PR #51457 完整报告

vllm-project/vllm

[Test] Add ROCm AITER FP8 MLA prefill accuracy test

合并时间：2026-08-09 08:42

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/51457

执行摘要

- 一句话：为 gfx950 新增 AITER FP8 MLA prefill 精度测试
- 推荐动作：值得精读。该 PR 展示了一种轻量级 kernel 精度测试模式：用 object.__new__ 绕过 __init__、以 SimpleNamespace 代替 dataclass，在无需初始化完整模型的前提下驱动真实的 metadata builder 与 impl 入口，避免测试与实现脱节。建议跟进 reviewer 的两个 follow-up：确认 device 字符串语义、对容差做数值验证；同时可参考 tests/v1/attention/test_sparse_mla_backends.py 等近期 MLA 测试的演进，评估是否可以把类似模式推广到更多 ROCm 内核路径。

功能与动机

PR body 明确指出：FP8 MLA prefill 路径（AiterMLAImpl._mla_fp8_prefill_attn -> mla_prefill_ps_asm_fwd + mla_reduce_v1）在 gfx950（MI355）上随 AITER 内核自动启用，但此前没有任何测试覆盖。新增测试是为了验证真实持久调度元数据契约（get_ps_metadata_v1）以及两个内核的输出正确性，让这条硬件专属路径获得回归保护。

实现拆解

1. 平台门控与测试环境：_fp8_prefill_available() 先判断 current_platform.is_rocm() 且 torch.cuda.is_available()，再调用 vllm.v1.attention.backends.mla.rocm_aiter_mla._fp8_mla_prefill_supported() 确认 AITER 已导出所需内核；不满足时整个文件被 pytestmark 跳过。autouse fixture _workspace_manager 负责初始化与重置全局 workspace manager，避免状态泄漏到其他测试。
2. 最小 impl 构造：_make_impl() 使用 object.__new__(AiterMLAImpl) 绕过 __init__，只设置 num_heads、v_head_dim、scale 以及两个内核函数引用，保证被调用的是真实 _mla_fp8_prefill_attn 入口而非测试自造逻辑。
3. 真实元数据构建：_build_prefill_metadata() 同样用 object.__new__(AiterMLAMetadataBuilder) 构造 builder，调用 _init_fp8_prefill_ps_buffers 和 _build_fp8_prefill_ps_metadata 生成 fp8_prefill_* 元数据，用 SimpleNamespace 代替 dataclass 最小化替身，从而覆盖真实元数据契约。
4. 数值验证：test_fp8_prefill_matches_reference 参数化 seq_len 为 128/512，固定随机种子生成 bf16 输入，调用 impl._mla_fp8_prefill_attn；参考实现 _reference 在 fp8 量化输入上做因果 SDPA；断言输出 isfinite 且以 atol=1e-1、rtol=5e-2 接近。容差来自 MI355 实测 e4m3 舍入噪声（最坏元素绝对误差约 0.054）。

关键文件：

- tests/kernels/attention/test_rocm_aiter_mla_fp8_prefill.py（模块 内核测试；类别 test；类型 test-coverage；符号 _fp8_prefill_available, _workspace_manager, _make_impl, _build_prefill_metadata）：唯一变更文件，新增 gfx950 专属的 AITER FP8 MLA prefill 精度测试，覆盖真实 metadata builder、impl 与两个 AITER 内核

关键符号：test_fp8_prefill_matches_reference, _build_prefill_metadata, _make_impl, _fp8_prefill_available, _reference, _workspace_manager, AiterMLAImpl._mla_fp8_prefill_attn, AiterMLAMetadataBuilder._build_fp8_prefill_ps_metadata

