PR #51674 完整报告

vllm-project/vllm

[Kernel][Perf] Add fused CUDA post-conv MTP decode kernel for Qwen3.5 GDN

合并时间：2026-08-14 11:44

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/51674

执行摘要

- 一句话：融合 GDN MTP decode 内核，解码延迟降低 1.2x-2.2x
- 推荐动作：值得精读，尤其适合关注投机解码与 attention kernel 的工程师。值得关注的设计决策包括：单 kernel 融合整条 decode 链路的取舍（含放弃 mixed-batch 融合）、基于 num_accepted_tokens 的 state rewind 语义、约束不满足时默认回退 vs 显式 raise 的配置策略、以及把新 custom op 显式登记为 piecewise CUDA graph split boundary 的编译集成。review 中关于 CUDA graph padding 与 torch.zeros 的讨论也很有价值。可结合 csrc/libtorch_stable/gdn/fused_gdn_decode_kernel.cu 与 tests/kernels/test_fused_gdn_post_conv.py 对照阅读。

功能与动机

PR body 明确指出性能瓶颈：During MTP decode, the Triton path launches a chain of small kernels per step (gating, delta-rule recurrence, state rewind/update, gated RMSNorm), which leaves the GPU latency-bound at decode batch sizes。在 Blackwell 上 MTP 投机解码每步串行启动多个小 kernel，GPU 处于延迟受限状态。微基准验证融合后冷缓存加速 1.17x-2.20x，且 BS=4-32 小 batch 区间收益最大，正是投机解码实际运行的延迟敏感区间。

实现拆解

实现按以下 5 个步骤拆解：

1. 新增融合 CUDA 内核：csrc/libtorch_stable/gdn/fused_gdn_decode_kernel.cu 新增 gdn_decode_post_conv_mtp_kernel 模板内核，支持 BF16/FP32 两种 recurrent state。每个 block 负责一个请求的一个 value head，基于 num_accepted_tokens 回退到最后一个被接受的 state slot，再对最多 8 个 draft token 顺序执行 gated delta-rule recurrence，最后一并完成 gated RMSNorm epilogue；kernel 内部用 cp.async 双缓冲加载 state、warp shuffle 做归约。该内核只依赖 cp.async 和 BF16 数学，因此从最初仅限 SM100 放宽到 SM80+，CMake 侧拆出独立的 FUSED_GDN_DECODE_ARCHS/FUSED_GDN_DECODE_SRC/VLLM_ENABLE_FUSED_GDN_DECODE（替换原先挂在 FUSED_KDA_DECODE_* 下的写法）。
2. op 声明与绑定：csrc/libtorch_stable/ops.h 声明 fused_gdn_decode_post_conv_mtp，csrc/libtorch_stable/torch_bindings.cpp 在 VLLM_ENABLE_FUSED_GDN_DECODE 下完成 def 与 impl，vllm/_custom_ops.py 增加同名 Python 包装

fused_gdn_decode_post_conv_mtp，默认 scale=128**-0.5、norm_eps=1e-5。

3. 模型层 dispatch 与约束校验：vllm/model_executor/layers/mamba/gdn/qwen_gdn_linear_attn.py 是核心改动（+303/-4）。__init__ 读取 VLLM_GDN_DECODE_KERNEL（默认 cuda），新增 _fused_gdn_decode_unsupported_reason 集中校验约束（非 interleaved GQA、K=V=128、SiLU gating、BF16 模型、BF16 conv state、BF16/FP32 recurrent state、CC8.0+、op 已编译）；显式设置 cuda 不满足约束时 raise，默认值则回退 Triton 并 logger.info_once。forward_cuda 增加 use_fused_gdn_decode 分支，直接走 packed custom op qwen_gdn_attention_core_fused_norm_packed（不再 split q/k/v/z/b/a）。spec-decode 路径新增 _forward_core_decode_spec_fused_norm（先 causal_conv1d_update 带 rewind）和 _forward_core_decode_spec_post_conv_fused_norm（调 fused kernel），_can_use_fused_gdn_mtp_decode 判定纯 spec-decode、num_v_heads == 8 * num_k_heads、MTP tokens 数不超过 8 等条件，不满足时整体回退 Triton。
4. 编译集成与 CUDA graph 边界：vllm/config/compilation.py 把 vllm::qwen_gdn_attention_core_fused_norm_packed 加入 CompilationConfig._attention_ops，使其成为 piecewise CUDA graph 的 split boundary——否则 torch.compile 会尝试 trace 该 op 的运行时 dispatch 与 state 原地更新，破坏 CUDA graph 捕获。
5. 测试与配套：新增 tests/kernels/mamba/test_gdn_fused_mtp.py 覆盖模型路径级 dispatch（pure MTP 走 fused、mixed/prefill/decode 回退）并与 Triton reference 逐层对比；tests/kernels/test_fused_gdn_post_conv.py 扩展 test_fused_gdn_decode_post_conv_mtp_ratio8 做 kernel 级数值等价（相对 L2 < 5e-4、state atol/rtol 3e-2），覆盖 TP4/TP16、BF16/FP32 state、ragged acceptance 模式；tests/test_envs.py 新增 test_gdn_decode_kernel_env 校验 env 取值合法性；tests/compile/test_config.py 新增断言确保新 op 默认出现在 splitting_ops 中。

