PR #30355 完整报告

sgl-project/sglang

[AMD] [Fix] Fix --attention-backend triton work for DeepSeek MLA on MI355 (null-K + decode dispatch + RoPE)

合并时间：2026-07-16 05:19

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30355

执行摘要

- 一句话：修复 DeepSeek MLA 在 MI355 上 triton 后端的 null-K 故障与精度错误
- 推荐动作：建议仔细阅读 _dispatch_mla_subtype 和 _skip_rope_for_aiter_fused_mla 的改动，理解如何通过 backend 标识隔离专有路径。Hermetic 测试的设计（无 GPU 依赖、Mock 关键属性）值得在其他类似场景推广。该 PR 也暴露了项目中 env var 和 backend 选择不一致的架构问题，值得团队后续系统性解决。

功能与动机

DeepSeek MLA models could not run with --attention-backend triton on gfx95 (MI300/MI355) when SGLANG_USE_AITER=1. Root cause is a recurring anti-pattern: several gfx95/aiter fused MLA paths are gated on env vars not on the actually selected attention backend. As a result the triton backend inherits aiter-only fused paths that it cannot consume, causing null-K GPU fault in prefill, decode dispatch fault, and accuracy degradation.

实现拆解

1. 在 forward_mla.py 的 forward_absorb_core 中，将 aiter-fused fused_qk_rope_cat_and_cache_mla 路径的条件从仅检测 _use_aiter_gfx95 扩展为同时检查 self.current_attention_backend == "aiter"，防止 triton 后端误入该路径。
2. 同样在 forward_mla.py 的 forward_absorb_prepare 的 RoPE 条件中增加 self.current_attention_backend == "triton" 放行，让 triton 正常应用 RoPE。
3. 修改 _skip_rope_for_aiter_fused_mla 方法，将其返回条件从 current_attention_backend not in FORWARD_ABSORB_CORE_ATTENTION_BACKENDS 收紧为 current_attention_backend == "aiter"，确保 triton 不会跳过 RoPE。
4. 在 attention_backend_handler.py 的 _dispatch_mla_subtype 中，为 MLA_FUSED_ROPE_ROCM 路径添加 attn.current_attention_backend == "aiter" 限制，使 triton decode 始终返回标准 MLA 方法。
5. 新增 Hermetic 测试文件 test_deepseek_mla_dispatch.py，通过 patching _is_hip 和伪造 attn、forward_batch 对象，验证不同 backend 和 decode/extend 条件下的 dispatch 结果，确保 triton decode 返回 MLA 而 aiter decode 返回 MLA_FUSED_ROPE_ROCM。该测试注册在 CUDA base-b 和 AMD stage-b 流水线上。

关键文件：

- test/registered/unit/models/test_deepseek_mla_dispatch.py（模块 测试；类别 test；类型 test-coverage；符号 _fake_forward_batch, _fake_attn, TestDispatchMLASubtype, test_hip_aiter_decode_takes_fused_rope）：新增 Hermetic 单元测试，固化 dispatch 行为，防止 triton 误入 aiter-only 路径，是验证修复的关键保障。
- python/sglang/srt/models/deepseek_common/attention_forward_methods/forward_mla.py（模块 MLA 前向；类别 source；类型 data-contract；符号 forward_absorb_core, forward_absorb_prepare, _skip_rope_for_aiter_fused_mla）：核心修复文件，修改了三个关键方法：forward_absorb_core、forward_absorb_prepare 和 _skip_rope_for_aiter_fused_mla，分别修复 null-K GPU 故障、RoPE 精度问题和 keep backend 隔离。
- python/sglang/srt/models/deepseek_common/attention_backend_handler.py（模块 后端分发；类别 source；类型 data-contract；符号 _dispatch_mla_subtype）：修复 decode dispatch 的关键文件，修改 _dispatch_mla_subtype 函数，防止 triton decode 路由到 aiter-only 的 MLA_FUSED_ROPE_ROCM 路径。

关键符号：_dispatch_mla_subtype, forward_absorb_core, forward_absorb_prepare, _skip_rope_for_aiter_fused_mla

