PR #30105 完整报告

sgl-project/sglang

[AMD][Spec] Fix aiter GQA packing + split-KV routing in NEXTN spec attention (verify & draft_extend)

合并时间：2026-08-23 05:22

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30105

执行摘要

- 一句话：修复 aiter 后端 spec 注意力两处内核路由，长上下文提速达 1.76×
- 推荐动作：值得精读。重点关注三点：1）通过内核 trace 签名（num_query_heads_16 / num_queries_per_kv_1）定位 GQA 摊还丢失的方法论；2）verify 与 draft_extend 统一走 unified_attention 的设计——同一个内核同时提供 GQA 打包与 split-KV，避免为 spec 阶段维护两套 kernel 路由；3）在 kernel 无法进 CI 的场景下，用“bit-identical + accept A/B + 端到端准确率”组合构建精度证据链。建议后续将 test/manual 的 flag 守卫测试迁入注册 CI，并考虑用可移植的契约测试（mock unified_attention 参数）覆盖路由分支。

功能与动机

在 ROCm/gfx950 上使用 aiter 注意力后端时，NEXTN 投机解码（EAGLE）在长上下文（约 40K 之后）反而输给非投机基线，且随上下文增长持续劣化——即使 accept_length 保持健康（2.4-3.4）。torch trace 显示 GPU 利用率 95-99% 无调度气泡，唯一随上下文线性恶化的就是 attention。根因是两个独立的内核路由病态：target_verify 的 GQA 摊还丢失（.expand() 导致 num_queries_per_kv=1，退化为全 MHA 平铺）与 draft_extend 的 occupancy 饥饿（落入无 split-KV 的 mha_batch_prefill_func，短 Q 时仅约 0.2% HBM 带宽）。修复目标就是让 spec 在长上下文重新赢回 non-spec。

实现拆解

1. 变更入口：python/sglang/srt/layers/attention/aiter_backend.py 的 forward_extend 是 aiter 后端 extend 形态注意力的统一入口，target_verify 与 draft_extend_v2 两个 spec 阶段都经过它；两个修复均在该方法内以条件分支落点，其他后端路径（NVIDIA/ 非 HIP）完全不受影响。
2. 修复 1：verify 恢复 GQA 打包：删除 target_verify 分支中 K/V 的 stride-0 .expand()。旧代码为了让 unified_attention 匹配 Qwen3.5 的 GQA 32:2 头部映射，把本地单个 KV 头在 head 维 expand 成 tp_q_head_num，导致内核推导 num_queries_per_kv = 1 并按全 MHA 平铺；删除后按 forward_decode 同样的方式传入真实 KV 头数，内核推出 GQA group = 16。该改动只改变 tiling 不改变数学，输出与 accept_length 均不变。
3. 修复 2：draft_extend 改道 unified_attention：新增 forward_extend 分支，条件为 _use_unified_verify、is_draft_extend_v2() 与 SGLANG_AITER_UNIFIED_DRAFT_EXTEND 三者同时满足。路径内通过

_build_unified_page_table_from_spec 从 spec 元数据构建 2D block_table（含滑动窗口回退），用 kv_indptr 差分计算 seqused_k，并处理 qk_head_dim != v_head_dim 时的输出张量形状；unified_attention 的 split-KV（num_segments）让短 Q 场景恢复 occupancy。

4. 开关与测试配套：python/sglang/srt/environ.py 注册 SGLANG_AITER_UNIFIED_DRAFT_EXTEND = EnvBool(True)，与 SGLANG_AITER_UNIFIED_VERIFY 相互独立，可单独 A/B 或禁用回退；aiter_backend.py 同步增加 from sglang.srt.environ import envs。测试 test/manual/test_aiter_unified_draft_extend_env.py 断言 flag 注册、默认开启与 override 上下文切换恢复；注意该文件位于 test/manual，未注册进 CI，kernel 路径依赖 ROCm/gfx950，由 PR 内的精度与性能数据人工覆盖。

