PR #26206 完整报告

sgl-project/sglang

[GDN] Optimize prefill QKV split dispatch

合并时间：2026-06-02 16:48

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26206

执行摘要

- 一句话：融合 Triton kernel 优化 GDN prefill QKV 拆分
- 推荐动作：该 PR 值得精读，尤其关注以下设计决策：
  - 融合 kernel 的 stride 支持：同时支持连续和非连续输入，避免额外 contiguous() 调用。
  - 单配置 autotune 与环境变量：在安全性（避免多配置破坏状态池）和灵活性（允许调优）之间取得平衡。
  - 渐进式优化：先从最明显的开销切入，通过量化数据验证收益。后续可进一步优化 strided 输入路径。

功能与动机

通过 PyTorch profiler 定位到 GDN prefill 中 post-conv QKV 拆分操作（torch.split(...).view(...).contiguous()）在 H200 上消耗了 18.97ms，是 prefill 中一个显著热点。该操作在 FLA 输入防护中触发多个 aten::copy_ kernel launch。为了降低开销，作者实现了一个融合 kernel，在单次 launch 内完成拆分和重新布局。同时，为了避免 chunk_delta_h kernel 的多配置 autotune 损坏状态池，但又要允许调优，将该 kernel 的硬编码配置改为环境变量参数化。

实现拆解

1. 新增 fused QKV split Triton kernel（python/sglang/jit_kernel/triton/gdn_fused_proj.py）：
   - 定义 fused_qkv_split_gdn_prefill_kernel JIT kernel，接受输入 tensor mixed_qkv 及其步长（stride），通过 tl.load 按 token 读入，再根据编译时常量计算出的偏移分别 tl.store 到 Q、K、V 输出 tensor。
   - 上层包装函数 fused_qkv_split_gdn_prefill 根据输入确定输出 shape，以 seq_len 为一维 grid 启动 kernel，BLOCK_SIZE 自动设为 qkv_dim 的 next power of 2。
2. 集成到 GDN backend（python/sglang/srt/layers/attention/linear/gdn_backend.py）：
   - 在 forward_extend 方法中，通过条件判断（is_cuda() 且 qkv_dim <= MAX_FUSED_QKV_SPLIT_DIM = 8192）选择是否调用 fused 函数。若条件不满足（非 CUDA 或维度超限），则 fallback 到原有的 torch.split + view 路径。
   - 由于 fused 函数期望的输入可能来自 causal_conv1d_fn 的结果，该结果在 transpose 后可能非连续，kernel 内部通过 stride 参数处理。

3. 参数化 chunk_delta_h kernel 配置（python/sglang/srt/layers/attention/fla/chunk_delta_h.py）：
   - 引入三个环境变量 SGLANG_GDN_CHUNK_H_BV、SGLANG_GDN_CHUNK_H_NUM_WARPS、SGLANG_GDN_CHUNK_H_NUM_STAGES，默认值分别为 32、4、2（与之前硬编码一致），使用 int(os.getenv(key, default)) 读取。
   - 保持单 config 的 @triton.autotune 装饰器不变，避免多 config 导致 autotune 阶段损坏状态池。配置更新注释，说明环境变量的用途。
4. 新增微基准测试（benchmark/bench_linear_attention/bench_gdn_qkv_split.py）：
   - 实现 split_reference 模拟旧路径，fused_qkv_split_gdn_prefill 调用新路径，使用 torch.testing.assert_close 做正确定性验证。
   - 分别对 contiguous 和 strided 布局进行性能测试，输出 baseline 和 fused 的延迟及加速比。

关键文件：

- python/sglang/jit_kernel/triton/gdn_fused_proj.py（模块 JIT 内核；类别 source；类型 core-logic；符号 fused_qkv_split_gdn_prefill_kernel, fused_qkv_split_gdn_prefill）：核心新增：融合 QKV split Triton kernel，实现优化的核心逻辑。
- python/sglang/srt/layers/attention/linear/gdn_backend.py（模块 注意力层；类别 source；类型 dependency-wiring）：将 fused kernel 集成到 GDN 前向扩展路径，并添加条件 fallback。
- python/sglang/srt/layers/attention/fla/chunk_delta_h.py（模块 FLA 层；类别 source；类型 configuration）：将硬编码 Triton 配置参数化为环境变量，实现安全调优。
- benchmark/bench_linear_attention/bench_gdn_qkv_split.py（模块 基准测试；类别 source；类型 test-coverage；符号 parse_args, make_non_contiguous_view, split_reference, benchmark）：新增微基准测试脚本，提供性能数据和正确性验证。

关键符号：fused_qkv_split_gdn_prefill_kernel, fused_qkv_split_gdn_prefill, forward_extend, chunk_gated_delta_rule_fwd_kernel_h_blockdim64, split_reference

