PR #36004 完整报告

sgl-project/sglang

[AMD][DSV4] perf: use full 1024-thread block for indexer top-k on ROCm

合并时间：2026-08-23 15:22

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36004

执行摘要

- 一句话：ROCm 上 DSv4 top-k 用满 1024 线程块，解码吞吐提升约 3%
- 推荐动作：值得精读，尤其适合关注 GPU kernel 性能分析或 AMD ROCm 优化的工程师。两个值得借鉴的设计决策：一是用「kernel 时间是否随 batch 增长」来判定延迟受限还是吞吐受限，从而确定优化方向；二是单元测试 pin 住 kernel 契约（结果正确性）而非具体 block size，让测试在 512/1024 两档下都有效并顺带守护 CUDA 路径。若要跟进，可关注 CI 失败原因与 GSM8K 微小波动的后续表现。

功能与动机

deepseek_v4_topk_transform 每行只启动一个 block，成本集中在 O(c4_len) 的直方图与发射阶段。PR body 给出的关键证据是：128k 上下文下 kernel 时间从 batch 8 到 batch 256 基本平坦（约 36.8 us），说明瓶颈是单 block 扫描延迟（latency-bound）而非占用率；而 CDNA 的 wavefront 是 64 lanes，512 线程只有 8 个波前，仅用掉一半扫描宽度。把 ROCm 分支的 kBlockSize 提到 1024，即可直接翻倍每 block 的扫描并行度，且不改变选中索引集。作者同时用 GSM8K 全量 1319 题与 3860 行索引集对照做了正确性验证。

实现拆解

1. 变更入口与核心逻辑：python/sglang/kernels/aot/csrc/elementwise/deepseek_v4_topk.cu 新增 #ifdef USE_ROCM 条件编译，将常量 kBlockSize 拆为平台两档——ROCm 下 1024、CUDA 下保持 512（#else 分支）。该常量直接决定每行 block 的线程规模，影响直方图与发射阶段的扫描宽度，是本次性能收益的唯一来源。
2. 性能论证与收益规模：作者先以「kernel 时间不随 batch 增长」判定延迟受限，再结合 CDNA 64-lane wavefront 推导出 512 线程只利用了一半扫描宽度。隔离测试显示 128k/64k/32k/8k 上下文分别提速 1.60x/1.43x/1.30x/1.07x，无退化；端到端在 MI355X TP8 上并发 16 时吞吐 +3.0%，并发 64 时降至 +1.6%，与「固定 kernel 节省在更大 decode 步骤中占比变小」的推断一致。
3. 正确性验证：GSM8K 5-shot 全量 1319 题对比为 baseline 94.62% vs 本 PR 94.47%（差 2 题）；作者另在 3860 行（c4_len 2048-32768、batch 1/64/128、4 个 seed）上确认选中索引集与旧 kernel 完全一致。
4. 测试配套：在 python/sglang/kernels/aot/tests/test_topk.py 新增 test_deepseek_v4_topk_transform，以 torch.topk 为 ground truth，恒等页表让发射的 paged slot 直接等于 token 位置；参数化覆盖 c4_len = 2048/8192/32768 × bs = 1/48

，其中 32768 正是 128k 上下文形态。测试 pin 住 kernel 契约而非具体 block size，因此 512/1024 两档都能通过，兼作 CUDA 路径的守护；op 仅 ROCm 导出，NVIDIA 上自动跳过。

5. CI 与审查状态：PR 面板显示 PR Test（Base）与 AMD ROCm 7.2 两个 run 失败、Extra run 通过；HaiShaw 直接 approve 且无 review 评论。失败原因在提供材料中未说明，需结合 CI 日志确认与本次改动无关。

关键文件：

- python/sglang/kernels/aot/csrc/elementwise/deepseek_v4_topk.cu（模块 TopK 内核；类别 source；类型 core-logic；符号 kBlockSize）：核心性能改动：通过 #ifdef USE_ROCM 将 kBlockSize 从 512 提升到 1024，是本次端到端吞吐提升（1.6%~3.0%）的唯一来源；CUDA 路径保持 512 不变，风险隔离清晰。
- python/sglang/kernels/aot/tests/test_topk.py（模块 TopK 内核；类别 test；类型 test-coverage；符号 test_deepseek_v4_topk_transform）：新增 test_deepseek_v4_topk_transform，补上该 op 长期缺失的单元测试；以 torch.topk 为基准、恒等页表简化对照，并覆盖 128k 上下文形态，实现对 kernel 契约的守护。

