PR #25539 完整报告

sgl-project/sglang

[Spec] `FrozenKVMTP` fold assistant seed into captured draft graph

合并时间：2026-06-02 13:27

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25539

执行摘要

- 一句话：将 Frozen-KV MTP 辅助种子步骤融合到捕获的草稿 CUDA 图中
- 推荐动作：该 PR 值得精读，特别是了解如何将 *eager* forward 步骤融合到现有的 CUDA 图中以减少 kernel launch 开销。设计思路（将第一轮迭代纳入循环）可推广到其他类似场景。

功能与动机

Frozen-KV MTP 在捕获的循环草稿图之前运行一个单 token *eager* assistant seed forward。由于 *seed* 和 recurrent 迭代共享相同的 seq_lens - 1 rope 位置（相对于冻结的目标 KV），分离它们只会导致每次 *decode* 额外的 launch 和同步。将 *seed* 融入图中可减少开销。

实现拆解

1. 修改 _run_assistant_seed_step（frozen_kv_mtp_worker.py）：该方法不再执行模型 forward，而是将 *seed* 输入（bonus_tokens、hidden_states、占位的 topk_p/topk_index 零张量）存放到 batch.spec_info 中，供后续捕获的图使用。原本的模型 forward 逻辑（设置 forward mode、_set_positions、_init_frozen_kv_metadata 等）被移除，改为在 draft_forward 中作为迭代 0 执行。
2. 重构 draft_forward（frozen_kv_mtp_worker.py）：将 *seed* 迭代作为 recurrent loop 的第一次迭代（iter 0）集成到捕获的图中。不再需要单独处理 topk==1 的快捷路径（已删除）。
3. 扩展 FrozenKVMTPInputBuffers（frozen_kv_mtp_cuda_graph_runner.py）：新增 bonus_tokens 缓冲区，用于向图传递 *seed* token。相应地，在 *capture_one_batch_size* 和 replay 中复制 bonus_tokens，并移除对 topk_p/topk_index 的复制（现在由 *seed* iter 自身产生）。
4. 添加 profiling 支持（frozen_kv_mtp_cuda_graph_runner.py）：在 replay 方法的图执行周围添加 torch.profiler.record_function span，便于性能分析。
5. 更新单元测试 fixture（speculative_draft_runner.py）：_make_dense_frozen_kv_mtp_draft_inputs 和 _make_dense_frozen_kv_mtp_forward_batch 改为提供 bonus_tokens 而非 topk_p/topk_index，与新的输入约定对齐。

关键文件：

- python/sglang/srt/speculative/frozen_kv_mtp_worker.py（模块 推测解码；类别 source；类型 core-logic；符号 _run_assistant_seed_step, draft_forward, forward_batch_generation）：核心改动所在，修改了 assistant seed 步骤的实现方式和 draft forward 的迭代逻辑。
- python/sglang/srt/speculative/frozen_kv_mtp_cuda_graph_runner.py（模块 CUDA 图；类别 source；类型 core-logic；符号 FrozenKVMTPInputBuffers, capture_one_batch_size, replay）：CUDA 图运行器扩展了输入缓冲区，新增 bonus_tokens 字段，并调整了 replay 逻辑以支持 seed iter 的集成。
- python/sglang/test/kits/attention_unittest/runner_modes/speculative_draft_runner.py（模块 测试工具；类别 test；类型 test-coverage；符号 _make_dense_frozen_kv_mtp_draft_inputs, _make_dense_frozen_kv_mtp_forward_batch）：更新了测试 fixture 以匹配新的输入约定，确保单元测试覆盖新的逻辑路径。

关键符号：_run_assistant_seed_step, draft_forward, replay, capture_one_batch_size, _make_dense_frozen_kv_mtp_draft_inputs, _make_dense_frozen_kv_mtp_forward_batch

关键源码片段

python/sglang/srt/speculative/frozen_kv_mtp_worker.py

核心改动所在，修改了 assistant seed 步骤的实现方式和 draft forward 的迭代逻辑。

```
# python/sglang/srt/speculative/frozen_kv_mtp_worker.py

def _run_assistant_seed_step(
    self,
    batch: ScheduleBatch,
    last_token_ids: torch.Tensor,
    last_hidden_states: torch.Tensor,
    seq_lens_cpu: Optional[torch.Tensor] = None,
    mm_input_embeds: Optional[torch.Tensor] = None,
    draft_input: Optional[FrozenKVMTPDraftInput] = None,
) -> None:
    """Stash seed inputs on ``batch.spec_info``; the forward runs inside
    the captured draft graph (see ``draft_forward``'s seed iter)."""
    del seq_lens_cpu, mm_input_embeds, draft_input # unused after folding

    if batch.forward_mode.is_idle() or last_token_ids.numel() == 0:
        batch.spec_info = FrozenKVMTPDraftInput.create_idle_input(
            device=batch.device,
            hidden_size=self._recurrent_hidden_size,
            dtype=self.model_config.dtype,
            topk=self.topk,
            capture_hidden_mode=CaptureHiddenMode.LAST,
        )
        return

    stashed = FrozenKVMTPDraftInput()
```

```
    stashed.bonus_tokens = last_token_ids.to(torch.int64)
    stashed.hidden_states = last_hidden_states
    # Real-shaped zeros so inherited `filter_batch`/`merge_batch` can slice
    # them between iters; overwritten by the captured seed iter.
    bs = last_token_ids.shape[0]
    device = last_token_ids.device
    stashed.topk_p = torch.zeros(
        (bs, self.topk), device=device, dtype=torch.float32
    )
    stashed.topk_index = torch.zeros(
        (bs, self.topk), device=device, dtype=torch.int64
    )
    stashed.capture_hidden_mode = CaptureHiddenMode.LAST
    stashed.num_tokens_per_req = 1
    stashed.num_tokens_for_logprob_per_req = 1
    batch.spec_info = stashed
```

