PR #30808 完整报告

sgl-project/sglang

[AMD] [GLM5] Enable dense-MHA short-context prefill fallback on gfx950

合并时间：2026-08-16 10:30

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30808

执行摘要

- 一句话：gfx950 启用密集 MHA 预填充回退，TTFT 降 22%-43%
- 推荐动作：值得精读，尤其是 FP8 KV 缓存布局差异（原始 vs 缩放布局）和 chunked-prefill 拆分时 KV refetch 导致 stale 数据的问题，这对 AMD 平台上的 MLA 注意力实现有借鉴意义。建议关注后续对 get_mla_kv_buffer 的改动，以及是否有补充 gfx950 路径测试的计划。

功能与动机

PR body 指出：在 gfx950 上 GLM-5.2 DSA prefill 即使在短上下文也总是走 triton sparse-MLA 路径，而稀疏索引器 top-k + gather + mask 的开销超过其剪枝 KV 的收益。密集 MHA 回退已在 NVIDIA SM90/SM100 上使用并由 SGLANG_DSA_PREFILL_DENSE_ATTN_KV_LEN_THRESHOLD 门控，但被硬编码为仅 NVIDIA，尽管 aiter flash_attn_varlen_func 内核在 ROCm 上可用。

实现拆解

1. 设备门控扩展：在 dsa_backend.py 的 set_dsa_prefill_impl() 中，将 use_mha 的设备条件从仅 NVIDIA（SM90/SM100）扩展为包含 _IS_GFX95，使 gfx950 在满足 max_kv_len <= SGLANG_DSA_PREFILL_DENSE_ATTN_KV_LEN_THRESHOLD（GLM-5.2 默认 2048）等条件时启用密集 MHA 预填充。
2. 路由澄清：_forward_standard_mha() 中补充注释说明 gfx950 报告 sm_(9,5)，不会进入 device_sm_major >= 10 的 Blackwell 分支，自然落入 aiter flash_attn_varlen_func 路径，无需额外 _is_hip 判断，NVIDIA 行为不变。
3. FP8 KV 反量化兼容：在 forward_mha.py 的 _get_mla_kv_buffer_from_fp8_for_dsa() 中新增 _use_aiter_gfx95 分支，改用 get_token_to_kv_pool().get_mla_kv_buffer() 反量化原始布局的 FP8 MLA KV，避免 dequantize_k_cache_paged 对缩放布局（dim==656）的断言在 HIP 原始布局（dim==576）下崩溃，并处理 DCP 本地索引过滤。
4. 验证与部署：无新增单元测试；作者提供 GSM8K 0.955 准确率与 MI355X TP4 基准，依赖 #30519、#30715 及 aiter 调优 MoE 配置。CI 中该路径未被实际执行，失败均与无关子系统相关。

关键文件：

- python/sglang/srt/layers/attention/dsa_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 set_dsa_prefill_impl, _forward_standard_mha）：核心设备门控逻辑：将 gfx95x 纳入 use_mha 预填充回退条件，并澄清 _forward_standard_mha 的 ROCm 路由。
- python/sglang/srt/models/deepseek_common/attention_forward_methods/forward_mha.py（模块 前向逻辑；类别 source；类型 data-contract；符号 _get_mla_kv_buffer_from_fp8_for_dsa）：FP8 KV 反量化路径修复：gfx950 使用 HIP 感知的 get_mla_kv_buffer，解决 chunked-prefill 拆分时 KV 布局不匹配崩溃。

关键符号：set_dsa_prefill_impl, _forward_standard_mha, _get_mla_kv_buffer_from_fp8_for_dsa