关键文件：

- python/sglang/srt/layers/attention/aiter_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 forward_extend, _build_unified_page_table_from_spec, _use_unified_verify）：核心改动所在：target_verify 删除 stride-0 .expand() 恢复 GQA 打包；draft_extend_v2 新增 unified_attention 路由分支，消除长上下文 verify 42× 延迟与 draft 路径约 400× 带宽浪费。
- python/sglang/srt/environ.py（模块 环境配置；类别 source；类型 configuration；符号 SGLANG_AITER_UNIFIED_DRAFT_EXTEND）：注册 SGLANG_AITER_UNIFIED_DRAFT_EXTEND（EnvBool 默认 True），作为独立 kill-switch，与 SGLANG_AITER_UNIFIED_VERIFY 解耦，便于对修复 2 单独 A/B 或禁用。
- test/manual/test_aiter_unified_draft_extend_env.py（模块 单元测试；类别 test；类型 test-coverage；符号 TestAiterUnifiedDraftExtendEnv, setUp, test_registered_and_default_on, test_override_toggles_and_restores）：守卫新 env flag 的注册与默认值，防止改名或默认翻转；kernel 路径需要 gfx950 无法进入通用 CI，此测试是当前唯一的自动化配套。

关键符号：forward_extend, _build_unified_page_table_from_spec, TestAiterUnifiedDraftExtendEnv

关键源码片段

python/sglang/srt/layers/attention/aiter_backend.py

核心改动所在：target_verify 删除 stride-0 .expand() 恢复 GQA 打包；draft_extend_v2 新增 unified_attention 路由分支，消除长上下文 verify 42× 延迟与 draft 路径约 400× 带宽浪费。

```
# aiter_backend.py —— forward_extend 内的 NEXTN spec 注意力两处修复
# 目标：让 target_verify 与 draft_extend_v2 共用 unified_attention，
# 同时获得 GQA 打包与 split-KV。

# —— 修复 1：target_verify 恢复 GQA 打包 ——
# 旧代码把单 KV 头在 head 维做 stride-0 .expand() 到 tp_q_head_num，
# 使 unified_attention 认为 num_kv_heads == tp_q_head_num，
```

```
# 推出 num_queries_per_kv = 1，退化为全 MHA 平铺（KV 流量放大约 7×）。
# 现在按 forward_decode 的方式传入真实 KV 头数，内核自动推导
# num_queries_per_kv = tp_q_head_num（GQA group），单次 KV 加载服务全部 Q 头。
if self._use_unified_verify and forward_batch.forward_mode.is_target_verify():
    k_cache, v_cache = self.token_to_kv_pool.get_kv_buffer(layer.layer_id)
    page_table = self.forward_metadata.kv_indices
    max_kv_len = page_table.shape[1] * self.page_size
    window_size = (-1, -1)
    if layer.sliding_window_size is not None and layer.sliding_window_size > -1:
        window_size = (layer.sliding_window_size - 1, 0)
        if self.forward_metadata.swa_page_table is not None:
            page_table = self.forward_metadata.swa_page_table

    q_unified = q.view(-1, layer.tp_q_head_num, layer.qk_head_dim)
    k_unified = k_cache.view(-1, self.page_size, layer.tp_k_head_num, layer.qk_head_dim)
    v_unified = v_cache.view(-1, self.page_size, layer.tp_v_head_num, layer.v_head_dim)
    # seq_lens 与 draft_num 的加法必须留在 CUDA graph 区域内执行，
    # 主机侧预加会分配新张量，破坏每次 replay 已捕获的指针。
    unified_attention(
        q=q_unified, k=k_unified, v=v_unified,
        out=o.view(-1, layer.tp_q_head_num, layer.v_head_dim),
        cu_seqlens_q=self.forward_metadata.qo_indptr,
        seqused_k=forward_batch.seq_lens + self.forward_metadata.max_q_len,
        max_seqlen_q=self.forward_metadata.max_q_len,
        max_seqlen_k=max_kv_len,
        softmax_scale=layer.scaling, causal=True,
        window_size=window_size, block_table=page_table,
        softcap=layer.logit_cap, k_descale=k_descale,
        v_descale=v_descale, sinks=sinks,
    )
    return o.view(-1, layer.tp_q_head_num * layer.v_head_dim)

# —— 修复 2：draft_extend_v2 改道 unified_attention ——
# EAGLE-v2 KV catch-up 形态是"短 Q 对长分页 KV"，旧路径落到
# mha_batch_prefill_func（prefill FMHA，无 split-KV），短 Q 时仅约 1 个
# Q tile，HBM 带宽利用约 0.2%（约 400× 高于内存地板）。
# 改道后与 target_verify 共用统一内核，获得 split-KV 恢复 occupancy。
if (
    self._use_unified_verify
    and forward_batch.forward_mode.is_draft_extend_v2()
    and envs.SGLANG_AITER_UNIFIED_DRAFT_EXTEND.get()
):
    bs = forward_batch.batch_size
    if layer.qk_head_dim != layer.v_head_dim:
        o = q.new_empty((q.shape[0], layer.tp_q_head_num * layer.v_head_dim))
    else:
        o = torch.empty_like(q)
    k_cache, v_cache = self.token_to_kv_pool.get_kv_buffer(layer.layer_id)
    # 从 spec 元数据构建 2D block_table，必要时回退到滑动窗口 page table
```

```
    page_table, swa_page_table = self._build_unified_page_table_from_spec(
        self.forward_metadata, bs
    )
    pt = page_table
    de_window = (-1, -1)
    if layer.sliding_window_size is not None and layer.sliding_window_size > -1:
        de_window = (layer.sliding_window_size - 1, 0)
        if swa_page_table is not None:
            pt = swa_page_table
    kv_indptr = self.forward_metadata.kv_indptr
    # 每条序列的实际 KV 长度由 kv_indptr 差分得到，不再依赖扩展后的 seq_lens
    seqused_k = (kv_indptr[1 : bs + 1] - kv_indptr[:bs]).to(torch.int32)
    unified_attention(
        q=q.view(-1, layer.tp_q_head_num, layer.qk_head_dim),
        k=k_cache.view(-1, self.page_size, layer.tp_k_head_num, layer.qk_head_dim),
        v=v_cache.view(-1, self.page_size, layer.tp_v_head_num, layer.v_head_dim),
        out=o.view(-1, layer.tp_q_head_num, layer.v_head_dim),
        cu_seqlens_q=self.qo_indptr[: bs + 1],
        seqused_k=seqused_k,
        max_seqlen_q=self.forward_metadata.max_q_len,
        max_seqlen_k=pt.shape[1] * self.page_size,
        softmax_scale=layer.scaling, causal=True,
        window_size=de_window, block_table=pt,
        softcap=layer.logit_cap, k_descale=k_descale,
        v_descale=v_descale, sinks=sinks,
    )
    return o.view(-1, layer.tp_q_head_num * layer.v_head_dim)
```

