# PR #40426 完整报告

vllm-project/vllm

[ROCM] [FEAT] Integrate Aiter hipBLASLt GEMM online tuning

合并时间：2026-06-05 14:45

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/40426

## 执行摘要

- 一句话：集成 Aiter hipBLASLt GEMM 在线调优与测试
- 推荐动作：推荐 ROCm 相关开发者精读。该 PR 展示了如何在 vLLM 内核选择系统中集成第三方库 kernel 的模式：通过 is_supported/can_implement 门控、process_weights_after_loading 预处理权重、apply_scaled_mm 执行计算。环境变量组合条件检查的设计值得借鉴（平台检查 + 多个 flag 组合）。fake impl 的维度正确性对 torch.compile 至关重要。

## 功能与动机

ROCm 平台上的 FP8 GEMM 性能高度依赖 hipBLASLt 的调优结果。离线调优流程繁琐且不易移植，Aiter 的 hipb_mm 内核支持运行时调优（online tuning），可以避免离线步骤，提升部署灵活性。此 PR 旨在将 Aiter hipBLASLt GEMM 集成到 vLLM 的内核选择框架中，并通过单一环境变量统一控制在线调优和内核启用。

## 实现拆解

1. 新增环境变量 VLLM_ROCM_USE_AITER_LINEAR_HIPBMM：在 vllm/envs.py 中定义，默认关闭。在 vllm/platforms/rocm.py 模块导入时，若该变量与 VLLM_ROCM_USE_AITER、VLLM_ROCM_USE_AITER_LINEAR 同时启用且为 MI3xx 平台，则自动设置 os.environ['HIP_ONLINE_TUNING'] = '1'，使得 hipBLASLt 在首次 GEMM 时执行在线调优。
2. 注册自定义 FP8 hipb_mm 操作：在 vllm/_aiter_ops.py 中添加 _ensure_hipb_mm_extension_initialized() 确保每个设备仅初始化一次扩展。添加 _rocm_aiter_hipb_mm_fp8_impl（真实执行）和 _rocm_aiter_hipb_mm_fp8_fake（用于元信息推断），并通过 direct_register_custom_op 注册为 hipb_mm_fp8 操作。rocm_aiter_ops 类新增 is_linear_hipbmm_enabled 方法，整合三个环境变量和 MI3xx 检查。
3. 实现 AiterHipbMM 线性核：在 vllm/model_executor/kernels/linear/scaled_mm/aiter.py 中新增 AiterHipbMMPerTokenFp8ScaledMMLinearKernel，继承自 FP8ScaledMMLinearKernel。其 is_supported 检查 ROCm、aiter 库和门控标志；can_implement 验证 per-token/per-channel 量化、bf16 输出、N>=16 且 N/K 可被 16 整除；process_weights_after_loading 预转置权重（保持 bpreshuffle 布局的非连续视图）和权重缩放（若维度 >1），避免每次前向重复转置；apply_scaled_mm 调用

rocm_aiter_ops.hipb_mm_fp8 并重塑输出形状。

4. 注册内核到平台选择链：在 vllm/model_executor/kernels/linear/__init__.py 中导入新内核类，并插入到 ROCm 平台内核列表首位，使其成为 FP8 线性层的优先候选。
5. 测试覆盖：新增 tests/rocm/aiter/test_aiter_hipb_mm_linear_kernel.py，共 406 行。包含门控测试（确认缺少任一环境变量时 is_supported 返回 False）、online tuning 环境变量转发测试、can_implement 成功与拒绝路径测试、bpreshuffle 运行时支持探测、CSV 缓存行查询、以及将 Aiter hipb_mm FP8 输出与 dequantized fp32 参考比较的数值精度测试。

关键文件：

- vllm/model_executor/kernels/linear/scaled_mm/aiter.py（模块 线性核；类别 source；类型 core-logic；符号 AiterHipbMMPerTokenFp8ScaledMMLinearKernel, is_supported, can_implement, process_weights_after_loading）：核心变更文件，新增 AiterHipbMMPerTokenFp8ScaledMMLinearKernel 类，实现内核的门控、量化条件检查、权重预转置和 forward 方法。
- vllm/_aiter_ops.py（模块 Aiter 操作层；类别 source；类型 core-logic；符号 _ensure_hipb_mm_extension_initialized, _rocm_aiter_hipb_mm_fp8_impl, _rocm_aiter_hipb_mm_fp8_fake, is_linear_hipbmm_enabled）：实现 hipb_mm FP8 自定义 op 的注册、真实和 fake 实现，以及 is_linear_hipbmm_enabled 门控方法。
- tests/rocm/aiter/test_aiter_hipb_mm_linear_kernel.py（模块 测试集；类别 test；类型 test-coverage；符号 enable_hipb_mm_kernel, _make_config, _find_csv_row, _skip_if_no_hipb_mm_solution）：新增 406 行测试文件，覆盖门控、环境变量转发、can_implement 路径、bpreshuffle 运行时和数值精度。
- vllm/platforms/rocm.py（模块 平台初始化；类别 source；类型 core-logic）：在模块导入时根据环境变量组合自动设置 HIP_ONLINE_TUNING=1，确保调优在 hipBLASLt 初始化前生效。
- vllm/envs.py（模块 环境配置；类别 source；类型 configuration）：新增环境变量 VLLM_ROCM_USE_AITER_LINEAR_HIPBMM 的定义和解析。
- vllm/model_executor/kernels/linear/__init__.py（模块 内核注册；类别 source；类型 data-contract）：导入并注册 AiterHipbMMPerTokenFp8ScaledMMLinearKernel 到 ROCm 平台内核列表首位。

