

PR #53382 完整报告

vllm-project/vllm

[Perf][Kernel] Tune cooperative topk for medium batch-sizes

合并时间: 2026-09-01 04:44

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/53382>

执行摘要

- 一句话: 调优 top-k 协作内核, 中批解码提速最高 3.3x
- 推荐动作: 值得精读。该 PR 展示了如何通过新增更小 cluster size 的内核变体来拓展既有协作内核的适用 batch 范围, 并用参数化测试固定新派发边界。建议重点关注 cooperative_topk.cuh 中 CS=2 的 streaming 双 pass 设计, 以及 sparse_attn_indexer.py 中 dispatch 条件的变化; 同时留意测试对中间 batch 值的覆盖率不足, 后续可补充。

功能与动机

在 main 分支中, 当 batch size 超过 32 时, sparse_attn_indexer 的 use_cooperative_topk 判断会失效, 导致后端回退到 FilteredTopK, 而后者在中 batch、大上下文场景下性能明显不如 cooperative_topk。PR body 提供的 benchmark 表显示, cooperative_topk 在 33-64 行区间相对 FilteredTopK 的加速普遍在 1.4x 以上, 最高超过 3x。因此本 PR 的目标就是把 cooperative_topk 的支持行数从 32 提升到 64, 并为新增区间设计更合适的协作粒度配置, 从而填上这一段性能空洞。

实现拆解

1. 新增 CS=2 内核模板: 在 csrc/libtorch_stable/cooperative_topk.cuh 中新增 cooperative_topk_cs2 全局 kernel, 使用 __cluster_dims__(1, 2, 1), 复用已有的双 pass streaming 路径, kSmemSize2 直接复用 kSmemSize4 的尺寸。
2. 更新 launch 派发逻辑: 在 csrc/libtorch_stable/cooperative_topk.cu 的 launch_cooperative_cluster 中显式支持 CS == 2; 在 launch_cooperative_topk_impl 中把 num_rows 上限从 32 放宽到 64, 并把最后两级派发拆成 num_rows <= 33 使用 CS=4, 否则使用 CS=2。
3. 放宽调用方条件: vllm/model_executor/layers/sparse_attn_indexer.py 中 use_cooperative_topk 的判定从 num_rows <= 32 改为 num_rows <= 64, 其他平台、top_k 集合和 stride 对齐条件保持不变, 确保 64 行以上仍然回退到 persistent_topk 或 FilteredTopK。
4. 测试配套: tests/kernels/test_top_k_per_row.py 将 _run_topk_backend 中 cooperative_topk 的跳过阈值改为 64, 并新增参数化测试 test_cooperative_topk_cs2, 覆盖 batch=64、seq_len=65536、top_k 为 512/1024/2048 的场景, 直接验证 CS=2 路径的数值正确性。

关键文件：

- `vllm/model_executor/layers/sparse_attn_indexer.py`（模块 稀疏索引；类别 source；类型 core-logic；符号 `sparse_attn_indexer`）：这是让 `cooperative_topk` 在 33-64 行区间被实际选中的关键改动，仅一行阈值变化就决定了线上 dispatch 行为。
- `csrc/libtorch_stable/cooperative_topk.cu`（模块 协作内核；类别 source；类型 core-logic；符号 `launch_cooperative_cluster`, `launch_cooperative_topk_impl`）：内核 `launch` 入口的派发逻辑在这里更新，新增 CS=2 分支并把行数上限提高到 64。
- `csrc/libtorch_stable/cooperative_topk.cuh`（模块 协作内核；类别 source；类型 core-logic；符号 `cooperative_topk_cs2`）：定义了新的 CS=2 全局 kernel 模板，是新增内核路径的核心实现。
- `tests/kernels/test_top_k_per_row.py`（模块 内核测试；类别 test；类型 test-coverage；符号 `test_cooperative_topk_cs2`）：更新了测试对 `cooperative_topk` 行数上限的假设，并新增对 64 行的回归测试，防止新路径退化。

关键符号：`sparse_attn_indexer`, `launch_cooperative_cluster`,
`launch_cooperative_topk_impl`, `cooperative_topk_cs2`, `test_cooperative_topk_cs2`