python/sglang/srt/environ.py

注册 SGLANG_AITER_UNIFIED_DRAFT_EXTEND（EnvBool 默认 True），作为独立 kill-switch，与 SGLANG_AITER_UNIFIED_VERIFY 解耦，便于对修复 2 单独 A/B 或禁用。

```
# environ.py —— Envs 类内的新开关注册
# aiter 后端（ROCm）：把 NEXTN spec 的 draft_extend（EAGLE-v2 KV
# catch-up）从无 split-KV 的 mha_batch_prefill_func 改道 unified_attention
# （GQA 打包 + split-KV），作为独立的 kill-switch，与
# SGLANG_AITER_UNIFIED_VERIFY 解耦，可单独 A/B 或禁用。默认开启。
SGLANG_AITER_UNIFIED_DRAFT_EXTEND = EnvBool(True)
```

评论区精华

- 关于 draft_extend 元数据开销：1am9trash 提问："In the forward_extend() function, it builds the page table via _build_unified_page_table_from_spec() and recomputes seqused_k. Would this additional overhead cause any regression at short ctx, where attention compute is small?" hsthe29 用 MI350X 实测回应：eager 模式累计约 23.4-23.9 us，但 graph replay 下真实增量仅 7.3-11.5 us；并给出 batch 1/8/32 在 ctx 128/512/2048 的对比表——ctx 128 时最差慢 41.4%（batch 1），ctx 512 起即反超（快 39.5%），长上下文快 60-80%。1am9trash 回复 "It's super clear."

