PR #26651 完整报告

sgl-project/sglang

Fix DRAFT_EXTEND_V2 CG metadata: align test fixture and Triton with production seq_lens convention

合并时间：2026-05-29 17:46

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26651

执行摘要

- 一句话：对齐 seq_lens 约定，修复 FA3/FA4 CG 测试回归
- 推荐动作：该 PR 值得精读，特别是它展示了如何系统地定位一个从生产约定到测试夹具再到多个 backend 的连锁不一致问题。设计决策（在 Triton CG 中从 seq_lens 计算 kv_lens 而不是直接使用）和 clamp 技巧值得关注。

功能与动机

PR #26628 回退了 #26512，导致 FA3/FA4 的 DRAFT_EXTEND_V2 CUDA Graph 测试再次失败。根因是测试夹具设置 seq_lens = prefix_lens，而生产环境在调用 init_forward_metadata 前会将 seq_lens 增加 num_draft_tokens（即 prefix + extend）。此外，Triton 的 extend kernel 将 extend K/V 作为单独张量接收，因此缓存索引必须只基于前缀长度，而之前的 CG 路径直接使用 seq_lens 导致双倍计数。

实现拆解

步骤：

1. 修改测试夹具 _set_draft_extend_v2_prefix_lens（speculative_draft_extend_runner.py），将 batch.seq_lens 设为 prefix + extend，匹配生产环境。
2. 修改 Triton CG 捕获路径 init_forward_metadata_capture_cuda_graph（triton_backend.py），对 DRAFT_EXTEND_V2 模式计算 kv_lens = seq_lens - extend_seq_lens，并用于 kv_indptr/kv_indices 的构建。
3. 修改 Triton CG 重放路径 init_forward_metadata_replay_cuda_graph（triton_backend.py），进行相同的计算，并添加 torch.clamp(kv_lens, min=0) 以处理填充行可能产生的负值问题。
4. 移除 FA3/FA4 测试中 test_runner_mode_eagle_draft_extend_v2_cuda_graph_cases 的 skipTest，恢复测试执行。
5. 更新文档：从 KNOWN_FAILURES.md 移除 FA3/FA4 条目，在 dense/README.md 中将对应单元格标记为 ✓。

关键文件：

- python/sglang/srt/layers/attention/triton_backend.py（模块 注意力层；类别 source；类型 core-logic；符号 init_forward_metadata_capture_cuda_graph,

init_forward_metadata_replay_cuda_graph）：核心修复文件：修改了 CUDA Graph 捕获和重放路径中 DRAFT_EXTEND_V2 模式下的 kv_indptr/kv_indices 构建，关键变更量 +37/-4。

- python/sglang/test/kits/attention_unittest/runner_modes/speculative_draft_extend_runner.py（模块 推测解码测试；类别 test；类型 test-coverage；符号 _set_draft_extend_v2_prefix_lens）：测试夹具修复：修改 _set_draft_extend_v2_prefix_lens 使测试中的 seq_lens 约定与生产环境一致，关键变更量 +12/-7。
- test/registered/attention/unittests/dense/test_fa3.py（模块 密集注意力测试；类别 test；类型 test-coverage；符号 test_runner_mode_eagle_draft_extend_v2_cuda_graph_cases）：移除 DRAFT_EXTEND_V2 CG 测试的 skipTest，恢复测试执行，验证修复生效。
- test/registered/attention/unittests/dense/test_fa4.py（模块 密集注意力测试；类别 test；类型 test-coverage；符号 test_runner_mode_eagle_draft_extend_v2_cuda_graph_cases）：移除 DRAFT_EXTEND_V2 CG 测试的 skipTest，恢复测试执行，验证修复生效。
- test/registered/attention/unittests/dense/README.md（模块 文档；类别 docs；类型 documentation）：更新测试矩阵，标记 FA3/FA4 的 EAGLE-DE runner 和 DRAFT_EXTEND_V2 为已通过。
- test/registered/attention/unittests/KNOWN_FAILURES.md（模块 文档；类别 docs；类型 documentation）：从已知失败列表移除 FA3/FA4 DRAFT_EXTEND_V2 条目，表明问题已修复。

关键符号：init_forward_metadata_capture_cuda_graph, init_forward_metadata_replay_cuda_graph, _set_draft_extend_v2_prefix_lens, test_runner_mode_eagle_draft_extend_v2_cuda_graph_cases

