PR #44053 完整报告

vllm-project/vllm

[Bugfix][V1][TurboQuant] Reserve workspace before CUDA graph capture

合并时间：2026-06-23 06:47

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44053

执行摘要

- 一句话：提前预留 TurboQuant workspace 防止 CUDA graph 锁定断言
- 推荐动作：此 PR 修复了一个影响 TurboQuant 用户的关键 bug，代码简洁且测试充分。建议阅读了解如何通过提前预留 workspace 解决 graph capture 后的动态分配问题。设计上保持了关注点分离（逻辑在 attention backend 内）。推荐合入。

功能与动机

TurboQuant 的 decode 和 continuation-prefill scratch buffer 是懒加载的，但 CUDA graph capture 会锁定 workspace。当长上下文请求在 graph capture 后触发更大的 buffer 分配时，会触发 'Workspace is locked' 断言，导致服务崩溃。本 PR 在 metadata builder 初始化时预留最大 workspace，从而在 graph capture 前完成分配，避免该断言。详见 PR body 及 issue #40798 / #41565。

实现拆解

实现分为以下步骤：

1. 在 TurboQuantMetadataBuilder.__init__ 末尾调用 self._reserve_workspace() - 将 workspace 预留逻辑放在 attention 后端内部，避免污染 GPUModelRunner。
2. 预留 decode workspace - _reserve_workspace 方法通过 current_workspace_manager().get_simultaneous() 预留 decode 所需最大形状的三个 tensor：注意力分数（(max_num_reqs, num_heads, max_num_splits, head_size+1) float32）、dequantized K（(max_num_reqs, num_heads, head_size) 模型 dtype）、softmax 统计量（(max_num_reqs, num_heads) float32）。这些形状基于 max_num_seqs、注意力头数、KV splits 和 head_size 计算。
3. 判断并预留 continuation-prefill workspace - 如果启用了 chunked prefill 且 max_num_batched_tokens 大于 _CONTINUATION_DECODE_THRESHOLD，则额外预留两个 FP16 buffer 用于 continuation-prefill 的 dequant 过程。buffer 形状为 (1, num_kv_heads, alloc_len, head_size)，其中 alloc_len 基于 max_model_len 对齐 block_size 向上取整。
4. 新增测试覆盖 - 在 tests/quantization/test_turboquant.py 中添加 TestTurboQuantWorkspaceReservation 类，通过 monkeypatch 模拟 current_workspace_manager 和 is_workspace_manager_initialized，验证两种场景下

get_simultaneous 的调用参数，确保预留逻辑正确。

关键文件：

- vllm/v1/attention/backends/turboquant_attn.py（模块 TurboQuant 后端；类别 source；类型 core-logic；符号 _reserve_workspace）：核心修复文件，在 TurboQuantMetadataBuilder 初始化中新增 _reserve_workspace 方法，预留 decode 和 continuation-prefill workspace，避免 CUDA graph capture 后动态分配导致断言。
- tests/quantization/test_turboquant.py（模块 TurboQuant；类别 test；类型 test-coverage；符号 TestTurboQuantWorkspaceReservation, _fake_vllm_config, _fake_kv_cache_spec, test_metadata_builder_reserves_decode_and_continuation_prefill_workspace）：新增 TestTurboQuantWorkspaceReservation 测试类，通过模拟 workspace manager 验证预留行为，确保修复逻辑的正确性。

关键符号：_reserve_workspace

关键源码片段

vllm/v1/attention/backends/turboquant_attn.py

核心修复文件，在 TurboQuantMetadataBuilder 初始化中新增 _reserve_workspace 方法，预留 decode 和 continuation-prefill workspace，避免 CUDA graph capture 后动态分配导致断言。

```
def _reserve_workspace(self) -> None:
    # 如果 workspace manager 尚未初始化，跳过预留
    if not is_workspace_manager_initialized():
        return

    # 从 config 中提取参数
    scheduler_config = self.vllm_config.scheduler_config
    model_config = self.vllm_config.model_config
    parallel_config = self.vllm_config.parallel_config

    max_num_reqs = scheduler_config.max_num_seqs
    num_heads = model_config.get_num_attention_heads(parallel_config)
    num_kv_heads = self.kv_cache_spec.num_kv_heads
    head_size = self.kv_cache_spec.head_size
    max_num_splits = self.vllm_config.attention_config.tq_max_kv_splits_for_cuda_graph

    # 预留 decode workspace: 三个 tensor，用于注意力分数、dequantized K、softmax 统计
    current_workspace_manager().get_simultaneous(
        ((max_num_reqs, num_heads, max_num_splits, head_size + 1), torch.float32),
        ((max_num_reqs, num_heads, head_size), model_config.dtype),
        ((max_num_reqs, num_heads), torch.float32),
    )

    # 判断是否需要预留 continuation-prefill workspace
    reserve_continuation_prefill = (
        scheduler_config.enable_chunked_prefill
```

```
        and scheduler_config.max_num_batched_tokens > _CONTINUATION_DECODE_
        THRESHOLD
    )
    if not reserve_continuation_prefill:
        return

    # 计算缓存 buffer 大小，对齐 block_size 向上取整
    max_cached_len = max(0, model_config.max_model_len - 1)
    alloc_len = round_up(max_cached_len, self.kv_cache_spec.block_size)
    cache_buf_shape = (1, num_kv_heads, alloc_len, head_size)
    # 预留两个 FP16 buffer，用于 continuation-prefill 的 dequant 过程
    current_workspace_manager().get_simultaneous(
        (cache_buf_shape, torch.float16),
        (cache_buf_shape, torch.float16),
    )
```