关键源码片段

tests/kernels/attention/test_rocm_aiter_mla_fp8_prefill.py

唯一变更文件，新增 gfx950 专属的 AITER FP8 MLA prefill 精度测试，覆盖真实 metadata builder、impl 与两个 AITER 内核

```
# tests/kernels/attention/test_rocm_aiter_mla_fp8_prefill.py —— gfx950 专属
# AITER FP8 MLA prefill 精度测试（节选：impl 构造、元数据构建、主测试）

def _make_impl():
    """用 object.__new__ 绕过 __init__，只补 AiterMLAImpl 中
    _mla_fp8_prefill_attn 实际读取的字段，保证调用的是真实入口。"""
    from aiter import mla_prefill_ps_asm_fwd, mla_reduce_v1
    from vllm.v1.attention.backends.mla.rocm_aiter_mla import AiterMLAImpl

    impl = object.__new__(AiterMLAImpl)
    impl.num_heads = NUM_HEADS # 16，FP8 prefill 要求头数 16 对齐
    impl.v_head_dim = V_HEAD_DIM # 128
    impl.scale = SCALE # 1/sqrt(192)
    impl._mla_prefill_ps_asm_fwd = mla_prefill_ps_asm_fwd
    impl._mla_reduce_v1 = mla_reduce_v1
    return impl


def _build_prefill_metadata(seq_lens: list[int], device: torch.device):
    """用真实 builder 生成 fp8_prefill_* 元数据，不手写契约逻辑。"""
    from vllm.v1.attention.backends.mla.rocm_aiter_mla import (
        AiterMLAMetadataBuilder,
    )

    # 先算 query 起始位置，模拟持久调度下的长度布局
    qo_indptr_cpu = torch.zeros(len(seq_lens) + 1, dtype=torch.int32)
    qo_indptr_cpu[1:] = torch.tensor(seq_lens, dtype=torch.int32).cumsum(0)
    qo_indptr = qo_indptr_cpu.to(device)
    total_q = int(qo_indptr_cpu[-1].item())
    max_q = max(seq_lens)
```

```
    # object.__new__ 构造 builder，补充 _init_fp8_prefill_ps_buffers 所需字段
    builder = object.__new__(AiterMLAMetadataBuilder)
    builder.num_heads = NUM_HEADS
    builder.mla_dims = SimpleNamespace(v_head_dim=V_HEAD_DIM)
    builder._init_fp8_prefill_ps_buffers(
        max_num_reqs=len(seq_lens),
        max_prefill_qlen=max_q,
        max_num_batched_tokens=total_q,
        device=device,
    )

    # SimpleNamespace 代替 metadata dataclass：builder 只读取这几个字段
    prefill = SimpleNamespace(query_start_loc=qo_indptr, max_query_len=max_q)
    metadata = SimpleNamespace(prefill=prefill, num_decodes=0)
    common = SimpleNamespace(query_start_loc_cpu=qo_indptr_cpu)
    builder._build_fp8_prefill_ps_metadata(metadata, common)
    return metadata, total_q


@pytest.mark.parametrize("seq_len", [128, 512])
@torch.inference_mode()
def test_fp8_prefill_matches_reference(seq_len: int) -> None:
    torch.manual_seed(0) # 固定随机种子，保证数值可复现
    metadata, total_q = _build_prefill_metadata([seq_len], torch.device("cuda"))

    # q/k 携带 nope+rope 共 192 维，v 只携带 v_head_dim 128 维，
    # 与 MLA kv_b_proj 解压后的布局保持一致
    q = torch.randn(total_q, NUM_HEADS, QK_HEAD_DIM,
                    dtype=torch.bfloat16, device="cuda")
    k = torch.randn(total_q, NUM_HEADS, QK_HEAD_DIM,
                    dtype=torch.bfloat16, device="cuda")
    v = torch.randn(total_q, NUM_HEADS, V_HEAD_DIM,
                    dtype=torch.bfloat16, device="cuda")
    out = torch.zeros(total_q, NUM_HEADS * V_HEAD_DIM,
                      dtype=torch.bfloat16, device="cuda")

    # 执行真实入口：内部依次调用 mla_prefill_ps_asm_fwd 与 mla_reduce_v1
    impl = _make_impl()
    impl._mla_fp8_prefill_attn(q, k, v, metadata, out)

    # 参考实现与 kernel 输入保持同一 fp8 量化级别
    out_ref = _reference(q, k, v)

    assert torch.isfinite(out).all()
    # 容差依据 MI355 实测 e4m3 舍入噪声设定：最坏元素绝对误差约 0.054
    torch.testing.assert_close(
        out.view(total_q, NUM_HEADS, V_HEAD_DIM).float(),
        out_ref.float(),
        atol=ATOL,
```

```
        rtol=RTOL,
    )
```

评论区精华

reviewer AndreasKaratzas 提出两个 minor 跟进项：一是在 _workspace_manager 中 torch.device("cuda") 的语义疑问，ROCm 环境下是否应为 hip/rocm；二是容差 ATOL/RTOL 需要做数值验证，期望测试不仅能通过，还能在真实回归出现时失败。这两点均未阻塞合入，maintainer 最终 APPROVED 并触发 /ci run，Buildkite CI #82949 通过。

- workspace manager 中 device 字符串的语义疑问 (question): 标记为 minor follow-up NIT，未阻塞合入，PR 最终 APPROVED。
- 容差需要数值验证以捕获回归 (testing): 已 APPROVED；数值验证与容差收紧留作 follow-up 处理。

风险与影响

- 风险： 1) 覆盖范围受限：测试 gfx950-only，非 gfx950 平台全部跳过，AITER 内核若在未来扩展到其他架构会立即失去保护。 2) 外部依赖脆弱：测试直接 import aiter.mla_prefill_ps_asm_fwd 与 mla_reduce_v1，AITER 版本升级导致符号变动或未安装时会 import 失败（不过跳过条件已通过 _fp8_mla_prefill_supported 兜底）。 3) 构造方式脆弱：object.__new__ 绕过 __init__，若 AiterMLAImpl 或 AiterMLAMetadataBuilder 后续新增依赖字段，测试可能失真或漏检。 4) 容差较宽松：atol=1e-1 约为实测噪声 2 倍，可能掩盖中等程度的精度回归，reviewer 也提出了同样的担忧。 5) 全局状态：_workspace_manager 初始化全局工作区，若测试中途异常退出，reset_workspace_manager 可能不执行，存在泄漏到同进程其他测试的风险。
  - 影响：影响面集中在 ROCm/gfx950 的 AITER FP8 MLA prefill 路径：为该路径提供首个精度回归基线，未来改动 rocm_aiter_mla.py 相关逻辑时能快速发现偏差。对非 gfx950 平台零运行时影响（文件被跳过）；对 CI 影响有限，仅在具备 gfx950 + AITER 的机器上新增 2 个参数化用例。对团队而言，它补充了 decode（test_rocm_aiter_mla_head_padding）之外的 prefill 覆盖，完善了 MLA 注意力后端的测试矩阵。
  - 风险标记：硬件专属测试覆盖有限，依赖 AITER 外部符号，绕过 __init__构造对象，容差依赖实测噪声，全局状态 fixture

关联脉络

- PR #50365 [Perf][Sparse MLA] Drop the atomic contention in the index remap: 同属 v1 MLA 注意力后端（vllm/v1/attention/backends/mla）的测试与优化线，分别覆盖 sparse MLA 与 AITER MLA 两个子路径，可对照阅读测试组织方式。
- PR #51298 [DSv32/GLM Perf] Skip short prefill topk for dense mha layer, 97.9% kernel level latency reduction: 同为 prefill 阶段注意力路径的性能与正确性演进，且涉及 MLA 相关模型（DeepSeek V3.2），与本测试关注的 prefill 阶段形成关联脉络。