关键源码片段

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

核心修复文件：修改了 CUDA Graph 捕获和重放路径中 DRAFT_EXTEND_V2 模式下的 kv_indptr/kv_indices 构建，关键变更量 +37/-4。

```
# DRAFT_EXTEND_V2 分支：Triton CG 重放路径（init_forward_metadata_replay_cuda_graph）
if forward_mode.is_draft_extend_v2():
    # 生产环境将 seq_lens 提升为 prefix + extend（eagle_info_v2.py 在调用本函数前
    # 已经 bump 了 seq_lens）。Triton extend 内核将 extend K/V 作为单独张量接收，
    # 因此 kv_indptr/kv_indices 必须只覆盖前缀部分。
    # 使用 clamp 应对填充行：当 GPU 图填充时，填充行的 seq_lens 保持为填充值（1），
    # 而 extend_seq_lens_tensor 则为 num_tokens_per_bs（ >1 ），减法可能产生负值。
    assert spec_info is not None and getattr(spec_info, "extend_seq_lens_tensor", None) is not
    None
    kv_lens = torch.clamp(
        seq_lens - spec_info.extend_seq_lens_tensor[:bs].to(torch.int32),
        min=0,
    ).to(torch.int32)
else:
```

```
    # DRAFT_EXTEND_V1: seq_lens 已经是前缀长度
    kv_lens = seq_lens
kv_indptr[1 : bs + 1] = torch.cumsum(kv_lens, dim=0)
kv_indices = self.cuda_graph_kv_indices
create_flashinfer_kv_indices_triton[(bs,)](
    self.req_to_token,
    req_pool_indices,
    kv_lens, # 之前传的是 seq_lens，现在传 kv_lens 避免双倍计数
    kv_indptr,
    None,
    kv_indices,
    self.req_to_token.stride(0),
)
```

python/sglang/test/kits/attention_unittest/runner_modes/speculative_draft_extend_runner.py

测试夹具修复：修改 _set_draft_extend_v2_prefix_lens 使测试中的 seq_lens 约定与生产环境一致，关键变更量 +12/-7。

```
def _set_draft_extend_v2_prefix_lens(batch, case, *, device: str):
    # 生产环境（eagle_info_v2.py）在调用 init_forward_metadata 前会将
    # seq_lens 提升为 prefix + extend。测试夹具必须对齐该约定。
    seq_lens = tuple(p + e for p, e in zip(case.prefix_lens, case.input_lens))
    batch.seq_lens = torch.tensor(seq_lens, dtype=torch.int32, device=device)
    batch.seq_lens_cpu = torch.tensor(seq_lens, dtype=torch.int32, device="cpu")
    batch.seq_lens_sum = sum(seq_lens)
```

评论区精华

- Codex 机器人在 Review 中指出：在 DRAFT_EXTEND_V2 重放路径中，当 raw_bs 小于捕获的 batch size 时，填充行的 seq_lens 保持为 Triton 填充值（1），而 extend_seq_lens_tensor 则指向填充了 num_tokens_per_bs（通常 >1）的缓冲区，导致 kv_lens 为负。作者在第三次 commit 中通过 torch.clamp(..., min=0) 解决了此问题，确保负值不会传入 cumsum 和 create_flashinfer_kv_indices_triton。
- 该问题的根本原因在于不同 attention backend 对 KV plumping 的拆分方式不同：FA3/FA4 使用统一的 cache read（cache_seqlens_int32 需要 prefix + extend），而 Triton 的 extend kernel 将前缀 KV 和后缀 K/V 分开传递。
- DRAFT_EXTEND_V2 重放路径中填充行负 KV 长度 (correctness): 作者在第三次 commit 中通过 torch.clamp(kv_lens, min=0) 解决，确保负值不会传入后续计算。

风险与影响

- 风险：主要风险在于修改了 CUDA Graph 捕获和重放的关键路径，任何计算错误都可能导致静默的注意力值错误（如之前修复前测试中提到的 ~82% 错误）。但通过对齐测试夹具和生产约定，并添加 clamp，此风险已显著降低。对 DRAFT_EXTEND_V1 和 V2 的区分通过 forward_mode.is_draft_extend_v2() 条件隔离，不影响非 speculative 或 V1 模式。生产环境中的 empirical check 显示 Triton CG 路径在实际 workload 中通常回退到 eager，因

此该修复主要是正确性和一致性改进，性能影响不大。

- 影响：影响范围：speculative decoding 中 DRAFT_EXTEND_V2 模式的 CUDA Graph 路径，具体涉及 FA3、FA4 和 Triton backend。其他 attention backend（FlashInfer、torch_native 等）不受影响。对团队：测试覆盖恢复，已知失败项减少；CI 中相关测试从 skip 变为执行。对用户：在生产中如果使用 Triton CG 且实际捕获条件匹配，缓存索引的正确性得到保证；否则无行为变化。
- 风险标记：核心路径变更，测试覆盖已补充，已处理负值边界条件，跨 backend 约定差异

关联脉络

- PR #26628 回退 #26512 因测试失败：本 PR 直接修复了 #26628 回退后重新暴露的测试失败，对齐了 test fixture 和生产约定。
- PR #26512 修复 DRAFT_EXTEND_V2 测试 fixture 与 FA backend 的不一致：本 PR 的另一种方案，但之前的修复在生产中导致双倍计数而回退；本 PR 采用正确的对齐方式。