PR #20319 完整报告

sgl-project/sglang

[AMD] Support fp8 MLA for diffusion model

合并时间：2026-05-08 15:56

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/20319

执行摘要

- 一句话：AMD 扩散模型 FP8 MLA 注意力优化，加速 MI355X 推理
- 推荐动作：值得精读，尤其是_build_mla_prefill_metadata 和 _can_use_mla_prefill 的实现，它们展示了如何将一个为 DeepSeek 设计的 ASM 内核（MLA）适配到扩散模型的 DiT 注意力中。关注 @torch.compiler.disable 的后续拆分计划。该 PR 为 AMD 平台 DIffusion 模型树立了 FP8 注意力优化的范例。

功能与动机

PR 描述中明确提出："Replace the FP8 per-tensor flash attention path with the FP8 MLA prefill ASM kernel, which delivers significantly better performance on MI355X." 此外，依赖外部库 ROCm/aiter PR#2481 的升级。

实现拆解

实现分为以下几步：

1. 环境变量门控与 GPU 架构检查：新增 SGLANG_AITER_FP8_ATTN 环境变量控制 FP8 注意力开关，导入 _use_aiter_gfx95 函数检测是否为 MI350/MI355（gfx950）；定义模块级常量 _MLA_PREFILL_V_HEAD_DIM=128 和 _MLA_PREFILL_HEAD_TILE=8 记录内核约束。
2. 形状安全检查：实现 _can_use_mla_prefill(v_head_dim, num_heads) 函数，仅在 GPU 架构为 gfx950、V 头维度为 128 且头数能被 8 整除时返回 True；否则走回退路径。
3. 持久调度元数据构建：实现 _build_mla_prefill_metadata 函数，根据批次大小、序列长度、头数等参数构造 MLA 预填充内核所需的 indptr、indices 和 work/reduce 分区元数据；从 SGLang SRTAiter 后端模式改编而来。
4. MLA 内核调用与零填充：在 AITerImpl.forward 方法中新增分支：当启用 FP8 且形状允许时，先将 Q/K 从原生头维度（如 128）零填充到 192 维，调用 mla_prefill_ps_asm_fwd 和 mla_reduce_v1 内核，然后移除填充；否则使用原始的 aiter.flash_attn_func（BF16）。
5. 测试与配置：未包含单元测试；配置上通过环境变量 SGLANG_AITER_FP8_ATTN=1 启用，需结合 aiter 升级版本使用。

关键文件：

- python/sglang/multimodal_gen/runtime/layers/attention/backends/aiter.py（模块 扩散模型注意力；类别 source；类型 core-logic；符号 _can_use_mla_prefill, _build_mla_prefill_metadata, _mla_prefill_ps_attention）：核心变更文件，新增 FP8 MLA 预填充 ASM 内核支持，通过环境变量门控，包含形状检查、元数据构建和内核调用逻辑。

关键符号：_can_use_mla_prefill, _build_mla_prefill_metadata, _mla_prefill_ps_attention

