PR #39226 完整报告

vllm-project/vllm

[Bugfix] Fix workspace resize leaking reserved GPU memory

合并时间：2026-04-24 04:50

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/39226

执行摘要

- 一句话：修复 MoE workspace 动态调整时的显存泄漏问题
- 推荐动作：值得精读。该 PR 解决了实际部署中遇到的显存泄漏问题，展示了在动态形状推理中管理 GPU 显存的典型权衡。建议关注 empty_cache 的性能影响，并评估后续是否在初始化阶段预留最坏情况 workspace 以避免运行时调整。

功能与动机

移除默认 chunking 后，profiling 阶段不再有最坏情况分配，MoE workspace 会在推理过程中动态扩容。当前实现虽先删除旧 buffer 再分配新 buffer，但 CUDA 缓存分配器可能不立即回收，导致 2x 显存驻留。PR body 指出：'sometimes we will have 2x the buffer resident in memory which may cause OOM or cause less space to be reserved for KV cache'。

实现拆解

1. 删除循环批量调整逻辑：原代码循环 for ubatch_id in range(self._num_ubatches) 对所有 ubatch 同时调整 workspace 大小。新代码改为只调整当前请求的 ubatch，其余 ubatch 延迟到下次调用时再调整，避免 DBO 下其他 ubatch 仍持有旧 tensor 视图导致泄漏。
2. 显式调用 empty_cache：在删除旧 tensor 后、分配新 tensor 前，插入 torch.accelerator.empty_cache() 调用，强制 CUDA 缓存分配器回收刚释放的显存，使新分配能复用它，防止峰值翻倍。
3. 调整调试日志：日志从报告所有 ubatch 的总内存改为仅报告当前 ubatch 的调整信息，并输出 ubatch_id。

涉及的唯一文件是 vllm/v1/worker/workspace.py，核心方法是 get_simultaneous 内的 workspace 分配逻辑。

关键文件：

- vllm/v1/worker/workspace.py（模块 Workspace 管理；类别 source；类型 core-logic）：核心变更文件，修改了 workspace 的动态调整逻辑，修复显存泄漏并优化 DBO 场景。

关键符号：未识别

关键源码片段

vllm/v1/worker/workspace.py

核心变更文件，修改了 workspace 的动态调整逻辑，修复显存泄漏并优化 DBO 场景。

```
# 仅调整当前请求的 ubatch，其他 ubatch 延迟调整以避免 DBO 下视图泄漏
self._current_workspaces[ubatch_id] = None
del current_workspace

# 显式释放已回收的显存段，使缓存分配器能重新使用 GPU 内存进行更大的分配
# 没有这一步，每次调整可能留下未释放的段，导致峰值内存升高
torch.accelerator.empty_cache()

# 分配新 workspace
self._current_workspaces[ubatch_id] = torch.empty(
    (required_bytes,), dtype=torch.uint8, device=self._device
)

# 更新局部引用，供后续返回
current_workspace = self._current_workspaces[ubatch_id]
```

评论区精华

> gemini-code-assist[bot]: 'Calling torch.cuda.empty_cache() inside a performance-critical path...can cause significant performance degradation due to the global synchronization of the CUDA allocator.' 建议文档化性能瓶颈或用更高效的内存管理策略。 LucasWilkinson (approver): 'Makes sense to me; I think we should add the worst case allocation back during init though if possible.' 倾向于在初始化时分配最坏情况大小以避免运行时调整。 czhu-cohere: 回复说明无 chunking 时最坏情况分配可能过大（如 256k tokens），实际上不可行。

已解决的问题：显存泄漏与峰值。未解决：empty_cache 的性能开销；初始化时分配最坏情况容量的长期方案。

- empty_cache 性能开销 (performance): 作者接受该 trade-off，但未采用替代方案。已在 PR body 中说明 empty_cache 耗时约 5ms（5GB buffer），认为对于大模型 10GB+ 的 buffer 来说，内存节省更重要。
- 初始化时预留最坏情况 workspace (design): 当前方案作为临时修复接受，后续可能重新考虑初始化分配策略。

风险与影响

- 风险：
  1. 性能风险：torch.accelerator.empty_cache() 会触发 CUDA 全同步，每次 resize 约耗时 ~5ms（5GB buffer），在高频 resize 场景下可能累积影响。当前仅当 workspace 需要扩容时才触发，频率较低，但仍需监控。
  2. 兼容性风险：使用了 torch.accelerator.empty_cache()（torch 2.0+ 的通用加速器 API），如果项目仍使用较老 PyTorch 版本可能不存在此 API。但 vLLM 已要求较新版本，风险低。

3. 回归风险：修改了 workspace 调整的循环逻辑，从同步调整所有 ubatch 改为仅调整当前 ubatch。依赖原行为（如预分配所有 ubatch）的代码可能受影响。但 DBO 分支已通过 ubatch_id 区分，且新逻辑通过延迟调整避免泄漏，经过验证。
4. 测试覆盖：无直接对应的单元测试，依赖集成测试和手动内存剖析验证。 - 影响：影响范围：使用 MoE 模型且使用 vLLM v1 engine 的用户，尤其是大 EP（专家并行）和 DBO（双批次重叠）场景。用户影响：降低 OOM 风险，允许更大的 KV cache 预留，可能提升吞吐。代价是 resize 时少量延迟（微秒到毫秒级）。系统影响：修改了 core workspace 管理逻辑，但仅限于 v1 路径，不影响 v0 或其他后端。团队影响：验证了动态 workspace 方案的可行性，为后续优化（如初始化时预留）提供基线。

- 风险标记：性能同步开销，无直接测试覆盖，动态路径变更

关联脉络

- PR #39402 [kv_offload+HMA][10/N]: Support load with multiple KV groups: 同样涉及 workspace 管理（v1 调度器），可能共享底层显存管理机制。
- PR #39167 [DP][Ray] Pin DP control bundle to same node as first GPU bundle: 同样修复分布式推理中的显存 / 资源管理问题，属于同一领域。