关键源码片段

python/sglang/srt/layers/attention/dsa_backend.py

核心设备门控逻辑：将 gfx95x 纳入 use_mha 预填充回退条件，并澄清 _forward_standard_mha 的 ROCm 路由。

```
def set_dsa_prefill_impl(self, forward_batch: Optional[ForwardBatch] = None):
    """Decide all attention prefill dispatch strategies for this batch."""
    # 为简洁省略部分 import
    from sglang.srt.utils import get_device_sm, is_blackwell

    # 图重放中不能按 seq_lens_cpu 分支，强制关掉 MHA 以保证正确性。
    if is_in_tc_piecewise_cuda_graph() or is_in_breakable_cuda_graph():
        self.use_mha = False
    elif forward_batch and forward_batch.forward_mode.is_extend_without_speculative():
        assert forward_batch.seq_lens_cpu is not None
        max_kv_len = forward_batch.seq_lens_cpu.max().item()
        sum_seq_lens = sum(forward_batch.seq_lens_cpu)
        device_sm = get_device_sm()

        # 要求：H200/B200/MI355X、短序列、受支持的 dtype、能放进 chunk。
        # 新增 _IS_GFX95：gfx950（MI355X）短上下文也启用密集 MHA，
        # 避免 sparse-MLA 索引器开销；长上下文仍由下方阈值切回 sparse。
        self.use_mha = (
            (
                device_sm == 90
                or (device_sm >= 100 and device_sm < 110)
                or _IS_GFX95
            ) # SM90/SM100（NVIDIA）或 gfx95x（MI355X）
            and max_kv_len
            <= envs.SGLANG_DSA_PREFILL_DENSE_ATTN_KV_LEN_THRESHOLD.get() # 短到划算
            and self.token_to_kv_pool.dtype in [torch.bfloat16, torch.float8_e4m3fn]
            and sum_seq_lens
            <= forward_batch.get_max_chunk_capacity() # 能放进 chunk
            and (not is_dsa_enable_prefill_cp()) # 未启用 CP
            and (self.hisparse_coordinator is None)
```

```
        )
    else:
        self.use_mha = False # Decode/verify 始终用 MLA

    # 未走 MHA 时再决定 MLA 实现（flashmla_sparse / flashmla_kv）
    if not self.use_mha and self.enable_auto_select_prefill_impl:
        # … 原有 MLA 实现选择逻辑保持不变
```

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

FP8 KV 反量化路径修复：gfx950 使用 HIP 感知的 get_mla_kv_buffer，解决 chunked-prefill 拆分时 KV 布局不匹配崩溃。

```
def _get_mla_kv_buffer_from_fp8_for_dsa(
    self: DeepseekV2AttentionMLA,
    forward_batch: ForwardBatch,
):
    """Dequantize FP8 KV cache to BF16 for MLA attention (DSA-specific format).

    返回 (kv_a, k_pe)，均为 BF16。
    """
    backend = get_attn_backend()
    if isinstance(backend, TboAttnBackend): # 若启用 tbo，取 primary backend
        backend = backend.primary
    kv_indices = backend.forward_metadata.page_table_1_flattened
    assert kv_indices is not None, \
        "page_table_1_flattened should have been generated for FP8 MHA path"

    if _use_aiter_gfx95:
        # ROCm（gfx950）以原始 (kv_lora_rank + qk_rope_head_dim) 布局存储
        # FP8 MLA KV，而 dequantize_k_cache_paged 期望 scaled 656 字节布局
        # （内部断言 dim == 656）。这里改用 pool 的 HIP 感知的
        # get_mla_kv_buffer 反量化原始布局，与 BF16 MHA 路径一致。
        # 缺少此分支时，chunked-prefill 拆分（extend_prefix_lens != 0）
        # 读取缓存 prefix KV 会以 "576 != 656" 崩溃。
        kv_indices = filter_dcp_local_kv_indices(kv_indices=kv_indices)
        kv_a, k_pe = get_token_to_kv_pool().get_mla_kv_buffer(
            self.attn_mha, kv_indices, torch.bfloat16
        )
        kv_a = kv_a.squeeze(1).contiguous()
        return kv_a, k_pe

    # NVIDIA/MUSA/CPU 等继续走 scaled FP8 的反量化路径。
    kv_cache_fp8 = get_token_to_kv_pool().get_key_buffer(self.attn_mha.layer_id)
    kv_latent_bf16 = dequantize_k_cache_paged(kv_cache_fp8, kv_indices)

    kv_a = kv_latent_bf16[:, :, : self.kv_lora_rank].squeeze(1).contiguous()
    k_pe = kv_latent_bf16[:, :, self.kv_lora_rank :]
```

```
    return kv_a, k_pe
```

