PR #28096 完整报告

sgl-project/sglang

[Spec] Fix EagleDraftWorker draft-extend attn backend assignment

合并时间：2026-06-13 06:47

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28096

执行摘要

- 一句话：修复 EagleDraftWorker 草稿扩展注意力后端未赋值问题
- 推荐动作：值得合并。此 PR 修复了一个早期重构引入的回归，并改进了自适应状态切换的代码健壮性。建议在后续类似重构中保留跨模块的前向传播依赖检查。

功能与动机

测试 test_eagle_worker_v2_topk1_fastpath.py::TestEagleWorkerV2BackendFallback::test_uses_draft_extend_backend_when_available 在 main 分支上持续失败。PR #23862 重构了 EagleDraftWorker.init_attention_backend，但在拆分初始化过程时丢失了将 draft_runner.attn_backend 指向草稿扩展后端的赋值语句。草稿扩展前向传播会读取 draft_runner.attn_backend，没有赋值则会导致使用解码后端进行元数据初始化，从而引发错误。

实现拆解

1. 在 EagleDraftWorker.init_attention_backend 中添加条件赋值：在创建 draft_extend_attn_backend 后，增加判断 if self.draft_extend_attn_backend is not None:，若条件成立则将 self.draft_runner.attn_backend 指向该后端。这与 multi_layer_eagle_worker_v2 中相同的赋值逻辑保持一致。
2. 在 EAGLEWorkerV2.apply_runtime_state 中添加同步赋值：当自适应运行时状态切换步骤配置时，同步更新 dw.draft_runner.attn_backend，确保其指向激活步骤对应的草稿扩展后端。同样使用 if state.draft_extend_attn_backend is not None: 条件保护，以避免 None 覆盖。
3. 增强 EAGLEWorkerV2._override_worker_state 的备份 / 恢复机制：在上下文管理器中新增对 dw.draft_runner.attn_backend 的备份与恢复，确保自适应状态构建过程中 init_attention_backend 的赋值不会泄漏到活动工作器中。修复了 Codex 指出的 P1 风险。
4. 新增并完善测试用例：
   - 新增 _make_adaptive_worker 辅助方法，构建用于自适应状态测试的模拟工作器。
   - 新增 test_override_worker_state_restores_runner_attn_backend：验证 _override_worker_state 上下文退出后 draft_runner.attn_backend 恢复为初始值。
   - 新增 test_apply_runtime_state_updates_runner_attn_backend：验证 apply_runtime_state 正确更新 draft_runner.attn_backend 为状态中的草稿扩展后端。

关键文件：

- python/sglang/srt/speculative/eagle_worker_v2.py（模块 推测解码；类别 source；类型 core-logic；符号 init_attention_backend, apply_runtime_state, _override_worker_state）：核心源码文件，修复了 init_attention_backend 中缺失的赋值，并增强了 apply_runtime_state 和 _override_worker_state 以正确处理 draft_runner.attn_backend。
- test/registered/unit/spec/test_eagle_worker_v2_topk1_fastpath.py（模块 推测解码；类别 test；类型 test-coverage；符号 _make_adaptive_worker, test_override_worker_state_restores_runner_attn_backend, test_apply_runtime_state_updates_runner_attn_backend）：测试文件，新增了针对自适应状态切换的测试用例，验证修复的正确性并防止回归。

关键符号：init_attention_backend, apply_runtime_state, _override_worker_state, _make_adaptive_worker, test_override_worker_state_restores_runner_attn_backend, test_apply_runtime_state_updates_runner_attn_backend

关键源码片段

python/sglang/srt/speculative/eagle_worker_v2.py

核心源码文件，修复了 init_attention_backend 中缺失的赋值，并增强了 apply_runtime_state 和 _override_worker_state 以正确处理 draft_runner.attn_backend。

```
# 文件：python/sglang/srt/speculative/eagle_worker_v2.py

    def init_attention_backend(self):
        # 创建多步注意力后端和 cuda graph runner
        self.draft_extend_attn_backend = None

        draft_backend_factory = DraftBackendFactory(
            self.server_args, self.draft_runner, self.topk, self.speculative_num_steps,
        )

        # 初始化解码注意力后端
        self.draft_attn_backend = draft_backend_factory.create_decode_backend()

        # 初始化草稿扩展注意力后端（遵循 speculative_attention_mode 设置）
        self.draft_extend_attn_backend = draft_backend_factory.create_draft_extend_backend()

        self.draft_runner.draft_attn_backend = self.draft_attn_backend
        # 修复：当草稿扩展后端存在时，将 runner 的 attn_backend 指向它
        # 草稿扩展前向传播（eagle_info_v2.prepare_for_extend_to_fill_draft_kvcache）
        # 会读取 draft_runner.attn_backend 来初始化前向元数据。
        if self.draft_extend_attn_backend is not None:
            self.draft_runner.attn_backend = self.draft_extend_attn_backend
        self.tree_mask_mode = TreeMaskMode.FULL_MASK
```

