PR #26997 完整报告

sgl-project/sglang

Reland spec v2 tree drafting (eagle topk>1) with page_size==1 (#26866)

合并时间：2026-06-04 03:40

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26997

执行摘要

- 一句话：Spec v2 多路径草稿重上线 (Eagle topk>1)，要求 page_size==1
- 推荐动作：建议 SGLang 推测解码模块的维护者和使用者仔细阅读此 PR。重点关注 _finalize_accepted_tree_path 的压实策略、move_kv_cache 在 MLA 和 DSA 池中的分层实现，以及空闲批次注意力元数据的兼容性处理。这些设计决策展示了 SGLang 在支持复杂草稿拓扑时的架构思考。

功能与动机

原 PR #26866 在合并后因某些场景（如 DP attention、mamba 模型）出现行为异常而被回退（#26981）。此 PR 在修复了相关问题后重新引入该功能，使用户能够在使用 Eagle 推测解码时享受 topk>1 带来的多路径草稿性能收益。

实现拆解

1. 调整草稿扩展的 Token 数量：在 eagle_worker_v2.py 的 _draft_extend_for_decode 中，将每请求的 token 数从 speculative_num_steps + 1 改为 speculative_num_draft_tokens，使其与完整的树宽匹配，保证 DP MLP 同步填充的一致性。
2. 验证后路径压实：新增 _finalize_accepted_tree_path 方法，在验证后将被接受的树路径（KV 槽位、predict 标签、hidden_states）移动到每个请求块的连续前端，以满足下游链布局代码的假设。核心实现包括调用 move_accepted_tokens_to_target_kvcache 和 _compact_accepted_to_front。
3. 支持多种 KV 池的移动操作：在 memory_pool.py 中，为基础 KV 池添加零层池的短路返回；为 MLATokenToKVPool 新增 move_kv_cache 以移动压缩后的 MLA KV（latent + rope）；为 DSATokenToKVPool 重写 move_kv_cache 以同步移动 DSA 索引器缓存。
4. 处理空闲批次的注意力元数据：在 flashattention_backend.py 和 triton_backend.py 中，为 draft-extend 的空闲批次（用于 DP MLP 同步）构建简单的元数据，避免因缺少树索引导致的错误。在 forward_batch_info.py 中使 seq_lens_sum 在 gpu_only 批次中为 None 时条件跳过。
5. 配置逻辑与约束：在 speculative_hook.py 的 _handle_eagle_family 中，当 page_size>1 或检测到 mamba/linear-attn 模型时，强制回退到 Spec v1（因为 v2 的 topk>1 仅支持 page_size==1 且不支持 mamba 模型）。

6. 测试与验证：更新了两个测试文件（test_spec_eagle_topk.py 和 test_spec_eagle_stress.py），分别增加 TestEagle3Topk16SpecV2 和 TestEagle3Topk16V2Retract 测试用例，覆盖 topk=16 的 Spec v2 场景。

关键文件：

- python/sglang/srt/speculative/eagle_worker_v2.py（模块 草稿核心；类别 source；类型 core-logic；符号 _finalize_accepted_tree_path, _compact_accepted_to_front）：核心推测解码 worker，实现了验证后接受树路径的压实（compaction）和 KV 缓存移动，是 Spec v2 topk>1 的关键所在。
- python/sglang/srt/mem_cache/memory_pool.py（模块 内存池；类别 source；类型 core-logic；符号 move_kv_cache）：基础 KV 池添加零层池短路；MLA 和 DSA 池新增 move_kv_cache 以支持树路径压实后的 KV 缓存移动。
- python/sglang/srt/model_executor/forward_batch_info.py（模块 批次信息；类别 source；类型 data-contract）：调整 seq_lens_sum 填充逻辑，允许 gpu_only 批次为 None，避免同步开销；支持 d2h 同步优化。
- python/sglang/srt/arg_groups/speculative_hook.py（模块 配置钩子；类别 source；类型 core-logic）：配置 Spec v2 回退逻辑：当 page_size>1 或模型为 mamba/linear-attn 时强制使用 v1。
- python/sglang/srt/layers/attention/flashattention_backend.py（模块 注意力后端；类别 source；类型 core-logic）：为 draft-extend 的空闲批次构建简单的注意力元数据，避免因缺少树索引而崩溃。
- test/registered/spec/eagle/test_spec_eagle_topk.py（模块 草稿测试；类别 test；类型 test-coverage；符号 TestEagle3Topk16SpecV2）：新增 TestEagle3Topk16SpecV2 测试用例，覆盖 topk=16 的 Spec v2 场景。

关键符号：_finalize_accepted_tree_path, _compact_accepted_to_front, move_kv_cache, move_accepted_tokens_to_target_kvcache, _handle_eagle_family, init_forward_metadata, _pad_inputs_to_size

