PR #31221 完整报告

sgl-project/sglang

[AMD] Derive AITER verify tokens-per-req from input shape

合并时间：2026-08-01 15:29

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/31221

执行摘要

- 一句话：AITER verify 长度改由输入形状推导，修复非 MLA 验证路径
- 推荐动作：建议 AMD 平台 spec decoding 相关同学精读。值得关注的设计决策：以 use_mla 为分界保留固定 draft length 的例外处理，兼顾 AITER 内核约束与动态验证场景；以及 max_q_len 从固定值改为动态值后对 forward_extend 中 seqused_k 的连带修改。另可关注后续是否补齐非 MLA 动态 verify 的 PR CI 覆盖。

功能与动机

PR body 明确说明：在 AITER attention 后端的 target_verify 和 CUDA graph 路径中，per-request verify token 数量取自固定的 init 时 num_draft_tokens（self.num_draft_tokens / spec_info.draft_token_num）。当每个请求的实际 verify token 数在运行时变化时（例如 SBD spec decoding），该取值不正确，导致 qo_indptr 等元数据与真实输入长度不匹配。因此需要从精确大小的捕获 / 回放输入推导 verify tokens per request。

实现拆解

本 PR 是对 AITER 注意力后端 CUDA 图 target_verify 元数据构造逻辑的定点修复，涉及 python/sglang/srt/layers/attention/aiter_backend.py 一个文件：

1. 入口计算动态 verify token 数：在 init_forward_metadata_out_graph 中新增 verify_tokens_per_req 计算，仅在 forward_mode.is_target_verify() 时取 forward_batch.input_ids.shape[0] // forward_batch.batch_size，否则为 None，并作为新参数传入 _apply_cuda_graph_metadata。
2. 非 MLA 分支从输入推导 draft_num：init_forward_metadata 的非统一 verify 分支中，draft_num = forward_batch.input_ids.shape[0] // bs 替换原先的 spec_info.draft_token_num，随后才重新赋值 bs = len(forward_batch.req_pool_indices)，保证 qo_indptr 等缓冲区按真实验证券数量构建。
3. target_verify 分支引入 tokens_per_req：_apply_cuda_graph_metadata 中新增 verify_tokens_per_req 参数并加 assert；在 is_target_verify() 分支内，tokens_per_req = self.num_draft_tokens if self.use_mla else verify_tokens_per_req。MLA 路径保持固定 draft 长度（AITER MLA 内核要求固定长度），非 MLA 路径使用动态值，并同步用于 qo_indptr 构造、max_q_len 设置以及 _build_verify_unified_metadata 的调用参数。

4. forward_extend 同步修正 seqused_k：将 seqused_k=forward_batch.seq_lens + self.num_draft_tokens 改为 seqused_k=forward_batch.seq_lens + self.forward_metadata.max_q_len，使非 MLA 动态 verify 场景下的 KV 使用长度与实际 qo 长度一致。
5. 配套整理：提交历史包含一次 format 和一次恢复 import 排序（isort profile=black）。测试上未新增独立单元测试，依赖 PR CI 中 test_deepseek_v3_mtp.py（EAGLE + attention_backend='aiter'，MLA 路径）覆盖；非 MLA 分支的 test_deepseek_v32_mtp.py 被标记为 nightly，未在 PR CI 运行。

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 init_forward_metadata_out_graph, init_forward_metadata, _apply_cuda_graph_metadata, forward_extend）：唯一的变更文件，集中实现 AITER 后端 target_verify 与 CUDA 图路径中 verify token 数的动态推导，并保留 MLA 例外。

关键符号：init_forward_metadata_out_graph, init_forward_metadata, _apply_cuda_graph_metadata, forward_extend

关键源码片段

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

唯一的变更文件，集中实现 AITER 后端 target_verify 与 CUDA 图路径中 verify token 数的动态推导，并保留 MLA 例外。

```
# 入口：CUDA 图捕获 / 回放前的元数据准备。
# 关键改动：verify token 数不再取固定的 self.num_draft_tokens，
# 而是从精确大小的回放输入推导（每请求 token 数一致时，
# input_ids.shape[0] // batch_size 即为每请求 verify token 数）。
def init_forward_metadata_out_graph(self, forward_batch: ForwardBatch, in_capture: bool = False):
    seq_lens_cpu = (
        forward_batch.seq_lens.cpu() if in_capture else forward_batch.seq_lens_cpu
    )
    verify_tokens_per_req = (
        forward_batch.input_ids.shape[0] // forward_batch.batch_size
        if forward_batch.forward_mode.is_target_verify()
        else None
    )
    self._apply_cuda_graph_metadata(
        bs=forward_batch.batch_size,
        req_pool_indices=forward_batch.req_pool_indices,
        seq_lens=forward_batch.seq_lens,
        seq_lens_sum=None if in_capture else forward_batch.seq_lens_sum,
        encoder_lens=forward_batch.encoder_lens,
        forward_mode=forward_batch.forward_mode,
        spec_info=forward_batch.spec_info,
        seq_lens_cpu=seq_lens_cpu,
```

```
            verify_tokens_per_req=verify_tokens_per_req,
        )


    def _apply_cuda_graph_metadata(self, ..., verify_tokens_per_req: Optional[int]):
        # ... 前置分支省略（decode/idle 路径）...
        elif forward_mode.is_target_verify():
            bs = len(req_pool_indices)
            assert verify_tokens_per_req is not None
            # MLA 内核要求固定 draft length，继续用 num_draft_tokens；
            # 非 MLA 统一路径按本批输入动态推导，兼容 SBD 等场景。
            tokens_per_req = self.num_draft_tokens if self.use_mla else verify_tokens_per_req
            qo_indptr = self.qo_indptr[: bs + 1]
            qo_indptr[: bs + 1] = torch.arange(
                0,
                (1 + bs) * tokens_per_req,
                step=tokens_per_req,
                dtype=torch.int32,
                device=self.device,
            )
            if self.use_mla:
                kv_lens = seq_lens + self.num_draft_tokens # MLA 保持固定 draft 长度
            else:
                kv_lens = seq_lens
            # ... kv_indices / kv_last_page_len 构造省略 ...
            if self.use_mla:
                max_q_len = self.num_draft_tokens
            else:
                max_q_len = verify_tokens_per_req # 非 MLA 路径使用动态值
```

