return backends
```

vllm/model_executor/kernels/linear/scaled_mm/aiter.py

FP8 block-scaled linear 内核的 is_supported/can_implement 是 gfx12 能否走 Triton 快速路径的关键阀门。

```
# vllm/model_executor/kernels/linear/scaled_mm/aiter.py
class AiterFp8BlockScaledMMKernel(Fp8BlockScaledMMLinearKernel):
    @classmethod
    def is_supported(cls, compute_capability=None):
        # gfx9 走 is_linear_enabled（CK/Triton 皆可），gfx12 走 rdna 分支（仅 Triton）
        if (
            rocm_aiter_ops.is_linear_enabled()
            or rocm_aiter_ops.is_rdna_linear_enabled()
        ):
            return True, None
        return (
            False,
            "Only supported on ROCm platform with aiter package installed.",
        )

    @classmethod
    def can_implement(cls, config: FP8ScaledMMLinearLayerConfig):
        can_implement_base, reason = super().can_implement(config)
        if not can_implement_base:
            return can_implement_base, reason

        act_quant_desc = config.activation_quant_key.scale
        if act_quant_desc.group_shape != GroupShape(1, 128):
            return (
                False,
                "Supports only dynamic per token group activation "
                "quantization with group_shape=(1,128).",
            )

        # gfx12 只有 AITER Triton blockscale 后端，该后端要求 (N, K) 已在调优列表。
        # 未调优的 shape 直接拒绝，让上层调度器回退到通用内核。
        if rocm_aiter_ops.is_rdna_linear_enabled():
            n, k = config.weight_shape
            if not rocm_aiter_ops.is_triton_gemm_w8a8_tuned(n, k):
                return (
                    False,
```