tests/quantization/test_turboquant.py

新增 TestTurboQuantWorkspaceReservation 测试类，通过模拟 workspace manager 验证预留行为，确保修复逻辑的正确性。

```
def test_metadata_builder_reserves_decode_and_continuation_prefill_workspace(
    self, monkeypatch
):
    from vllm.v1.attention.backends import turboquant_attn
    calls = []
    # 模拟 WorkspaceManager 记录 get_simultaneous 调用
    class FakeWorkspaceManager:
        def get_simultaneous(self, *shapes_and_dtypes):
            calls.append(shapes_and_dtypes)
    monkeypatch.setattr(turboquant_attn, "current_workspace_manager",
                        lambda: FakeWorkspaceManager())
    monkeypatch.setattr(turboquant_attn, "is_workspace_manager_initialized",
                        lambda: True)
    # 使用默认参数构造 metadata builder，应触发预留
    turboquant_attn.TurboQuantMetadataBuilder(
        kv_cache_spec=self._fake_kv_cache_spec(),
        layer_names=["layers.0.self_attn.attn"],
        vllm_config=self._fake_vllm_config(),
        device=torch.device("cuda"),
    )
    # 验证调用了两次 get_simultaneous
    assert calls == [
        (
            ((16, 8, 4, 129), torch.float32),
            ((16, 8, 128), torch.float16),
            ((16, 8), torch.float32),
        ),
        (
            ((1, 4, 8192, 128), torch.float16),
```

```
            ((1, 4, 8192, 128), torch.float16),
        ),
    ]
```

评论区精华

- MidasMining（用户反馈）：同样受 workspace lock-violation 影响（8× A4000），认为 PR 范围更小、更干净，愿意在 CI 通过后测试确认。
- alankessler（用户反馈）：在 Intel Arc B70 上确认修复了 workspace crash，但注意到 decode 在长上下文时性能严重下降（从 ~28 tok/s 降至 ~0.1 tok/s），猜测可能与 PR 无关，但值得后续关注。
- Workspace lock-violation in 8x A4000 (other): 正向反馈，确认修复方向，等 CI 通过后自行验证。
- Confirmed fix on Intel Arc B70 but decode performance degrades (performance): 确认 workspace 修复有效，但揭示了长上下文 decode 性能问题需要单独调查。

风险与影响

- 风险：
  - 覆盖范围有限：此修复仅针对 TurboQuant attention backend，其他后端（如 FlashAttention）仍可能存在类似问题。
  - 预留浪费：预留基于 max_model_len，当实际请求远小于最大长度时，workspace manager 可能记录不必要的预留，在极端情况下增加显存占用。
  - speculative-decoding 未修复：PR 声明不修复 spec-decode 场景，该场景下仍可能触发其他断言。
  - 阈值依赖：continuation-prefill 预留仅在 chunked prefill 启用且 max_num_batched_tokens > _CONTINUATION_DECODE_THRESHOLD 时生效，若配置不当可能未预留。
  - 测试覆盖：仅测试了预留调用参数，未验证实际 decoding 流程不触发断言，集成测试不足。
- 影响：
  - 用户：使用 TurboQuant KV cache dtype（如 turboquant_3bit_nc）且启用 CUDA graph 的用户将不再因长上下文请求崩溃，服务可靠性提升。
  - 系统：Benchmark 显示 serving 吞吐量提升约 5%（0.3186→0.3343 req/s），TTFT 降低约 7%（16809→15629 ms），TPOT 基本不变。
  - 团队：代码改动集中于 attention backend 内部，维护成本低。
  - 风险标记：speculative-decoding 兼容性，长上下文性能，workspace 预留可能浪费

关联脉络

- PR #40798 Superseded by this PR: 原始尝试修复相同 bug 的 PR，被本 PR 替代。

- PR #41565 Reported workspace lock issue: issue 中用户报告 workspace lock 问题，本 PR 修复。