python/sglang/srt/speculative/frozen_kv_mtp_cuda_graph_runner.py

CUDA 图运行器扩展了输入缓冲区，新增 bonus_tokens 字段，并调整了 replay 逻辑以支持 seed iter 的集成。

```
# python/sglang/srt/speculative/frozen_kv_mtp_cuda_graph_runner.py

@dataclass
class FrozenKVMTPInputBuffers(ForwardInputBuffers):
    req_pool_indices: torch.Tensor
    positions: torch.Tensor
    mrope_positions: torch.Tensor
    seq_lens: torch.Tensor
    seq_lens_cpu: torch.Tensor
    topk_p: torch.Tensor
    topk_index: torch.Tensor
    hidden_states: torch.Tensor
    # Consumed by the captured seed iter; see `FrozenKVMTPWorker.draft_forward`.
    bonus_tokens: torch.Tensor
    global_num_tokens_gpu: Optional[torch.Tensor]
    global_num_tokens_for_logprob_gpu: Optional[torch.Tensor]


class FrozenKVMTPCudaGraphRunner:
    ...
    def __init__(self, frozen_kv_mtp_worker: FrozenKVMTPWorker):
        ...
        with torch.device(model_runner.device):
            ...
            bonus_tokens = torch.zeros((self.max_bs,), dtype=torch.int64)
            ...
        self.buffers = FrozenKVMTPInputBuffers(
            ...,
```

```
            bonus_tokens=bonus_tokens,
            ...,
        )

    def replay(self, forward_batch: ForwardBatch):
        ...
        # `topk_p`/`topk_index` are produced by the captured seed iter.
        buffers.bonus_tokens[:raw_bs].copy_(forward_batch.spec_info.bonus_tokens)
        ...
        # NVTX span: the graph bypasses `model_runner.forward`'s record_function.
        span_name = f"step[DRAFT_LOOP raw_bs={raw_bs} bs={bs} topk={self.topk}]"
        if torch.autograd._profiler_enabled():
            with torch.profiler.record_function(span_name):
                self._replay()
        else:
            self._replay()
```

评论区精华

gemini-code-assist[bot] 提出了两项优化建议：

- 清理未用参数（medium 优先级）：_run_assistant_seed_step 的形参 seq_lens_cpu、mm_input_embeds、draft_input 不再使用，建议从签名中移除并更新调用者，以保持 API 整洁。
- 使用 torch.empty 替代 torch.zeros（medium 优先级）：占位用的 topk_p 和 topk_index 会被 seed iter 覆写，建议用 torch.empty 避免不必要的 GPU 初始化开销。 以上建议未被采纳，PR 在主要 reviewer 批准后合并。
- 清理 _run_assistant_seed_step 未用参数 (design): 未采纳；PR 作者选择保留参数并使用 del 语句避免警告。
- 使用 torch.empty 替代 torch.zeros 避免初始化开销 (performance): 未采纳；PR 保持 torch.zeros 以确保 filter_batch/merge_batch 在覆写前有明确定义的值。

风险与影响

- 风险：
  1. seed iter 正确性风险：将原本独立的 eager forward 嵌入 CUDA 图可能引入边界错误，尤其是位置设置和 frozen KV metadata 初始化。PR 通过确保 draft_forward 中的 seed iter 设置正确的位置和 forward mode 来缓解。
  2. topk==1 路径回归：commit 历史显示曾恢复 topk==1 路径（restore code path for topk==1），说明该场景需要特别处理，最终版本保留了兼容性。
  3. 未采纳的代码清理：未移除未用参数可能导致后续维护困惑，但不会引发运行时错误。
  4. 测试覆盖：单元测试 fixture 已更新，但未新增针对 topk>1 或不同 batch size 的额外测试。 - 影响：用户侧：使用 Frozen-KV MTP（如 Gemma4 31B）的推理可获得性能提升（PR body 显示延迟从约 6.25ms 降至 5.45ms）。功能性无变化，精度保持（GSM8K 评分通过阈值）。系统侧：CUDA 图捕获和回放逻辑略作调整，但兼容现有调度

和 KV 缓存路径。团队侧：减少了代码分支（移除 topk==1 快捷路径），降低了维护成本。

- 风险标记：核心路径变更，性能优化需精度验证，未清理未用参数

关联脉络

- PR #26981 Revert "Support spec v2 tree drafting (eagle topk>1) with page_size==1": 都是 speculative-decoding 模块的修改，且涉及 topk 处理，间接相关。
- PR #23273 [NVIDIA] [GDN] Enable FlashInfer MTP verify on SM100+ (Blackwell): 同样是 MTP（Multi-Token Prediction）相关的性能优化，共享类似的技术栈。