关键源码片段

python/sglang/srt/layers/attention/fla/chunk_delta_h.py

将硬编码 Triton 配置参数化为环境变量，实现安全调优。

```
# 环境变量配置，默认值与之前硬编码一致
GDN_CHUNK_H_BV = int(os.getenv("SGLANG_GDN_CHUNK_H_BV", "32"))
GDN_CHUNK_H_NUM_WARPS = int(os.getenv("SGLANG_GDN_CHUNK_H_NUM_WARPS", "4"))
GDN_CHUNK_H_NUM_STAGES = int(os.getenv("SGLANG_GDN_CHUNK_H_NUM_STAGES", "2"))

@triton.autotune(
    configs=[
        triton.Config(
            {"BV": GDN_CHUNK_H_BV},
            num_warps=GDN_CHUNK_H_NUM_WARPS,
```

```
            num_stages=GDN_CHUNK_H_NUM_STAGES,
        )
    ],
    key=["H", "K", "V", "BT", "USE_GK", "NT_BUCKET"],
    **autotune_cache_kwargs,
)
@triton.jit(do_not_specialize=["T"])
def chunk_gated_delta_rule_fwd_kernel_h_blockdim64(
    k, v, w, v_new, g, gk, h, initial_state, initial_state_indices,
    cu_seqlens, chunk_offsets, T,
    H: tl.constexpr, Hg: tl.constexpr, K: tl.constexpr, V: tl.constexpr,
    BT: tl.constexpr, BV: tl.constexpr,
    USE_G: tl.constexpr, USE_GK: tl.constexpr,
    USE_INITIAL_STATE: tl.constexpr, INPLACE_UPDATE: tl.constexpr,
    SAVE_NEW_VALUE: tl.constexpr, IS_VARLEN: tl.constexpr, NT_BUCKET: tl.constexpr,
):
    # ... (kernel 实现主体不变 )
```

评论区精华

- 非连续输入兼容性：gemini-code-assist 指出 mixed_qkv 在 forward_extend 中通常是 transposed 后的非连续 tensor，若 fused kernel 不考虑 stride 可能读取错误。作者已在 kernel 中通过 MIXED_QKV_STRIDE_T 和 MIXED_QKV_STRIDE_D 处理，但 reviewer 建议进一步确认或调用 .contiguous()。最终该问题被判定为已解决。
- chunk_delta_h 默认配置回归：commit 记录显示，早期 commit 将 num_stages 默认设为 4，导致在 H100/H200 上因共享内存不足（K=V=256 需要 279KB，而最大 232KB）测试失败。后续将默认值回退到 2，并通过环境变量提供覆盖能力。
- CI 状态：PR 经过多次 rerun-ci 和 lint 修复后通过 extra CI，最终获得 yuan-luo 的批准。
  - 非连续输入兼容性 (correctness): 作者已在 kernel 中添加 stride 参数处理，该评论标记为已处理。PR 最终获批。
  - num_stages 默认值与共享内存限制 (correctness): 后续 commit 将默认 num_stages 回退到 2，并改为环境变量可调，以保证兼容现有硬件。

风险与影响

- 风险：
  - 非连续输入性能退化：当前 fused kernel 对 strided 输入加速仅 1.03x，与 contiguous 的 2.38x 差距明显。若生产环境中输入常为非连续，优化效果有限。可通过在调用 fused 前调用 .contiguous() 来规避，但会增加一次拷贝开销。
  - chunk_delta_h 配置兼容性：环境变量默认值（num_stages=2）兼容现有 NVIDIA GPU，但若用户随意设置为 4，可能在某些 shape 上因共享内存不足导致 kernel 启动失败。建议在文档中说明界限。
  - 单平台验证：所有性能数据仅在 B200 上采集，H100/H200 等其他 GPU 上的效果未明确验证。chunk_delta_h 的共享内存限制在 H100/H200 上已经暴露问题。

  - 缺少直接单元测试：benchmark 脚本包含了正确性检查，但未作为自动化测试集成。改动核心逻辑未添加独立单元测试，可能遗漏边界情况。
- 影响：
  - 用户影响：无 API 变化，使用 Qwen3.6 GDN 模型的用户可自动享受性能提升（Chat 吞吐 +2.7%，TTFT -17%）。其他 GDN 模型（如 Kimi-Linear）也可能受益。
  - 系统影响：新增环境变量 SGLANG_GDN_CHUNK_H_* 可调整 GPU 线程配置，用户可根据硬件情况微调。chunk_delta_h kernel 的配置调整不会影响其他模块。
  - 团队影响：新增的 fused kernel 和 benchmark 脚本需要维护，但其位于独立文件，耦合度低。
  - 风险标记：非连续输入性能退化，chunk_delta_h 配置兼容性，缺少直接单元测试

关联脉络

- 暂无明显关联 PR