演进取舍：commit d78b330b 主动放弃 mixed-batch 融合路径（需要 spec_token_prefix_len metadata 和 prefix-slicing，收益小），commit 38a1c179 修复了 rebase 冲突导致 a_spec/b_spec gating 被覆盖的回归，commit c33dbbaa 将 env 改为 VLLM_GDN_DECODE_KERNEL={cuda,triton} 并默认 cuda。

关键文件：

- vllm/model_executor/layers/mamba/gdn/qwen_gdn_linear_attn.py（模块 线性注意力；类别 source；类型 data-contract；符号 _fused_gdn_decode_unsupported_reason, _forward_core_decode_spec_fused_norm, _forward_core_decode_spec_post_conv_fused_norm, _forward_core_fused_norm_packed）：核心调度与 dispatch 逻辑所在地：新增 fused 路径的约束校验、forward_cuda 分支、spec-decode fused norm 入口和 _can_use_fused_gdn_mtp_decode 判定，是整个 PR 的功能开关中枢。
- csrc/libtorch_stable/gdn/fused_gdn_decode_kernel.cu（模块 融合内核；类别 source；类型 core-logic；符号 gdn_decode_post_conv_mtp_kernel, copy_state_chunk, warp_reduce_sum_pair）：新增的融合 CUDA 内核本体（557 行），是整个 PR 的性能来源：在单次 kernel launch 内完成 gating、delta-rule recurrence、state 回退 / 更新与 gated RMSNorm，支持 BF16/FP32 state 和 SM80+。

- vllm/_custom_ops.py（模块 自定义算子；类别 source；类型 core-logic；符号 fused_gdn_decode_post_conv_mtp）：新增 fused_gdn_decode_post_conv_mtp Python 包装，是模型层调用 CUDA op 的唯一入口，统一参数顺序与默认值。
- vllm/envs.py（模块 环境配置；类别 source；类型 configuration；符号 VLLM_GDN_DECODE_KERNEL）：新增 VLLM_GDN_DECODE_KERNEL 环境变量（默认 cuda），决定 GDN MTP decode 走融合 CUDA 内核还是 Triton 路径，是用户侧唯一开关。
- csrc/libtorch_stable/ops.h（模块 算子声明；类别 source；类型 core-logic；符号 fused_gdn_decode_post_conv_mtp）：声明新的 CUDA op 接口，与 torch_bindings.cpp 共同完成 C++ 层注册。
- csrc/libtorch_stable/torch_bindings.cpp（模块 算子绑定；类别 source；类型 core-logic）：在 VLLM_ENABLE_FUSED_GDN_DECODE 宏下完成 op 的定义与 CUDA 实现绑定。
- vllm/config/compilation.py（模块 编译配置；类别 source；类型 core-logic）：把 vllm::qwen_gdn_attention_core_fused_norm_packed 加入 attention ops 列表，保证 piecewise CUDA graph 正确拆分、不被 torch.compile trace。
- tests/kernels/mamba/test_gdn_fused_mtp.py（模块 模型测试；类别 test；类型 test-coverage；符号 _TestGatedNorm, _make_vllm_config, _build_layer, test_fused_forward_uses_packed_entrypoint）：新增的模型路径级测试：验证纯 MTP 走 fused 入口、mixed/prefill/decode 回退 Triton，并逐层对比参考实现，是功能正确性的主要保障。
- tests/kernels/test_fused_gdn_post_conv.py（模块 内核测试；类别 test；类型 test-coverage；符号 test_fused_gdn_decode_post_conv_mtp_ratio8）：扩展 kernel 级数值等价测试 test_fused_gdn_decode_post_conv_mtp_ratio8，覆盖 TP4/TP16、BF16/FP32 state、ragged acceptance 模式，是对照 Triton reference 的核心保障。
- tests/test_envs.py（模块 环境测试；类别 test；类型 test-coverage；符号 test_gdn_decode_kernel_env）：校验 VLLM_GDN_DECODE_KERNEL 的默认值、合法取值与非法取值报错。
- tests/compile/test_config.py（模块 编译测试；类别 test；类型 test-coverage）：新增断言确保新 custom op 默认出现在 splitting_ops 中，防止未来重构破坏 piecewise CUDA graph 的拆分边界。
- CMakeLists.txt（模块 构建脚本；类别 infra；类型 configuration）：构建配置：为 fused GDN decode kernel 拆分独立的 arch 列表与编译宏，避免与 Kimi-K3 KDA 内核的 arch 集合耦合。