关键源码片段

python/sglang/multimodal_gen/runtime/layers/attention/backends/aiter.py

核心变更文件，新增 FP8 MLA 预填充 ASM 内核支持，通过环境变量门控，包含形状检查、元数据构建和内核调用逻辑。

```
# SPDX-License-Identifier: Apache-2.0

import logging
import os
from typing import Optional

import aiter
import torch

from sglang.srt.models.deepseek_common.utils import _use_aiter_gfx95

logger = logging.getLogger(__name__)

_use_fp8_attn = os.environ.get("SGLANG_AITER_FP8_ATTN", "0") == "1"
_fp8_dtype = torch.float8_e4m3fn

# ── MLA prefill ASM kernel constraints ──────────────────────────────
# 硬编码的内核约束（仅 gfx950 可用）
_MLA_PREFILL_QK_HEAD_DIM = 192
_MLA_PREFILL_V_HEAD_DIM = 128
_MLA_PREFILL_HEAD_TILE = 8

if _use_fp8_attn:
    logger.info("DiT FP8 attention enabled via SGLANG_AITER_FP8_ATTN=1")

def _can_use_mla_prefill(v_head_dim: int, num_heads: int) -> bool:
    """Check if the MLA prefill ASM kernel supports the given shape and GPU."""
    return (
        _use_aiter_gfx95 # AMD gfx950 架构检测
        and v_head_dim == _MLA_PREFILL_V_HEAD_DIM # V 头维度必须为 128
        and num_heads % _MLA_PREFILL_HEAD_TILE == 0 # 头数必须能被 8 整除
    )

def _build_mla_prefill_metadata(
```

```
    batch_size: int,
    seq_lens: torch.Tensor,
    num_heads: int,
    num_kv_heads: int,
    is_causal: bool,
    block_size: int = 1,
    tile_q: int = 256,
    tile_kv: int = 128,
    kv_seq_lens: Optional[torch.Tensor] = None,
) -> dict:
    """Build persistent-scheduling metadata required by mla_prefill_ps_asm_fwd."""
    if kv_seq_lens is None:
        kv_seq_lens = seq_lens

    device = "cuda"
    gqa_ratio = num_heads // num_kv_heads

    qo_indptr = torch.zeros(batch_size + 1, dtype=torch.int32, device=device)
    kv_indptr = torch.zeros(batch_size + 1, dtype=torch.int32, device=device)

    qo_indptr[1 : batch_size + 1] = torch.cumsum(seq_lens, dim=0)
    actual_blocks = (kv_seq_lens + block_size - 1) // block_size
    kv_indptr[1 : batch_size + 1] = torch.cumsum(actual_blocks, dim=0)
    num_blocks = int(kv_indptr[-1])

    kv_indices = torch.arange(num_blocks, dtype=torch.int32, device=device)
    max_qlen = seq_lens.max()
    # ( 后续 work 分区与 reduce map 计算省略 )
    return {
        "qo_indptr": qo_indptr,
        "kv_indptr": kv_indptr,
        "kv_indices": kv_indices,
        # ... 其他元数据 ...
    }
```

评论区精华

review 中的核心讨论围绕 @torch.compiler.disable 装饰器的使用：

- avjves指出将 @torch.compiler.disable 施加在整个 forward 方法上，会禁用 BF16 路径的编译优化，建议只门控 MLA/FP8 分支。
- yichiche回应这是保守选择，因为该路径混合了自定义 aiter 扩展、运行时形状分支和量化逻辑，当前解决方案简单可靠；承诺作为 Future Work 改进。
- 最终决定保留全局 disable，以待后续重构。
- torch.compiler.disable 覆盖整个 forward 的合理性 (performance): 暂时保留全局 disable，作为 future work 后续拆分。

风险与影响

- 风险：具体风险：

1. 架构依赖：ASM 内核仅适用于 MI355X（gfx950），其他 AMD GPU（如 MI300X）即使设置了环境变量也会回退，但若 _use_aiter_gfx95 误检可能崩溃。
2. 形状约束：num_heads % 8 != 0 或 v_head_dim != 128 的组合可能导致内核 OOB 读取或错误结果；当前检查已覆盖，但缺少测试验证。
3. 外部依赖：依赖 ROCm/aiter PR#2481 的特定版本，若未升级则导入失败或行为不一致。
4. 编译影响：全局 @torch.compiler.disable 阻止了 BF16 路径的图优化，可能影响其他模型性能。 - 影响：影响范围：

- 用户：仅影响使用 AMD GPU（特别是 MI355X）运行扩散模型（如 Wan2.2）的用户。启用 FP8 后单卡 81 帧 720p 视频生成总时间减少约 19%。
- 系统：新代码仅在设置 SGLANG_AITER_FP8_ATTN=1 且 _use_aiter_gfx95 为 True 时激活，默认不开，对其他模型无影响。
- 团队：AMD 和扩散模型团队需要维护新增的 MLA 内核调用和元数据构建逻辑，以及形状约束表。
- 风险标记：依赖外部 aiter 版本，缺少单元测试，仅 MI355X 支持，全局 torch.compiler.disable

关联脉络

- PR #23955 [AMD] Add AMD FP8 MLA attention test for Wan2.2-T2V-A14B: 此后续 PR 添加了针对本 PR 核心功能（FP8 MLA 注意力）的单元测试，覆盖 Wan2.2 模型。