PR #28757 完整报告

sgl-project/sglang

[AMD] [GLM5] skip redundant -inf pre-fill of HIP indexer MQA-logits

合并时间：2026-06-25 14:21

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28757

执行摘要

- 一句话：跳过冗余 -inf 预填充以加速 HIP DSA 预填充
- 推荐动作：该 PR 体现了通过参数对齐消除冗余计算的典型优化手段，且提供了详尽的基准测试和单元测试验证。对于关注 AMD 平台长上下文推理性能的开发者，值得精读。设计决策值得在其他类似场景中借鉴。

功能与动机

在 HIP DSA 索引器路径上，_get_topk_ragged 调用 aiter 的 fp8_mqa_logits 时默认 clean_logits=True，导致每次层前都会初始化一个 [tokens x seq_len_kv] 的全 -inf 张量。该预填充随上下文长度二次增长，在 16x8192 的 gfx950 上占预填充 GPU 时间的约 11%。CUDA deep_gemm.fp8_mqa_logits 路径已传入 clean_logits=False 并依赖 topk_transform 屏蔽无效位置，因此 HIP 路径也应对齐以避免这一冗余工作。

实现拆解

1. 在 python/sglang/srt/layers/attention/dsa/dsa_indexer.py 的 _get_topk_ragged 方法中，修改了两个 HIP 分支对 fp8_mqa_logits 的调用：在非分块分支和分块分支中均添加 clean_logits=False 参数。原 HIP 调用省略了该参数，默认 clean_logits=True；现显式设置与 CUDA 路径一致。
2. 新增 test/registered/amd/test_dsa_skip_logits_clean.py 单元测试，注册到 AMD CI 套件 stage-b-test-1-gpu-small-amd-mi35x。测试通过运行 fp8_mqa_logits 在 clean_logits=True 和 clean_logits=False 两种模式下，分别应用生产级 fast_topk_v2 进行 masked topk，验证最终选中的 KV 位置一致。测试还包含健全性检查，确保 clean_logits=False 确实在无效位置留下了非 -inf 的值。
3. 测试精度：GSM8K 5-shot 精度从 0.936 变为 0.938（无回归）。速度基准测试（sglang.bench_serving，输入 8192 / 输出 1024）显示 TTFT 中位数改善 1.8%-2.7%，ITL 无变化，E2EL 持平略好。

关键文件：

- python/sglang/srt/layers/attention/dsa/dsa_indexer.py（模块 DSA 索引器；类别 source；类型 core-logic）：核心修改文件：在 _get_topk_ragged 方法中为 HIP 路径的 fp8_mqa_logits 调用添加 clean_logits=False，消除冗余预填充。

- test/registered/amd/test_dsa_skip_logits_clean.py（模块 AMD 测试；类别 test；类型 test-coverage；符号 _cast_kv_to_fp8, TestDSASkipLogitsClean, _run_case, test_skip_logits_clean_topk_equivalence）：新增单元测试，验证 clean_logits=False 下 topk 选择与 clean_logits=True 一致，并注册到 AMD CI。

关键符号：_get_topk_ragged

关键源码片段

python/sglang/srt/layers/attention/dsa/dsa_indexer.py

核心修改文件：在 _get_topk_ragged 方法中为 HIP 路径的 fp8_mqa_logits 调用添加 clean_logits=False，消除冗余预填充。

```
# python/sglang/srt/layers/attention/dsa/dsa_indexer.py
# 在 _get_topk_ragged 方法中，非分块分支：
if not need_chunk:
    with self._with_real_sm_count():
        if _is_hip:
            from aiter.ops.triton.fp8_mqa_logits import fp8_mqa_logits
            kv, scale = kv_fp8
            # 匹配 CUDA deep_gemm 路径 (clean_logits=False)：topk_transform
            # 通过 ks/ke/lengths 屏蔽无效位置，因此对 logits 缓冲区的 -inf
            # 预填充是冗余的，且随上下文长度二次增长。
            logits = fp8_mqa_logits(
                q_fp8[:q_offset],
                kv,
                scale,
                weights[:q_offset],
                ks,
                ke,
                clean_logits=False, # <-- 新增参数
            )
        else:
            logits = deep_gemm.fp8_mqa_logits(
                q_fp8[:q_offset],
                kv_fp8,
                weights[:q_offset],
                ks,
                ke,
                clean_logits=False, # 现有 CUDA 路径已设置
            )

# 分块分支类似，也添加了 clean_logits=False。
```

