PR #24775 完整报告

sgl-project/sglang

Optimize MHC pipeline: DeepGemm, fused norm, fused hc_head

合并时间：2026-05-10 19:03

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24775

执行摘要

- 一句话：优化 DSV4 MHC pipeline：融合 kernel、折叠 reduction、利用 DeepGemm
- 推荐动作：该 PR 展示了高性能 MLA 场景下的 kernel 融合策略，值得研究其折叠 reduction 和使用 triton.next_power_of_2 等技巧，但合并前应确保有端到端 benchmark 验证；对于 DSV4 用户，加速效果明显，建议优先合并。

功能与动机

根据 PR 描述，原始的 mhc_pre 和 hc_head 调用包含分离的 kernel launch（split-K reduction、RMSNorm、线性层等），产生了不必要的 HBM 读写和启动开销。通过融合这些操作，可以显著减少延迟，特别是对于 DSV4 这种每个 decoder layer 调用 2 次 mhc_pre 和 1 次 hc_head 的模型。作者提供了 microbenchmark 证明了融合后的加速效果。

实现拆解

本 PR 的优化分以下步骤实现：

1. 折叠 split-K stage-1 reduction（python/sglang/srt/layers/mhc.py）：在 mhc_pre_big_fuse_tilelang 中新增参数 gemm_last_dim，使 kernel 能够直接接收已经局部归约的 GEMM 输出，从而跳过单独的 stage-1 reduction kernel launch。该行为在 num_tokens <= 2048 时由 _compute_num_split_for_mhc_pre 自动选择最优 split 数。
2. 可选 DeepGemm prenorm GEMM（python/sglang/srt/layers/mhc.py 和 python/sglang/srt/models/deepseek_v4.py）：当环境变量 SGLANG_OPT_DEEPGEMM_HC_PRENORM 启用时，hc_pre 方法调用 deep_gemm.tf32_hc_prenorm_gemm，该核函数同时输出内积和平方和，进一步减少 global memory 访问。
3. 融合 RMSNorm 到 big_fuse（python/sglang/srt/layers/mhc.py）：新增 mhc_pre_big_fuse_with_norm_tilelang tilelang 内核，在原有 big_fuse 中增加一条 pipelined sweep，用于累积 layer_input 的 sum_sq，并应用 rsqrt * norm_weight 后写回 HBM，替代原本分离的 RMSNorm kernel launch。
4. 新增融合hc_head Triton kernel（python/sglang/srt/layers/mhc_head.py，新增151行）：为最后 PP rank 上的 hc_head 算子编写了纯 Triton 内核 _hc_head_kernel，将 RMSNorm、线性投影、Sigmoid 门控和加权求和合并为一个 1-CTA-per-token 的双 pass 内核，消除了多次 kernel launch 和中间张量读写。

5. 模型和 CI 配套调整：DeepseekV4DecoderLayer.forward 中根据 hc_pre 返回的 norm_fused 标志跳过外部 layernorm；hc_head 默认调用 fused_hc_head（保留 torch fallback）；CI 斜杠命令白名单增加了 DSV4 专用 stage。

关键文件：

- python/sglang/srt/layers/mhc_head.py（模块 MHC 层；类别 source；类型 core-logic；符号 _hc_head_kernel, fused_hc_head）：新增文件，定义融合 Triton kernel _hc_head_kernel 和入口函数 fused_hc_head，将 RMSNorm + Linear + Sigmoid-gate + weighted-sum 合并为单个 kernel launch，是本 PR 的重要性能改进组件。
- python/sglang/srt/layers/mhc.py（模块 MHC 层；类别 source；类型 dependency-wiring；符号 _compute_num_split_for_mhc_pre, mhc_pre_big_fuse_with_norm_tilelang）：核心文件，添加了 mhc_pre_big_fuse_with_norm_tilelang（融合 RMSNorm 的 big_fuse 变体）和 _compute_num_split_for_mhc_pre（自动计算 split-K 数量）；同时修改了 mhc_pre_big_fuse_tilelang 以支持可选的 gemm_last_dim，为折叠 stage-1 reduction 做准备。
- python/sglang/srt/models/deepseek_v4.py（模块 模型定义；类别 source；类型 data-contract）：修改 hc_pre 方法签名以接收可选的 norm 参数，并在 hc_head 方法中调用 fused_hc_head；DeepseekV4DecoderLayer.forward 中根据 norm_fused 标志跳过外部 layernorm 调用。
- scripts/ci/utils/slash_command_handler.py（模块 CI 脚本；类别 infra；类型 infrastructure）：将 stage-c-test-dsv4-4-gpu-b200 和 stage-c-test-dsv4-8-gpu-h200 加入 /rerun-stage 白名单，便于 DSV4 测试独立触发。

关键符号：_hc_head_kernel, fused_hc_head, _compute_num_split_for_mhc_pre, mhc_pre_big_fuse_with_norm_tilelang, DeepseekV4Model.hc_pre, DeepseekV4Model.hc_head, DeepseekV4DecoderLayer.forward