关键符号：test_deepseek_v4_topk_transform

关键源码片段

python/sglang/kernels/aot/csrc/elementwise/deepseek_v4_topk.cu

核心性能改动：通过 #ifdef USE_ROCM 将 kBlockSize 从 512 提升到 1024，是本次端到端吞吐提升（1.6%~3.0%）的唯一来源；CUDA 路径保持 512 不变，风险隔离清晰。

```
namespace {
constexpr uint32_t kMaxTopK = 1024;

#ifdef USE_ROCM
// CDNA3/CDNA4 上该 kernel 以一行一个 block 的方式运行，瓶颈在 O(c4_len) 的
// 直方图与发射阶段，属于延迟受限而非吞吐受限。CDNA 的 wavefront 是 64 lanes，
// 512 线程只有 8 个波前，扫描宽度减半；1024 线程满 block（16 个波前）在 128k
// 上下文（c4_len = 32768）下约提速 1.6 倍，短上下文下也无退化。
constexpr uint32_t kBlockSize = 1024;
#else
// CUDA 路径维持 512，不改动 NVIDIA 平台行为。
constexpr uint32_t kBlockSize = 512;
#endif
```

python/sglang/kernels/aot/tests/test_topk.py

新增 test_deepseek_v4_topk_transform，补上该 op 长期缺失的单元测试；以 torch.topk 为基准、恒等页表简化对照，并覆盖 128k 上下文形态，实现对 kernel 契约的守护。

```
@pytest.mark.skipif(
    torch.version.hip is None,
    reason="deepseek_v4_topk_transform_512 只在 ROCm 上构建",
)
```

```
@pytest.mark.parametrize("bs", [1, 48])
@pytest.mark.parametrize("c4_len", [2048, 8192, 32768])
@torch.inference_mode()
def test_deepseek_v4_topk_transform(bs: int, c4_len: int) -> None:
    # c4_len 32768 对应 128k 上下文的 decode 形态，是 kernel 扫描最长、
    # 对启动 block 大小最敏感的场景。
    from sgl_kernel import deepseek_v4_topk_transform_512

    torch.manual_seed(42)
    topk, page_size = 1024, 64

    scores = torch.randn(bs, c4_len, dtype=torch.float32, device="cuda")
    seq_lens = torch.full((bs,), c4_len, dtype=torch.int32, device="cuda")

    # 恒等页表让发射出的分页槽位正好等于原始 token 位置，从而可以直接与
    # torch.topk 的索引比较，不需要额外还原映射关系。
    num_pages = (c4_len + page_size - 1) // page_size
    page_table = (
        torch.arange(num_pages, dtype=torch.int32, device="cuda")
        .unsqueeze(0)
        .expand(bs, -1)
        .contiguous()
    )
    page_indices = torch.full((bs, topk), -1, dtype=torch.int32, device="cuda")

    deepseek_v4_topk_transform_512(
        scores, seq_lens, page_table, page_indices, page_size
    )

    indices_ref = torch.topk(scores, topk, dim=-1, sorted=False).indices
    assert_equal(
        scores,
        torch.sort(indices_ref, dim=-1).values,
        torch.sort(page_indices, dim=-1).values,
        bs,
        topk,
        c4_len,
    )
```

评论区精华

整个 review 只有一条讨论：HaiShaw 在关联 issue 的评论中标注「HIP CDNA3/4 specific」，界定了优化的适用平台，与代码中#ifdefUSE_ROCM的隔离方式一致，随后直接approve、无 review 评论。另有价值的权衡分析藏在 PR body 里：作者明确写出收益随并发升高而收窄的规律（吞吐提升从 3.0% 降至 1.6%），并用 torch-profiler 整步分解（每 decode 步 kernel 执行 30 次、节省 431 us / 21.5 ms ≈ 2.0%）与端到端测量相互印证，避免了「只看 kernel 加速、忽略端到端占比」的常见误区。

- 平台适用范围：HIP CDNA3/4 specific (design): 改动被严格限定在 ROCm 平台，HaiShaw 直接 approve，无未解决疑虑。

风险与影响

- 风险：
  1. 行为一致性：GSM8K 全量出现 2 题准确率差异（94.62% → 94.47%），虽然作者声称索引集在 3860 行上完全一致，端到端仍出现波动，后续跑分需留意是否复现。
  2. 资源占用：1024 线程块可能提高共享内存与寄存器压力，PR body 只覆盖了目标上下文形态，更长序列或更小 topk 场景未覆盖。
  3. CI 状态：PR Test（Base）与 AMD ROCm 7.2 两个 run 显示失败，当前材料无法判断失败原因，合入前应确认与本次 kernel 改动无关。
  4. 兼容性：CUDA 路径保持 512，NVIDIA 平台零影响；SGL_TOPK_DYNAMIC_SMEM_BYTES 动态共享内存逻辑未被触碰。 - 影响：影响范围：仅 USE_ROCM 编译的 CDNA3/4 平台、DeepSeek-V4 索引器（indexer）的 top-k kernel；NVIDIA/CUDA 完全不受影响。对 AMD 上部署 DSv4 的用户，decode 阶段 TPOT 改善 1.6%~2.9%、总吞吐提升 1.6%~3.0%，属于低风险的平台性能优化。对团队而言，该改动与 AMD DSv4 cookbook 更新（PR#35854）形成配套，进一步完善 ROCm 上 DSv4 的部署体验；同时新增的单元测试弥补了该 op 长期无测试的缺口。 - 风险标记：平台限定改动（ROCm/CDNA），GSM8K 出现 2 题准确率波动，CI 面板存在失败记录

关联脉络

- PR #35854 [AMD] Update amd deepseek v4 cookbook 0822: 同属 AMD 上 DeepSeek-V4 的配套演进：前者更新部署 cookbook 与融合配置，本 PR 提供 kernel 级性能优化，两者共同完善 ROCm 上的 DSv4 体验。