关键源码片段

test/registered/unit/models/test_deepseek_mla_dispatch.py

新增 Hermetic 单元测试，固化 dispatch 行为，防止 triton 误入 aiter-only 路径，是验证修复的关键保障。

```
import unittest
from types import SimpleNamespace
from unittest import mock

from sglang.srt.models.deepseek_common import attention_backend_handler as abh
from sglang.srt.models.deepseek_common.attention_forward_methods.forward_methods import
(
    AttnForwardMethod,
)
from sglang.test.ci.ci_register import register_amd_ci, register_cuda_ci
from sglang.test.test_utils import CustomTestCase

register_cuda_ci(est_time=10, stage="base-b", runner_config="1-gpu-small")
register_amd_ci(est_time=10, suite="stage-b-test-1-gpu-small-amd-mi35x")

def _fake_forward_batch(is_decode: bool):
    return SimpleNamespace(forward_mode=SimpleNamespace(is_decode=lambda: is_decode))

def _fake_attn(backend: str, rocm_fused_decode_mla: bool = True):
    return SimpleNamespace(
        current_attention_backend=backend,
```

```
        rocm_fused_decode_mla=rocm_fused_decode_mla,
    )

class TestDispatchMLASubtype(CustomTestCase):
    def test_hip_aiter_decode_takes_fused_rope(self):
        # aiter + decode 时，应选择 fused ROPE 快速路径（保证无回归）
        with mock.patch.object(abh, "_is_hip", True):
            method = abh._dispatch_mla_subtype(
                _fake_attn("aiter"), _fake_forward_batch(is_decode=True)
            )
        self.assertEqual(method, AttnForwardMethod.MLA_FUSED_ROPE_ROCM)

    def test_hip_triton_decode_stays_plain_mla(self):
        # 修复目标：triton 后端即使 rocm_fused_decode_mla 为 True，
        # 也必须返回标准 MLA，避免 GPU fault。
        with mock.patch.object(abh, "_is_hip", True):
            method = abh._dispatch_mla_subtype(
                _fake_attn("triton"), _fake_forward_batch(is_decode=True)
            )
        self.assertEqual(method, AttnForwardMethod.MLA)

    def test_hip_aiter_extend_stays_plain_mla(self):
        # fused 路径仅用于 decode，extend/prefill 使用常规 MLA。
        with mock.patch.object(abh, "_is_hip", True):
            method = abh._dispatch_mla_subtype(
                _fake_attn("aiter"), _fake_forward_batch(is_decode=False)
            )
        self.assertEqual(method, AttnForwardMethod.MLA)

if __name__ == "__main__":
    unittest.main()
```

python/sglang/srt/models/deepseek_common/attention_forward_methods/forward_mla.py

核心修复文件，修改了三个关键方法：forward_absorb_core、forward_absorb_prepare 和 _skip_rope_for_aiter_fused_mla，分别修复 null-K GPU 故障、RoPE 精度问题和 keep backend 隔离。

```
# file: python/sglang/srt/models/deepseek_common/attention_forward_methods/forward_mla.py

def _skip_rope_for_aiter_fused_mla(self) -> bool:
    """仅当使用 aiter 后端且处于 gfx95 平台时跳过 RoPE，
    因为 aiter 的 fused kernel 内部处理 RoPE。
    之前对于任何不在 `FORWARD_ABSORB_CORE_ATTENTION_BACKENDS` 中的后端都返回
    True，
    导致 triton 错误跳过 RoPE 产生 0.03 准确率。
    """
    return _use_aiter_gfx95 and self.current_attention_backend == "aiter"
```

```
# forward_absorb_prepare 中 RoPE 条件增加 triton 放行
if (
    self.rotary_emb is not None
    and (not fuse_rope_for_trtllm_mla)
    and (not skip_rope_for_dsa_tilelang_fused)
    and (not skip_rope_for_aiter_fused_mla)
    and (
        not _use_aiter
        or not _is_gfx95_supported
        or self.use_dsa
        or self.current_attention_backend == "triton" # triton 需要正常应用 RoPE
    )
):
    q_pe, k_pe = self.rotary_emb(positions, q_pe, k_pe)

# forward_absorb_core 中限制 aiter-fused 路径
if _use_aiter_gfx95 and self.current_attention_backend == "aiter":
    # aiter 专用 fused 路径，内部处理 RoPE 并返回空 tensor 作为 k
    q, _, _, k = fused_qk_rope_cat_and_cache_mla(
        q_nope_out, q_pe, k_nope, k_pe,
        get_token_to_kv_pool().get_key_buffer(self.attn_mqa.layer_id),
        forward_batch.out_cache_loc, positions, cos, sin, ...
    )
else:
    # 其他后端（triton、flashinfer 等）走标准路径：显式拼接 nope 和 pe，
    # 并将完整 k 保存到 KV cache，供后续 attention 读取。
    k = torch.cat([k_nope, k_pe], dim=-1)
    ...
```