关键符号：AiterHipbMMPerTokenFp8ScaledMMLinearKernel.is_supported, AiterHipbMMPerTokenFp8ScaledMMLinearKernel.can_implement, AiterHipbMMPerTokenFp8ScaledMMLinearKernel.process_weights_after_loading, AiterHipbMMPerTokenFp8ScaledMMLinearKernel.apply_scaled_mm, _ensure_hipb_mm_extension_initialized, _rocm_aiter_hipb_mm_fp8_impl, is_linear_hipbmm_enabled, hipb_mm_fp8

## 关键源码片段

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

核心变更文件，新增AiterHipbMMPerTokenFp8ScaledMMLinearKernel类，实现内核的门控、量化条件检查、权重预转置和 forward 方法。

```
class AiterHipbMMPerTokenFp8ScaledMMLinearKernel(FP8ScaledMMLinearKernel):
    '基于 Aiter hipBLASLt bpreshuffle 的 FP8 线性核，支持在线调优。'
    '权重在 process_weights_after_loading 中预转置为 [K, N] 并 shuffle，'
    '前向时直接调用 hipb_mm，避免运行时开销。'

    @classmethod
    def is_supported(cls, compute_capability=None):
        if not current_platform.is_rocm():
            return False, 'requires ROCm.'
        if not rocm_aiter_ops.is_linear_hipbmm_enabled():
            # 必须同时设置三个环境变量才启用
            return False, (
                'requires setting `VLLM_ROCM_USE_AITER=1`, '
                '`VLLM_ROCM_USE_AITER_LINEAR=1`, '
                'and `VLLM_ROCM_USE_AITER_LINEAR_HIPBMM=1`.'
            )
        try:
            import aiter
        except Exception:
            return False, 'requires aiter library to be installed.'
        if not hasattr(aiter, 'hipb_mm'):
            return False, 'requires aiter hipb_mm support.'
        return True, None

    @classmethod
    def can_implement(cls, c: FP8ScaledMMLinearLayerConfig):
        # 检查 per-token activation / per-channel weight 缩放
        is_ptpc = (
            c.activation_quant_key.scale.group_shape.is_per_token()
            and c.weight_quant_key.scale.group_shape.is_per_channel()
        )
        if c.weight_shape is None:
            return False, 'weight_shape is required for Aiter kernels'
        N, K = c.weight_shape
        # 输出必须为 bfloat16
        if c.out_dtype is not torch.bfloat16:
            return False, 'requires bfloat16 output dtype.'
        if not is_ptpc:
            return False, 'requires per token activation scales and per channel weight scales.'
        # hipb_mm 要求 N >= 16 且 N, K 均可被 16 整除
        if not (N >= 16 and N % 16 == 0 and K % 16 == 0):
            return False, (
                'requires N >= 16 and both N and K divisible by 16, '
                f'received N={N} and K={K}.'
            )
        return True, None
```

```
    def process_weights_after_loading(self, layer):
        # 获取权重和权重缩放参数
        w_name, w_s_name, *_ = self.layer_param_names
        w, w_s, *_ = self._get_layer_params(layer)
        # 先转置为 [K, N]，再做 shuffle，保持非连续视图以维持 bpreshuffle 布局
        shuffled_w = rocm_aiter_ops.shuffle_weight(w.t().contiguous())
        replace_parameter(layer, w_name, torch.nn.Parameter(shuffled_w.t(), requires_grad=
        False))
        # 若权重缩放维度 >1，转置为连续布局
        if w_s.ndim > 1:
            replace_parameter(layer, w_s_name, torch.nn.Parameter(w_s.t().contiguous(), requires_
            grad=False))

    def apply_scaled_mm(self, *, A, B, out_dtype, As, Bs, bias, output_shape):
        output_shape[-1] = B.shape[1]
        return rocm_aiter_ops.hipb_mm_fp8(A, B, As, Bs, bias, out_dtype).view(*output_shape)
```