关键源码片段

`vllm/model_executor/layers/sparse_attn_indexer.py`

这是让 `cooperative_topk` 在 33-64 行区间被实际选中的关键改动，仅一行阈值变化就决定了线上 dispatch 行为。

```
# vllm/model_executor/layers/sparse_attn_indexer.py
# 选择 top-k 后端: cooperative_topk 仅在 CUDA sm90（非 sm120）、
# top_k 为 512/1024/2048、行数不超过 64 且满足 TMA 对齐时启用。
# 行数上限从 32 放宽到 64，正好覆盖新增的 CS=2 内核。
use_cooperative_topk = (
    current_platform.is_cuda()
    and topk_tokens in (512, 1024, 2048)
    and num_rows <= 64
    and logits.stride(0) % 4 == 0 # TMA 16-byte 对齐
    and current_platform.has_device_capability(90)
    and not current_platform.is_device_capability_family(120)
)

if use_cooperative_topk:
    # 复用 workspace manager 的预分配缓冲区，避免每次调用重新分配
    workspace_manager = current_workspace_manager()
    (topk_workspace,) = workspace_manager.get_simultaneous(
        ((RADIX_TOPK_WORKSPACE_SIZE,), torch.uint8),
    )
    torch.ops._C.cooperative_topk(
        logits,
        seq_lens,
        topk_indices,
```

```
    topk_workspace,  
    topk_tokens,  
    attn_metadata_narrowed.max_seq_len,  
)
```

csrc/libtorch_stable/cooperative_topk.cu

内核 launch 入口的派发逻辑在这里更新，新增 CS=2 分支并把行数上限提高到 64。

```
// csrc/libtorch_stable/cooperative_topk.cu  
// 新增 CS=2 分支：行数超过 33 但不超过 64 时，使用两 CTA 协作内核，  
// 避免 CS=4 在更大 batch 下的协作开销  
if (num_rows <= 33) {  
    launch_cooperative_cluster<TopK, 4>(params, ct::kSmemSize4, stream);  
} else {  
    // 此前该区间只能回退到 FilteredTopK，现在由 CS=2 接管  
    launch_cooperative_cluster<TopK, 2>(params, ct::kSmemSize2, stream);  
}
```

评论区精华

该 PR 没有出现实质性的 inline review 讨论。mgoin 通过 `/ci run` 触发 Buildkite CI（#86392），随后直接批准合并；claude[bot] 因 PR 来自 fork 自动跳过了审查。因此主要的技术论证都体现在 PR body 的 benchmark 表格中，用三组 topK（512/1024/2048）、四种 batch（33/40/48/64）的数据支撑了扩展行数上限的决策。

- 暂无高价值评论线程

风险与影响

- 风险：边界值覆盖不足：新增测试只覆盖 batch=64，33、40、48 等中间行数没有单独用例，若 CS=2 在非 64 行时存在边界问题，现有测试无法发现。dispatch 条件中保留了原有的 `num_rows <= 16 / <= 8` 分支，后者实际上不可达（16 分支已覆盖），但这并非本 PR 引入，属于既有代码的边界冗余。kSmemSize2 直接复用 kSmemSize4，后续若调整 CS=4 的 smem 配置可能意外影响 CS=2。回退路径完整：当行数超过 64、top_k 不在 512/1024/2048、stride 未对齐或平台非 sm90（含 sm120 排除）时，仍会走 persistent_topk 或 FilteredTopK，不会引入功能性错误。
- 影响：影响范围集中在使用稀疏注意力的大上下文模型（如 DeepSeek）的解码阶段，中等 batch（33-64 行）的 top-k 索引步骤会有 1.4x-3.3x 的加速；短序列、小 batch 或非 sm90 平台不受影响。新增一个 CUDA 内核实例会增加编译时间和二进制体积，但运行时开销可忽略。团队内核维护者需要理解 CS=2 的协作模式，后续若要继续扩大 batch 范围需要额外评估 CS=1 或 persistent 方案的取舍。
- 风险标记：新增 CS=2 内核变体，中间行数覆盖不足，dispatch 条件变更

关联脉络

- PR #53574 [Bugfix][SM120] DSv4: pass contiguous C128A decode topk indices on SM120: 同为 DeepSeek 稀疏注意力 top-k 索引路径的 kernel 修复，与本 PR 在

sparse_attn_indexer / DeepSeek 场景上有功能关联。