关键源码片段

python/sglang/srt/layers/mhc.py

核心文件，添加了 mhc_pre_big_fuse_with_norm_tilelang（融合 RMSNorm 的 big_fuse 变体）和 _compute_num_split_for_mhc_pre（自动计算 split-K 数量）；同时修改了 mhc_pre_big_fuse_tilelang 以支持可选的 gemm_last_dim，为折叠 stage-1 reduction 做准备。

```
@tilelang.jit( pass_configs={ tilelang.PassConfigKey.TL_DISABLE_WARP_SPECIALIZED: True, tilelang.PassConfigKey.TL_DISABLE_TMA_LOWER: True, tilelang.PassConfigKey.TL_PTXAS_REGISTER_USAGE_LEVEL: 10, }, ) defmhc_pre_big_fuse_with_norm_tilelang( gemm_out_mul, gemm_out_sqrsum, hc_scale, hc_base, residual, post_mix, comb_mix, layer_input, norm_weight, hidden_size: int, rms_eps: float, hc_pre_eps: float, hc_sinkhorn_eps: float, hc_post_mult_value: float, sinkhorn_repeat: int, norm_eps: float, n_splits: int = 16, hc_mult: int = 4, gemm_last_dim: int = -1, ): """将 layer_input 的 RMSNorm 融合进 mhc_pre big_fuse kernel。 对于 layer_input 的加权求和，在第一个 sweep 中先累积
```

sum_sq， 然后第二个 sweep 应用 rsqrt(D/ + norm_eps) * norm_weight 并写回 HBM。 """ num_tokens = T.dynamic("num_tokens") hc_mult3 = hc_mult * (2 + hc_mult) if gemm_last_dim < 0: gemm_last_dim = hc_mult3 hidden_block = math.gcd(1024, hidden_size) gemm_out_mul: T.Tensor[[n_splits, num_tokens, gemm_last_dim], T.float32] gemm_out_sqrsum: T.Tensor[[n_splits, num_tokens], T.float32] # ... 其他参数声明省略 ... layer_input: T.Tensor[[num_tokens, hidden_size], T.bfloat16] norm_weight: T.Tensor[[hidden_size], T.bfloat16] ENABLE_PDL = is_arch_support_pdl() with T.Kernel(num_tokens, threads=96) as i: # 累计 rms sum_sq rms = T.alloc_fragment(1, T.float32) mixes = T.alloc_fragment(hc_mult3, T.float32) T.clear(mixes) rms[0] = 0 # ... 主循环计算 rms 和 mixes，与 mhc_pre_big_fuse_tilelang 相同 ... # 但增加了对 layer_input 的 sum_sq 累积（通过 pipelined 方式） # 最后输出带 Norm 的 hidden_states 请注意：由于代码较长，此处仅展示函数签名和核心意图。完整实现包含两层 pipelined 循环以同时计算 layer_input 的 sum_sq 和最终归一化写回。

评论区精华

该 PR 没有公开的 review 讨论；作者多次使用 /rerun-stage 和 /rerun-test 命令触发 DSV4 专用 CI（stage-c-test-dsv4-4-gpu-b200, stage-c-test-dsv4-8-gpu-h200），最终所有 DSV4 相关测试通过，确认功能正确性。

- CI 测试验证 (other): 确认功能正确性

风险与影响

- 风险： 1) 新 Triton/TileLang 内核（fused_hc_head、mhc_pre_big_fuse_with_norm_tilelang）缺少独立的单元测试，但 torch fallback 路径保留，降低风险； 2) _compute_num_split_for_mhc_pre 依赖 torch.cuda.get_device_properties(0).multi_processor_count，在 MIG、虚拟化或不对称 GPU 环境下可能返回不合理值，导致性能退化； 3) mhc_pre_big_fuse_with_norm_tilelang 使用 TL_PTXAS_REGISTER_USAGE_LEVEL: 10，可能降低 warp 占有量或引发寄存器溢出； 4) hc_head 现在默认使用新 Triton kernel，虽然 torch 实现仍保留（如果 fusion 路径异常可回退），但可能隐含精度差异（在非 DSV4 场景未经测试）。
  - 影响：对 DeepSeek-V4 模型推理延迟有显著降低（microbench 上 1.3-3.6x），但端到端收益因 workload 而异；所有优化通过环境变量（SGLANG_OPT_USE_TILELANG_MHC_PRE, SGLANG_OPT_DEEPGEMM_HC_PRENORM）控制或默认启用，不影响非 DSV4 模型；团队需要维护新增的 Triton/TileLang 内核代码，增加长期维护成本。
  - 风险标记：新内核缺少测试覆盖，依赖 GPU SM 计数可能环境不兼容，寄存器压力可能降低 Occupancy，hc_head 默认走新 Triton 路径

关联脉络

- PR #24793 [DSV4] Cherry pick missing commits from deepseek_v4 branch and enhance tests: 同一功能线（DeepSeek-V4 优化），该 PR 增强了 DSV4 测试，本 PR 则优化了 DSV4 核心性能，二者配合确保优化后的功能正确性。