python/sglang/srt/models/deepseek_common/attention_backend_handler.py

修复 decode dispatch 的关键文件，修改 _dispatch_mla_subtype 函数，防止 triton decode 路由到 aiter-only 的 MLA_FUSED_ROPE_ROCM 路径。

```
# file: python/sglang/srt/models/deepseek_common/attention_backend_handler.py

def _dispatch_mla_subtype(attn, forward_batch):
    """
    根据平台和 attention 后端选择 MLA 前向方法。
    在 HIP 平台上，fused decode 路径（MLA_FUSED_ROPE_ROCM）仅适用于 aiter 后端；
    triton 后端必须使用标准 MLA 方法以避免 GPU fault。
    """
    if _is_hip:
        # 新增 backend == "aiter" 条件
        if (
            attn.rocm_fused_decode_mla
            and forward_batch.forward_mode.is_decode()
            and attn.current_attention_backend == "aiter"
        ):
            return AttnForwardMethod.MLA_FUSED_ROPE_ROCM
```

```
        else:
            return AttnForwardMethod.MLA
    else:
        # 非 HIP 平台已有 Intel AMX 特殊路径
        if hasattr(attn, "fused_qkv_a_proj_with_mqa") and use_intel_amx_backend(attn):
            return AttnForwardMethod.MLA_FUSED_ROPE_CPU
        else:
            return AttnForwardMethod.MLA
```

评论区精华

Review 中 hnyls2002 提出测试文件不应放在 test/registered/unit/models/ 下（该目录可能废弃），建议移至 test/registered/unit/ 下。HaiShaw 赞同，作者 yichiche 随后将测试文件移动到 test/registered/unit/ 并重新触发 CI。无其他未解决讨论。

- 测试文件位置 (design): 文件移动到 test/registered/unit/ 下。

风险与影响

- 风险：
  1. 后端隔离风险：虽然改动声称不影响 aiter/CUDA 等后端，但 forward_mla.py 中 RoPE 条件的放宽 (or self.current_attention_backend == "triton") 可能会使其他非 aiter 后端（如 flashinfer）也跳过 aiter 特有限制，但该条件原本就是为 aiter 设计，增加 triton 放行后对其他后端无影响（因为它们不满足 _use_aiter_gfx95）。
  2. 准确率差异：triton 后端在 gsm8k 上准确率为 0.945，相比 aiter 的 0.955 略低但可接受，但生产环境需关注。
  3. 未修复 spec 解码：EAGLE speculative decoding 与 triton 后端同时使用仍会导致 GPU 故障，用户需明确知晓此限制。
  4. 测试覆盖有限：新增测试为 Hermetic 单元测试，没有 e2e 回归测试验证实际模型推理，可能遗漏 runtime 问题。 - 影响：对 AMD MI355 用户，triton 后端现在可用于 DeepSeek MLA 模型，但仅限于非 speculative 场景。对非 AMD 平台无行为变更。长期需维护 aiter 和 triton 两条 MLA 路径，增加维护成本。新增测试可在 CI 中快速检测 dispatch 回归。 - 风险标记：缺少 spec 解码覆盖，准确率略低于 aiter，测试未覆盖 e2e

关联脉络

- 暂无明显关联 PR