关键符号：fused_gdn_decode_post_conv_mtp, _fused_gdn_decode_unsupported_reason, _forward_core_decode_spec_fused_norm, _forward_core_decode_spec_post_conv_fused_norm, _forward_core_fused_norm_packed, _can_use_fused_gdn_mtp_decode, _rms_norm_gated_cuda, test_fused_gdn_decode_post_conv_mtp_ratio8, test_fused_model_path_matches_reference

关键源码片段

vllm/_custom_ops.py

新增 fused_gdn_decode_post_conv_mtp Python 包装，是模型层调用 CUDA op 的唯一入口，统一参数顺序与默认值。

```
# SPDX-License-Identifier: Apache-2.0
# vllm/_custom_ops.py 中新增的 Python 侧包装。
# 统一 fused GDN MTP decode 的入参顺序，并给 out/scale/norm_eps 提供默认值，
# 供 qwen_gdn_linear_attn.py 的 spec-decode fused 路径直接调用。

def fused_gdn_decode_post_conv_mtp(
    mixed_qkv: torch.Tensor,
    a: torch.Tensor,
    b: torch.Tensor,
    A_log: torch.Tensor,
    dt_bias: torch.Tensor,
    state_indices: torch.Tensor,
    cu_seqlens: torch.Tensor,
    num_accepted_tokens: torch.Tensor,
    state: torch.Tensor, # recurrent state，kernel 内原地更新
    output_gate: torch.Tensor, # gated RMSNorm 的 gate 输入
    norm_weight: torch.Tensor,
    out: torch.Tensor | None = None,
    scale: float = 128**-0.5,
    norm_eps: float = 1e-5,
) -> torch.Tensor:
    if out is None:
        out = torch.empty_like(output_gate)
    torch.ops._C.fused_gdn_decode_post_conv_mtp(
        mixed_qkv, a, b, A_log, dt_bias, state_indices, cu_seqlens,
        num_accepted_tokens, state, output_gate, norm_weight, out,
        scale, norm_eps,
    )
    return out
```

评论区精华

review 中核心讨论集中在以下几个点：

- 默认启用策略：gau-nernst 问 Why can't we enable it by default?，作者回应只测过 gsm8k 和随机数据、未在多样数据集验证；gau-nernst 随后认为只要约束满足就可以默认启用，ZJY0516 表示 I think it's okay if it's always faster，并建议命名用 cuda 而非 fused。最终收敛为 VLLM_GDN_DECODE_KERNEL 默认 cuda。
- torch.zeros 是否有必要：gau-nernst 质疑 fused 分支里 torch.zeros() 是否可换成 torch.empty()，作者解释了 CUDA graph 下 batch 有 padding、kernel 只写 unpadded 行、torch.empty 会把垃圾值流入 out_proj，且与非 fused 分配保持一致（关联 #28182）；gau-nernst 后转述 ZJY0516 的说法称该问题在 Kimi-K3 KDA 层已修，但决定保留现状留给未来 PR。
- compute capability 约束：gau-nernst 指出 kernel 未使用 SM100 专属特性、应兼容 SM80+，作者据此放宽到 compute capability 8.0+，并拆分 CMake arch 列表。

- test_config.py 的回归守卫：gau-nernst 问新增断言的作用，作者详细解释 qwen_gdn_attention_core_fused_norm_packed 必须留在 splitting_ops 中作为 piecewise CUDA graph 的 split boundary，否则重构遗漏时测试仍会通过但 CUDA graph 捕获被破坏。
- helper 内联与命名清理：gau-nernst 建议把只有单一用户的 gdn_decode_state_utils.cuh 内联进 kernel 文件，作者已处理；CMake 命名 nit 也随 arch 拆分一并解决。
  - fused kernel 的 compute capability 约束 (question): 作者确认后放宽为 compute capability 8.0+，并拆分独立 arch 列表（8.0;8.6;8.9;9.0a;10.0f;12.0f）与 VLLM_ENABLE_FUSED_GDN_DECODE 宏。
  - 环境变量默认启用策略与命名 (design): 最终收敛为 VLLM_GDN_DECODE_KERNEL={cuda,triton}，默认 cuda；显式设置 cuda 但约束不满足时 raise，默认值则回退 Triton 并记录日志。
  - torch.zeros 是否有必要（CUDA graph padding 安全性） (correctness): 保持 torch.zeros，与非 fused 分配保持一致，待 #28182 有安全方案后统一切换。
  - 新增 custom op 的 piecewise CUDA graph 回归守卫 (testing): 断言保留在 test_splitting_ops_dynamic 中，作为默认配置下 splitting_ops 的回归保护。
  - gdn_decode_state_utils.cuh 是否内联 (design): 作者按 review 意见内联（commit 4b58b455）。
  - CMake 命名与 arch 拆分 (style): 拆分完成，命名各归各内核。