test/registered/amd/test_dsa_skip_logits_clean.py

新增单元测试，验证 clean_logits=False 下 topk 选择与 clean_logits=True 一致，并注册到 AMD CI。

```
# test/registered/amd/test_dsa_skip_logits_clean.py
```

```
class TestDSASkipLogitsClean(CustomTestCase):
    def _run_case(self, s_q, s_k, num_heads, head_dim, topk, seed=0):
        # 准备随机张量，模拟 KV 缓存中的有效范围 [ks, ke)
        ks = torch.zeros(s_q, dtype=torch.int32, device='cuda')
        ke = torch.randint(s_k // 2, s_k + 1, (s_q,), dtype=torch.int32, device='cuda')

        # 运行两种情况
        logits_clean = fp8_mqa_logits(q_fp8, kv_fp8, scales, weights, ks, ke, clean_logits=True)
        logits_dirty = fp8_mqa_logits(q_fp8, kv_fp8, scales, weights, ks, ke, clean_logits=False)

        # 健全性检查：clean_logits=False 应在无效位置留下非 -inf 的值
        # （否则测试无意义）
        for i in range(s_q):
            if ke[i] < s_k and (logits_dirty[i, ke[i]:] != float('-inf')).any():
                break

        # 应用生产级 masked topk
        lengths = (ke - ks).to(torch.int32)
        topk_clean = fast_topk_v2(logits_clean, lengths, topk, row_starts=ks)
        topk_dirty = fast_topk_v2(logits_dirty, lengths, topk, row_starts=ks)

        # 断言选中的 KV 索引完全一致
        for i in range(s_q):
            sel_clean = sorted(x for x in topk_clean[i].tolist() if x >= 0)
            sel_dirty = sorted(x for x in topk_dirty[i].tolist() if x >= 0)
            self.assertEqual(sel_clean, sel_dirty,
                f"topk selection differs at row {i}")
```

评论区精华

- HaiShaw 要求运行 pre-commit 以避免 lint 错误。
- HaiShaw 要求添加单元测试覆盖该代码变更；Raiden-Makoto 随后添加了测试文件并上传了测试结果。
- amd-bot 的 CI 状态指出该变更未被常规 PR CI 测试覆盖，但所有失败均与此 PR 无关，且该路径由 nightly=True 的 AMD 套件覆盖。添加的专用测试解决了这一覆盖缺口。
- 请求添加单元测试 (testing): 测试已添加，并通过 AMD CI 验证。

风险与影响

- 风险：低风险。变更仅添加参数 clean_logits=False，逻辑与 CUDA 路径完全一致。通过专门的单元测试验证了 topk 选择的一致性。精度基准测试 GSM8K 显示无回归。性能基准测试确认无 decode 回归。风险在于该路径仅在 AMD gfx950 平台上执行，而新测试已覆盖该硬件。
- 影响：影响范围：仅影响使用 HIP DSA 索引器的模型（如 GLM-5.1-MXFP4）在 AMD gfx950 上的长上下文预填充性能。TTFT 中位数提升 1.8%-2.7%，decode 延迟无变化。对其他硬件（NVIDIA）或非 DSA 模型无影响。影响程度：正面，可减少预填充瓶颈。

- 风险标记：低回归风险，已验证精度正确

关联脉络

- 暂无明显关联 PR