- 关于 CI 覆盖缺口：amd-bot 警告 “The changed code is not exercised by PR CI... the only test added — test/manual/test_aiter_unified_draft_extend_env.py — lives under test/manual/ (not registered in CI, never runs) and only asserts the env flag registers/defaults”。hsthe29 回应：真正执行该路径的 AMD MI35x 作业（stage-c-test-large-8-gpu-amd-mi35x 两个 shard、stage-b-test-1-gpu-small-amd-mi35x）全部通过；其余红色作业为 runner 基础设施问题（NPU/XPU 缺 tvm_ffi）、main 上预存在失败（test_deepseek_v32_basic.py 的 ninja JIT 错误、VLM 模型在 ROCm 上要求 NVIDIA CUDA）或与本改动无关的 PD KV 传输超时。
- 合入判定：1am9trash 与 HaiShaw 最终 APPROVED，1am9trash 明确 “No effect on non-hip code path”，并在等待 gb300 作业状态确认后完成合入。
- draft_extend 元数据开销是否造成短上下文回归 (performance): 开销集中在短 Q 小上下文且有明确 crossover；graph replay 下增量可忽略，接受该改道。
- PR CI 未执行变更的 kernel 路径（测试覆盖缺口） (testing): 双方一致认为需要人工 benchmark 与精度证据补充；PR 内提供了 conc-1 logit-diff 为 0、accept A/B 无回归、gsm8k 0.971 等证据支撑合入。
- 合入前的 CI 等待与 gb300 作业状态 (other): AMD bot 与维护者介入评估，最终由 1am9trash 与 HaiShaw 批准合入。

风险与影响

- 风险：
  - 短上下文回归：draft_extend 每次调用新增 page table 构建与 seqused_k 重算，graph replay 下增量 7.3-11.5 us；在 ctx 128、batch 1 场景整次 forward_extend 慢 41.4%，存在小幅短上下文开销，但 ctx 512 起即反超。由于 flag 默认开启，极短上下文小批量场景可能观察到微量 TPOT 上升。
  - 正确性验证局限：verify 修复不改数学（仅 tiling），draft_extend 改道在 conc=1 下与旧 FMHA 结果 bit-identical；但 batch>1 时 temp-0 解码在 ROCm 栈上有运行间不确定性（FP 归约顺序变化导致 argmax 翻转），无法用输出 hash 做精确 A/B，依赖 accept 对比与 gsm8k 端到端结果。
  - 测试覆盖缺口：新 kernel 路径只在 ROCm/gfx950 + NEXTN 组合下运行，PR CI 没有任何作业实际执行新分支；test/manual 测试只守卫 env flag，后续对 forward_extend 或 _build_unified_page_table_from_spec 的重构可能无声破坏该路径。
  - 交互与前置条件：draft_extend 改道依赖 _use_unified_verify，若 verify flag 关闭则两个修复同时失效；此外 PR 提到 mamba radix cache eviction 在并发不同 prompt 下有预存在崩溃，需 --disable-radix-cache 规避，与本改动无关但影响复现与验收环境。
- 影响：
  - 性能影响：MI350X/gfx950 上 NEXTN spec 从“40K 后输给 non-spec”变为 1.4K-88K 全范围 1.35-1.76× 胜出；88K 上下文 TPOT 从 13.8 ms 降到 6.8 ms；高并发（conc 4-16）下 MXFP4/FP8 两种量化均保持 1.34-1.58× 每 token 加速。核心里程碑：verify attn 272→104 ms，draft_extend attn 约 773→约 5 ms。
  - 用户与平台：影响仅限 AMD ROCm 上启用 aiter 后端 + NEXTN/EAGLE 投机解码的用户；非 HIP 路径零影响（reviewer 明确确认）。新路径默认开启，现有部署无需改配置

即可获益，也保留 env flag 独立回退。

- 团队协作：PR 以两个 commit 拆分 verify 与 draft_extend 修复，便于 bisect；10 个 commit 中多数为 main 合并，体现长周期合入。未来需要把 kernel 路径测试纳入可注册 CI，或建立契约测试防止路由逻辑漂移。
- 风险标记：核心路径变更，缺少 CI 测试覆盖，默认开启新路径，短上下文开销

关联脉络

- PR #31675 （讨论中引用的基线修复 PR，标题未提供）：PR 讨论与 CI 排查中明确提及：早期 MI35x 形状不匹配失败是 stale-base 问题，由 #31675 修复并合入本分支后相关 AMD 作业转绿。
- PR #36004 [AMD][DSV4] perf: use full 1024-thread block for indexer top-k on ROCm: 同为 AMD ROCm 内核性能优化（DSv4 top-k 用满 1024 线程块，解码吞吐提升约 3%），与本 PR 同属 AMD 性能修复演进线，但文件不重叠。
- PR #34490 [AMD] Add Radix-4 MoE top-k router kernel for Kimi-K3 routing: 同为 AMD ROCm 内核性能投入（Kimi-K3 radix-4 top-k 路由内核），展示 AMD 侧持续的性能内核工作，与本 PR 无文件交集。