关键源码片段

python/sglang/srt/speculative/eagle_worker_v2.py

核心推测解码 worker，实现了验证后接受树路径的压实（compaction）和 KV 缓存移动，是 Spec v2 topk>1 的关键所在。

```
def _finalize_accepted_tree_path(
    self,
    batch: ScheduleBatch,
    accept_index: torch.Tensor,
    accept_lens: torch.Tensor,
    predict: torch.Tensor,
    logits_output,
    bs: int,
) -> torch.Tensor:
    '''Tree drafting (topk > 1): move the accepted path -- KV slots, predict,
```

```
    hidden_states -- to the contiguous front of each per-req block, which the
    downstream chain-layout code (draft-extend select_index, committed-KV reads)
    assumes. Returns compacted predict; mutates logits_output.hidden_states
    (moved only when present).'''
    # 先移动 KV 缓存：将接受的路径移到每个请求的前端
    self.move_accepted_tokens_to_target_kvcache(
        batch, accept_index, accept_lens - 1
    )
    # 压缩 predict 张量
    predict = self._compact_accepted_to_front(predict, accept_index, bs)
    # 如果存在 hidden_states，同样压缩
    if logits_output.hidden_states is not None:
        logits_output.hidden_states = self._compact_accepted_to_front(
            logits_output.hidden_states, accept_index, bs
        )
    return predict
```

python/sglang/srt/mem_cache/memory_pool.py

基础 KV 池添加零层池短路；MLA 和 DSA 池新增 move_kv_cache 以支持树路径压实后的 KV 缓存移动。

```
# MemoryPool.move_kv_cache 在开头加入零层池保护
def move_kv_cache(self, tgt_loc: torch.Tensor, src_loc: torch.Tensor):
    # 零层池（如 all-SWA 模型的 full 子池）没有缓冲区，直接返回
    if self.layer_num == 0:
        return
    # ... 原有校验和拷贝逻辑 ...

# MLATokenToKVPool.move_kv_cache：移动压缩后的 MLA KV
def move_kv_cache(self, tgt_loc: torch.Tensor, src_loc: torch.Tensor):
    size_limit = self.size + self.page_size
    maybe_detect_oob(tgt_loc, 0, size_limit, 'move_kv_cache tgt_loc')
    maybe_detect_oob(src_loc, 0, size_limit, 'move_kv_cache src_loc')
    if tgt_loc.numel() == 0:
        return
    tgt_loc_flat = tgt_loc.view(-1).long()
    src_loc_flat = src_loc.view(-1).long()
    for kv_cache in self.kv_buffer:
        kv_cache[tgt_loc_flat] = kv_cache[src_loc_flat]
```

评论区精华

本 PR 的讨论主要集中在 CI 结果上。PR 作者通过多次 /rerun-test 命令确保关键测试通过，如 test_constrained_decoding_spec_reasoning.py 和 test_qwen3_next_models_mtp.py。最终 Base 测试（Run #26898846215）通过，Extra 测试（Run #26898843678）初次失败后通过 rerun 成功。无其他实质性设计争论。

- CI 测试通过 (other): 最终 Base 和 Extra 测试均通过。

风险与影响

- 风险：
  1. 核心路径变更风险：Spec v2 verify 逻辑改动可能影响已有的 topk=1 场景，但已有测试覆盖基础功能。
  2. KV 缓存移动新操作：新增 move_kv_cache 在多个 KV 池中实现，可能引入性能抖动或未发现的边界条件（如零层池已通过早期返回处理）。
  3. 配置约束：page_size>1 和 mamba 模型用户会静默回退到 v1，可能导致性能期望落空。
  4. 注意力后端适配：空闲批次的元数据构建在 FlashAttention 和 Triton 后端分别实现，可能与其他优化（如 TBO、重叠调度）交互异常。 - 影响：用户影响：使用 Eagle 推测解码且满足 --speculative-eagle-topk >1 和 --page-size 1 的用户将自动获得多路径草稿加速，提升解码吞吐。其他配置保持不变。系统影响：SWA 和 MLA 模型的 KV 缓存移动增加 GPU 内存带宽消耗，但通过原生移动和分块拷贝优化。测试影响：新增的 stress 测试有助于防止回归。
- 风险标记：核心路径变更，新增 KV 缓存移动操作，配置约束 (page_size==1)，多注意力后端适配

关联脉络

- PR #26866 Spec v2 tree drafting (eagle topk>1) with page_size==1: 此 PR 是 #26866 的重新提交，修复了原 PR 的问题
- PR #26981 Revert spec v2 tree drafting: 此 PR 回退了 #26866，本 PR 旨在修复后重新上线