评论区精华

核心讨论围绕 MLA 路径的 draft token 长度限制展开：

- HaiShaw 在 review 中提出：@jinzhenfan with MLA, aiter has limitation on the range/length of draft_tokens, do you encounter issues so far? cc @kkHuang-amd，要求确认 AMD MLA 后端支持的 draft token 范围。
- 作者 jinzhenfan 回应：Updated the PR to leave MLA part unchanged.，即 MLA 路径继续使用固定 num_draft_tokens，仅非 MLA 路径改为动态推导。HaiShaw 随后感谢并批准。
- amd-bot 汇总 CI：本 PR 修改的代码路径被 test_deepseek_v3_mtp.py（mi325、EAGLE、aiter）覆盖并通过；多项 AMD job 因硬件 GPU-Hangs/OOM 失败，NVIDIA base-c 因无关 HF-offline 失败级联，NPU/XPU job 因其他后端失败，均与本 PR 无关；但非 MLA verify 分支缺少 PR CI 执行。
- MLA 后端 draft token 长度限制确认 (design): MLA 路径继续使用固定 num_draft_tokens（AITER MLA 内核要求），非 MLA 路径动态推导。
- PR CI 覆盖与失败归属判断 (testing): 本 PR 无可归因失败；但非 MLA 动态 verify 分支缺 PR CI 覆盖。

- base-c-test-4-gpu 基础设施问题确认 (other): 未在本 PR 内解决，属于独立 CI 问题。

风险与影响

- 风险：
  1. 非 MLA verify 分支测试缺口：test_deepseek_v32_mtp.py 被禁用并移到 nightly，PR CI 只覆盖 MLA 路径，动态推导逻辑（SBD 等场景）没有持续回归保障，后续改动可能悄悄破坏该路径。
  2. 输入长度推导前提：input_ids.shape[0] // batch_size 假设每个请求的 verify token 数一致，若 SBD 等场景下同一 batch 内各请求验证长度不均，整除结果可能失真，qo_indptr 会构造错误。需确认调用方保证 batch 内齐长。
  3. CUDA 图捕获与回放一致性：capture 时 seq_lens_sum=None，动态值依赖真实回放输入；若回放时输入形状与捕获时分配的最大缓冲不匹配，assert_buffer_fits 会拦截，但逻辑上仍需依赖调度器保证图内 shape 稳定。
  4. forward_extend 的 seqused_k 变更：改用 self.forward_metadata.max_q_len 后，max_q_len 在非 MLA 路径由动态值驱动，若该 metadata 在多层复用间被覆盖，可能导致 KV 长度计算不一致。
  5. 影响范围：仅 AMD 平台 + AITER 后端 + 非 MLA 的 spec decoding 用户受益，其他平台 / 后端无行为变化。 - 影响：用户侧：修复 AMD 平台上 AITER 注意力后端配合 EAGLE/MTP 等 spec decoding 在动态 verify token 数量（如 SBD）下的元数据错误，避免验证结果出错或崩溃；MLA 用户不受影响。系统侧：改动局限在 CUDA 图 target_verify 元数据构造，不改变图结构、不新增显存占用。团队侧：AMD CI（mi325）已通过覆盖该路径的测试；由于非 MLA 分支测试移入 nightly，团队需要留意 nightly 回归结果。整体影响面中等，但正确性收益明确。 - 风险标记：缺少非 MLA verify 分支的 PR CI 覆盖，依赖 batch 内每请求 verify token 齐长，CUDA 图捕获与回放长度一致性，仅 AMD AITER 后端生效

关联脉络

- PR #33090 [AMD][Fix] Restore aiter-padded MoE weight dims for serialized checkpoints: 同为 AMD 平台 AITER 相关修复，关注 AMD 后端正确性，可交叉参考 AMD CI 覆盖策略。
- PR #33127 [Fix] Bound FULL_MASK verify-mask reuse by the captured max_bs: 同为 speculative decoding 验证路径的正确性修复，涉及 CUDA 图捕获边界与回放一致性，思路可互相参考。
- PR #33087 [Fix] Repair verify mask test fixture: 同为 verify 路径测试夹具修复，说明 spec decode 验证路径近期有多个正确性修复点，存在共同演进脉络。