## vllm/_aiter_ops.py

实现 hipb_mm FP8 自定义 op 的注册、真实和 fake 实现，以及 is_linear_hipbmm_enabled 门控方法。

```
# 全局集合，记录已初始化扩展的设备
_HIPB_MM_INITIALIZED_DEVICES: set[int] = set()

def _ensure_hipb_mm_extension_initialized() -> None:
    '每个设备在使用 hipb_mm 之前创建一次 aiter extension，避免重复初始化。'
    import aiter
    device = torch.accelerator.current_device_index()
    if device not in _HIPB_MM_INITIALIZED_DEVICES:
        aiter.hipb_create_extension()
        _HIPB_MM_INITIALIZED_DEVICES.add(device)

def _rocm_aiter_hipb_mm_fp8_impl(
    A, B, As, Bs, bias=None, output_dtype=torch.bfloat16
) -> torch.Tensor:
    from aiter import hipb_mm
    _ensure_hipb_mm_extension_initialized()
    # solution_index=-1 表示自动选择，bpreshuffle=True 启用预设 shuffle 布局
    return hipb_mm(
        A, B, solution_index=-1, bias=bias, out_dtype=output_dtype,
        scaleA=As, scaleB=Bs, scaleOut=None, bpreshuffle=True,
    )

# 在 rocm_aiter_ops 类内部的方法：
@classmethod
@if_aiter_supported
def is_linear_hipbmm_enabled(cls) -> bool:
    from vllm.platforms.rocm import on_mi3xx
    # 需要同时启用 AITER 和 LINEAR 及 HIPBMM 标志，且仅限 MI3xx 架构
```

return cls.is_linear_enabled() and on_mi3xx() and cls._LINEAR_HIPBMM_ENABLED

## 评论区精华

Review 中主要讨论点包括：

- 环境变量设计：最初使用两个标志，后合并为单一 VLLM_ROCM_USE_AITER_LINEAR_HIPBMM，简化用户配置。
- 权重布局错误：gemini-code-assist 指出 _rocm_aiter_hipb_mm_fp8_impl 中对权重 B 和缩放 Bs 做了运行时转置，但 B 已在 process_weights_after_loading 中预处理，导致二次转置产生错误。修复方案是将转置逻辑移到预处理阶段，在加载时完成。
- 输出维度 Bug：fake 实现中 n = B.shape[0] 应为 B.shape[1]，导致 shape 推断异常。
- 错误消息优化：is_supported 的拒绝消息被重写为牛津逗号分隔的三标志列表，提高可调试性。
- 精度测试争议：AndreasKaratzas 要求添加内核精度测试，tjtanaa 认为第三方库精度应由库自身保证，但最终作者添加了数值比较测试，以 torch.allclose 验证 FP8 输出与 fp32 参考的一致性。
- 环境变量合并 (design): 采用单标志方案，添加注释。
- 权重布局与转置错误 (correctness): 将权重预转置逻辑移到 process_weights_after_loading，保持 B 为 [K,N] shuffle 后视图，Bs 为连续转置。
- fake impl 输出维度错误 (correctness): 修复为 n = B.shape[1]。
- 是否添加精度测试 (testing): 添加了 dequantized fp32 参考的 allclose 测试。
- is_supported 错误消息改进 (documentation): 按照建议修改错误消息。

## 风险与影响

- 风险：
  1. 回归风险：新内核默认关闭（False），不影响现有路径。但若用户显式启用，需确保 aiter 库已安装且版本支持 hipb_mm，is_supported 进行了前置检查。
  2. 性能风险：HIP_ONLINE_TUNING=1 会在首次触及 hipBLASLt 时触发调优，延迟约 1-2 秒，调优结果以 CSV 缓存，后续复用。
  3. 兼容性风险：内核仅在 MI3xx 平台验证，其他 AMD GPU（如 gfx90a、gfx942）未测试；can_implement 要求 N>=16 且对齐 16，若模型权重形状不满足将 fallback 到其他内核。
  4. 安全风险：无。 - 影响：用户影响：ROCm 用户可通过设置 VLLM_ROCM_USE_AITER=1 VLLM_ROCM_USE_AITER_LINEAR=1 VLLM_ROCM_USE_AITER_LINEAR_HIPBMM=1 启用 hipBLASLt 在线调优的 FP8 GEMM，预期 MI3xx 上 Qwen3-32B 等模型获得性能提升（见 PR 描述中的 lm-eval 对比表）。系统影响：新增环境变量、自定义 PyTorch op 和内核注册，但不改变默认行为。团队影响：ROCm 团队需维护新增内核与测试，未来可考虑在环境变量稳定后移除，自动启用。

- 风险标记：新环境变量，依赖外部库（aiter），仅 MI3xx 验证，默认关闭，核心路径变更（内核选择器）

## 关联脉络

- PR #41002 [ROCm][perf] Use workspace manager for sparse indexer allocations: 同为 ROCm 性能优化，共享 aiter 运行时基础设施。
- PR #44436 [ROCm][CI] Add test for Aiter unified attn kernel: 同为 ROCm 平台 aiter 相关测试增强。