评论区精华

1. FP8 KV 布局崩溃：Jacob0226 报告 chunked-prefill 拆分时 dequantize_k_cache_paged 断言 576 != 656，Raiden-Makoto 修复为 gfx950 走 HIP 感知的 get_mla_kv_buffer 路径，后续确认不再崩溃。
2. k_pe/k_nope 形状不匹配：Jacob0226 在 GLM-5.1-FP8 上复现 _concat_and_cast_mha_k 的 token 数不匹配崩溃，Raiden-Makoto 定位为 refetch 后复用了 stale 的 kv_a_quanted，修复后 GSM8K 0.931 且无异常。
3. 性能回归争议：clintg6 最初报告高并发下 E2E 吞吐下降约 8%，Raiden-Makoto 在干净环境下 A/B 验证无回归（TTFT -14%~-28%，TPOT/E2E 持平），clintg6 在更新容器后确认无回归并批准。
4. 代码风格讨论：sogalin 建议复用 is_gfx95_supported() 替代本地 _detect_gfx950()，Raiden-Makoto 已切换；同时移除多余的 _is_hip 判断（gfx950 报告 sm_(9,5) 已天然避开 SM100+ 分支）。
5. CI 覆盖缺口：amd-bot 指出没有 PR-CI 测试实际执行此 gfx950 + DSA + FP8 chunked-prefill 路径，绿色运行不能验证该功能，相关失败均与无关子系统相关。

- FP8 KV 布局不匹配导致 576 != 656 崩溃 (correctness): 通过 gfx950 分支改用 HIP 感知的 get_mla_kv_buffer，解决 KV 布局差异。
- chunked-prefill 拆分时 k_pe 与 k_nope 形状不匹配 (correctness): 修复 refetch 后重建 k_pe，确保 token 对齐。
- 高并发下 E2E 性能回归争议 (performance): 回归源于旧容器 / 周边栈，PR 本身无解码回归。
- 使用共享 is_gfx95_supported 替代本地 _detect_gfx950 (design): 统一使用共享 helper，移除冗余 _is_hip 条件。
- CI 未实际覆盖 gfx950 DSA 路径 (testing): 代码合并依赖人工 review 与作者提供的 benchmark，测试覆盖缺口未解决。

风险与影响

- 风险：
  1. 缺少测试覆盖：PR 未新增单元测试，该路径依赖 gfx950 + DSA/MLA + FP8 chunked-prefill 组合，PR-CI 中无实际覆盖，存在回归风险。
  2. FP8 KV 布局契约脆弱性：forward_mha.py 依赖 get_mla_kv_buffer 的 HIP 原始布局行为，若该函数或 KV 池布局变化，可能导致 576 != 656 类问题回归。
  3. 阈值选择的经验性：默认阈值与模型 index_topk 绑定（2048），在不同模型或上下文分布下可能不是最优，长上下文仍走 sparse 路径，需要针对负载调优。
  4. 依赖栈不稳定性：基准依赖 #30519、#30715 和 aiter 调优配置，若这些上游未合并或 aiter 版本变化，性能收益可能打折。
  5. 硬件特定性：仅 gfx950 开启，其他 AMD 设备（gfx90a/gfx942）不受影响，但该判定基于 _IS_GFX95，若检测逻辑变化可能影响 MI350/MI355X 全系。 - 影响：对 gfx950

（MI355X）上使用 DSA/MLA 的 GLM-5.x/DeepSeek 模型用户，短上下文（kv_len <= 2048）时 TTFT 显著下降（22%-43%），吞吐小幅提升；长上下文仍走 sparse-MLA 路径，解码阶段不受影响。NVIDIA 及其他 AMD 平台行为完全不变。对团队的影响是新增了一个硬件相关的性能开关，且需要在 gfx950 上维护 FP8 KV 布局兼容性，未来同类问 题可在 forward_mha.py 中集中处理。 - 风险标记：缺少测试覆盖，核心路径变更，硬 件特定分支，FP8 KV 布局契约依赖

关联脉络

- PR #34837 [AMD] Add concat_and_cast_mha_k_pad_kernel to support 12-head and enable K3 aiter prefill kernel: 同一 concat_and_cast_mha_k 前向路径的 AMD aiter 预填充增强，本 PR 中该函数是崩溃点，两者共同演进 AMD MHA 前向支持。
- PR #30900 [AMD][Quantization][Bugfix] Fix bug related to fp8 max on gfx95x for per-token-group quant (ROCm): 同为 gfx95x 上的 FP8 量化正确性修复，本 PR 的 FP8 KV 布局问题也属于同类 AMD FP8 路径缺陷。
- PR #34517 [AMD][Spec] Accelerate Qwen3.5 verification with grouped-head shared KV: 同为 AMD 上 MLA 注意力性能优化，且都涉及 verify/prefill 的 kernel 选择，说明 AMD 注意力路径正在被系统性优化。