```
                f"(N={n}, K={k}) is not in the aiter Triton blockscale tuned list.",
            )
        return True, None
```

评论区精华

> gemini-code-assist[bot]（critical）：Importing vllm.platforms.rocm unconditionally during environment variable evaluation will cause a crash on non-ROCm platforms... 在 enable_envs_cache() 启动阶段，无守卫导入 vllm.platforms.rocm 会在 NVIDIA 上因访问 gcnArchName 抛 AttributeError。→ 作者回复 "fixed"，随后 dllehr-amd 建议改为 helper 函数而非 lambda 内嵌导入，作者再次采纳。
>
> tjtanaa：is_linear_enabled 与 is_triton_linear_enabled 冗余，建议复用前者；is_triton_gemm_w8a8_tuned 的改动会直接影响 CDNA4 (gfx950)，必须把 rdna4 与 gfx950 的条件拆开。→ 作者全部 "Fixed"，最终实现中 is_triton_linear_enabled 被移除，tuned 集合按架构拆分。
>
> tjtanaa：on_gfx950 已经是函数，不需要再包一层 arch_only 抽象，并且要用 if current_platform.is_rocm(): 守卫从 vllm.platforms.rocm 的导入。→ 作者回复已改为字符串参数懒解析并加平台守卫（最终head中装饰器已不存在，说明已回归简单函数调用）。
>
> dllehr-amd：can_implement 里重复调用 is_triton_gemm_w8a8_tuned 是否必要？可以直接用 __init__ 里已有的 use_triton 标志。→ 作者解释：can_implement() 是 classmethod，在 kernel 实例创建前调用，实例级 use_triton 尚不可用，因此需要独立校验。该讨论无进一步修改，最终 approve。
>
> AndreasKaratzas：fused_batched_moe.py 的 FP8 支持扩展可能需要 upstream maintainer 批准。→ 作者指出该判定与 TritonExperts 中已合入的做法一致。

- envs.py 无条件导入 rocm 导致非 ROCm 启动崩溃 (correctness): 作者回复 "fixed"；dllehr-amd 随后建议用 helper 函数替代 lambda，作者再次采纳。
- is_triton_gemm_w8a8_tuned 直接影响 gfx950 行为 (correctness): 作者 Fixed，最终实现中 gfx950_tuned 与 rdna4_tuned 分开构造，rdna4 集合通过并集扩展。
- is_linear_enabled 与 is_triton_linear_enabled 冗余 (design): 作者 Fixed，最终实现中 is_triton_linear_enabled 被移除，仅保留 is_linear_enabled 与 is_rdna_linear_enabled。
- arch_only 装饰器引入不必要抽象层 (design): 作者回复已改为字符串参数懒解析并加平台守卫；最终 head 中该装饰器已不存在，说明回归到直接函数调用。
- can_implement 重复调用 is_triton_gemm_w8a8_tuned (question): 作者解释 can_implement() 是 classmethod，kernel 实例尚未创建，use_triton 不可用，因此必须在类方法中独立校验；讨论接受，未再修改。
- GFX120x 上的 AITER 测试覆盖 (testing): 作者在 conversation 中列出 attention benchmark 与 Qwen3-30B-A3B-FP8 e2e serving 结果；dllehr-amd 最终 APPROVED。
- fused_batched_moe FP8 支持需上游维护者批准 (design): 作者回应与 TritonExperts 的既有做法一致，最终合并者 dllehr-amd approve。

风险与影响

- 风险：
  1. 非 ROCm 平台启动回归：envs.py 中 VLLM_ROCM_USE_AITER_RMSNORM 的默认值计算曾无条件导入 vllm.platforms.rocm，会在 NVIDIA/CPU 上崩溃；评审中已修复，但最终代码形态未在文档中展示，仍需确认 helper 函数已正确落盘。
  2. gfx950 调优 shape 回归：is_triton_gemm_w8a8_tuned() 从单个列表改为 gfx950_tuned | rdna4_tuned 结构，若构造时误改 gfx950_tuned 集合，会让 MI355 等 CDNA4 卡丢失已调优 shape 而回退到慢路径；评审中 tjtanaa 已要求拆分，最终 head 中两者分开构造。
  3. gfx12 上 untuned shape 静默回退：can_implement() 对不在 tuned 列表的 (N, K) 直接拒绝，调度器会回退到通用 Triton/ 原生内核。该列表是硬编码，新模型权重 shape 不在其中时性能会无提示地骤降。
  4. AITER RMSNorm 在 gfx12 上损坏：默认在 rdna4 上关闭，但用户仍可通过 VLLM_ROCM_USE_AITER_RMSNORM=1 强行开启，可能产生错误结果甚至崩溃，需在文档或启动日志中提示。
  5. 自动化测试缺失：本 PR 没有配套单元测试或 CI 测试，arch 判定与 tuned 集合的回归只能靠人工在具体硬件上验证，后续改动容易在无 gfx12 硬件的 CI 中漏检。 - 影响：直接影响 RDNA4 消费级 GPU（R9700、RX 9070 XT）用户：attention prefill/extend 延迟下降 46%-72%，decode 场景也有 5%-15% 提升，同时获得 FP8 MoE 与 FP8 linear 的 Triton 快速路径。MI3xx（gfx942/gfx950/gfx90a）行为保持不变。间接影响包括：编译融合 pass 的默认匹配方式随架构分流、MoE 后端选择逻辑在 rdna4 上跳过 AITER、以及 supports_fp8() 语义从 CDNA 扩展到含 gfx120x——这会影响所有调用该方法的 ROCm 内核选择路径。团队层面，该 PR 为后续 gfx12 原生内核（非回退）支持铺平了架构判定基础。 - 风险标记：非 ROCm 平台启动回归，gfx950 tuned shape 拆分风险，tuned shape 硬编码，RMSNorm 默认关闭，缺少自动化测试，MoE 后端回退路径依赖

关联脉络

- PR #50582 [ROCm][Kimi-K3] aiter moe environment variable cleanup: 同样深度改动 vllm/_aiter_ops.py、vllm/envs.py 与 AITER MoE 相关路径，与本 PR 的 AITER gating 与 env 默认值逻辑高度耦合。
- PR #50728 [ROCm][Test] Fix AITER MXFP4 oracle contract: 同为 ROCm AITER 生态的测试 / 契约修复，涉及 MoE oracle 内核选择，本 PR 改动的 fp8.py 与之同属一组后端选择逻辑。