test/registered/unit/spec/test_eagle_worker_v2_topk1_fastpath.py

测试文件，新增了针对自适应状态切换的测试用例，验证修复的正确性并防止回归。

```
# 文件：test/registered/unit/spec/test_eagle_worker_v2_topk1_fastpath.py

    def _make_adaptive_worker(self, runner_attn_backend):
        """创建一个 EAGLEWorkerV2，其草稿工作器的状态机字段填充了哨兵值，
        足以驱动 _override_worker_state / apply_runtime_state 而无需触及 GPU。"""
        draft_runner = SimpleNamespace(
            draft_attn_backend=object(),
            attn_backend=runner_attn_backend,
        )
        draft_worker = SimpleNamespace(
            speculative_num_steps=2,
            speculative_num_draft_tokens=3,
            draft_attn_backend=object(),
            draft_extend_attn_backend=object(), # 非 None，触发条件赋值
            cuda_graph_runner=object(),
            cuda_graph_runner_for_draft_extend=object(),
            draft_runner=draft_runner,
            _rebuild_topk1_chain_buffers=lambda: None,
        )
        worker = object.__new__(EAGLEWorkerV2)
        worker._draft_worker = draft_worker
        worker._target_worker = SimpleNamespace(
            model_runner=SimpleNamespace(
                attn_backend=object(), decode_cuda_graph_runner=object()
            )
        )
        worker.speculative_num_steps = 2
        worker.speculative_num_draft_tokens = 3
        worker.server_args = SimpleNamespace(
            speculative_num_steps=2,
            speculative_num_draft_tokens=3,
            cuda_graph_bs_decode=None,
            disable_cuda_graph=False,
        )
        return worker, draft_worker

    def test_override_worker_state_restores_runner_attn_backend(self):
        # build_adaptive_runtime_state 会在内部对每个候选步骤
        # 在 _override_worker_state 上下文中调用 init_attention_backend；
        # 上下文退出后必须恢复原始的 attn_backend。
        initial_backend = object()
        candidate_backend = object()
        worker, dw = self._make_adaptive_worker(initial_backend)

        with worker._override_worker_state(3, 4):
            dw.draft_runner.attn_backend = candidate_backend
            self.assertIs(dw.draft_runner.attn_backend, candidate_backend)

        self.assertIs(dw.draft_runner.attn_backend, initial_backend)
```

评论区精华

Codex 自动化审查发现 P1 问题：Codex 在第一次提交 d1c4541d 中注意到，当 --speculative-adaptive 有多个候选步骤时，build_adaptive_runtime_state() 会在 _override_worker_state() 上下文中调用 init_attention_backend()，但由于 _override_worker_state 之前未备份 dw.draft_runner.attn_backend，新赋值的后端会泄漏到活动工作器中，导致初始步骤仍使用上一个候选步骤的后端。

作者 ch-wan 的回应：作者确认了该问题，并在第二次提交 ae8d6f3a 中进行了修复：在 _override_worker_state 中添加了 draft_runner.attn_backend 的备份 / 恢复，并在 apply_runtime_state 中添加了同步赋值。

审核者 hnyls2002：最终批准了 PR。

- 自适应状态构建时后端赋值泄漏 (correctness): 作者在第二次提交中修复了该问题：在 _override_worker_state 中添加了 draft_runner.attn_backend 的备份 / 恢复，并在 apply_runtime_state 中添加了同步赋值。

风险与影响

- 风险：
  - 回归风险低：变更集中且条件保护 (if self.draft_extend_attn_backend is not None) 确保了仅在存在草稿扩展后端时才赋值，不影响现有行为。
  - 测试覆盖完善：新增了针对自适应状态切换的测试用例，覆盖了泄漏防护和状态同步。
  - 影响范围有限：仅影响使用了草稿扩展注意力后端的推测解码场景（如 EAGLE）。
- 影响：
  - 用户影响：修复了使用 EAGLE 推测解码且启用了草稿扩展后端时的潜在崩溃或性能退化问题。
  - 系统影响：对 draft_runner.attn_backend 的正确赋值确保了草稿扩展前向传播使用正确的注意力后端，避免了元数据初始化错误。
  - 团队影响：代码库维护成本降低，测试套件更健壮地验证了后端赋值契约。
  - 风险标记：回归修复，测试覆盖完善，核心路径变更

关联脉络

- PR #23862 Fix --mem-fraction-static not accounting for EAGLE draft model KV cache: 本 PR 修复了 PR #23862 引入的回归：该 PR 重构了 EagleDraftWorker.init_attention_backend，但丢失了将 draft_runner.attn_backend 指向草稿扩展后端的赋值语句。
- PR #28032 [Spec] Centralize dummy verify-input capture; add carries_draft_hidden_states: 同属推测解码模块的近期重构，改进了草稿验证输入捕获逻辑和隐藏状态传递。