风险与影响

- 风险：主要风险点：

1. 默认路径变更：VLLM_GDN_DECODE_KERNEL 默认 cuda，虽有不满足约束自动回退 Triton的保护（qwen_gdn_linear_attn.py的_fused_gdn_decode_unsupported_reason），但任何约束校验遗漏都会静默或显式改变decode行为，需要关注日志中的fallback提示。
2. 真实模型 e2e 验证缺失：PR 作者明确说明未跑 lm_eval，因为公开 checkpoint（0.8B 16/16、35B 16/32、397B 16/64）都不满足 num_v_heads == 8 * num_k_heads 的 head 布局约束，fused 路径只在 kernel 级和模型路径级做了数值等价测试；未来一旦出现满足约束的模型，新路径会默认生效而缺少端到端质量背书。
3. rebase/merge 正确性回归：最后一笔 commit 38a1c179 修复了 rebase 冲突导致的 a_spec/b_spec gating 回归，说明该路径在持续 rebase 中容易被上游改动静默破坏，需要有更强粒度的测试守住（当前 test_gdn_fused_mtp.py 覆盖了 mixed fallback，但未覆盖 spec gating 的数值回归）。
4. 混合 batch 性能分叉：fused 路径只覆盖纯 spec-decode batch，混合 batch 回退 Triton，线上 batch 构成复杂时收益会打折，且两个路径的输出精度仅靠 5e-4 相对 L2 阈值保证。
5. torch.zeros 依赖：fused 分支依赖 padded tail 清零来保证 CUDA graph 下输出正确，与 #28182 的演进存在耦合，后续若改用 torch.empty 需要同步验证两个调用点。 - 影响：对用户：满足约束（非 interleaved GQA、K=V=128、BF16 模型、CC8.0+ 等）的 Qwen3.5 GDN MTP 部署会自动获得 1.17x-2.20x 的 decode 加速，且可通过 VLLM_GDN_DECODE_KERNEL=triton 回退；现有公开 checkpoint 因 head 布局约束不

会触发新路径，默认行为不变，因此对存量用户影响低。对系统：新增一个约 557 行的 CUDA kernel 和新的 custom op，增加构建面（SM80+ 各 arch）与维护成本，但只在 GDN 层启用；CompilationConfig._attention_ops 增加一项，影响 piecewise CUDA graph 的拆分行为。对团队：需要在 GDN decode 这一条功能线上持续维护 Triton 与 CUDA 双路径的一致性，并保持 dispatch 约束与 CMake 开关的同步。 - 风险标记：默认启用新解码路径，缺少真实模型 e2e 验证，rebase 冲突曾致正确性回归，混合 batch 回退性能分叉，CUDA graph padding 依赖 zero 初始化

关联脉络

- PR #50062 [Model Runner V2][Spec Decode] Add KV cache support for multi-layer MTP: 同属 MTP 投机解码功能线，本 PR 依赖其引入的 spec_state_indices_tensor、num_accepted_tokens 等 attention metadata 语义来完成 state rewind。
- PR #52030 [Bugfix] Fix packed GDN decode launch for large batch-head grids: 同属 GDN decode 内核线，说明该路径在持续演进，两个 PR 都对 GDN 解码延迟与 kernel 启动边界敏感。
- PR #50685 [Bugfix][Refactor] Keep Qwen3Next layer boundaries sequence parallel: 同文件家族（qwen_gdn_linear_attn.py 与 Qwen3Next 相关模型）的 EP+SP 布局契约变更，与 fused kernel 对 head 布局（num_v_heads == 8 * num_k_heads）和 TP 切分的约束存在交互。
- PR #48666 [Kernel] Gemma-4 FA4 FP8 Kernel: 同为 speculative decoding 场景下的内核融合与精度对齐工作，可对照其在 flash_attn 后端融合 MTP 相关 scale 修复的做法。